Conference on
Digital Forensics, Security and Law
Richmond, Virginia
June 10-12, 2013

Conference Chairs

David Biros
david.biros@okstate.edu
Oklahoma State University
Oklahoma
USA

Glenn Dardick
gdardick@dardick.net
Longwood University
Virginia
USA

Association of Digital Forensics, Security and Law

Copyright © 2013 ADFSL, the Association of Digital Forensics, Security and Law. Permission to make digital or printed copies of all or any part of this journal is granted without fee for personal or classroom use only and provided that such copies are not made or distributed for profit or commercial use. All copies must be accompanied by this copyright notice and a full citation. Permission from the ADFSL is required to make digital or printed copies of all or any part of this journal for-profit or commercial use. Permission requests should be sent to Dr. Glenn S. Dardick, Association of Digital Forensics, Security and Law, 1642 Horsepen Hills Road, Maidens, Virginia 23102 or emailed to office@adfsl.org.

ISSN 1931-7379
Thank You to Our Sponsors

LONGWOOD UNIVERSITY

OKLAHOMA STATE UNIVERSITY

(ISC)²

EDITH COWAN UNIVERSITY

VIRGINIA STATE UNIVERSITY

REGINALD F. LEWIS SCHOOL OF BUSINESS

JONES & BARTLETT LEARNING

Digital Forensic Investigator

Belkasoft

forensics made easier
Contents

Committee .............................................................................................................................................. 4
Schedule .................................................................................................................................................. 5
The Development of Computer Forensic Legal System in mainland China ........................................... 9
Fei Xu, Kam Pui Chow, Yonghao Mai*, Rongsheng Xu, and Jun Zhang*
Electronic Discovery and Proportionality in Minnesota: A Harbinger for the Federal Court System? ........................................................................................................................................... 15
Vicki Luoma* and Milt Luoma*
Cybercrime and Punishment: An Analysis of the Deontological and Utilitarian Functions of Punishment in the Information Age .................................................................................................. 17
Karim Jetha*
Invited Paper: Resolving the Cloud Forensics Conundrum .................................................................. 23
John Bagby*
KeyNote Speaker 1: Craig Valli ........................................................................................................... 25
Identifying Peer-to-Peer Traffic on Shared Wireless Networks .............................................................. 27
EJ Jung*, and Simon Piel
Workshop 1 and Panel: Examining the Forensic Expert (John Bagby) ................................................... 31
KeyNote Speaker 2: Jake Kouns ............................................................................................................ 33
Money Laundering Detection Framework to Link the Disparate and Evolving Schemes ................. 35
Murad Mehmet, Duminda Wijesekera, and Miguel Fuentes Buchholtz
System-Generated Digital Forensic Evidence in Graphic Design Applications .................................... 59
Enos Mabuto, and Hein Venter
Significance of Semantic Reconciliation in Digital Forensics .............................................................. 71
Nickson Menza Karie*
KeyNote Speaker 3: Daniel Ryan .......................................................................................................... 81
An Image Forensic Scheme with Robust and Fragile Watermarking for Business Documents ........ 83
Sai Ho Kwok*
First glance: An Introductory Analysis of Network Forensics of TOR ............................................ 107
Raymond Hansen*
A Thematic Review of User Compliance with Information Security Policies Literature ................ 103
David Sikolia*
An Ontology-Based forensic Analysis Tool ........................................................................................ 123
Mohammed Ali Alzaabi*
A Forensic Study of the Effectiveness of Selected Anti-Virus Products against SSDT
Hooking Rootkits .................................................................................................................................... 139
Sami Al-Shaheri*, Dale Lindskog, Pavol Zavarsky, and Ron Ruhl
Workshop 2: Forensic Readiness (Barbara Endicott-Popovsky) ......................................................... 163
* Author Presenting and/or Attending
Conference Committee

The 2013 ADFSL Conference on Digital Forensics, Security and Law is pleased to have the following as co-chairs of the conference, chairs of the conference committee, and administrators of the conference:

**David Biros**  
david.biros@okstate.edu  
Oklahoma State University  
Oklahoma, USA

**Glenn Dardick**  
gdardick@dardick.net  
Longwood University  
Virginia, USA

**Diane Barrett**  
dbarrett@uat.edu  
Chair - Program Committee  
University of Advanced Technology  
Arizona, USA

**Don Costello**  
dcostello2@unl.edu  
Chair - Sponsorship/Exhibits Committee  
University of Nebraska  
Nebraska, USA

**Monique Ferraro**  
monique@techforensicexperts.com  
Chair, Publicity Committee  
Connecticut, USA

**Diane Barrett**  
dbarrett@uat.edu  
Chair - Program Committee  
University of Advanced Technology  
Arizona, USA

**Glenn Dardick**  
gdardick@dardick.net  
Longwood University  
Virginia, USA

**Diane Barrett**  
dbarrett@uat.edu  
Chair - Program Committee  
University of Advanced Technology  
Arizona, USA

**Don Costello**  
dcostello2@unl.edu  
Chair - Sponsorship/Exhibits Committee  
University of Nebraska  
Nebraska, USA

**Monique Ferraro**  
monique@techforensicexperts.com  
Chair, Publicity Committee  
Connecticut, USA

**Dian Barrett**  
dbbarrett@uat.edu  
Chair - Program Committee  
University of Advanced Technology  
Arizona, USA

**Don Costello**  
dcostello2@unl.edu  
Chair - Sponsorship/Exhibits Committee  
University of Nebraska  
Nebraska, USA

**Monique Ferraro**  
monique@techforensicexperts.com  
Chair, Publicity Committee  
Connecticut, USA

The 2013 ADFSL Conference on Digital Forensics, Security and Law is pleased to have the following as members of the program committee:

**John Bagby**  
ibagby@ist.psu.edu  
The Pennsylvania State University  
Pennsylvania, USA

**Mohamed Chawki**  
chawki@cybercrime-fr.org  
University of Aix-Marseille III  
France

**Fred Cohen**  
f@all.net  
California Sciences Institute  
Livermore, CA, USA

**David Dampier**  
dampier@cse.mstate.edu  
Mississippi State University  
Mississippi, USA

**Gareth Davies**  
gddavies@glam.ac.uk  
University of Glamorgan UK

**Nick Vincent Flor**  
nickflor@unm.edu  
University of New Mexico  
New Mexico, USA

**Kevin Harris**  
Kevin.L.Harris@nscc.edu  
Nashville State Comm College  
Tennessee, USA

**Dr. Mike Johnstone**  
m.johnstone@ecu.edu.au  
Edith Cowan University Western Australia, Australia

**Andy Jones**  
andrew.jones@kustar.ac.ae  
Khalifa University UAE

**Gary Kessler**  
gck@garykessler.net  
Gary Kessler Associates  
Vermont, USA

**Stephen Larson**  
stephen.larson@sru.edu  
Slippery Rock University USA

**Ki Jung Lee**  
leekijung@glink.drexel.edu  
Drexel University, Pennsylvania, USA

**Jigang Liu**  
jigang.liu@metrostate.edu  
Metropolitan State University  
Minnesota, USA

**Milt Luoma**  
milt.luoma@metrostate.edu  
Metropolitan State University  
Minnesota, USA

**Marcus Rogers**  
rogersmk@purdue.edu  
Purdue University  
Indiana, USA

**John Riley**  
jriley@bloomu.edu  
Bloomburg University  
Pennsylvania, USA

**Brad Rubin**  
brubin@stthomas.edu  
University of St. Thomas  
Minnesota, USA

**Joseph J. Schwerha IV**  
scherwha@calu.edu  
Owner, TraceEvidence, LLC  
California U of Pennsylvania, USA

**Jill Slay**  
jill.slay@unisa.edu.au  
Polytechnic of Namibia  
Windhoek, Namibia

**Craig Valli**  
c.valli@ecu.edu.au  
Edith Cowan University Western Australia, Australia

**Eli Weintraub**  
eliew@afeka.ac.il  
Afeka Tel Aviv Academic College of Engineering, Israel

**Andrew Woodward**  
a.woodward@ecu.edu.au  
Edith Cowan University  
Security Research Western Australia, Australia

**Bob Zeidman**  
bob@zeidmanconsulting.com  
Zeidman Consulting  
California, USA
Schedule

Monday, June 10

- 08:15 AM CONTINENTAL BREAKFAST
- 08:15 AM On-site Registration
- 08:30 AM Introductions
  - Glenn S. Dardick, Director of the ADFSL
- 08:45 AM Welcome
  - Paul Barrett, Dean of the College of Business and Economics at Longwood University
- 09:00 AM Presentation 1
  - Fei Xu, Kam Pui Chow, Yonghao Mai, Rongsheng Xu, and Jun Zhang, China: The Development of Computer Forensic Legal System in mainland China
- 09:40 AM Presentation 2
  - Vicki Luoma and Milt Luoma, USA: Electronic Discovery and Proportionality in Minnesota: A Harbinger for the Federal Court System?
- 10:20 AM BREAK
- 10:35 AM Paper 1
  - Karim Jetha, USA: Cybercrime and Punishment: An Analysis of the Deontological and Utilitarian Functions of Punishment in the Information Age
- 11:15 AM Presentation 3
  - John Bagby, USA: Resolving the Cloud Forensics Conundrum
- 12:00 PM LUNCH (provided)
- 01:00 PM Keynote Speech I
  - Craig Valli, Director of ECU Security Research Institute, Congress Chair for the annual secau Security Congress, Editor of Journal of Network Forensics, and Co-Editor of the Journal of Information Warfare, USA
- 01:45 PM Presentation 4
  - EJ Jung, and Simon Piel, USA: Identifying Peer-to-Peer Traffic on Shared Wireless Networks
- 02:25 PM BREAK
- 02:40 PM Workshop 1 and Panel: Examining the Forensic Expert: an ADFSL Panel
  - Moderator: John Bagby, USA
Schedule

Tuesday, June 11

- 08:00 AM  CONTINENTAL BREAKFAST
- 08:00 AM  On-site Registration
- 08:30 AM  Announcements
- 08:45 AM  Keynote Speech II  
  ▪ Jake Kouns, Director of Cyber Security and Technology Risks Underwriting for Markel Corporation, USA: What’s not to trust?
- 09:30 AM  Paper 2  
  ▪ Murad Mehmet, Duminda Wijesekera, and Miguel Fuentes Buchholtz, USA and Chile: Money Laundering Detection Framework to Link the Disparate and Evolving Schemes
- 10:10 AM  BREAK
- 10:25 AM  Paper 3  
  ▪ Enos Mabuto, and Hein Venter, South Africa: System-Generated Digital Forensic Evidence in Graphic Design Applications
- 11:05 AM  Paper 4  
  ▪ Nickson Menza Karie, South Africa: Significance of Semantic Reconciliation in Digital Forensics
- 11:50 PM  LUNCH (provided)
- 12:50 PM  Keynote Speech III  
  ▪ Daniel Ryan, lawyer, businessman, and educator: Certification of Forensics Professionals, USA
- 01:35 PM  Presentation 5  
  ▪ Sai Ho Kwok, China: An Image Forensic Scheme with Robust and Fragile Watermarking for Business Documents
- 02:15 PM  BREAK
- 02:30 PM  Paper 5  
  ▪ Kenneth Johnson, USA: Journey into Windows 8 Recovery Artifacts
- 03:10 PM  Presentation 6  
  ▪ David Sikolia, USA: A Thematic Review of User Compliance with Information Security Policies Literature
- 03:50 PM  Paper 6  
  ▪ Raymond Hansen, USA: First glance: An Introductory Analysis of Network Forensics of TOR
Schedule

Wednesday, June 12

- 08:15 AM  CONTINENTAL BREAKFAST
- 08:30 AM  Announcements
- 08:45 AM  Paper 7  
  - Mohammed Ali Alzaabi, UAE: An Ontology-Based forensic Analysis Tool
- 09:25 AM  Paper 8  
  - Sami Al-Shaheri, Dale Lindskog, Pavol Zavarsky, and Ron Ruhl, Canada: A Forensic Study of the Effectiveness of Selected Anti-Virus Products against SSDT Hooking Rootkits
- 10:05 AM  BREAK
- 10:20 PM  Workshop 2  
  - Barbara Endicott-Popovsky, USA: Forensic Readiness
- 12:20 PM  Conference Close
THE DEVELOPMENT OF COMPUTER FORENSIC LEGAL SYSTEM IN CHINA
(Briefing Paper/Presentation)

Yonghao Mai
Hubei Police College, China

K. P. Chow
University of Hong Kong
Hong Kong

Rongsheng Xu
IHEP
Chinese Academy of Science, China

Gang Zhou
Wuhan Engineering science and Technology Institute

Fei Xu, IIE
Chinese Academy of Science, China

Jun Zhang
Hubei Police College, China

ABSTRACT

The computer forensic discipline was established around 2000 in China, which was further developed along with Chinese judicial appraisal system in 2005. The new criminal and civil procedure laws of the People’s Republic of China was enacted on 1 Jan 2013. The new laws specified electronic data is legal evidence and has great impact on the current practice on handling electronic evidence. This paper introduces the electronic data and electronic evidence examination procedure in mainland China, the general concept of computer forensic legal system, the management of computer judicial experts, the management of computer judicial expertise institutions.

Keywords: China legal system, computer forensic, judicial expert, judicial expertise institution.

1. ELECTRONIC DATA

1.1 Electronic Data

In March 14 2012, the National People's Congress approved the new criminal procedure law for the People's Republic of China Section 48: anything that can be used to prove the truth of an event is an evidence, which includes documents, witness statements, victims statements, suspect or defendant’s statement, forensics report; inquest, investigation report, identification, inspection experiment reports, audio visual materials, and electronic evidence. According to Section 52 of the same law, the administrative organ can collect evidence, documents, audio visual materials, etc. during the administrative law enforcement and criminal investigation. All these materials can be used as evidence in a criminal prosecution. On 31 August 2012, the 11th National People’s Congress Standing
Committee 28th meeting about “Amendment of Civil Procedure Law of the People’s Republic of China” confirmed that digital evidence is considered as evidence in any legal proceeding.

The new criminal and civil procedure laws of the People’s Republic of China was enacted on 1 Jan 2013. The new laws specified electronic data is legal evidence and has great impact on the current practice on handling electronic evidence.

In real application, digital evidence is usually presented through judicial appraisal. Traditional judicial appraisal includes forensic pathology, identification of criminalistics and audio visual information appraisal, which does not include electronic evidence forensics analysis. According to the Section 2 Paragraph 4 of the “Standing Committee of the National People's Congress on the forensic management issues” in People’s Republic of China, it states “according to the needs to be determined by the judicial administrative department of the State Council, together with the Supreme People's Court and Supreme People's Procuratorate, judicial experts and forensic institutions should register their corresponding practicing areas with proper authority”. Therefore, Judicial Administrative Department has established procedure to support registration of judicial experts and forensic institutions in the area “electronic evidence”.

Before the enacting of the new criminal and civil procedure laws on 1 Jan 2013, electronic data was not considered as legal evidence. On the other hand, judicial appraisal forensic reports are considered as legal evidence, which has been used to transform electronic data into legal evidence. The new criminal and civil procedure law will have great impact on legal cases that involving electronic data, both in theory and practice. We therefore believe research needs to be done on legal cases that involving electronic data when taking the new criminal and civil procedure laws in consideration.

1.2 Electronic Evidence Examination

On 1 July 2010, the implementation of “The provisions on reviewing issues of handling of evidence by the Supreme People's Court, Supreme People's Procuratorate, Ministry of Public Security, the Ministry of National Security, the Ministry of Justice on handling death penalty cases” recommended proper procedure when handling the following types of electronic evidence: electronic mails, electronic data interchange, Internet chat records, Internet blogs, mobile phone SMS, digital signature, domain names, etc. The procedure should include the following:

a. Document the storage media of the electronic evidence, such as hard disks, optical disks, and should be submitted together with the printed copy of the electronic evidence;

b. The information about the collection of electronic evidence, which includes time and place of collection, owner of the electronic evidence, the person that performs the collection and the collection steps, and the equipment used, etc;

c. Document the process on acquiring, storing, transferring, and presentation of the electronic evidence in a proper and legal way;

d. Proper signature and/or chop by the corresponding parties, which includes the person that collects the electronic evidence, the owner of the electronic evidence, and corresponding witnesses, etc;

e. When analyzing the electronic evidence, the judicial expert should confirm the integrity of the electronic evidence, and to ensure the electronic evidence has not been tampered, modified or artificially created;

f. The relationship of the electronic evidence with the case. When there in doubt on analyzing the electronic evidence, the judicial expert should perform additional integrity checking by considering related evidence of the case.
2. HOW TO BECOME A COMPUTER JUDICIAL EXPERT

In mainland China, qualified judicial experts should get the practicing qualification certificate from the corresponding authority. The practicing certificate is a legal proof that the person can engage in judicial appraisal. Judicial experts should meet the qualification requirements of the practicing certificate.

2.1 Qualification Requirements to Engage in Computer Forensic

Following are basic qualification requirements to practice computer forensic in China:

a. Has a senior professional technical title relating to computer forensic;

b. Has professional practicing qualifications relating to computer forensic, or has a university diploma relating to computer or information security or higher education, or has an intermediate title of relevant specialty, and has pertinent working experience of 5 or more years;

c. Has 10 or more years of working experience relating to computer forensic and has relatively professional technical skills.

Anyone who has a record of criminal offences, has been dismissed from a government position, or is a deregistered judicial expert, shall not engage in judicial appraisal.

2.2 Judicial Expertise Institution

A judicial expert should practice judicial appraisal in a judicial expertise institution. All judicial appraisal requests should be submitted through judicial expertise institutions and then conducted by individual judicial expert. A judicial expert should avoid potential conflict of interests in the following scenarios which may lead to bias opinions:

a. The judicial expert is the plaintiff/defendant or a close relative of the plaintiff/defendant;

b. The judicial expert or his/her close relative has an interest in the case;

c. The judicial expert has been served as a witness, advocate or defender in the case;

d. The judicial expert has an interest with the plaintiff/defendant in the case.

2.3 Judicial Expert’s Responsibility

A judicial expert’s responsibility system should be adopted for judicial appraisal. Judicial expert is a neutral third party to provide independent and unbiased opinions. When conducting computer forensic examination, there should be two or more qualified judicial experts. One judicial expert will conduct forensic analysis and be liable for the expert opinion in the report. He/she will affix his/her name or seal on the expert report. The other qualified judicial experts are responsible to validate the examination process and check the report.

Expert report contains expert opinions and/or facts. The court usually relies on the expert report when a verdict is made when the report covers specialized knowledge in a particular subject beyond that of the average person. On the other hand, the judicial appraisal will be affected by many factors. The accuracy of expert opinion will be more or less affected. For example, judicial experts will design and carry out different tests in each case which depends on their own background and experience. Therefore, it is possible that different judicial experts may draw different conclusions. If a judicial appraisal is jointly conducted by two or more judicial experts and different forensic conclusions were drawn, they should be noted in the expert report. Once an expert is identified to be the judicial expert of a case under legal procedures, he should perform the forensic analysis by himself, and make a
signature on the expert report. The expert may require to appear at the court as expert witness when he/she receives a court order or the prosecutor/defender request.

2.4 Rights of Judicial Experts

As a participant of legal proceedings, a judicial expert may enjoy the following rights:

a. Access the information and materials relating to the judicial appraisal, and interview the plaintiffs, defendants or witnesses relating to the judicial appraisal;

b. Require the instructing party of the judicial appraisal to freely provide materials and samples as required by the judicial appraisal;

c. Carry out the relevant inspections, examinations and simulated experiments as required by the judicial appraisal;

d. Refuse to accept any request of judicial appraisal that is illegal, fails to meet the requirements of judicial appraisal or is beyond the practice scope as registered;

e. Refuse to answer any question irrelevant to the judicial appraisal;

f. Preserve different views in the case of any inconsistency of expert opinions about the findings;

g. Receive continue professional development as required by the profession;

h. Obtain remunerations;

i. Enjoy any other right as prescribed by law and regulations.

2.5 Duties of Judicial Expert

A judicial expert possesses the following obligations:

a. Designated by his/her judicial expertise institution to conduct forensic examination according to the relevant provisions in a timely and independent manner and to produce expert reports;

b. Responsible for the expert opinions as produced;

c. Avoid cases that may have conflict of interests;

d. Preserve the materials, samples and other relevant information of the judicial appraisal in a proper manner;

e. Observe confidentiality when handling information and materials of cases, which includes state secrets, commercial secrets as well as individual privacy;

f. Appear in a court as expert witness and answering any query relating to the judicial appraisal;

g. Subject to audit and inspect by authority of justice;

h. Attend proper training and continue professional development of judicial appraisal;

i. Perform any other obligation as prescribed by law or regulation.

3. HOW TO BUILD A JUDICIAL EXPERTISE INSTITUTION

Judicial expertise institution is an institution where a legal person or any other organization can apply for practicing in judicial appraisal. It must satisfy the following conditions with respect to the current law in China:

a. Have its own name and premises;

b. Have a capital of at least 0.2 million RMB and up to 1 million RMB;
c. Have clear defined practicing scope;

d. Have instruments and equipment that can meet the needs of judicial appraisal within its practicing scope;

e. Have a testing lab which has passed measurement certification or laboratory accreditation and can meet the needs of judicial appraisal within its practicing scope;

f. Have 3 or more judicial experts for each judicial appraisal practicing scope.

Where a legal person or any other organization applies for practicing in judicial appraisal, the administrative authority of justice shall not accept it and shall produce a Decision on Rejection under the following circumstances:

a. Where the legal person representative or the person in charge of judicial expertise institution has been subject to criminal liabilities or has been dismissed from a public body or government department;

b. Under any other circumstance as prescribed by law or regulation.

The administrative authority of justice that has decided to accept an application shall produce a Decision on Acceptance and shall conclude the relevant examination according to the statutory time limit as well as statutory procedures. The administrative authority of justice shall organize experts to carry out an examination on the equipment and the testing laboratory as required for the applicant to engage in the designated judicial appraisal practicing scope. The time for the examination shall not be calculated into the time limit of the examination.

As to any institution that meets the relevant requirements upon examination, the administrative authority of justice at the provincial level shall make a decision on approving the registration and issuing a Judicial Appraisal License. For any institution that fails to meet the relevant requirements upon examination, the administrative authority of justice at the provincial level shall make a decision on disapproving the registration and inform the relevant applicant in written form with explanation.

The Judicial Appraisal License is the practice certificate of a judicial expertise institution. The judicial expertise institution shall carry out its activities of judicial appraisal according to law upon the strength of the Decision on Approving the Registration as issued by the administrative authority of justice at the provincial level as well as the Judicial Appraisal License. The valid term of the Judicial Appraisal License is 5 years.

The administrative authority of justice shall offer guidance on procedures, which includes administration and examination procedures, for judicial expertise institutions as well as their activities of judicial appraisal. The professional body of judicial appraisal shall establish professional practice of judicial appraisal according to law.

Judicial expertise institution established by an investigating authority for the purpose of investigation should not accept judicial appraisal request from the general public. Moreover, the people’s court or the administrative authority of justice cannot have its own judicial expertise institution. There is no hierarchical relationship among the judicial expertise institutions. A judicial expertise institution is not subject to any geographical restriction when accepting judicial appraisal requests.

4. CONCLUSION

With the development of judicial appraisal system in China, the development of computer forensic is improved significantly, which includes standardization and legalization. Computer forensic teams and institutions have achieved remarkable success in the past few years. Also, the credibility and functionality of judicial appraisal has become more and more important. Meanwhile, some
requirements about judicial appraisal system in mainland China were published by the Ministry of Justice, as well as the National People's Congress and Chinese People's Political Consultative Conference. Therefore, in order to move forward and catch up with the rest of the world, more efforts should be put into the computer forensic area in mainland China, which include research, development and professional practice.

REFERENCES


ELECTRONIC DISCOVERY AND PROPORTIONALITY IN MINNESOTA: A HARBINGER FOR THE FEDERAL COURT SYSTEM?
(Briefing Paper/Presentation)

Vicki Luoma
Minnesota State University, Mankato
Mankato, MN 56001
USA
vicki.luoma@mnmsu.edu

Milt Luoma
Metropolitan State University
Saint Paul, MN 55106
USA
Milt.Luoma@metrostate.edu

ABSTRACT

Courts have begun to recognize there are limits to what is reasonable for the scope of electronic discovery in litigation. Traditionally, the costs of producing information in response to discovery requests is borne by the party producing the information unless the courts find that such costs are overly burdensome or are otherwise unreasonable for the producing party. In those situations, courts initially resorted to cost-shifting approaches by placing the cost burden on the requesting party. While courts may still impose cost-shifting on a case-by-case basis, the concept of proportionality is taking hold. Proportionality refers to the idea that courts should take into account the value of the lawsuit when considering what is an appropriate level of electronic discovery. It makes little sense to permit open-ended and the sky-is-the-limit discovery when the actual cost of the discovery approaches a significant proportion of what is actually at stake in the litigation. The Minnesota Supreme Court has incorporated the concept into a revision of the Minnesota Rules of Civil Procedure. Can such a common sense approach to electronic discovery be far behind for the Federal Rules of Civil Procedure?
CYBERCRIME AND PUNISHMENT: AN ANALYSIS OF THE DEONTOLOGICAL AND UTILITARIAN FUNCTIONS OF PUNISHMENT IN THE INFORMATION AGE

Karim Jetha
University of Georgia
313 Brooks Hall
310 Herty Drive
Athens, GA 30602
kjetha@uga.edu

ABSTRACT
This conceptual piece analyzes the role of criminal punishment and the nature of cyber crime to investigate whether the current punishment schemes are appropriate given the deontological and utilitarian goals of punishment: retribution, deterrence, incapacitation, and rehabilitation. The research has implications for policymaking in cybercriminal law.

Keywords: cybercrime, criminal law, punishment, retribution, deterrence, information economics

1. INTRODUCTION

Academics in business, law, and the humanities have studied cybercrime with the ultimate intent of reducing the probability or magnitude of a cyber attack. This paper refers to classical legal scholarship in criminal law to analyze whether punishments for cyberlaw—as applied by the U.S. Attorney’s Office—are appropriate given the deontological and utilitarian frameworks that underlie criminal law. Following this introduction, the paper is organized into three segments. First, I discuss the four functions of criminal punishment. Then, I describe the ways in which differs from “real world” crime with respect to the constraints that govern online behavior. I conclude with an example of the ways that “real world” laws are applied to cybercrime and a plan to conduct future research in this area.

The threshold question to any inquiry of criminal law should question why our society has created criminal law and distinguished it from civil law. After all, parties that have suffered losses or harms can use the tort system—a facet of civil law—to recover damages from tortfeasors. In addition to the “traditional” tort of negligence, common law has developed intentional torts that allow aggrieved parties to recover in cases in which a tortfeasor has interfered with their rights of property. These torts cover actions such as trespass to property and conversion, the civil analogue to criminal theft. Additionally, in cases involving particularly malicious or willful and wanton conduct, plaintiffs may be entitled to punitive damages—additional money damages intended to deter and reform future defendants. These types of actions can be used to compensate victims for losses and are particularly useful in cases of economic crimes. In fact, some scholars have argued that civil law may even be more effective at addressing online malfeasance (Virgo, 2012).

What, then, is the distinction between criminal law and civil law? In a seminal essay on the subject published in 1958, Henry Hart noted that there is actually very little that distinguishes the two general systems of law. For example, he argues, one cannot distinguish crimes on the basis of civil wrongs on the basis of general social interest because society has a significant interest in due fulfillment of contracts and “most of the other stuff of civil litigation” (Hart, 1958). Additionally, the distinction cannot lie in the fact that public officials enforce criminal laws because public officials may also bring many kinds of civil enforcement actions. As a result, he concludes somewhat tautologically that anything which is called a crime is a crime and that the distinction is the result of a judgment of
community condemnation (Hart, 1958). If an act, then, is so morally delinquent as to be labeled a crime by a legislature, conviction in a court of law carries the “unpleasant physical consequence” of a punishment (Hart, 1958).

2. THE FUNCTIONS SERVED BY CRIMINAL PUNISHMENT

Classical scholarship in criminal law has identified four main goals of punishment: retribution, deterrence, incapacitation, and rehabilitation (Bonnie et al., 2004). These goals fit into two ethical frameworks: deontological and utilitarian. Table 1 illustrates the way the goals fit into the two ethical frameworks. The deontological framework, as the name suggests, suggests that a duty exists upon society to punish a criminal individual. Punishment under this framework, then, is not justified by any positive social consequences that may result from the act of punishment but rather because of the inherent fairness of the punishment itself (Bonnie et al., 2004). Retribution is the primary punishment goal under a deontological ethical framework. The utilitarian framework, on the other hand, is primarily concerned with the use of punishment to further social goals—particularly to deter and minimize future behavior (Bonnie et al., 2004). Deterrence, incapacitation, and rehabilitation fit under the utilitarian framework. Neither the two ethical frameworks nor the four goals within them operate in a vacuum; the criminal justice system relies on a mixture of these principles and constantly changes the importance it attaches to any one (Alschuler, 2003). The remainder of this section examines the four goals in greater detail.

The retribution goal of punishment is predicated upon the assumption that individuals who commit crimes are moral agents with decision making capacity. Retributive justice is not necessarily motivated by revenge but rather by “holding actual wrongdoers… to the same high standards to which we hold ourselves” (Bonnie et al., 2004). As a result, criminal law has carved out exceptions for children and other individuals that lack the capacity to understand the consequences of their actions (e.g., the mentally “insane”). Punishing these individuals under a model of retributive justice makes little sense because they cannot be held to the “same high standards” that society expects of itself. Legal scholars have compellingly argued that, when carried to its natural extension, these exceptions to retributive justice also preclude us from punishing criminal defendants if, “at the time of [their] unlawful conduct, [their] mental or emotional processes or behavioral controls were impaired to such an extent that [they] cannot justly be held responsible for [their] act” (Bazelon, 1975). In a world in which retributive justice functioned in this way, instructions to juries would be amended to allow jurors to acquit in cases in which the crime was caused by physiological, psychological, environmental, cultural, educational, economic, and hereditary factors, rather than by the accused’s free choice (Bonnie et al., 2004). Although our current notion of retributive justice does not encourage these types of jury instructions to limit the blameworthiness of criminal defendants, it does carry an important limitation to the application of punishment: proportionality.

Because retribution is based on a deontological perspective of punishment, proportionality tempers the extent of the punishment with the degree of wrong-doing involved in the crime. Essentially, we are saying that some crimes do not deserve to be punished as harshly as others. As a result, the principle of proportionality serves to create distinctions between, for example, felonies and misdemeanors with the intention that particularly harsh penalties should be reserved for particularly serious offenses. Currently, proportionality is determined by legislatures and is fairly granular; for example, the Federal Sentencing Guidelines distribute crimes “according to their relative seriousness among 43 different offense levels” (Bonnie et al., 2004). Although the sentencing guidelines provide substantial guidance to prosecutors, they cannot be relied upon entirely to assess blameworthiness of criminal defendants; most sentencing schemes give judges some discretion to act when the facts of a particular case do not fit neatly into the recommendations offered by the guidelines. Graded proportionality has utilitarian benefits as well, particularly with respect to disincentivizing risky or escalated behaviors. This is known as the principle of marginal deterrence.
The goal of deterrence has a slightly different set of assumptions than the goal of retribution. Whereas retribution assumes that individuals have the capacity to make choices for themselves, deterrence assumes that individuals perform—prior to acting—a “hedonistic calculus of pain and pleasure” when selecting from courses of conduct (Bonnie et al., 2004). Deterrence also differs from retribution in that deterrence is forward-looking while retribution is backwards-looking. In order for a potential punishment to deter a crime, the punishment must outweigh the expected pleasure attributed by the crime, discounted by the belief that the criminal will get away with the crime. As a result, for deterrence to be effective at all, the potential criminal must believe that there is at least some risk that he or she will be caught and arrested for committing the crime. If such a belief does not exist, punishment cannot have any deterrent value.

Marginal deterrence, as mentioned above, is the utilitarian implication of proportional grading of sentences. In order for marginal deterrence to be effective, there must be a large gap between the punishments for the most serious crimes and the least serious crimes (Bonnie et al., 2004). For example, if kidnapping was, like murder, punishable by the death penalty, kidnappers would have no incentive to protect the lives of the individuals that they kidnap. Because, however, the death penalty is generally reserved for extremely serious crimes, individuals that commit “relatively minor” crimes have an incentive to ensure that their actions are contained within the ambit of their original intent.

Incapacitation and rehabilitation are similar in that they both attempt to eliminate the likelihood that an individual will continue to commit crimes. Incapacitation, quite simply, refers to the fact that during the period in which individual is incapacitated, his or her ability to commit crimes has been significantly diminished. As a practical matter, any crimes committed while incarcerated either must be committed using a non-incarcerated intermediary agent or must be committed within the physical confines of the prison. Rehabilitation becomes particularly relevant once the individual is no longer incapacitated [though it is often considered a process that begins during incarceration] and refers to a “humanitarian intervention that promises to cure offenders and return them to their law-abiding ways” (Bonnie et al., 2004). Although the process of rehabilitation can take forms that range from talk therapy to electroshock therapy (see e.g., Lutze, 1998), the ultimate goal is to enhance an individual’s “impulse control and life satisfaction” in order to reduce the likelihood that he or she will want to commit future criminal activity.

Given the theoretical underpinnings of criminal law as described above, the next section will describe what makes cybercrime different from traditional crime in order to make an assessment about whether or not cybercrime deserves special consideration with respect to punishment.
3. CRIME IN THE INFORMATION AGE

As society slowly shifted from an industrial economy to a knowledge economy, the nature of products changed, relying more heavily on information goods (see e.g., Shapiro and Varian, 1999). Intellectual property regimes that protected those goods evolved as scholars developed more theory to the distinctions between intellectual property and real property (see e.g., Maskus, 2000). Finally, and most recently, information goods have largely transformed from analog to digital in form (see e.g., Boyle, 2003). James Boyle, a professor of law studying the emergence of the public domain, illustrates the evolution of information goods over time:

Imagine a line. At one end sits a monk painstakingly transcribing Aristotle’s *Poetics*. In the middle lies the Gutenberg printing press. Three-quarters of the way along the line is a photocopying machine. At the far end lies the Internet and the online version of the human genome. At each stage, copying costs are lowered and goods become both less rival and less excludable. My MP3 files are available to anyone in the world running Napster. Songs can be found and copied with ease. The symbolic end of rivalry comes when I am playing the song in Chapel Hill, North Carolina at the very moment that you are both downloading and listening to it in Kazakhstan—now that’s non-rival.

The field of information economics evolved to tackle these changes, particularly with respect to analog and digital information goods. Scholars in the field found that information goods are easy to reproduce—meaning that they generally had high fixed costs and low marginal costs—while also having the unique property that one person’s use did not adversely affect another person’s ability to derive value from the good—non-rivalry (Shapiro and Varian, 1999). Scholars of intellectual property followed suit: noting that intellectual property may be non-excludable, they identified tradeoffs of static and dynamic efficiency and helped develop policy to maximize overall efficiency. These efficiency gains are most prominent in the structure of patent law, which promotes dynamic efficiency by incentivizing new investments for information due to the monopoly period while also promoting static efficiency by granting wide access to patented material following the monopoly period (Maskus, 2000). The new scholarship indicated the extent to which the relationships among individuals and between individuals and their property had been fundamentally changed.

As society evolved along the Boyle scale, criminal legislation has remained relatively stagnant. Prior theoretical work on cybercrime suggests that cybercrime can be divided into two distinct types. The first and more common type is “nothing more than the migration of real-world crimes to cyberspace” and that when computer technology is used in the commission of a crime, that crime is considered to be “cybercrime” (Brenner and Clarke, 2005). The second type consists of specifically-defined cyber-offenses such as hacking or distributing malicious software. These crimes may, as a technical matter, also be placed in the “migration” category but are often distinct enough to justify the dichotomy. It is the first type—migration crimes—that indicate the extent to which legislation establishing cybercrime as a distinct field has lagged behind changes in technology. For example, in hacking cases, defendants are often charged with wire fraud using a statute that has been in place since the late 1800s (see e.g., United States Attorney’s Office 2011).

