

Digital Forensic Model for a Cloud Environment

by

Mhlupheki George Sibiya

Submitted in fulfilment of the requirements
for the degree

Philosophiae Doctor

in the subject of

Computer Science

in the

Faculty of
Engineering, Built Environment and
Information Technology

at the

University of Pretoria

February 2015

Supervisor

Prof. H.S. Venter

© 2015 - *University of Pretoria*
ALL RIGHTS RESERVED.

Digital Forensic Model for a Cloud Environment

ABSTRACT

Cloud computing is a relatively new computing paradigm that builds upon virtualisation technologies to provide hardware, platforms and software as services over the Internet. The cloud can be deployed in four basic deployment models namely private cloud, community cloud, public cloud and hybrid cloud. Private cloud is owned and utilised by a single organisation and may be hosted internally and by a third party. The community clouds is meant for organisations with similar business interests, while the public cloud is accessible to the general public over the Internet. The hybrid cloud is a combination of any of the other cloud deployment models. All the cloud deployment models are characterised by multi-tenancy, namely data belonging to multiple users reside on the same physical host. Powering off a multi-tenant host would disrupt co-hosted services in a physical host which would then affect their availability. This affects other tenants that are not related to an incident. The cloud is distributed and often spans multiple jurisdictions. Its distributed nature also prevents conventional procedures for collecting evidence data and preservation. New approaches in conducting digital forensic investigations are required. In this thesis, different dimensions of digital forensic challenges brought by the advent of cloud computing are presented. The extent to which traditional digital forensic approaches address the issue of digital forensics in cloud environments are also presented. Digital forensic standards are considered important in this thesis as they are an aspect that can contribute positively to investigating cloud environments when multi-jurisdictional collaboration is required. Standards can also enhance acceptability of digital forensic evidence gathered from cloud environments. As a solution towards addressing issues of digital forensic investigation in cloud environments, in this thesis the author presents standard procedures that can be used to conduct a digital forensic investigation in cloud environments.

To enable execution of these procedures, a cloud forensic service model is presented that guides digital forensic investigators through a standardised collaborative process of investigating cloud environments. Both proposed digital forensic procedures and the service mentioned above were evaluated in a private cloud environment. Evaluation results have shown that a collaborative environment can be used to investigate cloud-based incident scenes in a standardised and cost efficient manner.

Contents

1	INTRODUCTION	1
1.1	Introduction and Background	2
1.2	Motivation	3
1.3	Statement of the Problem	4
1.4	Goal and Objectives	5
1.5	Methodology	6
1.6	Organisation of the Thesis	6
2	BACKGROUND	8
2.1	Introduction	9
2.2	Background on Cloud Computing	10
2.3	Cloud Infrastructure Attack Vectors	12
2.4	Unique Cloud Attributes	14
2.5	Background on Digital Forensics	16
2.6	Digital Forensic Challenges	19
2.7	Conclusion	21
3	RELATED WORK	22
3.1	Introduction	23
3.2	Digital Forensic Processes Review	24
3.3	Cloud Forensic System Design Requirements	30
3.4	Digital Forensic Architectures	34
3.5	Cost Minimisation Approaches	39
3.6	Conclusion	39
4	CLOUD FORENSICS AS A SERVICE	42
4.1	Introduction	43

4.2	Approaches in addressing the research questions	44
4.3	Cloud Forensic as a Service (CFaaS)	49
4.4	Scope and key assumptions of the study	51
4.5	Conclusion	53
5	CFAAS DIGITAL FORENSIC PROCESS DESIGN	54
5.1	Introduction	55
5.2	Cloud Forensic Process Model	56
5.3	Conclusion	69
6	CFAAS REMOTE HOSTS PRIORITISATION ALGORITHM	70
6.1	Introduction	71
6.2	Incident Scene Modelling	72
6.3	Algorithm	74
6.4	Conclusion	78
7	CFAAS SERVICE MODEL ARCHITECTURAL DESIGN	79
7.1	Introduction	80
7.2	CFaaS Service Model Architecture	81
7.3	How the architecture addresses the requirements	103
7.4	Conclusion	104
8	CFAAS PROCESS MODEL TECHNICAL IMPLEMENTATION	105
8.1	Introduction	106
8.2	JBoss jBPM	107
8.3	Processes Implementation	107
8.4	Conclusion	126
9	CFAAS SERVICE MODEL IMPLEMENTATION	128
9.1	Introduction	129
9.2	CFaaS Service Implementation	130
9.3	Conclusion	136
10	CFAAS EVALUATION AND TESTING	137
10.1	Introduction	138
10.2	Process Evaluation	139
10.3	Remote Hosts Prioritisation (SLOE) Algorithm Evaluation	145
10.4	Architectural Design Evaluation	165

