LEIA: The Live Evidence Information Aggregator
A Scalable Distributed Hypervisor-based Peer-2-Peer Aggregator of Information for Cyber-Law Enforcement

Irvin Homem
[ irvin@kth.se ]

Master’s Thesis

KTH – The Royal Institute of Technology
Stockholm University

October 2013
Stockholm, Sweden
Abstract

The Internet in its most basic form is a complex information sharing organism. There are billions of interconnected elements with varying capabilities that work together supporting numerous activities (services) through this information sharing. In recent times, these elements have become portable, mobile, highly computationally capable and more than ever intertwined with human controllers and their activities. They are also rapidly being embedded into other everyday objects and sharing more and more information in order to facilitate automation, signaling that the rise of the Internet of Things is imminent.

In every human society there are always miscreants who prefer to drive against the common good and engage in illicit activity. It is no different within the society interconnected by the Internet (The Internet Society). Law enforcement in every society attempts to curb perpetrators of such activities. However, it is immensely difficult when the Internet is the playing field. The amount of information that investigators must sift through is incredibly massive and prosecution timelines stated by law are prohibitively narrow. The main solution towards this Big Data problem is seen to be the automation of the Digital Investigation process. This encompasses the entire process: From the detection of malevolent activity, seizure/collection of evidence, analysis of the evidentiary data collected and finally to the presentation of valid postulates.

This paper focuses mainly on the automation of the evidence capture process in an Internet of Things environment. However, in order to comprehensively achieve this, the subsequent and consequent procedures of detection of malevolent activity and analysis of the evidentiary data collected, respectively, are also touched upon. To this effect we propose the Live Evidence Information Aggregator (LEIA) architecture that aims to be a comprehensive automated digital investigation tool.

LEIA is in essence a collaborative framework that hinges upon interactivity and sharing of resources and information among participating devices in order to achieve the necessary efficiency in data collection in the event of a security incident. Its ingenuity makes use of a variety of technologies to achieve its goals. This is seen in the use of crowdsourcing among devices in order to achieve more accurate malicious event detection; Hypervisors with inbuilt intrusion detection capabilities to facilitate efficient data capture; Peer to Peer networks to facilitate rapid transfer of evidentiary data to a centralized data store; Cloud Storage to facilitate storage of massive amounts of data; and the Resource Description Framework from Semantic Web Technologies to facilitate the interoperability of data storage formats among the heterogeneous devices. Within the description of the LEIA architecture, a peer to peer protocol based on the Bittorrent protocol is proposed, corresponding data storage and transfer formats are developed, and network security protocols are also taken into consideration.

In order to demonstrate the LEIA architecture developed in this study, a small scale prototype with limited capabilities has been built and tested. The prototype functionality focuses only on the secure, remote acquisition of the hard disk of an embedded Linux device over the Internet and its subsequent storage on a cloud infrastructure. The successful implementation of this prototype goes to show that the architecture is feasible and that the automation of the evidence seizure process makes the otherwise arduous process easy and quick to perform.

Keywords:

Digital Investigation, Incident Response, Digital Evidence, Automated Evidence Seizure, Collaborative Forensics, Live Forensics, Big Data, P2P Networks, Cloud Storage, Hypervisors
# Contents

List of Figures .......................................................................................................................... v

List of Tables ........................................................................................................................... vi

List of Abbreviations / Glossary .............................................................................................. vii

1. Introduction ................................................................................................................................. 1
   1.1. Distribution, Integration, Convolution and Information ...................................................... 1
   1.2. Organization of the thesis ....................................................................................................... 2
   1.3. Problem Definition .................................................................................................................. 3
   1.4. Purpose and Goals .................................................................................................................... 4
   1.5. Scope and Limitations ............................................................................................................ 4

2. Method of Research ..................................................................................................................... 7

3. The Current State of Affairs – The Problem Matrix ..................................................................... 11
   3.1. Prognosis of the Environment ............................................................................................... 11
   3.2. Digital Forensics: The Digital Crime Scene and Law Enforcement ................................... 17
   3.3. Digital Forensic Artifacts of Interest ...................................................................................... 24
   3.4. Malicious Activity: Classification & Detection ...................................................................... 30
   3.5. Storage of Information and the Semantic Web ....................................................................... 36
   3.6. Resource Availability, Distribution and Collaboration ....................................................... 43
   3.7. Virtualization and Hypervisors ............................................................................................. 61

4. The Concept of the Solution ......................................................................................................... 63
   4.1. Background: Recap and Summary of the Problems ............................................................ 63
   4.2. The Live Evidence Information Aggregator (LEIA) ............................................................ 64
   4.3. Related Work ......................................................................................................................... 67
   4.4. The Architecture – Details ..................................................................................................... 71
   4.5. Communication Protocol Considerations ............................................................................... 91
   4.6. Information Storage Paradigm ............................................................................................... 93

5. Towards Practical Implementation – The Proof of Concept ....................................................... 96
   5.1. The Experiment ....................................................................................................................... 96
   5.2. Tools & Setup ......................................................................................................................... 97
   5.3. Testing the Experimental Infrastructure ............................................................................... 102
   5.4. Outcomes of the Experiment ............................................................................................... 106
   5.5. Discussion of the Outcomes of the experiment .................................................................... 110

Abbreviations

Tables

Figures
5.6. Drawbacks of the Experiment.............................................................................................................. 111

6. Conclusions & Future work....................................................................................................................... 112
   6.1. Conclusions ....................................................................................................................................... 112
   6.2. Concerns around the LEIA Architecture........................................................................................... 113
   6.3. Areas of Future work ......................................................................................................................... 114

7. Bibliography ............................................................................................................................................ 115
List of Figures

Figure 1: Overview of the Digital Investigation Process ................................................................. 5
Figure 2: Overview of the Design Science Method as described in [5] .............................................. 8
Figure 3: Diagrammatic representation of the Stages of this Research following the Design Science Method .................................................................................................................. 10
Figure 4: The Dimensions arising in ubiquity [18] ............................................................................ 13
Figure 5: Taxonomy of Digital Investigation Artifacts by Component Decomposition .................. 29
Figure 6: Taxonomy of Digital Investigation Evidence Artifacts by Size and Characteristics ........ 29
Figure 7: The Original Semantic Web Architecture ......................................................................... 38
Figure 8: Modified Semantic Web Architecture .............................................................................. 39
Figure 9: Example of RDF format .................................................................................................. 40
Figure 10: The varying levels of control possessed by the different participants in a cloud infrastructure ............................................................................................................................ 48
Figure 11: Visualization of the NIST recommended Cloud Computing definition .......................... 49
Figure 12: Diagram representing the directions that automation of DFIR is taking ......................... 68
Figure 13: Pictorial depiction of the Live Evidence Information Aggregator (LEIA) ....................... 72
Figure 14: The Live Evidence Information Aggregator (LEIA) as a 4-Tiered Layer Model ............ 73
Figure 15: The Host-based Hypervisor (HbH) System and it's components ................................... 75
Figure 16: The General Incident Response Procedure from the P2P-da perspective .................... 81
Figure 17: The file structures involved in the Data Collection/Seizure Process .............................. 82
Figure 18: The typical flow of data in the Data Acquisition process via the P2P-da .................... 83
Figure 19: Schematic view of the Cloud-based Backend System components ............................ 85
Figure 20: HDFS Architecture that forms the basis of the Storage System in the CBB ............... 87
Figure 21: The File System Information of the Chumby device ..................................................... 98
Figure 22: Output from the /proc/mounts file indicating the various mounted filesystem information ................................................................. 98
Figure 23: The LEIA Architecture’s 2 main Network Compartments .............................................. 99
Figure 24: The Networking setup details of the experiment ............................................................ 100
Figure 25: The CBB Cloud Storage infrastructure using Hadoop HDFS ..................................... 102
Figure 26: Graphs portraying LEIA metrics with tests on HTC Incredible S ............................... 107
Figure 27: Graphs portraying LEIA metrics relationships on HTC MyTouch 4G Slide .................. 108
Figure 28: Graphs portraying LEIA metrics with tests on Samsung Galaxy Tab 2 ..................... 109
List of Tables

Table 1: Expected lifespan of data on certain types of media................................................................. 23
Table 2: The various lists of neighbours used in maintaining the Hierarchical Overlay ......................... 78
Table 3: Summary of types of messages exchanged in the P2P-da for Malicious Event Detection ....... 80
Table 4: Order of Incident Severity and the corresponding evidence category deemed necessary .... 88
Table 5: Specification of the nodes that comprise the Cloud-Based Backend System in the Experiment ........................................................................................................................... 101
Table 6: Specifications of the small scale (mobile) devices used in the testing .................................... 104
Table 7: “HTC Incredible S” Test Cases .................................................................................................. 105
Table 8: “HTC MyTouch 4G Slide” Test Cases .................................................................................... 105
Table 9: "Samsung Galaxy Tab 2 - WiFi Only" Test Cases ..................................................................... 105
Table 10: Results from Test Cases on "HTC Incredible S" ................................................................... 106
Table 11: Results from Test Cases on "HTC MyTouch 4G Slide" .......................................................... 106
Table 12: Results from Test Cases on "Samsung Galaxy Tab 2" .......................................................... 107
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFF4</td>
<td>Advanced Forensic File Format version 4</td>
</tr>
<tr>
<td>Any-Scale</td>
<td>Encompassing the range from Small Scale all the way up to and including Large Scale</td>
</tr>
<tr>
<td>CBB</td>
<td>Cloud-based Backend</td>
</tr>
<tr>
<td>CBB-DE</td>
<td>Cloud-based Backend Differencing Engine</td>
</tr>
<tr>
<td>CIA</td>
<td>Central Intelligence Agency</td>
</tr>
<tr>
<td>CTC</td>
<td>Counter-Terrorism Center</td>
</tr>
<tr>
<td>Cyberhacktivism</td>
<td>Malicious activity on the Internet with “activist type” of intentions</td>
</tr>
<tr>
<td>em-IDS</td>
<td>Embedded Intrusion Detection System</td>
</tr>
<tr>
<td>FRE Standard</td>
<td>Federal Rules of Evidence Standard</td>
</tr>
<tr>
<td>HbH</td>
<td>Host based Hypervisor</td>
</tr>
<tr>
<td>IoT</td>
<td>Internet of Things</td>
</tr>
<tr>
<td>IDA</td>
<td>Incident Data Archive</td>
</tr>
<tr>
<td>IDS</td>
<td>Intrusion Detection System</td>
</tr>
<tr>
<td>KBF</td>
<td>Known Benign Files (“kbf”)</td>
</tr>
<tr>
<td>L-KFHL</td>
<td>Local - Known-File Hash List</td>
</tr>
<tr>
<td>LEC</td>
<td>Law Enforcement Controller</td>
</tr>
<tr>
<td>LEIA</td>
<td>Live Evidence Information Aggregator</td>
</tr>
<tr>
<td>M-KFHL</td>
<td>Master - Known-File Hash List</td>
</tr>
<tr>
<td>NIST</td>
<td>National Institute of Standards and Technology</td>
</tr>
<tr>
<td>NAS</td>
<td>Network Area Storage</td>
</tr>
<tr>
<td>OWL</td>
<td>Web Ontology Language</td>
</tr>
<tr>
<td>P2P</td>
<td>Peer to Peer</td>
</tr>
<tr>
<td>P2P-da</td>
<td>Peer-to-Peer Distribution Architecture</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>RDF</td>
<td>Resource Description Framework</td>
</tr>
<tr>
<td>RFID</td>
<td>Radio Frequency Identification</td>
</tr>
<tr>
<td>SCADA</td>
<td>Supervisory Control and Data Acquisition</td>
</tr>
<tr>
<td>SAN</td>
<td>Storage Area Network</td>
</tr>
<tr>
<td>SS</td>
<td>Storage System</td>
</tr>
<tr>
<td>W3C</td>
<td>World Wide Web Consortium</td>
</tr>
</tbody>
</table>
1. Introduction

1.1. Distribution, Integration, Convolution and Information

Long standing debates on the scarcity of resources have for many years dominated discussions in the subject area of Economics. It is no different today, resources still maintain their nature of scarcity and no material source of malleable matter has yet proved to be irrevocably renewable, perpetually self-replenishing or eternally non-perishing. Absolute self-sustenance and independence of all nature is a concept uncharacteristic of man, technology and matter as a whole. It is with these concepts in mind that we strive rather more towards collaborative strategies to ameliorate the situation of scarce resources, than to attempt in futility to achieve aseity, that is, absolute self-sustenance.

The current state of affairs in the digital world seems to portray a picture of the future which seems to be moving towards having all devices connected to each other by some subtle means or another thereby allowing seamless communication and sharing of information. The boundaries of wires and even those of tethered wireless hotspots are quickly becoming artifacts of history at the helm of mobility and ubiquity. Normal objects of everyday use, including clothes, textiles, building materials and even household appliances, are also joining the ranks of the fully fledged computers of today. It is because of this that the numbers of interconnected devices or objects, as it may be, are projected to quickly get into in the billions and possibly more. “In the 2000s, we are heading into a new era of ubiquity, where the ‘users’ of the Internet will be counted in billions and where humans may become the minority as generators and receivers of traffic. Instead, most of the traffic will flow between devices and all kinds of ‘things’, thereby creating a much wider and more complex Internet of Things.” [1]

This concept of the “Internet of Things”, sometimes also referred to as “Ubiquitous computing,” imbibes a sense of freedom of mobility among these intercommunicating devices allowing objects to be in contact with others regardless of place or time in the most idyllic of circumstances. This mobility improves productivity and makes life easier in general and thus it has a positive influence on the self-growth and uptake of ubiquitous systems.

On the other hand, managing the intricacies of such systems can easily be overwhelming. The communication paths between several devices making up a single system with a specific goal at hand may be immensely complex. They may involve lots of back and forth communication, signaling feedback, requests for more information, storage of data or further processing, joining of new objects into the system and updates on the dropping out of others. The possibilities of the semantics of the information flows are infinitely large. Yet for the end user it may be the accomplishment of a simple task that requires only one step of interaction at some interface edge of the “Internet of Things.”

Social networks and collaborative technologies have also permeated the present information landscape with lots of chit-chat, socializing, planning, recreation and other more serious day to day activities now happening with the assistance of social media on the internet. The moral, immoral and amoral activities of today have also followed suit, taken shape and firmly set up camp on the social media sphere, eagerly urging all sorts of other new applications and uses of internetworks to come aboard.
Given the particular case of malicious, immoral or illegal activities occurring on such a diverse dynamic interconnected digital environment as described above, it is not impossible to perceive the need to efficiently trace back specific actions to their perpetrators in a bid to achieve accountability, responsibility or bring the purported culprit to book with reasonable certainty. This is what is considered part of the current mandate of today’s law enforcement in the physical world that we live in. However, the law-enforcement of today is considered to be largely focused on physically perceivable crimes, of more localized areas of concern in the physical environment, and not as yet fully equipped to take on crimes on the digital scene. For the most part, they are also considered to be quite overburdened with petty civilian criminal acts. [2]

In the digital environment, the sheer volume of information that is required to be stored for a forensic investigation to be carried out is immense and poses serious problems to economies of processing and storage capacities. This is not only because of scarcity of availability of processing power and storage space, but also because of the complexity of determining what could be useful, what needs to be stored, where it can be stored and how to maintain it in a forensically sound manner so as not to compromise the investigation. This can become especially difficult in situations where the devices are mobile, distributed and aptly capable - having their own processing and storage capacities. Additionally the aura of social interaction and information dissemination being abundantly abound on computer networks at the moment compounds the complexity of the problem. However, such a daunting scenario can also be turned around to be of benefit to the situation through the present interactivity of devices and aspects of collaboration, as well as delegation, through the assistance of peer to peer technologies.

A grim perspective is seen when the need for security, privacy, forensics and cyber-law enforcement, is placed within context of present “digitalized” human activities. The information realm of business, leisure, politics, education and healthcare seems to take on a ghastly face, in that it seems that there is simply too much information that needs to be captured, identified, analyzed, stored, reported or secured. There is some sort of an imminent information overload or an impending data flood that needs to be dealt with in order to make some headway towards the desired goal of efficiently and effectively collecting the relevant information to aptly perform and conclude a digital investigation. The need for specificity in discovering a specific element within the tsunami of information today is aptly quoted by Paul Pillar (Former CIA Chief of Analysis, CTC) in an analogy “it is not so much ‘looking for a needle in a haystack’ it’s more ‘looking for a needle in a stack of needles’.” [3] At face value it can already be seen that it is a rather arduous task for one single global system to claim to attempt to be able to perform these tasks within the given environment with the ultimate aim of cyber-law enforcement.

This is the state of the information age that we live in and solutions of how to make best use of such technologies, as well as how to maintain a suitable consensus for all, in terms of the common good of humanity, is sought after day in, day out.

1.2. Organization of the thesis

This study is organized in 6 broad sections. The first section describes the general outline, the problem area, the motivations, the goals, the scope and the limitations of this study. The second section describes the methodology of the study and its justification. The third section spans the introductory descriptions of the topics of concern that serve as the background information and basis for the discussion of the main subject material. It also outlines the dual problem of scarcity of
resources and abundance of information within the realm of modern networks and cyber-law enforcement.

The fourth section briefly describes the environment and the specific requirements of the system. It then goes into the detailed description of the solution, its architecture, communication protocols, data formats and security considerations. The fifth section outlines the implementation and testing of a partial proof of concept, as well as a description of the results of this implementation and testing. The sixth and final section presents the conclusions of this study and further work that may be done in order to realize this idea.

1.3. Problem Definition

1.3.1. Impetus

As briefly outlined in the introduction of this study, there is a vast amount of information traversing the internet superhighway and other related complex data networks. Human activity and day to day interactions, from the mundane to the mission critical, are slowly tending towards greater usage of these data networks both to the advantage of human society and to its disadvantage.

The focal point of this study centers upon the area of automating data collection and pre-processing for digital investigations on complex data networks such as the Internet of Things. Of primary concern in this study is the sheer amount of data that needs to be stored and pre-processed in the event of some criminal activity occurring. No doubt, this data needs to be availed in some form or another in order to realize that a security policy has been violated, as well as to provide some leads towards the culprit(s) and bring them to book. Thus, at the heart of any digital investigation, data must be captured and stored appropriately before being subjected to forensic analysis.

Following from this, the core research question that this study aims at answering is:

“How can relevant information on criminal activity be efficiently detected and rapidly dealt with (collected) in a digital investigation, given the scarcity of resources available and the complexity of information interactions and data networks of today?”

In all simplicity, an automated collaborative framework providing the possibility of capturing suspect data from heterogenous devices in realtime, filtering for relevance, providing storage as well as a platform for further analysis is what is sought after. If such a system is conceivably implemented globally, and can prove its mettle and efficiency, without infringing people’s privacy beyond rational extents, then the cyber-world could indeed become a safer place. Suffice to say there are numerous pertinent issues that need to be addressed in order to achieve this desired state of affairs – these are the core ideas to be discussed in this study.

1.3.2. Feasibility

Proposing an automated global system that aims for compatibility on a wide variety of devices running different platforms is a tall order. In order to maintain the feasibility of this study within the time and resource constraints of a master’s thesis project, the focus will be mainly on the architectural components that enable rapid and efficient acquisition of evidentiary data, maintenance of its availability and its integrity (forensic soundness). Aspects of analysis and processing of the data will not be dealt with in this study.
The study itself would begin with an analysis of the current and proposed cyber-realm and its complexities. This would then lead towards the provision of a detailed analytical description of the design and functionality of the different components of the proposed system, motivated by its environment, together with a hypothetical exposé of the system’s capabilities. Thus, the main result that is aimed for through this study is a theoretical design (framework) of the proposed solution. A limited functionality prototype of the system may be developed to demonstrate the feasibility; however actual implementation of the whole system is outside the scope of the proposed study.

1.4. Purpose and Goals

The main purpose of this study is to propose a collaborative solution aiming at capturing data related to digital criminal activity in realtime based on the current environment, while remediating the major problems of time and resource constraints that greatly burden digital forensic investigations and delay cyber-law enforcement. A framework of collaboration among devices is to be proposed and designed, aiming towards the shared processing and analysis of complex cybercrimes thereby cutting costs and resource usage, furthering the aim of speedier investigations. It is said that for a law enforcement system to be effective, justice needs to meted out as quickly as possible, as “Justice delayed is justice denied.”

In order to satisfy these aims, further smaller specific goals have been described in order to accomplish the study as a whole. These goals are as follows:

i. To portray the current cyber-society and its intricacies, based on personal experience, current events and forecasts of future projections from scientific literature, in order to pinpoint the problems and identify the particular requirements of a possible solution.

ii. To propose a distributed, global, collaborative, platform-independent solution that aims to increase efficiency in information gathering and processing, thus reducing the imminent time and resource constraints as has been seen in current times.

iii. To describe the entities, agents and elements; their relationships and communication protocols that enable the necessary speed and efficiency, while also ensuring secrecy, integrity, authenticity, privacy and availability within the system.

iv. To provide a small scale prototype with limited functionality to demonstrate the feasibility of the designed framework.

v. To suggest possible weaknesses, plausible solutions and possible future work that could forge such a system towards becoming a reality.

1.5. Scope and Limitations

The realm general digital investigation process can be summarized in four main processes, that is, Detection of Malevolent Activity, Seizure/Collection of Evidence, Analysis of the Data Collected and finally the Presentation of Valid Postulates (Hypotheses) as depicted in Figure 1 below.
Automation is currently suggested as one of the means in which to solve the problems experienced in today’s current investigation, however, to the author’s knowledge this has not yet been applied to the particular process of “Seizure / Collection of Evidence.” It is with this in mind that we focus on this particular area, while also laying out the landscape for the integration of the other processes in a larger scale automated system that LEIA is envisioned to be. Thus, it should be mentioned that we inevitably touch slightly on the subsequent process of “Detection of Malevolent Activity” as well as the consequent one of “Analysis of Data Collected” in order to give a fuller picture.

This particular process of “Collection of Evidentiary” data is to be addressed mainly in terms of enabling rapid & efficient acquisition of evidentiary data, maintenance of its availability and ensuring its integrity (forensic soundness).

As shall be discussed in Section 3 [3.2 - Digital Forensics: The Digital Crime Scene and Law Enforcement], there are various types of crimes, as well as a multitude of types of devices that could be involved in these crimes and even further a variety of types of data that could be considered of evidentiary value. This means that there are a large number of variables to take into account.

In terms of the types of crimes, we only take into account digital crimes that involve interactivity among devices, transfer of information or human interaction with devices related to other crimes of a more physical nature. Examples of crimes that are valid to this study include Man in the Middle Attacks, Spoofing attacks, Phishing attacks, Malware Distribution, Data Usurpation, Denial of Service attacks and the like. Other more person oriented crimes such as drug trafficking, rape, murder, harassment, espionage and child pornography can also be included insofar as they make use of devices that contain digital evidence.

For the purposes of the small scale, limited functionality prototype, the scope of the evidence artifacts to be captured as well as the type of device(s) to be experimented upon must be delineated. In terms of various possible types of digital evidence artifacts, we shall only deal with the collection of a single type of evidence artifact, that is, raw disk dumps of permanent storage from suspect devices. The choice of disk dumps is made because they are generally considered the richest in evidentiary data, as well as the fact that they are the most mature evidence artifact in standalone, static forensic seizure and analysis, and thus should be the first to be dealt with as we move towards automated and collaborative methods.

As shall also be discussed in Section 3 [3.3 - Digital Forensic Artifacts of Interest], there are various types of devices that could be involved in a digital crime. For the purposes of this study we will only deal with (small scale) mobile and embedded devices that run on some form of Linux platform. This choice of small scale devices is motivated by their current prevalence and high level of penetration among society. Additionally, as they are devices that are quite closely connected to human activities
in today’s world, they are bound to be targets of malicious activity. Furthermore, the field of mobile and embedded device forensics is quite new and pretty much uncharted. The choice for a Linux based platform is motivated mostly by the fact that most small scale devices use some form or flavour of Linux.

Finally it should be noted that this research aims at creating an innovative solution, making use of intrusion detection systems, hypervisors, peer-to-peer networks, the resource description framework and cloud storage systems, in a novel automated and collaborative manner in order to solve the problems that plague the digital forensics and cyber-law enforcement realm. To the author’s knowledge such a framework does not exist and thus emphasis is made on the fact this study is rather preliminary, explorative and could have rather grey boundaries. It is for this reason that the study is largely limited to personal experience with digital investigation cases, current cases in the media and forecasted projections from scientific literature on where technology and the cyberworld are headed.

All factors taken into consideration, this would mean that the final outcome will, for the most part, be a model or framework of a proof of concept. This forecasted product could possibly be put into some tangible form for testing at a later stage. This tangibility of the product is not part of the scope of this master’s thesis and is proposed as further work.
2. Method of Research

In terms of research paradigms that were considered, as is described by Hevner [4], there are 2 main contenders that characterize most research in the area of Information Systems, that is Behavioural Science and Design Science. The Behavioural Science paradigm “seeks to develop and verify theories that explain or predict human or organizational behavior”, while the Design Science paradigm “seeks to extend the boundaries of human and organizational capabilities by creating new and innovative artifacts.” [4]

Behavioural Science was deemed not appropriate for this study as we do not aim to attempt to understand, explain or predict human or organizational behavior. This study aims at developing a model that aims at solving some of the practical problems that are encountered in the area of Digital Forensics and Incident Response. Psychological, economic, political and social considerations of human interactions within organizations may be the hallmark of Behavioural Science, however, these are not of concern in this study. Rather we take a more technical stance aiming at bringing several different technologies that have differing application areas together in a novel way in order to improve the current technical deficiencies that affect the discipline of Digital Forensics and Incident Response. These latter elements of concern are materialized in the Design Science method in the form of artifacts, and thus this method is seen to be more appropriate for this study.

This study is directed towards being explorative research, while at the same time also attempting to be ‘constructive’ in nature. It aims towards crafting a novel solution to the identified problems. The nature of this study is founded on the principles of engineering a solution through bringing different aspects and elements from different subject areas together in order to build a plausible working model. The model developed through this study aims to curtail the problems identified, and attempt to provide an innovative near optimal solution, or at least edge closer towards the solution.

As a reference model for our research, we adopt the Design Science research method as described by Johannesson et al. in [5]. Design Science is described as “the scientific study and creation of artifacts as they are developed and used by people with the goal of solving practical problems of general interest” [5]. From this we can see that the end goal of a design science based study is to produce an artifact that aims to solve a practical problem.

An artifact is described by Johanesson et al. as “an object made by humans with the intention to be used for addressing a practical problem” [5]. There are 4 main types of artifacts that are outlined: Constructs, Models, Methods and Instantiations. In our case the artifact that this study aims to achieve is a model, which is described as an entity “that depicts or represents another object” [5]. Further to this, there are 3 types of models that are described by Johanesson, that is, descriptive models, prescriptive models and predictive models. For our purposes we shall be developing a prescriptive model whose essence is a “description of a possible future solution that can solve a practical problem” [5].

The general structure of the Design Science Method is shown in the diagram below:
Based on this framework of the Design Science research method we have prescribed 7 steps that more practically address the way in which this particular study will be conducted. These steps are annotated with the particular steps from the Design Science Method that they correspond to for better clarity. The flow of these 7 steps is described in textual format below:

1. Define the generic problem area *(Explicate the Problem)*

   This step aims at identifying the main problem area that this study focusses on. As the area of Digital Forensics and Incident Response is quite new and not yet mature, the generic problem area is identified from personal experience combined with observations from colleagues undertaking the Digital Investigation process. The issues around the stage of collection of data of probative value or potential evidentiary use, is the particular area of focus.

2. Review literature that is relevant to the areas of concern, its environment and the generic problem area *(Explicate the Problem)*

   Through document studies in the form of review of scientific literature and current popular media this step aims to lay down the stage that describes the environment that encompases Digital investigations today, and how this poses further problems to the area being studied. Literature to be studied includes current trends, predictions and forecasts of the digital environment. The state of the art of the technologies that are to be used, as well as the weaknesses of these technologies are also to be studied so as to get a firm grip on the problems that are experienced.
3. Identify specific areas of concern, their individual topical problems and review literature in detail on these areas. *(Describe the requirements)*

Based on scientific literature, popular media, personal experience and suggestions from colleagues (experts in the area), this step aims to outline the key elements that the proposed solution should address in order to alleviate the problems that are identified in the previous steps. These sources for the requirements are considered relevant and valid as the area is still quite new and very much still an art, rather than a science. Furthermore practitioners are also few in the area thus it is difficult to get access to a sizeable proportion to gather more meaningful results.

4. Propose a *theoretical solution* combined from varied areas. *(Outline the Artifact)*

In this stage the different technologies are brought together and a simple description is made of how the technologies aim to achieve the desired outcome. This is achieved through combining the technologies in different ways in order to achieve the goals. Informal discussions and brainstorming are used as input in order to select the most plausible formation.

5. Specify *detailed descriptions* of how the combined individual areas can work together to solve the problem and fine-tune the theoretical solution into a model/framework *(Design & Develop the Artifact)*

This step involves describing in detail the interactions between the different technologies, their functionalities, the algorithms and protocols that are to be used in order to achieve the desired solution. It involves a creative thinking and analysis process, rather than an information elicitation process.

The intended outcome is a model in the form of the architectural details, algorithms and protocols of that aid the system in achieving its intended goals.

6. Develop a limited functionality proof-of-concept based on the model *(Demonstrate the Artifact)*

As is described in the Design Science research paradigm, the demonstration of an artifact is more effective when it is limited to a single specific case. Thus, in order to demonstrate the developed model, a small scale, limited functionality prototype is to be developed. This proof-of-concept serves to show that the model can achieve at least some of its intended goals.

7. Test the model with practical experiments & describe hypothetical scenarios *(Evaluate the Artifact)*

This stage is intended to evaluate the extent to which the artifact achieves its goals. There are 2 main methods prescribed for evaluation in the Design Science research paradigm, that is *ex ante* and *ex post*. The *ex ante* method makes use of a theoretical analysis that relies on “informed arguments” and is preferred for highly novel artifacts, while the *ex post* method involves a full-scale evaluation of the entire artifact in its proposed environment. This study makes use of the *ex ante* method as the artifact being developed is novel. Furthermore, an *ex post* evaluation cannot be performed due to the need of a fully developed artifact, the lack of easy availability of the real-world environment as well as time and resource costs. For the *ex ante* evaluation, surveys, case studies, experiments, ethnography and theoretical
analysis are said to be the main methods of performing the evaluation. In this study we make use of experiments in order to evaluate the model.

A pictorial depiction of the process described above, with more specific detail, is shown in the diagram below:

Figure 3: Diagrammatic representation of the Stages of this Research following the Design Science Method

Further to the justifications for this method mentioned earlier, this Design Science based research method is also deemed to be appropriate for the study at hand because we aim to craft a model initially (i.e. an artifact), and later on implement this model into a working system in order to address the practical problems seen in the Digital Investigation realm. The Design Science research paradigm supports this form of model creation as well as the further development of the model into another form of artifact, that is, an instantiation (i.e. a tangible creation). Thus it is seen that Design Science is the most appropriate method of choice given the intended outcome of this research.
3. The Current State of Affairs – The Problem Matrix

Once, a grid was controlled by electromechanical and pneumatic devices. Now, it is controlled by computers running Windows or Linux, using the Internet Protocol (IP) to communicate. Wireless and Bluetooth capabilities are appearing in SCADA devices. These new features open an entire world of possibilities and an entire world of risks. [6] This is just a singular simplistic example of how the advent of the “Internet of Things” and the innumerable connectivity options presented today has permeated almost every known human activity. Communication and information sharing possibilities between people, devices, things and literally almost everything today is interlinked and intertwined. The variety of endpoints, interfaces, entities, protocols, media and topics of information is of behemoth proportions of variety and thus complexity. Literally, the information superhighway is increasing at an astronomic pace in terms of its lanes, vehicles, destinations and goods in transit among various other possible elements.

Such could easily span very many topics of discussion. Those that will be focused upon in order to give introductory ideas to serve as foundational concepts for the grounding of this discussion include: “The Internet of Things”, “Critical Information Infrastructures of Today”, “Digital Communication & Malevolent Actions”, “Forensics on the Digital Crime Scene” and “The Requirements of a Digital Investigation”

3.1. Prognosis of the Environment

3.1.1. The Internet of Things

The phenomenon of the “Internet of Things” (IoT) is not easily perceived or defined with specific precision as it covers a variety of intertwined concepts. However, some of the major concepts that it embodies include ubiquitous communication, pervasive computing and ambient intelligence [7].

Ubiquitous communication refers to the idea of elements interconnected through omnipresent networks allowing for exchange of information between these parties at any time and from anywhere. The key concepts here are the location independence and the time-activity impartiality. This means that within the IoT, communication can happen regardless of the location of the communicating devices or elements because they are “always” on and “always” connected. Time impartiality gives a sense that since the elements are always connected at every location, there can be communication between them regardless of the time of day or the activity being performed at that time. The connectivity is persistent through the different activities that an element could be performing. This is a notable characteristic seen in mobile telephony. Mobile phones are always communicating with base stations and their home location register and thus the mobile network operator, so long as they are on (powered), have a valid SIM card (or its equivalent) inserted and are within range of the radio waves of a recognized base station.