In spite of some degree of facial similarity—particularly when considering “migration crimes”—cybercrime differs from so-called “real world” crime along four dimensions: proximity, scale, physical constraints, and patterns (Brenner and Clarke, 2005). Table 2 summarizes these dimensions. Although there are notable exceptions within each of the categories, “real world” crimes generally require closer proximity between the perpetrator and victim, require a one-to-one level of focus (i.e., breaking and entering into one house at a time and being forced to focus on committing the crime at hand), are subject to physical requirements (i.e., physically breaking and entering a house, leaving behind trace evidence, being seen by passersby), and pattern recognition due in part to geographic limitations.
| Table 2 Theoretical Distinctions between Cybercrimes and “Real World” Crimes |
|-----------------------------|------------------------------------------------------------------------------------------------------------------|
| Proximity                   | Perpetrator and victim must be in reasonably close physical proximity                                                |
| Scale                       | Real-world crime tends to involve one perpetrator and one victim; perpetrator must focus on crime for its duration |
| Physical Constraint         | Physicality of real-world crime require additional preparation; physical trace evidence can be left at the scene of the crime |
| Pattern                     | Patterns emerge around demographics of both perpetrators and victims, as well as around the seriousness of crime     |

Susan Brenner and Leo Clarke, legal scholars prolific in the field of cybercrime write that, “The phenomenon of increasingly frequent and severe ‘cybercrime,’ however, does not require us to rethink how we defined ‘crime’ as much as it demonstrates that we need to rethink how we should enforce our criminal laws to deter and prevent cybercrime” (Brenner and Clarke, 2005). But if we grant two assumptions (a) that the nature of digital information goods is substantially dissimilar to non-information goods of the age at which some cybercrime laws were written and (b) that cybercrime is substantially dissimilar to real world crime, we can conclude that it may be time to rethink whether the migration of “real world” criminal law to “cybercrime” is an appropriate solution, given the function of punishment in criminal law. This is particularly true in cases of “migration crimes,” which are governed legislative histories that did not conceive of the statutes being used in these ways.

4. CRIME AND PUNISHMENT: FUTURE RESEARCH

The Aaron Swartz case presents a vivid illustration of these tensions. Aaron Swartz, an instrumental creator of the social news site Reddit, the RSS protocol, and the creative commons, was charged with two counts of wire fraud and eleven violations of the Computer Fraud and Abuse Act for downloading 4.8 million scholarly articles from JSTOR by stashing a laptop in an unlocked closet at MIT. If convicted, he faced up to 35 years in prison and fines of up to $1 million. Swartz committed suicide in early 2013, before he was convicted. We can use this case, which was widely reported in the popular media, to examine whether the punishments meet the stated goals and to determine whether they offend our collective sense of justice.

Given the reproducibility of digital information goods, the application of technology to crime can dramatically increase the scale of the crime with minimal “effort” on behalf of the perpetrator. To what extent should the number of documents downloaded affect the retributive punishment in this case? Is the proposed punishment in this case (approximately 35 years in prison and $1m in fines) proportional to the crime? For reference, in some jurisdictions manslaughter carries a 10-year sentence, bank robbery carries a 20 year sentence, and selling child pornography carries a 20 year sentence? Does the proposed punishment leave a wide enough gap for marginal deterrence? Only after answering questions such as these can we make judgments about the appropriateness of punishment schemes for cybercriminals in the United States.

Future research will attempt to answer these questions head-on by using psychometric research tools to determine if the types of punishments used for “migration” crimes are overly harsh or otherwise misapplied. This type of broad, psychometric sampling is particularly relevant for determining the ways in which we enact policy to enforce deontological goals (whereas we can examine recidivism statistics to shape utilitarian policymaking). Particularly as the legislative structure of cybercrime is
being established with the debate surrounding, for example, the CFAA and CISPA, this type of research can increase the credibility and the effectiveness of these and future laws by ensuring that the punishments for each of the crimes does not offend our sense of justice.

REFERENCES


AUTHOR BIOGRAPHY

Karim Jetha is an attorney and first-year doctoral student in the department of Management Information Systems at the University of Georgia. His research interests include information sharing behaviors and consumer social movements.
ON RESOLVING THE CLOUD FORENSICS CONUNDRUM
(Invited Paper/Presentation)

John Bagby
The Pennsylvania State University
Pennsylvania, USA
jbagby@ist.psu.edu

ABSTRACT
The “cloud” is idiom for an ill-defined set of online services. The cloud simultaneously offers IT savings and promises advances in functionality (e.g., ubiquity). However, the cloud also imposes poorly understood burdens on security and it may provoke injustice. Thus, the cloud presents a durable and seemingly irreconcilable conundrum for the digital forensics community. First, cloud proponents make efficiency promises for cloud services (SaaS, IaaS, PaaS). These translate well into the digital forensics domain. Indeed, the cloud may enable crowd sourcing of investigatory data vastly lowering costs of dispute resolution. For example, cloud-based litigation war rooms may reduce electronic discovery costs substantially. Furthermore, expansion of cloud-based evidence repositories could encourage settlements on litigation claims theretofore considered infeasible. Second, however, the current architecture of many cloud services arguably undermines justice. Proliferation of cloud services arguably undermines several due process presumptions made to support litigants’ needs in their case preparation. For example, the cloud increases opacity complicating forensics because file and directory structures are unstable and in constant flux. Indeed, cloud practices may compromise the forensic quality of evidence due to the (1) off-shoring of data and (2) practices that result in persistent file rotation with frequent metadata modification (e.g., activity logs). Many other nations that typically host cloud services have generally under-developed laws regulating privacy, security and litigation process rights. Therefore, these prevailing international practices erect barriers of cost, reliability, and access (lack of reciprocity) to accurate forensics. Indeed, all these conditions are inconsistent with U.S.-style litigation expectations. This paper attempts provisional resolution of this conundrum by recommending better deployment of existing standards from the ISO, NIST, GARP, GAAP, GAAS and other sources. Proposals are evaluated for the development, diffusion and implementation of new standards that would address the likely evolution in cloud architectures. This analysis proposes to restore traditional expectations for evidence transparency as data continues its migration into the cloud.
KEYNOTE SPEAKER

Craig Valli
Director of ECU Security Research Institute
Perth, Australia

BIOGRAPHY

Craig Valli is the Director of ECU Security Research Institute. He has over 20 years’ experience in the IT industry and consults to industry on network security and digital forensics issues.

His main consultancy focus is on securing networks and critical infrastructures, detection of network borne threats and forensic analysis of cyber security incidents. He is the Congress Chair for the annual secau Security Congress.

Craig is also the Editor of Journal of Network Forensics and Co-Editor of the Journal of Information Warfare. He has over 60 publications to his name on security related topics.
IDENTIFYING PEER-TO-PEER TRAFFIC ON SHARED WIRELESS NETWORKS
(Briefing Paper/Presentation)

Simon Piel (simonpiel.cs@gmail.com)
EJ Jung (ejung@cs.usfca.edu)
Department of Computer Science
University of San Francisco
2130 Fulton St
San Francisco, CA 94117
Phone: (415) 422-5422
Fax: (415) 422-5800

Keywords: peer-to-peer, contraband download, tracing, investigation tool

ABSTRACT
Tracing contraband downloads leads investigators to an IP address, and in turn Internet Service Providers (ISP) can provide a physical location using this IP address. However, most homes and offices share this IP address among many computers using wireless networks. In other words, there needs to be another investigation to find out which computer was responsible for contraband downloads. To make matters worse, these shared wireless networks often have vulnerabilities in access control such as using WEP or using weak passwords. In such cases, any computer in range, not necessarily at the given physical address, could be responsible. We use shallow packet analysis to identify which computer in the shared wireless network is participating in peer-to-peer downloads. Our approach does not require the packet content, thus does not require wiretapping warrant. We discuss characteristics of peer-to-peer traffic and show how we derive and use them. Our approach monitors the patterns in the duration, the frequency, the amount of information uploaded and downloaded, and the download speed in all connections. In particular, we monitor the traffic distribution over time for each connection and combine them based on their unencrypted header information to learn which connections are likely to stem from which application.

1. INTRODUCTION
Contraband, such as child pornography and copyright infringement materials, is a prime target of investigations. Tracing contraband on the Internet is challenging already, but even when the investigators succeed in tracing, it often leads into a shared wireless network. Identifying which computer in the shared wireless network is responsible for contraband transfer is not trivial. Even when the owner of the wireless network is cooperative, wireless network may have vulnerabilities such as using a weak protocol (e.g. WEP) or weak passwords, and the owner may not be aware of unauthorized users. If the offender is well versed in computer technology, the culprit might be encrypting the transfer between his computer and the source of illegal material. Unfortunately, decrypting the traffic requires excessive time and resources to the point that monitoring the content of the network becomes infeasible. Also, it requires wiretapping warrant to collect the content of the traffic. We therefore forego the approach of deep-packet analysis that requires on the contents of the packets, and instead focus on the lower layers of the network stack, which are necessarily unencrypted due to their role of directing packet flow through the network. Specifically, we focus on the headers of the transport and network layers, which contain important flow and routing information. This allows us to measure the network traffic characteristics, then to make an educated guess on the type of
applications running on each computer in the shared network. Since peer-to-peer application usage has a high correlation to the contraband transfer, or we focus on identifying computers participating in peer-to-peer file transfer only with header information.

2. GOALS

The goals of this research are:

1. Peer-to-Peer Application’s Network Characteristics
   Find and establish distinct characteristics of peer-to-peer file transfer based on the header information.

2. Tool Development
   Develop a software that analyzes the shallow packet information (encrypted traffic or traffic logs obtained without wiretapping warrant) and produces traffic patterns of each computer.

3. Guideline for identifying computers participating in peer-to-peer file transfer.
   Based on the results from step 1, provide guidelines in identifying computers in peer-to-peer file transfer.

3. PEER-TO-PEER TRAFFIC CHARACTERISTICS

Peer-to-peer networks have the advantage that they don’t rely on a single source of distribution. This benefits the user, as the file’s download speed is not limited by a single server’s upload speed, but rather by the collective upload speed of all the users “seeding” the file in question. Furthermore, since the files are distributed over many seeds, there is no single point of failure that could be exploited by law enforcement to restrict access to a given file. These traits of peer-to-peer file sharing make it a popular choice for spreading and sharing contraband over the Internet. Our tool measures and plots these traits for visual inspection. The plots are easy to understand without deep understanding of network technology. We discuss each characteristic with example plots below.

3.1 Probing to multiple IP addresses

Because peer-to-peer applications have to probe many users to establish what chunks of the file each seed possess and whether or not the user can download from that particular seed, we can use this probing communication to multiple unique IP-addresses as a reliable indicator for peer-to-peer traffic. The probing and connecting to multiple seeds results in a large number of unique connections that contain a varying number of packets exchanged for each connection. Note that this probing does not only happen in the beginning of file transfer, but periodically happens to optimize transfer speed. We can visualize this distinguishing behavior by plotting the number of connections over the size of the packet exchanges. This results in long-tail distribution shown in Figure 1. X-axis shows the unique connections, and Y-axis shows the number of packets transferred in each connection. The connections are sorted by the number of packets, and show the long-tail distribution of per-connection traffic.

3.2 Steady Traffic over Time

When plotting the number of packets (x-axis) versus time (y-axis) for a single port and direction, we detected that peer-to-peer software typically has a slow and steady increase in packets as shown in
Figure 1, whereas other network traffic, such as video streaming or website visits, can be distinguished by sharper edges on the graph as shown in Figure 2.

### 3.3 Inbound Versus Outbound Traffic

Peer-to-peer file distribution works best with a high number of seeds, thus, peer-to-peer software encourages its users to seed files in order to increase their download speed. As a result, comparing the inbound versus outbound traffic of a port will yield a similar shape on both graphs. While the height of both graphs will vary depending on the download/upload ratio the user has selected, the overall shape is very similar, as shown in Figures 3 and 4.

#### 3.4 More connections

We have found that a peer-to-peer application plot will have a more gradual fall compared to other network traffic. In other words, p2p traffic connects to many more IP addresses than other applications, and data transfer per connection varies. This gradual fall is shown in Figure 5.
4. IMPLEMENTATION

To have full control over our capture environment, we used Wireshark to tap into and capture our own test network’s traffic. We elected to build a platform-independent Java application to do the packet capture analysis and plotting. Since the number of ports getting captured is very high we choose to create an easy to maintain and easy to customize configuration file, that can be used to exclude specific ports (such as port 80 for web-traffic) or white-list ports that are commonly used by peer-to-peer software. In addition to showing the plots for the desired ports and saving them as images on disk, we also output the information in text format, so that future analysis can use these files to automate the evaluation process, or the investigator can modify the configuration file to include other suspicious ports for analysis. All our figures are produced by our own software, and show consistent characteristics of p2p file transfer.

4.1 Adjustment for changing ports

Most peer-to-peer software has the option of randomizing its port for each program execution. We group all traffic between a unique pair of IP addresses to detect p2p traffic on any ports.

4.2 Data Collection

We reorganize the data from the raw capture file (.pcap file in text format) by extracting each packet’s receivers and senders IP addresses from the network layer header as well as the sender’s and receiver’s ports from the transport layer header. Each packet’s data is then stored in data structures that can produce the plots to show the frequency and the amount of traffic of each connection.

5. CONCLUSION

The identification of peer-to-peer traffic through shallow packet analysis opens up a promising new way to detect computers participating contraband transfer in a shared wireless network. We have implemented a proof of concept and shown that our software provides reasonable results. The shallow packet analysis is far less intrusive than alternative deep-packet analysis methods and provides useful information even when the data transfer is encrypted. Our software is available at [http://www.cs.usfca.edu/~spiel/packeteer/](http://www.cs.usfca.edu/~spiel/packeteer/).
WORKSHOP 1 AND PANEL:
EXAMINING THE FORENSIC EXPERT

Moderator:
John Bagby
The Pennsylvania State University
Pennsylvania, USA
jbagby@ist.psu.edu

DESCRIPTION

In today's litigation climate, battles are often waged between experts throughout all phases of litigation. Forensic quality evidence is the primary deliverable from expert witnesses who are generally retained by litigants. This panel specifically explores the often daunting dynamics of cyberforensic expert testimony given in pre-trial depositions, as expert witness testimony at trial, and before various other tribunals. The panel will deploy role play to explore a range of issues confronting expert witnesses and special masters to illustrate how they approach negotiating engagement letters, performing expert forensic work, complying with forensic standards of professional conduct, establishing their own credibility on the witness stand, and “holding up” under intense cross-examination by opposing counsel. The panelists will play roles of judge, sponsoring legal counsel, opposing counsel and expert cyberforensic witnesses. The audience is invited to participate as mock jurors who can “debrief” during Q&A to confirm the impact of particular behaviors of cyberforensic experts on the witness stand.
KEYNOTE SPEAKER
WHAT'S NOT TO TRUST?

Jake Kouns
Director of Cyber Security and Technology Risks Underwriting
Markel Corporation

BIOGRAPHY

Jake Kouns is the Director of Cyber Security and Technology Risks Underwriting for Markel Corporation. In this unique role, Mr. Kouns is responsible for strategy and oversight of the Enterprise Information Security Program for the company as well as the management of Cyber Liability insurance products. In his role as product line leader, he has broad responsibility for all aspects of the products including the development of underwriting guidelines, pricing, risk analysis, claims oversight, training & marketing initiatives as well as risk management services for policy holders.

In addition, Jake is the co-founder, CEO, and CFO of the Open Security Foundation, a non-profit organization that oversees the operations of the Open Source Vulnerability Database (OSVDB.org), DataLossDB.org, Cloutage.org and SECore.info. All projects are independent and open Web-based databases that provide detailed and unbiased technical information on security vulnerabilities, cloud security, security conferences and data loss incidents world-wide. Mr. Kouns has presented at many well-known security conferences including RSA, DEF CON, CISO Executive Summit, EntNet IEEE GlobeCom, CanSecWest, SOURCE and SyScan. He is the co-author of the book Security in an IPv6 Environment, Francis and Taylor, 2009, Information Technology Risk Management in Enterprise Environments, Wiley, 2010 and The Chief Information Security Officer, IT Governance, 2011. He holds both a Bachelor of Business Administration and a Master of Business Administration with a concentration in Information Security from James Madison University. In addition, he holds a number of certifications including ISC2's CISSP, and ISACA's CISM, CISA and CGEIT.
MONEY LAUNDERING DETECTION FRAMEWORK TO LINK THE DISPARATE AND EVOLVING SCHEMES

Murad Mehmet
Duminda Wijesekera (dwijesek@gmu.edu)
Miguel Fuentes Buchholtz
Department of Computer Science
Volgenau School of Engineering
George Mason University
Fairfax, VA

Keywords: Anti Money Laundering, Social Network Analysis, Complex Event Processing

ABSTRACT

Money launderers hide traces of their transactions with the involvement of entities that participate in sophisticated schemes. Money laundering detection requires unraveling concealed connections among multiple but seemingly unrelated human money laundering networks, ties among actors of those schemes, and amounts of funds transferred among those entities. The link among small networks, either financial or social, is the primary factor that facilitates money laundering. Hence, the analysis of relations among money laundering networks is required to present the full structure of complex schemes. We propose a framework that uses sequence matching, case-based analysis, social network analysis, and complex event processing to detect money laundering. Our framework captures an ongoing single scheme as an event, and associations among such ongoing sequence of events to capture complex relationships among evolving money laundering schemes. The framework can detect associated multiple money laundering networks even in the absence of some evidence. We validated the accuracy of detecting evolving money laundering schemes using a multi-phases test methodology. Our test used data generated from real-life cases, and extrapolated to generate more data from real-life schemes generator that we implemented.

1. INTRODUCTION

Current Anti Money Laundering (AML) systems are designed to function based on the requirements of adopting organization. They vary from the multi-component and complex systems such as FINCEN (FAIS) to the specialized single-purpose systems used by banks to report Due Diligence and Suspicious Activity Reports (SAR). To capture increasingly complex money laundering schemes (MLS) call for integrating new techniques such as Social Network Analysis (SNA) (Wasserman et al., 1994), in addition to the already used rule based analysis and risk modeling. An efficient AML system must have many components, where some components are being purely deterministic and others being purely probabilistic. An example of a probabilistic component is the risk analysis, and SNA is an example of a deterministic component. Generally, deterministic models consider social aspects and statistical models consider financial aspects (Wasserman et al., 1994).

The FINCEN AI System (FAIS) (Senator et al., 1995, 1996, 1998) designed for internal use analyzes SARs filed by banks. The system combines offline SAR data analyzed by human experts to identify possible hidden linkage among transactions using link analysis techniques. However, FAIS (Senator et al., 1995, 1996, 1998) only links and evaluates the database (DB) of the reported suspicious transactions offline. KDPrevent (Jacobs et al., 2003; Kuns et al., 2004) by KDLabs, a commercial product/service utilized by banks in Switzerland collect customer, account and transaction information for offline analysis, combining data-mining techniques with expert legal knowledge of legal experts.
Two models of sequence matching and link analysis (Liu et al., 2008; Schwartz et al., 2008) are relevant to our research in detecting evolving patterns of sequence. Liu (Liu et al., 2008) proposes a sequence matching algorithm to discover suspicious transaction sequences, using transaction histories of an individual’s accounts and transaction information histories from a peer group. (Liu et al., 2008) focus only on the bank transactions, without covering other financial transactions such as stock market. Schwartz (Schwartz et al., 2008) proposes a model to find criminal networks using social network analysis, building upon Borgatti’s SNA-based key player approach (Schwartz et al., 2008). One drawback of Borgatti’s model is the failure to assign weights to actors and actor-actor relationships. Gunestas, et al.’s framework (Gunestas et al., 2008, 2010) is similar to ours, but with a narrower focuses on detecting Ponzi schemes.

The rest of the paper is organized as follows: Section 2 explains the Money Laundering Evolution Detection Framework (MLEDF) and Section 3 describe proposes a new ML detection algorithm. Section 4 evaluates the performance results of MLEDF using real-life cases. Section 5 describes related work and Section 6 concludes the paper.

2. MONEY LAUNDERING EVOLUTION DETECTION FRAMEWORK (MLEDF)

The framework is composed of four different phases. Each phase will communicate with the next phase, and the output generated from each phase is sent into the next phase. The phases and their function are explained below.

Transaction Data Collection: The transaction agents or data input collectors from Automated Clearing House such as (EPN, FEDWIRE, and CHIPS) will send in their data format. The different types of transaction data are: Banking, Stock market, Derivative market, Web Services, Trading, Electronic Money, and Money Brokering. Once the industry-specific transaction data is gathered, relevant information are extracted for analysis. For example, there are more than 20 fields in stock order forms and we use only time, sender, receiver, price, quantity, symbol, market, sellerOrderID, buyerOrderID, tradeID, and country. Also, we use transaction-independent data used in the analysis, such as the economic status of the country, sales trends of the stock, and the stock value during the day.

1. Data Processing: The data collected from different systems are used to create patterns of the well-known MLS (Mehmet et al., 2010). The MLS-related data that is extracted from the streaming events is filtered before submitting them into the detection algorithms. The extracted data associated with each MLS pattern assigned to a specific MLS type using the following components: (I) Business Rules: MLS business rules and red flags associated with each pattern, the rules associated with specific sector are used by the MLS detection algorithms to identify the MLS patterns. (II) MLS Template: Well-known MLS templates will be used during this phase. Currently, the templates have seven major pattern types with their different subtype combinations. This acts as a repository of known MLS. If a new form of MLS is discovered, then it will be added to this DB. (III) ML Economic Models: Three ML economic models (Mehmet et al., 2013) will be used to validate and increase the accuracy of the detection algorithms for well-known ML patterns. Those economic models determine if the evolution of MLS is within the accepted trend of the models.
2. **MLS Algorithms and MLS Network Detection:** There are six major heterogeneous algorithm modules (Smurfing, Trade, Stock, Derivative, EMoney, DirtyEFT). Each algorithm uses a different method to capture the network associated with the specific type of MLS. In real-time, the algorithms output, the discovered networks associated with the specific MLS patterns, each into a different database. Then, the discovered networks are reformatted and saved in a single database referred to as the “Network” Database. This process facilitates faster and efficient analysis of the links among MLS networks.

3. **Evolution Detection Analysis and Generating the Fraud Trail and Suspicious Trail:** Four separate algorithms are run to find the “Full-Trail” (Mehmet et al., 2013), “Missing-Trail” (Mehmet et al., 2013), and “Suspicious-Trail” (Mehmet et al., 2013) of MLS networks, and saved in separate databases. Full-Trail is a long series of MLS’s that span over many countries and involves many cycles of MLS. In essence, it is a concatenated sequence of related schemes (MLS) act in itself to transfer money from one MLS to the other until it reaches the final MLS, where we refer to the orchestrator (i.e. the money launderer) as the “EndBoss” in the final MLS. Any MLS or trail will have the originator “StartBoss” and the terminator “EndBoss”, in addition to the associates that maintain the MLS or trail. The “Associates” are the list of the people involved in the sequence of detected fraud. The “StartBoss” is the entity whom starts the MLS or trail. Missing-Trail is a short Full-Trail that does not exceed the depths of three related MLSs. We assume that the Missing-Trail is a premature Full-Trail with broken parts and missing links or evidence. A Suspicious-Trail is a combination of discovered Full-Trails and/or Missing-Trails, it will be constructed using algorithms that incorporate SNA and numerical analysis techniques. The module “Detection Analysis” (Mehmet et al., 2013) determines the evolution of the “Full-Trail”’s such as the change to the number of involved associates, the changes to the cost of laundering, and changes to the laundering locations.

### 2.1 MLS Pattern Detection Modules in MLEDF

MLEDF process is fed by data from many types of transaction data, where each feed is from a particular market or a finance industry. The main detection module is divided into sub-modules,
where each sub-module detects money laundering patterns relevant to that specific market. This is because the data set of each market and industry is different than others and the money laundering techniques differ between them. MLEDF uses a core set of business rules to detect the evolution of MLS reported annually by FATF, with six detection patterns for each finance sector that we chose to include in our experiment.

2.2 Gathering Transaction Data and Generation of the Transaction Evidence Data

A “Message” sent between two parties in the framework consists of the following components: (1) Common Mandatory Fields: Sender, Receiver, Time, Transaction ID, and a field that reflects the amount of funds transferred or price of the transaction. (2) Pattern Specific Mandatory Fields: A set of attributes pertinent to the transaction type. For example, the Smurfing transaction will have only the "EFT" field that reflects whether the banking transaction is an EFT or not. The stock transaction has more fields as in "Quantity, Symbol, Market, TradeID, Country, etc.". (3) Auxiliary Third Party Fields: The framework retrieves critical data from third party sources, used in conjunction with the transaction data. The auxiliary data includes information such as recent market stock and derivative data, current product market price, and country economic status. This data is used to compare the transaction price and product price with the nominal price.

The “Comprehensive Output” is the MLS pattern-specific output generated by the MLS detection algorithm. As seen in figure 2, the output produces comprehensive evidence for each detected pattern, and it is different from other patterns. The output is saved in a separate database. For example, the field “Associates” exists in most of the outputs, it is a list of the people involved in the sequence of detected fraud. The size of the list varies because the list will expand as the money transfers from one entity to the next entity, until it reaches the final destination or terminates with a fund withdrawal.

The number of the transactions (which can be modeled as steps in an event) required to create a MLS vary based on the MLS type. We link the set of transactions that constitute the comprehensive output for a specific MLS. Any previously examined transaction that is related to the current transaction under examination is linked to the output of the current transaction, if the transactions share common fields and involve a fund transfer. For example, all the associates who are involved in a Smurfing fraud will be placed in the list of “Associates”, and the manipulator is represented in the field “Boss”.

All the data gathered from pattern detection are reduced to the minimum set that represents what we define as the “Network”, which constitutes the entities created the MLS and information about MLS. These entities are “EndBoss”, “StartBoss”, and “Associates”. The “Network” DB contains fields of participants and vital information of a detected MLS. Each network will be assigned a unique ID. The nine fields in the “Network” are: NetworkID, EndBoss, StartBoss, AmountLaundered, Associates, Type, DetectionTime, StartTime, EndTime. A network can be associated only with one type of MLS; therefore, the field “Types” represents the various well-known types of MLS (Mehmet et al., 2010).

The evolution-output “Detection-Schema” is generated by the “Detection-Analysis” module. The DB “Detection-Schema” contains information about the evolution of the ML trail, such as information of average cost and average number of associates used in each of the sequence of MLSs in the Full-Trail.
3. SOCIAL NETWORK ANALYSIS MODULE TO CREATE A “SUSPICIOUS-TRAIL”

There are many cases in which money launderers intentionally obfuscate the money laundering trail, either by hiding it (for instance by increasing the transaction quantity and reducing the transaction amount), or performing it in a none-reported method as in Hawala. It becomes the task of an AML to detect these concealed relations and transactions. As a solution, we offer an additional module for social network analysis among transactions to unravel the hidden relations among MLS networks. MLEDF is designed from bottom-up with the concept of detecting and linking MLS trails (networks) even with missing evidence.

The major task of this module is to detect components of an actual “Full-Trail” even if there is a missing piece of evidence. The module will investigate the available trails (Full-Trail and Missing-Trail) by using our SNA DB that contains the weights of relationships among MLS participants. This is in order to determine if two trails are related by considering some attributes such as the amount of funds involved, location, affinity of participants, time, and methods used for laundering.

The SNA module is more resource consuming when compared to other modules, due to the extensive use of SNA, and link and weight calculations. The “Suspicious-Trail” module uses the “SNA” module to produce a new trail. This new trail contains two or more trails that are related based on SNA, even if we do not have captured a transaction joining them or any other evidence. The new trail is created after making a scientific calculation based on (SNA) results of a possible relationship between two or more “Full-Trails” and “Missing-Trails”.

The generated evolution patterns and strategies are collected into the “Suspicious-Trail” Database. This module contains the “SNA” sub-module that calculates and assesses the social network connections of individuals, peer-groups, and money laundering trails. The sub-module “SNA” is used to derive the associated suspicious trails based on the techniques of SNA. The table “Weight” that is used within the module is completely constructed with data output from running the “SNA” module.
3.1 Using Complex Event Processing in the Social Network Analysis Module

Any MLS has the originator “StartBoss” and the terminator “EndBoss”, in addition to the associates that maintains the MLS. The output of MLS detection algorithms contains information about the participants, in addition to information such as amount laundered, final amount of funds, type of MLS, duration and start and end time. The critical question that ML experts contend to answer is “How fast and how well can we relate the different events in this universe of detected MLS?” Using the introduction of Complex Event Processing (CEP) systems like StreamBase, we developed an algorithm to create the full and accurate chains of related MLSs, such relations are used to transfer a fund to the next MLS until it reaches the final destination. That is, the flow of the dirty money never stops until it reached the ultimate account. This cycle continues until it reached the final destination where the money launderer withdraws the money. Current AML systems have scalability issues in associating the multitude of different events of various MLS. We model each detected MLS as an event, and have various patterns of events categorized under six different types of MLS. For example, Full-Trail algorithm outputs a trail by using the functionality of CEP of perceiving the MLSs as a set of events. Without the CEP the MLS should dissolve into the constituent transactions to be analyzed and linked with the other transactions from another MLS (Time consuming and resource consuming). The CEP can link MLSs, perceived as events, using various criterions without the need to add more complex sub-algorithms for each criterion. That is, the Full-Trail connects the dots that exist, but it is harder and slower to connect them without CEP capabilities. Full-Trail captures the trail in cases where all evidence is available, whereas the Suspicious-Trail attempts to construct the path where some edges along the path is missing.

3.2 Integrating the “SNA” Module into MLEDF

The major objective of the “SNA” module is to detect components of a undiscovered Full-Trails by performing analysis on the four databases “Network”, “Detection-Schema”, “Full-Trail”, and “Missing-Trail”. During the relationship analysis stage, the SNA module investigate the available trails, by using SNA Database that contains the weight of relations to determine if two trails are related considering attributes, such as amount of funds involved, geography, affinity of participants, time, method for laundering, relation.

The module “SNA” analyzes the end and start points (“EndBoss” and “StartBoss”) of discovered transaction sequence (trail) to discover any broken parts of such none-discovered trail. This analysis includes assessing the social relationships between the endpoints (“EndBoss” and “StartBoss”) of trails with each other, using the DB created that includes different level of relationships. The participants and bosses of money laundering trails may change, but the key players stay the same and they swap roles (Mehmet et al., 2013).
3.3 Input from Algorithmic Modules and their Databases into the “SNA” Module

This section describes modules called inside the MLEDF and feed their output (DB) as an input to the “SNA” module.

1. **“Full-Trail” module**: We create the long trail “Full-Trail” of complex MLS series that span over many countries. A “Full-Trail” is a concatenated sequence of related ML schemes acts as a whole to transfer money from one MLS to another. A “StartBoss” of a “Full-Trail” is the “StartBoss” of the first MLS in the series of MLS that constitute the “Full-Trail”. Whereas the “EndBoss” of a “Full-Trail” is the “EndBoss” of the last MLS in the MLS series that constitute the “Full-Trail”. We detect the “StartBoss” and the “EndBoss” of the Full-Trail, along with intermediary bosses of linked schemes. The “StartBoss” is the earliest known launderer (that we have proven evidence for) that initiates the sequence of ML transactions. The “EndBoss” is the final launderer that withdraws the funds or transfers them using remittance (Hawala) systems that do not keep any financial records. We start with the “EndBoss” and compute the laundering path towards its beginning. Then we follow all possible paths that originate at the detected launderer “StartBoss” and link to another launderer. During concatenation of the schemes we consider the amount of funds involved, geography, affinity of participants, time, relation, and method for laundering.

<table>
<thead>
<tr>
<th>Networks</th>
<th>TrailID</th>
<th>Duration</th>
<th>Withdraw</th>
<th>Amount</th>
<th>StartBoss</th>
<th>EndBoss</th>
</tr>
</thead>
<tbody>
<tr>
<td>24, 51, 67, 92, …,</td>
<td>1932</td>
<td>56 Days</td>
<td>Yes</td>
<td>988,000</td>
<td>Boss 756</td>
<td>Boss 17</td>
</tr>
<tr>
<td>2, 15, 98, 126, …,</td>
<td>72468</td>
<td>99 Days</td>
<td>No</td>
<td>1,213,234</td>
<td>Boss 29</td>
<td>Boss 592</td>
</tr>
<tr>
<td>415, 783, 999, …,</td>
<td>97246</td>
<td>92 Days</td>
<td>Yes</td>
<td>1,050,230</td>
<td>Boss 324</td>
<td>Boss 75</td>
</tr>
</tbody>
</table>

2. **“Missing-Trail” module**: We create short trails that do not exceed the depth of three consecutive MLSs, or three levels depth of MLSs. We assume that Missing-Trail is a premature Full-Trail with broken parts and missing evidence. Therefore we capture such shorter trails for “Suspicious-Trail” analysis by saving them in the “Missing-Trail” DB.
Table 2 Sample Output of the Missing-Trail (Do not exceed 3 related MLS networks)

<table>
<thead>
<tr>
<th>Networks</th>
<th>TrailID</th>
<th>Duration</th>
<th>Withdraw</th>
<th>Amount</th>
<th>StartBoss</th>
<th>EndBoss</th>
</tr>
</thead>
<tbody>
<tr>
<td>14, 219,921</td>
<td>1232</td>
<td>16 Days</td>
<td>No</td>
<td>23,234</td>
<td>Boss56</td>
<td>Boss 151</td>
</tr>
<tr>
<td>2452, 315</td>
<td>1208</td>
<td>29 Days</td>
<td>No</td>
<td>90,165</td>
<td>Boss 170</td>
<td>Boss 882</td>
</tr>
<tr>
<td>405, 7831</td>
<td>97246</td>
<td>19 Days</td>
<td>No</td>
<td>200,230</td>
<td>Boss 884</td>
<td>Boss 975</td>
</tr>
</tbody>
</table>

3. **Evolution-Detection module**: We analyze the input feed from “Full-Trail” and “Missing-Trail” algorithms, and generate DB “ML-Networks” and DB “Detection-Schema”. The DB “ML-Network” contains three DBs of “Boss-Boss”, “Boss-Associate” and “Associate-Associate”. The three DBs reflect the all discovered pair relationships among bosses and associates of MLSs and trails. An associate is a participant of the MLS who facilitates the success of MLS, such as the deposit makers in smurfing or the stock broker in stock based MLS. The DB “Detection-Schema” contains statistical information about the evolution of the ML trail, such as information of average cost and average number of associates used in each MLS of the sequence of MLSs in the Full-Trail. Table 3 shows the shortened output of the detected schema of the real-life case.

Table 3 Sample Shortened Output of Detection-Schema

<table>
<thead>
<tr>
<th>DetectionTime</th>
<th>Type</th>
<th>SubType</th>
<th>NetworkID</th>
<th>Location</th>
<th>StartBoss</th>
<th>EndBoss</th>
</tr>
</thead>
<tbody>
<tr>
<td>201209151114</td>
<td>HiLo</td>
<td>Hi</td>
<td>2213</td>
<td>USA</td>
<td>Boss756</td>
<td>Boss 17</td>
</tr>
<tr>
<td>20120819139</td>
<td>Stock</td>
<td>LowSale</td>
<td>9786</td>
<td>Germany</td>
<td>Boss 324</td>
<td>Boss 75</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Associates</th>
<th>Cost</th>
<th>Amount</th>
<th>WithdrawalTime</th>
<th>StartTime</th>
<th>EndTime</th>
</tr>
</thead>
<tbody>
<tr>
<td>A, B, G, U</td>
<td>25,000</td>
<td>1,825,000</td>
<td>201209151114</td>
<td>20120830</td>
<td>20120915</td>
</tr>
<tr>
<td>N, O, W, Y</td>
<td>14,700</td>
<td>1970,000</td>
<td>20120819124</td>
<td>20120725</td>
<td>20120819</td>
</tr>
</tbody>
</table>

The “SNA” module will use the content of the DBs produced by the three evidence generation algorithms as an input. The contents of the output generated from the three modules listed in above tables will be saved into three DB named “Detection-Schema” DB, “Full-Trail” DB, and “Missing-Trail” DB. Every algorithm will create a DB with the same name of the algorithm name. Additionally, the DB “Network” associated with each MLS type will also feed into the SNA module.

4. **3.4 The Components and Output of the “SNA” Module**

The SNA module will generate the two Databases as outputs. The “SuspectWeight” Database contains the weight of all different relations detected in the MLEDF. The “Relations” Database contains the record of business and family relations among pairs based on the assumption that we have access to such records. The method to calculate the calculation is explained in the next section. The “SNA” module continuously updates the “SuspectWeight” with the value (score) of existing relations among the entities existing in all the DBs created in the MLEDF.
The SNA module contains nine sections to continuously update the two Databases. Section 1 through 6 creates hashes of various relations. The sections 7 through 9 update, query, and calculate SNA weight of family, business, and various ML relations derived in earlier stages. The hash of binary relations, that involves pair of entities, is used as the basis to calculate the accumulative relations of every type. The sections list is as follows:

a. Section1: Creates Hashes for all “StartBoss” and “EndBoss” relations, we refer to it as “Hash SchemaBosses”. The “StartBoss” and “EndBoss” are unique to a MLS.
b. Section2: Creates Hashes for all detected “Boss-Associate” relations we call “HashBossesAssociate”.
c. Section3: Creates Hashes for all detected “Associate-Associate” relationships. This hash represents the combinations of relationships among the associates of the same MLS, even if they do not interact/transact with each other directly.
d. Section 4: Creates Hashes for “StartBoss”–“EndBoss” pairs of Full-Trails, we call “HashFullTrailBosses”.

e. Section 5: Creates Hashes for “Relationships” among socially or business-wise related pairs.

f. Section 6: Creates Hashes for the associated (related) Full-Trail’s and Missing-Trail’s, we call “AggregateOfAssociatedTrails”. This hash contains a pair of TrailIDs, which are unique IDs assigned to each detected trail. The hash is used to relate trails by their TrailIDs.

g. Section 7: Inserts all the outputs (hashes) of sections 1,2,4,5, and 6 into “SuspectWeight” DB, using “FirstInsert”.

h. Section 8: Queries the “SuspectWeight” for the continuously updated outputs (hashes of sections 1-6), then, feed the updated hashes values into last section.

i. Section 9: Calculate the accumulative weight of relations using the “UpdatedWeight”, updating the “Relationship” table, and then updating the “SuspectWeight” with updated result combined from the new calculated weight and with “SuspectWeight”. The formula (method) used in this stage to sum up the accumulative weight, by adding the hash relations from sections 1 through 6, is explained below in the next section, This final and updated weight will be used in the “Suspicious-Trail” module to link trails among each other.