10.5	Implementation Evaluation	169
10.6	Conclusion	174
11	CFAAS EVALUATION RESULTS DISCUSSION	176
11.1	Introduction	177
11.2	Process Evaluation Results Discussion	179
11.3	Remote Hosts Prioritisation (SLOE) Algorithm Evaluation Discussion	180
11.4	Architecture Evaluation Results Discussion	181
11.5	Experimental Evaluation Results Discussion	182
11.6	Conclusion	186
12	CONCLUSION	187
12.1	Introduction	188
12.2	Summary of research problem addressed	189
12.3	Research problem solution	190
12.4	Future Work	193
12.5	Conclusion	194
A	CHAPTER 3 TABLES	196
B	CFAAS DATA FLOW DIAGRAMS	200
B.1	Logical View	201
B.2	Development View	222
C	PROCESSES TASK COMPLETION FORMS	225
D	EVALUATIONS BACKGROUND THEORY	239
D.1	Algorithm Correctness Analysis Rules	240
D.2	Big-O Analysis Theory	240
D.3	ATAM Analysis tables	241
	REFERENCES	260

Listing of figures

1.1	Thesis Organisation	7
3.1	Standard Digital Forensic Process[61]	28
3.2	Digital Forensic Procedures Literature Analysis	30
3.3	Digital Forensic Service Literature Visualisation	40
4.1	Short figure name.	50
5.1	Cloud Forensic Process (Adapted from [61], see Figure 3.1)	58
5.2	Incident Detection	59
5.3	First Response	59
5.4	Planning Process	60
5.5	Preparation Process	60
5.6	Evidence Identification	61
5.7	Evidence Acquisition	62
5.8	Evidence Transportation	63
5.9	Evidence Storage	63
5.10	Evidence Analysis	65
5.11	Evidence Interpretation	68
6.1	Hosts Prioritisation algorithm	77
7.1	Context Diagram	82
7.2	CFaaS logical diagrams relationships	83
7.3	CFaaS data flow diagram: Diagram 0	85
7.4	CFaaS Data Flow Diagram: Child Diagram 8.5 - Determine Remote Hosts.	88
7.5	CFaaS Conceptual Model	90
7.6	CFaaS Conceptual Model: Platform	92

7.7	CFaaS Conceptual Model: Identity and Security Management	92
7.8	CFaaS Conceptual Model: CFaaS Task Server	95
7.9	CFaaS Conceptual Model: CFaaS Service	97
7.10	Scenario View: CFaaS Use Cases	99
7.11	Process View: Conduct Investigation Activity Diagram	101
8.1	Digital Forensic Process Implementation	108
8.2	Case registration	109
8.3	Incident description	110
8.4	Statistics Viewing	124
8.5	Case Validation	125
8.6	Reporting	125
8.7	Presentation	126
8.8	Investigation Closure	126
9.1	CFaaS service deployment environment	130
9.2	Remote files list request output	136
10.1	Presentation Sphere [12]	145
10.2	Termination function	156
10.3	Order of growth example functions	161
10.4	SLOE algorithm Time Complexity	164
10.5	Quality Attributes Tree	167
10.6	Welcome Page	170
10.7	New Investigation Initialisation	171
10.8	Case Registration Summary	172
10.9	Incident Description Summary	172
10.10	Establish Secure Connection Summary	173
10.11	Secure Connection Summary	173
10.12	Enable Secure Logging Summary	174
10.13	Forensic Team Organisation Summary	174
10.14	Investigation Report Snippet	175
B.1	CFaaS data flow diagram: Child Diagram 12-Analyse Evidence	202
B.2	CFaaS Data Flow Diagram: Diagram 4	205
B.3	CFaaS Data Flow Diagram: Diagram 5	206
B.4	CFaaS Data Flow Diagram: Diagram 6	208
B.5	CFaaS Data Flow Diagram: Diagram 7	209