This is the essence of mobile telephony because of the intrinsic need of a persistent connection that needs to be available always and “everywhere” so as to be able to anticipate possible phone calls which can come at any time. It is a replica of the fixed telephony networks however mobility and ubiquity are now introduced into the equation. Due to the success of this aspect of technology, a paradigm shift has been created and many more elements of previously fixed/tethered networks are now appreciating the push towards mobility and ubiquity as part of their core abilities. One major technological advance that is seen as a movement towards this is the gradual replacement of
desktop and laptop computers with tablets (tablet personal computers) or pads, as the case may be. Computing power is now being packed tightly into more portable devices rather than the bulkier desktop equivalents so that the capacity of mobility and ubiquity can be further appreciated. The connectivity options still remain on the level of wireless communication protocols to maintain the mobility.

Pervasive computing embodies the concept of enhancing the environment around us with processing power [7]. This not only means that more and more everyday objects begin to possess some capability to capture, process, store or pass on data; it also means that an increasing amount of human activities around these objects are now being automated as well as undergoing a shift towards moulding these activities around the capabilities of computers. It can be said that computers have begun to invade almost all aspects of human activity today.

When looking at many of the core activities that mankind carries out today, one can, with almost sure certainty, identify some use of a computer along the line. Business, leisure, education, health, government and transportation have already visibly adopted some sort of automation or assistance through the use of computers or digital devices. A host of other rather mundane tasks such as finding a parking slot [8] [9] [10], turning on light-switches [11] [12], monitoring the weather [13] and even checking the tyre pressure [14] [15] [16] and wear-level of your car’s wheels [17] have now been automated through the use of computers.

The current list of application areas is almost endless and it won’t be long before other hitherto computer-unrelated activities also begin being integrated into the digital realm. The realm of tangible objects that contain some sort of digital device of computer is also expanding alongside the activities being automated. Today items of clothing, cooking devices, envelopes, books, washing machines and even car seats are coming equipped with some sort of microchip and radio transmitter. This has become very common especially with RFID (Radio Frequency Identification) tags and other wireless micro-sensors.

As computer components disappear towards nanometric sizes they begin to fit on the most unexpected objects and their presence therein sparks new application areas and possibilities of usage towards new previously unfathomed ideas. They thus begin permeating new areas that were not thought of before and further embed the IoT further into the fabric of human activities.

As a side note, it is worth mentioning that one of the major champions pushing the integration of computer systems into almost every human activity is the propensity of humanity for automation, making life easier, improving productivity and making optimal use of available natural resources. These aims can be seen in either one of two perspectives: promoting laziness or promoting productivity. All the same the end is similar, computers end up permeating almost every possible human activity that can be conjured up by man, either in their entirety or as an assisting mechanism.

The final major concept that is embodied in the IoT is Ambient Intelligence, which is described roughly as the ability of an inanimate object to be able to perceive and react to the environment within which it is located [7]. This means that objects can perceive physical stimulus through some form of sensors and actively interact with the environment producing appropriate actions depending on the case as it may be. This concept is rather important to the phenomenon of the IoT because of the perceivably numerous interactions that elements within the IoT may undergo during their normal span of activity in the world. The fact that these entities are intrinsically mobile and virtually ubiquitous within their environment, makes the capacity of interactions among different entities
expand enormously towards what could be described as an infinite set of multi-faceted stochastic interactions among the elements of the IoT, unless the environment is specifically bounded. Due to this possibility, the elements need to be aware of their environment and be able to process various kinds of stimuli, both known and unforeseen, so as to react in an appropriate manner suitable to their purpose.

Summarily it can be seen that within the IoT there is an inherent element of impartiality towards location, time (activity) and peer/respondent/neighbor. This is expressed pictorially in the diagram below.

![Diagram](image)

Figure 4: The Dimensions arising in ubiquity [18]

The impartiality measure above is not to mean that it is absolutely ignored and not taken into consideration at all. Contrary to this feeling is that the impartiality actually means that these factors are of great importance to the system at large, however they are somehow transparent, or made to seem transparent to active elements interacting or performing actions.

In this study the “Internet of Things” is a crucial element because of its characteristic distribution, ubiquity and interconnectivity of devices which provides a grounding structure upon which it is postulated that future technology will greatly make use of.

3.1.2. Critical Information Infrastructures of Today

“Knowledge is Power.” Among the greatest stores and modes of transfer of this knowledge today is information. Information is understood as the cornerstone of knowledge, as it is the most common depiction or representation of what we describe today as “what is known”, i.e. knowledge. It could be said that when an “intelligent” or smart element, that is one that exhibits the ability to either reason or make decisions, is equipped with information they are considered more apt, capable and empowered towards achieving their intended goal in a better way than they were prior to the availability of the information. (That is of course if the information is not deceitful in nature).
Popular opinion says that we are in the information age at the moment, given the wide expanse of all the flows of data and metadata traversing across the landscape of modern human activities. Most of the information is either stored in situ in some digital format within some storage medium, or is in transit from one information resource to another or towards an information “crunching” (processing) point.

The application areas, needs, usage, quantity, quality and expression of this information vary greatly depending on several factors that need not be discussed in depth in this study. However, one of the most important factors, that is well worth mentioning, is the concept of the context within which the information resides.

The context is important because it determines what information is useful to the information users as well as what could be useful to a third person attempting to understand what relevance the information had among the intended users in terms of its usage as well as its goal or outcome. This is especially important in digital forensics because the average scenario that a digital forensic analyst is presented with contains possible data sets of mammoth proportions that can almost not be fathomed with the human mind at one go. Somehow the forensic analyst has to sift through the possible data sets in order to extract that which is relevant and further postulate what the information represents. The context also affects how the information is to be interpreted or to be understood. Varying the context may vary the meaning of the information and thus the utility of the information either for the direct information consumers or to third-parties.

In most cases the utility of the information to its direct consumers or to third parties determines the level of importance that is accorded to it by either of these 2 identified parties. Information could be used towards a certain end that could be trivial or of critical importance. This depends on the viewpoint of the direct consumers, the aims of the system towards its beneficiaries and other stakeholders (direct or indirect), or the cost of loss of the information to adversarial parties.

The systems that are used in human activities of today which hold this information, or assist in its conveyance from one party to another, are what may be loosely termed as critical information infrastructures. The information that they bear within them, together with the implicit dependence factor that humanity has now grown on them, accords these systems this title.

The systems that could be identified as critical information infrastructures depend on the information consumers in focus, their level of dependence on the information systems (infrastructures), as well as the overall perceived cost of the loss or absence of the infrastructure by the consumers. When looking at the human activities of today, one could arguably classify critical information infrastructures into 3 broad categories:

- The Social and Personal sphere
- The Business and Productivity sphere
- The National and Administrative Sphere

These categories may not be entirely exhaustive and may be argued to have some overlapping features among them, however they are deemed to be suitable enough for this study.

**3.1.2.1. The Social and Personal Sphere**

The social and personal sphere may be described as the zone in which an individual maintains information about his person, and exchanges, due to his social nature, such information with other
similar individuals in the form of social interactions. The information in itself could vary greatly in content and focus, however, the essence is descriptive of the subject’s self, thoughts, ideas, ambitions, resolutions and perhaps what could be considered as personally identifiable information – that is information that could identify, or single out the owner of the information against a backdrop of other individuals.

Depending on personal, social, cultural, environmental, political or even economic factors this information, as a whole or in part, may be considered critical and in urgent need of safeguarding. Such is the case especially seen in social networks today where there are massive amounts of personally identifiable information as well as thoughts, ideas, expressions, ambitions and other statements that could otherwise be considered personal. The gain that a third party, with a malicious or simply intrusive intent, could obtain is immense in terms of the knowledge of the person/subject that they are focusing on. Such gain could be quantified in terms of names, email addresses, telephone numbers, residential addresses, locations visited recently, means of transport used, daily schedules, food preferences, circles of friends, clusters of constant communication, photos and even one’s current state of mind. The list is endless if one thinks of the permutations and combinations that could be generated from only the above mentioned information.

Other technologies that could be considered as part of the critical information infrastructures in the social and personal sphere include mobile, portable and ubiquitous devices and their services (E.g. phones, GPS’s, tablets), ad-hoc networks (like blue-tooth pico-networks), sensor networks and even further smart-objects of everyday use that are quickly becoming more popular, for example, home electronic appliances, clothing, jewelry and books. All of these items are slowly being “chipped” with RFID transceivers and minute wireless sensors with some processing power and storage for the sake of automation or personalization. As we use them more frequently in daily activities, they begin to hold small fragments of information related to our using them, and this could be pieced together to produce a larger picture of events. This could be of significant importance some day in a forensic investigation.

3.1.2.2. The Business and Productivity Sphere

The business and productivity sphere largely involves the area of activities that are aimed at efficient money or value generation through the production, use, or movement of readily exchangeable goods and services. In today’s business sphere, productivity is of great concern and is widely achieved through automation. This is spurred onward by fierce competition pushing various organizations to re-invent their business processes and innovate in order to keep ahead of the pack.

Among the more common ways of innovating is through tapping into the potential of the information available, streamlining the movement of possibly relevant information from the point of capture to the information consumers, of which primarily are the stakeholders. These stakeholders are usually the decision makers and could include the product/service consumers, the middlemen or the business owners and managers.

Another way in which information could be exploited as a means of innovation is through “data mining” which could be described as creating or generating information from sources where it did not exist previously in a perceivable format. This could involve extracting and combining metadata or in certain case, through observation of certain activities and collecting data about them.

Both of these means of making use of information in innovative ways assist in automation and productivity. There may be others including the improvement of the quality of the information and
these collectively drive down business’s costs in the long run and are also said to make life easier on the working force. The down side of this is that the amount of available information increases, thus increasing the possible risk factor in terms of leakage, loss of integrity, loss of control, inadequate storage and information overload. With this in mind, the need for more control and safeguarding of this information is seen as a necessity. This is of course if the information is not considered of public concern and various information-type-specific factors, or motives could affect the need for control and safeguarding.

Businesses in general hold a lot of information that they consider private and critical for their proper functioning. Further to this, as they embrace more automation, the amount of information increases and their business processes begin to heavily rely on the information, thus the sense of criticality of this information. Some of the information that businesses could hold and could be considered critical, depending on the circumstances includes human resource data including health data and salary information, organizational structure and means of communication; financial data including shares, investments, profits, taxes, bonuses, commissions, costing and pricing data; inventory information including asset information and location; Information and communication systems infrastructure; customer information, company trade secrets and other intellectual property. Other industry specific data could include supply chain and logistics information, marketing strategies and competitive strategies. Any of these, if found in the wrong hands, could cause significant damage, loss of reputation or spell immediate doom in possibly several other ways for a company or its stakeholders.

An example of the need and extent to which critical information infrastructure protection is taken in the business sphere is illustrated in [19], where the critical capabilities of mobile device management are elaborated upon. A vivid description of the infiltration of consumer controlled devices into the business environment is depicted leading to the loss of control of sensitive business information as the boundaries between personal and business information begin to blur. Thus the critical areas of management of this information are outlined in intricate detail towards the purpose of managing the infrastructure that is handling the sensitive data. It is not difficult to perceive the extreme criticality of such information infrastructures through elaborate nature of the descriptions of the capabilities needed of such a management system.

3.1.2.3. The National and Administrative Sphere

The National and Administrative sphere pertains to the area consisting of activities related to geographic nations or states, international organizations and the collective group of members that they are composed of as well as their administration. Here the collective activity of the members within these bodies as well as between these bodies, usually aimed at their sustenance, are of primary concern. Such activities include government systems, national treasury and economic systems, immigration and border control systems, judiciary systems, electoral systems, healthcare systems, military and defense systems, electricity systems, water systems, sewerage systems, transportation systems, natural disaster preparedness, early warning and recovery systems as well as weather monitoring systems, heating systems and ventilation systems in countries prone to adverse weather conditions.

Several of these aforementioned systems are categorized as SCADA systems (Supervisory Control and Data Acquisition systems) and are generally considered mission critical because of the scale of damage or disruption that could be caused if they were to fail. Today, in more developed countries,
primary amenities such as electricity, water, sewerage and transportation are managed through SCADA systems. Any failure of these systems, however small, would result in serious repercussions or backlashes from the public as these are considered to be government provided services through the so-called tax-payers contributions. Furthermore several thousand if not millions of people rely on these systems to function day in day out because of other activities, or business processes that they have tied to these. The slightest loss of information from these SCADA systems to malicious parties could result in these malicious parties gaining information that could allow them to cause havoc and disruption within the systems and thus the importance of keeping the information secure.

Due the large scale of parties that could be affected by the failure of any of these systems, national instability could result and even further the collapse of a nation into anarchy. This is considered to be a dreadful consequence which is greatly undesired and thus such system and their infrastructure need to be kept always in proper running order and their related information safeguarded and controlled by the appropriate parties at all costs.

Considering the aforementioned 3 major spheres of critical information infrastructures one can easily perceive that there is a lot of information that is considered critical for various reasons. This information could be used for malevolent purposes by malicious actors or conversely to detect malicious activities, trace and track down culprits so as to bring them to book.

By extension, the technologies that are used to convey, process or store this critical information are also to be considered critical not only because of their part that they play in the larger scheme of things in their normal activity, but rather because of the traces of information that they keep within them that could allow an investigator to piece together fragments of data from different sources (devices) in order to be able to redraw the events that lead to some malpractice or criminal actions. This is the area of greatest concern to the digital investigator: that is, gathering relevant data in order to assist in law enforcement.

3.2. Digital Forensics: The Digital Crime Scene and Law Enforcement

Digital forensics is 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 facilitation or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations” [20] [21]. The area within which this procedural discipline can be applied and is applied at the moment is vast given the immense infiltration of automated systems and digital devices in the realm of individual, business and governmental activities of today.

In the last few years the discipline of Digital Forensics has grown steadily from a relatively obscure tradecraft, experimental activity, hobby or even an “urban legend” into an integral and fundamental building block of investigations in our times. The tools, procedures, methods and skillsets that spurn out of the development of this field are readily being used by analysts and examiners within private investigation firms, businesses, law enforcement, military and even government intelligence agencies. [22] The need for development in this area is growing due to the overall effect of the digitalization and automation of human activities, as well as the widely spread collection, use and dissemination of otherwise hitherto unforeseen information.
3.2.1. Law and Law Enforcement

The norm of society, or communities at large, is to have some sort of laws, rules or regulations so as to ensure a modicum of balance or status quo among the members of the community. The stability of a community is generally founded upon such laws which are meant to maintain a mutually beneficial relationship among the members of the community as well as for the greater good of the society. The rules are also mean to restrain wayward members of the community who go against the grain, infringing other members’ rights or the integrity of the society as a whole.

This is exemplified typically in common associations that we may term as “communities” whether large as small. Such communities could include sports or recreational clubs, political parties, neighbourhood associations, families, companies and almost any other possible association where individuals come together towards a common goal or mutually beneficial goals. Typically they all have their own constitution, memorandum of understanding or rules of the game that they follow.

The internet society is no different. Strictly monitored structured communities and also less structured more flexible communities have sprouted on the interconnected web of communicating devices in the form of web forums, shoppers on e-commerce websites, gamers, social networks, users of certain device platforms, subscribers to certain cultural practices, tribes, religions and even sports. These are only a few. The list of possibly identifiable communities on the internet, or even further, on “the Internet of Things” is largely innumerable.

Even further, when one detaches themselves from the general internet, there are private networks partially or wholly disconnected from the normal internet. The ones of great concern here are those who are partially disconnected particularly for business, security or privacy reasons, however, they still use the internet for communication at some point or another. In this category one could include the different users of different proprietary or in-house applications or information systems that different companies make use of.

The common overarching factor among all different communities, as can be seen is that they all have rules that the members must subscribe to in order to be considered a member. This may range from implicit rules of a house in a family, to those rules of a web-forum, to the terms and conditions of a social network, the terms and conditions of a bank’s online-banking system, the employment contract of a business or a service contract between two companies.

In a similar way, it follows that the jungle of interconnected, intercommunicating users, devices and applications that we may term as the Internet or further – the “Internet of Things” (IoT) – also needs to have laws, rules and regulations to maintain a peaceful and mutual beneficial Internet Society/Community. This regulation and creation of laws for the Internet has been an ongoing effort for the last few years. It is still in progress and still needs a lot of work in order to globalize the jurisdiction and effect of the laws. At the moment laws governing Internet usage and activities on the Internet, as well as their enforcement are largely confined to the more developed countries, who have seen the need to do so due to the rapid increase of computer or network based crime mainly through the use of the Internet.

Due to the development of laws, rules and regulations related to digital crimes on the Internet (put in place to keep the variety of different participants therein from violating each other’s rights or disrupting the collective unity of the community), there has to be some form of body in place to implement the rules, ensure they are followed and enforce the necessary procedures and controls that are to befall those parties who willfully decide to go against the prescribed laws.
So far this law enforcement has been extended from the normal civilian law enforcement in the natural geographic societies that we live in. Police forces, military departments, the judiciary and government intelligence agencies form the bulk of this. Though this may be seen as a step in the right direction, in that the law enforcement already has the prior notion of what is expected of it from its prior day-to-day activity in the physical world, it may also be said that, due to the change in the actual environment, the law enforcement also has to change its outlook. This is because the rules of the game in the digital world are slightly different from those in the actual world, however the actors could possibly be same, that is, human perpetrators who also live in the same world as the civil law enforcement. The significant differences between the real world and the digital (virtual) world shall be pointed out further on in this document.

Due to the rules of the game being slightly different, the law enforcement also has to adapt to the change in environment. This is mainly seen in the range of forms of infarctions related to digital and cyber law, the types of proof or evidence that is needed to convict a perpetrator of such unlawful actions, as well as the level of acceptability or admissibility of such evidence in a court of law depending on the relevant local or international laws or law enforcement practices.

This is where the specialized discipline of Digital Forensics comes in to assist in law enforcement related to crimes of a digital or cyber nature. It assists, equips and provides the civilian law enforcement with a different set of tools, procedures and perspective in order to tackle crimes that have occurred in a semi-real dimension. “Semi-real” meaning that for the most part the actions or events and damage that were conducted and considered unlawful are usually not really tangible per se.

The development of Digital Forensics discipline is obviously seen to be more vigorous and potent in the further developed countries. This is because of their maturity in the use of information systems and automation in everyday activities, and thus the higher likelihood of malicious behavior (that could otherwise occur among normal human activities) also now occurring within the digital sphere. Thus the demand for Digital Forensics, its development and its usage, is bound to be seen as a necessary requisite component of an investigation in the modern world especially in the more developed nations.

Less developed countries are not to be left behind and are quickly following suit especially in the use of mobile devices, due to their relative cost-effectiveness and penetration of relevant mobile device based information services. It is only a matter of time before the lesser developed countries mature and begin to embrace digital technology use and automation. The legal systems in these areas will thus also have to begin incorporating relevant digital and cybercrime laws as well as their enforcement. This would in turn increase the need for Digital Forensics and increase the worldwide demand and aid in a better understanding of this field of study.

Most legal systems and law enforcement today observe the maxim of “Innocent until proven guilty” first stated as "Proof lies on him who asserts, not on him who denies" [23]. This means that any person or organization arrested for being alleged to have perpetrated a criminal activity is initially assumed to be innocent, but held in restraint until damning evidence can be produced, otherwise the subject is acquitted if a reasonable amount of time has passed and no worthy evidence is forthcoming.

In the same way that this applies to civil and criminal law, this also applies to the digital and cyber legal systems and law enforcement. A person or organization who is alleged to have participated,
orchestrated or been involved with a crime in the digital sphere, or using digital implements would be remanded temporarily pending the surfacing of relevant proof of the fact, or otherwise. However in the case of such crimes of a digital nature it is highly likely that the evidence is also of digital nature perhaps hidden among terabytes and terabytes of information or distributed among various systems on an interconnected network. The process of acquiring such evidence in a forensically sound manner from various distributed devices physically and geographically dispersed is considered a daunting task. Even further the sought after data might be hidden in loads of other similar data as such as network traffic data; or the platforms upon which the data may reside may be of an exotic character of which requires equally exotic procedures of extraction.

Just in the same way that technology is continuously changing and new forms of technologies, protocols, platforms and applications are mushrooming everywhere – being combined, merged and mashed-up in countless different innovative ways – so is the critical information that may need to be extracted for the execution of an investigation. The possible evidence that needs to be found and extracted could be permuted in several different ways camouflaging it among the chaos of the information super-highway. The task of collecting evidence suitable for use in digital investigations of today is thus seen to be immensely difficult, complex, dynamic and time-consuming.

This poses a major hurdle to the law enforcement agencies, in that the number of cases involving digital investigations are increasing, and at the same time the amount of information collected so as to undergo analysis is also on the rise. The complexity of collection of evidence, filtering for relevant information and analyzing it is also on the increase. This means the procedures of applying Digital Forensics techniques on an investigation are becoming more and more time-consuming.

This poses a visible problem of backlogs of work occurring and thus criminal cases possibly being thrown out because the critical pieces of evidence to be found cannot be found in a reasonable time frame, or even found at all, within the colossal set of possible information that can be collected from a digital crime scene. “In the UK you have 28 days in which to get your case together before a suspect must be released, which isn’t a lot of time to look through what might be terabytes of data in a single case. The average backlog of data awaiting forensic analysis in the UK is 18-24 months across the 43 police constabularies, according to comments made by the Metropolitan Police Central e-Crime Unit at InfoSec. [24]

Logically, one could say that the same immensely powerful technology and its advances that are causing the problems should be turned around in some way so as to be the solution to this visibly evident problem. This is what this study intends to attempt to do.

3.2.2. The Digital Crime Scene

The concept of traditional crime can be readily understood as any action that deprives another person or entity of what justly due to them. Such actions come under several terms that could include murder, theft, rape, exploitation, fraud among others. These are generally considered as unlawful activities that occur in the real world.

Digital crime on the other hand is sometimes considered a little more difficult to perceive as its realm of activity is not always in reality. In many cases such illicit activities happen on a virtual platform while making use of logical/virtual facilities such as data and connectivity. However, this does not necessarily have to be the case. Digital crime does not necessarily have to happen with the confines of some sort of virtual reality, such as on a network, within volatile memory or within a hard disk. In order to be classified as such, digital crime needs only to have a computer (or some
variation of one) involved in the crime, whether as a tool to facilitate the crime, as a target of the crime, or as part of the effects of the victim, the perpetrator or the crime scene itself.

As such traditional crimes can also be considered on the digital realm in the cases where computers, mobile devices, GPS or their components are related in some way to the traditional crime. Thus murders, theft, rape can also be found to be digital crimes if there are digital devices involved. On the other hand there are such crimes that are solely found on the digital realm such as Man in the Middle Attacks, Spoofing attacks, Phishing attacks, Malware Distribution, Data Usurpation and Denial of Service attacks. Such illicit activities have originated from the digital realm, among the various interactions between computers on networks involving digital data exchange, and thus may be termed as purely digital crimes.

The digital crime scene is said to be quite similar to the traditional crime scene in some respects. In both types of crime scenes similar steps of planning, preparation, a methodical approach and a set of specific skills are required. However there are also some distinct differences that make the approach to handling such investigations rather different. These differences mainly arise in the location in which the evidence could be present, the forms in which the evidence may found in and the size of the surroundings of the crime scene that need to be captured for analysis.

The location of evidence in the traditional crime scene is usually found somewhere in the vicinity of the crime or the suspected culprit. These locations can be reasonably delineated. However, in the case of the digital crime scene evidence could be found at the crime scene, but more often than not is found spread through the ecosystem of computer systems and devices that directly or indirectly were used in the criminal act. The geographical spread is potentially enormous.

The form which the evidence takes is also a point of variance. In traditional crime scenes it usual takes the form of tangible, humanly perceivable or recognizable artifacts that the reasonably untrained eye can see. This is usually in the form of fingerprints, footprints, strands of hair, blood and a host of other personally identifiable characteristics. In digital investigations the evidence is usually referred to as “digital evidence” and in the simplest form can be represented by IP addresses, files, pictures, server logs, software applications, processes in memory, packets in network communication streams, call logs, emails, SMS’s, geographic coordinates and even software application artifacts. These are only some of the current commonly identified sources of evidence. With the dynamic nature of technology, the lack of standardization of digital data formats and rapid changes in the state of the art, new forms of attributive digital evidence are sure to be developed or discovered.

The size factor is another area of difference. In a normal traditional crime scene the evidence and the environment within which it is found is photographed, bagged, tagged and loaded into lorries – to be stored at a secure guarded facility. With digital evidence, even though the physical size of devices may be acceptably small, it is the amount of data on these devices that constitutes the size of the evidence. The amount of data in the form of binary digits can be depressingly massive. It is even difficult to envisage that in some cases, a part of the Internet may be part of a crime scene, and capturing this, given its interconnected nature, could be significantly difficult. The perception of size of the possible data or items at the crime scene is difficult to quantify in a digital crime scene.

Particularly in digital crime scenes and digital investigations, digital evidence is primarily categorized into 2 main groups. These 2 categorizations are: Persistent and Volatile Data. In colloquial terms sometimes these 2 categories are also referred to as static and dynamic data respectively.
3.2.3. Persistent and Volatile Data in Computer Forensics

Persistent data may be described as data that is stored on some medium on a more permanent basis due to the nature, capability and use of the storage medium, as well as the effects of the environment with which it exists – whether debilitating or promoting longevity of the stored data. It is considered to be static and non-changing while it is not in active use, or being accessed by any party. This means that persistent data is meant stay as-is after its last access, as well as to outlast the loss of power to the storage medium, maintaining its state without any form of significant natural deterioration. Forms of media that are considered as containers of persistent data would include hard disks, tape drives, USB drives, memory sticks/cards and even floppy drives. The data on these storage media could include data files, configuration files, server logs, web history logs, error logs, backups, memory dumps, database files and application executables among other data artifacts of potential interest.

Volatile data, on the other hand is termed as data whose lifetime is considerably very short, or is prone to rapid and dynamic changes due to its nature, or the capabilities and use of its medium of storage, or effects from the environment within which it exists. Volatile or dynamic data in essence changes very often and thus could be an important indicator depicting the reality of events that are occurring at the time of seizure. This is in contrast to persistent data which usually is indicative of past events that have more often than not achieved their completion. It is thus important to capture snapshots of live running systems in the least intrusive manner in order to capture the state of systems, should they still be in the process of orchestrating some criminal or malevolent actions. This is because the evidence of the actions could be easily captured and presented from such.

Volatile media of storage within which digital evidence could reside in include registers, cache memory, peripheral device memory, main memory (RAM) and network peripherals their physical media and intermediate devices. [27] The potential evidentiary data that they may be used to extract, though incognito as the case may be, include currently running processes, memory resident malware, IP addresses, active connections and ports, covert channel messages, network packets and perhaps a host of other yet to be classified, though interesting artifacts.

Discussions related to digital evidence identification and collection within a digital crime scene are never complete without the mention of the concept of the Order or Volatility (by Farmer and Venema) [27] being mentioned. The Order of Volatility describes a set of best practices that postulates that among the digital evidence or artifacts of interest at a digital crime scene, the ones that have the highest volatility should be given the highest priority and should be collected first and with extreme caution not to affect these artifacts themselves, or those with lower levels of volatility still awaiting collection.

Farmer and Venema implied the following order of collection of data from their respective media sources from highest priority to lowest: Registers, Caches, peripheral memory, main memory, network state, running processes, disks, floppies, backup media, CD-ROMs and finally printed out material. This is derived from the Order of Volatility table that they suggested [27], as depicted below.

<table>
<thead>
<tr>
<th>Data Source</th>
<th>Timeframe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registers, peripheral memory, caches, etc.</td>
<td>nanoseconds</td>
</tr>
<tr>
<td>Main Memory</td>
<td>nanoseconds</td>
</tr>
<tr>
<td>Network state</td>
<td>milliseconds</td>
</tr>
<tr>
<td>Running processes</td>
<td>seconds</td>
</tr>
</tbody>
</table>
3.2.4. The Daubert Standard, Integrity and Admissibility

The Daubert Standard (1993) is a set of standards that have been used a lot in the recent past to analyze the admissibility of evidence in a court of law. There are several other standards, of which those which are also notably mentioned are the Frye standard (1923) and the Federal Rules of Evidence (FRE) Standard (1975). Most of these standards have come about as the result of precedents set at legal trials. The Frye Standard arose from *Frye v. United States* (1923), and the Daubert Standard arose from *Daubert v. Merrell Dow Pharmaceuticals* (1993) [28]. The FRE standard is the only exception and it was developed by the United States Supreme court outside any specific trial to supersede the Frye Standard, which was considered inadequate at the time.

The Daubert standard particularly delineates the criteria for the admissibility of expert witness testimony in a court of law [29] [30]. The criteria are as follows:

- The evidence must be *relevant* to the legal proceedings at hand, that is that they must be related to the consequences and the circumstances of the trial.

- The evidence must be *reliable*. Considerations for the level of reliability are judged along the following 5 criteria:
  - The evidence, theory, method or technique upon which the testimony is based must be falsifiable and able to be subjected to empirical testing.
  - It must be subject to peer review.
  - The known or potential error rates must be available.
  - There must be known standards that are maintained and in operation governing the theory, method or technique in question.
  - The theory, method or techniques used must be generally accepted in the relevant scientific community.

The Federal Rules of Evidence (FRE) have since been updated (in April 2000 and April 2011) particularly in “Rule 702” to include the principles brought out through the Daubert Standard.

The Daubert Standard is particularly important in Digital Crime Investigations because when evidence of a digital nature is brought to court, forensic analysts and practitioners from the field of Computer Science may be brought to court to dissect the digital artifacts presented. Furthermore, the methods in which the artifacts where collected and maintained up to the point of presentation at the trial could also be put to question. Thus, the admissibility of digital artifacts in a court of law is of great importance because evidence presented may be discounted on technical grounds.

The integrity of the evidence from the point of seizure to the point of presentation therefore becomes a very important factor. This is usually termed as the “chain of custody” which is the maintenance of a means of verifying that the evidence has not changed as well as ruling out, through proof, the possibility of the evidence having changed in transit. At the moment this is done
through using one way hashing algorithms to prove that the digital artifacts are still in their original form that they were collected in.

### 3.3. Digital Forensic Artifacts of Interest

Approaching the problem of identifying vastly varying and drastically dynamic sets of artifacts from a perspective of a crime scene within the realm of the “Internet of Things” suggests that the possible quantities of potentially useful artifacts of interest could be enormous. Collecting all the information that could be deemed useful could be said to be close to impossible and naturally very difficult.

However, attempting to classify and collect everything from a digital crime scene and then later looking for anything that could be useful is said to be a more intelligent option, rather than intently looking directly for something specific among a jungle of information. Quoting Darryl Zero on “The Zero Effect” in [27] by Farmer and Venema: “When you go looking for something specific, your chances of finding it are very bad. Because, of all the things in the world, you’re only looking for one of them. When you go looking for anything at all, your chances of finding it are very good. Because, of all the things in the world, you’re sure to find some of them.”

With this in mind, the seemingly best option is to attempt to classify all the possible artifacts of interest that could be found in a digital crime scene in the form of a taxonomical classification. Naturally this taxonomy can only be considered as only a partial semblance of completion at this point in time because technology is ever-changing and new forms of artifacts may arise in future that have not been given the necessary consideration here. Scalability of such a taxonomy of evidentiary artifacts of interest is thus a desired characteristic.

An attempt at creating a taxonomic description of all elements in the forensic process is depicted in [31]. Not only does it show the possible sources of evidence, it goes further to classify all elements that can occur or be applied in the forensic process. The scope is wide and covers several axes including sources of evidence, equipment, procedures, hardware and software. The taxonomy is not exhaustive and focuses on cyber forensics, however it gives a formidable basis from which one can further describe a more detailed and focused ontology or taxonomy of elements.

The next few sub-topics will attempt to describe, from different perspectives, a similar taxonomy of potential evidentiary data sources, however more focused towards potential sources of such data, specifically in an Internet of Things scenario. The ability to drill-down further into the specific details of what kind of specific information can be gathered from these, or the actual data structures that contain the information will not be outlined here as it is a whole research area on its own – subject of greater discourse – and somewhat outside the current focus.

#### 3.3.1. Digital Storage Artifacts

Digital storage facilities include any form of devices whose purpose to store data and ensure that it persists there for a reasonably long period of time irrespective of the loss of an active power supply to the storage medium. The host of devices with this sort of capability is wide and includes, disk drives, tape drives (especially for back-ups), optical disks (CD’s, DVD’s, Blueray disks), magnetic removable disks (floppy disks), removable flash storage (thumb-drives, pen drives, memory cards), fixed flash memory (in portable devices) among others that fit the aforementioned description.

These forms of data repositories usually have some sort of file management system. Examples include NTFS, FAT, EXT, HFS, HPFS, YAFFS, their variants and other less popular or proprietary
formats. These are notoriously known to be great sources of forensic information and further to that there are ways that file systems are also manipulated in order to hide data within legitimate storage space (E.g. Alternate Data Streams), within the management structures of the file system (E.g. MBR), or totally outside the scope of the file system, however still within physically exploitable space on the storage device (E.g. the Host Protected Area).