### 3.5 Social Network Analysis Algorithm

The “Social Network Analysis” algorithm computes the weight for different relationships involving the bosses and associates of MLS and trails. The values of the weights are chosen based on the importance of relation in a scheme, that is to say a relation of certain type is not treated equally as a relation with less importance. Also the margin of weights chosen allowing an iteration of certain relation to be equal in weight value to another relation with a higher weight, for example two “Boss-Associate” relations is value wise equal to one “Boss-Boss” relation.

A relationship weight is defined for each possible associate couple. The larger the weight, the more likely the relationship between two entities to occur. Weight is calculated by adding parameters for each of the corresponding events; therefore, the result is considered as the relationship weight.

- a. For each detected schema, 10 will be added to start/end boss couple, 5 for each boss/associate combination, and 1 for each associate/associate non-repeating combination.

- b. For each missing trail, 15 will be added to each associate non-repeating combination.

- c. The full trails will add 20 to each associate combination and 25 to the start and end boss.

- d. Other strong relationships are also counted, family ties will add 250 to the couple, and each business relationship will add 250 to the couple.

---

```plaintext
1 SET BUSINESS as 250; FAMILY as 250; FULL_BOSS as 25;
FULL_ASSOCIATE as 20; MISSING_ASSOCIATE as 15; SCHEMA_BOSS as 10;
SCHEMA_ASSOCIATEBOSS as 5; SCHEMA_ASSOCIATE as 1;

2 FUNCTION String HASH (String person1, String person2)
{ return concatenate(sort(person1,person2)) ; }

3 STREAM DetectionInputStream DetectionSchema detectedMLS;
STREAM RelationshipInput RelationshipSchema relationship;
STREAM MissingTrailInputStream MissingTrailSchema missingTrail;
STREAM FullTrailInputStream FullTrailSchema fullTrail;

4 STORE hashRelations IN hashAndRelationsNumberOfMemoryDB;
```
5 UPDATE hashAndRelationsNumberMemoryDB as H
   SET suspectSchemaBoss++
       WHERE H.hash == HASH(detectedMLS.startBoss, detectedMLS.endBoss);

6 UPDATE hashAndRelationsNumberMemoryDB as H
   SET suspectSchemaAssociateBoss++
       WHERE H.hash == HASH(detectedMLS.associate, detectedMLS.endBoss);

7 UPDATE hashAndRelationsNumberMemoryDB as H
   SET suspectSchemaAssociateBoss++
       WHERE H.hash == HASH(detectedMLS.associate, detectedMLS.startBoss);

8 FOR EACH detectedMLS.associates as assoc1
    FOR EACH detectedMLS.associates as assoc2
        UPDATE hashAndRelationsNumberMemoryDB as H
            SET suspectSchemaAssociate = suspectFullAssociate++
                WHERE H.hash == HASH(assoc1, assoc2);

9 FOR EACH fullTrail.associates as assoc1
    FOR EACH fullTrail.associates as assoc2
        UPDATE hashAndRelationsNumberMemoryDB as H
            SET suspectFullBoss = suspectFullBoss++
                WHERE H.hash == HASH(fullTrail.startBoss, fullTrail.endBoss);

10 UPDATE hashAndRelationsNumberMemoryDB as H
    SET suspectFullBoss = suspectFullBoss ++
        WHERE H.hash == HASH(fullTrail.startBoss, fullTrail.endBoss);

11 FOR EACH missingTrail.associates as assoc1
    FOR EACH missingTrail.associates as assoc2
        UPDATE hashAndRelationsNumberMemoryDB as H
            SET suspectSchemaBoss = suspectMissingAssociate++
                WHERE H.hash == HASH(assoc1, assoc2);

12 UPDATE hashAndRelationsNumberMemoryDB as H
    SET suspectBusiness = suspectBusiness + 1
        WHERE H.hash == HASH(relationship.person1, relationship.person2)
            AND relationship.type == "BUSINESS";

13 UPDATE hashAndRelationsNumberMemoryDB as H
    SET suspectFamily = suspectFamily + 1
        WHERE H.hash == HASH(relationship.person1, relationship.person2)
AND relationship.type == "FAMILY";

14 SELECT H.hash,
    (FULL_ASSOCIATE*H.suspectFullAssociate +
    FULL_BOSS*H.suspectFullBoss +
    MISSING_ASSOCIATE*H.suspectMissingAssociate +
    SCHEMA_ASSOCIATE*H.suspectSchemaAssociate +
    SCHEMA_ASSOCIATEBOSS*H.suspectSchemaAssociateBoss +
    SCHEMA_BOSS*H.suspectSchemaBoss +
    BUSINESS*H.suspectBusiness +FAMILY*H.suspectFamily) as WeightOutputStream
FROM hashAndRelationsNumberMemoryDB as H;

Query 1 The Social Network Analysis Algorithm

In 1 and 2 we define the constants associated with the different weights and the hash functions. In steps 3 and 4 we create the input feeds and local (temporary) MemoryDB. In steps 5 through 8 we create the hashes of “Boss-Boss”, “Boss-Associate”, and “Associate- Associate” of MLSs. In steps 9 through 11 we create the same hashes of Full-Trails. In steps 12 and 13 we create the hash for family and business “Relations”. In steps 14 we calculate the WeightOutput of a hash H.

3.6 The “Suspicious-Trail” Analysis Module

Using the “SNA” module, the “Suspicious-Trail” module produces a new trail that contains the full path of an evolution, after making a scientific assumption of a possible relation between two or more “Full-Trail” and “Missing-Trail” lists. This module continuously calls the “SNA” module to fetch the social network connections of individuals, peer-groups, and money laundering trails. The sub-module “SNA” is used to derive the associated suspicious trails based on SNA techniques to calculate the link weight (ML relations found in all MLEDF DBs) and attribute (business and family relations) weight among trail actors. The table “Weight” that is used within the “Suspicious-Trail” module is completely constructed with data output from running the “SNA” module. The generated evolution patterns and strategies are collected into the “Suspicious-Trail” Database.

3.7 Suspicious-Trail Algorithm

The “Suspicious-Trail” algorithm searches for missing and hidden links among ML trails. The analysis starts with any new “Full-Trail” with no more than 30 networks (step 5). Then, the current “Full-Trail” is matched with each trail that complies with (step 6):

a. Having ±10% of the current “Full-Trail” amount.
b. A time window of 90 days between both trails’ timestamps.
c. Location of first network of current “Full-Trail” is the same as the location of the last network of possible “Full-Trail” match.
d. Weight between current “Full-Trail” first boss and possible match last boss is larger than 1000.
e. All matches are then treated separately as study cases and related to the current “Full-Trail” forming a “Suspicious-Trail” (steps 7 and 8).
SET NetworkLimit as 30; Similar_FundPercentage as 0.1;
SET DayWindow as 90; SET WeightRelation as 1000

STREAM FullTrailInputStream FullTrailSchema Trails;
STREAM SNAOutputStream WeightSchema weight;
STORE Trails IN TrailsMemoryDB
SELECT Trails FROM Trails WHERE
(Trails.length <= NetworkLimit);
SELECT Trails, db as matchTrails FROM Trails as m, TrailsMemoryDB as db WHERE
((m.finalAmount * Similar_FundPercentage < current.finalAmount) AND
(m.finalAmount > db.finalAmount* Similar_FundPercentage) AND
((m.detectionTime-db.detectionTime) <= days(DayWindow)) AND
(lastelement(db.networks).location == firstelement(m.networks).location));
SELECT CONCATENATE(t,m) as SuspectedTrailOutputStream
FROM Trails as m, matchTrails as t, weight as w WHERE
w.hash == sort(t.startBoss,m.endBoss) AND w.weight >= WeightRelation;
SELECT CONCATENATE(m,t) as SuspectedTrailOutputStream
FROM Trails as m, matchTrails as t, weight
FROM Trails as m, matchTrails as t, weight as w WHERE
w.hash == sort(m.startBoss,t.endBoss) AND w.weight >= WeightRelation;

Query 2 The Suspicious-Trail Analysis Algorithm

In step 1 we define the constants associated with limits set to relate and compare trails. In steps 2 through 4 we create the input feeds including the SNA-generated Weight DB, and local (temporary) DB TrailsMemoryDB. In step 5 we filter the analyzed trails that do not exceed 30 levels. In step 6 we compare and link trails that match based on criteria listed above. In steps 7 and 8 we concatenate the matching trail with another trail, either trail or matching trail leading to the generated outcome.

4. EXPERIMENTAL EVALUATION

It is hard to obtain real-life data in the domain of ML, where one can find some samples used to explain complex real-life cases. Therefore we approached several organizations to collect sanitized real-life cases that validate the testing of MLEDF, meeting the requirements imposed by the organization that provided the sanitized cases. Our case studies are based on data provided from the organization we refer as Trusted Third Party (TTP), which is legally allowed to collect information and track records of financial exchange. The identity of the TTP cannot be disclosed due to a Non-Disclosure Agreement. The sanitized cases were provided on the basis of having the MLEDF also tested in the infrastructure of TTP.

4.1 Experimental Setup

Using the real-life dataset, we generate a larger dataset that contains different levels of random transaction using a module we implemented using Java. We used a template from real-life cases to generate the synthetic data that is similar to those cases by selecting a subset of real-life cases to create more samples and develop new patterns, based on criteria such as preserving duration flexibility, geography variation, multitude of fund transferred, crowded trails, trails with low funds,
complex instruments such as derivative products, continuous transition from one financial sector to the other, splitting a transaction with large fund into many connected small funded MLSs, etc. Once we generate artificial data sets we unite all the databases to create a large dataset to be inserted into the MLEDF for validation and testing. All the “endBoss” and “startBoss” of trails and generated MLS series are compared against the detected ones of MLEDF. The same test was repeated with inserted random patterns of small trails and MLSs in some interval to confuse the MLEDF and test the false positive rate (FPR) and false negative rate (FNR). By adding a combination of randomly generated MLS series we imitate the daily production environment of stock brokerage and a retail bank. The real validation test for MLEDF is accomplished by assessing its ability to detecting patterns with accuracy in a noisy environment, similar to the real life transactions that are filled with noise.

4.2 Using Real-Life Cases in the Validation Test of the MLEDF

Our test case spans over 5 countries involving 11 companies, 10 associates, and 8 innocent entities. As shown in Figure 5 and Table 5, the scheme has 3 different full trail cycles. The laundered amount is lower than the amount transferred by the “StartBoss” of the cycle. The amounts in the case only reflect the amount laundered, either by gaining and losing, or by means of transferring the value. The amount does not represent the full amount of the transaction, which is higher than the amount laundered. The masterminds of the scheme are Company1 and “EndBoss”, based on the information provided to us by TTP. The sub-cycles of the case occur independently of each other and each has different rounds. Each round in each sub-cycle is in tandem with other rounds of the same sub-cycle. The information of individuals and companies, locations, and dates are all sanitized.

Figure 2 Real-Life Case of Evolved and Sophisticated MLS
4.3 Experimental Evaluation

We introduced a three phase testing prototype to examine MLEDF and detection algorithms. All of the three phases focused on testing and validating the components of MLS, Full-Trails, and Suspicious-Trails. The first phase focus on testing all components and the other tests focus on Full-Trail and Suspicious-Trail components.

Table 6 Used Testing and Validation Methodology

<table>
<thead>
<tr>
<th>Test-Validation Type</th>
<th>Patterns Used</th>
<th>Pattern Generation Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test (I) Without Noise</td>
<td>Single MLS, Missing-Trail, Full-Trail</td>
<td>StreamBase-Generated single MLS, StreamBase-Feed Pair of MLS, Feed Full-Trail's from Real-life Cases</td>
</tr>
<tr>
<td>Test (II) Subtle Noise</td>
<td>Entities, Transactions, Single MLS</td>
<td>Inject same entities into Full-Trail, Inject subtle transactions into Full-Trail, Inject similar MLS into same Full-Trail</td>
</tr>
<tr>
<td>Test (III) Controlled Data</td>
<td>30L Deep Trail, 20L Deep Trail, 10L Deep Trail</td>
<td>Full-Create 30 levels vertically deep trails, Full-Create 20 levels vertically deep trails, Full-Create 10 levels vertically deep trails</td>
</tr>
</tbody>
</table>
### Table 7 Defining False Negative Rate for Each Test Phase

<table>
<thead>
<tr>
<th>Test Phase</th>
<th>Patterns Used</th>
<th>FNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test I</td>
<td>Single MLS, Pair MLS, Full-Trail</td>
<td>Detected MLS list is less than the input list, Detected MLS pairs is less than the input list, or MLS pair is detected as single MLS, Detected Full-Trail list is less than the input list, or Full-Trail is detected as Missing-Trail’s (shorter trails) and single MLSs</td>
</tr>
<tr>
<td>Test II</td>
<td>Entities, Transactions, Single MLS</td>
<td>Missed detection of MLS, because similar participants injected, Missed detection of MLS, because identical transactions injected, Missed detection of trail, because similar MLS injected</td>
</tr>
<tr>
<td>Test III</td>
<td>30L-Deep, 20L-Deep, 10L-Deep</td>
<td>30L-Deep is missed in detection, and it will cause to be detected as (FPR) other level deep combination of Full-Trail (less than 30L), Missing Trails and MLS, Missed detection, and causing generation of FPR of Full-Trail (less than 20L), Missing Trails and MLS, Missed detection, and causing generation of FPR of Full-Trail (less than 10L), Missing Trails and MLS</td>
</tr>
</tbody>
</table>

### Table 8 Defining False Positive Rate for Each Test Phase

<table>
<thead>
<tr>
<th>Test Phase</th>
<th>Patterns Used</th>
<th>FPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test I</td>
<td>Single MLS, Pair MLS, Full-Trail</td>
<td>Not possible as MLS is either detected or missed, as there is no noise Not possible as MLS pair is either detected or missed, in no-noise data Full-Trail is not captured, instead MLS pairs, MLS triple, or shorter Full-Trails are captured in lieu of the Full-Trail</td>
</tr>
<tr>
<td>Test II</td>
<td>Entities, Transactions, Single MLS</td>
<td>Detect MLS with different participants (Associate and/or Boss), Detect MLS with different participants (Associate and/or Boss), Missed detection of the actual Full-Trail. Instead of that MLS, Missing-Trails and shorter Full-Trails that form the actual Full-Trail captured</td>
</tr>
<tr>
<td>Test III</td>
<td>30L-Deep, 20L-Deep, 10L-Deep</td>
<td>Full-Trail of desired depth (10L, 20L, 30 L) is not captured, instead a combination of Full-Trail of less depth, Missing-Trail, and MLS are captured in lieu of the actual Full-Trail</td>
</tr>
</tbody>
</table>

**Test without noise:** This is designed to test every module of MLEDf, including detection algorithms and trail analysis modules. These tests evaluate the FPR and FNR by comparing the results of the test
with the data feed that contains the patterns of single MLS, pair of MLSs, and Full-Trails. The desired result is to have a list of the validation result identical to the list in the data feed. We test the efficiency to keep up with the speed of the data feed by using the time window feature in the StreamBase (StreamBase 2012, 2014). By setting the time window to glide over only one event at a time tick in the StreamBase system, we required the detection algorithms to be fast at normal speed of one event at one time tick of the CEP system. By design, algorithm that cannot attain the speed of event production will not be able to capture MLS events or the Full-Trail, thereby generating false negatives.

Each of the six detection algorithms are tested with its own dataset feeds in order to verify that we detect without FPR and FNR. The algorithm-specific dataset feed is generated using the built in feed generator working with our pattern specific event generator. Afterwards, we tested the “Missing-Trail” by feeding linked pairs of MLSs into the MLEDF. The linked/related pairs are randomly selected from the set of six types of MLS. As mentioned, any pair of linked MLS will make it to “Missing-Trail” and not into “Full-Trail”, due to the required depth. Moreover, we finally tested the detection and evolution of “Full-Trail”s by feeding trails generated using the various laundering strategies of the real-life cases.

The process of creating the “Full-Trail” will start with creating an MLS type out of the six MLS types of Smurfing, Trading, DirtyEFT, Stock, Derivative, E-Money. Once the selection of first MLS is made, we create the linked MLS series based on considerations such as geography, money-amount, time, complexity, difficulty of tracking. The trails were created considering different levels of criteria, the randomization of the criteria is uniformly distributed. The Full-Trail feeds were created by the generator that does not exceed 10 levels of depth of linked MLSs. The trails are either a variant or a subsection of one of the real-life cases that are similar in terms of complexity and people involved to the case explained previously.

At the normal speed, of one event at one time tick of the CEP system, the test result is zero for the false positive rates and false negative rates. It is highly improbable to get a false positive trail due to the business rules that define them, and due to the accuracy and granular level of linking transactions. We did not get any false positive schema in the MLS tests, due to the synthetic nature of the data. When we increased the speed of the feed of data generated to 10 times and 100 times the normal speed, we observed a FPR and FNR in the objects detected (Full-Trail). Increasing the speed of data-feed processing did not produce FPR and FNR for single MLS, but it produced FPR and FNR for pair MLS at speed 100X. The term “object” in the graph refers to the three different patterns of single MLS, pair MLS, and Full-Trail in the proprietary test of the specific object (Object in the first pattern test refers to the first pattern single MLS, in the second to MLS pair, in the third to Full-Trail). The values of FRP and FNR reflect the number of falsely detected objects.
Test with subtle noise: This is the most relevant test of the accuracy of our detection algorithms. The goal is to mislead the detection algorithms to generate both a false positive and false negative, with the use of subtle synthetic data. The test has three separate phases: injecting the scheme participants, injecting subtle transactions, and inserting similar MLS. A subtle transaction means an identical transaction with ±5% of an actual transaction amount in a MLS. A similar MLS means an identical MLS with the same set of participants but with the MLS value is ±10% of the laundered amount of the MLS. The injection speed was performed at normal processing speed, 10 times faster speed, and 100 times faster speed. The test of injecting transactions and MLS is setup considering each MLS type. For example, in the test of smurfing we create only smurfing MLS and smurfing transactions that can extend vertically up to 20 levels of depth and horizontally to 30 levels of depth. When generating the MLSs, our measures vary based on the MLS. We do not use artificially created none-real life cases. For example, we did not use a smurfing MLS with 100 levels deep, as that is uncommon and impractical to launder money using such MLS. We do not inject other MLSs into the injection testing of specific MLS. However, in the Full-Trail testing we inject all the types of MLS. By design of full-Trail it is required that we related different types of MLS under the same Full-Trail.

As it can be seen from Figures 7-9, the test produced low FNR and low FPR for transaction and MLS injection when the phases were executed at normal processing speed. Those rates increased in the phases when tests were executed at faster processing speed. One way to imitate the data feed pace of real production environment is to run the CEP tests at faster pace, which means overloading the system with processing and analytics while attempting to keep pace with the data feed. The goal was to evaluate the effectiveness of “Full-Trail” detection when the system absorbs data at a higher rate while performing the analysis. Due to the design methodology of detection algorithms and the complexity of the business rules of MLS detection, their false detection rates stayed at low levels even with injection similar transactions and MLSs, at a higher data-feed speed.

Meeting the design principles, the “Full-Trail” and “Suspicious-Trail” results remained at low rates for both false positive and false negative. Therefore, all the subtle single MLS created with our injected data ended in the “Missing-Trail”s, where they do not exceed the depth of 3 consecutive MLSs. Among the reasons for such success in trail analysis and avoiding any negative impact are the following: (1) Designing MLEDF in such a strict and granular method, especially for matching the MLSs within the trails. (2) Using the SNA in the trail analysis algorithms. (3) Adopting the criterion of following the direction of the money flow. MLS is not expected to terminate with funds remaining.
in the account. The money must flow in some direction in order to be laundered, or must be withdrawn by the launderer.

Figure 3 Details and False Detection Results of Test II-Entity Injection

Figure 4 Details and False Detection Results of Test II-Transaction Injection

Figure 5 Details and False Detection Results of Test II- Similar MLS Injection

ADFSL Conference on Digital Forensics, Security and Law, 2013
**Test with longer synthetic full-trails:** This is the hardest level of performance-testing of the system and accuracy-testing of the detection algorithms. The dataset is permuted over a repository of the different real-life cases. Afterwards, the dataset is combined with randomized MLS to generate deep vertical levels of “Full-Trail” and “Suspicious-Trail”. The randomization follows the same principles we used in Test II, the injection testing. The test is designed to assess the performance of MLEDF in capturing real-life data and analyzing them on the fly. The desired test result is to generate low FPR and FNR. The test module generates all synthetic data from real-life cases. The test is as similar as it can be to real-life scenarios, considering that there are limited ways to manipulate a MLS. The test program functions as follows: (1) Set a trail depth. The program enters a loop and builds a trail by choosing a first scheme from of each MLS type at random, as it was described in Test I in building the Full-Trails. (2) The loop continues by creating an MLS that can be linked by funds, time, location and complexity wise to the current MLS. We repeat the step above with the exception of not creating any Smurfing MLS for the rest of the levels. (3) The permutation continues until we reach the last level, where we always choose an MLS of type DirtyEFT with withdrawal, in order to generate the trail termination point, as by definition a trail will end with withdrawal. (4) The test repeats the process of trail generation forever, and at the maximum possible speed. (5) The testing module saves the arrival time of the last DirtyEFT and subtracts that from the build time of the trail. Thereby, we obtain the difference in Milliseconds, which represents the time duration for trail processing.

The data was generated for worst-case scenarios. By doing so we ensure that the generated data is more complex and that the performance is evaluated only in most resource consuming cases. Displayed results represent the performance of data generated without any repetitive bosses or associates. Hence, the dataset is consumes a significant number of resources.

![Chart](chart.png)

**Figure 6 False Detection Results and Details of Test III**

### 4.4 Data Characteristics

We introduced six MLS types with different subtypes for each sector. The combination of schemes is novel in its entirety as they were driven from real-life cases. The novelty lies in (1) creating patterns from real life cases using CEP system, (2) developing a software that can read patterns from real-life or synthetic cases and evolve it based on criteria we implemented, (3) attempting to link the networks of from those cases to produce different MLS variations that involve all well-known MLS.

In the first test, we used the StreamBase (SB) simulator (StreamBase 2012, 2013) and MLS DB to feed data into the application. In the tests of the MLS pair and “Full-Trail”, the simulator retrieves the DB data and generates samples. In the single MLS test, we feed the DB data into the simulator to generate samples from a pool of a specific set of MLS. The “Full-Trail” dataset contains variants and
subsections from five real-life cases. The dataset was in CVS format, and contained 292,000 MLS records that constituted a total of 12,000 “Full-Trail” records. The dataset contained 1.4 million transactions. In the second test, we used a modified version of the dataset that contains subtle transactions and identical MLSs. We injected 20,000 identical MLS and 95,000 subtle transactions using the simulator. In the third test, we implemented a Java program to stream the generated data into the MLEDF.

4.5 Testing Performance

In order to test performance using the environment described above, we executed three major tests over three different data sets. The third simulation test was the most crucial test, as the data feed resembles the data feed seen on a daily basis in a bank or in a brokerage house.

The processing ratio is ideal, obviously workable to be faster. The trail processing time decreases with the deeper trails because there are less MLS per object in this case. However, it is a good indicator because it means that in more healthy situations, the system will be faster in responding to troubles. The processing time both for objects and MLSs is set to be ideal on 20-depth trails. This implies that the system is more suitable to trails of that particular depth. With our assessed speed to get ~0.5 objects/millisecond processing, and for such a system, this is significant. While the processing ratio can be very subjective to the number of markets, for banks and other feeds connected to the system, a productive version of a system like this will need to be scaled in order to meet that ratio.

<table>
<thead>
<tr>
<th>Pattern Generation Speed</th>
<th>CPU and Memory Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLS/second</td>
<td>Objects/second</td>
</tr>
<tr>
<td>30-Depth</td>
<td>11.54</td>
</tr>
<tr>
<td>20-Depth</td>
<td>20.79</td>
</tr>
<tr>
<td>10-Depth</td>
<td>30.86</td>
</tr>
</tbody>
</table>

Figure 11 Performance Test Results

4.6 The Validation Statement

We used the SB Studio Feed Simulation platform for performing our tests during the validation, (1) to connect it to module generator of Test III to retrieve the deep Full-Trails, (2) to connect it to the data-feed we used in Test II, (3) to create noise-free pure data in Test I. Although there are higher performance opportunities promised by vendors, SB in particular, we observed that our queries reveal reasonable outcomes in terms of performance tests, even using this none-enterprise test environment. We also observed that the queries result in reasonable accuracy values given the specially crafted synthetic data sets. Given the data set, we successfully determined a reasonable window size for window-based queries. Using attributes, we could also successfully converge the evidence outcomes by tuning the force of time, property, and key-based patterns directing those queries.

RELATED WORK

The system created by FINCEN AI System (FAIS) (Senator et al., 1995, 1998) is a similar system in terms of the concept of detecting MLS. The drawback of the system is that it does not capture live data (i.e., running data connected to banking systems) and requires the involvement of an expert in the link
analysis, and tying MLS and transactions. Whereas our system does not need the involvement of an expert and captures live, as well as feeds the data into the SB engine. The KDPrevent system (Jacobs et al., 2003; Kuns et al., 2004) is the most private product that is similar to our product in terms of the logic of including the background of the transactions of individual and groups. The KDPrevent system is also based on data mining techniques that is not real-time and necessitates the involvement of experts. Gunestas, et al. (2010) conducted a similar study in the financial transaction forensics, with a focus only on Ponzi schemes. The framework only captures one form of transaction only from the web services transactions. Our framework can be accepted as a continuance to (Gunestas et al., 2010) for MLS and accepting all forms of transactions, including the banking transactions.

6. CONCLUSION

We have created a framework to detect the evolution of MLS and implemented a system to include SNA for detecting and linking related ML networks. The linkage will function properly even when all evidence is unavailable. We defined the choreographies that could be used to detect the evolution of the sophisticated MLS. We have shown how to detect and capture the evolving and complex trails of MLS using SB. Although our choreographies only specify well-known money laundering schemes, the framework can be updated with business rules to capture any form of other MLS that can be mined from repositories of financial transactions. Our ongoing work addresses the extension of our method in developing an online warning system that detects MLS that appear legitimate from an abstract view, but are illegitimate from the detailed view. Currently, we are working to produce algorithms to prevent transactions from a sequence of financial transactions, based on our detection system and proprietary scoring system.

REFERENCES


SYSTEM-GENERATED DIGITAL FORENSIC EVIDENCE IN GRAPHIC DESIGN APPLICATIONS

Enos Mabuto, Hein Venter
University of Pretoria
South Africa

ABSTRACT
Graphic design applications are often used for the editing and design of digital art. The same applications can be used for creating counterfeit documents such as identity documents (IDs), driver’s licences, passports, etc. However, the use of any graphic design application leaves behind traces of digital information that can be used during a digital forensic investigation. Current digital forensic tools examine a system to find digital evidence, but they do not examine a system specifically for the creating of counterfeit documents created through the use of graphic design applications.

The paper in hand reviews the system-generated digital forensic evidence gathered from certain graphic design applications, which indicates that a counterfeit document was created. This inference is made by associating the digital forensic information gathered with the possible actions taken, more specifically, the scanning, editing, saving and printing of counterfeit documents. The digital forensic information is gathered by analysing the files generated by the particular graphic design application used for creating the document. The acquired digital forensic information is corroborated to the creation of counterfeit documents and interpreted accordingly. In the end determining if a system was utilised for counterfeiting.

Keywords: Digital evidence, Digital forensic, Digital forensic artifacts, Graphic design applications.

1. INTRODUCTION

Industries including but not limited to advertising, newspaper printing, architecture, fashion and design, project management and manufacturing make use of graphic designs for their corporations. Graphic design applications have enhancing tools like paint brushing, vector drawing, digital pen and pencil drawing, and many more. These graphic design applications are used to facilitate the creation of unique art for company logos, magazine advertising or computer-aided design, to mention but a few. Most industries make use of graphic design applications for visual presentations and use pictorial expressions that aid communication and the expression of ideas.

Forged or counterfeit documents are, however, encountered and in circulation all over the world. The same graphic design applications used in modern industry can also be used for illegitimate purposes like creating counterfeit documents. Due to the exceptional editing and design capabilities of these applications they can easily be exploited and misused to create counterfeit documents like IDs, passports or drivers licences. According to a newspaper report by Ilham Rawoot of the Mail & Guardian, terrorist’s target fake South African passports because of the ease with which they can be faked [1]. Criminal activities such as these confirm the need for digital forensic investigations.

Similar digital forensic papers have been published that identify image forgery or tampered images [18, 19]. However, not much has been done in such research to identify whether a specific system was used during a counterfeiting exercise. Therefore, if no evidence is available for proving that a counterfeited document exists, counterfeiting criminals can potentially get away with it. It is, thus, relevant to examine a system specifically for the potential existence of counterfeit documents.

The use of graphic design applications leaves behind traces that can be revealed during a digital forensic investigation. A digital forensic investigation generally consists of the following phases consisting of the acquisition, examination, analysis and reporting [2]. Assuming that an individual is
suspected of creating counterfeit documents, the regular process of acquisition is followed. The phases of acquisition and reporting are generally similar in different cases; hence the emphasis is on the examination and analysis phases.

This paper identifies the digital traces left behind when certain graphic design applications had been used. This is achieved by associating the possible actions taken during document creation with the traces left behind. The source of potential evidence referred to above equates to the results of possible actions (i.e. document scanning, editing, saving and printing) taken during document creation. Most of this evidence would originate from the application log files, referred to as system-generated evidence.

The work covered in this paper continues from previously-published work by the authors on “User-generated digital forensic evidence from graphic design applications” [16]. The mentioned paper elaborates on gathering potential evidence on the actual files with counterfeit value created by the counterfeiter intentionally. As opposed to the previous paper [16], the focus of this paper is on the files generated by the graphic design application itself, mostly for the purpose of metadata that would hold potential evidence. Another similar paper published by the authors titled “Finding digital evidence from graphic design applications” [17], presented digital evidence on a high level.

To address the problem, the authors focus on identifying the digital forensic information that shows whether a document was created through the mentioned four actions. In doing so, a link with the potential criminal may be established. However, it is not the aim of this paper to link the crime to an actual person but merely to establish that a counterfeit document was indeed created.

The remainder of the paper is structured as follows: Section two starts off with some background on digital forensics, followed by a brief discussion on graphic design applications. Section three presents the system-generated digital forensic evidence gathered by means of two experiments, while Section four is an evaluation and discussion of the evidence extracted from the graphic design applications. Section five serves as conclusion to this paper.

2. BACKGROUND

In part A, the authors discuss the studied literature on digital forensics, followed by an explanation of digital evidence and a definition of digital forensic artifacts. Part B contains a brief discussion of the three Adobe graphic design applications used for the purposes of this study.

2.1 Digital Forensics

At the Digital Forensics Research Workshop (DFRWS) in 2001, digital forensics was defined as the use of scientifically derived and proven methods toward the preservation, collection, validation, identification, analysis, interpretation, documentation and presentation of digital evidence derived from digital sources for the purpose of facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorised actions shown to be disruptive to planned operations [3]. To reconstruct and understand what happened on a system in the past, data has to be gathered and analysed in a transparent manner.

A digital forensic investigation focuses on finding digital evidence when a computer or network security incident has occurred, or locating data from systems that may form part of some litigation, even if such data has been deleted. In this context, evidence is critical and any items that can be considered to be of evidential value should be identified and collected [4].

Computer evidence or digital evidence is defined as any hardware, software or data that can be used to prove one or more of the ‘who, what, when, where, why and how’ questions pertaining to a security incident [5]. Computer evidence furthermore consists of digital files and their contents that are left behind after an incident. Casey defined digital evidence as any data that can be used to establish that a crime was committed or that can prove a link between a crime and its victim or an offender [6].
evidence consists entirely of sequences of binary values called bits [7]. It is important to keep in mind, however, that the evidence should be presented in its logical form in court or at a disciplinary hearing.

Traces left behind from the use of an application or operating systems are referred to as digital forensic artifacts [8]. An examiner reveals the truth of an event by discovering and exposing the remnants of the event that have been left on the system. Because of the loaded legal connotations binding the term ‘evidence’, the term ‘artifacts’ is preferably used instead to refer to these remnants. When a perpetrator tries to remove these artifacts, it potentially leaves other artifacts behind. For example, in trying to remove log files from a system, one typically might use a removal tool, which leaves additional traces indicating that a log removal tool was used. The scattered evidence inside a system can indicate what has happened for a particular digital forensic investigation.

Application artifacts left by installed applications can be an excellent source of potential evidence when performing an analysis. An artifact, however, does not become evidence unless its ability to prove a fact has been established [9]. Hence it is necessary to reconstruct events that occurred by gathering all the possible digital information from a system.

The amount of research and development that has been undertaken in this field has not, to date, focused on the skills and of graphic design software, which is a particular area that is nearly always exploited for the purpose of creating counterfeit documents and images. Most research work that has been undertaken up till now has concentrated on image forensics, which is the kind of investigation that is able to determine whether or not an image as been forged or tempered [18,19].

Lien [18], proposed a method that uses a pre-calculated resampling weighting table to detect periodic properties in error distribution within an image. The errors in the distribution within an image are used to determine if the image has been forged. Stamm [19] proposed a method to detect contrast enhancement and addition of noise in ^{jpeg}^ compression images. Changes in contrast and noise within an image are determined through the use of an algorithm that calculates pixel values within the image. The values are then used to detect forgery within the image. Cohen [20] proposed a method that determines characteristics associated within digital still camera images to determine the origin of the image. The characteristics are compared to the exact replicas and derivates of other statistical images to detect forgery. These, [18, 19, 20], and other related work focus on determining forgery using statistical data within the image [21, 22, 23, 24].

Very little of the research carried out to date has specifically investigated the ways and means in which documents are counterfeited. These ways also include the methods and procedures that can be used to detect such activities from graphic design applications, which is the focus of this paper.

How and where evidence is located differs, depending on the crime being investigated, the platform (operating systems) and the application used to commit the crime.

2.2 Graphic design applications

Of the many graphic design applications currently available in the industry, Adobe Systems Incorporated is regarded as the largest software maker in the graphic design software category [10] and hence the reason for focussing on graphic design software from Adobe Systems for this research. Adobe Systems Incorporated owns software technologies that are used for online transactions, business applications and social technologies [11]. The case study for the current research was therefore conducted with Adobe graphic design applications, namely Photoshop and In-Design.

3. DIGITAL FORENSIC EVIDENCE GATHERED FROM GRAPHIC DESIGN APPLICATIONS

In this section, the authors start off by explaining the research method used in this study to create the counterfeit documents, referred to as the experiments. Secondly the authors illustrate the results
obtained from the experiments, referred to as the gathered digital forensic artifacts. A summary elaborating on the results concludes this section.

3.1 Experiments

‘System-generated digital forensic artifacts’ refer to those artifacts created by the application without direct user intervention, while ‘user-generated digital forensic artifacts’ refer to artifacts intentionally and directly created by the user. The latter are not analysed in this paper.

The research experiments were conducted in two stages. The first experiment was conducted to simulate the activities that can be performed by an offender and is referred to as the 'counterfeiter experiment'. The second experiment was carried out to trace the activities of the offender and is referred to as the 'investigator experiment'. An explanation of the two experiments follows.

3.1.1 Counterfeiter experiment: Creating the counterfeit documents

The researcher created approximately three hundred dummy counterfeit documents by using the graphic design applications that were discussed earlier in this text. The motivation behind the creation of approximately three hundred documents is as follows. These documents were created during the experiment by editing the following four components within a South African Identity Document (ID), passport and drivers license: the barcode, fingerprints, signatures, and photographs of human faces. This required a combination of twenty four options (4! (Factorial)= 24) on eleven examined file types. The combination for all file types equalled two hundred and sixty four (24 x 11), and included a few extra repetitions for clarity, yielding almost three hundred documents. This was so that the authors could be able to notice the difference or the changes to the digital forensic artifacts as more documents are created. Different application versions usually bring about more application capabilities and enhanced digital tools which can result in potential changes to digital forensic artifacts. These changes will be explained later in the results section.