B.6 CFaaS Data Flow Diagram: Diagram 8	211
B.7 CFaaS Data Flow Diagram: Diagram 9	213
B.8 CFaaS Data Flow Diagram: Diagram 10	215
B.9 CFaaS Data Flow Diagram: Diagram 11	216
B.10 CFaaS Data Flow Diagram: Diagram 13	218
B.11 CFaaS Data Flow Diagram: Diagram 14	219
B.12 CFaaS Data Flow Diagram: Diagram 15	220
B.13 CFaaS Data Flow Diagram: Diagram 16	221
B.14 CFaaS Component Diagram	223
C.1 Incident Scene Connection Establishment	226
C.2 Forensic Team Organisation	226
C.3 Network Types Identification	227
C.4 State of the art RAM and Hardware Identification	227
C.5 Evidence Sources	228
C.6 Task Assignments	228
C.7 Tools specifications	229
C.8 Tools acquisition	229
C.9 Corrupted data identification	230
C.10 Securing Online Transportation Link	230
C.11 Removable Drive Transportation	231
C.12 Integrity Verification	231
C.13 Online Storage	232
C.14 Removable Drive Storage	232
C.15 Loaded modules identification	232
C.16 Active processes listing	233
C.17 Hidden processes identification	233
C.18 Suspicious processes memory space analysis	234
C.19 Suspicious process files identification	234
C.20 Memory String Searching	235
C.21 Host Information Visualisation	235
C.22 Anti-forensic Rootkits Identification	236
C.23 Anti-forensic Rootkits Deactivation	236
C.24 Decoding Network Connections	237
C.25 Network Flow Viewing	237
C.26 Network Files Viewing	238

Dedication

This thesis is dedicated first to my daughter Silondiwe Silindokuhle Sibiya and my beautiful “happily ever after”, Nozipho Zamambo Mkhize-Sibiya. It is also dedicated to my parents who made sure we acquire education through the little that they have. It is dedicated to my brothers and sisters Thalitha, Amos, Moses, Piet (aka Bhantshi), Nkosinathi (aka Tsimba), Sindile (aka Last born), Thandi, Zanele and Lazaro (Maheza) and all my sisters-in-law (skwizas) and nieces.

Acknowledgments

I wish to acknowledge and extend my gratitudes to Prof. H.S. Venter for his contribution through his guidance throughout the research conducted for this thesis. I acknowledge and appreciate the technical support tendered by Thomas Fogwill and the entire Meraka Institute's Next Generation ICT & mobile systems research group. I would like to acknowledge and appreciate the CSIR and the NRF for the financial support during the course of the studies. I would also like to thank Dr. Adele Botha who assisted in shaping up my research methods and her motivation. I would also like to extend my Sincere gratitudes to my beautiful wife N.Z. Mkhize-Sibiya who did not become jealous of my second wife - the laptop - and for her support all the way.

Most of all I wish to extend my gratitude to God Almighty who made sure that myself and my family and I were in good health through out the entire journey. It is also through His will that my mind was always in a correct state to carry out the task. To Him be all the glory.

1

Introduction

1.1 INTRODUCTION AND BACKGROUND

The use of information technology has grown at a rapid rate. Recent surveys have shown that the number of Internet users had passed the two billion mark by the end of 2010, compared to just below 361 million users 10 years earlier [131]. This can be partly attributed to the capability of the more affordable mobile devices to access the Internet. Mobile devices can now perform almost all tasks that can be performed by their desktop counterparts. Computing paradigms are changing due to continuous improvements in technology. These developments have led to the advent of cloud computing [145]. A cloud infrastructure is virtual, distributed in nature and usually spans over multiple jurisdictions. In cloud environments hardware, platforms and software are managed by a third party and they are not in the full control of the owners, as has been the case in traditional settings. Service providers may lease hardware storage services from third parties to store their enterprise and client data.