Generally, activity on storage devices have the File-System as their pivot point. It is with this in mind that one can roughly categorize evidentiary artifacts in 3 broad areas: Intra-File-System elements, Extra-File System Elements and Digital Storage Management Elements.

The Intra-File-System Elements include OS-dependent file structures, directories, files, file contents, metadata, files hidden within files and files hidden within space purportedly allocated for legitimate files, among other cases of intelligent manipulation of the File System at hand.

Extra-File-System Elements include those that are outside the control or bounds of the filesystem, and cannot be directly accessed via the File-Systems’ handles. Such data includes that which is stored within artifacts such as the Host Protected Area or Device Configuration Overlays on some types of storage media. Extra-File-System artifacts are generally independent of the file-system and can only be accessed through hardware driver interfaces as they are not accessible by the BIOS or the OS. [32]

Digital Storage Management Elements include those artifacts that describe the layout and architecture of the storage medium. They are generally the elements that enable the BIOS and the operating system to communicate with the storage medium in an appropriate manner. Such elements include the Master Boot Record, partition tables, volume boot records and such related configuration management data structures.

### 3.3.2. Processing and Computation Artifacts

Most devices or systems worthy of being called computers, or capable of performing some sort of processing towards a desired goal or outcome contain some form of space for temporary, short-term, rapid use. This space is usually termed as memory and is used to contain inputs, by-products, or intermediate sub products of malleable data that is to be directed towards some goal be it processing or transmission. Such memory usually resides adjacent to the actual processing facilities (E.g. The Central Processing Unit, or Graphics Processing Unit ) or transfer medium (E.g. A Network Interface Card). These memory spaces subscribe to a variety of shapes, sizes, technologies and uses. Some of the common names that they go by include: RAM/Main memory, buffers, cache memory and registers.

The use of more permanent digital storage space as an extension of this short-term, temporary memory may also be considered as part of this area. Swap space, Virtual memory and scratch disks are some of the names that such storage space goes by. There may be several other brand names relating to this concept, or something similar, that are found on other less common proprietary operating systems.

The processing unit of a computer system may also be considered as an artifact of interest in a digital investigation. To the writer’s knowledge, at the moment, this has not yet been harnessed and the importance, relevance and contribution to digital investigations have not yet been fully determined. [33]
Some of the particular evidentiary artifacts of interest that can be found in live memory include Running Process artifacts (actual executing programs, open files, DLL’s and included libraries, registry handles, hierarchies of linked processes), System Connectivity artifacts (open network connections, open sockets/ports, addresses, partially conveyed data), Loaded OS Kernel artifacts (OS version, system time, loaded modules), Memory Management artifacts (process addressable memory, orphaned processes, virtual address to physical address mappings, crash dumps, data from memory efficiency tweaks) [34]

3.3.3. Communication Media Artifacts

Information or data in transit between different systems or devices is also a potentially interesting source of evidentiary artifacts. The data traffic that passes between connected or internetworked devices is usually in the form of bit-streams, frames or packets and is representative of communication happening between two or more devices. This could provide great insights especially in the case of remotely perpetrated criminal actions, or attempts to move, remove or destroy data that was previously stored on physical storage media or in live memory.

Within today’s Internet, the amount of communication that occurs is massive and is indicative of the actions that different devices or systems are undertaking. Many protocols of communication are quite informative in that they follow a flow of communication requesting data or a service, responding to a request or otherwise informing other devices of their actions. This form of rich, though cryptically encoded, conversations are hallmarks of network related evidence for digital investigations.

This information from communicating networked devices usually comes in the form of packet captures sniffed from some intermediate communication channel bridging two or more devices. The communication between terminals, sensors, hubs, switches, routers, wireless access points, servers, firewall interfaces, base stations and ingress or egress system interfaces are the main areas where such data is collected. Bit-streams, or other non-packet/frame/fragment oriented data, non TCP/IP data may also be captured from other smaller communicating components of devices or the devices themselves, however the parsing of this data into more comprehensible forms has still not yet been popularized as in the case of TCP/IP data, though still interesting evidentiary data, as it may be.

Communication media evidentiary artifacts maybe broadly categorized in the following areas: End-Point System Artifacts (OS fingerprinting, open ports, addresses, applications running), Communication Protocol Artifacts (sessions, supported encoding schemes and cryptography schemes, flags, header contents, credentials and other protocol specific artifacts) and Conveyed Data Artifacts (payload data including files, messages, credentials and other cleartext or encoded data). [35] [36]

3.3.4. Complex and Compound conglomerations and their Artifacts

In some cases various heterogeneous technologies have been tightly and symbiotically packed together into one homogenous mass that has a specific goal and that is rather difficult to break apart and analyze the sub-components independently. This is usually because of the relative complexity, obscurity or proprietary nature of such technologies even though they may have high utility among the human population of today such as mobile phones and industrial control systems. Another reason is that in some cases analyzing the sub-components may result in the singular analysis of
individual components without considering how they affect each other either in the process of disassembly or while the system was in its prior production state.

The relative obscurity of the working of such technologies varies with the uptake (interest of the public) and time, thus the potential evidentiary data is still not standardized nor the specific areas of interest easily pinpointed because the technologies are still rather new and not fully understood.

The range of systems that could be included in this area include: mobile devices, their sensor and protocols (GSM, GPRS, NFC, RFID, Zigbee); industrial control systems and SCADA systems; smart home appliances and perhaps social networks. These all include some form of previously proprietary protocols of communication, file storage systems, data formats and encoding, however as their popularity and uptake increases, they are being forced to standardize their techniques of data communication, processing and storage as well as to publicize it. They are being urged to avoid the age-old and faulty axiom of “Security by obscurity.”

As these complex and compound conglomerations of systems are still at their very core computer systems they still have data storage components, data processing components and data communication components, thus their evidentiary artifacts of interest to a digital investigation should naturally be similar. It is just the barrier of their obscurity that needs to be overcome.

### 3.3.5. Software Application and Operating System Artifacts

Software or application programs are the major controllers of systems, especially those that have been geared to automation, or those that have been tailored towards making human-machine interaction as simplistic and as painless as possible. The instructions in an application are the drivers (controllers) of devices as they instruct physical or electronic components on what to do.

Naturally software, even that powered by artificial intelligence, is not absolutely independent of human intervention. Software is usually created by a human person and even further code-generation still originates from the efforts of the human creator that put in the code generation instructions. However as software to some extent acts independently, especially with regards to automation, it does not “think” per se and could be manipulated by human actors to perform actions that were not otherwise intended according to the original goals of the software. This is the reason for software having logs maintained of the operations performed, and these are crucial artifacts that can be used in a digital investigation. These logs could include local activity of the system, errors, recent remote device connectivity or remote agent activity among other more application specific system logs.

Other than logs, a lot of software applications also contain databases, either as part of the system’s goals or as a means of maintaining data for other systems/ applications. Common database information in applications include user data, communication data, timestamps, address books and a host of other possible application specific data. These are usually found in the form of SQL databases or flat files. Configuration data may also be found either as part of databases or as individual files.

In recent times, location based data has also become a common feature of many applications especially in applications seeking to use location based services. Thus, geographic locations (GPS Coordinates) are also commonly found as application artifacts.

In the age of social applications, networks of closely related elements such as friends, users with similar tastes, opinions or backgrounds, whether known to them or not may also be extracted from applications, or conglomerations of social applications.
The aforementioned potentially interesting software application oriented evidentiary data may be categorized into the some distinct areas, however, the delineation criteria may be rather blurry because the range of capabilities of software today is dynamic and always on the rise. The following are a proposed categorization of the evidentiary data: Operating System Artifacts (System Activity and Error logs, OS Specific Descriptors, Device Connectivity, Databases, Registry and Configuration files, Location Data) and Application Artifacts (Application specific databases, user, activity, and application logs, communication and external relationship data, configuration data, location based data, malware and unexpected tendencies)

3.3.6. Summary of the Artifacts of Interest

In summary, there are generally 2 major sources of artifacts of interest, that is, Hardware Oriented Artifacts and Software Oriented Artifacts. Among these 2 broad sources, there are 6 primary classes of evidentiary artifacts of interest (Digital Storage Artifacts, Processing/Computational Artifacts, Communication Artifacts, Operating System Artifacts, Software Application Artifacts, and Complex and Compound Conglomerations).

Five of these classes can be easily categorized under Hardware or Software oriented artifacts as depicted below.

Hardware oriented:
- Digital Storage Artifacts
- Processing / Computational Artifacts
- Communication Artifacts

Software oriented:
- Operating System Artifacts
- Software Application Artifacts

Complex and Compound conglomerations are considered as a category on their own because they are made of various components that are found both in the Hardware Oriented Category as well as the Software Oriented Category. As mentioned earlier they are rather complex and difficult to break down into smaller components for analysis thus they are considered on their own.

The level of interplay of the primary classes of artifacts forming the compound ones are of varying degrees depending on the systems or application areas involved, however the compound artifacts can in certain cases be reasonably be broken down into their constituent primary components.

To some extent, some Software Oriented artifacts may also be considered as part of Complex and Compound Conglomerations because of their dynamic nature of instructions and their ability to use hardware for storage, processing or communication purposes. The distinction may be somewhat blurry, however for the purposes of this study, they are considered as a category of their own.

Given the aforementioned classification of digital evidence artifacts, it must be said that this is only a singular taxonomy based on component decomposition of IT infrastructures. Other criteria such as the formation of a taxonomy based on size and characteristics may also be applied.

The following 2 diagrams depict the 2 different proposed taxonomies of digital investigation evidence artifacts, however, the first (Figure 3) is the one of primary concern and focus of this study and shall be used in development of the conceptual system model to be proposed. The 2nd taxonomy
(Derived from [31] and [37]) is only posited as another possible means of classifying evidence artifacts of interest in a digital investigation.

Figure 5: Taxonomy of Digital Investigation Artifacts by Component Decomposition

Figure 6: Taxonomy of Digital Investigation Evidence Artifacts by Size and Characteristics.
3.4. Malicious Activity: Classification & Detection

A core part of the conceptual solution being proposed is to be able to detect malevolent activity, so as to be able to initiate the process of identifying, collecting and preserving information that could be of evidentiary use in a court of law. A metric of classification, as well as a set of identification heuristics are the proposed means of effecting such a “malevolent activity classifier.” The first step in doing this is to describe, categorize and characterize that which could be considered malicious activity.

In section 2.3, (summarized in Figure 3) the areas in which potential artifacts of interest could be sourced from have been described and categorized. From these areas, this section now attempts to describe particular actions that may indicate that malevolent activities are in progress.

As far as is known to the author there is no absolute foolproof method of detecting all forms of malicious activity – previously seen as well as those which are new and previously unforeseen. That which can be done is to use statistics to watch for patterns [38]; or derive heuristic indicators (signatures) of malicious activity; or the usage of logs of events, [39] on live systems or on honeypots [40], to detect patterns and match them against some preconfigured security policy. These are the main methods used today in their simples form.

An interesting point of notable concern is that the current malicious activity detection methods seem mostly to be focused on discovering the more technical arena of attack vectors such as port scans, dictionary/bruteforce attacks, flooding/ denial of service attacks, malware propagation, buffer overflows and script injection. Naturally, it has been deemed easier to detect such because of their visible characteristics. Other more sinister attack vectors, such as Man-In-the-Middle attacks, Covert Channels, Privilege Escalation, Identity (IP/MAC) Impersonation/Spooﬁng, Application Layer Attacks, Unauthorized Access or Smuggling of Data, have proven to have enough entropy among the different ways of orchestrating these attacks that it is increasingly difficult to differentiate a legitimate user’s actions from that of an assailant.

In addition to the technical array of attacks there are also the less technical ones which also still constitute malevolent actions. Such activities include phishing, pharming, invasion of privacy, cyber-stalking, illegitimate collection and distribution of private data, social engineering, ﬂaming and trolling. These “softer” less perceivably invasive types of malicious activities are not easily detected, paid notice to or even considered offensive until they have been orchestrated and sometimes even completed and thereafter considered thoughtfully in retrospect. All the same they are still considered malevolent because they generally go against the wishes of the victim/user at hand, or some legitimate system’s goals.

3.4.1. The Anatomy of a Malicious Event

Malicious events occur in greatly varying styles and are usually aimed at besieging the victim in the stealthiest way possibly. This is because the most successful attacks are those that are perpetrated without the knowledge of the victim until it is perhaps too late, that is, when the attacker has achieved their goals or it is relatively ineffective for the victim to do anything in retaliation.

Attacks come in various shape, sizes and goals. The range could span from simple buffer overﬂows, script injection attacks, denial of service attacks, spoofing attacks, phishing attacks, man-in-the-middle attacks to more complex attacks such as botnet controlled attacks, fast ﬂux stealth attacks and strategically planned attacks on Industrial Control Systems. There are perhaps a host of other
technical sounding names of attacks, however even though they may seem to have small scale direct aims, the real aims are usually classed as disclosure, deception, disruption and usurpation.\cite{41}

These aims are usually embodied in some form of malevolent action that we term as an attack. They are invariably described in \cite{42} under the following terms: Snooping, Modification (Alteration), Masquerading (Spoofing), Repudiation of Origin, Denial of Receipt, Delay and Denial of Service. Summarily, these can all be said to be attacks on the fundamental goals of computer security, that is, confidentiality, integrity, availability, authentication, authorization, accountability, non-repudiation and privacy.

The way in which an attack comes into effect, or comes into being, is through what is termed as an attack vector. An attack vector is described as the approach or path through which a malicious perpetrator makes his way into a computer system or network in order to deliver her payload or malicious outcome.\cite{43} \cite{44} Examples of attack vectors could be software vulnerabilities, lapses in physical security, faulty or inadequate security mechanisms, or human elements in systems. The range of attack vectors has great variation in that it could be a single weakness or several in combination.

Given that there is great variation among the parameters of an attack, there are still some similar actions that seem to generally span the layout of an attack. There are certain generic procedures that are carried out and these are sometimes referred to as “The Five P’s“.\cite{45} These procedures are: Probe, Penetrate, Persist, Propagate, Paralyze.

- **Probe**: This is the initial stage of an attack and mostly involves reconnaissance and data gathering techniques that could either be invasive, or from the periphery. Non-invasive procedures could involve mining data from the internet facing elements of the attack surface such as domain names, IP’s, company website data, blog data and data from social networks. More invasive techniques could include port scans, software fingerprinting and version mapping as well as the use of vulnerability scanners.

- **Penetrate**: When potentially vulnerable services have been identified, the next step is to attempt to make use of the vulnerabilities so as to gain access to unauthorized areas in order to deliver the desired malicious outcome. The aims here may vary from simply executing some code, to stealing data, or raising one’s privileges so as to be able to perform other more sinister operations, or to simply bring down the system gracefully or otherwise.

  This phase may be carried out in a variety of ways not limited to attacking faulty authentication methods, executing buffer overflows, using unexpected path traversals, exploiting system configuration flaws or input validation deficiencies.

- **Persist**: If penetration is successful, and the aim of the attack was not just simply penetration, then the most likely next step of an attack is to ensure that access to the system is maintained. This is also done because performing penetration usually involves breaking through defensive and preventive barriers that may leave traces. If the attacker has to perform the penetration again it is likely that more traces of his actions would be left, thus raising the likelihood of being caught. This may involve opening back doors, using remote control tools, creating false accounts with the desired privileges or cracking the passwords of otherwise unsuspecting users.

Removal of traces of the original penetration attempt is also part of this phase and may involve deleting logs, or directing system administrator attention elsewhere.
• **Propagate**: Once the attacker has successfully taken over the compromised system and established their continued presence, the normal next step on the agenda is to discover other related devices that might be of interest either as potential victims, as sources of data or simply to take over as much as possible.

• **Paralyze**: The final goal of most attackers, when they have garnered all the benefits that they were hoping to attain, and they don’t intend on returning, is to cause havoc. This may be through deletion of data, breaking of networks, or defacing public-facing elements such as company websites. This is sometimes also done as the main part of the attack, or as some show of the attackers prowess – giving them some sort of sought after free publicity.

In today’s world almost all attacks that reach their completion portray all these phases. The goal of any intrusion detection, intrusion prevention, incident response or live evidence information aggregator is to be able to identify the malicious activity somewhere along these stages before it reaches its fruition, after which only the static aftermath may be collected for forensic purposes.

### 3.4.2. Malicious Activity Classification

In this section we attempt to classify this malevolent activity in order to lay out a landscape of malevolent activities that we shall propose are imperative and of plausible benefit that they are curtailed within the scope of the proposed solution within this study.

The following classification proposes that there are mainly 2 types of malevolent activities, that is: **Active Malevolent Activity** and **Passive Malevolent Activity**. This classification mainly bases its criteria on the invasiveness of the activity and its visibility or ability to be perceived through tangible digital evidence artifacts. Invasiveness in this specific context refers to the ability of a knowledgeable independent party, victim or “onlookers” to capture, or perceive that an illicit activity is occurring, where an illicit activity is an action against a specified security policy, the dictates of standard operating procedures, or the expected working of the system of concern.

1) **Active Malevolent Activity**

Following the criteria as described above, Active Malevolent Activity may be described as actions that are visible or have directly perceivable effects or artifacts that are testimony to the fact of a violation of a security policy, the dictates of standard operating procedures or the expected working of a system. The idea of visibility or the having perceivable effects generally means that there are traces of evidence whether digital or tangible that could be collected. In the case of digital investigations the evidence is more likely to be digital. This could be in the form of server logs, packet captures, memory captures, disk images, storage device images, pictures, files, configuration files, application database and other a host of other possible digital artifacts as described in the previous section.

Examples of active malevolent activity are:

- Visible reconnaissance and probing activities such as port scanning, software fingerprinting and the use of vulnerability scanners would usually be detected or at least leave traces on firewall logs, system logs or intrusion detection systems.

- All penetration, persisting, propagation and paralysis activities are also very likely leave evidence traces, if they are collected in time. This includes exploitation of authentication mechanisms, buffer overflows, script injections, unauthorized creation of privileged
accounts, opening of back doors, denial of service and several others. Notably, attack persistence activities may hide such traces and paralysis activities may render systems totally inaccessible. (This is an indicator that timely collection and preservation of the evidence artifacts is of prime interest).

- Rootkits and covert channels are also examples of this type of malicious activity because they indeed leave evidence of their presence, however they are extremely stealthy and difficult to discover.
- Cyber terrorism, cyber warfare and cyberhacktivism are also examples of active malevolent activity because their intention is to cause damage and to be visibly noticed. While attempting to do this there will definitely be evidence of the fact of the actions being perpetrated.

2) Passive Malevolent Activity

This kind of malevolent activity does not directly involve the visibility, or presence of perceivable evidence of the actions of malicious intent. Such malicious activities usually do not have the intent to be seen visibly or to cause visible damage, however they aim to perform more sinister activities that may not be considered malicious at face-value but the underlying intents are malicious.

They still follow the general anatomy of an attack (probe, penetrate, persist, propagate), however paralysis is generally not executed. This is because the attacker attains greater benefits by silently maintaining their stance and behaving that they are not attempting anything sinister. Sometimes this is also shrouded by symbiotically providing the victim some benefit or relief such that they do not feel that they are being exploited.

Examples of this type of malevolent activity include:

- Phishing, pharming and their derivatives which attempt to surreptitiously steal identities or authentication credentials
- Invasion of informational self-determination (privacy), illegitimate collection and distribution of private data which involves the collection of otherwise considered private data for purposes that the owner of the actual data did not intend
- Social engineering and cyber-stalking, which involve surreptitiously or indirectly interacting with the victim collecting information from them without their knowledge for use in other unsolicited activities.
- “Flaming” and “trolling” involve the authoring texts or images usually posted in web forums or places of public online discussion generally with the aim of being a nuisance, mildly annoying others or making comments sarcastically or in bad taste. The border between passive and active malicious activity is tested here because the semantics of the authored content as well as the culture of the people and the mood of the forum of discussion may dictate whether the content is inflammatory and malicious or whether it is not.

3.4.3. Detection of Malicious Activities

Given the aforementioned classification of malicious activity it can be deduced that it is generally easier to detect active malevolent activity and capture its corresponding evidence. On the other hand, passive malicious activity can prove to be difficult to detect as well as capture anything that
can be considered as undeniable evidentiary data of use in a digital investigation. This is because some forms of passive malevolent activity require knowledge of the context, the semantics of the information being exchanged or passed, as well as the motives of the purported perpetrator – concealed as they may be.

There are 2 main types of facilities that are used to detect malicious activities in the present computing environment: Intrusion Detection Systems (IDS) and Antivirus software.

Intrusion Detection Systems are mainly oriented towards alerting systems administrators of malicious activities on networked systems, while antivirus (antimalware) software is mainly directed towards detecting and preventing malware from executing on individual host environments. In many cases, because of the complexity of attacks and malware today, as well as the overlap of services that either side provides, they are sometimes classed together as Intrusion Detection and Prevention Systems. However, that which is of key focus here is the means that these systems use to detect malicious activity – be it on individual host or other more complex networked systems.

According to Denning [46] a malicious action requires the abnormal use of normal commands or instructions. This suggests that if one is to detect a malicious action one should be focusing on looking out for abnormalities. [42] Denning in her generic model implies that abnormalities fall in 3 main categories:

- Deviation from usual actions (Anomaly modeling)
- Actions that lead to unexpected states (Misuse modeling)
- Actions inconsistent with system specifications (Specification Modeling)

Anomaly modeling corresponds to anomaly detection [42] which involves creating a set or model of expected values on certain metrics that describe expected behaviour of a system, and constantly checking whether current system metrics correspond to those that depict expected behaviour. In simpler terms it suggests looking for unusual things happening. Mathematical or statistical models are usually used to implement this method, and typically either simple numeric thresholds, statistical measures or Markov models are used to determine whether the system is within the boundaries of normal operational metrics.

Misuse modeling corresponds to misuse detection [42] where a potential intrusion is determined by identifying whether a sequence of instructions being performed individually or in a specific order is known to violate the security policy. This mechanism of effecting intrusion detection relies heavily on the knowledge of potential vulnerabilities of the system at hand and their corresponding attack vectors modeled into a rule-set. The folly of this method is that it implies that attacks that are not covered by the rule-set, including previously unknown attacks or variations of known attacks, would not get detected.

Specification modeling corresponds to specification-based detection [42] in which malicious events are described as actions which are not known to be good, or according to system specifications. This method takes the stance that any action, which does not conform to the specifications of the system, is classified as potential malicious actions. Notably it is said to work best with formalized depictions of what is considered as normal and expected operation of a system, however this is still an area that is currently being developed and researched upon. To the author’s knowledge there is no standardized way of doing this yet.
3.4.4. Malicious Event Detection Information Collection Strategies

The aforementioned malicious event detection indicators describe in a simplistic manner how one would distinguish a regular system action from that which needs to be flagged as potentially malicious. In order to deploy this idea within a system there have been several different strategies proposed – each with its own peculiarities in terms of the actors, the benefits and drawbacks of the layouts.

There are 3 main strategies for deploying malicious event detection systems, mainly based on the location and method of collecting the information to be fed as indicators of malicious events. As described in [42] these are:

- **Host Based Information Gathering**, which involves gathering local information from individual host devices. This would involve gathering information generally from system components themselves, rather than from the interconnecting communication media between the system components.

- **Network-Based Information Gathering**, which involves gathering information from sources of communication between various interconnected hosts. The information usually differs in context from Host-based information because in this strategy the focus is on the information passing between the hosts rather than within the host itself.

- **Hybrid Combination of Information Sources**, which involves combining both of the above mentioned strategies in order to achieve a better, more wholesome view of the activities within the system at hand.

3.4.5. Malicious Event Detection System Implementation Architectures

In order to implement the information collection strategies, as well as the malicious event detection indicators mentioned above, several system architectures have been proposed based on the needs of varying systems. The commonly mentioned ones are: Simple Network Monitoring, Combined Distributed Host and Network Monitoring, and Autonomous Agent Based Monitoring [42].

- Simple Network Monitoring Architectures base their assumption in intrusions mainly occurring through the access of facilities through communication media, thus it is *only the communication media* that are monitored for suspicious activities. The data is collected, stored and processed centrally by specialized analysis engines that analyze the data from the respective activity data in storage, issuing alerts when necessary. This implies that the entire malicious event detection system is centralized in nature.

- Combined Distributed Host and Network Monitoring Architectures combine the monitoring of network communication activity together with the activity on distributed host devices as well as their communication interfaces. The information is aggregated from the different sources at a central location and analyzed for correlations before alerts are issued. This also implies a centralized architecture of the malicious event detection system.

- Autonomous Agent Based Monitoring Architectures are intrinsically different from the 2 aforementioned architectures in that the malicious event detection system is distributed through agents. The agents are independent elements that collect, carry and process information and issue alerts on their own. They are capable of being independent of any specific elements of the system, however, there is general cooperation and distribution of
information among the agents making the system easily scalable. Due to the cooperation and distribution of information, there is a greater amount of communication overhead as compared to the 2 previously mentioned architectures. There is also a more pressing need for security among the agents and their communication because of their seeming independence and substantial individual capabilities.

The details of each of the architectures are considered to be intricate and specific to particular situations that they are applied in. This brings about several differences, however, they do share some common generic components worth noting. All the architectures have some form of data collection agent that is responsible for collecting the data that is to be analyzed for malicious activity. Additionally each of the architectures proposes a data management and processing facility that is in charge of filtering the data and discovering whether there are artifacts that are suspicious or worth noting. The final component is an alert mechanism whose main purpose is to receive the data from the processing facility and issue the necessary notifications.

The major difference that is noted among the different system architectures is the locality of the core components mentioned above. From the architectures described, either of 2 options are suggested, that is, a centralized orientation or a decentralized orientation. In both cases the components have fixed capabilities and are specialized in their functions. In this study a distributed Peer to Peer architecture with the participants having similar or equal capabilities will be described and analyzed.

3.5. Storage of Information and the Semantic Web

In a digital investigation one of the most important aspects is the capturing and storage of as much useful and viable data from the environment of the crime scene. This is because it is rather difficult from a first glance to pick out what may be important and what is not. Thus a more generous approach to collecting as much as possible is usually employed.

The current state of matters is that the complexity of malicious events has increased enormously and thus it has become difficult to zero-in on what is actually useful data that can be use either to trace back the events that occurred or to prove an event, action or motive of the perpetrator for the sake of seeking out justice. The reality is that at the moment, among the daily interactions between computer systems, there is a lot of peripheral data crisscrossing endpoints that is probably the outcome of normal use of the computer system under scrutiny, that is clouding out the sought after data or information that is imperatively collected and maintained. In this respect, the normal data that is representative of accepted use of the computer system at hand is actually considered as potentially noise and is relegated to being discarded as unimportant.

Presently the most common way of achieving this is through applying Known File Filtering (KFF) which basically works towards removing known operating system files within file systems by analyzing them for their integrity. This is done by checking their hashes against previously computed hashes of the same files on the file systems of known uncompromised computer systems. The same concept could also be applied to network traffic in terms of measuring the entropy of certain values of characteristics of the captured traffic against that which is considered as the normal entropy of the characteristics in question. The major problem with this, especially with network traffic is that manipulation of protocols in order to work as covert channels with data being piggy-backed in seemingly unsuspicious messages could distort the measurements. The legitimate use of cryptography could also cause similar outcomes that distort the measurements.
With respect to file systems, known file filtering is only viable when the file system is known to be of the same standard of security policy defining similar standards of what is considered uncompromised. Thus if different participants or controllers of the file systems have different concepts of what is considered uncompromised, then false-positives are sure to arise due to the variations from the differences in the understanding of the norm. False negatives would also probably occur, just that they would appear in a less salient manner.

In order to alleviate this problem of different understandings of what is considered useful, or an indicator of evidence of malicious events or intents, one would need to have some form of universal understanding. One emerging technology that aims to provide a universal understanding as well as an unambiguous and precise meaning to data distributed among computer systems is the concept of the Semantic Web.

The Semantic Web may be described as a structured manner of augmenting otherwise simple, meaningless and disconnected masses of incongruous data with appropriate identifiers, attributes, qualifiers and relationships in order to give the data a more definitive meaning and a clearer context within a specific domain of concern. It is regarded as a means to seamlessly imbue integration across different types of content, information applications and systems. [47]

The original idea [48] was developed as an enhancement to the standard World Wide Web (on the Internet). The classic Internet has grown to become a maze of enormous amounts of disconnected pieces of data and information serving individual purposes. As it is known, the amount of data on the Internet is vast and largely not known, not easily traversable, nor searchable except through well renowned search engines (E.g. The Google Search Engine). This is quite easily acknowledged through the fact that most of the time today when we get onto the Internet, the first stop, more often than not, is a search engine in order to get to our intended Internet destination. This is because of the mind-boggling massive and rapid creation of data on the Internet. It quickly relegates that which was relevant, fashionable, trending and easily accessible yesterday, to being outdated, irrelevant and crowded out by the latest information of today in a matter of hours. Thus finding specific and accurate data that is chronologically distant in history can get unnervingly difficult as it gets fogged out in clouds of new data.

This problem of information overload, relevant information acquisition and information discrepancy resolution [49] is what is experienced on the Internet as well as in digital forensic investigations of today. Human or manual collection, analysis or digestion of these massive amounts of data in order to solve these problems is simply not feasible anymore. Thus, the Semantic Web aims to make use of machines to traverse this information space in order to solve these problems. Once an agent (a computer or a set of computers) has performed the first level of information processing and management, a user can then access or manipulate the results with more ease. [49]

The major benefit of the Semantic web contributes is that it flattens the landscape of data or information allowing different agents, systems or entities with different capabilities all to have a singular and equal understanding or perspective of the subject matter at hand. In this way it removes ambiguity among different information consumers as well as ensuring equality in information dispersal among interested parties. This makes it easier for machines (agents) of all characters to extract the same understanding from a piece of data and in turn these machines (agents) can provide this same meaning/ view/ perspective to their respective controllers.
3.5.1. Core functionality of the Semantic Web

The core of the Semantic web as described by Berners-Lee et al. [48] centers around a tiered architecture composed of 7 distinct layers sometimes referred to as the “Semantic Web Stack” or the “Semantic web layer cake”. The distinct layers in the stack are applied to data resources in order to give them an unambiguous semantic meaning, context and relationships to other data as well as a framework to provide logical understanding, authentication, proof and trust to the data and its corresponding agents and data consumers. The main aim of this is to enable computers or machines to make logical conclusions about data that they are presented with in order to reduce the burden of human analysis which is becoming more and more inefficient due to the huge amounts of data presented. This is commonly presented as the problem of “information overload”.

The semantic web stack is composed of 7 layers [47] [48]:

- **Layer 1**: Unicode and Uniform Resource Identifiers (URI's)
- **Layer 2**: Namespaces and XML
- **Layer 3**: Resource Description Framework (metadata and schema)
- **Layer 4**: Web Ontology Language (OWL knowledge representation language) vocabulary and rules
- **Layer 5**: Logic framework
- **Layer 6**: Proof
- **Layer 7**: Trust

This is depicted diagrammatically below in figure 5 and 6. Figure 5 depicts the original semantic web stack as described by Berners-Lee et al. [48], while Figure 6 depicts a modified version of the same stack from [47] adapted with newer more mature and established technologies developed to fit into the same web stack as well as some of the original constituent components that are still being developed.

![Figure 7: The Original Semantic Web Architecture](image-url)
The Semantic Web stack is largely still a work in progress and the various technologies bundled there in are still undergoing changes for the better, however the conceptual idea for the functioning of each of the layers towards the aforementioned goals is already described and accepted as plausible. The current descriptions and work of the layers is described below:

**Layer 1: Unicode and Uniform Resource Identifiers (URI’s):**

Humanly understandable languages can all be expressed in a written format with the use of characters (i.e. letters) representing different phonetic sounds which when combined, communicate words with meaning. Words are then combined to form sentences which contain ideas or information that is being passed. The foundational element here that begins the process is identified as a character.

Computers can “understand” characters by assigning specific numbers to specific characters in order to identify them. The Unicode Standard (Universal Character Set/ISO10646) is one of these mappings. It is the main standard that attempts to map (encode) all the characters of human written languages to corresponding numerical values. [50] This is the first step towards getting a computer to understand human communication – one of the largest uses of computer systems today. Unicode is particularly suited because it caters for international use across all written languages.

A Uniform Resource Identifier (URI) is defined as a compact string of characters that is used for identifying an abstract or physical resource. [51] A resource is described as any entity that has an identity, while an identifier is described as an object that can act as a reference to something that has an identity. In the case of URI’s, this identifier generally conforms to a specific syntax. The uniformity comes about from the form of the URI’s in terms of their syntax and ability to be reused or replaced in different application areas without the need of radical reconstruction.

These 2 elements form the most basic layer of the semantic web forming a link between the actual resource and the rest of the elements in the semantic web stack that give it meaning and context.