Since most graphic design application users prefer the latest editions, the most recent version of Adobe, CS5, was used for this study as the base experiment. Further experiments were carried out on CS3 and CS4 for comparative purposes. Three different computers were used, each with a different Adobe version installed on it. The counterfeit documents were created by performing the actions mentioned before (scanning, editing, saving and printing). The 'platform' refers to the operating system on which the counterfeit documents were created. According to software reviews in 2011, the Windows operating system is still ranked most popular [12, 13] and the analysis of digital forensic artifacts was consequently conducted on a Windows 7 platform.

3.1.2 Investigator experiment: Searching for the evidence

Once the counterfeit documents had been created, experiments were carried out to search for pertinent evidence left behind from the use of the graphic design applications. The operating systems’ registry editor tool, ‘regedit’ was used to search for associated registry entries, while a hex editor, Winhex [14] was used for analysing the binary data of the log files.

To respond to the problem stated earlier, that there are no digital forensic investigation software tools available yet to investigate crimes where graphic design applications can be used for creating counterfeit documents; four possible actions taken during the creation of a document were used as a hypothesis to gather digital forensic information related to the graphic design applications. The analysis is formulated to find the digital forensic information that indicates that the actions (scanning, editing, saving and printing) had indeed taken place. By tracking the actions performed, an investigator is able to conduct a systematic investigation aimed at acquiring not only the files used to create the document, but also the actual documents created to be used as potential evidence. For example, if the document was scanned, then the next step would probably be that it was edited. If
never scanned then probably it was edited only. In the end, it becomes possible to state if the document created was a counterfeit document or not.

If none of the four actions were taken, then there is no need to ascertain whether the application was used for document creation. An illustration of the results from the experiments follows.

3.2 Results from the experiments: Gathered digital forensic artifacts

The discussion that follows highlights the digital forensic artifacts found in graphic design applications where the source of the potential evidence is mainly system-generated and results derive mostly from application log files.

Experimental results obtained from digital forensic artifacts related to the four actions (scan, edit, save and print) are elaborated on in each of the subsections to follow.

3.2.1 Artifacts related to document scanning

Generally, when one attempts to create a fraudulent document, an original document has to be acquired to imitate or copy its identity. Scanning is a common option that results in the original document being available on computer for digital editing. The different models of scanners that are currently available use various software packages for executing scan commands. For the purposes of this research, the focus is therefore on commands generated from within the graphic design application and used for editing the scanned document.

Adobe Photoshop has the capability to scan a document using the ‘import WIA support’ document menu option. The document scanned is loaded into a destination folder as prompted. The application creates a folder, saves the scanned image and opens the scanned image in the application.

After a document is scanned, the application records the digital artifact (evidence for scanning) into one of its log files named Adobe Photoshop CSX Prefs.psp located in C:\Users\<username>\AppData\Roaming\Adobe\Adobe Photoshop CSX\Adobe Photoshop CSX Settings. The X in CSX represents the version of the graphic design application, which can be 3, 4 or 5. After the authors analysed this psp log file, they identified an entry recorded of the location of the scanned file at certain address offsets to be discussed in section 3.3 summary. Through examining this location, the authors were able to identify the copies of the original documents scanned for possible counterfeiting.

Adobe In-Design is not capable of scanning a document. In this case, if the application used cannot scan a document. Then the user could use the scanners own software, this means that the scanned document will be loaded into the application through the “place” function. As long as the application user has inserted the scanned document into the graphic design application, it is possible to trace the particular image inserted as shall be described in the sub section “artifacts related to document editing”. Even if not all actions are exercised(scan, edit, save and print), the traces obtained from any recognised actions are used to determine, for example what was inserted in the document and what the saved document created is. This would enable an investigator to visualise these aspects and determine if a counterfeit document was created.

After scanning, the regular process followed by a potential criminal is to edit the acquired document in a bid to falsify its content. This editing process is discussed in the next section.

3.2.2 Artifacts related to document editing

Document editing is one of the important stages of creating a counterfeit document as it allows one to insert objects of interest. For example, a human face, a bar code or a fingerprint can be inserted in the scanned document. A number of editing actions can be performed, including typing, colouring or drawing. Our focus is on editing by insertion of an image or object, as this can later be used to determine if the document created was counterfeit or not. Regarding inserted objects, experiments
were executed to establish what can be inferred from a system that indicates to the examiner what was inserted and from which location it was inserted. The terms ‘inserting’, ‘attaching’ or ‘placing’ an image are considered to refer to the same action, though called differently in various applications. In this paper, the term ‘inserting’ is used henceforth.

The same log file, *Adobe Photoshop CSXPrefs*, records digital information with the name of the inserted file and the location from which it was inserted.

Adobe In-Design can also perform the action of inserting an image into a document. In-Design log files consist of *FindChangeData*, *FontMaskCache*, *In_DesignDragDrop* and *idletask*. This application records digital artifacts for editing entries into one of its log files. The log file named *InDesign SavedData* (without a file extension), which is located at `C:\Users\<username>\AppData\Local\Adobe\InDesign\Version 5.0\Cache`, contains the information that indicates the name of the location from which an image was inserted. Unlike Adobe Photoshop, Adobe In-Design only records the folder location or the path of the inserted images, and not the full name of the inserted image.

From these locations, the authors were able to obtain the actual images used during document editing, for example, images of a human face and fingerprint images. These images are essentially necessary for counterfeit investigations as they can be used for compare to the images within the suspect counterfeit document.

### 3.2.3 Artifacts related to document saving

Once a document has been edited, the user (or potential criminal) usually needs to save it either for later printing or further editing. In this section the authors examine what is found in the system relating to saved documents. This information is vital as it can point to an examiner the name of the potentially fraudulent saved file and where the file was saved to. If the file was deleted or moved, search commands can also be generated based on the names of the files saved. This is done by specifying the name of the file when searching, thereby extending the search filter or search domain during an investigation.

Adobe Photoshop log file records the digital artifacts that indicate saving entries. The same log file, *Adobe Photoshop CSXPrefs*, contains information about the name, location and type of the saved file.

The log file *InDesign SavedData* contains information about the name and type of the file that has been saved, as well as the location to which the file was saved.

In both cases, the names are arranged in order of the last saved file first. From this information the authors managed to obtain the documents created by the graphic design application and recognise the ones which are counterfeit documents.

Adobe Photoshop records both the name of the ‘saving folder’ location and the full name of the saved file. The name of the ‘saving folder’ is recorded in the beginning of the log file, while the entry with the names of the saved files appears towards the middle of the log file. It is noted that the log file records a maximum of 22 entries of saved files. As more files are saved, the log file overwrites the older entries with new entries. Adobe In-Design records an unlimited number of saved documents.

The digital artifacts for saved documents can be verified or compared to the registry entries. Values for the visited directories are acquired from the registry key `HKEY_CURRENT_USER\Software\Adobe\Photoshop\<version #>\VisitedDirs`. Generally, saved files from any graphic design application can also be verified or checked by looking at recent documents available in folder `C:\Users\<username>\AppData\Roaming\Microsoft\Windows\Recent`. 
3.2.4 Artifacts related to document printing

Printing is one of the last stages of counterfeit document creation. A user might need to create a hard copy of the edited document so that it can be used in a physical environment. Unlike scanning actions, printing actions can be commanded from all the graphic design applications in question via the print menu command. The artifacts illustrated in this section are valid for any of the examined graphic design applications. To locate which printer(s) are used to print a document, one uses the registry entries below. The registry keys from which a list of printer connections can be established are the following:

1. HKLM\Adobe\Photoshop 11.0\Plugin path.
2. HKEY_CURRENT_CONFIG\System\CurrentControlSet\Control\Print\Printers
3. HKEY_USERS\<username>\Software\Microsoft\WindowsNT\CurrentVersion\PrinterPorts
4. HKEY_USERS\<username>\Software\Microsoft\Installer\Products\<productid>\SourceList

After establishing the names of the printers from the above, the physical existence of the printers can be verified. This usually assists an investigator in cases where the actual printers have been removed. Physical printers are necessary in an investigation so as to match the digital evidence to the actual printer for supporting a case during court proceedings.

For each print job, two spool files are generated by the operating system located in C:\Windows\System32\spool\PRINTERS. The first is XXXX.shd and the second is XXXX.spl, where XXXX represents the job number in decimal format. Analysing the binary data of these files indicates the name of the spooled document. Additionally, print jobs that were queued to print but have not actually been printed yet can also be found within print spools. Table 1 shows the recognised printing artifacts including examples.

<table>
<thead>
<tr>
<th>Recognised printing artifact</th>
<th>Spool file containing artifact</th>
<th>Address offset for recognised artifact (in HEX)</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of printed document</td>
<td>spl</td>
<td>0X20</td>
<td>Johnstone_passport_final_edit.psd</td>
</tr>
<tr>
<td>Name of printer</td>
<td>shd</td>
<td>0X88</td>
<td>HP Laserjet 2605_2605dnPCL</td>
</tr>
<tr>
<td>Name of printer (repeat)</td>
<td>shd</td>
<td>0X3B0</td>
<td>HP Laserjet 2605_2605dnPCL</td>
</tr>
<tr>
<td>Name of the application that generated the print request</td>
<td>shd</td>
<td>0X2120</td>
<td>Adobe Photoshop CS5</td>
</tr>
<tr>
<td>Username and name of file</td>
<td>shd</td>
<td>0X2400</td>
<td>Robert_graphics_editor. Johnstone_passport_final_edit.psd</td>
</tr>
</tbody>
</table>

The column and row headings for Table 1 are briefly explained for the sake of clarity. Recognised printing artifact is the name of the digital artifact obtained from the stated print spool file (column Spool file containing artifact). Address offset for recognised artifact represents the address pointer in hexadecimal format for the digital artifact, pointing to the named artifact contained in the spool file. Example is an example of a digital artifact for the recognised printing artifact. Name of printer is the address offset where an entry of the name of the printer that generated the print job can be found, and this entry is repeated at another place in the shd spool file as shown in the second column Name of
printer (repeat). The reason for this repetition is not known, however, as far as digital forensic evidence is concerned, the repetition merely confirms again that the printer that was indeed used. *Name of the application that generated the print request* is the offset of the name of the application that generated the print job. *Username and name of file* is the address offset of the name of the user that generated the print job and the name of the printed potential counterfeit document (evidence for printing).

### 3.3 Summary

A log file may consist of thousands of pages of binary data, of which only a few pages will contain the required digital forensic artifacts, which, in addition, may be scattered throughout these few pages. Figure 1 shows an example of an Adobe log file, indicating a path recognised for scanned documents.

One can use a hex editor to scroll, for example, approximately 60% down the log file consisting of thousands of pages to reveal the evidence that is required. This can result in wasting too much time and, ultimately, running the risk that critical evidence being omitted from the search.

![Figure 1 graphic design application log file containing 16980 pages](image)

Another reason for recognising the locations of digital forensic information is that the digital forensic artifacts from the log files do not make use of evidence identifiers such as prefixes and tags. (Evidence identifiers are discussed in the previously mentioned paper by the authors [16]) In other words, the investigator does not know what to search for using keyword searching. The chart presented in this section guide the investigator to look for this evidence at a pre-determined location, for example, about six tenths (or three fifths) down the file. **It is therefore necessary to identify the location of this information by making use of radar chart** in order to pinpoint where the evidence can be found within the log file. Figure 2 illustrates the distribution of the digital forensic artifacts within the Photoshop psp log file.
Figure 2 A graphic illustration of digital artifacts distribution in a Photoshop log file

The chart in Figure 2 shows that the digital forensic artifacts are located mostly in the middle of the log file for any action. In this chart, the centre represents the beginning of the log file represented by a 0 and the outer edges represent the end of the log file represented by a 1. The numbers one to fifty represent the number of counterfeit documents created. Such a chart helps the examiner to appreciate that they can access most of the information at the same location inside a log file. Figure 3 illustrates the distribution of digital forensic artifacts within the log file, *InDesign Save data*.

Figure 3 A graphic illustration of digital artifacts distribution in an Adobe In-Design log file

The radar chart (figure 3) shows that most digital forensic artifacts from the Adobe In-Design log file are located towards the end of the file. Some, however, are scattered all over the file from the beginning until the end. It can be recognised that the radar charts do not contain printing distribution; this is because the printing artifacts outlined in section 3.2.4 are fixed address offsets as displayed in Table 1.

Based on the experiments conducted in this study, the authors managed to establish the locations to which scanned documents were saved. In these locations one could discover several other counterfeit documents that were scanned. In respect of the action of *editing*, the authors established the names, file
types and file locations of inserted objects. By tracking the latter, the actual insertions were recognised by means of fingerprints and human face images inserted into the counterfeit documents. The saving action enabled the researchers to recognise potential digital evidence that reveal the location of the actual counterfeit documents created. The printing action exposed registry and spool files that revealed the names of the printers that had been used for document printing, as well as the names of those documents printed.

4. DISCUSSION

Given that a digital forensic investigation was initiated into a suspected counterfeit document creation crime, and given that the document was generated using a graphic design application, a digital forensic examiner can use the identified digital forensic artifacts to establish the route along which the document was created and corroborate the gathered evidence. For example, the digital forensic examiner is able to discover the human face, fingerprint, and/or bar code images that were used to create the counterfeit document. The inserted image can then be compared to match the image in the suspected counterfeit document. Such evidence can be presented in a court of law for prosecution. Presenting proof of the actions taken during the process of document tampering (scanning, editing, saving and printing) provides valuable support when a case of counterfeit document creation is brought before the court as evidence indicating how the document was created and what entities were used to create the document. In the end, determining if the system was used for counterfeiting purposes.

These results are essential for a digital forensic examiner to find and locate digital evidence related to the creation of counterfeit documents. This increases the transparency and reliability of the investigation process in cases where the crime tool was a graphic design application.

5. CONCLUSION

As mentioned before, that previously-published work, i.e., user-generated digital forensic evidence in graphic design applications [16], involves detecting a counterfeit document directly created by the user. That research lead to another question whether there exist system-generated evidence indirectly created by a system rather than directly created by a user, which then led to this paper, which identifies if a system was used for counterfeiting purposes.

The gathering of system-generated digital forensic evidence is effective in addressing cases where counterfeit document editing is largely associated with particular graphic design applications. Although this approach addresses only case studies involving Adobe products, the same can be done for other graphic design applications and for many other types of applications. A shortcoming of the approach is, however, that it does not tackle issues where the user only edits a hard copy, or scans and prints without using any pre-installed graphic design application. Another drawback of this approach is the fact that this exercise needs to be carried out on all new graphic design applications in order to detect where exactly potential evidence can be found within such a new graphic design application.

The techniques discussed in this paper can, however, be incorporated in commercial digital forensic tools like FTK or Encase, or it can possibly be used in the design of a new digital forensic investigation tool capable of specifically detecting counterfeit document creation. For example, a tool can be created similar to the ‘porn detection stick’ created by Paraben [15], which is a thumb drive device that scans and detects pornographic content on a computer.

Future research can include administering this process to other graphic design applications such as CorelDraw and also to other types of applications that could similarly be used to commit digital document fraud.
REFERENCES


SIGNIFICANCE OF SEMANTIC RECONCILIATION IN DIGITAL FORENSICS

Nickson M. Karie
Department of Computer Science
Kabarak University
Private Bag – 20157
Kabarak, Kenya
menza06@hotmail.com

H. S. Venter
1Department of Computer Science
University of Pretoria
Private Bag X20, Hatfield 0028
Pretoria, South Africa
hventer@cs.up.ac.za

ABSTRACT

Digital forensics (DF) is a growing field that is gaining popularity among many computer professionals, law enforcement agencies and other stakeholders who must always cooperate in this profession. Unfortunately, this has created an environment replete with semantic disparities within the domain that needs to be resolved and/or eliminated. For the purpose of this study, semantic disparity refers to disagreements about the meaning, interpretation, descriptions and the intended use of the same or related data and terminologies. If semantic disparity is not detected and resolved, it may lead to misunderstandings. Even worse, since the people involved may not be from the same neighbourhood, they may not be aware of the existence of the semantic disparities, and probably might not easily realize it.

The aim of this paper, therefore, is to discuss semantic disparity in DF and further elaborates on how to manage it. In addition, this paper also presents the significance of semantic reconciliation in DF. Semantic reconciliation refers to reconciling the meaning (including the interpretations and descriptions) of terminologies and data used in digital forensics. Managing semantic disparities and the significance of semantic reconciliation in digital forensics constitutes the main contributions of this paper.

Keywords: Digital forensics, semantic disparity, managing semantic disparity, semantic reconciliation, significance of semantic reconciliation

1. INTRODUCTION

Digital forensics plays a very important role in both incident detection and digital investigations. However, the investigation process in most cases demands cooperation between the computer professionals, law enforcement agencies and other forensic practitioners. Unfortunately, this has created an environment replete with semantic disparity within the domain that needs to be resolved and/or eliminated. Semantic disparity as defined by Xu and Lee (2002) refers to disagreements about the meaning, interpretation, description and the intended use of the same or related data. Moreover, according to Oxford Dictionaries (2013), disparity refers to the state of being different (lack of uniformity). If semantic disparity is not detected and resolved in digital forensics, it may lead to misunderstandings. In addition, semantic disparity may become a serious problem, for example, when trying to harmonise data/information from different sources (Piasecki, 2008).
Moreover, in the case of a digital forensic investigation process, the cooperation between the computer professionals, law enforcement agencies and other forensic practitioners presupposes the reconciliation of semantic disparities that are bound to occur in the domain. Unfortunately, DF lacks comprehensive methodologies, specifications and ontologies that can assist in resolving the semantic disparities that exist between the different digital forensic practitioners.

In this paper, therefore, we discuss semantic disparities in DF and further elaborate on how to manage it. In addition, this paper also presents the significance of semantic reconciliation in digital forensics. Furthermore, the presentation in this paper is a novel contribution that offers a simplified comprehension of semantic disparities in digital forensics. Moreover, this paper is also meant to spark further discussions on the development of methodologies and specifications for resolving semantic disparities in DF.

As for the remaining part of this paper, section 2 presents background concepts of semantic disparity while section 3 elaborates on how to manage semantic disparities in digital forensics. The significance of semantic reconciliation in digital forensics is handled in section 4. Finally, conclusions and future research work are considered in section 5.

2. BACKGROUND

In this section of the paper, the authors present background concepts on semantic disparities. Note that, semantic disparity as discussed in this paper is sometimes addressed as semantic heterogeneity in other previous research works (Xu and Lee, 2002; Sheth and Larse, 1990; Wang and Liu, 2009). However, for the purpose of this paper we adopt the use of the term semantic disparity in place of semantic heterogeneity.

To begin with, Sheth and Larsen (1990) argue that, semantic disparity is a problem that is not well understood in many domains and in the case of this paper digital forensics as well. There is not even an agreement regarding a clear definition of this problem (Xu and Lee, 2002; Sheth and Larse, 1990). However, different researchers have identified different forms of semantic disparity that are worth mentioning. A majority of these semantic disparities, however, focus more into the field of databases while others focus on distributed systems.

According to Lin et al. (2006), the problem of semantic disparity is extremely critical in situations of extensive cooperation and interoperation between distributed systems across different enterprises. In the case of digital forensics, for example, such a situation would make it difficult to manipulate distributed data/information in a centralized manner. This is because; the contextual requirements and the purpose of the information across the different systems may not be homogeneous.

Another effort by Colomb (1997) presented the case for structural semantic disparity (structural semantics define the relationships between the meanings of terminologies). Bishr (1998) on the other hand, elaborates on schematic disparity. The major problem as presented by Colomb (1997) lies in what can be called the fundamental conceptual disparity. Fundamental conceptual disparity occur when the terms used in two different ontologies, for example, have meanings that are similar, yet not quite the same (Xu and Lee, 2002). Schematic disparity, on the other hand, arises when information that is represented as data in one schema, is represented within the schema (as metadata) in another (Bishr, 1998; Miller, 1998).

Although the database perspective on semantic disparity is good and offers insights (Xu and Lee, 2002), it limits the understanding of semantic disparity and how to manage it in other domains. In the section that follows, therefore, we elaborate on how to manage semantic disparities focusing on the digital forensic domain.
3. MANAGING SEMANTIC DISPARITIES IN DIGITAL FORENSICS

Managing semantic disparities in a growing field like digital forensics can be a daunting task. This is because; the technological trends in DF are ever-changing; new terminologies are constantly introduced into the domain and new meanings assigned to existing terms (Karie and Venter, 2012). Therefore, methodologies and specifications need to be developed in digital forensics with the ability to effectively assist in managing semantic disparities that may crop up as a result of technological change or domain evolution. Such methodologies will further assist in establishing an efficient semantic reconciliation process in the domain. Furthermore, the requirement for semantic reconciliation methodologies and specifications in digital forensics is exceptionally important both for the advancement of the field as well as for the effective use of different domain terminologies and the representation of domain information.

Therefore, understanding the different potential circumstances and conflicts under which semantic disparity may arise in digital forensics can be of great significance in establishing a meaningful semantic reconciliation process.

3.1 Potential Conflicts that can Cause Semantic Disparity in Digital Forensics

Semantic disparity may occur in digital forensics, for example, when the communicating parties (computer professionals, law enforcement agencies, forensic practitioners, etc.) use different meanings, interpretations, descriptions and representations of the same or related domain terminologies and data. This causes variations in the understanding of domain information and how it is specified and structured in different components. This also implies that, perfect communication between the sender and the receiver of the information will be scanty. Having the ability to identify and avoid semantic disparities in digital forensics can assist investigators, for example, in decision making.

In the sub-sections that follow, therefore, we survey and present (based on our review of the literature) various conflicts (including examples where applicable) that can cause disparities in DF. Note that the conflicts discussed in this section only serves as common examples to facilitate this study and should not be treated as an exhaustive list.

3.1.1 Semantic Conflicts

Semantic conflicts occur when different people involved in the same domain do not perceive exactly the same set of real world objects, but instead they visualize overlapping sets (Bishr, 1998). As a result, disagreement about the meaning, interpretation and the descriptions of the same or related data and terminologies occur. Table 1 shows examples of the semantic conflicts (descriptions and interpretation of terminologies) in digital forensics.

<table>
<thead>
<tr>
<th>DF Terminology</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>First response</td>
<td>Include the first response to the detected incident (Valjarevic and Venter, 2012).</td>
</tr>
<tr>
<td>Initial response</td>
<td>Perform an initial investigation, recording the basic details surrounding the incident, assembling the incident response team, and notifying the individuals who need to know about the incident (Mandia et al., 2003).</td>
</tr>
<tr>
<td>Incident response</td>
<td>Consists of the detection and initial, pre-investigation response to a suspected computer crime related incident, such as a breach of computer security. The purpose of Incident response is also to detect, validate, assess, and determine a response strategy for the suspected security incident (Beebe and Clark, 2005).</td>
</tr>
</tbody>
</table>
3.1.2 Descriptive Conflicts

Descriptive conflicts include naming conflicts due to homonyms and synonyms, as well as conflicts on attribute domain, scale, cardinalities, constraints, operations etc. (Bishr, 1998; Sheth and Gala, 1989; Larson et al. 1989). In the case of digital forensics, descriptive conflicts can occur, for example, when two terminologies representing related ideas of the domain concepts are described using different sets of properties. Table 2 present some of the descriptive conflicts identified in the digital forensic domain. Note that the terminologies in Table 1 and Table 2 are only selected examples to facilitate this study and by no means an exhaustive list.

Table 2 Descriptive Conflicts in Digital Forensic Terminologies

<table>
<thead>
<tr>
<th>DF Terminology</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Analysis</td>
<td>Determine significance, reconstruct fragments of data and draw conclusions based on evidence found. The distinction of analysis is that it may not require high technical skills to perform and thus more people can work on this case (Reith et al., 2002).</td>
</tr>
<tr>
<td>• Analysis</td>
<td>Analysis involves the use of a large number of techniques to identify digital evidence, reconstruct the evidence if needed and interpret it, in order to make hypothesis on how the incident occurred, what its exact characteristics are and who is to be held responsible (Valjarevic and Venter, 2012).</td>
</tr>
<tr>
<td>• Analysis</td>
<td>The use of different forensic tools and techniques to make sense of the collected evidence (Sibiya et al., 2012).</td>
</tr>
<tr>
<td>• Examination</td>
<td>Examination is an in-depth analysis of the digital evidence and is the application of digital forensic tools and techniques that are used to gather evidence (Lalla and Flowerday, 2010).</td>
</tr>
<tr>
<td>• Examination</td>
<td>An in-depth systematic search of evidence relating to the suspected crime. This focuses on identifying and locating potential evidence, possibly within unconventional locations. Construct detailed documentation for analysis (Reith et al., 2002).</td>
</tr>
</tbody>
</table>

The authors found that the terminologies in Table 1 and 2 are mostly used by digital forensic investigators and the law enforcement agencies during and after a digital forensic investigation process, hence the motivation for this study.

3.1.3 Structural Conflicts

Structural conflicts occur when two or more people use the same model, but choose different constructs to represent common real-world objects (Lee and Ling, 1995). In the context of digital forensics structural conflicts can occur, for example, when different domain members use the same digital forensic investigation process model but choose different constructs to present their results/findings. Note that, the term constructs, is used to mean ideas or theories containing various conceptual elements, and considered to be subjective but not based on any empirical evidence (Houts and Baldwin, 2004).

After attending several sessions of expert testimony (potential evidence presentation) in court and civil proceedings the authors found that different constructs are used by different digital forensic experts to convince the court that the potential digital evidence presented is worthy of inclusion into the criminal process. However, the constructs used during potential evidence presentation were based on
experience rather than standardised guidelines or digital forensic logics. This is backed up by the fact that, there are currently no standardised guidelines for even presenting the most common representations of potential digital forensic evidence in court or civil proceedings (Cohen, 2011). In the sub-section that follows, we explain different approaches that can assist in managing semantic disparity in DF.

3.2 Different Approaches to Manage Semantic Disparity

There exist different approaches that can assist in resolving semantic disparities in digital forensics (Farshad and Andreas, 2001). However, as with other examples explained earlier, the list discussed in this section present only selected examples and therefore should not be treated as an exhaustive list.

3.2.1 Building Ontologies

Ontologies can help deal with the problem of semantic disparity by providing formal, explicit definitions of data and reasoning over related concepts. Moreover, ontologies in most cases capture the conceptualization of experts in a particular domain of interest (Falbo et al., 1998). Ontology mapping can also be employed to find semantic correspondences between similar elements of different ontologies, thus allowing people to agree on terms that can be used when communicating (Noy, 2004).

In digital forensics, building a proper domain ontology in terms of its explication and its accordance with the conceptualization of domain experts can help in managing the semantic disparity that occurs in the domain. However, according to Kajan (2013), considering that anyone can design ontologies according to his/her own conceptual view of the world, care must be observed during the process of designing ontologies because, ontological disparity among different parties can become an inherent characteristic.

3.2.2 Representation of Ontologies and Reasoning Based on these Ontologies

According to Farshad and Andreas (2001), the representation of ontologies and reasoning based on these ontologies makes it possible to capture and represent ontological definitions and the important features that can be used in representing ontologies for reasoning. In the case of digital forensics such an approach would help create clear definitions of the different terminologies used in the domain. Moreover, this approach can also assist in managing semantic disparity in DF because the relationships that hold among domain terminologies can be realized and structured. For more information in this regard we refer the reader to (Palmer, 2001; Caloyannides, 2004 & Crouch, 2010) respectively.

3.2.3 Semantics Integration

Semantics integration deals with the process of interrelating information from diverse sources to create a homogeneous and uniform semantic of use (Noy, 2004). In the case of digital forensics, this can make communication easier by providing precise concepts that can be used to construct domain information. Furthermore, semantic integration can facilitate or even automate communication between different systems thus offering the ability to automatically link different ontologies (Gardner, 2005).

3.2.4 Explicit use of common shared semantics

The explicit and formal definitions of semantics of terms have always guided many researchers to apply formal ontologies (Guarino, 1998) as a potential solution of semantic disparity. A formal ontology usually consists of logical axioms that convey the meaning of terms for a particular domain (Bishr et al, 1999; Kottman, 1999). Furthermore, formal ontologies are usually concerned with the understanding of the members of the domain and help to reduce ambiguity in communication (Farshad and Andreas, 2001), understanding, representation and interpretations of information.

In the next section, we present the significance of semantic reconciliation in digital forensics.
4. SIGNIFICANCE OF SEMANTIC RECONCILIATION IN DIGITAL FORENSICS

While there are a lot of research activities in digital forensics even at the time of this study very little have been towards semantic reconciliation. The authors believe that, semantic disparity in any domain can alter the context as well as the purpose of any information delivered by an individual and thus should to be avoided. In digital forensics, methodologies and specifications need to be developed that can effectively assist in semantic reconciliation. Furthermore, such methodologies and specifications can also be used, for example, as fundamental building blocks in resolving the present and future semantic disparities in the domain. Semantic reconciliation, in the authors’ opinion, is a promising conception towards resolving semantic disparities in digital forensics. The sub-sections that follow will explain in more details some of the significances of semantic reconciliation in digital forensics.

4.1 Perfect Communication

Semantic disparities can be a serious barrier to perfect communication in any domain. Semantic reconciliation, on the other hand, can be used to bridge the semantic gap between different communicating parties thus bringing with it perfect communication in the domain (Parsons and Wand, 2003). This also implies that, information between the different digital forensic stakeholders (computer professionals, law enforcement agencies and other digital forensic practitioners) can be interpreted in such a way that the sender's desired effect is achieved. Moreover, after a security incident has occurred, for example, if the communication, interpretation and representation of information are done correctly, it is much easier and useful in apprehending the attacker, and stands a much greater chance of being admissible in the event of a prosecution (Brezinski and Killalea, 2002). Wrong interpretation and representation of evidence information, on the other hand, might create loopholes for intruders to escape and thus making it had to convict and prosecute them. Therefore, semantic reconciliation in digital forensics is inevitable if perfect communication is to be achieved.

4.2 Common Understanding

Semantic disparities may arise in digital forensics as a result of different representation or interpretation of terminologies and data; this may include the use of different alternatives or definitions to describe the same domain information. However, with semantic reconciliation the different digital forensic experts can achieve common understanding by reconciling the meaning of terms thus having common representation or interpretation of domain terminologies (Parsons and Wand, 2003). This also implies that, the meaning of information as interpreted by the receiver will align with the meaning intended by the sender (Anon, 2013). In the case of court or civil proceedings common understanding will also help different stakeholders treat queries conveniently and at the same time maintaining consistency in their understanding of the various digital forensic terminologies and data used during such proceedings.

4.3 Correct Interpretation

When two or more independent digital forensic practitioners with varying professional backgrounds are to cooperate during an investigation process, semantic conflicts may occur. It is, therefore, very important and critical that semantic disparities be resolved and/or eliminated to facilitate correct interpretation of domain information. Semantic reconciliation is one of the ways that can improve on correct interpretation through detecting the semantic similarities between the different terminologies and data used by the independent practitioners to describe or represent domain information (Parsons and Wand, 2003).

4.4 High-levels of collaboration

Many organisations are increasingly promoting collaborations as an important feature in organisation management (Tschannen-Moran, 2001). However, effective collaborations demands reasoning as well as effective communication. Therefore, semantic reconciliation in digital forensics can lead to high-
levels of collaborations between the computer professionals, law enforcement agencies and other
digital forensic practitioners. Furthermore, semantic reconciliation can also help create uniformity in
the use of both terminologies and data in the digital forensic domain thus easing cooperation.

4.5 Uniform Representation of Domain Information
In the case of potential evidence presentation in any court of law, information conveyed with very
many semantic variances can be semantically unreliable. Therefore, semantic reconciliation can help
create uniform representation of domain information. This is backed up by the fact that, semantic
reconciliation can also make interpretation and representation of domain information much easier and
more accurate (Wang et al., 2005).

4.6 Faster Harmonisation of Information from Different Sources
Efficient information management and processing have become more and more important within
enterprises or when enterprises are merging together (Ubbo et al. 2002). Moreover, to achieve
semantic interoperability across information system using different terminologies, the meaning of the
information that is interchanged has to be harmonised across the systems (Ubbo et al. 2002). However,
semantic disparity may arise whenever two contexts do not use uniform interpretation of the same
information. Therefore, the use of semantic reconciliation for the explication of implicit and hidden
knowledge is a promising approach to overcome the problem of semantic disparity in digital forensics
and can assist in faster harmonisation of information from different sources.

4.7 Less Errors during Analysis of Potential Digital Evidence Information
Errors in analysis and interpretation of digital evidence, in the case of an investigation process, are
more likely where there are semantic disparities. Even more where there are no standardised
procedures or formal representation of domain information (Chaikin, 2006). Semantic reconciliation,
on the other hand, will enable computer professionals, law enforcement agencies and practitioners in
digital forensics to agree on terminologies or keywords to be used in representing certain key
information in the case of an investigation and also establish keyword structures so that their
relationship to each other are easily known. This will enhance the analysis of potential digital evidence
information in the domain.

5. CONCLUSION AND FUTURE WORK
The problem addressed in this paper was that of semantic disparity in digital forensics. Different
approaches to manage semantic disparities in digital forensics have also been explained. Moreover, the
paper has also elaborated on the significance of semantic reconciliation in the digital forensic domain.
The presentation in this paper is a new contribution in digital forensics and is meant to spark further
discussion on the development of methodologies and specifications for semantic reconciliation in the
domain. As part of the future work, the authors are now engaged in a research project to try and
develop specification and/or ontologies that will create a unified formal representation of the digital
forensic domain knowledge and information. In addition, the authors also aim at developing a digital
forensic semantic reconciliatory model as a way towards resolving the semantic disparities that occur
in digital forensics. However, there is still much research to be carried out so as to provide directions
on how to address semantic disparities in the digital forensic domain. More research also needs to be
conducted in order to add on the work discussed in this paper.

REFERENCES
17.HTML on April 25, 2013.

investigations process. Digital Investigation, 2(2), 146-16.


Kajan, E. (2013). Electronic business interoperability: Concepts, opportunities and challenges - Google Books. Retrieved from [http://books.google.co.za/books?id=fNh2Frj7oUC&pg=PA287&lpg=PA287&dq=Even+if+two+ontologies+use+the+same+name+for+a+concept%2c+the+associated+properties+and+the+relationships+with+other+concepts+are+most+likely+to+differ&source=bl&ots=Q8q74gKNzP&sig=vxThC1hAOQ7miwqAhxUJPn+i&hl=en&sa=X&ei=gNUwUeiuMI-LhQdloDYBQ&ved=0CDYQ6AEwAg#v=onepage&q=Even%20if%20two%20ontologies%20use%20the%20same%20name%20for%20%20a%20concept%20and%20the%20associated%20properties%20and%20the%20relationships%20with%20other%20concepts%20are%20most%20likely%20to%20differ&f=false](http://books.google.co.za/books?id=fNh2Frj7oUC&pg=PA287&lpg=PA287&dq=Even+if+two+ontologies+use+the+same+name+for+a+concept%2c+the+associated+properties+and+the+relationships+with+other+concepts+are+most+likely+to+differ&source=bl&ots=Q8q74gKNzP&sig=vxThC1hAOQ7miwqAhxUJPn+i&hl=en&sa=X&ei=gNUwUeiuMI-LhQdloDYBQ&ved=0CDYQ6AEwAg#v=onepage&q=Even%20if%20two%20ontologies%20use%20the%20same%20name%20for%20%20a%20concept%20and%20the%20associated%20properties%20and%20the%20relationships%20with%20other%20concepts%20are%20most%20likely%20to%20differ&f=false) on March 1, 2013.


78


KEYNOTE SPEAKER

Daniel J. Ryan, Esq.
danryan@danjryan.com

BIOGRAPHY

Daniel J. Ryan is a lawyer in private practice, a businessman providing analyses of policy issues significant to national security, and an educator teaching cyberlaw and information security as a Senior Fellow at the National Defense University and as an adjunct professor for George Washington University. Prior to returning to the private sector in 2012, he served for eight years as a Professor at the National Defense University, teaching cyberlaw, information security, information assurance, cryptography, network security and computer forensics. Prior to joining NDU, he was a lawyer in private practice.

Earlier he served as Corporate Vice President of Science Applications International Corporation with responsibility for information security for Government customers and commercial clients who operate worldwide and must create, store, process and communicate sensitive information and engage in electronic commerce. While at SAIC, he developed and provided security products and services for use in assessing security capabilities and limitations of client systems and networks, designing or re-engineering client systems and networks to ensure security, enhancing protection through a balanced mix of security technologies, detecting intrusions or abuses, and reacting effectively to attacks to prevent or limit damage.

Prior to joining SAIC, Mr. Ryan served as Executive Assistant to the Director of Central Intelligence. Before that, he was Director of Information Systems Security for the Office of the Secretary of Defense serving as the principal technical advisor for all aspects of information security. He developed information security policy for the Department of Defense and managed the creation, operation and maintenance of secure computers, systems and networks. His specific areas of responsibility spanned information systems security (INFOSEC), including classification management, communications security (COMSEC) and cryptography, computer security (COMPUSEC) and transmission security (TRANSEC), as well as TEMPEST, technical security countermeasures (TSCM), operational security (OPSEC), port security, overflight security and counterimagery.