Growth in the number of Internet users and the different types of devices used on the Internet raises computer security concerns [76] including in the Internet-based cloud environments. Cloud computing is gaining wider acceptance despite the security concerns that still prevail. When security breaches occur in cloud environments, digital forensic investigations need to be carried out. In the cloud, critical data is more vulnerable and at the same time, difficult to acquire when an incident that requires a digital forensic investigation arises. In multi-tenancy, cloud service providers may host data belonging to more than one cloud customer. Multi-tenancy therefore becomes an issue when data belonging to a cloud user needs to be isolated for investigation purposes [35]. It is for these reasons (among others) that the traditional forensic processes and tools cannot be directly applied in the cloud.

Due to the digital forensic investigation challenges that still exist, criminal cases that occur in the cloud are often abandoned. The challenges include the lack of digital forensic tools and standards that are suitable for cloud environments. Numerous researchers [52, 87, 137] have attempted to address these challenges but these efforts still leave a lot to be desired.

The remainder of Chapter 1 is organised as follows: In Section 1.2, the motivation of the study is stated. In section 1.3, the statement of the problem is presented. In Section 1.4 focuses on the goal and objectives of the study and in Section 1.5 the research methodology is presented and discussed. Section 1.6 is dedicated to the organisation of the thesis.

1.2 MOTIVATION

In virtual environments, service providers deploy their services in virtual infrastructures that run on physical infrastructures that are owned by third parties who offer these infrastructures as services. This provides opportunities for Small Medium and Micro Enterprises (SMMEs) who can therefore focus on their business without worrying about the ICT infrastructure required. Although such developments ease the costs burden on the side of SMMEs, they come at a price as they come with more security concerns. As service providers and clients in virtual environments lease ICT infrastructure from third parties, the data that they exchange while communicating is stored and transmitted through many intermediate parties online. Malicious activities on-line are on the rise, given the rise in the number of Internet users and the advent of social networks. Cyber-attacks are also constantly taking place as can be seen on live cyber-attack maps such as Norse [2]. The data is therefore vulnerable to attacks in many layers. Such vulnerable data could include crucial data such as passwords, bank account numbers and credit card numbers. Solutions are available that attempt to address security concerns in cloud environments. These security solutions however cannot completely mitigate security breaches as perpetrators are constantly working on ways to violate security measures, especially on the Internet.

Cyber-security concerns are not limited to individuals and/or companies. Recently there have been incidents where state based organisations were accused of being involved in compromising foreign computer systems. An example of such incidents occurred in November 2014 when the United States FBI accused North Korea of attacking Sony's computer networks [104]. If a state organisation gets involved in such attacks, this can possibly escalate into cyber warfare as a state under attack may also choose to retaliate. Another example of a national level of a cyber-attack is the Stuxnet [48], which was reported to have been targeting Iranian national nuclear facilities.

In all cases, be it national level or organisational level or individual level, when an incident occurs, digital forensic services would be required when the cases are taken courts or even to international criminal courts. This thesis provides a solution and addresses digital forensics with a specific focus on cloud computing since more and more critical data is being stored in the cloud – and hence attracts more cyber criminals.

1.3 STATEMENT OF THE PROBLEM

Cloud computing is characterised by distributed systems that span multiple jurisdictions, as well as multi-tenancy and virtualisation. These characteristics pose a challenge when a digital investigation needs to be carried out in a cloud environment. Due to the infancy of the cloud, digital forensics still lacks standards that can be directly applied when digital investigations are to be conducted in cloud environments. Digital forensic standards such as the ISO/IEC 27037 (Information technology – Security techniques – Guidelines for identification, collection, acquisition and preservation of digital evidence) are better suited for conventional and physically accessible computing setting. In providing a solution to digital forensic challenges in cloud environments, this research work addresses the following main research question:

On what framework can a cloud forensic solution be based for a cost-effective digital forensic investigation in the cloud?

The framework for the research is provided by addressing the following sub-questions:

1.3.1 WHAT ARE STANDARDISED PROCEDURES THAT CAN BE CARRIED OUT IN A CLOUD ENVIRONMENT WHILE CONDUCTING A DIGITAL FORENSIC INVESTIGATION?

Digital forensics lacks standardised procedures, particularly for cloud environments. The well-developed and accepted traditional digital forensic procedures cannot be applied directly in a cloud environment. Answering the above question assists in enabling successful convictions of cases that involve cloud environments and also supports collaboration among law enforcement agencies in multi-jurisdictional cases.