**Layer 2: Namespaces, XML and XML Schema**

The Extensible Markup Language (XML) is a standard that describes a format of describing data such that it is both human readable and machine readable allowing for interchange between systems [52]. Above this, it is a markup language whose textual structure is not solely used for design
purposes, but rather as a representation that communicates a description of the content that it is applied on through the structure and order of its elements. [53] This type of a markup language is sometimes referred to as a meta-language, which is simply a language that describes another language. [47]

An XML Schema is a textual document that defines the model upon which a specific set of XML documents can be derived from for a specific application, knowledge domain or system. It assists in providing the platform for exchange of information without the need of translation or re-encoding. It generally describes the possible arrangement of the elements, the properties (attributes) of these elements and the possible textual values that the attributes can contain. [54]

According to [55] a namespace is described as a collection of names, identified by a URI reference. [51] Namespaces are embodied as a collection of related element terms and are used in order to disambiguate elements which may have similar naming constructs but have different attributes, meanings and uses. They are used to qualify elements and attribute names such that they are can unambiguously be distinguished from similarly named elements or attributes. [47]

Layer 3: Resource Description Framework (metadata and schema)

The Resource Description Framework (RDF) is one of the core enabling features of the semantic web. The RDF is composed of data models, syntactic rules and a schema. The main function of these is to provide a metadata description mechanism for the upper layer language technologies. [47] This description is done in order to make the metadata machine readable and understandable to a certain extent.

This essential behaviour is achieved by use of a generic data model where a statement describes an entity (a resource) in terms of its attributes (properties or characteristics). Such a statement is composed of 3 main parts: a subject, a predicate and an object. This is sometimes referred to as the subject-predicate-object triple. [49] The subject is the resource that is being described by the statement. The predicate is the attribute or set of characteristics (properties) of the resource being considered. The object is the particular value that is designated to an attribute.

An example of this is portrayed below:

```xml
<?xml version="1.0"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/2012/02/22-rdf-syntax-leia#"
 xmlns:aei="http://www.leia.org/aei#">
 <rdf:Description rdf:about="http://www.leia.org/aei/IDS">
   <aei:id>73ef4da</aei:id>
   <aei:size>480,000</aei:size>
   <aei:timestamp>2012-07-01 09:41-UTC</aei:timestamp>
   <aei:hash>d3e45y1k395kh3x3b4zs5bd8sw5mgv2k</aei:hash>
   <aei:signer>Sensor-Bob-342</aei:signer>
   <aei:verifier>HQ-Central-34</aei:verifier>
   <aei:type>Network_Capture</aei:type>
   <aei:content>00000000000000</aei:content>
   <aei:notes>NULL</aei:notes>
 </rdf:Description>
</rdf:RDF>
```

Figure 9: Example of RDF format
The RDF schema provides a set of guidelines that allows for RDF descriptions to be organized within application specific relationships generally in the form of classes, properties and values. This forms the basis of a data typing system for RDF models [56] thus providing another level of semantics to the resources being described through relationships, for example, between instances of classes or subclasses of classes. [57] In this way the RDF Schema is said to provide a framework for describing a domain specific vocabulary that is composed of classes, subclasses and their corresponding attributes that are related to in each other in specific manners described by the structure provided by the schema (largely similar to that found in object-oriented models).

**Layer 4: Web Ontology Language (OWL) vocabulary and rules**

An “ontology” in the context of knowledge representation refers to the description (formal specification) of the concepts and relationships that can exist for an agent or a community of agents in a specific domain. [58] [59] This means that it describes the context and relationships that a certain element has or can have in regard to what exactly it is. While RDF Schema provides the vocabulary, structure and relational constraints in order to express the metadata related to a resource, the semantic meaning is accorded through the use of an ontology language. The important value-adding feature that an ontology language ascribes to what it is describing, is that it allows for ontological modeling through the semantics and thus machine driven reasoning.

The Web Ontology Language (OWL) is a knowledge representation language that is able to describe both the syntax and semantics of the information resources in a specific domain. [47] It is designed to be used by applications that process the content of information/data rather than simply presenting it to human information consumers. It facilitates greater machine interpretability of Web content than that supported by XML, RDF, and RDF Schema (RDF-S) by providing additional vocabulary along with a formal semantics (rules). [60]

Thus, it can be inferred that ontologies could provide a means to ensure a common understanding of information through the semantics that they give information elements. They also could possibly allow for seamless and unambiguous communication of information between humans and machine information consumers due the “reasoning” ability that they bestow upon machines.

**Layer 5: Logic framework**

The logic framework is the heart of reasoning and inference within knowledge representation. It forms the core precepts and structure of how information is to be arranged using a formal specification (language) in order to infuse machines with the capacity of reasoning, inference and unambiguous meaning to statements constructed according to the specific logic framework. [61]

Machines are capable of following rules and instructions. When they are fed with data that has been impregnated with descriptions, instructions and formal constraints, they can use this metadata and following a logic framework in order to formulate a logical result. This is how machine reasoning and making inferences is automated in practice.

The results of this inferencing and reasoning can be yet again stored as RDF resources with descriptions within the same framework in order to serve as input for another reasoning cycle.

**Layer 6: Proof**

The concept of proof is still under development and is currently being adapted and modified to fit into the semantic web stack. As such it will not be elaborated in much detail.
There are 2 major areas in a proof framework: “Proof validation” and “Proof generation”. It has been established that proof validation would be a necessary requirement in order to validate the reasoning or inference carried out by the lower layer of logic. Proof generation is not as yet seen as an important factor to be included in the basic structure of the semantic web stack.

Proof validation is synonymous with theorem proving and could be roughly described as the process of validating the veracity of a statement through the logical analysis of a list of inference items that were used to derive the statement under scrutiny. [62] [63]

Layer 7: Trust

Trust is described as the belief that the other participant is benevolent, competent, honest or predictable in a given situation, as well as the willingness of the interacting participants to depend on one another in the situation at hand. [64] [65] Within the realm of the semantic web the concept of trust extends also to the authenticity of the information being contained within or being communicated between different participants. Authenticity in most cases is difficult to ascertain given only one source of the aforesaid claim of veracity. Thus the common way that is used is to verify the veracity of the information from another source, sometimes called “Third party.” This does not fully solve the problem, however, it adds some credibility to the claim if the “Third party” verifies the claim. The reality is that the trustworthiness of the “Third party” could also be questioned, thus it becomes a considerably unsolvable problem.

This is where the time tested security concepts of the Public Key Infrastructure, Certificates and Digital Signatures could come into play. So far they are the current method of solving the problem of authentication especially used in security services where the identity of an entity needs to be verified. Practically it has worked so far, but is not infallible as there is a given assumption that there is a Third Party that everyone trusts undeniably.

The other infrastructure used to implement this authentication is that used in the Web of Trust model (or the Pretty Good Privacy –PGP – model). It makes use Certificates and Digital Signatures, however, there is no singular trusted authority that every entity must trust irrevocably. The decision to trust is based on the assumption that every party has a set of peers that it trusts. If any of these peers or a subset of them trust another party, then one would trust (by proxy) the external entity that is not already among the trusted peer group.

Cryptography: Digital Signatures and Encryption

Digital signatures are cryptographic functions based on public key cryptography that can be used to determine and validate the identity of a communicating participant. The Digital Signature Standard (FIPS 186) is the common standard that is used in various applications today for purposes pf authentication and integrity of information. It is proposed and recommended by NIST (National Institute of Standards and Technology). [66]

Encryption is the process of converting an otherwise human readable text into a form that is not understandable by any unintended parties, but is reversible only by the intended recipient. The W3C XML Encryption standard is currently one of the proposed methods of implementing encryption.

Though these cryptographic ideas were not part of the original idea as specific layers, their importance has been noted and acknowledged especially in providing confidentiality, integrity, authentication, trust and accountability. They were therefore recommended and have been adopted into the standard. [47] Cryptography can be applied to all the layers as a whole and as such cannot
be singularly placed in one layer. Thus it is depicted in the Semantic Web Stack as an element that traverses all the other layers.

Summary

The Semantic Web is a facilitator of unambiguous communication of knowledge and reasoning capabilities among machines. It assists with problems associated with information overload, and gives machines the capability of “understanding” and reasoning over the data that they are processing. In this respect it could be useful in helping make the forensic investigation process a little faster and more efficient through assisting in sifting through the massive amounts of information and organizing them in a way that could be used to quickly correlate data as well as share it among collaborating units (whether human or machine). Such sharing and collaboration is necessary in order to spread the work load, or to make use of specialized services that other units/departments/machines could provide.

In this study, however, the full capabilities of the Semantic web will not really be exploited in their entirety. There is room for these extra capabilities however they are deemed out of scope for the purposes of this study. The capabilities that shall be harnessed are within the layers in the stack from the RDF layer (Layer 3) and below. These shall be explained in greater detail further along this document.

3.6. Resource Availability, Distribution and Collaboration

As has been suggested previously in this document, one of the biggest problems besieging the digital forensics community, particularly those involved in the collection and analysis of potentially evidentiary data, is the sheer amount and rate of increase of the information that they have to contend with. [67] This, combined with an ever increasing amount of variety among the possible sets of data that can be collected from vastly varying systems, creates a deathly mixture of problems. The unstructured and uninform nature of such data further compounds the complexity of the problem, making it immensely difficult to conjure up a foolproof solution.

In the case of digital investigations, as with several other computational related fields of study, the main resources of concern are storage and processing power. When there are larger amounts of data to be dealt with, larger storage and processing capacities are needed. However, the exploitation of natural resources towards these causes is limited. Consequently storage space is limited as with the capability of processing units to perform computations. On the other hand, the capacity for humanity to generate information out of their day to day activities seems to tend towards being unlimited. If this is to be recorded for the sake of providing information of evidentiary use, there is a risk that the information generated can far outstrip the capacity to store and process it. This is characteristic of an information overload.

Having said this, outlining a bleak picture of what needs to be contended with, it has not as yet been considered an intrinsically impossible task to at least provide a workable solution to deal with all this information.

The major problem that this section deals with is the seemingly immense amount of data to be collected and processed given the correspondingly minimal availability of resources and their limited capacity. This is sometimes referred to as the “Big Data Problem”. The commonly cited solutions to this problem that exist within today’s technology lie along the lines of distributed resource sharing and collaboration for solving large problems mainly manifested through Distributed Databases,
Cloud Infrastructures, Peer to Peer systems, and Crowdsourcing. When drilled down further, the area of concern is seen to be the efficient use of resources through distributed and collaborative mechanisms.

An overview of the interesting issues and points of concern with regard to these resource scarcity problems and the current solutions is described in the following sections.

With resource scarcity being the main issue at hand, the overarching theme of these sections will reflect the two major computing resources needed, that is, Storage Capacity and Processing Power.

### 3.6.1. Distributed Datastores / Databases

Initially storage of digital data that needed to be maintained in a persisting state between computer processing sessions was being done in the form of flat files stored on some form of storage medium. Punched-cards, magnetic storage, optical storage and flash memory have been part of the evolution of the storage media, upon which bits and bytes have been imprinted so as to maintain their persistence. The structured format in which these bits and bytes have been laid out in order to allow for them to be retrieved in an orderly manner for future use, has also evolved in time, however, somewhat independently of the capabilities of the storage media. This evolution has seen the use of a directory and file hierarchy structure, a relational database structure, an object oriented database structure and now the current trend of distributed file systems and distributed datastores. Currently the distributed storage structure strategy seems to be moving towards adopting a previously proprietary paradigm dubbed the NoSQL (Non-relational database structure) initiative [68] that could possibly be the next step in the future of data storage.

While advances in technology, the need for storage capacity and the cost factor have driven the development of different materials influencing the forms of storage media, the business needs of different organizations and their respective systems has correspondingly driven the evolution of the methods and data structures used in ordering data on the underlying storage medium. It is well worth noting the major factors that have influenced this in order to understand and justify this evolution and the movement towards the NoSQL initiative.

Some of the factors that have influenced the method of storage and the corresponding data structures over time are as follows:

- The ever-increasing need for information and the explosion in the supply of information
- The inherent use of information in every aspect of modern technology and activity
- The increasing inability to store massive amounts of data on a single device/data storage structure
- The need for availability through redundancy and compartmentalization
- The need for random access rather than sequential access of data
- The need for storing complex entities and their data
- The need for introducing relationships among entities and their respective data elements
- The need for speed in performing transactions including insertions, updates, retrievals or deletions
- Concurrency issues in multi-user environments

44
3.6.1.1. The NoSQL Initiative

Today the major data storage structures widely in use are relational databases, object oriented relational databases and – more recently – the NoSQL genre of databases. Each has their own advantages and disadvantages, however that of key concern here is the genre of NoSQL databases.

The recent sudden rise in the uptake of NoSQL storage structures has arisen from the relative inability of relational databases to efficiently manage colossal amounts of data in the orders of Petabytes [69]. The large amounts of data have resulted in the increasingly critical need for good scalability in terms of easily supplementing storage and processing capacity, as well as ensuring that transactions (insertions, updates, deletions) and other database management tasks are not adversely affected by the massive amounts of data.

Vertical scaling, that is, increasing the computational power of a single node, is seen as a temporary solution which is quickly outdone when the physical resource capacity is eventually exceeded. [68] For practical reasons, not elaborated here in detail, it is easily noticeable that the capacity of any single device for processing or storage of data can be quickly rendered inept due to the currently explosive growth of data in the IT realm. Thus it is conclusive that vertical scaling is not quite the mode of scalability that would be preferred because singular large nodes have capacity limits in themselves.

Horizontal scaling is the favoured method of scaling and is the hallmark of the NoSQL initiative. It involves dividing the data into smaller clusters and distributing them among several different nodes according to some standardized application specific criteria. This partitioning of the data store and distributing it among several storage nodes is detrimental to relationships, the fundamental precept of relational databases, thus the emergence of non-relational database structures. Queries and transactions are independent of the standard SQL structure and are more oriented towards indexes key-value pairs and hash tables. In this sense NoSQL databases are used on applications whose data does not critically rely on relationships and thus the database schema is designed in a radically different manner omitting the concept of relationships among entities.

The NoSQL initiative is not seen as an entire replacement of relational databases. On the contrary, it is seen as a supplemental feature that improves other drawbacks that standard relational databases struggle with, such as availability, performance and fault-tolerance, in addition to the aforementioned issue of scalability.

Characteristics of NoSQL non-relational databases

As mentioned earlier, the NoSQL initiative is not meant to replace relational databases, but rather to supplement it in its failures. Non-relational databases are generally only used in systems where the relational features of RDMS’s are not really needed, or are very sparsely needed. Mostly this has been used successfully in large originally closed-source company projects and has been seen to be successful. Of particular note are Amazon’s Dynamo [70], Google’s MapReduce [71] and Bigtable [72], Apache’s Hadoop [73] and Facebook’s HBase [74]. Most of these have now been open-sourced and other large open-source projects have sprouted from these original ideas. Some of the related open-source projects include: Cassandra, Neo4j, Riak, CouchDB, MongoDB, Scalaris, Voldemort, Redis, Terrastore and RavenDB.

Each of these different NoSQL non-relational databases has its own individual unique noteworthy features. However, they also have several commonly featuring aspects that they often share. These
may be classified as the definitive characteristics of modern NoSQL Non-Relational databases. These characteristics include:

- Indexes
- Key-value storage
- Distribution among a large number of nodes
- Partitioning of data across several nodes
- Fault tolerance through redundancy

One of the greater hallmarks on NoSQL non-relational databases is the seeming ability to achieve a better balance between the ever elusive triad of Consistency, Availability and Partition Tolerance in distributed databases as illustrated by the CAP Theorem. [75] The CAP theorem shows that there is a trade-off between Consistency, Availability and Partition Tolerance in distributed systems and only 2 out of these 3 properties can be attained optimally in any one implementation.

Consistency refers to the property of databases where transactions must be completed in their entirety bringing the database from its original state to a new accepted stable state. This acceptable state is subject to a certain set of validation criteria that determine whether the transaction is valid and stable. Particularly in distributed databases, it implies ensuring that transactions that are to be performed have been successfully undertaken, in their entirety, on all the redundant copies of the data across the system. If a transaction fails to conform to the validation criteria, then all the operations performed must be reversed to take the database back to the original (previous) stable state. This characteristic is important so as not to induce incorrect states of database entities that could be falsely propagated forward as correct states.

Availability is the ability or capacity of a database to be ready to service a request on demand. This means that the resources that are required in servicing the incoming request should be available for use without any severely debilitating deficiencies that would hamper the proper fulfillment of the request. The major antagonists acting against availability are the problem of single points of failure and isolation of parts of the system. Thus strategies promoting availability are implemented towards avoiding these major adversaries.

Partitioning, also known as “sharding”, comes about from the need for large databases to be split and distributed among several nodes in the process of horizontal scaling. Partition Tolerance is the capability of such a database management system to continue to function acceptably even in the event of some nodes (containing certain partitions of the distributed database) temporarily or permanently losing network connectivity, becoming isolated from the rest of the system.

As has been alluded to earlier, the CAP Theorem is a fundamental precept of distributed databases. Standard relational databases, when distributed, conform to this precept and either one of the characteristics of Consistency, Availability or Partition Tolerance must be forfeited. However, in the case of the NoSQL genre of databases, this is slightly different. A work around has been attained in order to provide some sort of balance. This work around is achieved on the part of consistency through the concept of a weak form of consistency, also termed as eventual consistency. The availability and partition tolerance that are achieved are generally considered to be similar, if not more robust than that of the standard relational databases.

The concept of Eventual Consistency generally has 2 similar nuances of a particular definition – one being slightly more assertive of the conviction than the other. According to Vogels [76], a particular
set of transactions performed on a set of data eventually gets propagated throughout the system to all the replicas (redundant copies) given sufficient time and given that no other further modifications are performed on the replicas or the originally modified data entity. Terry et al. [77], on the other hand, asserts that once a modification is accepted in a system, the modification either becomes dominant and spreads out throughout the entire system, or the older replicas (copies) eventually get phased out (retire) over time.

### 3.6.2. Clouds & Cloud Computing Infrastructures

According to the National Institute of Standards and Technology (NIST), cloud computing is described as “a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (E.g. Networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.” [78] A cloud is also said to be characterized by 5 key properties, 3 service models and 4 deployment strategies. [78] It should be noted that there are still widely varying opinions about what exactly constitutes a cloud computing infrastructure. The definitions are still being debated upon in an effort to achieve standardization. However, for the purposes of this study, we shall assume the NIST recommended definition of a cloud, its properties, service models and deployment strategies.

According to the NIST recommended definition, a cloud has the following essential properties:

- **On-Demand Self-Service**, that is, a user can singularly request for a resource and get it without requiring any particular human intervention on the part of the service provider.

- **Broad Network Access**, which refers to the availability of seamless, reliable and efficient network connectivity towards the idea of ubiquitous and untethered communication.

- **Resource Pooling**, whereby the combined capacity of available resources available, such as processing cores or storage nodes, is harnessed as a single magnified resource. Multiple users are served simultaneously and continuously, with the resource (in both physical and virtual form) being dynamically allocated and repossessed transparently without affecting the users activity. This is characteristic of the concept of *multi-tenancy* in cloud computing.

- **Rapid Elasticity**, which means that the capacity of the cloud should be able to expand and contract depending on the amount of resources demanded.

- **Measured Service** is the concept that is most important from the business aspect, for both the provider and the consumer. It refers to the ability to quantify or measure the amount of resources consumed using suitable pre-agreed metrics, depending on the particular resource being consumed. Such metrics could be aligned to the units of computing hours, bandwidth or storage capacity used.

Cloud computing infrastructures can be provisioned according to one of 3 service models. These service models are generally founded on the varying amount of control that the consumer is given over the resources available in the cloud. The 3 service models are described below:

- **Software as a Service**: In this model the user is given a software interface, where they can only control the particular software application provided and perhaps some of its configurations that are available through the application itself. The user has no capabilities of making any changes to the provisioned operating system, and even less to the underlying
Cloud infrastructure. The service is usually presented to the user via some sort of a thin client interface, web browser or the applications interface.

- **Platform as a Service**: In this model the user is given slightly more freedom on the cloud infrastructure. The user is provided with application programming languages, interfaces, libraries and services with which they can develop applications upon. The user has control over the applications that they can deploy as well as some lower level configurations of how the application is served by the underlying infrastructure.

- **Infrastructure as a Service**: This is the most liberal model whereby the user is given the capability to leverage and manipulate actual resources such as storage, memory, processor cores and perhaps some networking devices. This comes across as a physical computing infrastructure available to be controlled by the user. The user also has the capability of deploying and configuring operating system platforms as well as custom built software on the available infrastructure. The underlying cloud infrastructure, however, is still fully managed and maintained by the service provider. The user is not granted privileges for making changes to the cloud infrastructure itself.

The diagram depicted on the following page expresses the varying levels of control that the different participants involved in a cloud infrastructure are granted in a generic case [79].

![Diagram of varying levels of control](image)

Figure 10: The varying levels of control possessed by the different participants in a cloud infrastructure.

Clouds like any other system are implemented with the aim of fulfilling certain goals. These goals are usually aimed at a certain population of users within a business context. Generally the users are either part of the production and operations of the business, or they are the consumers of the products or services of the business. The relationship of the users, to the business, and thus the cloud, determines the deployment strategy. There are 4 documented strategies as described by the NIST recommendations [78]:

...
• **Private Clouds** are deployed for use within the bounds of a single organization. They may be up distributed among the different business units of the organization, but do not extend outside this. The cloud infrastructure may be deployed on the premises of the organization or at a separate location. The management and control may be performed within bounds of the organization itself, or by a third party provider, or a combination of the two.

• **Community Clouds** are deployed among users who may not be part of the same organization however they have some form of relationship in common and have some common aims that can be achieved through the cloud. As with the previous case the deployment may be on or off-site and the management of the infrastructure may be the duty of one of the co-operating groups, a combination of them, or a third party provider.

• **Public Clouds** are deployed for use by the general public for their own innovative uses. Currently they are used a lot for research purposes or by organizations that need large computing power for a short period time. They are usually provisioned and managed by large businesses, academic institutions or governmental organizations usually on their own premises.

• **Hybrid Clouds** are a combination of two or more of the above forms of cloud deployment strategies usually distributed among different clusters of users within an organization. They maintain distinct boundaries between each other, but use the same underlying core infrastructure.

The concept of 5 essential properties of a cloud, 3 service models and 4 deployment strategies are summarized in the conceptual diagram below as visualized by the NIST Recommendation [78].

![Diagram of Cloud Computing Definition](image_url)

**Figure 11:** Visualization of the NIST recommended Cloud Computing definition
As mentioned earlier, the 2 major resources that are of key concern are storage capacity and processing power. The implementation of these resources as the primary provisioned resource of a cloud infrastructure is discussed in the following sections.

3.6.2.1. Cloud Processing Technologies

The initial fledgling resource that was first brought into the cloud realm was the shared use of processing power. This was initially brought on through the then innovative ideas of cluster computing, grid computing and utility computing.

The conceptual idea was first described on record in 1961 by Jon McCarthy as a prediction that “computation may someday be organized as a public utility” and he described how this could possibly come to pass. [80] This idea was not much considered until the mid-1990s when the constituent factors that could bring this idea into fruition began to mature—ready to be harnessed for greater developments. Some of these factors were: hardware oriented developments (such as virtualization and multi-core processors), new Internet Technology paradigms (such as web services, the server oriented architecture, Web 2.0), robust distributed computing infrastructures (such as cluster computing and grid computing) and automated systems management platforms. [81]

Arguably the most critical areas of technology as well as the comparatively most similar technologies that spurred the development of cloud computing are the ideas of cluster computing, grid computing and utility computing.

Cloud Computing Predecessor Concepts

i. Cluster computing and Parallel Processing

Cluster computing arose from the need to increase the amount of processing power. The computing power that can be generated from a processor is limited by natural laws of physics, such as the laws of thermodynamics and the speed of light, as well as the economics around the manufacturing of processors from their raw materials. Thus, one of the earlier cost effective and more viable ways of attempting to increase processing power was through collaboration. This was done through connecting multiple processors together and coordinating their efforts towards the common goal. [82]

This is sometimes referred to as parallel processing because several processing cores are bunched up together in an interconnected cluster and the work is smartly distributed among the processors, such that they work simultaneously in order to achieve the desired outcome in the most efficient way without encountering deadlocks. The “smartness” of the cluster lies mainly in the middleware and the scheduling mechanism that form a transparent layer distributing the work among the processors and other shared intermediary peripherals, allowing for efficient sharing of the work.

ii. Grid Computing and Utility Computing

As described in [83] “Grid computing is the technology that enables resource virtualization, on-demand provisioning, and service (or resource) sharing between organizations. Using the utility computing model, it aims at providing a ubiquitous digital market of services.”

Grid computing is the closest description to the concept of cloud computing and perhaps is arguably either a very similar concept, or greatly overlaps with it. Foster et al. in [80] present this idea, outlining that though they are very similar, there are still some subtle differences between clouds and grids especially in the way the internal modules are structured. It is
postulated that they have differences in the paradigms relating to security, data structures, programming methods and computation models. One of the more visible differences is seen in the business model in the fact that grids were primarily used in a proprietary and closed manner within the realm of an organization or group of organizations mainly for the purpose of sharing data. This may be mapped somehow to the idea of a private cloud, but does not encompass the other deployment models, as discussed earlier. For purposes of the is study, the concept of grid computing will be considered as the direct predecessor of cloud computing, however the idea was not entirely feasible at the time that it was proposed in the late 1990s, and it did not take off too well.

Grid computing has its roots in the idea of interconnecting several machines for the sake of facilitating resource sharing from the collective pool of resources in a transparent manner. Additionally the idea of grid computing was to provide computing power as service similar to electricity and water provision services. In this way units of computing power could be procured when necessary and the amount used, measured and billed. This is the idea of utility computing. This is similar to the model in cloud computing, however the enabling technologies and the overarching business need were not present at the time the grid computing was beginning to gain popularity.

iii. Virtualization and Autonomic Computing

Virtualization is described as “the separation of a resource or request for a service from the underlying physical delivery of that service.” [84] It may also be described as “a technique for hiding the physical characteristics of computing resources from the way in which other systems, applications or end users interact with those resources.” [85] In other words, virtualization is a way in which an abstraction layer is put in between the actual physical resource and the representation of the resource to entity or process making use of the resource. This abstraction layer is usually termed as the hypervisor or virtual machine monitor. Its purpose is to simulate the entire physical resource, part of it, or another necessary related resource to the resource consumer. It acts as the bridge to the physical resource while managing the virtual representations of the given resource presented to the resource consumer.

One of the initial and major uses of virtualization has been to make better use of resources through the sharing of the underlying physical resource. This meant sharing the physical resource based on time slices given to the virtual resources and consequently to the resource consumers. This idea and its implementation is the major reason why virtualization is considered as one of the key forebearers of cloud computing. This is because it allows for on demand service through the unequivocal provision of resources to any demanding consumer, while at the same time it allows for resources to be pooled together for common use by the consumers. These are some of the essential characteristics of cloud computing.

The level of maturity of virtualization has increased over time and now hypervisors are able to manage the resource pooling and on demand service in an intelligent manner, almost devoid of any human intervention. This is the concept of autonomic computing where the hypervisor is intelligent enough to respond to varying levels of activity by either increasing capacity when the demand rises, or conversely reducing capacity when the activity levels
reduce. This is embodied in the idea of elasticity, which is another essential factor of cloud computing.

iv. Web Services

Over time there has been a need to have intercommunication between varying systems for the sake of facilitating specialized processing or sharing of data done by another entity. This led to the need of uniformity and standardization in the methods of communication between these different systems and thus arose the concept of “web services” as a means to achieve this uniformity.

Web services may be described as facilitators of a standard means of interoperation between different software applications, frameworks, systems or technologies running on different platforms. They are considered as an architectural framework that allows for seamless communication between different applications of different systems. [86] This is usually done by exposing certain methods or services through application programming interfaces (API’s) in order to allow other applications to make requests and receive responses to the requests either from a local call or a remote call.

In order for these methods/services/interfaces to be known by other potential interaction parties, there needs to be a machine-oriented means to express the capabilities of these interfaces. This is usually done by describing them in some form of universally acceptable and understood description standard that can be understood through some bidirectional communication protocol. The common description formats that have been in use are XML (Extensible Markup Language), SOAP (Simple Object Access Protocol) and WSDL (Web Services Description Language) – all usually used together.

Once the interfaces and their respective capability information have been exchanged among potential parties of a communication instance, each party now needs to be able to invoke the required services. That is, each party needs to know how to call the required service using the communication protocol format with which the other party communicates. The protocol format must be mutually understood between the parties. The most common forms of communication protocol formats (also termed as web service architectures or interaction models) that are used today are RPC (Remote Procedure Calls), SOA (Service Oriented Architecture) and REST (Representational State Transfer) [87]. The details of these communication protocol formats are not of great importance to be known in this study. It suffices to know that they generally differ in the amount of coupling/decoupling that they offer between service providers and service consumers.

With the maturity of these 4 major areas of cluster computing, grid computing, virtualization and web services, the original idea of cloud computing has now been able to become a reality. Popular industrial examples of cloud computing in practice include the Amazon Elastic Cloud family (Amazon Web Services), Google App Engine, Force.com (salesforce.com), Microsoft Windows Azure and Eucalyptus.

3.6.2.2. Cloud Storage Technologies

Among the more common technologies of today characterized notably by rapid adoption into the market is the revolutionary idea of cloud storage. “A cloud represents a ‘fuzzy’ container for data,
and the user doesn’t really care how the cloud provider implements, operates, or manages the cloud. A client, through the medium of a network, makes requests to the cloud storage to securely store and subsequently retrieve data at an agreed level of service.” [88]

If described in a very simple manner, cloud storage allows a user to store content on a network (Internet) accessible location and access the data from any location or device that has internet connectivity. This makes the accessibility of data that one stores on “the cloud” quite ubiquitous given the level of Internet connectivity today. The term “cloud” generally denotes the relative anonymity of the exact implementation details, operation specifics and particular location of ones data once it is stored on the service. The accessibility and availability of the data is the key concern here and other details such as the implementation and location specifics are relegated to a lower order of concern.

Cloud storage is considered an off-shoot of the original concept of cloud computing where clusters of interconnected devices can be harnessed on demand to seamlessly work together to perform a given task. Again the fundamental concept here was the ability to create some sort of a black-box scenario where a user submits input to a system, the location of the internal workings and processing are unknown, however the result is provided back to the user, or stored in a location unknown, though accessible to the user. The storage cloud is made of several smaller computers, devices or systems that are supplemented by some form of middleware that provides a seamless layer of interaction among the smaller components, however transparent to the user. The important thing for the user is that they have an aim and they seek a result through the processing of some goal-oriented system. The location of the achievement of the goal can be considered irrelevant as long as the user achieves the goal they sought after from the system.

One of the goals of cloud computing could be the ability to store data or information allowing for the user to access it wherever and whenever they please, given that they have a network connection to the cloud computing infrastructure. This is where cloud storage sprouted from.

Cloud storage services have been harnessed for both corporate and private individual uses. The demand for these services has been very high and the supply has correspondingly delivered, of not oversupplied the market. Some of the more popular services include: Dropbox, SugarSync, Google Drive, Box.net, ADrive, FlipDrive, FilesAnywhere, iCloud, Microsoft Skydrive and LiveMesh, Mozy, JungleDisk, ElephantDrive among a host of several others. In general there seems to be no shortage in supply of the services and they have crowded and clouded the market with their wide variety of product and service offerings, pivoting on the singular concept of cloud storage. This goes to show the potential seen in this market as well as the capacity for growth and improvement in the technologies in use spurred by the imminent fierce competition.

Cloud Infrastructure Network Implementation Paradigms

When looking at the paradigms through which resources are organized in a network in order to harness the useful capabilities of a cloud (i.e. on demand service (availability), broad network access, resource pooling and elasticity), one tends to think of the virtual order and implied structure among the devices interconnected. This in general could be referred to as the network topology. Two forms of order in the network topologies could be postulated here: An infrastructure-based topology and an ad-hoc topology.

- The Infrastructure based topology refers to a structured manner in which machines, devices or resources are organized such that they maintain their niche, purpose or place within the
requests because of Through network, shared bandwidth. Outside (devices) 3.6.3. contrast performed. a infrastructure be tested circumstances. There is an initial perceived disorder within such a topology though stability can gradually be attained with time. This is characteristic of peer to peer networks and their related systems.

Very few cloud infrastructure services seem to have adopted this as the basic underlying topology. Perhaps this is because it is a relatively new concept and has not yet been tried and tested to prove its versatility. Symform [89] [90] [91] is the only known company, (to the author) at the time of writing, that seems to have adopted this strategy in its purest form as its basis. Others, like most cloud storage services, seem to have adopted some form of hybrid infrastructure that has some part that is ad hoc and some part that is infrastructure based.