In private industry, he was a Principal at Booz•Allen & Hamilton where he served as a consultant in the areas of strategic planning, system operations and coordination, data processing and telecommunications. At Bolt Beranek & Newman, he supplied secure wide-area telecommunications networks to Government and commercial clients. He represented the Los Angeles-based Systems Development Division of TRW in Washington, D.C., and he was Director of Electronic Warfare Advanced Programs at Litton's AMECOM Division. He headed a systems engineering section at Hughes Aircraft Company where he was responsible for the design, development and implementation of data processing systems.

He began his career at the National Security Agency as a cryptologic mathematician.

Mr. Ryan received his Bachelor’s degree in Mathematics from Tulane University, a Master’s in Mathematics from the University of Maryland, a Master of Business Administration degree from California State University and the degree of Juris Doctor from the University of Maryland. He is admitted to the Bar in the State of Maryland and the District of Columbia, and has been admitted to practice in the United States District Court, the United States Tax Court, and the Supreme Court of the United States. He has been Certified by the United States Government as a Professional in the fields of Data Systems Analysis, Mathematics and Cryptologic Mathematics.
INVESTIGATING THE PROFESSIONALIZATION OF THE FIELD OF DIGITAL FORENSICS

ABSTRACT

Just as other professions have moved from a guild approach to educating and regulating those who claim expertise to formal credentialing of professionals, digital forensics has now reached a stage where credentialing is necessary. Organizations such as the American Academy of Forensic Sciences are exploring the requirements for digital forensics certifications. Some jurisdictions in the United States are relying on a licensing approach, requiring private investigator licenses for digital forensics practitioners. Other organizations in the States and globally are developing certifications at various levels of technical expertise.

Just as the nature of digital information has expanded and evolved, so have the requirements for digital forensics knowledge. Today’s professional must understand far more than simple hard drive analysis. Consequently, digital forensics certifications must assure that recipients have the necessary breadth and depth of knowledge and thinking skills needed to address today’s cyber forensics challenges.

There is a natural association between cyber security professionals and cyber forensics professionals. (ISC)², the organization that manages the CISSP cyber security credential, has developed a new credential, called the Certified Cyber Forensics Professional (CCFP), to meet the growing demand to distinguish digital forensics professionals by validating their knowledge, skills and abilities in forensic techniques and procedures, standards of practice, and legal and ethical principles that assure the accuracy, completeness and reliability of digital evidence.

Attendees to this session will gain:

Insight into how digital forensics and overall cyber security intersect

- Insight into the benefits and requirements of the credentialing of digital forensics professionals
- Knowledge of the digital forensics professional marketplace – what the training and certification landscape looks like, and how the CCFP credential fits
- Why professionalization of the field is critical to enhancing the security posture of governments and organizations
AN IMAGE FORENSIC SCHEME WITH ROBUST AND FRAGILE WATERMARKING FOR BUSINESS DOCUMENTS

(Briefing Paper/Presentation)

Sai Ho KWOK
Department of Information Systems
Business Statistics and Operations Management
The Hong Kong University of Science and Technology
Clear Water Bay, N.T.
Hong Kong SAR
Phone: (852) 2358 7652
Fax: (852) 2358 2421
Email: jkwok@ust.hk

Keywords: Image Forensics, Fragile and Robust Watermarking, Business Document.

ABSTRACT

This paper proposes an image forensic scheme with both robust and fragile watermarking techniques for business documents. Through a dual watermarking approach, the proposed scheme can achieve image forensics objectives of (a) identification of source; (b) authentication of documents; and (c) locating the tempered areas of documents due to attacks. An example is presented to prove the concepts of the proposed scheme.

1. INTRODUCTION

In business, digital documents include legal documents, official reports, contracts, agreements and so on. Currently, most of business documents are in MS Word and PDF formats and they contain both text and graphs. In this paper, business documents are represented as digital images technically. Digital forensic becomes crucial when these business documents are treated as evidence of a crime or attack. In the private sector, digital forensic is required during internal corporate investigations or intrusion investigation.

In general, digital forensic can be used to attribute evidence to specific suspects, confirm alibis or statements, determine intent, identify sources, or authenticate documents (Wikipedia, 2013). Zhou et al. (Zhou et al., 2011) point out that digital image forensics can be used to (a) judge whether an image is from a particular digital camera; (b) determine whether an image was produced by the same type of cameras or other equipment or software; (c) to determine whether an image has been processed (or attacked); and (d) to determine whether an image is the original one or not.

Digital watermarking is a key process in active image forensics (Xi et al., 2011; Zhou et al., 2011). It is a process of embedding relevant information (such as a logo, fingerprint, and serial number) into a digital content (Kung et al., 2003; Lin et al., 2001; Xi et al., 2011). The embedded data can be viewed as digital watermarks. There are two types of digital watermarks and they are visible (or fragile) and invisible (or robust) watermarks.

A visible or fragile watermark is the translucent logos that often appear at the corner of images, in an attempt to prevent copyright infringement. Such watermarking process operates in the spatial domain, where the corresponding pixel values are modified directly or indirectly (Kutter et al., 1998; Voyatzis et al., 1997). However, these visible watermarks can be targeted and removed rather simply by cropping the image, or overwriting the logos (Xi et al., 2011). A visible or fragile watermark is used for content authentication applications to verify or authenticate the integrity of digital documents.
An invisible or robust watermark can often resist intentional and unintentional attacks to the images. Common attacks are often referred to image processing, such as compression, rotation, filtering, zooming and so on (Kung et al., 2009). The robust watermarking usually operates in the transform domain (Briassouli et al., 2004; Lin et al., 2008; Lu et al., 2009; Tsui et al., 2008; Zou et al., 2008). In other words, robust watermarking techniques can embed data in the transform domain, such as frequency domain through Discrete Fourier Transform (DFT) (Tsui et al., 2008; Zou et al., 2008), Discrete Cosine Transform (DCT), and Discrete Wavelet Transform (DWT) (Lin et al., 2008; Lu et al., 2009). A robust watermark can be used to determine the source of the image and copyright protection (Zhou et al., 2011).

The objectives of digital image forensics for business documents include (a) identification of source; (b) authentication of documents; and (c) locating the tempered areas of documents due to attacks. Due to the nature of business documents, which are different from ordinary images, image forensics techniques for business documents could be quite different from those techniques in the literature of digital image forensics. This paper intends to fill this gap and proposes an image forensics scheme for business documents specifically. The proposed scheme falls into the category of active image forensics as digital watermarking is applied after the documents have been produced. The proposed scheme adopts a dual watermarking process, which is a rather innovative and promising approach in tackling image forensics problems (Zhou et al., 2011).

2. PROPOSED IMAGE FORENSICS SCHEME

The following are the terms used in the proposed image forensics scheme.

- A business document, \(D\) is treated as an 8-bit gray-scaled image, where its pixel values ranging from 0 to 255.
- A fragile watermark, \(W\) is a pre-defined watermark specifically for an organization, for example a company logo or trademark. This is also an 8-bit gray-scaled image.
- An authentication data, \(AD\) is data for identifying a particular person or computer in an organization. It can be a staff ID, an IP address of a computer, and so on. In digital watermarking, it is commonly known as the Key of a watermarking process.
- The embedded watermark, \(W_f\) is the watermark to be embedded into \(D\). It is a fragile watermark and is produced by a robust watermarking process, whose inputs are \(W\) and \(AD\).
- A watermarked document, \(D_w\) is an 8-bit gray-scaled image. It is resulted from a fragile watermark insertion process, whose inputs are \(D\) and \(W_f\). A simple implementation can be \(D_w = D + W_f\).
- An evidence of business document, also known as a modified watermarked document is denoted as \(D_w'\), which a watermarked document picked up by the police force or internal/external audit for examination. \(D_w'\) can be (i) exactly the same as \(D_w\), so the content of the document is identical to the original document, or (ii) a modified version of the original document, where the content of the document has been modified by attacks.

2.1 Watermark Insertion

In the proposed scheme, the watermark insertion process as depicted in Figure 1 is responsible for inserting a digital watermark, \(W_f\) into a business document, \(D\). The watermark, \(W_f\) is applied to all pages of the document. The process consists of two different watermarking techniques: a robust watermarking technique and a fragile watermarking technique. It is also known as dual watermarking technique. The primary objective of the robust watermarking process is to incorporate authentication data, \(AD\) into a fragile watermark, \(W\). It results in another fragile watermark, \(W_f\) for inserting to the business document, \(D\).
The fragile watermark insertion process operates at the spatial domain. A simple implementation of the process can be an additional operator, which can be represented as follows.

\[ D_w = D + W_f \]

2.2 Watermark Verification

The proposed watermark verification process is depicted in Figure 2. The objective of the watermark verification process is to verify whether the embedded watermark, \( W_f \), has been modified or not. If \( W_f \) is modified, the process is able to locate the tempered area. As fraudulent documents are usually very much similar to the original documents, except several major parts of the document, it is expected that \( D_w' \) should be very much similar to \( D_w \). Based on this assumption, authentication can be achieved through the robust watermarking process.
2.3 Proof or Concepts

Assume the original business document, \( D \) contains only a letter “H”. Figure 3 shows all involving images during the watermark insertion process for the business document. The resulting watermarked document, \( D_w \) is produced. Then the watermark verification process is performed as \( D_w' - W_f \).

Figure 3 The watermark insertion process with an input of the letter “H”

Assume a digital evidence \( D_w' \) is presented for examination. The content of the business document has been changed from the letter “H” to the letter “I”. Figure 4 illustrates the watermark verification process with \( D_w' \). Figure 5 presents the result of the watermark verification when the \( D_w' \) is derived from a business document with the letter “I” without any changes.

By comparing the results of Figures 4 and 5, it is shown that the watermark verification process can distinguish between a changed document and an unchanged document.
Figure 4 The watermark verification process with $D_w$ of the letter “I”

Figure 5 The result of the watermark verification process with the $D_w$ derived from the letter “I”

3. CONCLUSIONS

The proposed image forensics scheme can achieve the following major objectives of image forensics.

a) Identifying the source of business documents as a fragile watermark, $W$ of company logo is used in the scheme;
b) Authenticating the documents as authentication data, $AD$ are embedded into the documents through watermarking; and
c) Locating the tempered areas through the watermark verification process.

The following are the remaining problems with the proposed scheme.

- The choice of robust and fragile watermarking techniques: The objectives of the watermarking techniques in the proposed scheme have been specified, but which particular watermarking techniques can provide the best performance are under reviews.
• The choice of the file format of watermarked business documents: The file format is likely to be PDF as this has been wide used by many organizations but further research is needed.
• The design of fragile watermark: In the example, a black-and-white fragile watermark was used but it is likely to be a gray-scaled image in practice. Further investigations are required for the design of fragile watermarks, which is likely to be adopted by organizations.

The proposed scheme is still under development and further tests are required to evaluate the performance and applicability of the scheme in business.

REFERENCES


JOURNEY INTO WINDOWS 8 RECOVERY ARTIFACTS

W. Kenneth Johnson
KPMG
USA
patories@gmail.com

Keywords: Windows 8, Digital Forensics, Recover Options, System Reset, System Refresh, File History

ABSTRACT

One of the most difficult processes of digital forensics is to understand how new technology interacts with current technology and how digital forensic analysts can utilize current Digital Forensics technologies and processes to recover and find information hidden. Microsoft has released their new operating system Windows 8, with this new release Microsoft has added some features to the operating system that will present some interesting complications to digital forensics.

Since the initial release of the Windows 8 Release Candidates there have been some research released that focus primarily on the new user created artifacts and a few artifacts that have been added by the operating system that might contain valuable information. This paper will look at the new recovery options that have been introduced in the final release of the Windows 8, and the impact that have on the artifacts.

This paper will investigate the impact on system and user artifacts when the Windows 8 recovery methods are used. This paper will look the artifacts that are created between the different recovery methods, as well as what artifacts can be recovered from the hard drive after a recovery method has been used.

1. INTRODUCTION

One of the most difficult processes of digital forensics is understanding how new technology interacts with current technology and how we can utilize current Digital Forensics technologies and processes to recover and find information hidden. Microsoft has released their new Operating System Windows 8, with this new release Microsoft has added some features to the Operating System that will present some interesting complications to digital forensics.

Over the last year there has been plenty of research released that focus on the new user created artifacts and a few artifacts that have been added by the operating system that might contain valuable information. This paper will cover the new recovery options that have been introduced in Windows 8, and the impact that have on the artifacts.

The first thing that this paper will cover is the artifacts discovered by the research of Amanda Thomson. Once these artifacts have been analyzed and verified the locations on the disk, a baseline dataset will be created to compare the impact of the recovery options on these artifacts. This baseline dataset will be used for finding artifacts on the system after a recovery option has been used.

The final thing that this paper will cover is how the various recovery options impact the artifacts that are found on the operating system. This will be done by installing Windows 8 in a Virtual Machine environment and taking snapshots of a base image and then utilizing the various recovery methods. Once the recovery method has been successful ran, analysis of the Virtual Machine will be done by importing the image into FTK Imager.
2. GATHERING OF INFORMATION

2.1 User Created Artifacts

Amanda Thomson released documentation on various artifacts that the new Metro interface of the Windows 8 operating system creates (Thomson, Propellerhead Forensics, 2012). This paper will focus on the artifacts that are contained in the communication apps, internet explorer 10, File History and user created documents. This will create a baseline of data to search for and extract from.

2.2 Volume Shadow Copy Services

Within the Microsoft Windows 8 Operating System, Microsoft has enhanced the Volume Shadow Copy Services to be utilized by more than just System Restore Points. This service is also now the backbone of the File History Service and is also utilized with the system refresh. This paper will look at the integration of the Volume Shadow Copy services and what artifacts will be useful for analysis. This paper will examine how these artifacts are impacted and used by the various recovery methods (Microsoft, n.d.).

2.3 System Restore Points

System Restore Points are created three different ways within Windows 8. Like previous versions of Windows these can be created via System Initiated process, and user initiated process. Within Windows 8, two new registry values have been created that allows applications to initiate the request for systems restore point creation. This paper will utilize the Communication Application artifact discovered from Thomson’s research as well as research of my own to compare between what was expected and how it can be recovered. As with previous versions of windows previous version copies are stored in the Volume Shadow Copies and can be recovered by mounting the drive and extracting the data.

2.4 System Refresh

Windows 8 has introduced the ability for users to recover from malware infection or stability issues by including the refresh option in the operating system. There are two options with the System Refresh that users can utilize. The first option is a default refresh which will revert the operating system back to a factory default setting and a custom refresh that allows the user to define the snapshot scope to revert back to. In gathering information on what artifacts are impacted and retained from utilizing the System Refresh this paper will analyze both the Custom and Default against my baseline and compare the difference. This paper will also identify new artifacts that are created on the hard drive from this process.

2.5 System Recovery

Windows 8 has introduced the ability for users to quickly reinstall the Operating System from a GUI for the user. The system recovery offers a few options for reinstallation; these options are Quick and Thorough Recovery. The differences between these two options are the ease of which data can be recovered. The quick recovery still allows for easy extraction of files on the machine; while the thorough recovery makes data recovery difficult. This paper will look at the impact on the baseline dataset artifacts when the recovery options are utilized and will also indicate new artifacts that are created when these are run.

3. FILE HISTORY SERVICES

Within the Microsoft Windows 8 Operating System, they have introduced file history backup, which changes the way backups were previously used. In previous versions windows could only maintain and restore backups using the default system. With Windows 8, Microsoft has implemented a solution that is more robust and allows backups to be stored both on removable media and remote network
shares. By default File History will back up the following folders: Music, Documents, Videos, Pictures, Desktop, Contacts and Favorites (Serban, n.d.).

There are a few artifacts that are established when the File History is turned on these include the file history folder and registry values. The file history folder can be found in the following path: `C:\Users\<USERID>\AppData\Local\Microsoft\Windows\Filehistory`

This directory is also written to the backup location. Within that directory there are two folders named Data, and Configuration. The data folder contains the files and folders that are tagged for backup using the file history. The configuration folder contains files that are both EDB and XML files. File names for the EDB follow the naming conventions of Catalog#.edb, and file names for the XML files are Config#.

<table>
<thead>
<tr>
<th>File History Config Values</th>
<th>What does it mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserName</td>
<td>The user account that this is configured for.</td>
</tr>
<tr>
<td>Friendly Name</td>
<td>First/Lastname tied to account. Inherited from Windows LiveID if configured</td>
</tr>
<tr>
<td>PC Name</td>
<td>Name of the computer</td>
</tr>
<tr>
<td>UserType</td>
<td>What type of user this is</td>
</tr>
<tr>
<td>Library</td>
<td>Directories that are being backed up</td>
</tr>
<tr>
<td>UserFolder</td>
<td>User Specific Directories being backed up. Sub Value of Library</td>
</tr>
<tr>
<td>LocalCatalogPath</td>
<td>Filepath to the Catalog.edb and the config files</td>
</tr>
<tr>
<td>Retention Policy</td>
<td>How long data is retained</td>
</tr>
<tr>
<td>DPFrequency</td>
<td>How often is the data backed up</td>
</tr>
<tr>
<td>Target</td>
<td>The backup location</td>
</tr>
<tr>
<td>TargetName</td>
<td>Name of the Location used for backup</td>
</tr>
<tr>
<td>TargetDriveType</td>
<td>The options Local, Remote, and Removable are for the drive type that the backup is sent to.</td>
</tr>
</tbody>
</table>

If the File History option has been turned on, there will be registry keys created in the HKU keys of the users that have this option turned on. This key can be found in the `Software\Microsoft\Windows\CurrentVersion\FileHistory`. Within this directory there is a key named ProtectedUpToTime which is a 64 Bit Hex Value – Big Endian, which can be deciphered by utilizing the DCode application. In Figure 1 the FileTime value is seen, in Figure 2 the value has been converted in the application DCode, this time represents the last time an update was pushed to the file history system.
Another area in the registry that may contain keys of importance is HKLM\System\Controlset001\Services\fhsvc. Within this Key, there is a parameter key that shows the location of the configurations values.

Another area to look at in gathering File History information is within the System Events. The following Event Sources provide us with information related to the File History: FileHistory-Catalog, FileHistory-ConfigManager, FileHistory-Core, FileHistory-Engine, FileHistory-EventListener, and FileHistory-Service. As of this research the event logs being parsed are the ones that show errors, and one that fires off with each successful backup, but claims something is missing and can’t be parsed. Until this operating system is further along, this issue might remain.

You can also utilize the Jump Lists for the File History to gather more information on it. I was able to pull from the file history jump list, the various drives I used for my backup locations. This will be beneficial if the user modifies their backup location.

If a Windows 8 machine has the File History Service turned on, it will persist over a system Refresh. These files can be found in the original directory.

4. COMMUNICATION APPLICATION

Some of the more useful artifacts that are included with Windows 8 as the Metro App known as the Communication App. Amanda Thomson went into great details in her research on the artifacts that can be recovered. She showed that analysts are able to extract artifacts that will show who and how a user interacts with various online organizations and people (Thomson, Windows 8 Forensic Guide, 2012).

From within the Communication App there are few locations that will provide artifacts for analysis; Thomson touched on the Cache and the Mail locations. This paper will discuss the following artifacts that I discovered in the application; AddressBook and Me directories. While there may be more of value, these two locations open up more insight on the users contacts and the accounts being used by the user.

From the cache files analysts are able to extract the user online contacts details. These details include email, twitter handle, profile pictures, and pictures that were shared by the users contact. From the cookies files analysts can extract conversation, email, email attachments, twitter communications and other communication transactions between users.

From the cookies files analysts are able to extract user messages that have appeared in the communication app. Thomson’s research showed how email messages appeared in the cookies directory, this example shows how a Twitter stream would appear in the cookies directory. In this example we can see the username that posted to tweet, the content of the tweet and the associated url that was included in the tweet.
From the mail files analysts can recover information about the emails the sender has sent, received, stored or even ads that have populated the inbox of the live account tied to the communication app. According to Thomson’s research the file path to the Mail directory is the user’s windows live account. This is no longer the case as it is now a random string. I have verified that by having the same windows live account across multiple machines and this 16 alpha numeric character directory does not share the same name across the machines.

Within the Mail directory are subdirectories that hold various files. The subdirectories on my machines met the following naming standards 1d0000# while in Thomson’s research these values were 120000#, it appears that this naming standard follows the similar hex pattern, although at this time it is unknown what the meaning of the pattern is. The files in the subdirectories in my testing followed the naming standard of:

2000000#_###############.eml.OECustomProperty

(14 alpha numeric characters)
The different subdirectories under the mail directory appear to be different directories in the Windows Live email system. In my case directory 1d000002 appears to be my inbox, the directory 1d000004 appears to have been my sent folder, and the 1d00000b appears to be my draft folder.

From the AddressBook file analysts are able to gather username of the contacts. All entries will contain a From field which will list the Contact name, at the end of the entry there will also be a list of all alias’s tied to that account. These alias can be email usernames, full names, first name, last name or even another screen name.

For a contact with an associated email the entry will contain a Subject line that will have the value HasEmail if this account is tied to an email address. There will be a unique Hex String for each email address associated with the contact in the TO: field.

For a Twitter contact the two fields at the bottom of the artifact will list the screen name first and the first and last name that was entered into the product. Since this is a twitter account and no email is associated with it there is no subject that shows the HasEmail value.

For a Facebook contact the artifact will contain the full name in the from field, and at the end of the artifact the first line will again be the full name, the second line will be the first name associated with the account and the last line will be the last name associated with the account.
The Me folder will contain an artifact entry that will contain all the accounts that the user has connected to the communication app. This artifact follows the same structure as previous examples; containing the hex string of the email address, user name and the associated full name. As shows in figure X, I used a Hex to ASCII converter to change my stored email from Hex to ASCII.

In the table below is an update to the artifact list which includes other artifacts be of interest to an analyst. I have included the location of the Cache and Mail artifacts as well as introducing the Addressbook and the ME locations.

<table>
<thead>
<tr>
<th>Artifact Type</th>
<th>Artifact Location</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cache</td>
<td><code>%Root\%User\%AppData\Local\Packages\microsoft.windowscommunicatisapps_8wek\yb3d8bbwe\AC\INetCache</code></td>
<td>Contains contacts email, screen name, or images the user has viewed.</td>
</tr>
<tr>
<td>Cookies</td>
<td><code>%Root\%User\%AppData\Local\Packages\microsoft.windowscommunicatisapps_8wek\yb3d8bbwe\AC\INetCookies</code></td>
<td>Copy of messages that have shown up in the Communication App.</td>
</tr>
<tr>
<td>Mail</td>
<td><code>%Root\%User\%AppData\Local\Packages\microsoft.windowscommunicatisapps_8wek\yb3d8bbwe\LocalState\Indexed\LiveComm\%randomString\%randomString\Mail</code></td>
<td>Copy of users emails, these will contain sender, recipient, subject, body and attachments.</td>
</tr>
<tr>
<td>Address Book</td>
<td><code>%Root\%User\%AppData\Local\Packages\microsoft.windowscommunicatisapps_8wek\yb3d8bbwe\LocalState\Indexed\LiveComm\%randomString\%randomString\People\Address</code></td>
<td>Contains username and screen name of contacts. If account has email address then email address is also stored in hex value.</td>
</tr>
<tr>
<td>ME</td>
<td><code>%Root\%User\%AppData\Local\Packages\microsoft.windowscommunicatisapps_8wek\yb3d8bbwe\LocalState\Indexed\LiveComm\%randomString\%randomString\People\Me</code></td>
<td>Contains username and screen name of users accounts and all associated email addresses is also stored in hex value.</td>
</tr>
</tbody>
</table>
5. WINDOWS 8 RECOVERY ARTIFACTS

5.1 System Restore Points

Windows 8 maintains the traditional Restore Points that have been seen in previous versions of windows with a few new tweaks. System Restore points automatically monitors files in the boot volume that are relevant for restore only, this is incompatible with previous versions. It allows users to undo a change that may have caused a problem with the system, or to revert to a day when the system might was preforming optimally (Microsoft, n.d.).

Within Windows 8 a new registry key was added that enables application developers to change the frequency of restore-point creation. If the key does not exist then when an application calls the SRSetRestorePoint function to create a restore point, Windows skips creating this new restore point if any restore points have been created in the last 24 hours (Microsoft, n.d.).

With this new registry key, developers can write applications that create the DWORD value SystemRestorePointCreationFrequency under the registry key HKLM\Software\Microsoft\Windows NT\CurrentVersion\SystemRestore. The value of this registry key can change the frequency of the restore point creation.

When the application calls SRSetRestorPoint to create a restore point, and the registry key value is 0, system restore does not skip creating a new restore point. If the application tries to create a restore point, and the registry value is the integer N, than system restore skips creating a new restore point if any restore points were created in the previous N minutes.

Developers can write applications that create the DWORD value ScopeSnapshots under the registry key HKLM\Software\Microsoft\Windows NT\CurrentVersion\SystemRestore. If this value is 0, System Restore creates snapshots of the boot volume in the same way as earlier versions of Windows. If this value is deleted, System Restore running on Windows 8 resumes creating snapshots that monitor files in the boot volume that are relevant for the system restore only.

5.2 System Refresh Points

Windows 8 introduces two new options for system recovery, these options are: Refresh Points and System Recovery. Within Refresh Point there are two options; you can utilize the default refresh point or a custom refresh point.

Both Refresh options can be utilized by Windows 8 to remove malicious files and corrupted entries into the operating system. When using Refresh it is important to understand that the operating system creates a Recovery Image that makes a backup of the Windows System Files. For the default recover these Windows System Files are from when Windows 8 was first installed. When the Custom Refresh option is used than the Windows System Files are from the date that the Custom Refresh was created, the Custom Refresh also will contain the desktop applications that you have installed. Refresh Images DO NOT contain your Metro-style apps, documents, personal settings or user profiles, this is because that information is preserved at the time you refresh your PC.

When looking at an image of the Windows 8 operating system from within the AccessData FTK Imager there are three things that are quickly noticed. There are two partitions and an unallocated space. Figure 11 shows this layout, this is different from previous versions because of the Partition 1.
The Recovery Partition comes up as Partition 1 in FTK Imager and is a 350MB partition. This partition is dedicated to the basic root of the operating system. This will contain information related to the refresh process of the machine. If a Refresh or Reset has been utilized there will be a new subdirectory called Logs, which will contain a file called Reload.xml.

Figure 12 shows what the recovery partition looks like before a Refresh or Reset has been utilized. Figure 13 shows what the recovery partition looks like after a Refresh or Reset has been utilized. By looking at the two figures the difference can be seen with the addition of the LOGS subdirectory under the Recovery Directory. Within the LOGS directory there will be at least one file which will be called ReLoad.xml. There is potential to be other files only if a Refresh or Reset option has encountered any errors in the process.

Within the subdirectories of Logs and the b1d8602f-6346-11e1-b08c4f298f72d4a8 there will be artifacts that can indicate if a Refresh or Reset has occurred. The Logs subdirectory is Install Date of OS, the date and time when the OS was successfully installed. The timestamp on the ReAgent.xml file which is found in the b1d8602f, indicates when the OS was successfully installed, this will update to after the timestamp on the Reload.xml.
Within the ReAgent.xml we can see various configuration options for the System Refresh as well as see if it is a default or custom refresh. The first screen shot is the default refresh, the second one is from a custom refresh stored locally, and the final one is a custom refresh stored on a remote drive.

- As we can see the images share similar data between them. This includes the following:
  - WinreBCD ID = is the same identifier of the folder on the system drive.
  - WinreLocation Path = Where the WIMRE.wim file is found on the system drive.
  - OSBuildVersion Path.

As we can see in all three images, there is a value called CustomImageLocation Path that is populated when a Custom Refresh Image is created. Figure 20 is the Reagent.xml with the default Refresh Point, while Figure 21 is the ReAgent.xml with a custom refresh location. The only apparent difference is the CustomImageLocation path is now populated with the new file path.

To create a custom recovery image, you need to use the recimg.exe. When you create a custom recovery image, recimg will store it in the specified directory, and it is set as the active image. If a custom recovery image is set as the active recovery image, Windows will use it when you refresh your PC. All recovery images have the filename install.wim. If no install.wim file is found in the active recovery image directory, Windows will fall back to the default image.

The System Partition is identified in FTK as Partition 2. On a refresh this partition contains the following folders: root, orphan and unallocated space. It also contains the files called backup boot sector and file system slack. The root folder contains the files and settings used within the operating system. Some key
information that can be found in this partition Registry Hives, File History configuration and data, contacts, documents, windows.old and other locations. Depending on how recent the refresh happened there might be a HTML file on the desktop that lists what applications are removed.

The windows.old folder, which is located in the root directory, holds artifacts of what files and programs were installed on the machine before the refresh. From this information it appears that on a refresh nothing is uninstalled or physically deleted from the hard drive but marked to allow the space being over written, you can still access them through devices such as FTK Imager. The files System Partition that are marked for deletion can still be accessed and analyzed as normal. I was able to pull the Registry Hive files and run them through RegRipper and Registry Decoder for data analysis.

The Unallocated Space, contains unallocated file space, while there might be information stored there that will have value, I did not find anything in this research scope. The System Partition contains a wealth of information that should be interesting to an investigator.

### 5.2 System Reset Artifacts

When the System Reset recovery option has been utilized on a machine the data that was once available on a live system is no longer available. Depending on what type of System Reset option is used the data may not be recoverable. If a Quick System Reset is used analysts will be able to extract artifacts from the unallocated space which will include application, system and user data. If a Thorough System Reset is used then user created data will not be found in the unallocated space. Both the System Reset and System Refresh will create identical artifacts in the Recovery Partition, these artifacts will be an updated ReAgent.Xml and a Logs Directory which will contain the Reload.xml file.

If we look into the Logs file we will see the file called Reload.xml within this file has some information related the refresh/restore process. We also notice that there is a file in the system32/Recovery called ReAgent.xml, this file can be found across all three images; Base, Restore and Refresh. When comparing the ReAgent file across the three platforms the files are identical. When comparing the ReAgent and the Reload files against each other they are identical except for the following line ScheduleOperation State. The first one is ReAgent, the second is the reload file.

The Reload.xml file is identical across both a Restore and a Refresh process. So this value must dictate that certain files where restored to the factory default. This may change with machines that are running the Unified Extensible Firmware Interface.
Even though Windows 8, has the ability to reset the operating system back to factory default, it is not impossible to do analysis and data carving on the unallocated spaces to find artifacts that once remained on the machine. As you can see from the image below, even after I did a reset on the machine, which should have returned everything to the factory default, I was still able to carve out my test file, my username, and some files related to the file history option.

![Search Performance](image)

Figure 24

6. CONCLUSION

With the inclusion of new features in Windows 8 there is an increased volume of artifacts that will be useful to analysis for profiling user behavior and actions. While the majority of the features have made it easier for analysis some have increased the difficulty for extraction.

One of the more Forensic friendly features added to Windows 8 would be the Communication App. This app if utilized by the end user is able to provide a wealth of Forensic artifacts that might normally be available on a users machine. Through the artifacts that are created I am able to see the users various contacts across all connected accounts as well as messages, usernames and email addresses. This application allows analysis to get information that at times required contacting individual third parties for the data.

With the addition of FileHistory in Windows 8 it has given analysts potentially new avenues for investigation. With the ability to have many accounts backing up to the same location analysts now have the ability to find outlying computers or accounts that were previously unknown. With the behavior of changing FileHistory locations not clearing data from previous locations this gives analysts the ability to find evidence on forgotten drives.

Finally the availability of an interactive GUI for system recovery allows end users to quickly restore their system to a previous state. Depending on the recovery method utilized on the system it would be beneficial for analysts to know which method result in the destruction of artifacts before spending time trying to analyze a system that has been reset.

Except for the loss of artifacts from a Thorough System Reset, Microsoft has made the ability to find forensic artifacts easier. With the multiple locations that an artifact may remain in, the ease of backing up data, and the importation of third party account information the amount of user identifiable information is greatly increased.

REFERENCES


**ACKNOWLEDGMENTS**

I would like to thank my committee chair, Dr. Yong Guan, and my committee members, Dr. Doug Jacobson, and Dr. Jennifer Davidson, for their guidance and support throughout the course of this research.

In addition I would also like to thank my family for their encouragement, to my wife, Jessica, and children, Savannah and Brady, their hours of patience, respect and love. Thank you for believing in me and pushing me to succeed.

Finally, thanks to Leocadia Adams for your encouragement and always seeing the best in me. The compassion and life lessons that you taught me will always be appreciated.

**AUTHOR BIOGRAPHIES**

Kenneth Johnson recently graduated from Iowa State University with a Master’s of Science in Computer Engineering and Information Assurance. His undergraduate degree is in Information Systems Security from ITT-Tech. His primary interest is in Malware Behavioral Analysis and Reverse Engineering. He currently works for a Big 4 Auditing firm in Chicago with responsibilities in digital forensics, incident response and malware analysis.

Kenneth has presented at SANS DFIR Summit 2012, GFIRST 2012, FACT 2012, TechnoSecurity Conference 2013. He is also the organizer of the BSidesIowa Security Conferences.
A THEMATIC REVIEW OF USER COMPLIANCE WITH INFORMATION SECURITY POLICIES LITERATURE
(Briefing Paper/Presentation)

David Sikolia
Ph.D. Candidate
Department of Management Science and Information Systems
Oklahoma State University
david.sikolia@okstate.edu

ABSTRACT

The adoption of computer and internet technology has greatly improved the way businesses operate. However, the risk to the confidentiality, integrity, and availability of organizational data and systems has greatly increased too. Information security is an ever present concern for all organizations. Financial estimates of the impact of security breaches to information and technology resources range from hundreds of billions to over one trillion dollars each year worldwide (D'Arcy et al., 2011b). Organizations have therefore developed a combination of technical, administrative, and physical controls to reduce this risk (D'Arcy et al., 2011a). Administrative measures include the development of information security policies, which are statements of the roles and responsibilities of the employee to safeguard the information technology resources of their organizations (Bulgurcu et al., 2010). Information security policy provisions include guidelines to employees on what they should do when interacting with information systems so as to secure the data and technology resources of their respective organizations.

Unfortunately, cases of employee intentional and non-intentional non-compliance with information security policies have been documented, with some security experts concluding employees are the weakest link in information security defenses (Aurigemma et al., 2012). Although popular media tends to headline the exploits of hackers or crackers, evidence suggests that a majority information security incidents occur as a result of trusted employees' actions (Hu et al., 2012; Karjalainen et al., 2011). Increasingly complex viruses, worms, Trojans, rootkits, and distributed botnet attacks are mounted by criminal gangs and sometimes foreign governments but the greatest threat of all is the insider threat, the trusted employee (Iffinedo, 2012; Warkentin et al., 2011). It has been claimed that over half of all information systems security breaches occur because employees do not comply with information security policies (Siponen et al., 2010b). Other reports indicate that 50% - 75% of security incidents originate from with the organization, perpetrated by the trusted employee (D’Arcy et al., 2009). However, it must be pointed out that not all violations are by malicious employees. Some violations might be accidental, others violations might be self-benefiting but without malicious intent. Nevertheless, regardless of the motivation, the end result is the same; rules are broken and possibly causing damage or security risk (Guo et al., 2011).

For over two decades, the information systems research community, starting with Straub (Straub, 1990) has published a sizable body of research on user compliance with information security policies. This body of research has been divided into three categories: (1) conceptual principles or studies without theoretical basis (2) theoretical models without empirical support; and (3) empirical support grounded upon theories (Pahnilaa et al., 2007). These theories were borrowed from reference disciplines such as criminology, economics and psychology.

Example theoretical lenses used include general deterrence theory (D’Arcy et al., 2009; Herath et al., 2009b; Pahnilaa et al., 2007; Siponen et al., 2010b), protection motivation theory (Herath et al., 2009b; LaRose et al., 2008; Lee et al., 2009; Pahnilaa et al., 2007; Workman et al., 2008), theory of planned
behavior (Bulgurcu et al., 2010; Herath et al., 2009a), rational choice theory (Bulgurcu et al., 2010), social cognitive theory (Rhee et al., 2009), technology acceptance model (Cynthia et al., 2010; Yajiong et al., 2011), theory of reasoned action (Siponen et al., 2010b), innovation diffusion theory (Siponen et al., 2010a), neutralization theory (Siponen et al., 2010b) and justice theory (Yajiong et al., 2011) amongst others. Of these theoretical lenses, deterrence theory has been used most but the findings have been mixed (D'Arcy et al., 2011a).

The goal of this study is to build upon the body of knowledge on user compliance with information security policies over the last two decades by reviewing previous work and identifying themes or concepts that are antecedents for this behavior. A review of relevant literature is essential for any academic study (Webster et al., 2002), helping researchers identify any gaps that may exist in the body of knowledge. Literature reviews can be written by senior scholars who have published many papers in a given stream of research or by junior scholars who have completed a literature review prior to embarking on a research project such as a dissertation (Webster et al., 2002).