1.3.2 WHAT ARE THE REQUIREMENTS FOR A CLOUD FORENSIC SYSTEM?

The disk-imaging techniques used in traditional digital forensic procedures cannot be applied in a cloud environment as the environment is virtualised, data is fragmented and distributed, and often there is no physical access to the servers. Accessing and imaging evidence from such an environment can be costly and costs are the main contributors to the decision to abandon criminal cases. Optimising the digital forensic process in the cloud can reduce costs, hence leading to successful convictions and the conclusion of cases. The physical inaccessibility of the potential evidence location requires that an investigation be carried out remotely. It is evident that a system that can aid in this regard is needed and requirements therefore need to be investigated.

1.3.3 HOW CAN A CLOUD FORENSIC SYSTEM BE DESIGNED TO MEET THE REQUIREMENTS OF THE CLOUD?

Certain digital forensic techniques are applied in traditional computing settings. These techniques are however not directly applicable in cloud environments due to the latter's uniquely distributed and virtualised nature. Addressing the above research question helps to provide a framework that can be used to design a cloud forensic system for conducting an investigation - despite the challenges presented by the cloud.

1.3.4 HOW CAN AN INVESTIGATION IN A DISTRIBUTED CLOUD ENVIRONMENT BE OPTIMISED?

The distributed nature of the cloud means that potential evidence may not be localised in a single cloud-based instance, which may require an investigation process to be extended to remote hosts. This process may be challenging due to the potentially large number of remote hosts that may be connected to an incident scene and the fact that investigating all of them can be time consuming and costly. There is therefore a need for an optimised process to identify hosts for further investigation.

1.4 GOAL AND OBJECTIVES

The current research work aimed to formulate a framework for conducting a standardised and cost-effective digital forensic investigation in a cloud environment. The objectives of this study can be summarised as follows:

1. The first objective was to conduct a literature survey on existing frameworks that address the issue of digital forensic investigation in cloud environments and point out weaknesses in existing digital forensic investigation frameworks. This also involves obtaining a thorough understanding of the cloud architecture including standards and processes that can be used to conduct a digital forensics investigation.
2. The second objective was to propose a digital forensic framework for the cloud that is based on a standardised framework that takes the cost effectiveness into account on a digital forensic investigation.
3. Thirdly, to propose a digital forensic service model that is based on the digital forensic framework.

4. Lastly, to demonstrate the effectiveness of the framework in a practical environment by implementing a prototype based on the framework.

1.5 METHODOLOGY

This research adopts a design science research [54] and the research approach is descriptive, formulative and evaluative in nature. The descriptive part of the research approach involves a literature survey of the existing research efforts that address the issue of digital forensics in cloud environments. This survey focuses on digital forensic standards, digital forensics processes, procedures and techniques in cloud environments. An evaluative analysis of existing techniques is carried out, after which the results are used as a benchmark for the formulation and development of the researcher's solution approach in addressing the issue of digital forensics investigations in cloud environments. A mathematical conceptual analysis is carried out on the solution approach to formalise the formulated algorithms and methods. The final part of the research approach involves a proof of concept. As a proof of concept the prototype of the framework is tested on the intended execution environment.

1.6 ORGANISATION OF THE THESIS

The remainder of the thesis is organised as follows as depicted in Figure 1.1. In Chapter 2, some background details on cloud computing which cover the architecture of the cloud and computer security challenges related to the cloud in the form of attack vectors is presented. Seeing that security breaches may occur in the cloud, digital forensics is required. The chapter therefore also presents the challenges associated with conducting an investigation in the cloud.

In Chapter 2, the background of digital forensics is also discussed. Processes that are proposed by other researchers in conducting digital forensics, as well as challenges that are encountered while conducting a digital forensic investigation, are discussed.

In Chapter 3, related work on digital forensic processes and digital forensic architectures is presented. The chapter also covers a critical analysis of aspects addressed by each individual digital forensic process and individual architecture.