3.6.3. Peer to Peer Systems

In contrast to the legacy Client-Server system, a Peer to Peer (P2P) System is characterized by a network architecture that does not contain a distinct centralized server or set of servers that receive requests and issue responses. Every node in a P2P network has the capability of acting as a client and a server depending on the circumstances, the state of the over system, or actions that need to be performed. Additionally an essential characteristic of this kind of system is that resources are shared directly, such that each node can provide part of its resources for use by other nodes in the network, while it can also request to use resources from other available nodes in the network. [92]

Through having network end-point devices acting as the major distributors of resources, P2P networks make use of the potentially underutilized resources that are at the edge of the network. This is important because it adds greater capacity to the network when there is an additional workforce, of otherwise underused devices added. The collective addition of resources benefits the entire network as a whole through the extra computing power, storage capacity and network bandwidth. The only major problem that is experienced by this strategy is that the end-point nodes (devices) are usually controlled by individual users and not a central body. This means that the devices can greatly vary in capabilities and can potentially join and leave the network unannounced leaving gaps and disconnects between parts of the overall network. There is an additional problem of disruption through random IP addressing, firewalls and Network Address Translation (NAT). This because most consumer end-point devices are within private networks and they are fed with DHCP leased private IP addresses, not to mention firewall restrictions, which limit their connectivity to the outside world for the sake of security and address depletion. The problem of address depletion
however could possibly be addressed with the onset of IPv6 which could do away with the problems that NAT has brought about.

Due to the relative instability and unpredictability of the connectivity of nodes in a peer to peer system, some form of management framework needs to be put in place to ensure continuity in the wake of the natural “chaos” in the system. This is where the concept of “overlay networks” comes in, where the peers themselves through a collaborative effort manage the instability brought out in the addition and desertion of nodes. The overlay network is a type of application layer protocol that manages the distribution and retrieval of data among the available peer nodes in a P2P network. This involves managing the procedures involved when a node leaves the network and its data has to be maintained, when a node joins the network and is looking for data as well as the routing of management messages among other tasks. Overlay networks are at the application layer and thus must rely on the barebones communication infrastructure of the standard TCP/IP network stack. One distinct addition that the overlay network needs is that it must operate outside the realm of the DNS system and must maintain its own node naming and tracking system. This is because of the dynamic nature of the joining and leaving procedures, random DHCP allocations, as well as the issues that NAT brings into the equation.

There are generally 3 types of P2P overlay networks:

i. **Centralized Overlays**

The centralized overlay network is characterized by a central node that maintains information about where the required data or resource is located. The network is still a peer-to-peer network per se, because the sharing of the resources is still directly done between individual nodes. The management of the location and status of the actual peers is done centrally in a sort of client server manner. This has the advantage of it being a simple and easily implementable system; however a cloud is cast upon it through the drawback of there being a single point of failure.

One of the common examples of systems that used this was Napster [93], where the storage of data was distributed among a P2P network, however the overlay network was centralized in nature.

ii. **Unstructured Overlays**

In unstructured overlays, both the overlay management and the resources are distributed. Data and information about the location of the data is distributed among the peers and generally no particular peer has all the information about other peers or the location of other data or resources. Resources or data are located or stored at random locations not predefined, or directly known. When a peer is searching for a resource, it floods the network with its request, hoping that at least one node will respond with information about the location of the desired resource. Once a node responds with information directly to the one soliciting (requestor) for the resource, the requestor will directly communicate with the node that contains the desired resource.

Redundancy of information throughout the network is a necessity because of the randomness of the joining and more importantly the leaving of nodes, however this does not entail a 100% guarantee that the desired resource will be found. This is because there is a possibility of the TTL (Time to Live) value of the message expiring if the distance is large, or if coincidentally all the nodes that have the information, or the resource desired, are all
offline. In practice this has been shown to work in the Gnutella [94] system and the event of resources being unavailable is negligible. Another implementation with a similar idea to this is Cyclon [95]. Other popular algorithms making use of unstructured overlays include FastTrack (Kazaa), BitTorrent [96] and Overnet (eDonkey2000).

iii. **Structured Overlays**

The basic concept of structured overlays lies in the fact that data (resources) and information about the location of the data (resources) and the data itself are distributed throughout the system in a structured manner that maintains some form of order and usually uniformity of the distribution. The overlay network topology is tightly controlled and resources or data are not placed at random peers, but at specific locations that would make subsequent queries more efficient. [97] Additionally, this tries to ensure that there is an element of balance such that there is no bias – favouritism or, conversely, overloading that could affect certain edges of the network. This offers some sort of primitive form of load balancing.

For most, if not all cases this strategy is implemented using Distributed Hash Tables, which are in their simplest form a large hash table that is distributed evenly throughout the network. A hash table contains “key-value” pairs where the key is a uniquely identifying parameter that is directly related to a particular value that is being sought after. The value could be any resource imaginable and the key just needs to be able to maintain some sort of permanently unique relationship with the value. The common way of maintaining this “key-value” pair relationship is through using one way hashing algorithms that have been tried and testing such that there are as little collisions as possible (as collisions are inevitable because of the Pigeonhole problem).

The method of distribution of the resources or data throughout the network while maintaining the key-value pair relationships needs also to be determined in order to maintain some order. This is usually done through making a relationship between the key composition and a similar identifier relating to the desired location of the node where the data is to be stored. Commonly this done by computing a similar hash as that performed on the data/resource and relating the resource hashes with the location hashes in some way such as numerical value closeness or some other means.

The issue of maintaining a balance of even distribution of information about the location of resources also needs to be addressed. This is because hashing functions could inadvertently result in the data/ resources being assigned with a bias to certain nodes. This is solved through ensuring that the hashes adhere to the property of *uniform distribution*. This ensures that there is little or no bias and that the hashes are unpredictable and uniformly distributed throughout the available hash-space thus ensuring a balance. Hashes that adhere to this are usually referred to as *uniformly distributed hashes*. In some specific cases a non-uniformly distributed hashing function may be chosen in order to create a centralization of nodes and resources depending on the capacity and demand for resources of the different players in the P2P network.

Summarily, in this method the data or resources are distributed throughout the network according to the distribution of the hashing function and its relationship to the location of each node. Each node needs to know how to look-up data with the help of other nodes
through collaboration. This is because each node holds a set of pointers to other known nodes that might indirectly lead to the sought after data or resource. These other known nodes are generally not entirely randomly chosen, but are picked based on careful selection of an algorithm that enables each node to have both “far reach” and “near reach” of the network. This facilitates efficient traversal in order to look for sought after resources.

Commonly known algorithms that make use of structured overlay paradigms include Chord [98], Kademlia [99] and Scribe [100].

3.6.3.1. Important Properties and Metrics of P2P Systems

As the number of different P2P algorithms increase together with the corresponding amount of variety of techniques used to solve inherent problems of the predecessors, there has been an effort to standardize the characteristics as well as the base substrate of what a P2P network should be composed of. With structured P2P overlays this seems to be a possibility as explained in [101] and [102], however with unstructured overlays it is a bit more difficult because of the lack of the standard use of DHT’s (Distributed Hash Tables).

All the same, given the diversity of characteristics that each particular algorithm may be composed of, in the attempt to improve on a previous version, there are a certain set of properties that could determine the quality and usefulness of a P2P Overlay network algorithm. These properties, as outlined in [97] are described below:

i. Level of Decentralization, which refers to the amount of distribution brought out in the different components of the system.

ii. Architecture/Network Topology, that is, virtual structure that the overlay system adopts, based on its algorithm operation

iii. The Lookup Protocol: This is arguably the most important component of a P2P overlay because it describes how a node locates another node that is being sought after. This affects the routing performance, that is, the efficiency of the transfer of messages between nodes, which is another metric that should be taken into account.

iv. Scalability, which refers to the overall capability of the system to effortlessly increase in size, without a corresponding substantial deterioration of performance, or increase in complexity of operation.

v. Peer Churn (Peer Join and Leave Behaviour), which refers to the reaction of the system as a whole, generally in terms of performance when nodes either join or leave.

vi. Reliability / Fault Tolerance, that is, the ability of the network overlay to withstand drastic unexpected changes in peer churn or unexpected faults, thus maintaining near normal performance.

vii. System Parameters and Tuning: This is the ability of a network overlay to be modified on demand using system properties based on the needs or demands of the circumstances or environment. It is becoming a more common trait as systems work in more dynamic and complex environments that need constant tuning.

viii. Security: This is a characteristic that is needed in all systems today, not just P2P overlays. Naturally the foundations of security lie in the ability of the system to sustain and maintain
facilities related to confidentiality, integrity, availability, authentication, authorization, accountability and non-repudiation.

This enumeration of descriptive properties is considered to contain only the most common and critical properties of a good peer to peer system. It is considered by no means an exhaustive list of what could be considered imperative for forthcoming systems. These are only the foundational properties and metrics for the currently known P2P systems.

3.6.3.2. The Peer Interaction Decision / Choice

When constructing an overlay network, the so called “Peer Interaction Decision” is critical for the survival of the network as a whole. This imminent choice of which peers a node should interact with usually determines the overall structure (topology) of the overlay network, that is, whether it is centralized, structured or unstructured. It also determines other very critical factors such as the level of availability, the resilience to massive drop-outs or other adverse conditions, the diameter of the network, the amount of clustering, the speed of network traversal, the amount of bandwidth usage and overhead, the speed of repair and healing after adverse conditions, the speed of convergence and the overall utility or performance capacity of each individual node.

This choice is inbuilt as part of the algorithm in the application layer software that each participating node contains. The common peer interaction choice frameworks are general of 2 forms, that is, random and deterministic.

i. The **Random interaction framework** usually takes up the form of Gossiping /Epidemic message dissemination algorithms, or flooding-based systems.

*Gossiping/Epidemic message dissemination* is fully random whereby each node communicates with a fixed number of randomly picked peers with which to communicate the information it has, or to share (exchange) information depending on which information is considered more fresh according to the synchronization of clocks. The method of randomly discovering peers differs from one algorithm or system implementation to the next, and there is no real standardization yet.

*Flooding based systems* rely on flooding the network in a systematic manner however the actual spread is not fully predictable with time based on the speed of spread being affected by individual node performance. To avoid heavy replication and unnecessary redundancy in communication, this relies on the fact that while flooding is going on, when a node discovers that the node it wants to communicate the message to has already received the latest message, then it should gracefully abort and move on flooding to other uncharted nodes. Another way of doing this is having a TTL (Time to Live) value be assigned to the message such that there is a limited amount of times that it can be communicated to others to avoid the unnecessary extra overflow of messages in the network once every node has received the message in the flooding process

Random interaction frameworks generally result in the unstructured form of overlay networks.

ii. **Deterministic interaction frameworks** usually follow algorithms with predictable or almost predictable outcomes of the choices based on the input. The input varies from the hash of the date or resource, the location of a node, the bandwidth of a node, the disk capacity, the node’s uptime, among other parameters that could be beneficial to the network’s aims.
There are generally 3 main forms of deterministic interaction frameworks. These are: *Uniform distribution systems, Hierarchical system structures and Publish-Subscribe systems.*

- **Uniform Distribution Overlays**

  These are usually characterized by the use of circular topologies of interconnected nodes. They make use of the concepts of uniformly distributed hashing as well as consistent hashing in order to uniformly distribute resources throughout the identifier space. The identifier space is finite, thus the circular nature of the topology as the upper boundary is merged with the lower boundary.

  They usually make use DHT’s (Distributed Hash Tables) which are distributed evenly through the overlay network, as the main method of distributing location information of resources. Each node thus has equal capabilities and equal opportunities of requesting for resources, storing resources, storing location information about resources, or forwarding (routing) of intermediary traffic between a source and the intended destination.

- **Hierarchical Overlays**

  Hierarchical overlay systems are characterized by the presence of specially endowed peers that are usually termed as super-nodes or super-peers. They generally have more capacity in terms of resources, neighbour connectivity, uptime or other critical metrics depending on the needs of the system. Normal nodes that are in the system are either permanently or temporarily assigned or connected to a super-peer depending on the algorithms, functionality or the needs of the system. There may also be some sort of specialization, division of labour among the nodes in the form of levels of hierarchies as well as implicit or explicit delegation of supervisory authority and control of capabilities.

  The general structure that is distinctive of this type of interaction framework is that there are levels within a hierarchy. A good example of this is seen in the FastTrack protocol seen in Kazaa as well as the similar proprietary protocol seen in Skype.

- **Publish-Subscribe systems**

  Publish-Subscribe systems are sometimes referred to as a category of event notification systems or message dissemination systems. They are generally composed of message publishers and message consumers, where the publishers produce the messages, or announce events and the subscribers are interested in receiving the messages. In order to receive the messages, interested nodes must subscribe to the given notification service provided by the subscriber.

  This method is adopted in order reduce the amount of unnecessary message overhead that is passed over the network to uninterested parties. In this way it makes the network more efficient and more easily scalable in that it can handle larger amounts of information (messages) being passed on, as well as a correspondingly larger number of nodes within the network.

  Publish-subscribe systems generally have an ad-hoc form of a hierarchical topology where subscribers have direct or indirect communication with one or more publishers. Typically, the system is characterized by 3 functional layers, that is, the overlay
infrastructure, the event routing system and the algorithm used to match events to subscriptions.

Summarily the various properties and metrics that have been discussed so far are those that have been found through literature review and seem to be common among P2P systems in general. However, it must be noted that these properties and metrics have been developed based on the requirements of particular systems in specific environments. Suffice to say that there might be some hitherto unforeseen parameters that may come up in this system that has the modern Digital Investigation as its environment.

### 3.6.4. Crowdsourcing Mechanisms

Crowdsourcing is described as a “participative online activity in which an individual, institution, non-profit organization, or company proposes to a group of varying knowledge, heterogeneity and number, via a flexible open call, the voluntary undertaking of a task. The undertaking of this task of varying complexity and modularity, and in which the crowd should participate bringing their work, money, knowledge, and/or experience, always entails mutual benefit. The user will receive the satisfaction of a given type of need, be it economic, social recognition, self-esteem, or the development of individual skills, while the crowdsourcer will obtain and utilize to their advantage that what the user has brought to the venture, whose form will depend on the type of activity undertaken.” [103]

In simpler terms this means that crowdsourcing involves a large task that is usually too difficult (mundane, repetitive, resource intensive or time consuming) for one entity to handle alone. Thus, this large task is split up into smaller flexible and modular tasks that are offered out to an open group of possible participants, who contribute to the subset parts according to their abilities. The contributors/participants in the crowdsourcing activities derive some sort of benefit either directly or indirectly from the effort put into the given task.

There are generally 2 major types of crowdsourcing [104]:

- **Explicit Crowdsourcing:** This is the form of crowdsourcing where the task at hand is directly broken up into smaller subtasks and these subtasks are offered out to the public. The voluntary participation of individuals in the solving of the subtasks directly contributes to the solving the larger problem at hand.

- **Implicit Crowdsourcing:** This is the form of crowdsourcing where a large task is infused into another massively populated participative host event, and as a byproduct of individual participation in the host event, the [*infused task*](#) also gets solved. There is usually an element of indirect and involuntary solving of the infused problem, as a byproduct of the desire to participate in the host event. An example of this is the reCAPTCHA project where the participants indirectly contribute to character recognition systems capturing contents of scanned books, while at the same time directly being tested, in an authentication event, whether the participants are actually human. There are generally 2 types of implicit crowdsourcing to be considered, that is: Standalone Implicit crowdsourcing, and Piggyback Implicit Crowdsourcing. The differences here may be quite subtle, however the former is built with the crowdsourcing (infused task) initiative in mind, while the latter picks up the crowdsourcing initiative (infused task) after a crowd has already begun using the service and
seems to be providing useful data. The piggyback form seems to have an inclination towards
data-mining in it.

The general concept of crowdsourcing is still new and at the moment it is still undergoing changes. A fixed and globally accepted definition of its essence has not yet been achieved. At the moment the essence seems to lie in the concept of distribution and delegation of human effort – as it is the main exploitable force when resources are considered scarce; however, in this study, we propose to have the various personal devices around the individual human person act on his behalf in order to contribute to the large task at hand. In this case, the resource intensive task at hand entails the collection, filtering and preparation of evidentiary information for law enforcement agencies, in the event of computer related malevolent activities. As has been discussed earlier, there are massive amounts of information transmitted among today’s information systems, and law enforcement agencies and forensic analysts are overwhelmed by this barrage of information when faced with computer related crimes. The available resources controlled by any single entity are simply not enough and are quickly overstretched in such cases. Crowdsourcing, and its related automation capabilities, seems to provide a possible means of solving large resource intensive problems that is worth pursuing, as has been seen with its ability to solve massive problems as seen in [105].

3.7. Virtualization and Hypervisors

Virtualization is a term that has come to be associated with the decoupling of a consumable resource from the physical infrastructure that supports the provision of this consumable resource. According to VMware, one of the pioneers of commercial virtualization products, it is described as “the separation of a resource or a request for a resource from the underlying delivery of that service.” [84] Virtualization has become progressively popular because of the advantages that it brings in terms of better hardware utilization, scalability and flexibility of system management. [84]

With respect to computers and their architecture, there are certain services that are considered the primary building blocks necessary for a computer to achieve its computation goals. These are: memory management, processing and input/output (I/O) services. These services in turn correspond to particular resources that can be seen as the actual facilitators of the services. These are the memory, the CPU and I/O devices. Decoupling of these particular resources, perhaps for scalability and utility reasons, from the respective services that they provide, is the starting point of virtualization. Virtual memory is a simple example of this, where the working storage memory is decoupled from the volatile memory chip and is temporarily stored on a hard disk (an I/O device).

Another example of virtualization in practice is the concept of Desktop Virtual Machines such as VirtualBox, VMware Workstation/Player and Windows Virtual PC. They are probably the most easily perceived and widely spread virtualization concept today. The basic principle behind these is the virtualization of the 3 basic components of a computer in order to facilitate optimization of the utility of available resources. This allows for users to have multiple operating systems/platforms run at the same time, sharing the same overall resources allowing for better utility of resources. Such abstracted platforms running atop another platform are usually termed as guest operating systems, with the latter being termed as the host operating system

Xen, KVM, QEMU, Microsoft HyperV and VMware ESX are the common server oriented virtualization platforms that are in essence founded on the same principles. However, due to the criticality of the kinds of services provided on servers, the management of the resources must be highly precise and meticulous. Thus, a dedicated management service, that has a higher privilege level, is introduced.
This management service is usually termed as a hypervisor or a virtual machine monitor. Traditionally the operating system has had the highest supervisory control of hardware resources that it makes use of. However, in order to have other operating systems make use of the same resources for the sake of improving resource utility, another element must be introduced in order to manage, distribute and control the resource usage among operating systems. Necessarily, this controller element – the hypervisor – must have the highest privileges.

In recent times hypervisors have become the subject of provision of security services. This is because of their natural position in the hierarchy of privileges – due to their necessity of controlling the sharing of resources among guest operating systems. Garfinkel and Warfield in [106] explain the various benefits that isolation, through virtual machine monitors (hypervisors), contributes towards the overall security posture of a virtualized architecture. Some of the benefits outlined include the separation of applications within distinct domains such that vulnerabilities cannot cross the domains; the ability to implement finer grained security policies since individual virtual machine domains are task specific; the ability to lockdown smaller more specific areas of concern; and the ability to quickly recover to the original state due to various logging and snapshotting capabilities available on most virtualized environments. Vogl in [107] further emphasizes the importance of isolation. He also posits that hypervisors have very specific goals and thus would have much smaller code-bases leading to less complexity and thus are more easily secured. Finally he suggests that moving security out of the realm of the virtual machine and into the supervisory control of the hypervisor is a more robust method to ensure security of the entire virtualized guest system, even in the event of this guest being compromised.

Douglas and Gehrmann in [108] highlight virtualization as a strong enabler of security, citing the inherent isolation of virtual machines as well as the major role that the highly privileged hypervisor can play as a trusted base for providing security services. They say, “A hypervisor has great visibility into and control over its virtual machines, yet is isolated from them, and thus forms an apt base for security services of many and varied persuasions.”

In our study we propose the use of hypervisors as the first point of detection of malicious activities similar to [109] and [110], however we go further to use them in the collection of artifacts of potential evidentiary use. In [111] the authors demonstrate the capabilities of using hypervisors to collect memory images from live running systems. In our study, among other novelties, we propose to extend this to use hypervisors to perform a comprehensive seizure of all the sources of potential evidentiary information from a running system. For our purposes, we consider a “comprehensive seizure” in a digital investigation to be composed of images of disk storage, volatile memory dumps and network traffic captures. Naturally, if a comprehensive seizure is made, it is more likely that a comprehensive analysis could also be possibly done.

The inherent security capabilities of hypervisors, given their relatively small code base, reduced complexity, higher privileged access and level of visibility into and control over their virtual machines, make them an ideal platform for the provision of the necessary levels of integrity that would ensure the forensically sound capture of artifacts for analysis in a digital investigation. Thus our choice of technology to perform the initial capture of artifacts of potential evidentiary interest is motivated.
4. The Concept of the Solution

This section begins with a summary of the major problems currently besieging digital forensic investigations especially in terms of resource scarcity in the collection, filtering and preparation of data for further expert analysis. From this, the fundamental requirements are drawn out of what should comprise a worthy solution to this problem. Other similar or closely related solutions are also taken into consideration. This section is concluded with a detailed description and analysis of how the best aspects of the previously discussed technologies can be moulded together towards providing a unified solution. The particular elements, protocols, structure and architecture of this solution are also described in detail.

4.1. Background: Recap and Summary of the Problems

4.1.1. The Backdrop of the Situation

Daily activities – be it personal, family or business related – are now more than ever being influenced by the use of information systems. Automation is the key selling point and almost everything we use today has some sort of computer input, processing, output or data storage activity attached to it. Government organizations, health care providers, transportation providers, communication systems operators, businesses and even home appliances are all slowly but steadily getting hooked up onto the internet and are sharing information. All these entities are moving towards computerizing and internetworking their normal set of activities. This is the age of the “Internet of Things.”

With this trend of “IP-fying” (Internet Protocol-fying) everything, human activities have taken on a computerized form. These activities include not only the beneficial /benevolent ones, but also the malicious and malevolent ones. The activity of keeping rogue destructive human activity has transcends the digital divide. Keeping malicious activity on the Internet and on computerized systems at large has now become widely commonplace and law enforcement agencies are having to cope with this relatively new spate of criminal activity.

The amount of information traversing between varying systems is also on the rise. The amount of information being created and transmitted from one point to another is at an all-time high. The computerization and “IP-fying” initiatives worldwide have helped this on. It is now becoming extremely difficult for law enforcement to sift through the barrage of information being directed to them on the grounds of computer based crime. It is not easy to immediately visibly cordon of spaces on the cyber-realm for further forensic analysis. All the data remotely related to the scene of a digital crime has to be appropriately collected and preserved. This has proved to be an insurmountable task so far as it involves a great investment of time, resources and skills in order to gather petabytes of information and process them within a time-frame considered reasonable by the courts of law.

A real-time automated solution to this problem of collection, preparation and preservation of colossal amounts of information on interconnected computer systems, in an efficient manner, is sought after.

4.1.2. The Problems

Given the highly dynamic, digitalized and interconnected scenario present in our current times, the duties and responsibilities of law enforcement entities can easily become insurmountably difficult to
cope with, leaving them highly inefficient and largely inept at performing their duties of bringing perpetrator of malevolent activity to book. Looking at the big picture, one can easily identify the dynamicity, the digitalization and the interconnected nature of the environment. With a deeper analysis, one can identify that the problem arises from the *scarcity of resources* in a *distributed environment* to deal with the required tasks in an *appropriate and efficient manner*. These 4 issues can be broken down into the following areas of concern:

- Highly distributed information systems
- Complexity of interactions
- The amount of information generated
- The resources needed to sift through large amounts of information
- The relatively short period given by the legal system to present evidence
- Maintenance of legal requirements of integrity of evidence
- Detection of the violation of the security policy so as to trigger data collection
- The problem of discerning what is relevant and what is not relevant
- The methods for collection and preparation of information for further processing
- The maintenance security and privacy while performing automated data collection

These are some of the possible problems that one would encounter when involved with a potential digital crime scene. The list may not be exhaustive, but it contains those issues that are considered important as the basis of this study.

### 4.2. The Live Evidence Information Aggregator (LEIA)

Having laid out the landscape of problems that are being encountered and battled with on a daily basis on the scene of digital investigations, we now attempt to briefly describe the system, outline the key features, characteristics (behaviour) and areas of concern that such a system, as LEIA, should embody in its functionality – as an assistant to law enforcement agencies.

#### 4.2.1. What is a LEIA?

The system that is termed as a Live Evidence Information Aggregator (LEIA) is envisioned as a hypervisor-based peer-to-peer distributed system that leverages on the collaboration between all possible participating entities on the Internet of Things to collect information of potential evidentiary use in a real-time manner. The collaboration is facilitated through the peer to peer overlay network controlled by the LEIA itself and supported by a persistent backend cloud architecture. The hypervisor platform provides a secure virtualized base through which evidentiary information can be collected from the overlying software infrastructures in a manner that ensures forensic soundness through minimal interference. Practically, this means that the LEIA forms a layer of managed virtualization between the hardware and the software of an already existing network of devices that have a particular niche within another distributed system with separate end goals.

All devices that participate in the LEIA functionality are equipped with the hypervisor layer that, through a virtual machine host, enables the capabilities of collection, preparation and distribution of information of potential evidentiary use. In some ways this may be said to be a variation of the
“Mobile Agent” paradigm, however in this case the software is not packaged to move around independently through a network. The hypervisor and the virtual machine host software are resident on every individual machine/device within the LEIA network and do not move. The hypervisor however, does have robust communication and processing facilities to enable it to cooperate with other similar machines/devices with the similar hypervisor platform in place.

In this way the LEIA network maintains a P2P overlay where information of potential evidentiary use is collected, pre-processed and stored in a distributed and collaborative manner in order to overcome the speed, efficiency, availability and resource scarcity drawbacks that have previously been experienced.

So far this only provides the basic overview of the proposed core functionality and envisioned architecture of the system. The next few subsections outline the specific areas of concern, the key features and the desired characteristics of the LEIA for it to be considered a useful tool.

### 4.2.2. Areas of Concern

As described briefly in the previous sections, the aim of the LEIA is to assist law enforcement agencies to collect and prepare potential evidentiary information in a rapid and efficient manner. The general overview of the architecture has been presented and now this section aims at drawing the reader’s attention to the particular areas of concern that the system itself will have to take into consideration. These areas of concern are outlined below:

- With the advent of the Internet of Things (IoT) and already numerous devices and systems heavily relying on the interconnectivity of the internet, the concept of *distributed systems* (processing, storage, communication) is a core area of concern that any comprehensive system that aims to collect information for forensic purposes needs to focus on.

- Another result of the Internet of Things and related distributed systems is the amount of information transferred from one entity to another. There are *massive amounts of data and transactions* crisscrossing over computer networks. This is usually in the form of requests, responses, payloads, messages, commands, statuses, flags and other semantically more meaningful data as part of various communication protocols. The relative ease of creation of information is also spurred on by the IoT.

- Due to the massive amounts of information available it has been seen that it is an extremely difficult task to pick out relevant evidentiary information, let alone store the data, process it or re-trace the events back to the culprit. Generally computationally difficult tasks can be solved with equivalently massive computational capabilities. However, individual entities at the moment do not really have the computational power to match the size of the datasets involved in the data collection problem of digital forensics. This boils down to the problem of *resource scarcity* with respect to the problem at hand. In an effort to solve this problem, a collaborative effort is sought after.

- Given the sensitivity of the information being handled, the *security, accuracy and integrity* of the systems needs to be guaranteed. This is mainly because of the fact that the information is to be used in a court of law, and likely to decide the fate of a person, a company or some other similarly important entity.

- The legal requirements of digital evidence must also be taken into account. The way that the information is collected, pre-processed and stored should adhere to these requirements of
handling digital evidence. The Daubert Standard and the “chain of custody” are of particular concern here.

- As the system is to be used by several parties at the same time, as well as being a critical support structure for law enforcement agencies, the availability of its services at all times, and on demand is of paramount importance. Redundancy and the ability to undertake massive workloads without collapsing also comes into play here.

- Due to the presently accumulated backlog of digital forensic cases, and the foreseeable future of forthcoming crimes involving large data sets with intricate and complex interconnectivity between involved entities, it is imperative that any proposed solution has processing speed and efficiency of undertaking tasks at the heart of all operations.

- In the Internet of Things where almost every conceivable tangible item is perceived to be able to hook up to the Internet, there is bound to be a reasonable amount of heterogeneity among the interconnected elements. This is also a supported by the fact that most vendors and manufacturers of devices are independent and usually adhere to their own set of principles and standards. This would prove to be a challenge as well as a necessary requirement in order to be able to integrate and aggregate information from invariably different devices. Thus the LEIA would have to implement some sort of standardization among its core functionality.

For the moment these are considered the major areas of concern for the foundation of a LEIA. As further analysis and development of the system proceeds, other points of concern may be raised, though these suffice for the moment.

4.2.3. Key Features and Desired Characteristics

Drawing from the previously discussed areas of concern that the system should embody, the following desirable characteristics emerge:

- **Distribution**: The first area of concern is the ability to deal with massive amounts of distribution in terms of participants, data storage, processing and dissemination. The system needs to be able to handle the heterogeneity that may come with distributed systems as well.

- **Scalability**: Large scale interconnectivity, as well as the possibility of new entities joining the system, as well as others leaving the system dynamically and gracefully without drastic negative effects on the system is also another point of concern. The ability to easily improve or extend the capabilities of the system through new modules should also be considered.

- **Availability**: Being distributed and involving several parties, constant availability of the system to the participants is important.

- **Universality**: With the impending aspect of heterogeneity and lack of standardization among vendors of different systems, there needs to be some form of standardization and common understanding between the systems on the level of communication and storage of potential evidentiary information.

- **Responsiveness**: In order to best collect potential evidentiary data or initiate the procedures to do this, the system should be able to detect when a security policy has been irrecoverably violated, thus collecting information in order to pursue the perpetrators of the criminal
4.3. Related Work

4.3.1. Previous Work done and Originality

From the inception of Digital Forensics and throughout its steady growth, there has always been talk of the amount of data and processing power, and thus, time and effort, that is needed in order to bring a cybercriminal to book. In recent times there have been initiatives to centralize processing and data collection onto centralized mainframes and data storage centres, respectively [24]. There has also been a push towards having the different parties involved in solving a case work together, particularly the legal experts and the techies of digital forensics. [112] Collaboration has been the mainstay of the attempt to get cases solved faster.

Reducing the amount of data that is needed to be collected is also a means of reducing the amount of time needed to analyze the data. This has previously been done through “Known File Filtering” as well as through heuristic analysis of the entire captured data. Network Security Monitoring has also been an avenue for gathering data through the assistance of Intrusion Detection Systems assisted through Artificial Intelligence. However, this has been the specific mandate of the Intrusion
Detection System, centralized or distributed as the case may be, with terminating (end) devices or intermediary devices generally play very minor roles in this task.

As far as is known to the author, there has not been much done, through any single initiative, in terms of expanding the scope of data captured to be the mandate of all possible devices of reasonable capability. Enabling individual devices to natively act as part of the Incident Response System, towards the aim of collecting potentially evidentiary data, has not been widely studied. Additionally, collaboration has been emphasized on the human processing level, but it has not been introduced among unrelated networked devices. These devices could possibly be harnessed to work together towards aiding in intelligent real-time capturing, filtering and processing in order to attain and retain that which could be considered as possible evidentiary data, antecedent to the event of a crime being detected. It is for these reasons that we delve into this area to explore it further.

The diagram below visually depicts this idea in 4 quadrants juxtaposed against a Cartesian plane indicating the directions that Digital Forensics and Incident Response (DFIR) research has been taking with respect to collaboration and the components involved in a digital investigation. The top right-hand quadrant indicates the area where not much research has been done, and thus our chosen area of focus, while the rest of the quadrants indicate the progression of previously done studies.

Figure 12: Diagram representing the directions that automation of DFIR is taking.
The proposed idea that this study covers is composed of several areas of specialization, namely: The Internet of Things (IoT), Intrusion Detection Systems, Peer to Peer Networks, Cloud infrastructures, Ontology Taxonomies, The Semantic Web and Virtualization infrastructures. Most of these technologies have been previously harnessed in different capacities, singularly or in small clusters, towards the benefit of digital forensics for today’s complex internetworked and intertwined cyber realm. However, to the author’s knowledge, there has so far not been any work done that aims to merge all these technologies together in order to provide a singular scalable solution that solves the recurring problems of large amounts of data, several sources of data, heterogeneity among systems, insufficient processing power, security and privacy – that are constantly troubling digital forensic analysts and law enforcement agencies worldwide.

Notable works that are similar to this study include [113] that describes and categorizes a wide range of network forensics tools according to their usefulness in the general digital forensics process model. It also outlines several proposed frameworks that bring these tools together to create idealistic architectures of network forensics evidence collection and processing systems. Conspicuously missing from this survey of tools, frameworks and architectures is the notion of peer-to-peer networks as a means of facilitating collection, communication and processing. The use of hypervisors and virtualization in the as a means of providing a more managed, secure and forensically sound evidentiary information collection platform is also not discussed in this survey.