Example literature review papers published in MIS quarterly over the last 30 years include management of information systems personnel (Bartol et al., 1982), knowledge management and knowledge management systems (Alavi et al., 2001), cognitive-affective model of organizational communication (Te'eni, 2001), the resource based view and information systems research (Wade et al., 2004), IT-dependent strategic initiatives and sustained competitive advantage (Piccoli et al., 2005), culture in information systems research (Leidner et al., 2006), Privacy in the digital age (Bélanger et al., 2011; Smith et al., 2011) and Absorptive capacity in information systems research (Roberts et al., 2012).

This study will help us understand the existing body of knowledge on user compliance with information security policies and any gaps that may exist. It will also help us place any future endeavors on this topic in the context of existing work (Levy et al., 2006). This will be accomplished through a systematic search of quality literature on this topic (Ellis et al., 2009), followed by a concept-centric review of the gathered material (Webster et al., 2002). The final outcome will be a model (Webster et al., 2002) to guide future research on user compliance with information security policies.

REFERENCES


FIRST GLANCE: AN INTRODUCTORY ANALYSIS OF NETWORK FORENSICS OF TOR

Raymond Hansen  
Department of Computer and Information Technology  
Purdue University  
401 N. Grant Street  
West Lafayette, IN 47907-2021  
Phone: (765) 796-9482  
Fax: (765) 496-1212  
hansenr@purdue.edu

Keywords: Tor, Forensic Analysis, Privacy & Anonymity

ABSTRACT

The Tor network is a low-latency overlay network for TCP flows that is designed to provide privacy and anonymity to its users. It is currently in use by many as a means to avoid censorship of both information to be shared and information to be retrieved. This paper details the architecture of the Tor network as a platform for evaluating the current state of forensic analysis of the Tor network. Specific attempts to block access to the Tor network are examined to identify (a) the processes utilized to identify Tor nodes, and (b) the resulting exposure of potentially inculpatory evidence. Additional known, but yet to be perpetrated, attacks are examined for a more holistic view of the state of forensics of the Tor network. Based on the combination of these studies, there is some evidence that a specific, individual flow of traffic over the Tor network is attributable to a single entity. However, the content of that flow has not been compromised within the Tor network. As such, the inculpatory evidence required for legal action is limited at this time.

1. INTRODUCTION

Tor is a popular 2nd generation implementation of the onion routing topology (Tor, 2012). This topology was developed in work for the U.S. Navy Research Lab in the mid-1990s (Goldschlage, Reed, Syverson, 1996). This is an overlay network of the public Internet that is designed to provide online privacy and anonymity to its users through two different mechanisms. The first mechanism of privacy relies on multiple encryption iterations in order to obfuscate the entirety of traditional IP packets. The second mechanism relies on seemingly random network ingress points, routing hops, and egress points to diminish an external observer’s ability to identify the end-to-end path of a traffic flow through traffic analysis or network surveillance.

Since Tor is designed to provide anonymity, it would be beneficial to have at least a rudimentary definition of anonymity. One could simply say it is the state of being anonymous. But that seems circular in definition and is less than useful for any rigorous discussion. So, instead, anonymity could better be thought of as “the state of not being identifiable within a set of subjects” (Pfitzmann & Kohotopp, 2001, p.1). Even using this definition, there are still murky waters concerning attribution of the Tor flows to a particular entity.

The anonymity provided by this network is intended to allow users to send and receive data across the network with little fear of being identified by an external observer, regardless of friend or foe status. Since this tool can be utilized by both the innocuous and nefarious with no immediate mechanism to distinguish between them, and the anonymity and privacy provided may afford the impetus for illicit online behavior, many governments and law enforcement agencies have become increasingly
concerned with the operations of this network. Additionally, many nations wish to censor communications both within and across their borders, including: Belarus, China, Cuba, North Korea, Syria, and more (Ho, 2009). In fact, recently, all encrypted traffic was blocked within Palestine (Ma’an News, 2012; York, 2012), all HTTPS traffic was blocked in Iran (Rezale, 2012), and access to Tor Directories and Bridges have been blocked by State-sponsored service providers (Tor Project, 2009; Tor Project, 2010, etc.). For many citizens within these countries, access to Tor was a potential solution to this censorship, if they were able to access the network. As such, shortly after these denials of entrance to the Tor network, access to the nodes was restored by the semi-anonymous operators of the Tor network, much to the dismay of the State.

Discussions of Tor Hidden Service nodes and client-side views of Tor are left for future work. This paper focuses on the network-side attributes of Tor as well as the inculpatory and exculpatory evidence from the data communications and networking aspects of its operations. This view uses cybersecurity as the lens to focus this view. As such, this view is driven by the understanding that cybersecurity and digital forensic analysis are inextricably linked. These two are complementary, supplementary, and co-dependent. An analysis of the security processes gives way to the mechanisms that are useful for performing forensic analysis.

This paper identifies types of uses and users of the Tor network which includes highlighting the history and operations of the Tor network while differentiating its operation from that of the traditional routing mechanisms in use on the public Internet. It then identifies known and executed “attacks” against the Tor network and provides details of their execution process(es). Utilizing previous State-sponsored efforts, earlier successful forensic approaches are identified for the purpose of determining if unique individual flows were identified, or merely particular nodes and their aggregate traffics were determined to be participatory. It will be shown in this paper that even though it might be possible to identify a particular flow of traffic, attribution to any individual is not provable through Tor network attributes. Yet, that correlation of traffic flow to a particular host device may be inculpatory enough for some jurisdictions. Next, known potential attacks against the Tor network are detailed. It should be noted that while these are known potential attacks, there is not yet any proof that they have been. These are intended to provide a foundation to identify potential forensic analysis of operations of the Tor network. Lastly, we detail the potential evidentiary findings of a generic Tor node and any applicable inculpatory indicators therein.

2. THE TOR ARCHITECTURE

Tor is a 2nd generation implementation of the onion routing topology initially developed in work for the U.S. Navy Research Lab in the mid-1990s (Goldschlage, Reed, Syverson, 1996; Tor, 2012). This implementation of the onion routing topology is intended to be a low-latency overlay network for TCP flows over the public Internet that intends to provide privacy and anonymity to its users. Specifically, Tor provides the functionality that “prevents a user from being linked with their communication partners” (Loesing, Murdoch, Dingledine, 204, 2010).

While the original design goal of the Tor network was to provide significantly more privacy to a user than provided in default Internet communications, Tor has recently been used by evade state-sponsored censorship attempts (Loesing, Murdoch, Dingledine, 2010). Dingledine has noted that there is an “ongoing trend in law, policy, and technology” that “threaten anonymity… (and) undermine our ability to speak and read freely” on the public Internet (n.d.). For example, in early 2012, Iran disallowed access to any sites that utilized HTTPS (Kabir News, 2012). Also, in May, 2012, the Palestinian government shut down eight news websites for posting critical opinion pieces of the president (Hale; OONI, 2012). In mid-2012, the Ethiopian Telecommunications Corporation began performing deep packet inspection on all ingress and egress traffic coming in to the country. As Ethiopia’s only service provider, they had direct access to all such traffic (Runa, 2012). York has
provided multiple cases where there have been calls for additional censorship, the creation of a censorship body, and even examples where citizens have been arrested for political or religious reasons (2012b). And so then, Tor is intended to provide not only privacy in these types of scenarios, but anonymity by providing protection against eavesdropping and man-in-the-middle attacks. Additionally, by using multiple iterations of encryption and intermediating nodes defining a radically different path from a client to its ultimate destination, deep packet inspection, traffic analysis, and timing analysis attempts are mitigated.

2.1 Traditional Internetwork Routing

In a traditional routing environment, each packet from a source is addressed with the ultimate address of the destination. A router will then examine the destination address, choose the best neighboring router based on the longest-match algorithm, and then forward the packet to that destination. Each router along the path from source to destination performs this action on every packet in a flow. This process, while resilient to the dynamic topology of the Internet, is inherently insecure. This process potentially leaks information about both the source and destination of the flow, the content of the packets, as well as the path that is used to deliver these packet flows. As such, there is little inherent privacy, and no anonymity in the default routing process.

A pair of options that were defined in the original IP standard for use in networks was loose source and record routing. These options allowed the source node of an IP packet to explicitly specify a partial, or complete, path for the packet to follow (RFC 791, 1981). Source-routing overrides the traditional routing process using the longest-match algorithm to search the routing table that occurs on each router in a path. Instead, the packet is routed along the destinations listed in the source-route contained in the IP packet header. This approach to routing a packet through the network relies on the assumption that the source of the IP packet has a complete view of the network topology to the ultimate destination and can provide explicit path information in the packet. By specifying each hop from the source to destination, or minimally at least one intervening hop, a “trusted” path can be used to deliver packets. Yet, even though the path may be “trusted”, this IP options approach provides no additional security of the packet; neither is there additional privacy, confidentiality, or integrity of the packet using this approach. Further, the source and destination node addresses are still directly readable by any passive monitor in this path. Extending this concept, the use of this IP option could potentially be used to reduce privacy by exposing known “trusted” nodes in the path to additional scrutiny by any interested party. Because of this, most routers on the public Internet ignore the path specified by a host with this option or discard the packet altogether (Zwicky, Cooper, Chapman, 2000).

Enhancements and additions have been made to attempt to add privacy to this original routing process. Encryption, in the form of SSL and TLS, has been added to secure the payload of each packet from inspection (intrusion) by outside parties. However, this approach does nothing to obfuscate the source or destination of the flow. Also, simply encrypting the payload still allows traffic analysis and timing analysis to occur and potentially identify the type of payload being transacted even though the exact payload is not identified. Other encryption approaches have been added to this process to further obfuscate the packet payload through the use of IPsec, which encrypts the IP packet payload. This approach hides the TCP or UDP port numbers, which makes protocol identification more difficult, but not impossible. Traffic analysis and timing analysis are still possible, as well as the immediate identification of the source and destination IP address information for the source and destination nodes. This is perhaps better recognized as providing integrity and confidentiality of the payload, not privacy or anonymity.

An additional implementation of IPsec allows for full encapsulation and encryption of the entire packet. This is referred to as Tunnel-Mode. In this implementation of IPsec, the packet is fully encrypted and then encapsulated within another packet. This new packet typically will have different
source and destination IP addresses than that of the encapsulated packet. These new IP addresses are the end points of a VPN tunnel. Each router along the path from the new source address to the new destination address passes the encapsulated packet with no knowledge of the original source and destination IP address. This approach provides some privacy for the packet flow, as an external observer will have difficulty identifying the payload information. However, the VPN end points are still fully capable of examining the payload and identifying the communicating participants.

2.2 Routing in the Tor Network

The Tor network is based on the original onion routing architecture described in Hiding Routing Information by Goldschlag, Reed, and Syverson (1996). The Tor network, while being based on the original onion routing architecture, has deviated from and enhanced significantly in operation. There are three defined connection-establishment processes for Tor (Dingledine, Mathewson, n.d.). This paper focuses on the most recent of those processes (v3).

A rudimentary understanding of the architectural components is critical to the discussion of the Tor network. So, a short description of the major components of the Tor network is included here:

- **Bridge (Bridge Relay)** – A Tor Relay that does not report to the Tor Directories and can only be reached through knowledge of their location on existing networks. These nodes are used to circumvent filtering techniques against known Relays.

- **Circuit** – The onion router path from a Host through a public internetwork to a subset of Relays that requires a unique set of encryption between the host and each Relay in the path. Routing of packets between the Host and each ordered Relay is handled with the traditional packet forwarding mechanism by the routers within the public internetwork.

- **Directory** – One of a set of Tor nodes that collect, verify, format, and disseminate Relay status information throughout the Tor network.

- **Host** – The source node in the Tor network that runs the Tor software proxy, initiates circuit construction, and passes data through a local SOCKS proxy into circuits for traversal over the encrypted paths.

- **Mix** – The combination of Relays provided by a Directory that a Host will use to establish a Circuit.

- **Relay** – A node in the Tor network that acts as a circuit routing device. This node responds to circuit construction requests by establishing an encrypted channel between itself and host. These node forwards packets based on the Circuit ID contained in the Tor packet header. Availability and capacity of this node are reported to the Tor Directories.

In order for the Tor network to be able to process any connections, each of the Relays must report their availability to a set of Directories. This is shown in Figure 1 below. Each of the Relays reports their name, exit policy, available capacity, and other defined information to the Directories, which then verify the reported capacity and utilize a voting process to establish a network consensus document for distribution to Hosts (Tor Directory Protocol, version 3, 2012).
Hosts must learn of the available Relays by periodically contacting a hard-coded Directory from a set of Directories, as shown in Figure 2. This connection is actually established over a single-hop onion route to obfuscate the source Host from the Directory. The Directory will respond with a consensus document to the requesting Host, which will then determine the mix of Relays to be used to establish the onion route.

Based on the mix derived by the Host from the previous steps, the Host will iteratively build an onion route a single Relay at a time, choosing the exit Relay first. This Relay is selected based on it having an acceptable exit policy and then the remaining Relays are pseudorandomly chosen (Dingledine & Matthewson, 2012). The host will initiate the first layer in the onion route to the initial Relay using the Diffie-Hellman Ephemeral Key Exchange to establish a TLS connection based on those derived keys, as seen in Figure 3, using preferred ciphersuites such as TLS_DHE_RSA_WITH_AES_256_CBC_SHA, TLS_DHE_RSA_WITH_AES_128_CBC_SHA, and SSL_DHE_RSA_WITH_3DES_EDE_CBC_SHA (Dingledine & Matthewson, 2012). Then, the Host sends a Tor specific command (CREATE) to indicate a new circuit is needed. The Relay (Relay A) will respond with a CREATED command if successful. The first layer of the onion route has then been created and is uniquely identified by a CircuitID that is meaningful only between the Host and Relay. Traditional routing processes will still occur to determine the path between the Host and the first Relay, and vice versa, in the onion route.
Figure 3 Building Channel to Relay A

The Host will then extend the circuit established in the previous step by negotiating the second layer of the onion route with the second Relay (Relay B). Again, the Diffie-Hellman Key Exchange is used to establish shared keys for the TLS connection, as seen in Figure 4. The Host sends an EXTEND command to Relay A, encrypted using the shared keys of Host-Relay A, which then forwards the content to Relay B. That content is a CREATE command for Relay B and is encrypted with the Host-Relay B keys. This then establishes the second layer of the onion route circuit. So, the host now manages two shared keys, one for each of the Relays that it has established connections with to build a circuit. The first layer of the circuit (Relay A) is unable to observe the content of the cells that flow over it from the Host to Relay B.

Figure 4 Extending Channel to Relay B

The onion route has one additional layer added to it through the use of a third Relay (Relay C). The same process is used to extend the circuit as in the previous steps, with an additional round of encryption occurring for the newly established connection between the Host and Relay C that traverses both Relay A and Relay B, as shown in Figure 5. Relay A and Relay B both process an EXTEND command, while Relay C processes a CREATE command. At this point, the onion route is complete. A Host now has an anonymized connection through the public Internet.
Using this established onion route, the Host can interact with any public resource as seen in Figure 6 by using the RELAY commands within the cell to be routed. Relay C will perform all DNS queries on behalf of the Host so as to limit any monitoring of the Host in an attempt to identify specific destinations or resources that are being accessed. In this way, each node in the path (initiating client, Relays, and destination device) only knows the identity of its immediately adjacent neighbors (Dingledine et al., 2004). It should be noted, however, that simply using the onion route doesn’t provide any protection of the data flowing between Relay C and the destination. In fact, monitoring traffic at an exit node is one of the viable methods for attempting to identify users of the Tor network. So, any unencrypted traffic between the exit node (Relay C) and the destination can be observed, recorded, and examined for specific content.

A Host does not use, or maintain only a single onion route. In fact, it regularly constructs and terminates circuits in a timely manner. That is, a Tor Host may initiate a new circuit a frequently as once per minute, and will terminate an unused circuit every five minutes. Most circuits are terminated after less than ten minutes of use as a mechanism to reduce traffic analysis attacks against the network and those using the network. A simple example of the diversity of connections maintained by a Host can be seen in Figure 7.
So then, this approach to obfuscating data provides anonymity of the source, as only the Directory and first Relay know the identity of the initiating client and not the identity of the destination. The remaining Relays will not know the identity of the initiating client. Only the exit Relay (Relay C) will know the identity of the destination.

3. CURRENT TOR NETWORK ATTACKS

There are many well-known mechanisms that can be used to censor materials on the Internet. These have ranged from simple source/destination IP address filtering, DNS filtering/injection/hijacking, as well as content filtering (Ho, 2009). However, these approaches assume a user with little expectation of privacy or anonymity. If a user utilizes encryption or other obfuscation approaches, then these censoring tasks become more difficult. This is the intent of those utilizing the Tor network for access to resources on the Internet – make difficult the analysis of their traffic content and its characteristics through encryption and anonymizing services. So then, for authorities to keep pace with the censorship of content flowing over the Tor network, they must also attempt to censor the utilization of this network.

Previously, performing deep packet inspection by a state-owned service provider would provide complete access to the traffic being sent or received by any user. At any ingress/egress point of the network, analysis could take place through a legal (or illegal) wiretap or other intercept process. The network could even be designed to have several “chokepoints” where this analysis could be expedited. Even if non-typical protocols were implemented by privacy- and anonymity-seeking users, digital fingerprinting and traffic analysis would typically be sufficient to identify potentially “dangerous” network traffic. However, the Tor network significantly, and sometimes dramatically, complicates this process for state-sponsored filtering.

As nation-states attempt to quell access to the Tor network, opposing actions have been taken to restore anonymous capabilities. The following is a partial listing that highlights a subset of current “attacks” against Tor (the service), the Tor network components, and the corresponding responses by Tor maintainers to circumvent these censorship attempts.
In September, 2011, Iran blocked access to the Tor network by adding a filtering rule to their national border routers. This rule specifically identified Tor traffic and filtered this traffic while still passing all other traffic that was not previously blocked. This filter identified a specific component of the traffic flow that established the Tor connections. Tor is designed to make all network traffic appear that of a client accessing an HTTPS web server. As such, this requires a secure handshake process. It was this handshake process that was uniquely identified by the Iranian government. The SSL session certificate expiration time on Tor Relays was set to two hours, which is uncharacteristic of certificates issued from a real, public certificate authority. The resolution, which was implemented the same day by Tor engineers, was to lengthen the expiration time of the Relay certificates to more closely emulate that of a true SSL certificate implementation (Tor, 2012).

The standard connection initiation process involves a client requesting a listing of Relays from a Directory. These Directories are statically coded in the Tor client, meaning they are directly known by any party using, or monitoring, Tor. Because of this, the Chinese government simply blocked access to the IP addresses of each of those known Directories, effectively eliminating a client’s ability to establish a connection into the Tor network (Lewman, 2009). Prior to this event in 2009, the Tor developers added functionality to the Tor architecture to resolve the potential of this vulnerability being exploited. The resolution to this vulnerability was the addition of Bridges into the Tor architecture. A Bridge functions just like a Tor Relay, but is not registered in the public Directory servers and is used as the first (entry) or last (exit) Relay in the path. So then, a Tor client must learn of these Bridge locations (addresses) by some other means. This is accomplished by using the graphical interface to Tor, Vidalia, or by manually configuring the Bridge IP addresses into the Tor client after retrieving the information from the Tor website. Considering that both of these approaches are likely filtered by any entity blocking access to Directories and Relays, an email option is also available. (Tor Projects, 2012). This architectural adaptation effectively circumvented the blocking of Directories and Relays.

In late-2011, the Chinese government began blocking access to Bridges within China. Analysis of the blocking actions revealed that connections were initially allowed, but were terminated within a matter of minutes (Wilde, 2012). Further analysis showed that filtering process was performing a passive fingerprint analysis of the communications between a host and the Bridge and then attempted to establish an active connection into the Bridge. It was determined that an identifiable client-side parameter of the SSL negotiation was unique to the communications of hosts to Tor Bridges. The intensity of the active scans suggested near line rate deep packet inspection, which requires significant processing capabilities (Wilde, 2012). This type of scanning lasted for only a few short weeks before it was abruptly terminated (Wilde, 2012). But, was apparently active again in March, 2012 when additional evaluation was performed (Winter and Lindskog, 2012). This attack is further mitigated in the most current handshake negotiation process, as TLS has supplanted SSL, and the parameter in the SSL/TLS cipher list that was uniquely identifiable has thus been removed (Wilde, 2012).

Not long after the Chinese government identified the client-side cipher list issue to block access to Bridges, the Kazakh government also began blocking access to Bridges within Kazakhstan (Lewman, 2012). Again, unpublished Bridges were being identified through continual deep packet inspection. Until this time, Relays were not blocked within the country and Tor was used somewhat extensively. Analysis of traffic arriving at the Bridges suggested no active scans, as the Chinese government had undertaken. However, it was determined that the passive scans of deep packet inspection have identified a unique parameter of the server-side hello message in the TLS negotiation. An additional tool, called obfsproxy, while not directly part of the Tor architecture, continued to function within the country during this time (Lewman, 2012). Thus, allowing citizens anonymized access to the Internet. As of this writing, no resolution to this filtering approach was found.

In the Bad Apple Attack an insecure application, such as a web browser, used over Tor is capable of revealing the Host’s IP address thorough information leaking via Flash, Java, JavaScript, etc. In this
case, they exploited a BitTorrent application. By leaking this information, there is the potential to be able to attribute particular quantities and patterns of packets to the source as member packets of this flow. By leaking this information, all other flows from that host could be considered suspect (Le Blond et. al., 2011).

Many more attacks against Tor can be found by browsing the Tor blogs and brief Internet searches. With the recent political unrest in Egypt, Syria, Libya, and more, there are moribund examples of State-sponsored censorship attempts. Interestingly, censorship is not necessarily only occurring locations of political upheaval. Access to the Tor website is blocked by non-government controlled cellular providers in both the UK and US (Tor, 2012).

4. PROPOSED TOR NETWORK ATTACKS

In addition to the many existing and perpetrated attacks that have already transpired against the Tor network, there are multiple theoretical attacks that have been described by the Tor developers. This section briefly describes some of those attacks as a means to discover Bridges and Relays and subsequently use them an attack launching point to (a) discover the topology of the Tor network; (b) break anonymity of the Tor user (profiling, timing, or traffic analysis); or (c) block access to the Tor network (Tor, 2011a; Tor, 2011b).

Many of these described attacks by the Tor developers and maintainers center on a nefarious party passively monitoring connections through an observation point within the Tor path. A more active version of this is to have the attacker actually participate in the Tor network as a Bridge, Relay, Guard, or even client. Both of these approaches allow the timing and traffic analysis attacks to be performed against a Tor user in such a manner that most Tor users would not recognize any analysis was occurring. Other approaches are more active in their style: including port scans, issuing malformed connection requests, and spoofing messages between Tor nodes. Combinations of these attacks may be performed to be more discriminating in actions against network users. That is, they may be concentrated in a more targeted manner to reduce collateral damage within the network.

Additional attacks have been proposed by Feamster and Dingle (2004); Murdoch and Danezis (2005); and Dingledine and Murdoch (2009). Each of the scenarios described in these papers seek to perform traffic analysis and timing attacks. That is, attribution of a flow of traffic to a particular location within the network at a specific moment in time. The content of these flows were not exposed in these scenarios.

Other attacks over the Tor network have been performed that have specifically identified the actual source’s IP address, hostname, time zone settings, and Internet browser type and version by exposing weaknesses in the applications that use Tor (Christensen, 2006a; Christensen, 2006b). Alternatively, an additional side-channel attack has been proposed by Shebaro (2012) in which a unique, identifiable, binary string is written to a client from a controlled or compromised destination; leaving the client potentially identifiable—eliminating privacy and anonymity upon investigation. That is, users have expected Tor to provide privacy and anonymity of their web browsing, yet their browsing habits allowed their anonymity to be compromised.

5. EVIDENTIAL ANALYSIS

Tor is not a panacea for all network related censorship issues. Tor can’t solve complete network blackouts or shutdowns, as in Syria in late 2012 (Renesys, 2012). There must be connectivity in place for Tor to utilize. Without a network path, Tor is as powerless as any other connection tool; anonymizing or otherwise. With that being said, this section will analyze the potential for identifying inculpatory evidence within the Tor network for the purpose of some action being taken by law enforcement agencies (LEA).
Each of the perpetrated and proposed attacks has little forensic value at first pass. The value of these attacks, when successful, is in the transparent monitoring capabilities that a State or LEA may have that would subsequently allow direct correlation of inculpatory evidence to a specific network host. This approach from a state-sponsored attack is highly reliant on their ability to track or trace a particular data flow. This ability to correlate a flow to a particular host is dependent on the ability to actually capture those flows. This is directly related to the number of egress (and by extension, ingress) points to those networks. An analysis of the number of connection points into and out of a nation has been performed by Renesys in order to determine the likelihood of a complete blackout occurring similar to the Syria 2012 incident (Forbes, 2012). An alternate view of this data could be taken that a state may not wish to fully disconnect their citizens from the Internet. Instead, these connection points can become the capture, and correlation points for their monitoring systems; thereby, providing a State an evidence-gathering facility directly in the network path.

5.1 Analysis
To date, there are no known attempts to break the encryption algorithms used specifically attributable to the Tor network. That is, all known attempts at obstructing access to the Tor network, or identifying a user of the Tor network rely on attacking the architectural components or passive observation of traffic over the Tor network. As seen in Lewman (2009, 2012); Tor (2011a, 2011b); and Wilde (2012), it is possible to identify the role(s) a particular node is performing in the Tor network. When a connection is established, it is possible to determine which device is the client by identifying the TLS “Client Hello” portion of the encryption exchange. Likewise, Wilde showed that Bridges are identified in China and Kazakhstan through the TLS “Server Hello” messages within the encryption exchange. So then, we know directly that there are certain identifiable characteristics attributable to each role within the Tor architecture. This has been discussed further in Tor (2011a) as a code implementation and auditing issue.

It may be possible to identify traffic flows through the Tor network without knowing the location of Tor Relays, Bridges, or hosts. Tor specifies a cell size of 512 bytes. As such, it may be possible to examine flows for multiple consecutive packets around this size, as it may indicate a Tor flow is present. This pattern of packets will differ from that of a typical web transaction, where many consecutive packets are sized at the MTU and only the last packet will be smaller than the MTU. This is one example of the many different traffic and timing analysis attacks that could be utilized to identify Tor traffic. Yet, this analysis doesn’t reveal the exact contents of the packets traversing the Tor network. Specific patterns of Tor cells may reveal the obfuscated protocol(s) and thereby types of traffic. However, no content is directly leaked out of the Tor network in any of the approaches. Anonymity is compromised via correlation of these flows to potential traffic patterns of known flows, or templates of flows.

5.2 Inculpatory Approaches
What, if any, evidence is available to prove participation with the Tor network? Many nations that are actively pursuing censorship of their population typically wish to identify those that are evading the systems in place that block or otherwise restrict access to the censored content. In identifying those persons, there is the potential for legal action, if there is inculpatory evidence. We have shown that many current nations that are blocking access to the Tor network are not yet actively pursuing the participants for legal action. Yet, that is not to say that they will not do so in the future.

It has been suggested that a LEA wishing to identify inculpatory evidence should host its own exit Relay within the Tor network and then actively perform traffic and timing analysis on the traffic as it exits and any response traffic that is generated (Schneier, 2008). Additionally, they could perform deep packet inspection on the traffic as it leaves the network, as the traffic will no longer be encrypted using the Tor network’s layers of encryption. In this way, any unencrypted traffic would reveal the
actual payload sent by the anonymous client. The difficulty with this approach is that a Tor client will use a single channel for a single flow for no more than five minutes, by default. So, if the flow is longstanding, hosting an exit node will only account for a portion of that flow.

As seen in the prior section, there are many proposed attack methods intended to circumvent anonymity in Tor. As of this writing, there are no known successful attacks on the actual underlying encryption standards used within Tor. However, it has been known for some time that the vulnerabilities of encryption reside in potentially poor implementation of the encryption protocols and standards, not the protocols themselves (Schneier, 1998). As such, this paper will not spend any additional efforts to describe encryption circumvention attempts.

So then, based on the known executed attacks and potential attacks, what is the state of inculpatory discovery attempts? There are many denial of service attacks, as seen by the Chinese, Kazakh, and Iranian examples (Lewman, 2009; Lewman, 2012; Wilde, 2012). However, these do not provide any inculpatory material for an investigator as the client’s identity has not been determined in any of these cases. That is, only access to the Tor Directory Service has been limited (Winter and Lindskog, 2012). Additional efforts are required to specifically identify the location and identity of the client in order to gain inculpatory evidence, depending on jurisdiction.

Other investigative efforts have been performed and identified that a direct attack of the Tor protocol and architecture is not the best means of identifying the users of the anonymizing service (Christensen, 2006a). Instead, attacks and manipulation of the application layer services being delivered over the Tor network are a much better means of determining Tor users (Christensen, 2006b; Fleischer, 2009).

Additionally, a LEA may wish to establish a mechanism for additional inculpatory evidence to elimination potentially indefensible scenarios. Exploiting the clients and services within this network could allow a LEA to place a unique “key” on a host under investigation, assuming they control part of the network, or a service requested by the client. Shebaro (2010) has proposed a mechanism by which a unique and recoverable bit pattern can be placed onto a host over the period of ~30 minutes with little possibility of timing analysis revealing its transmission.

Is Exculpatory evidence present or even feasible? As evidence may be present from other “regular” web browsing and file handling, specifics about exculpatory evidence are not explicit in this study. Understanding that there are differing thresholds for inculpatory evidence inclusion in any examination is critical (Loesing, Murdoch, Dingledine, 2010). In fact, there are hints of mere proposition of inculpatory evidence being sufficient to incarcerate or otherwise persecute individuals. (Hale, 2012).

6. CONCLUSIONS

The Tor network was devise and deployed as a low-latency anonymity-providing overlay network for TCP flows. It is implemented in a manner so that no node within the network can identify a complete flow through the network. A node is only capable of communicating with directly adjacent Tor nodes, even though they require potentially significant standard routing processes to connect those two adjacent nodes. This anonymity-providing network is now in use by those simply wishing to obfuscate their traffic from any potential observers as well as by those actively attempting to circumvent censorship. Because of this, those entities advocating censorship actively wish to maintain the censorship by eliminating the means used to evade it.

Traffic analysis of Tor nodes operating in a known infrastructure currently provides little inculpatory evidence without significant efforts to capture traffic at ingress and egress of multiple points of the network. Correlating this analysis to individual hosts on the Internet in a reliable and defensible manner poses a daunting challenge. Yet, this is, to date, the only executable attack against the Tor
network that attempt to determine “what” an individual flow might contain. The encryption methods used within Tor have not been circumvented. So, the raw payload of the traffic is not viewable.

There are potential attack points with the Tor architecture and the protocols implemented. Some, of which, have already been executed. However, these are not trivial exercises to perform or leverage these attacks. The more likely approach to breach the Tor network is to attempt to expose the identity of the users via exploitation of the weak (in terms of security) implementations of application-layer protocols and services that directly interact with those protocols.

Inculpatory evidence from the Tor network is difficult to obtain for most nation-states and their law enforcement agencies. Simple monitoring of existing Tor nodes will not directly reveal the types of traffic contained in the flows that passed, nor will any Tor node know the complete path that the flow is taking. Even if an agency were to implement a Tor Relay or Bridge and monitor connections to it, significant man-power and technical expertise is required to maintain and monitor these nodes for the purpose of ultimately reporting analyses on traffic behaviors. This is an implausible scenario for all but the largest of LEAs… typically those that are state-sponsored.

By looking at each individual component of the Tor network, the significance and amount of effort required to begin these attacks or analysis of the attacks can be seen. Beginning with inculpatory Analysis at a Directory–Since no user application data is sent to, passed through, or retrieved from the Directory there is little chance of inculpatory evidence. Also, since no single directory is authoritative for the state of all Relays in the Tor network there is no single point of attack, meaning also that there is no single point for investigation. As there is a voting process between all Tor Directories that establishes an agreed upon state of each Relay and publishes to all requesting Hosts as a network consensus document there could be the potential for learning some capability information of each Relay. But, this process is handled over encrypted links between each Directory and the Directory and Host. Additionally, there is no Directory discovery process for Hosts or Relays. Instead, the Directories are coded within the Tor toolset currently. So, all communications with a directory are encrypted. However, if this information is needed, it is not private information to the Tor network. A simple query to a Directory will result in the current network consensus document. This may have little inculpatory (or exculpatory) value unless it can be obtained during a specific monitoring period, as the document is updated on an ongoing basis.

Inculpatory Analysis at Entry node (Relay) – As shown in many of the insecure application attacks, the Onion Proxy (Host) can potentially be identified as the source of a flow of traffic. But, after the secure onion router circuit is established, traffic & timing analysis are the only viable methods, which is not significant in many cases, as attribution is significantly difficult.

Inculpatory Analysis at Intermediate relay – An intermediate (second) Relay in the onion routed path cannot identify source or destination without significant effort. This would require capabilities to monitor all relays in the onion route path in addition to performing traffic and timing analysis. This only provides the capabilities to perform attribution of ingress flows to egress flows with an uncertain level of probability. So then, only directly adjacent relays are known, and all communications along that channel segment are encrypted multiple times.

7. FUTURE WORK

While the forensic analysis that was briefly surveyed here shows little current inculpatory evidence is available by monitoring, and even participating in, the Tor network, there is significant interest in further evading detection of the Tor network. Some have proposed a censorship-detection add-on to the Tor network, while others are concentrating on continuing to push for the maturation of Tor. Extending the idea that Tor is a privacy- and anonymity-enhancing tool, and that those that block and
monitor Tor do so out of a desire to limit access to some resource, there are efforts required to define anti-forensic mechanisms within the Tor network.

Lastly, the mixes of Relays that are provide to the Tor client to establish circuits and connections are in need of more research. First, a thorough evaluation of the allocation policy and its potential to leak location or other identity information could be performed. Next, it has been suggested that even though multiple Relays are returned in a mix and are thought to be geographically diverse, they may still be part of the same administrative domain (BGP ASNs). There has been some effort to alleviate this, as no two Relays within the same /16 CIDR block will be selected in the same circuit. But, there has been no definitive proof that this is sufficient. So, it should be determined if this is indeed the case en masse, or as the outlier. To protect against such a situation, it is suggested that a module be added to the Tor Directory that documents the correlation between destination IP addresses and the BGP autonomous system in which those destinations reside. The ultimate purpose of this module will be to ensure that a mix provided by the Directory does not contain Relays from the same, or adjacent, ASes.

Further, there are additional architectural implementations of Tor that provide hidden services, where dual-party anonymity is possible. Additional efforts are needed to determine the state of forensics on those particular architectures and the services available over Tor network in this hidden nature. Identification of services offered and how they are processed through the Tor network with respect to the “standard” delivery of those services is of some interest.

REFERENCES


Dingledine, R. (n.d.). Tor: An anonymous Internet communication system.


Dingledine, R., and Murdoch, S.J. (2009). Performance improvements for Tor or, Why is Tor slow and what we’re going to do about it. DEFCON 17.


AN ONTOLOGY-BASED FORENSIC ANALYSIS TOOL

Mohammed Alzaabi¹ (mohammed.alzaabi@kustar.ac.ae)
Andy Jones¹,² (andrew.jones@kustar.ac.ae)
Thomas Anthony Martin¹ (thomas.martin@kustar.ac.ae)

¹Khalifa University of Science, Technology and Research, United Arab Emirates
²Edith Cowan University, Perth, Australia
Phone: 9 716 597-8888
Fax: 9 716 561-1789

ABSTRACT

The analysis of forensic investigation results has generally been identified as the most complex phase of a digital forensic investigation. This phase becomes more complicated and time consuming as the storage capacity of digital devices is increasing, while at the same time the prices of those devices are decreasing. Although there are some tools and techniques that assist the investigator in the analysis of digital evidence, they do not adequately address some of the serious challenges, particularly with the time and effort required to conduct such tasks. In this paper, we consider the use of semantic web technologies and in particular the ontologies, to assist the investigator in analyzing digital evidence. A novel ontology-based framework is proposed for forensic analysis tools, which we believe has the potential to influence the development of such tools. The framework utilizes a set of ontologies to model the environment under investigation. The evidence extracted from the environment is initially annotated using the Resource Description Framework (RDF). The evidence is then merged from various sources to identify new and implicit information with the help of inference engines and classification mechanisms. In addition, we present the ongoing development of a forensic analysis tool to analyze content retrieved from Android smart phones. For this purpose, several ontologies have been created to model some concepts of the smart phone environment.

Keywords: digital forensic investigation, digital forensic analysis tool, semantic web, ontology, android

1. INTRODUCTION

As a result of the ongoing trends towards larger storage capacities for digital devices, the digital forensics domain is facing a number of serious challenges with the increased time and effort required to analyze data from these devices. Such trends have influenced the process of identifying relevant traces which is usually surrounded by a vast volume of irrelevant traces. Additionally, the complexity of data formats and their diversity have made the investigator spend much of the time in understanding the structure of the data rather than locating relevant evidence.

The existing forensic tools, which have been referred to as First Generation (FG) forensic tools by Daniel Ayers (Ayers, 2009), have shown a number of limitations in addressing the increasing complexity and volumes of data. Popular FG forensic tools such as EnCase and FTK tend to have a common architecture that is based on the same conceptual model for searching and presenting information.