[114] describes an interesting perspective for a live network forensics system that provisions varying Intrusion Detection systems on host machines based on their respective resource costs. The system works with virtualized environments where snapshots can be taken periodically. These snapshots are used to revert the system back to the point before the attack begun, so as to implement varying IDS’s to collect different and presumably better information, as well as create attack vector graphs. This presupposes that the attacker will re-enact their malicious behavior repetitively each time their efforts are thwarted by the system. Storage of the potentially evidentiary information in a forensically sound manner is not particularly dealt with in this study, however, the generated attack graphs may be useful in understanding the actions that were undertaken by the attacker.

[115], [116], [117] and describe distributed system architectures with centralized data stores for proactive collection and summarization of evidence. Additionally they are intended to be useful only for closed domain enterprise systems, where there is some form of control and order instigated by the system administrators. In [115] the focus is particularly on text-based documents, where a centralized repository of changes to known documents within an enterprise are tracked from a variety of different devices where the document may be hosted. In [116] a collaborative intrusion detection system architecture is presented. Information and alerts gathered from from various network IDS’s as well as host-based IDS’s are collected , correlated and summarized in order to determine the severity of the attack and to better understand the attacker’s tactics. This is somewhat similar to the idea described in [114]. In [117] the main focus is on remote live collection of data/information from running systems within an enterprise. It brings about the issues privacy, scalability and self-auditing in the distributed forensics system architecture in the use of pre-installed agents within systems to assist in the remote collection and analysis of data. The software agents perform collection of data, while analysis and correlation are performed on a centralized server, where the investigation is managed from. Reporting is also coordinated from the centralized server.
Other network-based architectures that have been proposed to be used for collection of data of potential evidentiary use within a networked system include [118], [119], [120], [121], [122] and [123].

In terms of pre-processing and creating a framework for the uniform, seamless data exchange and communication of forensic artifacts between heterogeneous entities, [124] is the most relevant resource. Here, a detailed ontology for digital forensics entities is described with particular focus on the artifacts of interest found within the Windows registry. This study also suggests that such an ontology may be used to facilitate creation of knowledge through further machine learning and data mining processes.

The Open Forensic Integration Architecture (FIA) is discussed in [125]. It describes an architecture for the “integration of digital evidence from multiple evidence sources in a technology independent manner.” [125] It assumes that acquisition of the relevant data from different sources has already been performed and is available in a readable format which is provided to the described system which assimilates the data into its knowledge bases in the process of “Evidence Composition.” It is then able to perform further analysis and infer correlations through cross-referencing with previously stored data, or other databases of related publicly available information.

Similar to this idea of normalization and integration of data of potentially evidentiary use from disparate sources include the ideas from the Common Digital Evidence Storage Format Working Group [126]. Other efforts where this idea is also developed include [127], [128], [129], [130] and [131, 132].

The motivation in [134] is quite similar to the one in this study, in that they aim to provide a platform through which various entities involved in a digital forensic investigation can collaborate on an investigation in order to speed things up, consequently to reducing the time taken to perform analysis. However, in this case, the authors present a means through which collected evidentiary data may be easily availed to geographically separate human analysts thereby facilitating collaboration over great distances. The main enabler of this is the use of “NAS over SAN” storage technologies through which the entities involved in the forensic investigation can easily store data in one centralized network accessible location, as well as easily transfer it (or make it available) to their counterparts in different locations over high-speed networks. This idea is noteworthy, however it lacks the ideas of storing the data in a “flat” and normalized format that could enable data-mining or knowledge extraction from the data in order to harness the power of machines to co-relate data.

The work done by Redding in [135] is the most relevant and closely related study done in the area of pro-active and collaborative computer forensic analysis among heterogeneous systems. Redding proposes a peer-to-peer framework for network monitoring and forensics through which network security events can be collected and shared among the peers. “Analysis, forensic preservation and reporting of related information can be performed using spare CPU cycles” [135] together with other spare, under-utilized, or unused resources. The system described, however, is designed for an “administratively closed environment.” This means that all the devices that are within the domain of this system are centrally controlled. An administratively open system is what is not dealt with in Redding’s work, and thus it is what is sought after in the current study as will be described later in this paper.

Somewhat also slightly relevant to the foundations of this study is [136], which, through the “Computer History Model”, brings together the various forms of analysis that can be performed for
forensics purposes on any complex computer system. The Computer History Model is founded on the principal that “every computer has a history which is a sequence of states and events that occurred between two times.” [136] The concept of a unified digital forensics framework is developed and it is shown that there are a wide variety of frameworks or models that are used as the basis for analyzing what artifacts are of importance in a digital forensics investigation involving a complex computer system.

4.4. The Architecture – Details

The hypervisor-based peer-to-peer distributed system that is proposed as the solution to the previously discussed problems is composed of four critical and intrinsically distinct subsystems that work together to provide the required services. The first part is the Host-Based-Hypervisor System; the second is the Peer to Peer Distribution Architecture, the third is the Cloud-based Backend system and finally, the fourth – the Law Enforcement Controller System. The first 3 are the most important parts of the system that provide the necessary underlying services. The fourth is considered as a supervisory, management and querying service that maintains high-level control over the entire system.

The Host-based-Hypervisor (HbH) System has its main aim as the foremost malicious event detection system that also initiates collection of relevant data of forensic interest as the need arises. It is meant to be a secure hypervisor that maintains confidentiality, integrity and privacy of the user’s data while also at the same time responding to forensic data collection needs as determined by the overall controller of the LEIA system, or alternatively the owner of the host device. Data is to be stored in its original bit form maintaining forensic soundness through cryptographic hashing. Additionally RDF metadata is to be generated by the hypervisor system in order to describe simple relationships among the stored forensic data.

The Peer to Peer Distribution Architecture (P2P-da) is meant to facilitate the rapid distribution of data, particularly that of evidentiary use, from participating devices in the network towards the Cloud-based Backend System. It is envisioned that as soon as “useful data” is identified and is required to be stored, it is split into pieces and temporarily distributed among several “more potent” participating devices which would act as seeds in uploading the data to the Cloud-based Backend System. In this way the data storage burden is temporarily shared among devices, redundancy is introduced, better availability is achieved and the overall throughput of data towards the cloud is increased. Rudimentary processing such as indexing and creation of RDF metadata on the small pieces of data may also be facilitated through the temporary distribution afforded by this subsystem. The architecture of the peer-to-peer network providing the aforementioned functionalities is foreseen to be a hybrid between hierarchical overlays, gossip-based overlays and the Bit-torrent P2P protocol. Confidentiality, Integrity and anonymity of the data being passed around are also to be ensured on this level.

The Cloud-based Backend (CBB) System provides the main data storage resource. The data from the individual host-based hypervisor systems is stored here in a distributed database/filesystem based on Apache Hadoop. Hadoop is renowned for its inbuilt reliability, redundancy and large scale availability through peer to peer sharing among data storage nodes. The MapReduce capabilities in Hadoop are also to be harnessed in order to manage the large quantities of data acquired, thus effectively dealing with the “Big Data” problem presented. The CBB also contains a Differencing Engine (DE), that in conjunction with the HbH, facilitates hash based comparisons in order to
Differentiate between known operating system files and those that may be suspicious. This is done in order to narrow down the size of potential evidentiary data from participating devices. Furthermore, the simple processing that could be facilitated within the Host-Based-Hypervisor System and the Peer-to-Peer Distributed Architecture is augmented here. Availability, confidentiality and integrity of the actions performed on the data stored in cloud infrastructure are of paramount importance, as the cloud serves as the central last resort, and the main support pillar for the entire system.

The Law Enforcement Controller System acts as a supervisory control end-point that directly interfaces with the Cloud-based back-end infrastructure and indirectly with the Host-Based Hypervisor system through the Peer-to-Peer Distribution architecture. The main aim of this end-point is to control and manage the cloud service as well as to provide an interface through which Law Enforcement Agencies can submit on-demand queries to the LEIA system in order to produce almost live datasets containing information of evidentiary use from the cloud and the participating devices. The Law Enforcement Controller System, if enabled with enough computational capacity, may also be used to perform more complex operations on the data stored in the Cloud-based Backend System. Such operations may include extracting knowledge from the data using machine learning techniques and data mining procedures in order to assist in time-lining events, correlation of artifacts among other forms of forensic analysis.

Further details describing the intricate functionality and possible implementation details of these subsystems will be described further on in this section. However, in this study we will not dwell much on the intricacies and abilities of the Law Enforcement Controller System as it is considered beyond the scope of this study.

The diagram that follows depicts the 3-core lower-level subsystems, while that beneath depicts the 4 subsystems of the LEIA as a layered model.
4.4.1. The Host-based Hypervisor System

The Host-based Hypervisor (HbH) serves 6 major functions:

- The first function is to be a point of detection of malicious activity, whether it is an incoming event through some sort of input or communication interface, or whether it originates from an element that is already resident on some resource within the host.

- The second major function of the HbH is to facilitate collection and filtering of data from the host that may be used as evidence in a digital investigation, or as artifacts that may assist in corroborating events that may have occurred on a system. This function relies heavily on the accuracy of the malicious activity detection capabilities as well as the correspondence between the Cloud-Based Backend Differencing Engine (CBB-DE).

- The third major function is provision of data back-up services for the user (that shall be motivated later). The HbH would either collect and compress user specified data, or take logical incremental images of the user data to be later securely backed up on the cloud.

- The fourth major function that the HbH provides is temporary storage capabilities. For 2 core reasons: Before data is backed up on the Cloud-based Backend it is first temporarily stored on the device as the primary data source prior to being sent out; secondly in tandem with the Peer-to-Peer distribution architecture, the data from any given HbH is temporarily redistributed and stored in the memory of neighbouring peers, before seeding uploads to the CBB.

- The fifth function that the HbH provides is small scale processing capabilities. The HbH generates RDF metadata describing the data artifacts that have been collected.

Figure 14: The Live Evidence Information Aggregator (LEIA) as a 4-Tiered Layer Model
• The final function that the HbH provides is a means to inform stakeholders when a security policy has been violated and that data is being securely collected in order to attempt to remedy the situation.

Fundamentally, the Host-based hypervisor (HbH) is the starting point of the system where malicious activity is first detected within the entire LEIA system. The detection is facilitated through host-based intrusion detection systems that are embedded within a secured hypervisor that in its turn also hosts the guest operating system. Additionally any software within the guest operating system that provides some sort of security mechanism, such as an antivirus/ anti-malware tool, an intrusion detection system or a firewall, may also collect data and or logs that maybe used by the HbH. Periodic comparisons of known system files against the CBB-DE may also be used as indicators of potentially suspicious activity.

In order to improve the accuracy of alerts and data collection, individual HbH systems may communicate with each other through the Peer-to-Peer Distribution Architecture (P2P-da) in order to share information so as to harness the intelligence of the crowd in detecting surreptitious malicious activity. This could be done among smaller peer clusters, localities or neighbourhoods which have some commonality, or shared goal among them, or alternatively with the help of devices indifferent and external to the shared goal so as to provide a third party perspective of the locality. This crowd-oriented collaboration of HbH systems could also help in reducing the false positives and false negatives, commonly plaguing IDS’s, as well as to improve the accuracy of locations where data should be collected as a priority.

The HbH is able to directly interact with the file system, the network interfaces, the memory caches and perhaps other low-level resources that are usually primary sources of information of evidentiary use in digital investigations. The embedded IDS’s (em-IDS) also collect information mostly in the form of logs which are parsed to result in alerts.

The HbH maintains a SHA-1 hash-list (Local - Known File Hash-List, L-KFHL) of all the files that were placed on the devices’ persistent storage, particularly of operating system files. A corresponding hash list, referred to as the Master - Known File Hash-List (M-KFHL), is also maintained at the CBB in order to later facilitate data reduction. Any additions of applications and their data to the user platform are updated on the HbH Local File Hash-list and automatically synchronized to the CBB, after having been cross-checked against the M-KFHL, via the Cloud-based Backend Differencing Engine (CBB-DE). User data and the corresponding hashes are also maintained as a form of “profile” linked to the particular device in question. The user data profile in its entirety can be subjected to on-demand back-ups.

The diagram below depicts a pictorial representation of the aforementioned description highlighting the key components of the HbH sub-system.
As it may be noticeable, the HbH seems to be directed at terminal end-points of networked devices that are generally used by end-users. However, this is not entirely the case. As has been discussed earlier, the overall system is meant to work in the environment of the Internet of things. Thus, the HbH is directed at *all devices*, that is, both terminal user-end devices (such as desktop computers, laptops, phones, televisions, fridges and other home electrical appliances), as well as intermediate dedicated security and networking equipment that act as intermediaries, or proxies within a network, and other devices with other dedicated uses in larger core mission critical systems. Examples of such devices include dedicated firewalls, intrusion detection systems, routers, switches, embedded devices and even programmable logic controllers (PLC’s) used in Industrial Control and Automation Systems.

The storage capabilities, as already mentioned, are used for 3 main reasons: To temporarily hold a data back-up for the user; to cache information of evidentiary use for a digital investigation, prompted by the local intrusion detection systems; and to temporarily store small pieces of

![Sample Device Diagram]

*Figure 15: The Host-based Hypervisor (HbH) System and its components*
information of evidentiary use from other neighbouring peers in the network, before all these different categories of data are rapidly uploaded to the CBB via the P2P-da. Emphasis should be laid on the fact that the data is stored in temporary storage through the HbH, and is to be held in an encrypted form immediately after being collected, before being sent out to the CBB.

The storage facility works in such a way that each device participating in the LEIA system contributes a very small percentage of its working storage resources. This is in order to provide a temporary distribution of seeds of the data in order to provide the high upload throughput to the CBB. Devices or computer systems that are better endowed with larger working storage facilities contribute a slightly larger amount of space to the common-shared pool. This contribution is also affected by the relative availability and accessibility/reachability of the device in question. Devices that are on public IP addresses and those that are equipped with redundancy or fail-over mechanisms, in order to ensure their availability over computer networks, will generally be required to contribute a slightly larger percentage of their working storage capacity. Alternatively relatively smaller proportions of working storage may be required if there are generally large numbers of reliable devices available.

Mobile phones and networked “home appliances” (such as TV’s, fridges, microwave-ovens, baby monitors and radio systems) generally do not partake in computationally intensive workloads for long periods as part of their normal activities. They are usually used for achieving singular aims and are mostly in a stand-by mode for most of the day. Theoretically, it may be possible to make use of the resources from these devices while they are in low power modes, or not in use. The down side, however, is that utilizing resources from mobile devices that are not plugged in may quickly drain batteries, and impair other more prerogative uses. However, notably, the latter (i.e. networked “home appliances”) are usually perpetually plugged-in to power sources and if endowed with reasonable resource capabilities, they may be of much benefit to such a system as the one being described in this study. Thus, relatively unordinary factors, such as power availability and resource usage patterns must also be considered as key factors in deciding where to source spare or unused resources.

In this matter of sharing computational resources, devices on private networks, behind firewalls or within Network Address Translation (NAT) edges of private networks, need particular attention because of their relative inaccessibility. Furthermore, they are also important as there are probably a lot more devices within these private networks than on the current overall publicly addressable IP space. Techniques to by-pass the downsides of NAT functionalities (i.e. NAT Traversal techniques) need to be considered in order to acquire the usage of machines on private networks. Alternatively “private domains” may be created, such that devices within privately administered networks may participate only with other devices within the same private domain – sharing only resources within these controlled areas. This would assist greatly with privacy and confidentiality considerations of this system.

However, given that this system is meant to be running in the ideal IPv6 environment of the "Internet of Things", perhaps some of these issues – particularly with Network Address Translation and private IP addresses within private controlled networks – could be alleviated to some extent. The dawn of IPv6 will likely add some noteworthy solutions and quirks to this system and thus must also be taken keenly into consideration.
4.4.2. The Peer-to-Peer Distribution Architecture

The Peer-to-Peer Distribution Architecture (P2P-da) is meant to provide a reliable, scalable overlay network among heterogeneous devices. It is also aimed at supporting high levels of churn, while at the same time providing high responsiveness and high throughput.

The main activities of the P2P-da are outlined as follows:

- To share and aggregate information about available reliable peers in order to determine those with better resource capabilities (Epidemic/Gossiping protocols)
- To maintain a hierarchical peer to peer network where there is control and delegation of duties depending on the capabilities of the available peers and their location within the network (Gradient overlays/Hierarchical overlay like Skype)
- To facilitate sharing and aggregation of alerts from purported incidents of compromise among individual HbH systems in order to filter out inaccuracies and support more directed data collection (Epidemic/Gossiping protocols)
- To ensure availability through redundantly distributing pieces of information to a neighbouring set of peers, while uploading (Hierarchical Bit-torrent)
- To facilitate rapid data capture from participating devices through enabling re-distribution of captured data from the target HbH to nearby potent peers. (Hierarchical Bit-torrent)
- To enable rapid upload of data from the (seeded) potent HbH swarm towards the CBB system (Bit-torrent)
- To achieve reliability and availability in the face of high levels of churn
- To maintain scalability and responsiveness given large numbers of peer nodes.

The P2P-da is seen to be a convergence of various P2P protocols. The different protocols contribute to varying particular functionalities – as described above – and are aimed at ultimately providing the necessary services for the P2P-da to fulfill its essential duties. The link between the chosen protocols and their particular niche application in the P2P-da lies in the individual strengths that each of these protocols inherently possess. The particular P2P protocols that will contribute to the formation of the overall P2P-da functionality are: Epidemic (Gossiping) protocols, Gradient (Hierarchical) overlay protocols and the Bit-torrent protocol. The protocols may vary from the original standard implementations and may also form hybrids among them in order to achieve the intended goals.

4.4.2.1. Maintenance of the P2P Overlay

When focusing on the maintenance of the general P2P overlay, algorithms from gradient overlays [137] and epidemic/gossiping protocols [138] [139] are to be used. The neighbour selection and the sharing of the utility metrics are to be facilitated through the use of gossiping as is described in [137]. In brief, this is done through each peer maintaining 2 sets of neighbours, that is, the random set and the similar set. Peers periodically present both their sets to one of their random neighbours in a process of exchanging peer knowledge. Each peer picks an item, representing a peer that has the most similar /closely related utility metric, from the newly received set and adds it to its “similar set.” In this way, if the utility metric is based on how well endowed a peer is, clustering of similar performing peers occurs and thus the core stable peers in the gradient are congregated.
exchange of the “random set” serves the main purpose of reducing the probability of partitioning occurring in the network due to excessive clustering.

In addition to this, each peer should maintain another 2 lists: One that contains a set of better endowed peers, that is, the master list; and another containing a set of less endowed peers, that is the subordinate list. This is done in order to maintain a constant gradient relationship embodying a hierarchy. The master list is created in a similar way that Sacha et al. [137] discover stable peers using the “Gradient Search” algorithm that they propose. Each peer attempts to find other peers with the highest utility metric among their neighbours. Particular notice is paid to peers with a local maxima utility in this algorithm in order to avoid peers having a lopsided master list that could contain a local clustering of peers. A list of previously recently known peers is maintained in order to avoid cycling of lists among the local maxima group. The subordinate list is populated simply through the exchange of lists via gossiping and a set of the closest lesser-privileged peers are maintained till the next exchange.

The table below summarizes the various lists that are used in maintaining the different relationships between peers that are necessary to maintain the hierarchical orientation of the overlay.

<table>
<thead>
<tr>
<th>List Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Similar Set</td>
<td>Maintaining knowledge of neighbours that have a similar or closely related value of the utility metric.</td>
</tr>
<tr>
<td>Random Set</td>
<td>Reducing the possibility of pre-mature localized clustering of peers with similar utility metrics.</td>
</tr>
<tr>
<td>Master List</td>
<td>List containing the known peers who are better endowed with resources, that is, a slightly better utility metric value than the peer at hand.</td>
</tr>
<tr>
<td>Subordinate List</td>
<td>List containing the known peers who are less endowed with resources, that is, a slightly worse utility metric value than the peer at hand.</td>
</tr>
<tr>
<td>Recently Known Peers List</td>
<td>A list to maintain knowledge of recently met peers so as to avoid cycling of peers within a local maxima group.</td>
</tr>
</tbody>
</table>

Table 2: The various lists of neighbours used in maintaining the Hierarchical Overlay

It must be noted that these aforementioned lists are maintained throughout the lifetime of a peer. However, they are refreshed periodically through the peer gossiping, in order to ensure that each peer is always up-to-date with the latest information from its environment.

Notably, the better endowed peers that are clustered towards the core are generally considered to also have a more stable connection with the CBB System. Thus, they provide more reliable proxy uploads of data from the individual HbH nodes that may be further out in the gradient topology created.
4.4.2.2.  Dissemination and Aggregation of Malicious Behaviour Information & Alerts

The sourcing, distribution and collection of elements of information concerning malicious behavior is a crucial part of the system. This is because the accuracy, integrity and timeliness of this information greatly determines the efficiency of the system particularly in making decisions of where and when to collect information, as well as what to collect.

The actual aggregation of the information of evidentiary use is for the most part dealt with in the HbH system. However, the messages passed around between collaborating systems, in order to facilitate efficiency in detection mechanisms, determine the severity of a security incident, the parties involved in the incident, to initiate collection of data, to facilitate communication and distribution of the actual data, are all enabled through the P2P-da.

This communication that occurs among the collaborating systems is categorized mainly in 2 forms of messages, that is, management messages and security incident control messages

i.  Management messages could include the periodic sharing of software updates, detection heuristics, malware fingerprints and known system file hash signatures. These messages have a particular origination from the CBB, however, they are shared among individual devices through the P2P-da. Management messages can also originate directly from HbH devices and travel to other peers, or eventually to the CBB. Such messages include HbH status messages and community/domain/group threat-level status messages. The distribution of software updates, detection heuristics, malware fingerprints and known system file signatures is meant to keep the individual HbH systems up to date with the latest mechanisms for detecting malicious behaviour. The sharing of HbH status messages and community threat-level status messages is meant to provide a mechanism to maintain control of the security posture of individual entities on a network, specialized communities and the network at large. In this way there is collaboration among the entities in the LEIA network that share information and thus contribute to some form of “crowdsourced” global knowledge.

ii.  Security incident control messages could include notifications of detected malicious events, file hash comparison checks, requests and responses for data collection, quarantine notifications and other advanced incident response commands. These messages generally come when a security incident is realized. Malicious-Event notifications are meant to be specific messages that distribute information about the presence of particular malicious events or entities together with information about the source of the malicious event, when it was detected and the potential targets. File hash comparison checks are on-demand requests for known system file hashes against the CBB-DE. Quarantine notifications are dispatched if a certain part of a network, or a certain set of devices needs to be separated from the rest of the network in order to stem malicious behaviour, or in order prevent destruction of evidence. Data collection requests and responses are the messages that initiate and control the data collection process when an incident of malicious behavior is detected.

The resulting consequence of the traversal of the aforementioned communication messages is that the process of collection of information of potential evidentiary value is triggered at the right times, in most critical areas, and eventually the most relevant information is collected and transferred over to the CBB system. This process is dealt with in the following section.
The table below summarizes the typical types of messages that are involved in the general functioning of the malicious event detection part of the system. As can be seen, they are mainly messages used for the dissemination of information in order to assist in making decisions.

<table>
<thead>
<tr>
<th>Message Type: Example type of information / functionality:</th>
<th>Management Messages</th>
<th>Security Incident Control Messages</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Software Update information</td>
<td>• Malicious Event notifications</td>
</tr>
<tr>
<td></td>
<td>• Detection Heuristic update information</td>
<td>• File Hash comparison checks</td>
</tr>
<tr>
<td></td>
<td>• Malware fingerprint information</td>
<td>• Data collection requests/ responses</td>
</tr>
<tr>
<td></td>
<td>• Known System File Signatures</td>
<td>• Quarantine notifications</td>
</tr>
<tr>
<td></td>
<td>• HbH Status Messages</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Threat-level status messages</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Summary of types of messages exchanged in the P2P-da for Malicious Event Detection

4.4.2.3. Incident response data collection

Once a malicious event triggers a data collection event at a certain HbH system, the locally existing data surrounding the malicious event is rapidly collected. The decision of the extent of this data is determined by the HbH System in itself, through its inbuilt IDS’s as well as any other information it has received from other collaborating devices, or other IDS’s within the devices’ guest operating system. Additionally the correspondence with the Cloud-Based Backend-Differencing Engine (CBB-DE) assists in filtering known system files through facilitating the hash comparisons.

The data that is collected is acquired either in a logical form, through the operating system’s calls (which would be operating system specific artifacts), or through a physical bit-level data dump from the physical media. Typical datasets collected could include disk images, live memory captures from working storage, suspicious files with known malicious hash signatures or unknown hash signatures, and network packet captures seized through the network interface cards.

Based on this data, a primary analysis for IP addresses or hostnames is done (either on the HbH System, or delegated to the CBB). This process would indicate which other networked devices have recently communicated with the device in question. The P2P-da then directs messages to the other incriminated devices that are within the reach of the LEIA system, in order to initiate similar data collection procedures. This procedure is recursive and as many nodes as possible are contacted and data collection procedures are initiated. The identifiers (such as IP address, MAC Addresses and hostnames) of previously seen devices are logged centrally at the CBB system. Each logged device should send periodic heart-beats to indicate that they are already collecting data, as well as their progress status of this process.
The collection of the actual data of forensic interest is the most central and critical event of the entire system. There are 3 main parts to this major event:

i. **Data Partitioning**

Once the HbH has made the decision on which particular datasets from the particular guest device are to be collected, the extraction process is begun. If some parts of the determined datasets involve volatile data such as live memory, registers or network traffic, then a time range is determined within which either all the data in that time period is collected, or a set of snapshots is taken. Particularly for network traffic a continuous set of the flows of traffic are taken within the given time range. For RAM, registers and other forms of working storage, a small but reasonable number of snapshots is to be taken over the given period of time. For the moment the determination of this optimal number, as well as the parameters that affect it, is not going to be discussed here, and can be considered as future work for improving this proposed system.

Once the data is collected, it is stored temporarily on the HbH system in an annotated data structure that is similar to the structure prescribed by AFF (Advanced Forensics Format) [128]. This data structure will be referred to as the Incident Data Archive (IDA).

Further to this, each IDA data structure is partitioned in equal size pieces that will be referred to as *shards*. Each *shard* has a unique serial identifier with respect to the total number of shards created for the entire IDA artifact, as well as a SHA1 hash identifier (inner hash) so as to later facilitate integrity checks of each shard. Each shard is also to be encrypted and a second hash (outer hash) is calculated on the encrypted data. The encryption infrastructure that facilitates this follows the generic PKI infrastructure, and will be discussed briefly, later, together with other communication security and protocol concerns.

Another file containing metadata about the IDA that has been created, together with the number of shards, their identifiers and their inner and outer hash values, is also created and sent directly to the CBB. This metadata file is referred to as the *reflection* of the IDA. This is done in a way that mimics part of the BitTorrent protocol, which will later facilitate rapid uploading of the IDA data structure to the CBB.
At this point, in BitTorrent terms, the CBB acts as the “Tracker” and will eventually also act as a “Leecher”. The particular HBH that has collected and partitioned the IDA acts as the “initial seeder” of the data.

The diagram below depicts the core file structures, and their components, that are involved in the Incident Response data collection/seizure procedure, involving collection of data, transmission over the P2P-da and eventual storage on the CBB storage system.

![Diagram of file structures](image)

**Figure 17: The file structures involved in the Data Collection/Seizure Process**

ii. **Shard Distribution**

The next phase is to transmit the IDA in the form of multiple shards towards the CBB in the fastest way possible. This is done by the HBH distributing *each shard* to a random subset $S$ of the nearby, more stable/reliable peers that it knows through its *master list*. Depending on the levels of occupation of the resources at these nominated seeder peers (*supporters*), they may pass part of the burden onward to some better endowed peers that they know through their own master lists. When a peer HBH receives a shard that it is meant to upload to the CBB, this shard is referred to as a *hot shard*. If the HBH system is working beyond its normal occupation load, it is allowed to pass a shard onward towards a better endowed peer. The more a shard is passed
upwards to better endowed peers, the higher the “heat level” is ascribed to the particular shard. If the heat level of a shard rises beyond a certain level, termed as the *melting point*, the receiving HBH system must upload it directly to the CBB, or else notify the peers that have recently handled the shard of its inability to do so. These peers then nominate/elect one of them to prioritize the shard in question and immediately upload the shard to the CBB.

Another issue is that multiple shards of the same IDA may end up on the same peer resulting in large portions of the initial IDA depending on this single HBH system to perform a lot of the uploading. This is to be avoided as much as possible so as to prevent there being a higher likelihood of a single point of failure. This will be achieved by each HBH system monitoring how many pieces of a particular IDA it has received with respect to the entire size of the given IDA. This percentage is to be referred to as the *dependency value* which will be bounded by a threshold. If the threshold is exceeded, the peer is obligated to pass on the newly received shards onward to a random small set of peers in its master list who will continue the peer proxy storage process.

The *melting point*, the *optimal number of nominated seeders* that a particular shard should be passed to and the *dependency value* are seen to be the major parameters that need to be measured and optimized through simulation in order to determine their optimal values. The simulation and optimization of these parameters is considered as future work to be considered after the initial design of the LEIA system.

Figure 18: The typical flow of data in the Data Acquisition process via the P2P-da
iii. **Rapid fragment reconstruction**

As soon as a participating HBH system in the LEIA architecture receives a singular, or a set of hot shards, it is meant to immediately initiate uploading them to the CBB. The upload initiation is begun by the HBH system, that holds a hot shard, sending a message to the CBB requesting it to begin downloading. The CBB consequently takes note of (registers) all the “seeders/supporters” calling for a download, organizes them according to the specific IDA reflections that it had previously received, and begins downloading the individual shards. For a particular shard, downloads are initiated from all their respective supporter locations. This is done for redundancy purposes. Similar to the BitTorrent Protocol download, priority is given to the shards that are the least commonly available, that is, *those that have the fewest recorded supporters*.

As each shard is downloaded, its outer hash is verified against that which is recorded in the respective IDA reflection. If the outer hash does not match then the shard in error is discarded, and a download from another supporter is awaited. This is done in a similar fashion to the paradigm in UDP traffic so as to reduce the back and forth nature of message acknowledgements, that ultimately slow down network communication. If the outer hash matches, the shard is decrypted, verified against the inner hash in IDA reflection and then stored awaiting reconstitution. Reconstitution comprises simply appending all the family of shards that belong to a particular reflection of an IDA together, so as to reconstruct a full copy of the original IDA. A hash check is done at the end of this process in order to confirm that the integrity of the IDA has been maintained throughout the network transfer process.

Once successful transfer of the IDA is complete, a basic network clean-up process is initiated. This is done by the CBB sending messages back to the known supporters notifying them that they can now destroy any related shards corresponding to the IDA at hand, if they have not done so yet.

### 4.4.3. The Cloud-based Backend System

The Cloud-based Backend (CBB) system is meant to be the centralized back end storage service, as well as a source of stability for the rest of the system. The hallmark of the CBB is that it is meant to be highly available, scalable, responsive, secure and able to cope with colossal amounts of data – effectively dealing with the “Big Data” problem. Additionally through being a centralized storage service it provides a homogeneous source of data from otherwise heterogeneous origins. For analysis purposes, having the data centralized allows for ease of manipulation, especially in terms of applying machine learning techniques in order to deduce or infer new knowledge.

Summarily, the functionality of the CBB can be outlined as follows:

- To serve as the main centralized storage facility for the data collected by the HBH Systems
- To be scalable in terms of catering for quick an easy expansion of storage facilities
- To ensure security of the data in terms of integrity and confidentiality
- To provide high levels of availability, fault tolerance and reliability primarily through redundancy
- To provide responsiveness in the speed of retrieval and storage of data
- To facilitate the reduction of the data that needs to be stored, through a de-duplication process that involves filtering out known files
• To maintain stability to the rest of the network by providing a set of centrally maintained and controlled HBH super peers, that can also stand as IDS’s for the CBB as a whole
• To facilitate dissemination of updates of software and malware detection heuristics through the centralized HBH super peers

In order to achieve these goals, the CBB system is essentially subdivided into 3 major components: The Storage System (SS), the Differencing Engine (DE) and the HbH Master Peers. The diagram below portrays these components.

![Cloud-based Backend (CBB) System](image)

**Figure 19: Schematic view of the Cloud-based Backend System components.**

The Storage System (SS) provides the raw storage capabilities coupled with the ability to provide scalability, security, availability, reliability and responsiveness. The Hadoop software library [140] is the preferred data storage platform that would be used to provide these capabilities. The specific operation of the SS is not to be discussed here in detail as the functionality of the basic architecture will remain the same, at least for the initial prototype of the system, and it is deemed to be sufficiently described in the Hadoop documentation and related literature such as [141].

The Differencing Engine (DE), whose functionality is to filter out known files before having them stored on the CBB, has part of its functionality provided through map and reduce capabilities of Hadoop. The other part of the DE is to provide individual HBH systems participating in the LEIA system with information on known benign files (kbf) as part of the Master Known File Hash-List (M-KFHL). This is in order to reduce the size of the IDA that will need to be eventually transferred to the Storage System (SS) of the CBB. This is to be done through a query-response mechanism that runs over the previously described P2P-da.