Simson L. Garfinkel (Garfinkel, 2010) describes this type of model as a “Visibility, Filter, and Report” model. In this model, the retrieved traces are made visible to the investigator through a number of cascaded windows in the forensic tool. Derived traces to be analyzed are presented in a tree structure with a root object from which access to other data objects is attainable (Garfinkel, 2010). In the case of the analysis of a hard drive, the root of the tree structure may be a partition table of the drive which may lead to other directories and data objects that builds the file system stored in that drive. Individual
objects can also be viewed in a window. In order to reduce the number of displayed objects, the investigator can filter the results. Keyword search is currently dominating the techniques used to analyze the data (Louis and Engelbrecht, 2011). Finally, a report is auto-generated by the tool summarizing what was found. Instead of generating a comprehensive report of all of the collected data, most of the tools allow the examiner to choose what traces are to be included in the report.

FG forensic tools have a number of limitations as far as analyzing traces is concerned. Firstly, the investigator is usually presented with a familiar GUI interface that depicts a tree structure of the claimed files and directories from an image. However, since the volume of traces is increasing, the tree structure is also increasingly complex, which results in the investigator being overwhelmed with a large number of files and directories to be examined. Consequently, identifying relevant evidence becomes a more complex task. Cross analysis between different types of data sources is also becoming even more complicated as the sources to be analyzed are increasingly diverse.

Secondly, in order to identify potential evidence, examiners tend to use the advanced search techniques which are provided by most forensic tools. The examiner uses his/her experience and the available background information about the case to choose potentially suitable terms by which relevant traces can be revealed or more terms can be identified for further examination. Due to the dependence on the investigator’s experience and the availability of background information about the case, the selected search terms will only be as good as these two factors. Where these two factors are not sufficient, searching for potential evidence can be inefficient.

With these limitations, the need for a more automated forensic analysis tool becomes apparent. In light of this, we believe that the use of an ontology can greatly assist the development of an automated analysis tool. This tool may allow the examiner to handle the problem of the large volumes and the complexity of data more efficiently. In this paper, a novel framework is proposed for a potential next generation forensic tool. The framework is based on semantic web technologies where ontologies are used to model the environment under examination. This model encompasses resources and their relations in a graph-based dataset by utilizing the Resource Description Framework (RDF). With this dataset, which acts as a network of data, a solid, interconnected knowledge base of the different evidence objects extracted from the device may be provided. However, to the extent that the semantic network and ontology act as assumptions, they may also lead the examination astray and the examiner using these methods must understand both their nature and their limitations in order to avoid errors and omissions in their results.

2. SEMANTIC WEB

Due to the openness notion of the World Wide Web (WWW), contributing to its web of information is relatively easy and effortless. This has led to a massive increase in the amount of web content and has introduced a number of serious shortcomings in the underlying Hypertext paradigm of the Web. This paradigm simply defines the structure of the Web and allows users to access, connect, and share information over the internet. One of these shortcomings is the failure in organizing the vast amount of web content in a logical manner which can make searching and locating specific information a difficult task.

The idea of the semantic web was mainly introduced to limit this problem and to have more organized, integrated, and consistent web content (Fensel et al., 2002). Based on the underlying notion of giving a well-defined meaning for each information item (such as, text, picture, or video) in the Web, these information items cannot only be understood by humans, but also by machines. For instance, once the machine can understand what a person, place, and event is, it can help the user to store that event automatically to the calendar. In the semantic web, things that exist in the world or can be described using the associated methodology are known as Resources (Allemang and Hendler, 2011), and the relationships between these resources are known as Relations. These two concepts form the basis of
any semantic web document. The strength of the semantic web lies in its ability to explicate the relationship between multiple resources, even when the resources come from more than one source, allowing them to be easily integrated. For instance, in a library management system, the librarian may need to integrate data from multiple publishers. Presenting any authorship relation between the author (i.e. the writer) and the book using the relation hasAuthor will allow the system to interpret this information for all publishers dataset, even without human intervention.

However, just introducing resources and relations to the content of the web will not solve the problem and may result in a more chaotic dataset, specifically if they have not been properly modeled. Modeling the environment is a key step towards a semantically well-structured document. Models, as described in (Allemang and Hendler, 2011), permit people to collaborate and organize the information that they would want to share and provide a consolidated understanding of the environment. The semantic web standards have introduced some modeling languages that have become the de facto standards for creating semantic web documents, such as RDFS and OWL. A model in the semantic web is known as an Ontology. Pioneers in this field have gone through many attempts in order to define what an ontology is. Probably, the most well-known definition is by Gruber (Gruber 1993) who defines an ontology as “an explicit specification of a conceptualization”.

Authors such as Stevens et al. (2000), Mika (2005), and Guizzardi (2007) have explained the term conceptualization from Gruber’s definition as identifying relevant concepts (or resources), the relations between them, and any constraints that they hold between them in a domain of discourse. In other words, a typical ontology consists of a finite number of terms (or vocabulary) and the relations between them. These terms denote some concept of a particular domain in the world. For instance, and considering the library management system example, the domain that is described here is the library, where concepts such as Book, Author, Publisher, and Borrower are terms relevant to that domain. These concepts are also called Classes. Subclasses can also be derived from main classes to form a hierarchy of classes. For example, Science Book and Philosophy Book are two possible subclasses of the class Book.

Relations in an ontology represent associations between classes. Robert Stevens et al. (Stevens et al., 2000) have categorized ontology relations into two broad types: (1) Taxonomies, which allows for building a tree structure of classes/concepts in the ontology. A great example of this type is the inheritance relation, also known as “is a kind of” relation. For instance, a Science Book is a kind of Book. (2) Associative, which relates classes/concepts across the entire ontology. Examples of such a relation are many; however, one is the Nominative relation that describes the name of a concept - Publisher hasName publisherName.

One of the core technologies used in the semantic web is the Resource Description Framework (RDF). RDF is a World Wide Web Consortium (W3C) standard framework used to represent data on the Web (Klyne et al., 2004). It provides a formal method to encode information about Web resources in a graph-based data model. The syntactic construct of any RDF expression is what is called a triple. Each triple consists of three elements, namely subject, predicate, and object. A triple describes a binary relation and can be represented by two nodes (subject and object), connected through an edge (predicate). For instance, we can describe the relation between one book and its author as the following triple:

PracticalRDF hasAuthor “Shelley Powers”

where PracticalRDF is the subject, Shelley Powers is the Object, and they are connected by the predicate hasAuthor. It is important to mention that RDF is utilizing the Uniform Resource Identifier (URI) for the purpose of naming the three elements of the triple. For example, the predicate hasAuthor is denoted as http://digitalLibrary.com/hasAuthor. This will facilitate the ability to uniquely identify the resources, so no two different resources can have the same name. Therefore, integrating information from non-local resources will not lead to naming conflicts (Horrocks, 2008). Since using
URIs may result in a longer name, it is always preferable to use prefixes to shorten the name of resources. Given the triple above, it can be represented as follows:

```turtle
@prefix library: <http://digitalLibrary.com>
library: PracticalRDF
library: hasAuthor "Shelley Powers".
```

Generally, RDF triples can be serialized in multiple formats. The previous triple and the ones used in this paper are represented in the Turtle serialization. Other serializations are XML, N-Triples, and Notation 3.

3. THE SEMANTIC WEB AND DIGITAL FORENSICS

Semantic web technologies have not only influenced the way that web applications are developed, but have also paved the way for new contributions in various sectors. One of these sectors, and perhaps the most well-known, is the biomedical sector. Doctors and researchers in this field are constantly required to take well-informed decisions regarding different diseases and symptoms. As such, heterogeneous data sources are required to integrate data from various topics such as cells, drugs, and proteins. Other sectors are knowledge management in large enterprises (Fensel et al., 2002) and software engineering (Happel and Seedorf, 2006).

Semantic web technologies have also been seen in the digital forensics sector; however, published works in this particular area are still sparse.

DIALOG is an ontology which has been developed by Damir and Tahar (Kahvedžić and Kechadi, 2009), for the purpose of describing forensic investigation results. It defines a vocabulary of the main concepts involved in the forensic investigation domain. The ontology models four main dimensions related to forensic investigation; which are Crime Case, Evidence Location, Information, and Forensic Resource. Each of these dimensions is responsible for modeling a particular concept in a forensic investigation. For instance, the Crime Case dimension models concepts which are related to the type of crime. The taxonomy of the Crime Case defines the Cyber Crime Case and the Non Cyber Crime Case as the most general concepts to differentiate between types of crime cases. Cyber Crime Case in turn is also divided to involve high-tech crimes such as fraud and software piracy.

Turner (Turner, 2005) has introduced a new concept for data acquisition and representation called Digital Evidence Bag (DEB). DEB is an abstracted model that permits investigators to acquire data from multiple sources of evidence. Evidence sources are organized in a conceptual structure called Evidence Bag. Each evidence bag contains a set of files where information about the case (such as investigator information, evidence acquisition process, list of evidence contained) and the evidence source (such as list of files and directories of the image and their metadata) are stored. In addition, DEB emphasizes the issue of data integrity by including a hash signature for the evidence bag to maintain its provenance.

One year after proposing DEB, Schatz and Clark (Raghavan et al., 2009) extended this work and introduced the Sealed Digital Evidence Bag (SDEB). SDEB has provided a new representation approach where metadata and evidence information are integrated using a pre-defined domain ontology of the context of the case. SDEB uses RDF to annotate evidence related metadata and uses ontology to describe their vocabulary. For instance, an image file is annotated by imposing its metadata and is linked to the Image File concept in the ontology. This process is done recursively for all evidence resources and the obtained information is stored in the evidence bag.

Damir and Tahar (Kahvedžić and Kechadi, 2011) also proposed an ontology-based approach for representing evidence from forensic investigations. The authors refer to this approach as an evidence management methodology for encoding the semantic information of evidence. This approach aims to assist the investigator in report writing, evidence communication, and to relieve the investigator from
the task of manually describing the evidence. Part of the methodology is to involve the investigator in annotating the evidence. To illustrate this, the authors gave an example of a file that contains some text and an image. The text is referring to an individual and the image is also displaying an individual at a certain location. From the ontology point of view, the file, text, image, individual, and the location are all resources. Therefore, in this case, the investigator can link the text to the individual resource it describes, and link the image to the individual resource as well as to the location resource it captures.

It can be seen that all of the above research clearly has the emphasis on representing and managing results obtained from a digital forensic investigation. For instance, the taxonomy of the DIALOG ontology is developed to hold information about the committed crime cases (through the Crime Case dimension) and the forensic resources that were used to conduct the investigation (through the Forensic Resource dimension). Such information is crucial for evidence management and reporting tasks. The emphasis of our ongoing research, however, is towards the analysis of digital forensic evidence. Hence, utilizing the DIALOG ontology will not satisfy our needs.

The works proposed in (Raghavan et al., 2009) and (Kahvedžić and Kechadi, 2011) are similar to the annotation process performed in our framework. Nevertheless, the same reason why DIALOG cannot be applied in our framework is also valid for both proposals. They model the evidence for management purposes, but not for analysis purposes, which is the aim of our research. The framework presented in this paper not only annotates data, but also tries to correlate evidence from various sources and identify new, implicit information. This can be achieved by the use of inference engines and classification mechanisms.
4. FRAMEWORK FOR ONTOLOGY-BASED FORENSIC ANALYSIS TOOLS

Figure 1 Ontology-based framework for forensic analysis tools.

The proposed framework, shown in Figure 1, is a layered ontology-based framework that follows the principle of superimposing the metadata of existing evidence resources. This metadata is used by the Concept and Relationship Extraction layer where various concepts and the interrelationship between them are extracted and maintained. This extraction is conducted with the help of a structured representation of particular domains of interest which are referred to as Domain Ontologies. The encoded knowledge (which is represented in the concepts and their relationships) is maintained in an RDF/OWL database allowing forensic tools to interact with a centralized database that facilitates the organization, access, and reuse of potential evidence objects. The framework consists of five main layers, namely the Evidence Space, File Wrappers, File Description and RDF/OWL databases, Concept and Relationship Extraction, and Domain and Application Ontologies.

4.2 Evidence Space

The evidence space is where potential evidence objects such as files purported to be documents, images, videos, and databases are located. In forensic terms, this space can be a forensic image of a device under examination. It may contain potential evidence objects that are structured such as databases (which conform to relational models), semi-structured such as XML files (which may not conform to any rational model but are organized by tags), or less structured such as videos and images.

4.3 File Wrappers

The file wrapper is simply a program that extracts descriptive information about various types of files from the evidence space. A main source of this information is the metadata of a file which may
provide property-value pairs that may improve the process of relevant content retrieval. Examples of such pairs are: file size, date, and MIME type. In this framework, these file wrappers are tightly-coupled with the structure of the file system of the device. Thus, different devices may require different file wrappers to work with. This layer is the only layer that contains device-dependent components. Another primary purpose of the wrappers is to ensure a consolidated form for retaining the extracted data. This is particularly useful for managing and accessing the information.

4.4 File Description and RDF/OWL Database

This layer mainly holds two core databases; the File Description and RDF/OWL databases. The former is used by the file wrappers in order to store the extracted descriptive information of files from the Evidence Space. The latter, which is the RDF/OWL database, is used by the Concept and Relationship Extraction layer to store the entire knowledge base of the potential evidence under examination. The structure of this knowledge base is governed by the upper ontology layers. The format used in these databases is the Resource Description Framework (RDF).

4.5 Concept and Relationship Extraction

The major purpose of this layer is firstly to extract concepts from the File Description database and determine to which class this concept belongs to based on the upper ontology layers. Examples of such concepts that can be obtained from a smart phone are: a contact which belongs to the Contact class, an image which belongs to the Media class, and a Word document file which belongs to the Document class. By utilizing the metadata of these concepts, relationships with other concepts (and hence classes) can be maintained. For instance, each message instance may hold the sender information such as the name and the phone number in its metadata. This information can be linked to a contact instance from the Contact class given that the two instances have the same phone number.

4.6 Domain and Application Ontologies

These two layers collaboratively form an ontological model for a particular environment. This model consists of concepts (or classes) and the relationship among them. A domain ontology formally models the concepts and their relations in a particular domain, rather than modeling generic subjects of the world. For instance, in a smart phone environment, Message, Person, Email, and Event can be considered as individual domain ontologies. Dividing the whole ontology into more domain ontologies strengthens the structure of the model in various aspects. Firstly, since the domain ontology concentrates on a specific concept, designing the ontology should become less complex and result in a more homogenous concepts and relations. Secondly, any modification to the domain ontology will be more manageable and less time consuming. Thirdly, adding a new concept to the application ontology becomes a straightforward process as domain ontologies can be inserted as “add-ons”.

Defining domain ontologies on its own is not sufficient to model the entire environment of something such as a smart phone; therefore, the Application Ontology is used to assemble the entire picture of the model. The Application Ontology introduces a new layer of interconnected domain ontologies. This is done by adding high level relations between domain ontologies. For instance, a relation called \texttt{hasSent} can be used between the Contact class and the Message class (which means that a contact has sent a message) as a high level relation. This relation forms an abstract view of the relations between the Contact class and the Message class. More detailed relations can be derived from this abstract relation such as: \texttt{hasSentSMS}, \texttt{hasSentImMessage}, or \texttt{hasSentEmail}. 
5. SEMANTIC ORGANIZATION OF THE EVIDENCE SPACE

5.1 Annotation

By considering the evidence space as the primary source for classes instance data, the evidence space should be processed in a way that allows machines to understand its content. This is directly linked to the purpose of the semantic annotation. The annotation process, which is performed by the File Wrapper, is typically achieved by annotating the resources from the evidence space in a machine-understandable manner using metadata. In other words, the annotator creates or enhances the metadata of a file allowing that file to be processed more effectively by machines. This step facilitates a formal description as well as a new access method to the resources available from the evidence space. For instance, considering a trace assumed to be an email, the trace’s metadata generated by the automated process and associated with this trace may be the email address of the sender, recipients, title, and the received date.

In the proposed framework, the annotation process is taking place in the File Wrapper. The extracted resources from the File Wrapper will be annotated by imposing their metadata. The annotated data will then be stored in the File Description database. This database stores the information in the RDF format.

5.2 Association

Resources from the evidence space may have relations created between them by the automated process, representing some asserted association between the different entities they represent. The process of asserting these associations has many potential benefits to an investigator, as they may help the investigator to understand the overall picture of the environment under examination. The links established between resources may lead to a more probable source of evidence. Thus, the automatic discovery of such associations becomes a key feature for the next generation forensic tools allowing the investigator’s time to be organized more efficiently.

By providing a meaningful understanding of the association between resources, the investigator will not only be able to place these resources in the context of investigation, but will also be exposed to other evidence objects that were not immediately obvious. For instance, in a situation where the investigator would like to explore more about a particular resource of type Person, the association with other resources that have relations with it can be identified. Figure 2 depicts these relations with other resources of the same type (i.e., Person).

![Figure 2 Association representation between resources](image)

If Person A is being investigated, two associations with Person B and Person C may be asserted by the examination process. The asserted association between Person A and Person B is represented by a relation where Person A may have sent a message to Person B, while the asserted association between
Person A and Person C is represented by a relation where Person A may have called Person C. Although these two asserted associations can be sufficient to answer the investigator enquiries, further evidence may also be considered which may lead to more potential evidence. In this example, asserting that Person B has shared a location with Person C may open further questions to the investigation.

Two levels of asserted association are involved in this framework. The first level maintains asserted relationships among the resources themselves, a so-called Core Knowledge Association. On the other hand, the second level is called Indexing Association. This asserted derived association level maintains asserted derived relationships between the physical files from the evidence space and the identified resources.

### 5.2.1 Core Knowledge Association

In this level of asserted association, the asserted relationship between resources (or concepts in the ontology) is asserted. These associations represent the core knowledge of the environment under examination by defining how concepts are turned into interconnection in the ontology. Therefore, three types of asserted association may be distinguished in this level; asserted association among classes of a single domain ontology, asserted association among more than one domain ontology, and asserted association of the application ontology. Figure 3 shows these types of asserted associations.

<table>
<thead>
<tr>
<th>Single Domain Ontology</th>
<th>Multiple Domain Ontologies</th>
<th>Application Ontology</th>
</tr>
</thead>
<tbody>
<tr>
<td>IndividualContact</td>
<td>SmsMessage</td>
<td>IndividualContact</td>
</tr>
<tr>
<td>hasLastName</td>
<td>hasBody</td>
<td></td>
</tr>
<tr>
<td>hasFirstName</td>
<td>hasMessageType</td>
<td></td>
</tr>
<tr>
<td>literal</td>
<td>hasSMSAddress</td>
<td></td>
</tr>
<tr>
<td>hasMeansOfContact</td>
<td>hasDate</td>
<td></td>
</tr>
<tr>
<td>literal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MeansOfContact</td>
<td></td>
<td></td>
</tr>
<tr>
<td>literal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PhoneNumber</td>
<td></td>
<td></td>
</tr>
<tr>
<td>hasSentMessage</td>
<td></td>
<td></td>
</tr>
<tr>
<td>literal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 3 Illustration of the different types of Core Knowledge asserted associations based on how classes are connected between different ontology levels.

The previous classification is mainly based on how classes are asserted to be connected between different ontology levels. However, another way to classify the Core Knowledge asserted associations is based on the nature of the asserted relationship that each asserted association type represents. These types are: Instantiation, Inheritance, and Property.

The instantiation is used to show an asserted instance of a class. This relation is usually represented by the RDF property *type*. For instance, `<#Report1, rdf:type, #Document>` indicates that the resource Report1 is an instance of the Document class. The inheritance, on the other hand, facilitates a relation between two resources where one of them is a subtype of the other. This relation becomes handy.
particularly when a subclass of a main class is to be illustrated. For instance, the inheritance relation would be used to show that a class called SmsMessage is a subclass of the Message class. The property relation is defined by the user to express an environment-specific asserted relationship. For example, the asserted relation SentMessage from Figure 2 expresses a specific relation where Person A has sent a message to Person B based on the assertions of the relationships.

5.2.2 Indexing Association

Being able to express the asserted knowledge retrieved from the evidence space, through the previously discussed asserted relations, is almost certainly not adequate to perform a full examination. In some circumstances, the examiner may wish to associate the physical location of a particular asserted file to its corresponding resource in the knowledge base. Thus, the second level of asserted association relates each discovered resource to its location in the evidence space. For example, if the examiner comes across an asserted author of an asserted Word document and would like to explore and read more about that document, the indexing association will link the asserted author (which is a resource) to that asserted document file. In other words, the asserted indexing association indexes the evidence space to allow easy and fast retrieval of asserted files.

In the proposed framework, setting up the asserted indexing association actually means linking resources from the File Description database to the corresponding ontological concepts. Since the File Description database retains only the extracted asserted metadata of files from the evidence space, the asserted indexing association will be restricted to that information. Nevertheless, the level to which the resources could be extracted may vary depending on what information there is to examine. For instance, the asserted file’s content may be utilized to retrieve more resources. This process may be automated through techniques that are commonly seen in Artificial Intelligence (AI). Some techniques that may be utilized here are:

**Named Entity Recognition (NER).** NER is part of a multiple Information Extraction (IE) processes. It automatically recognizes named entities from texts. These named entities could include people names, organization names, location names, time, and date. Early NER systems were utilizing rule-based approaches in identifying named entities; however, recent systems are making use of machine learning approaches. Nadeau and Sekine (Nadeau and Satoshi 2007) presented a detailed survey about NER systems and their techniques. The benefit that NER techniques can add to the proposed framework is that of another source where more resources (such as people and organization names) can be indentified from file contents. The newly discovered entities can then be mapped to the proper ontological concept.

**Hyperlink Extraction.** Hyperlink extraction techniques can be used to identify and extract hyperlinks from file content such as from textual and XML files. Each hyperlink would be an instance of the domain ontology URL which creates a Core Knowledge asserted association. For instance, if we have an XML file with a URL in its content, there would be an asserted association between the concept XML File from the File domain ontology and the URL domain ontology that may be represented by the relation containsUrl. Indeed, there might be also an indexing association between that URL resource and the XML file.

**Terms and Theme Extraction.** To gain a wider view of the semantic content of a document or a collection of documents, Terms Extraction techniques can be used to gather words from these documents that may be used to describe their content. Terms Extraction can be also extended to classify a collection of documents based on the underlying asserted semantic context of their content; which is known as Theme Extraction. This allows for further classification of documents into the ontology.
6. IMPLEMENTATION

In this paper, we have limited the scope of discussion to the encoding of potential evidence acquired from smart phone devices. The current architecture of the most popular smart phone operating systems (such as Android and iOS) allows the sharing of data to be done internally between applications and externally between multiple platforms. For instance, the contact manager of a smart phone may fuse contacts gathered from more than one source. Contacts from social networking accounts and instant messaging accounts may all be merged and presented in one central place. Such a feature allows for more relations to be discovered from various information resources. For this paper, we have simulated a demonstrative tool using TopBraid1 – an ontology editing tool that facilitates the creation, editing and querying of ontologies and knowledge bases. The Android operating system has been chosen to apply the proposed framework to.

6.1 Ontology for Smart Phones

Since we are dealing with smart phones, a set of ontologies has been developed to model this environment. Once again, for demonstration purposes, only four simplified ontologies are implemented. The proposed framework divides ontologies into two types, namely Domain Ontology and Application Ontology. Domain ontologies are used to model a particular concept in the environment. The domain ontologies that have been developed for this demonstration are: Contact, SMS Message, and Call Log ontologies. These three ontologies form the basis of the application ontology (the forth ontology), which interconnects between domain ontologies through a set of asserted relations. At this stage, a set of terms has been identified to form classes and subclasses of the ontologies as well as the asserted relations between them. The language used to model these ontologies is RDF Schema (RDFS) and Figure 4 illustrates the application and domain ontologies that model a small element of the smart phone environment.

---

1 Available at: http://www.topquadrant.com/products/TB_Composer.html
Figure 4 Representation of domain and application ontologies of some concepts in the smart phone environment

6.2 Annotation of Data

The main layer in the proposed framework that is responsible for annotating data from the evidence space is the File Wrappers layer. As discussed in the File Wrappers Section, this layer is device-dependent; which means that different devices require different file wrappers. This is directly linked to the structure in which the data is stored in. Since we are focusing on Android smart phones, most of the relevant data to our analysis is typically stored in SQLite databases. Therefore, a set of file wrappers has been implemented to interact with data that corresponds to the developed ontologies (i.e. for Contact, SMS Message, and Call Log ontologies). These file wrappers superimpose the asserted metadata of resources found in these databases. For instance, contacts information is typically disseminated among multiple tables in the contact database. In this case, the purpose of the file wrapper is to gather contacts information along with its asserted metadata and convert it to the RDF format. The annotated traces are then stored in the File Description database. A portion of the annotated trace for a single contact resource is shown in Listing 1 (represented in the Turtle format).
6.3 Association between Data

The type of asserted association represented for this demonstration is limited to the Core Knowledge Association. This type of asserted association finds asserted relations between resources identified by the annotator and stored in the File Description database as RDF triples. This is performed by the Concept and Relationship Extraction layer which takes the vocabulary defined by the ontologies as an input. The layer will try to map between the ontologies’ vocabulary and the RDF triples; and hence, populate the ontologies.

Reasoning engines take an essential role in finding asserted associations which are not explicitly defined in the knowledge space (i.e. in the RDF triples). Such asserted associations allow the examiner’s view to be explored to potential evidence that cannot be easily interpreted by human minds. It also facilitates asserted linkages between data from multiple domain ontologies. This, of course, requires further rules to be added to the reasoning engine. As this is an ongoing research, simple reasoning rules have been used. One of these rules is permitting a more abstracted view of the evidence. For instance, since we did not explicitly declare that all instances of the SMS Message class are also instances of the Message class (although SMS Message is normally a subclass of the Message class), querying the knowledge without the reasoning engine will not return any instances from the Message class. In this case, using a reasoning engine will allow such inference to be carried out. This becomes particularly handy as the Message class will combine all asserted messages from its subclasses; which permits the examiner to see asserted messages from various resources.

6.4 Enquire of the Knowledge

Once the ontologies are populated and stored in the RDF/OWL database, the knowledge base becomes ready to be interrogated. The query language used is SPARQL Protocol and RDF Query Language (SPARQL). SPARQL was developed to query graph-based datasets. Like SQL language, SPARQL allows the user to retrieve data that satisfies a given selection statement. Although for this demonstration we use SPARQL to directly query the knowledge base, for a practical analysis tool, it can be extended to be used with a graphical user interface where the examiner does not need to have a prior knowledge about SPARQL.

Enquiring of the knowledge base can be performed in three levels; within a single domain ontology, between two or more domain ontologies, and within the application ontology. Table 1 explains these three levels with an example for each one.
### Table 1 Examples of SPARQL queries for different levels of the ontology

<table>
<thead>
<tr>
<th>Level</th>
<th>Targeted Ontology</th>
<th>Example</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Domain Ontology</td>
<td>Contact ontology</td>
<td>SELECT ?name ?phoneNumber</td>
<td>This query retrieves all contact first names and phone numbers.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WHERE {</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>?contactResource contact:hasFirstName ?name;</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>contact:hasPhoneNumber ?phoneNumberResource .</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>?phoneNumberResource contact:hasNumber ?phoneNumber.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
<td></td>
</tr>
<tr>
<td>Multiple Domain Ontologies</td>
<td>Contact and Message ontologies</td>
<td>SELECT ?phoneNumber ?body</td>
<td>This query retrieves all SMS messages that have been sent by a contact</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WHERE {</td>
<td>with the phone number X.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>?phoneNumberResource contact:hasNumber ?phoneNumber.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>?msgResource message:hasSmsAddress ?phoneNumber.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>FILTER regex(?phoneNumber, &quot;X&quot;)</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>}</td>
<td></td>
</tr>
<tr>
<td>Application Domain Ontology</td>
<td>Application ontology</td>
<td>SELECT *</td>
<td>This query retrieves all call log records for all contacts.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WHERE {</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>?contactResource app:hasCallLogRecord ?callLogRecordResource.</td>
<td></td>
</tr>
</tbody>
</table>

### 7. CONCLUSION

In this paper, we have presented the ongoing development of a potential next generation of forensic analysis tools. The framework described in this paper is based on the use of semantic web technologies which aims to provide a semantic-rich environment to facilitate evidence analysis. Ontologies are used to model the environment under examination. This model encompasses resources and their asserted relations in a graph-based dataset by utilizing the RDF language. We believe that this tool can reduce some of the problems associated with the large volumes and complexity of data under analysis. In addition, to validate the effectiveness of the proposed framework, a simulation program has been developed to analyze contents retrieved from Android smart phones. Several ontologies have also been created to model some concepts involved in the smart phone environment. However, several limitations should be acknowledged in this proposal, such as the assumptions made throughout the development of the ontologies. To some extent, the ontology represents the interpretation of the developer (or a group of developers) towards a domain of discourse in the real world. Such assumptions may mislead the examiner and result in a wrong interpretation of the discovered traces. Therefore, the examiner must consider both the nature of the methods used and their limitations to avoid errors or omissions in their results. Another limitation lies in presenting too many traces to the investigator which may lead him/her astray. As such, previous knowledge about the case may help the investigator to shed the light on some key traces.
REFERENCES


A FORENSIC STUDY OF THE EFFECTIVENESS OF SELECTED ANTI-VIRUS PRODUCTS AGAINST SSDT HOOKING ROOTKITS

Sami Al-Shaheri (Al-shaheri_sami@hotmail.com)
Dale Lindskog (dale.lindskog@concordia.ab.ca)
Pavol Zavarsky (pavol.zavarsky@concordia.ab.ca)
Ron Ruhl (ron.ruhl@concordia.ab.ca)

Department of Faculty of Professional Education
Concordia University College of Alberta
Edmonton AB T5B 4E4
Canada

Keywords: System Service Dispatch Table (SSDT), Anti-virus, Rootkits, Memory Analysis, Volatility

ABSTRACT

For Microsoft Windows Operating Systems, both anti-virus products and kernel rootkits often hook the System Service Dispatch Table (SSDT). This research paper investigates the interaction between these two in terms of the SSDT. To investigate these matters, we extracted digital evidence from volatile memory, and studied that evidence using the Volatility framework. Due to the diversity in detection techniques used by the anti-virus products, and the diversity of infection techniques used by rootkits, our investigation produced diverse results, results that helped us to understand several SSDT hooking strategies, and the interaction between the selected anti-virus products and the rootkit samples.

1. INTRODUCTION

SSDT hooking is a prevalent method employed by some security tools, in order to set restrictions on accessing a system's resources [12]. For example, anti-virus products often hook the SSDT in order to scan the newly launched program [6][7]. Anti-virus products usually achieve this hooking by altering addresses stored in the Native SSDT functions, causing them to point to the anti-virus' routines. The anti-virus software then checks and verifies the system call source, blocking all suspicious calls, but otherwise invokes the SSDT functions without any changes to the system call [15]. Rootkits usually do something similar. In either case, knowing the table address of the SSDT is required in order to index the target functions, and to perform the SSDT hooks.

SSDT is "write-protected in Windows XP and later version of Windows" and that the "write protect (WP) bit in the CR0 control register" [7]. Thus, in order to perform the SSDT hooking, some rootkits modify the protection of the SSDT as a first step of attacking the SSDT, by clearing specific bits of the control CR0 register [8][4]. In [3], the authors illustrate two techniques for disabling SSDT write protection, and also note that "to subvert the write protection on the SSDT, we need to temporarily clear the WP flag" [3]. Rootkits usually use the function NtDeleteValueKey to change the value of a registry key, in order to modify the SSDT's protection. Rootkit developers use several SSDT hooking methods in order to compromise processes and system files, or to modify records in the SSDT, causing it to point to the rootkit itself [3][13][4].
Volatility is a powerful framework that can be used to investigate SSDT hooks. Volatility uses thrdscan to scan ETHREAD objects and thus to detect when rootkits make copies of the existing SSDTs and assign them to a particular thread such as ETHREAD.Tcb [24]. Volatility also uses the ssdt_ex plugin and the HookedSSDT tags to determine which SSDT functions are hooked [13][24][1].

In this research paper we systematically investigate the way in which representative anti-virus software and kernel rootkits interact with the SSDT, and with each other in terms of the SSDT. Our experimentation was conducted in four stages. In the first stage, we explored SSDT hooking by anti-virus products, independently of any SSDT hooking by the rootkits. Then we studied rootkits independently of anti-virus products. Next, we investigated the effects of SSDT hooking rootkits in an anti-virus protected environment. Finally, we investigated machines that had been infected with a rootkit which an anti-virus product was attempting to clean. Our results show that there is a broad range in the effectiveness of anti-virus products in their ability to protect against a rootkit's strategic SSDT hooking techniques, and that a deep and current understanding of that strategy is essential to anti-virus development.

In the next section we review related research. In the third section, we describe our experimental methodology and results. Section 4 contains our analysis of those results, and based on this, our recommendations. Section 5 is our conclusion.

2. REVIEW OF RELATED RESEARCH

In 2010, the Matousec.com team conducted a study to determine whether and how anti-virus products can be evaded. They tested 35 anti-virus products, and found that every anti-virus product in that sample which implements SSDT hooks was vulnerable, including big names such as Kaspersky Internet Security and Norton Internet Security [15]. Even subsequent to disclosure [16], Matousec.com found that only some of the anti-virus software developers had fixed their vulnerabilities. They developed what they call the Kernel Hook Bypassing Engine (KHOBE) attack, which allows the malicious codes to bypass the anti-virus's protection mechanisms [15]. KHOBE has two components the 'attacker' which attempts to invoke the system service; and a 'faker thread' which attempts to modify the parameters such as CLIENT_ID. If the modification occurs after the security check by the anti-virus and before the original service gets invoked, the attacker will invoke the service, and the bypass attack will be successful [15]. The Matousec.com research illustrated a combined attack where KHOBE uses three components attacker and two faker threads, in this case the attacker needs a scheduler to switch between the faker threads [15].

Matousec.com did not investigate SSDT hooking methods from a forensics perspective, nor did they study computer memory in order to verify their claim and provide evidences. While Matousec.com implemented an attack to demonstrate their claims about the vulnerabilities of SSDT hooking by anti-virus products, Corregedor and Solms implemented two rootkits that "could collectively disable antimalware programs" [4]. The first rootkit was designed to sabotage a Windows OS, and the second to disable antimalware programs. The paper discusses SSDT hooking with a focus on the rootkit's effect on the KeServiceDescriptorTable. However, there are four System Service Dispatch Tables, and other rootkits have different techniques; for example, the rootkit Blackenergy sometimes uses more than one table [2][10]. Furthermore, the authors (Corregedor and Solms) stated that there are "no other papers that [specifically explore] how rootkits are implemented", and they requested further efforts investigating additional malware to gain greater knowledge of how rootkits work [4]. Similarly to Matousec.com's research, Corregedor and Solms also demonstrated that SSDT hooking by anti-virus products is vulnerable to manipulation by rootkits. Whereas Matousec.com conducted an attack to verify this, Corregedor and Solms demonstrated the steps required to implement these two rootkits. In both cases, a forensics investigation analyzing the environment of the attack was not conducted.
Hsu et al. developed a rootkit that they called "antivirus terminator" [9]. They proposed a mechanism, called ANtivirus Software Shield (ANSS), to prevent anti-virus software from being terminated. They tested their developed rootkit on five anti-virus products, with the result that it successfully terminated all five. They tested the same five with their anti-virus protection mechanism (ANSS) installed, with the result that ANSS in each case protected the anti-virus software from termination. The operative component of the ANSS is its filter, which has many rules, such as the rule that any invocation of SSDT functions should be through the ANSS. In addition, the ANSS filter prohibits applications from using the function NtTerminateProcess to terminate the anti-virus software, and also prohibits any modification or deleting of registry keys via functions like NtDeleteKey, NtSetValueKey, and NtCreateKey. This work is similar to Matousec.com's research in that both showed that antivirus products are vulnerable when they hook the SSDT. It is also similar to that of Corregedor and Solms in that they developed rootkits. Again, a forensics approach was not conducted to collect evidence from memory.

Arnold [2] conducted a comparative analysis of rootkit detection techniques against several rootkits. Unlike the aforementioned studies, Arnold did conduct a forensics analysis. He used a hybrid approach, which included viewing the processor's utilization on the infected system and comparing it to a clean system, and analyzing the output of network-based detection tools (e.g., netstat and nmap) for both the infected and clean systems. In addition, Arnold examined the system files' locations and registry modifications of the infected system. Arnold's approach did not provide significant evidence about the functions of the SSDT that were hooked. Rather, it provided indirect indicators and warnings, such as the processor utilization, and presented statistics concerning captured network packets. While Arnold investigated the CPU and the network, we investigate the memory, as that is where the most direct evidence of an SSDT attack is located.