The HbH Master Peers are the particular peers that are directly connected to the core CBB system (that is, the SS and DE). Topologically, they are seen to be on the outer perimeter of the core CBB system and provide an interface to the rest of the LEIA system through the P2P-da. They are to be well endowed with resources. Thus, when considering the hierarchy of HBH systems, which is developed through the functionality of the P2P-da, the HbH Master Peers are meant to naturally assume the “top-most” position of peers in the topology of the LEIA system. They are also meant to embody reliability and most importantly availability.
From a system’s management perspective, the HBH Master Peers are not like any other HBH systems in that they do not have other core functionalities unrelated to their LEIA responsibilities. This means that they essentially are the backbone for the P2P-da and ultimately the provider of connectivity of the LEIA system outwards to the other HBH systems in the wild. An additional responsibility as HBH Master Peers, they also serve as the central point through which system software updates and malicious event detection heuristics are originated from and disseminated outwards to the HBH systems in the wild.

4.4.3.1. The Storage (and Retrieval) Process

Among the aims of the storage and retrieval processes, greatest concern is put upon the availability of the storage media and the scalability of the available storage space. Availability is seen in terms of the ability to withstand faults and high ingress data traffic while maintaining responsiveness even in the event of the possible churn (check-ins and drop-outs) of the actual storage nodes within the cloud. Scalability is seen as the ability to easily and quickly expand the available storage space without negatively affecting the performance of system as a whole.

These major aims of availability and scalability are provided inherently through the use of Apache Hadoop. The Hadoop Files System (HDFS) is the fundamental component of Hadoop and is the main provider of availability and scalability. The HDFS architecture, in brief, is composed of 3 main components: the DataNodes, the NameNode and the client application. [142] The DataNodes (DN) are individual devices that have their own computing power and storage space. They commonly work together in a parallel and redundant manner, forming the core of storage media where data blocks are stored. They are generally organized in clusters normally termed as “racks” in the Hadoop architecture terminology. Any single file inherently has the possibility of being replicated on several DataNodes and this is governed by the configurable parameter termed as the Replication Factor. That is, a replication factor of 3 associated to a certain file or set of files means that the file would be replicated on 3 different DataNodes for the sake of availability concerns. The efficiency of the placement of replicas, specifically in terms of the reduction of latency in the process of retrieval of any of the replicas in the case that a replica gets corrupted is still an open problem. However, there are some simple strategies already in place to reduce latency by storing replicas as close as possible to the original data, usually with one copy in the same rack but on a different DataNode, and the third replica copy on another DataNode on a different, but nearby, rack. In terms of scalability, a new DataNode can be easily added to the system to increase storage capacity by simply instantiating it with the HDFS file system and associating it with an active NameNode.

The NameNode is usually a singular device that keeps track of the all the available DataNodes as well as block locations within individual DataNodes and the operations performed on these blocks of data. The centralized tracking of DataNodes and data block manipulations is necessary so as to ensure that there is integrity within the storage media. This however introduces a single point of failure that is addressed in the latest versions of Hadoop HDFS where there are stand-by NameNodes where the same tracking information (EditLog and Fsimage) from the active NameNode are replicated. In the event of the active NameNode going down, one of the stand-by NameNodes is immediately nominated to take over the block-tracking responsibilities. This ensures a better level of availability and integrity for the storage system at large.

The diagram below depicts the Hadoop HDFS architecture as adapted from the Hadoop Documentation [140], that forms the basis of the Storage System (SS) within the CBB.
On the part of the CBB, each Incident Data Archive that is transmitted to the CBB for storage should be stored as an individual file with a replication factor of 3, as seems to be the common practice. Subsequent changes on the actual HBH system, that are taken as new IDA’s should be considered as appendices to the original IDA and a differencing check should be done such that only new changes to the original files are added. The older files are not modified. Only new files, or recently modified files are added and any previously known files that arrive in latest IDA’s are discarded. The aim of this is to maintain a historical perspective of the changes that may have occurred among the data without the need to store already known data, while also avoiding the loss of older data that may have undergone changes. This is done in order to maintain a minimalistic perspective on the size of the stored IDA’s as much as possible.

4.4.3.2. The Data Reduction (Triage) Process

The Data Reduction (Triage) process is generally undertaken at 2 stages of the data collection process: The Pre-Transmission Stage and the Pre-Persistence Stage. The Pre-Transmission reduction is done with the aim of reducing the amount of data that will have to be transmitted over the P2P-da, while the Pre-Persistence reduction is done in order to reduce the amount of data that has to be stored per device on the CBB.

As per normal operation of the entire system, each HbH system maintains hash lists of all the operating system files as well as the applications added to it and the user data. These hash lists are used to determine which files have undergone changes since the system was initialized. Normally
majority of the changes should be among the user data and the application data. The operating system files should generally maintain their integrity.

Depending on the severity of the malicious activity taking place, which is determined by the HbH system’s IDS’s in correspondence with other HbH systems, the level of the amount of data to be collected is determined. For lower severity levels only logical captures of files that have undergone changes, as determined by the hash comparisons would be included in the Incident Data Archive that is to be sent to the CBB. Additionally network captures would be initiated at the lower severity levels in order to supplement the logical file analysis.

Network captures may be reduced similarly through hash comparisons of known network flows. This may assist in reducing the amount of actual packets that are to be included in the capture. As it may be difficult to perform the full data reduction of packet captures on an individual HbH device due to the global and dynamic nature of network activity, this should be performed at Pre-Persistence stage by the CBB-DE.

If the severity of the malicious events is considered to be higher, then live memory dumps are to be taken in their entirety as well as full physical disk images are to be captured. Similarly, the hash lists are used to discard bit-streams of files that have their original hash values, and thus, integrity, intact.

The table below summarizes the severity levels and the related categories of evidence that are deemed to be necessary

<table>
<thead>
<tr>
<th>Malicious Event Severity Level</th>
<th>Evidence Category Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>High Severity</td>
<td>Full physical disk images, Memory Dumps, Network Captures</td>
</tr>
<tr>
<td>Medium Severity</td>
<td>Full physical disk images, Network Captures</td>
</tr>
<tr>
<td>Low Severity</td>
<td>Logical capture of files, Network Captures</td>
</tr>
</tbody>
</table>

Table 4: Order of Incident Severity and the corresponding evidence category deemed necessary

At the Pre-Persistence stage, a prepared Incident Data Archive (IDA) is delivered to the CBB. The CBB-DE is tasked with checking if there are any files or bit streams that exist with the IDA that are already stored on the CBB for the particular device in question. This is done through comparing the hashes of the files contained in previous IDA’s with those in the newly arrived archive. The matching files are discarded and the new ones are appended to the collection of IDA’s of the particular device. Similarly with memory dumps, they may be reduced by comparing the hashes of the bit-streams in the memory dump with those that are in the already collected IDA’s from the disk data captured as is described in [143] using the MemD5 tool.

With regard to network captures collected, in addition to hash comparisons of known network flows, careful note should be made on identifying protocol tunneling techniques so as not to ignore seemingly harmless traffic that may contain surreptitious data. Statistical methods coupled with Artificial intelligence methods [144] may be used here to augment these data reduction capabilities of identifying known network flows.

At the end of this process an aggregate Incident Data Archive corresponding to a suspect device is assembled and stored on the CBB. This aggregate IDA contains all the suspicious operating system
files or application files, the user data, the reduced memory dumps and the reduced network capture.

### 4.4.4. Law Enforcement Controller System

The fourth and final major component of the LEIA system is the Law Enforcement Controller system. This, though a core component to the larger LEIA system, is not seen to be directly relevant to the current study involving distributed data collection from heterogeneous sources. It is seen to be supplementary to the study at hand and thus will only be described briefly and not considered in any more detail further on.

The Law Enforcement Controller is the main interface that law enforcement personnel would interact with in order to perform their directed forensic analysis for a particular digital investigation case. It is meant to be the system through which law enforcement can initiate specific queries to the data sets stored on the CBB. This is done in order to retrieve detailed, structured information as well as new knowledge inferred through correlation of data originating from different sources that may help in solving a case. The aim of this is to automate otherwise manual tasks of correlating data from different heterogeneous sources in order to pose valid assertions based on the data that could assist a forensic analyst in performing their duties of making sense of digital artifacts.

Additionally, from the new found knowledge patterns of malicious activities are to be learnt and stored. These Malicious Activity Patterns are to be used as feedback to the HbH systems in order to improve the detection capabilities of the inbuilt IDS’s and thereby also improve the accuracy of collection of data of potential forensic evidentiary use.

The use of Semantic Web technologies as the basis for correlating evidence and creating new knowledge as described by Dossis in [145] is deemed to be the core underlying capability that is to support this part of the system.

### 4.4.5. Additional Features of the LEIA System

So far the LEIA system has been described from the perspective of providing data acquisition services for the purposes of facilitating more efficient digital investigations. Apart from this, the LEIA system also has other complementary services that are worth mentioning as they could assist in the adoption and uptake of the system into society at large. These capabilities are:

- The ability of the user to perform on demand back-ups of their device and consequently restore their system based on the last back-up that they took.
- The presence of an inbuilt intelligent Intrusion Detection System that learns from its environment as well as from malicious activity that may have happened in remote parts of other networks.

#### 4.4.5.1. On-Demand Back-ups (ODB’s)

In addition to collecting data for forensic purposes, which is triggered by occurrence of malicious events, the user or owner of a particular device containing a LEIA HbH system is given the ability to back up their applications, settings/configurations of these applications and their own user data. Operating system files are deemed to be standardized and similar across different devices running the same version of the platform, thus they are not backed-up – though a record is maintained of
the actual operating system’s license /registration information, version, patch level and build as it may be useful for a restoration process.

The on-demand back-ups are to be collected and transmitted in a similar way to the process that involves actual Incident Data Archives (particularly with respect to hashing and use of the P2P-da), however they (the On-Demand Backups) are to be stored in an archive separate from any IDA’s related to the device in question. The backups are stored such that there is only one copy upon which any future backups are differenced against and the differences updated onto the original back-up thereby maintaining it as the latest image of the user system. This strategy also helps in maintaining the effective storage space used for back-ups to a bare minimum.

The user/owner of the device can either manually initiate a back-up, or otherwise create a pre-determined schedule when the back-ups should be made.

In the event of the need for restoration of the user’s system, the first step is to restore the operating system platform together with its relevant license. The user’s applications are then restored, and finally the user data is also restored as per the latest back-up.

The notable difference between the Incident Data Archives and the On-Demand Back-ups is that with the Incident Data Archives, the user cannot initiate physical deletion of data. Files that have been collected in IDA’s previously and have been deleted on the user system between the last IDA and the current one are simply marked with a flag as having been deleted. With the ODB’s the user files can be physically wiped out at the command of the user.

4.4.5.2. Inbuilt Intelligent Intrusion Detection System

Having an inbuilt Intrusion Detection System within the HbH system that aggregates data from the alerts and logs from anti-virus and IDS’s on the guest system, in addition to collecting its own logs is the basic functionality of the HbH System. This functionality is augmented with feedback from the Law Enforcement Controller that analyzes the data in real time in order to draw out patterns of attacks for future reference.

From this “intelligence” the HbH can also to be used as an active alert system that notifies the user of malicious activity occurring on their device or in the domain that they are operating within. It can also be perceived that it can be used as a mechanism to trigger intrusion prevention mechanisms in order to avert, curtail or reduce the impact of destructive events.

In this way the user can benefit greatly in that they can be made more aware of the security posture of their devices and the information there in.

4.4.5.3. System Updates

System updates in the LEIA system generally come either in the form of software patches or malicious event detection heuristics. In this way the HbH systems at large are kept up to date. The software patches are initially implemented on the HbH Master Peers and the update process flows outwardly through the P2P-da towards the remote HbH Systems. The updates are to follow the gossiping paradigm as in [139], and thus the system as a whole generally converges to the same updated state. The same distribution strategy follows with the malicious event detection heuristics (feedback) with the only difference being that the feedback originates from the CBB, where it has been previously stored after coming from the analysis performed by the Law Enforcement Controller. The feedback passes on from the CBB onward to the HbH Master Peers and finally
outward to the individual remote HbH systems. As with the software patches being transmitted through the peer-to-peer gossiping paradigm in [139], the feedback eventually converges on the entire system.

4.5. Communication Protocol Considerations

Aside from the particular functionality of the individual components of the LEIA system, the overall security, reliability and availability of the communication protocols, that support the system, are of great concern. The security of the communication protocols is considered mainly in terms of the confidentiality, authentication among participating entities and most importantly the integrity that they afford to the system. The concepts reliability and the availability are considered more on the part of ensuring that the service is delivered as it should be, when it is needed.

4.5.1. Security in Data Communication

Further to the provision of integrity mechanisms at the point of collection and at the storage media, the integrity of the data while in transfer is also of great importance. This is critical because it directly affects the forensic soundness of the data collected and thus the admissibility of the data in a court of law. Integrity mechanisms to ensure the authenticity of the data in transfer are to be built into the communication protocols themselves to order to provide this service.

Confidentiality is a desired property of the communication protocols because the data that is being transferred may contain sensitive, private or personally identifiable information, but not suitable for public audience. In addition to this, inadvertent leakage of the data that is being collected by the system should be avoided so as to prevent ill-intended elements from gaining any information that they may use in order to manipulate the system or other related evidence yet to be collected.

Authentication among participating devices is also a necessary feature such that only validly identified and verified devices can communicate with each other and share the data collected. The aim of this would be to as much as possible nullify the possibility of communicating with rogue devices that may aim to steal data, modify, insert false data, or otherwise perform some form of combination of events in order to affect the proper functioning of the system.

These aforementioned critical security services (confidentiality, integrity and authentication) in data communication are to be provided through the use of the Public Key Infrastructure (PKI) paradigm. The exact description and functionality of how the Public Key Infrastructure is to be implemented on the existing system is deemed beyond the scope of this study, however, a brief description of the main focal points will be given.

At the lowest level the PKI infrastructure makes use of Public and Private Keys through asymmetric cryptography to ensure confidentiality and integrity. The details of how these are ascertained is not to be discussed within this study. Authentication is provided through a combination of keys and certificates. The certificates prove that certain public keys belong to certain entities. These certificates in turn need to be validated by an authoritative “Third Party” in order to prove the ownership. This need for a Third party spurns the need for a hierarchical infrastructure where there is some form of vetting of these third parties by higher level authorities. This brings about the idea of the PKI infrastructure being hierarchical. The exact process of how authentication is achieved in a typical PKI infrastructure is not to be discussed further here, however, it suffices to say that bidirectional authentication between communicating parties can be achieved using the PKI infrastructure.
In practice, the PKI infrastructure needs the presence of some form of Certificate Issuer that should be responsible for verifying an individual entity’s identity before issuing them a certificate. There also needs to be a Certificate Authority (CA) and certification hierarchy that allows authoritative entities within a domain to verify certificates and in turn these authoritative entities can be verified by an individual Top Level Certificate Authority or through cross-verification among a number of Top Level CA’s. At the lowest level of individual entities that need to have their Public Keys certified, there needs to be some form of global, pseudo-global or domain-bound identification mechanism that can ensure that all entities participating in the system can be reasonably identified. The details of Public-Key cryptography will not be dealt with here.

If for some reason a system needs to be decommissioned or isolated, perhaps because it has been seriously compromised, such that it should not be allowed to communicate with other devices, then its certificates need to be revoked. This means that there needs to be some form of Certificate Revocation mechanism also in place. Of the 2 main methods, that is, using Certificate Revocation Lists, and the use of the Online Certificate Status Protocol, the latter is preferable. However, due the nature of the possibility of ad hoc networks in the Internet of Things, perhaps some form of hybrid mechanism that makes use of both certificate revocation mechanisms is needed.

With particular regard for the system described in this study, the Top-level CA functionality could be mapped as a component of the CBB. Lower level certificate authority functionality may be domain specific and given to well-endowed HbH systems within the network. Individual communication could be undertaken in a form similar to that done through the SSL/TLS protocol whereby there is an initial exchange and verification of certificates (in this case bi-directional verification is preferred, as opposed to the SSL default single-direction verification) through the use of public key cryptography, then there is agreement among the communicating parties on the shared key they are to use and then communication occurs henceforth using symmetric cryptography.

### 4.5.2. Reliability & Availability

In order for the LEIA system to be useful to law enforcement agencies, one of the major considerations is the delivery of the stipulated services to their fulfillment, as and when needed. This is embodied in 2 core concepts, that is, Reliability and Availability. Reliability is considered to be the ability of the system to carry out its given task in a “Best-Effort” manner even in the event of a failure in the normal operation. Availability is deemed as the provision of resources as and when they are needed.

Reliability and availability of the storage media has been considered through the use of the Hadoop HDFS. Reliability comes from the use of the Editlog, while availability is provided through replication of the data blocks over multiple data nodes depending on the chosen replication factor. In the transmission of collected data, the replication of shards of a given IDA over a number of participating HbH systems also provides some form of availability. This section considers provision of these aims (availability and reliability) for the LEIA system at Network Layer of the OSI networking stack, as well as in the communication provided in application layer through the P2P-da.

At the Network layer, for the transfer of IDA archives in the form of shards, the TCP protocol is used initially to transfer the metadata about the shards and thereafter the UDP protocol is preferred for the actual data transfer. In general, for the transmission of management related messages, the TCP protocol is preferred in order to ensure their reliable delivery through the use of acknowledgements. In the transmission of the IDA archives speed is critical and thus exchange of acknowledgement
messages of packets of this form of data are as much as possible avoided as they may cause latency and unnecessary amounts of extra traffic in the event of network interruptions. The fact that individual shards are distributed to multiple proxy HbH peers and then retransmitted through a bit-torrent like protocol towards the CBB storage system allows for the overlooking of acknowledgements because the enforced replication provides the necessary redundancy, and thus improved reliability and availability, in the event of failure of transmission of one or more shards. A cumulative acknowledgement of the status of the transfer of an entire IDA may be considered at the end of the transfer process.

At the application layer, where the P2P-da is concerned there is a possibility of high levels of traffic accumulating on the network. This could happen due to persistent lower level network interruptions, resource misalignment due to several overworked HbH peers passing on data to particular common better endowed HbH systems, thereby overloading them resulting in a vicious cycle climbing upwards through the hierarchy, or simply through some form of malicious flooding of traffic towards particular critical peers in the network.

The first two mentioned sources of congestion on the network may be considered as self-inflicted congestion through valid message/data transfer within the network. Methods to avoid such congestion to the network need to be put in place. These methods could include ideas similar to the use of the TTL (Time to Live) value in routing protocols. This would mean measuring the distance covered over the network by the message/data in question, as well as the duration the message has been on the network in order to determine whether the communication has effectively failed or not and is thus causing unnecessary use of resources on the network. The idea is that data traffic that is seen to be getting passed around within the P2P-da without necessarily reaching its intended destination – that is, it reaches a threshold value of hops from one device to another without reaching its destination should be termed as failed message/data communication. This failed message/data should be dropped and its originator notified of this event of dropping the message/data.

As discussed with the P2P-da Shard Distribution functionality, once a higher level peer communicates to a lower one that it is not able to upload a shard to the CBB a nomination process is begun. This nomination process is performed among the collection of lower level peers that has the shard in question, to elect one of the peers to directly upload it to the CBB.

For management messages that fail to be transmitted, since they are over TCP, once they are dropped and the sender is notified, a retransmission of the message is begun.

The final form of congestion mentioned above that may occur is generally considered to be an attack on the system in the form of a Denial of Service (DoS). For the moment, no particular methods for avoiding, preventing, responding or recovering from a DoS are specified as this is considered to be an on-going field of research and a research problem on its own. It is thus is considered to be further work to be later built upon the proposed design of the LEIA system.

4.6. Information Storage Paradigm

As previously discussed, the collected incident information that is resident on a HbH System before transfer to the CBB is stored in a structure termed as an Incident Data Archive (IDA). This IDA is transmitted in the form of shards through the P2P-da, whereby the particular data transfer protocol used to do this mimics the BitTorrent protocol. The shards are re-assembled at the CBB back into
IDA’s related to a particular case. In this section we briefly describe the envisioned format of the Incident Data Archive (IDA), before and after the transfer process, as well as the motivations for this choice.

From [128] and [146] it can be easily seen and motivated that some sort of annotation to the originally collected data provides a number of benefits. Some of these benefits include that fact that the basic annotation of evidential data fosters interoperability among forensic tools as well as facilitates sharing of evidence files particularly when collaboration among specialized analysts is required. The ability to correlate evidence from multiple disparate sources is also enhanced because the annotations can tell the story about a particular block of bytes that may easily help an analyst correlate the data with another set of annotated evidence. The movement towards automation can also be enhanced if the data is sufficiently annotated. Garfinkel in [128] advocates for an open evidence storage format that is annotated with metadata that describes the data with XML-like tags in order to foster forensic tool interoperability. In [146] Garfinkel furthers this idea by creating a number of tools that promote this XML annotation concept together with feasible use cases. Of particular note is the “fiwalk” tool which is the core engine that parses a raw disk image and annotates the presence of volumes, directories and particular files of interest.

Cohen in [129] extends on Garfinkel’s [128] work and develops LibAFF4. He brings in the idea of storage of various types of evidence formats that are usually present in typical digital investigation, such as hard disks, network packet captures and memory images, into the original AFF format. He enhances the distribution capabilities of the original AFF format through subtle changes in data structures and data storage paradigms within an individual AFF file. This allows for greater decoupling of the evidence files from particular analysis tools, allowing for more robust possibilities of sharing the evidence data. The compression of the original AFF files is also improved and this is done through modifying an implementation of ZIP64 towards the goals of allowing for having individual files of greater than 2GB, as well as supporting multiple access capabilities to the same archive.

In [147] Cohen describes the PyFlag architecture, which is a network forensics analysis tool that parses network packet captures (and disk images), storing various found parameters and their values as nodes in a tree data structure. This tree structure is termed as a “virtual file system.” The virtual file system is meant to be a flat data store where simple correlations and search traversals can be performed through the use of “scanners.” These “scanners” are described by Cohen “as modules which operate on VFS inodes and gather specific information about them, sometimes creating new VFS inodes and scanning them recursively” [147]. Simply put they are scripts that parse particular protocols and files in order to extract information and thus create or enhance the tree structure of evidence from a particular case. They thus assist in the integration of evidence data and thereby help in bringing out correlations among different pieces of data. Notably, the PyFlag architecture presupposes that the data of potential evidentiary use from the varying set of suspect devices has already been collected and is ready to be fed into the “scanners”. The massive amounts of data that are typical of digital investigations today are also not taken into consideration. Finally the need for realtime data collection and forensic analysis in response to security incidents is also not taken into account.

The LEIA architecture aims to address some of these issues through its innovative hybrid information storage paradigm. This storage paradigm aims to use Hadoop [140] in combination with the LibAFF4 format [129] and the Resource Description Framework (RDF) in order alleviate the aforementioned
deficiencies of PyFlag. The LibAFF4 format together with the RDF framework are to provide the desired homogeneity and linkability, while the Hadoop software library will provide the capabilities of reliably dealing with large amounts of data. Naturally, parsers similar to those ascribed to PyFlag would be needed in order to attain the RDF-annotated LibAFF4 homogeneous format. This is to be the generic format of the data within the Shards of IDAs as they are collected at individual HbH systems, as well as when they are augmented with other information as they are stored on the CBB Storage system.

The aspect of real-time data collection is not per se a function of the data storage paradigm in itself, but rather a combined effect of the functionality of 3 of the main components of the LEIA Architecture that is, the HbH Systems, the P2P-da and the CBB System. The initial collection of the data is performed by the HbH Systems before the first temporal storage process, in an RDF-annotated LibAFF4 format, is performed. The P2P-da facilitates the rapid accumulation of the data directed towards the CBB Storage system. At the CBB Storage System, the IDA’s undergo further enhancements in terms of annotations, while at the same time they may undergo reductions as per the functionality of the CBB-Differencing Engine.
5. **Towards Practical Implementation – The Proof of Concept**

In this study we have so far described a core part of a comprehensive automated digital investigation architecture, that is the LEIA system, which focuses mainly on the collection (acquisition) of potential evidentiary data in an accurate, rapid and automated manner such that it streamlines the overall investigation procedure. The intricate details of the sub-systems have been explained earlier in this section in a theoretical manner. This in itself is good in that it forms the theoretical underpinnings of the system, however it is not enough in that there may be some hitherto unforeseen practical issues that may come about in practically implementing the system. It is for this reason we attempt to implement a part of the system.

It should be noted that given the size and complexity of the LEIA system, as already described, it is already a herculean effort to just implement one of the sub-systems. Furthermore, within the scope of a Master’s Thesis it would be considered beyond the scope of the project to design the architecture of such visibly large system, as well as build the entire working system, or even one of the sub-systems fully. With this in mind, we intend to pick some generic (basic) functionality that cuts across all the subsystems and that could be useful in showing the viability of the system. We then simplify this generic functionality into a small experiment that serves as the partial proof of concept demonstrating general idea around the larger LEIA concept.

5.1. **The Experiment**

The functionality that has been chosen, in brief, is that of performing data acquisition from a small scale device, transferring the acquired data securely over a network and storing it on a distributed file system storage platform, where it is readily available for further processing / investigative analysis. This particular experiment has been chosen because it touches on some functionality of each of the 3 major components of the LEIA system, that is, the Host-based Hypervisor - HbH - (that performs the actual data acquisition procedure), the Peer-to-Peer Distribution Architecture – P2P-da – (that facilitates the transmission of the acquired data) and the Cloud-based Backend system – CBB – (that receives the data from the HbH system, via the P2P-da, and stores it centrally, ready for further processing). Furthermore, to the best of the author’s knowledge, there are no well-known commercial or open-source forensic tools that perform remote data acquisitions on small scale devices, thus this proof of concept that simulates part of the actual LEIA functionality serves as an innovative step in the direction towards building such tools.

The experiment itself involves 2 main phases, that is, the setup phase and the testing phase. Within the setup phase there are subsequently 2 more stages, that is:

i. The setup of a small cloud-based distributed storage infrastructure to provide the storage facility – serving the purpose of the Cloud-based Backend (CBB) system

ii. The development of a software application that performs the remote acquisition of data, transfers this data over the network securely and stores it on the CBB system. This serves the purpose of both the Host-based Hypervisor (HbH) system in acquiring the data, as well as the Peer-to-Peer Distribution Architecture (P2P-da) in the secure transmission of the data over a network.
It should be noted that even though the actual HbH system involves the use of a Hypervisor and an inbuilt host-based Intrusion Detection System, the development of these particular key elements has been left out of the initial proof-of-concept as they are deemed to be too large to be factored in within this study for the moment. The same constraint also applies to the P2P-da, whereas the algorithms that support this infrastructure are discussed within this study, building the actual P2P overlay would also be considered too large to be factored into this study. Summarily, these individual elements could each be the subject of deeper individual studies that support the building of the larger LEIA system.

The “Testing” phase simply involves the application of the developed tool to collect the data from designated small scale device. At this point in time the major parameter to be measured is the time taken to acquire the data with respect to the size of the actual acquired data. In this particular experiment, for the purposes of the proof-of-concept, we will focus only on disk data and as such we shall be performing a disk dump of the small-scale device. Later studies could measure the performance implications of this tool on the CBB’s resources as well as those on individual HbH systems on different devices. Measurements on network throughput and latency may also be measured in further studies that involve studying the P2P-da performance.

5.2. Tools & Setup

In order to perform the desired experiment, several different components need to be in place and set up appropriately in order to represent the individual elements that compose the proof-of-concept system. The major components that are to be set up include a small scale device that has the capability of being an SSH server and an SSH client (connected to a remote SSH sever), the cloud storage service and a local area network to simulate the network connectivity between the small scale device and the cloud storage service for the acquired data storage.

5.2.1. The Small Scale Device

The device that was used as the small scale device is a Chumby Classic. It is an embedded Linux device whose basic functionality is being an alarm clock, internet radio streaming device, news feed reader and weather information service. It is considered as being a legacy device, having been released in 2006, and probably among the first small scale devices that supported plugin applications, before the generation of the Apple AppStore and the Android Market Place. The Chumby Classic has the following specifications of concern [148] :

- Freescale iMX21 ARM926EJ-S 350MHz Processor
- 64MB RAM,
- 64MB NAND (Solid State Drive)

It runs on a modified Linux Kernel (Busybox v1.6.1), like many small scale devices of today, and thus supports several Linux like commands. The Chumby also has inbuilt SSH capabilities that can be activated on-demand, which makes it particularly suitable for this experiment. It is worth noting that other small scale devices such as Android based devices, or Apple iOS based devices also do have their particular SSH server applications that can be loaded onto the respective operating systems and used towards similar goals. The Chumby is just seen as a particularly convenient small scale device for experimental purposes because it does not have extremely large hard disk space (that
could be prohibitive in performing tests that include repetitive disk dumps) and does not require much configuration for setting up the SSH capabilities.

The given Chumby device has the following file system structure as displayed in the diagram below.

![The File System Information of the Chumby device](image1)

**Figure 21: The File System Information of the Chumby device**

As it can be seen from the output of the “df” command the device seems to have only 40MB of disk space as opposed to what was given as the normal specifications of a Chumby Classic device. The 40MB is partitioned in 2 physical partitions, that is, /dev/mtdblock5 and /dev/mtd/block1 with 38.8MB and 2MB allocated to each partition respectively. According [149] [150] and [151], the partitions labeled “none” are logical partitions in a temporary filesystem (tmpfs) that are dynamically mounted at boot-time and loaded as in memory or to the swap partition if they are needed. The data within this dynamically mounted partitions is lost if the device is rebooted. The output in Figure 22, from the “cat /proc/mounts” command indicates the partitions that are on temporary filesystems.

![Output from the /proc/mounts file indicating the various mounted filesystem information](image2)

**Figure 22: Output from the /proc/mounts file indicating the various mounted filesystem information**

In this experiment we will only focus on seizing the data that is mounted on the “root” partition (/), that is, /dev/mtdblock. The code within the program being built easily allows for modification so as to capture the other mounted partition (/psp) mounted on /dev/mtdblock1. However, for this experiment it suffices to capture just one partition for our purposes, and we choose the “root” partition to be the focus, as it is the area where potentially critical applications would run. The other partitions, that is, /dev, /tmp, /var, /mnt being virtual partitions, as it can be seen that they don’t have physical mount points, are only instantiated at boot time, in a dynamic manner, thus they are considered to be part of the volatile memory of the system, and thus out of the scope of this
experiment. (Furthermore, as it is out of the scope of this study, it has not yet been considered how to discover the physical addresses of such dynamically allocated partitions in order to capture the data on these partitions).

5.2.2. The Communication Network

As the LEIA system relies greatly on network communication among its components, the experimental proof of concept system must also manifest some form of network communication, albeit at a simpler level than what is expected in the real system.

In both the proof of concept experiment and the actual LEIA architecture, there are 2 main areas where network communication is highly relied upon. In the actual LEIA architecture, these 2 main areas are the P2P-da (Peer-to-Peer Distribution Architecture) and the CBB (Cloud Based Backend). The P2P-da ensures interconnection between individual HbH (Host-based Hypervisor) Systems and the CBB. Within the CBB, there are individual nodes that facilitate the storage capabilities. In order for these individual nodes to act together as a single entity (that is, the Cloud-Based Backend), as well as to provide a scalable centralized storage platform, these individual nodes need to have some form of interconnection among them. The diagram below portrays this idea.

![Diagram of LEIA Architecture's 2 main Network Compartments](image)

**Figure 23: The LEIA Architecture's 2 main Network Compartments**

As mentioned earlier, due to the impracticality of building a peer-to-peer network, given the time constraints around this study, a simple Client-Server architecture is adopted to simulate the communication between a single HbH node and the CBB. In order to ensure security on all the communication between all the entities, SSH connections are used. Thus, all communication between the HbH node and the CBB, as well as communication between individual CBB nodes within the CBB system is encrypted.
In this experiment, the Chumby Device (acting as the HbH) is connected through the Internet, over a Virtual Private Network (VPN) to the cloud storage service (CBB) in order to incorporate “internet-scale” distance into the experiment. The VPN allows for LAN-like communication to happen between the Chumby device and the cloud storage service. The communication on this LAN between the Chumby Device and the cloud storage service is further secured via SSH connections to prevent any other parties that might be on the LAN from eavesdropping. Both the Chumby device and a designated Master Node within the cloud storage are equipped with SSH services (both client and server) in order to facilitate this secured communication.

The individual nodes that comprise the cloud storage service are also connected together on their own LAN. They, in turn, also have SSH services enabled on them (both client and server) in order to facilitate secure transmission of management traffic and data blocks while providing the storage services. The diagram below shows the network setup for the experiment.

![Network diagram](image)

**Figure 24: The Networking setup details of the experiment**

### 5.2.3. The Cloud Storage Service

The final and most crucial part of the experiment is the Cloud Storage Service. This component acts as the Cloud-Based Backend (CBB) system within the LEIA system architecture.