Alzaidi et al. [1] extracted digital evidence from volatile memory. They performed their work in a virtualized environment, and they compared two forensic techniques, live response and memory image analysis, by examining the detection capabilities of two forensic utilities, Redline and Volatility, "when the SSDT has been hooked by a rootkit". They showed "that the limitations of this live response utility [Redline] are due to the fact that it relies on system calls for detection of SSDT hooks". When Alzaidi et al. used Volatility, it was much more effective, and even detected that the live response utility Redline was infected by BlackEnergy's hook. They did not discuss the hooking and release of SSDT functions by anti-virus software, and they called for additional efforts to be made in analyzing SSDT "function hooking by antivirus software in cases where rootkits are also in place" [1].

Our research has pursued an approach similar to [1], in that we investigate digital evidence extracted from volatile memory using the Volatility framework as a memory image forensics tool. The following section provides an introduction to the SSDT's structure and the SSDT hooking methods used by anti-virus and rootkit software.

3. METHODOLOGY AND RESULTS

We employed well-known anti-virus products [19] and a set of publicly available rootkits that target the SSDT. The rootkit samples were collected from many forums, such as KernelMode.info [20] and Offensive Computing [18]. Our experimentation was conducted in four stages; each stage involved a number of individual experiments using virtual machines, where some of these machines were acting as cases and one virtual machine as the control. The purpose of the control machine was for comparison: it was our 'clean' machine, and allowed us to easily identify the SSDT hooks arising as a result of rootkit and anti-virus installation and interaction. Windows XPSP3 was installed on all machines (cases and control) in each stage. The host machine was running Window 7, was equipped with an Intel i7 Core, 2.20 GHz CPU, and 16 GB RAM. A 1.5 TB external drive was used to store the
memory images. In each stage memory images were taken from VMware workstation 9.0 machines, and memory analysis was performed on the control machine.

We selected five anti-virus products (AVG, Kaspersky, McAfee, Avast, Trend Micro) and three rootkits (Blackenergy, Haxdoor, Papras). For Stage 1, we analyzed the SSDT hooking methods of the selected anti-virus products. For Stage 2, we analyzed the SSDT hooking methods of the selected rootkits. Stage 3 studied the interactions between the anti-virus products and rootkits, when the former were first installed, and then the latter. Finally, in Stage 4, we did the converse of Stage 3: the rootkits were first installed, and then the anti-virus products. Let us now look at the experimental results in detail.

**First stage:** The primary goal of this stage was to analyze the selected anti-virus’ SSDT hooking methods. In our first experiment we installed AVG Anti-virus, and discovered that it hooked the function \textit{NtOpenProcess} which “opens a handle to a process and sets the access rights” [17]. AVG terminated threads by calling \textit{NtTerminateThread}, and \textit{NtWriteVirtualMemory} was called in order to prevent any unauthorized write to or overwrite of virtual memory [22]. We also noted that AVG hooked certain functions related to keyboard inputs, such as \textit{NtUserGetKeyState} and \textit{NtUserGetAsyncKeyState}. Such functions can help an anti-virus product to prevent malicious code from reading keyboard related information located in memory or the keyboard buffer [32]. Table 1 in Appendix A shows more fully the SSDT hooks by AVG.

In the second experiment we found that Kaspersky hooked a huge number of functions, but in this paper we focus our attention only on those functions related to the operation of the selected rootkits. Kaspersky hooked functions more critical than AVG, apparently in order to prevent registry manipulation using function calls such as \textit{NtRestoreKey}, \textit{NtDeleteValueKey}, and \textit{NtDeleteKey}. In addition, Kaspersky hooked functions like \textit{NtAdjustPrivilegesToken}, in order to enable or disable the access privileges to a specified token that contains information for a logon session [17]. In addition to hooking those functions related to registry manipulation, Kaspersky also hooked \textit{NtClose} to prevent malicious attempts to close handles to critical objects such as processes, e.g., an installation process, or even to shut down the system. Table 2 in Appendix A shows more fully the SSDT hooks by Kaspersky.

McAfee Anti-virus was the subject of our third experiment. Volatility was unable to detect any SSDT hooking by McAfee. We concluded that McAfee did not at all hook the SSDT.

In the fourth experiment during this stage, Avast was also found to be hooking many critical functions, such as the \textit{NtDeleteValueKey}. In addition, it created a key using the function \textit{NtCreateKey}, and then hooked the function \textit{NtDeleteValueKey}, in order to prevent any modification to that registry key. Avast in fact hooks more functions related to registry keys than the other anti-virus products.

In our fifth and last experiment during this stage we tested TrendMicro. The SSDT hooking method found here was similar to Avast’s: many critical functions were hooked by TrendMicro. With TrendMicro, all hooks point to a hidden driver, and any call of these functions is routed through that hidden driver.

**Second stage:** The primary goal of this stage was to analyze the SSDT hooking methods of rootkits, in order to prepare to investigate the interaction between those rootkits and the anti-virus products in the third and the fourth stages of our experimentation. The focus at this stage was on rootkits that employ SSDT hooking as part of their exploitation techniques.

Our first experiment at this stage was to launch the rootkit Blackenergy in a new virtual machine. We found that Blackenergy hooks the SSDT. Blackenergy starts the SSDT hooking process with the function \textit{NtDeleteValueKey}; this function is typically used by rootkits to modify or add values in a specified registry key. Blackenergy hooked this function in order to break the protection of the SSDT.
Blackenergy also hooked functions like \texttt{NtOpenKey} and \texttt{NtSetValueKey} in order to gain write permission to the registry [3]. \texttt{NtTerminateThread} was hooked, the purpose of which might be for thread injection [5], and the \texttt{NtWriteVirtualMemory} function was also hooked, to write to or overwrite virtual memory, in order to address injected code [22]. This SSDT hooking method by Blackenergy allows it to avoid detection and deletion. In fact, Blackenergy attempts to hide its driver, as shown in Figure 1 below.

The rootkit Haxdoor hooks fewer functions; notably, it hooks \texttt{NtOpenKey} in order to manipulate a registry key [17]. There are many versions of this rootkit available in public; some of these samples just use the \texttt{NtCreateProcess} function to create a new process [14]. We selected a Haxdoor version that hooks more functions, as shown in Appendix B Figure 2. Its also notable is the fact that, while Blackenergy was able to hide its driver, this was not so with Haxdoor, as shown in Figure 2 below.

We used the rootkit Papras in our last experiment of this stage, and we found Papras was hooking the functions \texttt{NtQueryDirectoryFile} and \texttt{NtQuerySystemInformation} in order to retrieve information from a specific file. Papras can therefore be used to retrieve the credential information, e.g. an online banking login id that may be stored in a buffer [23]. Papras does something similar to Blackenergy in terms of hiding its driver, as shown in Figure 3 below.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure1.png}
\caption{Blackenergy is pointing SSDT function to a hidden driver}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure2.png}
\caption{Haxdoor is pointing SSDT function to the driver \texttt{vbagz.sys}}
\end{figure}

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure3.png}
\caption{Papras is pointing SSDT function to a hidden driver}
\end{figure}

\textbf{Third stage:} During this stage our goal was to observe and analyze the interactions between rootkits and anti-virus products. Here we studied the effect of SSDT hooking by rootkits operating in an antivirus protected environment. The anti-virus products were installed first, and the rootkits were launched subsequently. We examined the SSDT functions hooked either by the selected anti-virus products (as discovered in the first stage of our experimentation) or by the rootkits (as discovered in the second stage), in order to observe how the anti-virus products and rootkits interact as defender and attacker within the system.

We began with AVG and Blackenergy. We observed AVG requesting notification about registry key changes using the functions \texttt{NtNotifyChangeKey} and \texttt{NtNotifyChangeMultipleKeys}. Notification was positive and the registry key was changed and the SSDT compromised. The reaction by AVG was to hook the same functions back. For example, the function \texttt{NtOpenProcess} was reclaimed by AVG. Unfortunately, Blackenergy was able to return the favour, and change the registry key again; it then hooked \texttt{NtDeleteValueKey}. The reaction by AVG this time was different: AVG used the \texttt{NtUserGetAsyncKeyState} in order to return the status of all keys at a given moment. Figure 2 in Appendix C shows the reaction of AVG in order to prevent the SSDT attack by Blackenergy.

This kind of reaction by AVG is limited in its effectiveness, as the rootkit will continue deleting the values of the keys in the registry using the function \texttt{NtDeleteValueKey}; it seems that AVG might avoid this vulnerability by hooking or disabling the function \texttt{NtDeleteValueKey}. However, many processes were belonging to AVG were infected, carrying out the rootkit's functions. Blackenergy was able to camouflage itself as \texttt{vmtoolsd.exe}, and also took control of the process \texttt{avgwdsvc.exe}, the AVG Watchdog Service. AVG’s SSDT hooking method, therefore, was ineffective at protecting the
system’s processes or even its own processes. Figures 3 and 4 in Appendix C show the infected processes.

We executed the same experiment with Kaspersky and Blackenergy. As we know from the first stage, Kaspersky hooks critical functions in order to prevent registry manipulation, such as manipulation of NtDeleteValueKey. We found that Blackenergy was unable to hook any SSDT function, because, as we knew from our previous experiments in the second stage, Blackenergy relies on NtDeleteValueKey, and this function and other critical functions were already hooked by Kaspersky. In general, we found Kaspersky was able to protect all processes, including its own processes, such as avp.exe. See Figure 5, Appendix C.

Our third experiment at this stage employed McAfee and Blackenergy. As we know from the first stage, McAfee does not use SSDT hooking. After installing Blackenergy, McAfee reported that it had detected and was able to remove the rootkit, yet this appeared to be only partly true, since McAfee continued to report this even after it had apparently attempted to remove the rootkit. We investigated this further from a process perspective. We found to be infected the process Mcagent.exe, which is a process belonging McAfee, designed to ensure that its virus definitions are up to date. Further, Blackenergy was able to infect other processes that belong to McAfee, such as the Mcshield.exe, which is McAfee’s process to monitor computer processes, files and the registry, in order to detect and prevent virus infection. Similarly, McSvHost.exe, known as McAfee Service Host, became infected, as was Mepvtray.exe, McAfee’s AntiTheft process. Finally, MOBKbackup.exe, the McAfee Online Backup Service, was also infected. Figures 7-11 in Appendix C show these infected processes.

Avast and Trend Micro were able to protect the SSDT and prevent these Blackenergy attacks, due to the fact that, like Kaspersky, Avast and Trend Micro hooked many critical functions, such as NtDeleteValueKey. Since the findings were similar to that of Kaspersky, we do not show the details in this paper.

The five selected Anti-virus products were able to protect against the other two rootkits, Haxdoor and Papras. Volatility didn’t show any SSDT hooking by Haxdoor or Papras when any of the selected anti-virus products were installed. For example, this sample of Volatility output is from a machine where Haxdoor and MacAfee were both in place. The SSDT tables are not occupied, because MacAfee is not using the SSDT, and yet Haxdoor was still unable to function while the anti-virus software was running.

```
Created: 2012-12-13 00:34:30
Exited: 2012-12-13 00:49:37
Owning Process: 0x81caf928 "
Attached Process: 0x81caf928 "
State: Terminated
BasePriority: 0x8
Priority: 0x10
TEB: 0x00000000
StartAddress: 0x7c8106e9
ServiceTable: 0x80552f60
[0] 0x80501b8c
[1] 0xbf999b80
[2] -
[3] -
```

**Fourth stage:** The purpose of this stage was, like the previous stage, to observe and analyze the interactions between rootkits and anti-virus products. We investigated machines that had been first infected with a rootkit, which we then attempted to clean with an anti-virus product.
We began by launching Blackenergy and then installing AVG. However, the installation process did not complete and the system began an automatic shutdown. Our analysis of the memory image revealed that Blackenergy was calling the function `NtShutdownSystem`; this explains why the system was closing down (see Figure 1, Appendix D). We explored further, from a process perspective, concentrating on setup.exe and explorer.exe. We found that AVG was unable to execute setup.exe and that it was not running, and we further found that explorer.exe was infected by Blackenergy. Figures 2 and 3 of Appendix D show the details.

Interactions between Kaspersky and Blackenergy were similar, but slightly different from the foregoing. Kaspersky does not hook the function `NtShutdownSystem`, and consequently the same thing happened here as with AVG: the system shut down, and Blackenergy didn’t allow Kaspersky to be installed. Our analysis of the memory image produced unexpected findings: Kaspersky was not successfully installed, but attempted regardless to hook the SSDT, presumably during the installation process. Blackenergy, however, already had control over the Native SSDT functions (KeServiceDescriptorTable), and maintained that control, while Kaspersky took control of the GUI SSDT functions (KeServiceDescriptorTableShadow). For details, see Figures 4 and 5 in Appendix D. TrendMicro was similar to the foregoing: Blackenergy called `NtShutdownSystem` to thwart installation of the anti-virus. Avast was quite different, however: Avast indeed hooked `NtShutdownSystem`, so that Blackenergy could not shut down the system, but in addition hooked various functions in order to ensure the completion of its installation without disruption. For example, Avast hooked `NtSetBootOptions`, `NtModifyBootEntry`, and `NtAddBootEntry`. See Figure 6 in Appendix D.

Finally, we note here in passing that all five selected anti-virus products were able to clean the other two rootkits, Haxdoor and Papras. We omit the details from this paper.

### 4. DISCUSSION AND RECOMMENDATIONS

We investigated rootkits that target the SSDT, and we found that these rootkits usually use more than one strategy to conceal an attack. For example, Blackenergy manipulated the registry in order to break the protection of the SSDT, and used the function `NtDeleteValueKey` to change the value of registry keys, in order to modify the SSDT’s protection. In our fourth stage we observed Blackenergy closing down the system in order to stop the installation process of the anti-virus product. Haxdoor and Papras employed SSDT hooking in order to steal sensitive information, using the functions `NtQueryDirectoryFile` and `NtQuerySystemInformation`. Some Anti-virus products employ the SSDT hooking to set restrictions on accessing a system’s resources. For example, some will hook the SSDT to scan any new launched program [6][7][21].

Anti-virus products like Kaspersky, Avast and TrendMicro protect registry keys by hooking the function `NtDeleteValueKey`, which can be effective in preventing manipulation of registry keys to break SSDT protection. Avast and TrendMicro created their own key using the function `NTCreateKey`, and then protected the created key and its value using the functions `NtDeleteKey` and `NtDeleteValueKey`. This may be effective in preventing attacks against the unused SSDT tables, and makes it difficult for rootkits to modify the protection of the SSDT by clearing a specific bit of the processor's CR0 register [3][4]. Instead of using this well-known value, Avast and TrendMicro create a new key with a new value.

Generally, hooking critical SSDT functions is essential for Anti-virus products. In our experiments we found that the SSDT hooking decisions by Kaspersky, Avast, and TrendMicro were most effective in terms of protecting the SSDT. On the other hand, AVG missed many critical functions, and SSDT hooking was not used at all by McAfee. Since what constitutes a critical function depends to a great degree on malware design, we emphasize that SSDT hooking by anti-virus products should be based
on a precise understanding of current rootkit design. It is noteworthy that a recent paper published by Anti-virus team members did not show a full understanding of Blackenergy's current design [11].

5. CONCLUSION
We investigated the effectiveness of selected anti-virus products in defending the SSDT against malicious hooking, and exhibited the use of forensics tools and techniques for the investigation of rootkit attacks in the presence of anti-virus software. We recommend careful attention to rootkit SSDT hooking design when developing anti-virus products.

REFERENCES


APPENDICES
Appendixes A, B, C, and D provide detail regarding the results of our four stages of experimentation.

Appendix A Anti-virus SSDT hooking

Table A-1 SSDT functions hooked by AVG

<table>
<thead>
<tr>
<th>Function</th>
<th>Driver name</th>
</tr>
</thead>
<tbody>
<tr>
<td>NtOpenProcess</td>
<td>AVGIDSShim.Sys</td>
</tr>
<tr>
<td>NtTerminateProcess</td>
<td>AVGIDSShim.Sys</td>
</tr>
<tr>
<td>NtTerminateThread</td>
<td>AVGIDSShim.Sys</td>
</tr>
<tr>
<td>NtWriteVirtualMemory</td>
<td>AVGIDSShim.Sys</td>
</tr>
<tr>
<td>NtUserGetAsyncKeyState</td>
<td>AVGIDSShim.Sys</td>
</tr>
<tr>
<td>NtUserGetKeyboardState</td>
<td>AVGIDSShim.Sys</td>
</tr>
<tr>
<td>NtUserGetKeyState</td>
<td>AVGIDSShim.Sys</td>
</tr>
<tr>
<td>NtUserSetWindowsHookEx</td>
<td>AVGIDSShim.Sys</td>
</tr>
<tr>
<td>Function</td>
<td>Driver name</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>NtAdjustPrivilegesToken</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtClose</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtConnectPort</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtCreateEvent</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtCreateMutant</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtCreatePort</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtCreateProcess</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtCreateProcessEx</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtCreateSection</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtCreateSemaphore</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtCreateThread</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtCreateWaitablePort</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtDebugActiveProcess</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtDeleteKey</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtDeleteValueKey</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtDeviceIoControlFile</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtDuplicateObject</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtEnumerateKey</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtEnumerateValueKey</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtLoadDriver</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtLoadKey</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtLoadKey2</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtMapViewOfSection</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtNotifyChangeKey</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtOpenEvent</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtOpenMutant</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtOpenProcess</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtOpenSection</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtOpenSemaphore</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtOpenThread</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtQueryKey</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtQueryMultipleValueKey</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtQueryObject</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtQueryValueKey</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtQueueApcThread</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtRenameKey</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtReplaceKey</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtReplyPort</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtReplyWaitReceivePort</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtReplyWaitReceivePortEx</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtRequestWaitReplyPort</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtRestoreKey</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtResumeThread</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtSaveKey</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtSaveKeyEx</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtSaveMergedKeys</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtSecureConnectPort</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtSetContextThread</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtSetInformationToken</td>
<td>klif.sys</td>
</tr>
<tr>
<td>NtSetSystemInformation</td>
<td>klif.sys</td>
</tr>
</tbody>
</table>
Table A-3 SSDT functions hooked by Avast

<table>
<thead>
<tr>
<th>Function</th>
<th>Driver name</th>
</tr>
</thead>
<tbody>
<tr>
<td>NtAllocateVirtualMemory</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtClose</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtCreateKey</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtCreateSection</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtDeleteKey</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtDeleteValueKey</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtDuplicateObject</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtFreeVirtualMemory</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtLoadDriver</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtOpenKey</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtOpenProcess</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtOpenThread</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtProtectVirtualMemory</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtQueryValueKey</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtRenameKey</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtRestoreKey</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtSetValueKey</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtTerminateProcess</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtUnloadDriver</td>
<td>aswSP.SYS</td>
</tr>
<tr>
<td>NtWriteVirtualMemory</td>
<td>aswSP.SYS</td>
</tr>
</tbody>
</table>

Table A-4 SSDT functions hooked by TrendMicro

<table>
<thead>
<tr>
<th>Function</th>
<th>Driver name</th>
</tr>
</thead>
<tbody>
<tr>
<td>NtCreateKey</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtCreateMutant</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtCreateProcess</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtCreateProcessEx</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtCreateSymbolicLinkObject</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtCreateThread</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtDeleteKey</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtDeleteValueKey</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtDuplicateObject</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtLoadDriver</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtOpenProcess</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtOpenSection</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtOpenThread</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtRenameKey</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtRestoreKey</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtSetSystemInformation</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtSetValueKey</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtTerminateProcess</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtTerminateThread</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtWriteVirtualMemory</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtUserSetWindowsHookAW</td>
<td>Hidden</td>
</tr>
<tr>
<td>NtUserSetWindowsHookEx</td>
<td>Hidden</td>
</tr>
</tbody>
</table>
## Appendix B Rootkit SSDT hooking

### Figure B-1 SSDT functions hooked by Blackenergy

<table>
<thead>
<tr>
<th>Entry</th>
<th>Function описание</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0041</td>
<td>NtCreateProcess</td>
<td>vhagz.sys</td>
</tr>
<tr>
<td>0x0042</td>
<td>NtCreateProcessEx</td>
<td>vhagz.sys</td>
</tr>
<tr>
<td>0x0043</td>
<td>NtOpenKey</td>
<td>vhagz.sys</td>
</tr>
</tbody>
</table>

### Figure B-2 SSDT functions hooked by Haxdoor

<table>
<thead>
<tr>
<th>Entry</th>
<th>Function описание</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0045</td>
<td>NtQueryDirectoryFile</td>
<td>vhagz.sys</td>
</tr>
<tr>
<td>0x0046</td>
<td>NtQuerySystemInformation</td>
<td>vhagz.sys</td>
</tr>
</tbody>
</table>

### Figure B-3 SSDT functions hooked by Papras

<table>
<thead>
<tr>
<th>Entry</th>
<th>Function описание</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0047</td>
<td>NtEnumerateValueKey</td>
<td>vhagz.sys</td>
</tr>
<tr>
<td>0x0048</td>
<td>NtQueryDirectoryFile</td>
<td>vhagz.sys</td>
</tr>
<tr>
<td>0x0049</td>
<td>NtQuerySystemInformation</td>
<td>vhagz.sys</td>
</tr>
</tbody>
</table>
Appendix C SSDT hooking interaction when anti-virus is installed before rootkit

Figure C-1 SSDT hooking with AVG and Blackenergy

Figure C-2 AVG's vmtoolsd.exe infected
Figure C-3 AVG's avgwdsvc.exe infected

Figure C-4 Kaspersky protects its avp.exe process from Blackenergy
Figure C-5 Kaspersky protects its Datamnn~1.exe process from Blackenergy
Figure C-6 MacAfee's mcagent.exe process infected by Blackenergy

Figure C-7 MacAfee's mcshield.exe process infected by Blackenergy
Figure C-8: McAfee's McSvHost.exe process infected by Blackenergy
Figure C-9 MacAfee's McPvTray.exe process infected by Blackenergy

Figure C-10 MacAfee's MOBKbackup.exe process infected by Blackenergy
Appendix D SSDT hooking interaction when rootkit is installed before anti-virus

Entry 0x0041: 0x823ba1a1 (NtDeleteValueKey) owned by
Entry 0x0047: 0x823b9e39 (NtEnumerateKey) owned by
Entry 0x0049: 0x823b9f52 (NtEnumerateValueKey) owned by
Entry 0x0077: 0x823b9d6f (NtOpenKey) owned by
Entry 0x007a: 0x823b9aa9 (NtOpenProcess) owned by
Entry 0x0080: 0x823b9b31 (NtOpenThread) owned by
Entry 0x0089: 0x823ba3e6 (NtProtectVirtualMemory) owned by
Entry 0x0091: 0x823ba5bd (NtQueryDirectoryFile) owned by
Entry 0x00ad: 0x823b9956 (NtQuerySystemInformation) owned by
Entry 0x00ba: 0x823b92fa (NtReadVirtualMemory) owned by
Entry 0x00d5: 0x823b9fcf (NtSetContextThread) owned by
Entry 0x00f7: 0x823ba08f (NtSetValueKey) owned by
Entry 0x00f9: 0x823b9ca8 (NtShutDownSystem) owned by
Entry 0x00fe: 0x823b9dc9 (NtSuspendThread) owned by
Entry 0x0102: 0x823b9c16 (NtTerminateThread) owned by
Entry 0x0115: 0x823ba370 (NtWriteVirtualMemory) owned by

Figure D-1 Blackenergy using NtShutDownSystem

State: Waiting:UserRequest
BasePriority: 0xd
Priority: 0xf
TEB: 0x7ffdf000
StartAddress: 0x7c8106f5
ServiceTable: 0x824543a8
[0] 0x8248f998
[0x41] NtDeleteValueKey 0x823ba1a1
[0x47] NtEnumerateKey 0x823b9e39
[0x49] NtEnumerateValueKey 0x823b9f52
[0x77] NtOpenKey 0x823b9d6f
[0x7a] NtOpenProcess 0x823b9aa9
[0x80] NtOpenThread 0x823b9b31
[0x89] NtProtectVirtualMemory 0x823ba3e6
[0x91] NtQueryDirectoryFile 0x823ba5bd
[0xad] NtQuerySystemInformation 0x823b9956
[0xba] NtReadVirtualMemory 0x823b92fa
[0xd5] NtSetContextThread 0x823b9fcf
[0xf7] NtSetValueKey 0x823ba08f
[0xf9] NtShutDownSystem 0x823b9ca8
[0xfe] NtSuspendThread 0x823b9dc9
[0x102] NtTerminateThread 0x823b9c16
[0x115] NtWriteVirtualMemory 0x823ba370
[1] 0xbf999b80
[2] -
[3] -

Figure D-2 The process explore.exe infected by Blackenergy
Figure D-3 The process setup.exe compromised by Blackenergy
Figure D-4 Kaspersky installation unsuccessful, but trying to hook the SSDT
Owning Process: 0x81f6a020 'explorer.exe'
Attached Process: 0x81f6a020 'explorer.exe'
State: Waiting:UserRequest
BasePriority: 0x8
Priority: 0x9
TEB: 0x7ffd4000
StartAddress: 0x7c8106e9
ServiceTable: 0x82321188

[0] 0x81f676a8
   [0x41] NtDeleteValueKey 0x823121a1
   [0x47] NtEnumerateKey 0x823111e9
   [0x49] NtEnumerateValueKey 0x82311ff2
   [0x77] NtOpenKey 0x82311d6f
   [0x7a] NtOpenProcess 0x82311aa9
   [0x80] NtOpenThread 0x82311b31
   [0x89] NtProtectVirtualMemory 0x823123e6
   [0x91] NtQueryDirectoryFile 0x823125bd
   [0xba] NtReadVirtualMemory 0x823122fa
   [0xd5] NtSetContextThread 0x82311cfc
   [0xf7] NtSetValueKey 0x8231208f
   [0xfe] NtSuspendThread 0x82311c89
   [0x102] NtTerminateThread 0x82311c16
   [0x115] NtWriteVirtualMemory 0x82312370

[1] 0xb999b80
   [0x7] NtGdiAlphaBlend 0xb1d4aec8 klif.sys
   [0xd] NtGdiBitBlt 0xb1d4a640 klif.sys
   [0xbf] NtGdiGetPixel 0xb1d4ae82 klif.sys
   [0xe3] NtGdiMaskBlt 0xb1d4a7f6 klif.sys
   [0xed] NtGdiPlgBlt 0xb1d4b786 klif.sys
   [0x124] NtGdiStretchBlt 0xb1d4a6a6 klif.sys
   [0x12a] NtGdiTransparentBlt 0xb1d4b016 klif.sys
   [0x133] NtUserAttachThreadInput 0xb1d4babbe klif.sys
   [0x143] NtUserCallOneParam 0xb1d4a60c klif.sys
   [0x17a] NtUserFindWindowEx 0xb1d4a374 klif.sys
   [0x17f] NtUserGetAsyncKeyState 0xb1d4a168 klif.sys
   [0x1a0] NtUserGetKeyState 0xb1d4a1b8 klif.sys
   [0x1cc] NtUserMessageCall 0xb1d4a2bc klif.sys
   [0x1db] NtUserPostMessage 0xb1d4a208 klif.sys
   [0x1dc] NtUserPostThreadMessage 0xb1d4a260 klif.sys
   [0x1ea] NtUserRegisterHotKey 0xb1d4ac78 klif.sys
   [0x1eb] NtUserRegisterRawInputDevices 0xb1d4a4ea klif.sys
   [0x1f6] NtUserSendInput 0xb1d4a320 klif.sys
   [0x211] NtUserSetParent 0xb1d4aa4a klif.sys
   [0x220] NtUserSetWindowLong 0xb1d4afbe klif.sys
   [0x225] NtUserSetWindowsHookEx 0xb1d4a018 klif.sys
   [0x228] NtUserSetWinEventHook 0xb1d4a0c0 klif.sys
   [0x240] NtUserUnregisterHotKey 0xb1d4ad90 klif.sys
   [0x250] NtUserWindowFromPoint 0xb1d4a474 klif.sys

Figure D-5 explorer.exe under control of Blackenergy’s Native SSDT functions; Kaspersky taking the GUI
<table>
<thead>
<tr>
<th>Entry</th>
<th>Offset</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x0059</td>
<td>0xb22f59f0</td>
<td>NtAddBootEntry</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0058</td>
<td>0xb22f5e20</td>
<td>NtAddVirtualMemory</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0057</td>
<td>0xb22f5e50</td>
<td>NtAddProcess</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0056</td>
<td>0xb22f6080</td>
<td>NtCreateEvent</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0055</td>
<td>0xb22f60f0</td>
<td>NtCreateEventPair</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0054</td>
<td>0xb22f6170</td>
<td>NtCreateFile</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0053</td>
<td>0xb22f6210</td>
<td>NtCreateImage</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0052</td>
<td>0xb22f6280</td>
<td>NtCreateIdentification</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0051</td>
<td>0xb22f62e0</td>
<td>NtCreateSection</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0050</td>
<td>0xb22f6320</td>
<td>NtCreateSemaphore</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x004f</td>
<td>0xb22f63a0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x004e</td>
<td>0xb22f63c0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x004d</td>
<td>0xb22f63e0</td>
<td>NtCreateThread</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x004c</td>
<td>0xb22f6400</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x004b</td>
<td>0xb22f6420</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x004a</td>
<td>0xb22f6440</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0049</td>
<td>0xb22f6460</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0048</td>
<td>0xb22f6480</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0047</td>
<td>0xb22f64a0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0046</td>
<td>0xb22f64c0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0045</td>
<td>0xb22f64e0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0044</td>
<td>0xb22f6500</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0043</td>
<td>0xb22f6520</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0042</td>
<td>0xb22f6540</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0041</td>
<td>0xb22f6560</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0040</td>
<td>0xb22f6580</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x003f</td>
<td>0xb22f65a0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x003e</td>
<td>0xb22f65c0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x003d</td>
<td>0xb22f65e0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x003c</td>
<td>0xb22f6600</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x003b</td>
<td>0xb22f6620</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x003a</td>
<td>0xb22f6640</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0039</td>
<td>0xb22f6660</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0038</td>
<td>0xb22f6680</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0037</td>
<td>0xb22f66a0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0036</td>
<td>0xb22f66c0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0035</td>
<td>0xb22f66e0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0034</td>
<td>0xb22f6700</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0033</td>
<td>0xb22f6720</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0032</td>
<td>0xb22f6740</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0031</td>
<td>0xb22f6760</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0030</td>
<td>0xb22f6780</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0029</td>
<td>0xb22f67a0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0028</td>
<td>0xb22f67c0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0027</td>
<td>0xb22f67e0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0026</td>
<td>0xb22f6800</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0025</td>
<td>0xb22f6820</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0024</td>
<td>0xb22f6840</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0023</td>
<td>0xb22f6860</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0022</td>
<td>0xb22f6880</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0021</td>
<td>0xb22f68a0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0020</td>
<td>0xb22f68c0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x001f</td>
<td>0xb22f68e0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x001e</td>
<td>0xb22f6900</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x001d</td>
<td>0xb22f6920</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x001c</td>
<td>0xb22f6940</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x001b</td>
<td>0xb22f6960</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x001a</td>
<td>0xb22f6980</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0019</td>
<td>0xb22f69a0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0018</td>
<td>0xb22f69c0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0017</td>
<td>0xb22f69e0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0016</td>
<td>0xb22f6a00</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0015</td>
<td>0xb22f6a20</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0014</td>
<td>0xb22f6a40</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0013</td>
<td>0xb22f6a60</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0012</td>
<td>0xb22f6a80</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0011</td>
<td>0xb22f6aa0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x0010</td>
<td>0xb22f6ac0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
<tr>
<td>0x000f</td>
<td>0xb22f6ae0</td>
<td>NtCreateTimer</td>
<td>owned by avsSrv.SYS</td>
</tr>
</tbody>
</table>

Figure D-6 Avast using booting functions and NtShutdownSystem
WORKSHOP & PANEL:
DIGITAL FORENSICS READINESS

Moderator:
Barbara Endicott-Popovsky
Director of Center for Information Assurance and Cybersecurity
University of Washington, USA

DESCRIPTION

The concept of forensic readiness for a system describes the capability of the system to efficiently collect credible digital evidence that can then be used in legal proceedings. Efficiency for digital forensics has been described in terms of cost since costs tend to be significant especially for systems that are not forensics ready. Credible digital evidence refers to data that has been collected and preserved through a process that does not invalidate the legitimacy of the data.

Forensic readiness is one of the few proposed forensics characteristics discussed in the forensics literature. Forensic readiness was proposed by Tan in 2001 in order to meet two objectives for systems used in digital investigations: 1) Costs should be minimized for incident responses and 2) An environment’s ability to collect digital evidence should be maximized. In his original paper on forensic readiness, Tan described many specific techniques for achieving digital forensic readiness including logging techniques, IDS data usage, forensic acquisition and evidence handling.

A later paper describes how forensic readiness can be built into an enterprise forensics program and outlines ten steps to achieving forensic readiness. The main point is that forensic readiness makes sense from a business perspective and can result in cost savings should there be an investigation. Enterprises should be actively collecting potential evidence such as log files, network traffic records, e-mail and telephone records prior to involvement in an investigation.

Others described different aspects of system forensic readiness including policies for enhancing digital forensics readiness, incorporation into existing response plans and making sure forensic readiness leads to sound investigation. Another perspective discusses ensuring that hardware devices used to capture forensic evidence are reliable enough to enforce forensic readiness. As the wide divergence of these studies illustrate, there is no one methodology or approach to enabling forensic readiness within a system or enterprise.

Although the concept of forensic readiness has been circulating for years, only recently are applications appearing in products. With civil and criminal cases relying heavily on digital evidence, it’s time to revisit forensic readiness: what does the term mean today? What progress has been made in defining it? What products and systems incorporate forensic readiness concepts in their design?

Please join us for a discussion of these ideas. Your contributions will assist in defining the state of the digital forensic readiness in theory and practice. The outcome will be a call for chapters in this specific area.
Subscription Information

The Proceedings of the Conference on Digital Forensics, Security and Law is a publication of the Association of Digital Forensics, Security and Law (ADFSL). The proceedings are published on a non-profit basis.

The proceedings are published in both print and electronic form under the following ISSN’s:

ISSN: 1931-7379 (print)
ISSN: 1931-7387 (online)

Subscription rates for the proceedings are as follows:

Institutional - Print & Online: $120 (1 issue)
Institutional - Online: $95 (1 issue)
Individual - Print: $25 (1 issue)
Individual - Online: $25 (1 issue)

Subscription requests may be made to the ADFSL.

The offices of the Association of Digital Forensics, Security and Law (ADFSL) are at the following address:

Association of Digital Forensics, Security and Law
1642 Horsepen Hills Road
Maidens, Virginia 23102
Tel: 804-402-9239
Fax: 804-680-3038
E-mail: office@adfsl.org
Website: http://www.adfsl.org
Contents

Committee ................................................................................................................. 4
Schedule ...................................................................................................................... 5

The Development of Computer Forensic Legal System in mainland China ................ 9
Fei Xu, Kam Pui Chow, Yonghao Mai*, Rongsheng Xu, and Jun Zhang*

Electronic Discovery and Proportionality in Minnesota: A Harbinger for the Federal Court System? 15
Vicki Luoma* and Milt Luoma*

Cybercrime and Punishment: An Analysis of the Deontological and Utilitarian Functions of Punishment in the Information Age 17
Karim Jetha*

Invited Paper: Resolving the Cloud Forensics Conundrum ........................................... 23
John Bagby*

KeyNote Speaker 1: Craig Valli ...................................................................................... 25
Identifying Peer-to-Peer Traffic on Shared Wireless Networks .................................... 27
EJ Jung*, and Simon Piel

Workshop 1 and Panel: Examining the Forensic Expert (John Bagby) .......................... 31
KeyNote Speaker 2: Jake Kouns .................................................................................. 33

Money Laundering Detection Framework to Link the Disparate and Evolving Schemes 35
Murad Mehmet, Duminda Wijesekera, and Miguel Fuentes Buchholtz

System-Generated Digital Forensic Evidence in Graphic Design Applications ................ 59
Enos Mabuto, and Hein Venter

Significance of Semantic Reconciliation in Digital Forensics ........................................ 71
Nickson Menza Karie*

KeyNote Speaker 3: Daniel Ryan .................................................................................. 81

An Image Forensic Scheme with Robust and Fragile Watermarking for Business Documents 83
Sai Ho Kwok*

Journey into Windows 8 Recovery Artifacts .................................................................. 89
Kenneth Johnson*

A Thematic Review of User Compliance with Information Security Policies Literature ....... 103
David Sikolia*

First glance: An Introductory Analysis of Network Forensics of TOR ............................... 107
Raymond Hansen*

An Ontology-Based forensic Analysis Tool ..................................................................... 123
Mohammed Ali Alzaabi*

A Forensic Study of the Effectiveness of Selected Anti-Virus Products against SSDT

Hooking Rootkits ......................................................................................................... 139
Sami Al-Shaheri*, Dale Lindskog, Pavol Zavarsky, and Ron Ruhl

Workshop 2: Forensic Readiness (Barbara Endicott-Popovsky) ...................................... 163

* Author Presenting and/or Attending