The actual setup of the cloud storage for the experiment involves 4 nodes. One of these 4 nodes is a firewall, providing an Intrusion Prevention Service, thus securing the actual storage area. The other 3 nodes are part of the actual cloud storage system that is provided through the Hadoop Software Library. The particular service of the Hadoop Software Library that is used here is HDFS, that is, the Hadoop Distributed File System. As mentioned before, these 4 nodes are interconnected via a
separate internal LAN different from the one that connects the cloud storage system to the Chumby device, however, the Master node, via the firewall, links up the 2 networks.

In terms of physical resources, the 4 nodes that make up the CBB are Virtual Machines running on Virtualbox. The host machine upon which the virtual machines run on is a Lenovo Ideapad Y580 Laptop with 8GB RAM, 750GB hard disk and an Intel Core i7-3630QM Quad core processor, running Windows 7 Home Premium (64bit). The particular specifications of the virtual machines are given in the table below:

<table>
<thead>
<tr>
<th>Node</th>
<th>Specifications</th>
<th>Operating System</th>
</tr>
</thead>
<tbody>
<tr>
<td>Firewall</td>
<td>Processor: 1 CPU</td>
<td>Pfsense 2.0.3 (FreeBSD 64bit)</td>
</tr>
<tr>
<td></td>
<td>RAM: 512 MB</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HDD: 3 GB</td>
<td></td>
</tr>
<tr>
<td>CBB-node0</td>
<td>Processor: 1 CPU</td>
<td>Ubuntu 12.04 LTS 64bit (Server</td>
</tr>
<tr>
<td>(Master)</td>
<td>RAM: 2048 MB</td>
<td>Edition)</td>
</tr>
<tr>
<td></td>
<td>HDD: 32GB</td>
<td></td>
</tr>
<tr>
<td>CBB-node1</td>
<td>Processor: 1 CPU</td>
<td>Ubuntu 12.04 LTS 64bit (Server</td>
</tr>
<tr>
<td></td>
<td>RAM: 768 MB</td>
<td>Edition)</td>
</tr>
<tr>
<td></td>
<td>HDD 32 GB</td>
<td></td>
</tr>
<tr>
<td>CBB-node2</td>
<td>Processor: 1 CPU</td>
<td>Ubuntu 12.04 LTS 64bit (Server</td>
</tr>
<tr>
<td></td>
<td>RAM: 768 MB</td>
<td>Edition)</td>
</tr>
<tr>
<td></td>
<td>32 GB</td>
<td></td>
</tr>
</tbody>
</table>

Table 5: Specification of the nodes that comprise the Cloud-Based Backend System in the Experiment

The version of Hadoop that is set up across the CBB-nodes is version 1.2.0. This particular version was chosen because it was the current stable release at the time of setting up the experiment. Hadoop is Java based and thus requires the Java environment to be present on the systems that it is running on. The Oracle Java SDK 1.6 was deployed on all the CBB-nodes as it was specified as the version where Hadoop 1.2.0 had been tested and shown to be stable.

**CBB-node0** is designated as the master node that performs the communication with the external devices, in this case, with the Chumby device. It also acts as the NameNode managing the DataNodes in the Hadoop storage network as the join and leave the network. Consequently it also manages the information about which data blocks belonging to particular files are stored on which particular DataNodes and their replication on other DataNodes for backup. As the experimental set up is rather small due to resource limitations, **CBB-node0** also runs DataNode services, meaning that it also stores data blocks in a similar manner to the other dedicated DataNodes.

**CBB-node1** and **CBB-node2** simply act as DataNodes and serve the sole purpose of storing data blocks as designated by the NameNode, that is, **CBB-node0**.

As can be seen, CBB-node0 runs a lot of services and performs a lot more tasks than **CBB-node1** and **CBB-node2**, thus it follows that CBB-node0 must be better endowed with resources as compared to **CBB-node1** and **CBB-node2**. Thus the justification of the difference in resource specifications as is seen in Table 5.

The diagram below outlines the experimental setup for the cloud storage service that acts as the CBB system.
5.3. Testing the Experimental Infrastructure

After the experimental infrastructure was set up, a small software application was developed in order to facilitate the capability of making an on-demand disk dump of the small scale device (Chumby), transport it securely over the network and store it on the cloud storage service provided by the Hadoop setup. The application is written in Java. It makes use of the JCraft Java Secure Channel (JSch 0.1.50) library [152] and a Java implementation of “netcat” in order to facilitate the secure transmission of the data over SSH. The application also makes use of the Core Apache Hadoop libraries for enabling the storage on the Hadoop Cloud storage platform. The native Linux “dd” command is also used to facilitate the initial the disk dump.

This code for the application, developed by the author, has been open-sourced and is available for review, contributions and criticism through the Git repository at the following link: https://github.com/irvinhomem/DumpFSHadoop

The software application runs from the Master node in the CBB, that is, cbb-node0. In order to achieve the final outcome of acquiring a disk dump and storing it on the cloud storage platform, it performs the following steps:

- Initiate an SSH connection to the Chumby device
- Initiate “netcat” on the Master Node (cbb-node0) to listen on port 54137
• Perform Local Port Forwarding to enable the “netcat” listening port to listen for traffic coming through SSH from the Chumby device
• Initiate a “netcat” instance on the small scale device (Chumby) device on port 57314.
• Perform Local Port Forwarding
• Execute the “dd” command over SSH on the Chumby device.
• Compress the output of the ‘dd’ command into a tar.gz archive
• Pipe the tar.gz archive output out through the SSH port.
• Receive the incoming file data and pass it on to the Master Node (via a URI reference) to store it on the HDFS storage platform

While performing the software development, the Chumby disk data was acquired over the network over 300 times. This was done in order to test that the software actually performs as required in capturing the disk data of a small scale device. The size of the NAND memory of the Chumby device being just 64MB, and the particular partition being captured being 38MB, allowed for reasonably quick turn-around times while testing.

In addition to testing using the Chumby device, in order to further test the capabilities of the developed proof of concept, the capturing of various partitions of the persistent “disk” storage of 3 additional small scale (mobile) devices was performed. The aim of this testing was to measure the performance of the LEIA proof of concept when faced with different small scale devices that vary in specifications and capabilities. This performance measurement was done through benchmarking of time related and partition/file size related measurements, which are useful in forming a basis for the performance metrics of the most primitive form of remote data collection on such an automated system.

The particular measurements that are taken are:

• The “File Transfer Time” measuring the time taken to transfer the time from the small scale device to the Hadoop Cloud Storage Service (i.e. the CBB).
• The “Partition Size”, that is, the size of the partition within the small scale device that is being acquired.
• The “Used Space”, that is the amount of the partition being acquired that has been filled up with data.
• The “Evidence File Size”, that is the final file size of the acquired (seized) partition after compression, that is, the compressed (GZIP) Incident Data Archive (IDA) that is stored on the Hadoop Cloud Storage Service.

The 3 mobile devices that were picked were a HTC MyTouch 4G Slide, a HTC Incredible S and a Samsung Galaxy Tab 2. There was no real criteria in choosing these devices other than their availability to be used in this study at the time the experiments were being performed. That said, they however still form a reasonably good set for testing because they all have different hardware specifications and platform versions.

It also should be noted that in this small-scale proof of concept of the LEIA system, certain requirements are needed on the part of the small scale devices being captured in order for them to have the capability of being acquired. The major requirements for now are that he devices being captured must have an SSH server with root access and that they have some form of “netcat”
installed on them. On all the devices used in the testing, these requirements were fulfilled and thus they were compatible with the LEIA proof of concept.

The table below outlines the specifications of the particular devices at hand.

<table>
<thead>
<tr>
<th>Device</th>
<th>Platform</th>
<th>RAM</th>
<th>Processor</th>
<th>Chipset</th>
<th>Internal Storage</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTC Incredible S</td>
<td>Android OS v2.3.3 (Gingerbread)</td>
<td>768MB</td>
<td>1 GHz Scorpion</td>
<td>Qualcomm MSM8255 Snapdragon</td>
<td>1.1GB</td>
</tr>
<tr>
<td>HTC MyTouch 4G Slide</td>
<td>CyanogenMod 10.2 Alpha (Based on Android Jellybean 4.2.2)</td>
<td>768MB</td>
<td>Dual-core 1.2 GHz Scorpion</td>
<td>Qualcomm Snapdragon S3 MSM8260</td>
<td>4GB</td>
</tr>
<tr>
<td>Samsung Galaxy Tab 2 (WiFi Only)</td>
<td>Android OS, v4.0.3 (Ice Cream Sandwich)</td>
<td>1GB</td>
<td>Dual-core 1 GHz</td>
<td>TI OMAP 4430</td>
<td>8GB</td>
</tr>
</tbody>
</table>

Table 6: Specifications of the small scale (mobile) devices used in the testing

5.3.1. Test Cases

In order to capture data that would be useful in this experiment, partitions were chosen from the devices according to the amount of data that they contained in them. As may be intuitively noticed, as the devices were intrinsically different, the partitions on the different devices were also of different sizes and different levels of usage. This made the choice of partitions to be captured slightly difficult and thus supplementary partitions and data were added in order to achieve the desired sizes of data. These supplementary partitions and their respective data were not added directly to the file system within the internal memory of the devices, but were added through external MicroSD cards. All the devices used in the testing (save for the original Chumby device) supported supplemental external memory through MicroSD cards.

The aim of the tests was to have fixed amounts of data on partitions (whether on local storage, or on external storage) that would be seized from the devices through the LEIA proof of concept system in order to measure performance capabilities. The fixed amounts of data would be acquired across the 3 devices, particular metrics would be measured and the sizes would be incrementally adjusted for the next round of tests. Overall the idea was to determine how well the LEIA proof of concept would perform given a range of different devices to work on.

In essence, the same architectural set up that was used in testing the Chumby device is used to test the capabilities of the LEIA system on other devices, but with a variety of different data sizes.

One of the problems that the performance testing on this experimental set up had to overcome was the effects of other unforeseen random activity on the external networks that the LEIA was communicating on. These effects were neutralized through repetitive performance of the same captures with the same data over time. In general each test on a device with a particular dataset size being captured was performed 10 times and the averages of the 10 sets of results recorded. For the larger data sizes, where capture times were tending towards an hour, the number of runs was reduced to 5 because of the extreme lengths of time one would have had to endure for 10 tests to be performed. Summarily, approximately 300 data acquisitions from the devices were taken in total.
The tables below summarize the tests. The results are outlined in the following section.

<table>
<thead>
<tr>
<th>Partition</th>
<th>Mount Point</th>
<th>Partition Size</th>
<th>Used Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>/cache</td>
<td>/dev/block/mmcblk0p27</td>
<td>132MB</td>
<td>16MB</td>
</tr>
<tr>
<td>/system/lib</td>
<td>/dev/block/mmcblk0p29</td>
<td>139MB</td>
<td>133MB</td>
</tr>
<tr>
<td>/mnt/sdcard</td>
<td>/dev/block/vold/179:65</td>
<td>2000MB</td>
<td>250MB</td>
</tr>
<tr>
<td>/system</td>
<td>/dev/block/mmcblk0p25</td>
<td>545MB</td>
<td>507MB</td>
</tr>
<tr>
<td>/data</td>
<td>/dev/block/mmcblk0p26</td>
<td>2000MB</td>
<td>1000MB</td>
</tr>
<tr>
<td>/mnt/sdcard</td>
<td>/dev/block/vold/179:65</td>
<td>7300MB</td>
<td>1500MB</td>
</tr>
<tr>
<td>/mnt/sdcard</td>
<td>/dev/block/vold/179:65</td>
<td>7300MB</td>
<td>2000MB</td>
</tr>
<tr>
<td>/mnt/sdcard</td>
<td>/dev/block/vold/179:65</td>
<td>7300MB</td>
<td>3000MB</td>
</tr>
<tr>
<td>/mnt/sdcard</td>
<td>/dev/block/vold/179:65</td>
<td>7300MB</td>
<td>4000MB</td>
</tr>
</tbody>
</table>

Table 7: “HTC Incredible S” Test Cases

<table>
<thead>
<tr>
<th>Partition</th>
<th>Mount Point</th>
<th>Partition Size</th>
<th>Used Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>/vendor/firmware/adsp</td>
<td>/dev/block/mmcblk0p19</td>
<td>16MB</td>
<td>0MB</td>
</tr>
<tr>
<td>/vendor/firmware/misc</td>
<td>/dev/block/mmcblk0p17</td>
<td>199.8MB</td>
<td>21.4MB</td>
</tr>
<tr>
<td>/cache</td>
<td>/dev/block/mmcblk0p24</td>
<td>118.1MB</td>
<td>87.0MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/block/vold/179:65</td>
<td>7300MB</td>
<td>255MB</td>
</tr>
<tr>
<td>/data</td>
<td>/dev/block/mmcblk0p23</td>
<td>1100MB</td>
<td>368.8MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/block/vold/179:65</td>
<td>7300MB</td>
<td>500MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/block/vold/179:65</td>
<td>7300MB</td>
<td>709.4MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/block/vold/179:65</td>
<td>7300MB</td>
<td>1000MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/block/vold/179:65</td>
<td>7300MB</td>
<td>1550MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/block/vold/179:65</td>
<td>7300MB</td>
<td>2000MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/block/vold/179:65</td>
<td>7300MB</td>
<td>3000MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/block/vold/179:65</td>
<td>7300MB</td>
<td>4000MB</td>
</tr>
</tbody>
</table>

Table 8: “HTC MyTouch 4G Slide” Test Cases

<table>
<thead>
<tr>
<th>Partition</th>
<th>Mount Point</th>
<th>Partition Size</th>
<th>Used Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>/efs</td>
<td>/dev/block/platform/omap/omap_h_smmc.1/by-name/EFS</td>
<td>19MB</td>
<td>4MB</td>
</tr>
<tr>
<td>/cache</td>
<td>/dev/block/platform/omap/omap_h_smmc.1/by-name/CACHE</td>
<td>688MB</td>
<td>11MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/fuse</td>
<td>4000MB</td>
<td>250MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/fuse</td>
<td>4000MB</td>
<td>500MB</td>
</tr>
<tr>
<td>/data</td>
<td>/dev/block/platform/omap/omap_h_smmc.1/by-name/DATAFS</td>
<td>4000MB</td>
<td>1000MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/fuse</td>
<td>4000MB</td>
<td>1500MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/fuse</td>
<td>4000MB</td>
<td>2000MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/fuse</td>
<td>4000MB</td>
<td>3000MB</td>
</tr>
<tr>
<td>/storage/sdcard0</td>
<td>/dev/fuse</td>
<td>4000MB</td>
<td>4000MB</td>
</tr>
</tbody>
</table>

Table 9: “Samsung Galaxy Tab 2 - WiFi Only” Test Cases
5.4. Outcomes of the Experiment

As mentioned in the previous section, individual test cases were run several times (in groups of either 5 or 10 runs) in order to obtain averages of the metrics, particularly in terms of the times taken for the capture to be completed. It should be noted that the variations of the “File Transfer Times” that are observed when running each individual test case may be a result of several factors, some of which are beyond the control of this experiment. Some of these factors include:

- External network traffic in the internetworks between the captured device and the LEIA
- The change of base-stations that could result in some changes written to certain partitions
- The change of power status (Batter powered vs. Plugged in to charger)
- Processes running live on the device that make changes to internal storage
- Calls or SMSs received. (Affecting file sizes)
- Updates to applications initiated by software providers (E.g. Google Play Apps)

As the tests were aimed towards being as realistic as possible, no changes were made to the status of the phones, that is, they were left on (live), connected to their respective mobile / WiFi networks as when they were received. Only power was plugged in if the battery went low.

The tables below summarize the average times collected, the number of times the particular test case was run and the eventual file size of the acquired compressed Incident Data Archive.

### Table 10: Results from Test Cases on "HTC Incredible S"

<table>
<thead>
<tr>
<th>Partition Amount used</th>
<th># of Test Runs</th>
<th>Avg. File Transfer time (ms)</th>
<th>Compressed IDA Size (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>16MB</td>
<td>10</td>
<td>13664</td>
<td>1.35</td>
</tr>
<tr>
<td>133MB</td>
<td>10</td>
<td>84600.8</td>
<td>62.1</td>
</tr>
<tr>
<td>250MB</td>
<td>10</td>
<td>392323.9</td>
<td>254.5</td>
</tr>
<tr>
<td>507MB</td>
<td>10</td>
<td>553933.1</td>
<td>506.3</td>
</tr>
<tr>
<td>1000MB</td>
<td>5</td>
<td>978571.8</td>
<td>1003.6</td>
</tr>
<tr>
<td>1500MB</td>
<td>5</td>
<td>1360375</td>
<td>1507.8</td>
</tr>
<tr>
<td>2000MB</td>
<td>5</td>
<td>2932376.8</td>
<td>2044</td>
</tr>
<tr>
<td>3000MB</td>
<td>5</td>
<td>3877676.8</td>
<td>3699</td>
</tr>
<tr>
<td>4000MB</td>
<td>5</td>
<td>4814006.6</td>
<td>4093</td>
</tr>
</tbody>
</table>

### Table 11: Results from Test Cases on "HTC MyTouch 4G Slide"

<table>
<thead>
<tr>
<th>Partition Amount Used</th>
<th># of Test Runs</th>
<th>Avg. File Transfer time (ms)</th>
<th>Compressed IDA Size (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0MB</td>
<td>10</td>
<td>1375</td>
<td>0.02</td>
</tr>
<tr>
<td>21.4MB</td>
<td>10</td>
<td>8583</td>
<td>10.68</td>
</tr>
<tr>
<td>87.0MB</td>
<td>10</td>
<td>31467</td>
<td>41.15</td>
</tr>
<tr>
<td>255MB</td>
<td>10</td>
<td>230709</td>
<td>254.40</td>
</tr>
<tr>
<td>368.8MB</td>
<td>10</td>
<td>304527</td>
<td>391.84</td>
</tr>
<tr>
<td>500MB</td>
<td>10</td>
<td>338180</td>
<td>506.31</td>
</tr>
<tr>
<td>709.4MB</td>
<td>10</td>
<td>1076844</td>
<td>764.68</td>
</tr>
<tr>
<td>1000MB</td>
<td>10</td>
<td>1174482</td>
<td>1202.41</td>
</tr>
<tr>
<td>1550MB</td>
<td>10</td>
<td>1323845.90</td>
<td>1781.64</td>
</tr>
<tr>
<td>2000MB</td>
<td>5</td>
<td>1673928</td>
<td>2617.28</td>
</tr>
<tr>
<td>3000MB</td>
<td>5</td>
<td>2052952.40</td>
<td>3088.43</td>
</tr>
<tr>
<td>4000MB</td>
<td>5</td>
<td>3015056.60</td>
<td>4091.91</td>
</tr>
<tr>
<td>Partition Amount Used</td>
<td># of Test Runs</td>
<td>Avg. File Transfer time (ms)</td>
<td>Compressed IDA Size (MB)</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------</td>
<td>-------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>4MB</td>
<td>10</td>
<td>1235</td>
<td>0.04</td>
</tr>
<tr>
<td>11MB</td>
<td>10</td>
<td>67608</td>
<td>0.86</td>
</tr>
<tr>
<td>250MB</td>
<td>10</td>
<td>286947</td>
<td>254.4</td>
</tr>
<tr>
<td>500MB</td>
<td>10</td>
<td>426783</td>
<td>506.3</td>
</tr>
<tr>
<td>1000MB</td>
<td>10</td>
<td>960952</td>
<td>854</td>
</tr>
<tr>
<td>1500MB</td>
<td>5</td>
<td>1488236</td>
<td>1782</td>
</tr>
<tr>
<td>2000MB</td>
<td>5</td>
<td>2829355</td>
<td>2617</td>
</tr>
<tr>
<td>3000MB</td>
<td>5</td>
<td>2951551</td>
<td>3089</td>
</tr>
<tr>
<td>4000MB</td>
<td>5</td>
<td>3707556</td>
<td>4091</td>
</tr>
</tbody>
</table>

Table 12: Results from Test Cases on “Samsung Galaxy Tab 2”

From the tables above, graphs are drawn in order to ascertain the performance and scalability of the system. This is done by plotting the relationship between the size of the data being captured and the time taken to perform the operation, as well as the relationship between the file size of the eventful Incident Data Archive (IDA) and the time taken to acquire this compressed file. These graphs are portrayed below:

---

**Figure 26: Graphs portraying LEIA metrics with tests on HTC Incredible S**
Figure 27: Graphs portraying LEIA metrics relationships on HTC MyTouch 4G Slide
Figure 28: Graphs portraying LEIA metrics with tests on Samsung Galaxy Tab 2
5.5. Discussion of the Outcomes of the experiment

From the graphs plotted it is not entirely conclusive whether there is a linear relationship, or a more exponential-like relationship between the amount of data used in a partition and the time taken to capture the partition. In the case of the “HTC Incredible S” there seems to be a generally linear relationship; however there are points in the graph where the file transfer time increases at a higher rate leading to some form of exponential relationship at some points. This behavior is seen quite similarly in the “Samsung Galaxy Tab 2” results. However, in the tests with the “HTC MyTouch 4G Slide” the file transfer time seems to take a more exponential increase with respect to an increase in the amount of data in a partition used.

When considering the relationship between the eventual size of the compressed “Incident Data Archive” that is captured and the “File Transfer time” the relationship is seen to be more exponential than linear. It is clearly seen in the case of the “HTC MyTouch 4G Slide.” With the “Samsung Galaxy Tab 2” the exponential nature of the relationship is a little less, however it is still visible. The “HTC Incredible S” however exhibits a step like relationship with parts that seem linear and parts that seem exponential. Towards of the “HTC Incredible S” graph it seems to end in an exponential manner according to the data collected.

Overall from the results gathered in this experiment, there seems to be a greater inclination towards an exponential type of relationship between the data size and the transfer time. In a more generalized sense it seems that the transfer time may become exponentially large as the data sizes that need to be captured increase. This of course depends on the capabilities of the devices from which the data is being acquired/ seized.

Less capable devices with large amounts of data would naturally tend towards having an exponential increase in data transfer time, while those that are better endowed with capacity may exhibit a more linear relationship with little amounts of data, but may experience the same kind of exponential increase as the data size increases beyond a certain threshold. This behaviour is somehow visible in the graphs in that they seem to begin with linear relationships, however at some point there seems to be a “step” in the curve where there is a sudden rapid increase in “file transfer time”.

From the data collected in the tests performed, one cannot make conclusions on whether data sizes larger than those tested (beyond 4GB) would always result in an exponential increase in the “file transfer time”, or whether there would be a step like curve as is already seen. Further testing would be needed to be done with larger sizes tending towards what could be considered “extremely large sizes”. As at the time of writing, the latest available devices in the market have internal storage ranging between 8GB and 64GB. The ones that were available for testing were not the latest and thus their relative capacities were significantly lower.

In terms of other aspects of the experiment outside the testing of the performance capabilities, it is seen that the setup of the Hadoop HDFS cloud storage infrastructure in this experiment allows for scalability just as is expected in the CBB of the actual LEIA system. Practically, extra storage nodes may simply be added to the local internal network “intnet” via the switch. A simple configuration process of loading the Hadoop software library core packages, associating the new nodes IP addresses with domain names and adding these domain names in the register of the NameNode, adds more storage to the overall cloud storage infrastructure. This exhibits the scalability needs of the LEIA system.
In terms of reliability, it should be noted that there were cases during the testing where some of the storage nodes (datanodes) went offline, or encountered data block corruption errors, however the system as a whole was able to recover transparently. Block corruption errors and node connectivity errors were encountered, however the file sizes, hashes and the ability to access and export the captured IDA archives were maintained despite the errors. This shows that the necessary levels of reliability and the integrity of the storage system was attained.

Sniffing of the traffic between the CBB and the device being seized was performed using Wireshark and it was seen that the traffic was encrypted within an SSH tunnel, thus the confidentiality requirements of the system were achieved to a certain extent.

5.6. Drawbacks of the Experiment

While there have been several successes in the performance of this experiment in enabling on-demand live remote device acquisitions and storage on a cloud storage service, it should be said that there are also a number of shortcomings that this experiment has experienced with respect to the expectations of the overall LEIA system architecture.

As was mentioned at the outset of the setup of the experiment, there are various functionalities of the actual LEIA system architecture that were deliberately left out of the experiment due the level of difficulty and time required to develop them satisfactorily. Of particular note are the hypervisor based capabilities that are to be built into individual devices participating devices in the LEIA system; the inbuilt Intrusion Detection capabilities of the HBH systems; the Incident Data Archive format; and the Peer-to-Peer capabilities based on hybrid gossip-based-hierarchical and Bittorrent overlays enabling the rapid and reliable communication among participating devices. These have not been developed or tested, though they have been extensively.

Additionally, the experiment focused solely on capturing disk data, while leaving aside live working memory and network traffic captures. This was done generally in order to reduce the scope of the study in that it was deemed too wide to attempt to capture all these forms of digital evidence.

Further to this the testing of the partial proof of concept LEIA system was limited to only a few devices, as was mentioned earlier. This was due to the limited availability of devices at the disposal of the author of this study. In addition to this, the devices that were used in the testing were limited only to Android based devices that had “root” capabilities, the ability to install an SSH server and some form of “netcat”. In essence all Linux based platforms with the necessary administrator permissions are compatible. Microsoft Windows based devices and Apple iOS devices were not focused on or tested, even though theoretically the LEIA proof of concept could be slightly modified to support capturing data from them.

The data sizes that were tested were also limited due to the capabilities of the devices as well as the sizes of the memory cards available. Extremely large amounts of data were also not tested due to the large amounts of time that would have been involved in the testing. Further work should be performed in order to achieve results on larger data acquisition sizes.

The experiment works on the assumption that all devices have administrative permissions readily available, an SSH server that has administrative privileges and some form of “netcat” available. This is often not case in a real world scenario, thus the need for a hypervisor layer that has these capabilities.
6. Conclusions & Future work

This section marks the final stretch of this study and is arguably the most imperative in its contribution to the work done as a whole. This is because it looks back at the work done reflecting on the extent to which the successes have been achieved, goals attained and shortcomings experienced and how they affected the outcomes of the study.

It is also important because it provides a platform for readers, and other eventual consumers of this work, to be exposed to further work that could be done, based on the ideas expressed herein, in order to make a more significant contribution to the wider body of knowledge.

6.1. Conclusions

With regard to the area of Computer Forensics and Incident Response, the research question posed at the onset of this study was:

“How can relevant information on criminal activity be efficiently detected and rapidly dealt with (collected) in a digital investigation, given the scarcity of resources available and the complexity of information interactions and data networks of today?”

To this effect the aim of this study was to design a comprehensive architecture that efficiently addresses the problem of scarcity of resources in light of the current data deluge and imminent complexity of interactions among the systems involved in digital crimes today and looking forward into the future. Being an arduous task, and in order to make it more manageable, the focus was narrowed down mainly to the “Evidence Acquisition” phase while also touching on the preceding (Detection of Malevolent Activity) and succeeding (Analysis of Acquired Data) stages of the Digital Investigation / Incident Response procedure.

From experience and exposure to vast amount of popular literature in the area of Digital Forensics, the major problems that investigators experience while undertaking digital investigations were outlined. A novel and innovative combination of a variety of technologies is brought together in order to provide an architecture of an automated solution. The motivation for the use of the said technologies is mainly driven by the correlation between the benefits provided by these solutions and the shortcomings identified in current Digital Investigation / Incident Response procedures.

The particular technologies that are used are:

- Crowdsourcing among Intrusion Detection Systems for more effective malevolent activity detection on large networks
- Virtualization and Hypervisors in order to provide a privileged layer that can easily and securely acquire evidence in forensically sound manner
- Hybrid Peer-to-Peer overlays making use of Gossiping, Hierarchical Overlays and the Bittorrent protocol to enable rapid transfer large amounts of captured evidentiary data
- The Resource Description Format from Semantic Web technologies in order to provide metadata that describes the evidence captured (and eventually relationships between evidence sources)
- Scalable cloud storage and processing capabilities in the form of a Hadoop cluster in order to reliably store large amounts of evidence data captured
The architecture and eventual interplay between these technologies are described in the form of a 4-Tiered system that is termed as the Live Evidence Information Aggregator (LEIA). The individual functionality of each tier, the communication protocols and the data formats are also extensively described.

In order to demonstrate that such an architecture could possibly be developed, a small proof of concept of the LEIA system, with limited functionality is developed and its performance tested. The functionality of the LEIA proof of concept was limited to remote acquisition of small scale device, disk-based evidence and its eventual storage on a cloud storage infrastructure. This was successfully performed and tested with 3 different small scale (mobile) devices. Various metrics relating to the time taken to perform the process as well as the evidence sizes (on the specimens as well as after compression and storage) were recorded and graphs drawn.

In conclusion, the proof of concept LEIA system was capable of performing the remote acquisition tasks that it was built to undertake. Though the functionality was constrained due to time and resource limitations, the results were promising. Additionally, the software development process resulted in a clearer insight into the capabilities that would be eventually needed within the actual LEIA Host-based Hypervisor in order for it to perform its designated tasks.

6.2. Concerns around the LEIA Architecture

In the eyes of the author, this study would not be complete without some mention and discussion of the practical concerns and consequences of such a system being implemented and deployed into the real world.

From the perspective of the manufacturers of gadgets and systems that we use today, it would certainly be an uphill task to convince them to ensure that they build the LEIA Host-based Hypervisor into their current offerings. However it should not be considered entirely impossible as it has been recently discovered by the author (while concluding this study) that the latest version of Microsoft’s flagship operating system, Windows 8 does indeed contain an inbuilt hypervisor, and thus the hardware interfaces are largely virtualized. Given this observation, and the fact that Microsoft Windows OS probably has the largest market share of computer systems today, it is not impossible to imagine that an effort from such large software platform (OS) vendors can result in an extremely wide spread of an inbuilt hypervisor.

On the other hand, though there would be inertia from manufacturers to build in such a hypervisor into their offerings, the opposite may be said of governments who run large scale surveillance programs in order to beef up security and intelligence. Such governments would likely be in favour of such an implementation and could possibly pressure large operating system vendors into implementing such a hypervisor into all gadgets. Consequently, the security and intelligence branches of these governments would also gladly implement the other tiers of the LEIA architecture in order to automate their Law Enforcement activities particularly for Digital Crime.

On the part of the consumers/ users of the devices and systems that make life easier today one of the greatest concerns is privacy. Such a system as the proposed LEIA system would in effect render null and void all forms of privacy (related to anything containing the LEIA HbH) in favour of the possible security and intelligence benefits that it could provide. That said, the current trend of the disregard of privacy especially among young people on social networks begs to ask whether there will be any regard for privacy in the future. There is a cost-benefit trade-off with regard to the loss of
privacy and the respective benefits that could be realized that is gradually being eroded in favour of the benefits, especially in light of the fact that most people today are elated by the benefits that technology can provide – encouraging individual self-gratifying laziness. Summarily, this indicates that possibly in the future a surveillance culture may indeed be favoured in exchange for the perceived benefits that the exposure of personal information may provide.

One of the hallmarks of this study is the ability to remotely acquire evidentiary data from small scale devices over their network connection. Though this is a great advance towards speeding up acquisition times, it comes at the cost of using the devices already limited battery power as well as the user’s network bandwidth. This may raise concerns particularly from the users in that their resources are being consumed without any immediately visible benefit.

This shortcoming, particularly in the use of the network connection, may be addressed through ensuring that the user knows the data that is being collected from the device is being used for 2 purposes that play greatly towards their benefit. That is, the data being collected can help their inbuilt Intrusion Detection System perform better and secondly, the data being collected can serve as a back-up for their personal data on their device in case of some event that renders the data on the device irretrievable.

In the case of battery power being used “unnecessarily” one solution may be to perform such battery intensive processes such as Wi-Fi network connectivity preferably when the device is plugged directly into a power plug. This does not mean that it is only done when power is plugged into the device, only that there is a preference towards this. All the same in the event of a malicious event occurring, data would still be transferred regardless of the power status (as long as the device is on).

6.3. Areas of Future work

Though the architecture of the LEIA system has been extensively described, it is likely that there are certain aspects that the author has not included, or taken into consideration. Furthermore, as has been already mentioned, the scope of the study has been largely narrowed down to the evidence acquisition phase, thus there is a lot of room for the development of procedures to automate the other phases of the Digital Investigation / Incident Response process. Of particular note would be the Evidence Analysis Phase and Presentation Phase.

With regard to the development and design of the LEIA architecture further work needs to be put into implementing the Host-based Hypervisor in order to realize the Intrusion Detection capabilities as well as the data and message exchange formats. On the same note, the implementation of the Peer-to-Peer Distribution Architecture is also considered as future work that can be performed towards attainment of the LEIA system as a whole.

In terms of the actual architecture of the LEIA system one could think of revolutionizing the centralized CBB system by distributing the actual storage and processing capabilities out towards the individual HbH systems. In order to realize this it would necessitate that the individual HbH systems be reasonably well endowed with processing and storage capabilities in order to cope with their own normally inherent tasks, as well as the extra task of being part of the LEIA system.
7. Bibliography


[118] Kulesh Shanmugasundaram, Nasir Memon, Anubhav Savant, and Herve Bronnimann, "ForNet: A Distributed


 Sciences - Stockholm University, Stockholm, Masters Thesis 2013.