In Chapter 4 the researcher proposes a digital forensics framework that can be used in a cloud environment, and in Chapter 5 the design of the proposed digital forensic model that is based on the framework is discussed. The process model addresses the shortcomings of conventional digital forensic processes when an investigation is

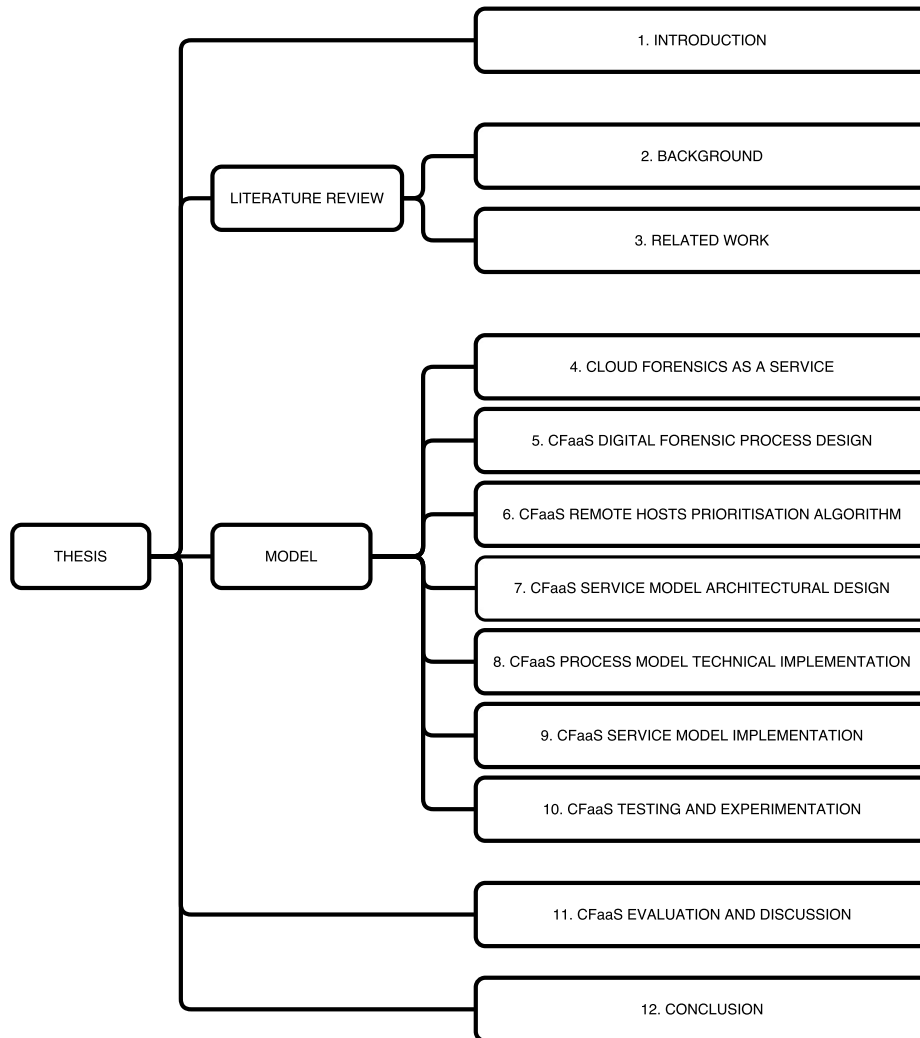


Figure 1.1: Thesis Organisation

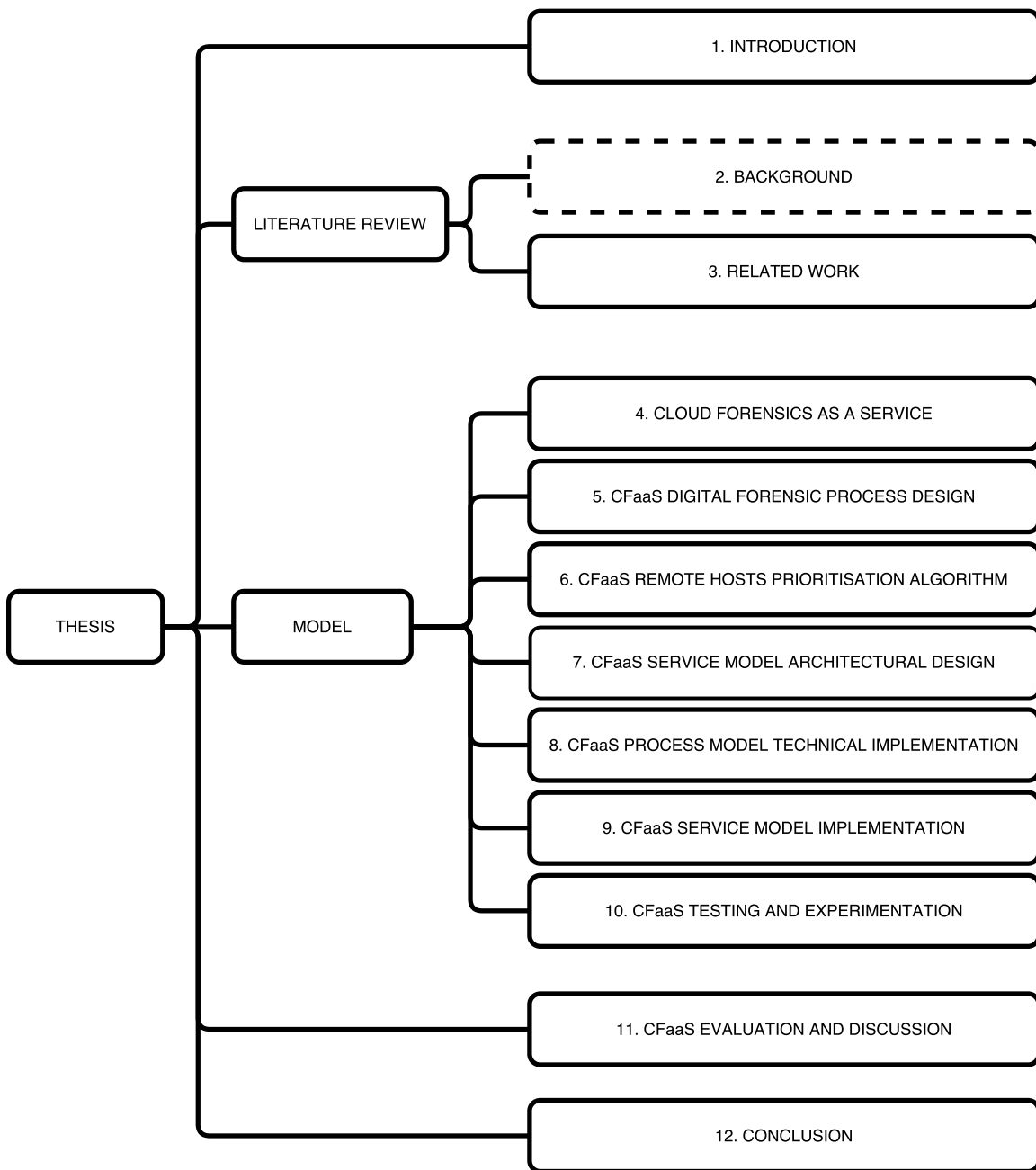
conducted in a cloud environment.

While performing an investigation at the incident scene, an investigator can be led to remote hosts with more evidence. Based on the connections leading from remote hosts to the incident scene, hosts can be selected for further investigation. In Chapter 6, an algorithm that can be used to prioritise remote hosts that are connected to the incident scene for further investigation is presented. Chapter 7 is devoted to the design of the proposed digital forensic service model that is used to execute the digital forensic process proposed in Chapter 5. As a proof of concept, in Chapters 8 and 9 the implementation as a prototype of the CFaaS process model and CFaaS service respectively are discussed.

In Chapter 10 the model and the results are evaluated, in Chapter 11 the results of this evaluation, and Chapter 12 concludes the thesis is discussed.

2

Background



2.1 INTRODUCTION

Continuous and ground-breaking developments in technology have led to the emergence of cloud computing. Unfortunately, cyber criminals have managed to keep up with the pace of technology developments, and even more advanced criminal attacks have occurred over time. Forensic investigators are left wanting when cybercrime is committed as the latter have also become increasingly advanced. Cloud computing

is one such technology that complicates digital forensic investigations. In the current chapter some background information on cloud computing, which covers the cloud architecture, cloud security and characteristics of the cloud that pose challenges to the investigation of forensics cases in cloud environments are presented.

In recent years, the incidence of cybercrime has increased hugely. Unfortunately the advent of technologies such as cloud computing is aggravating the situation as the same technology is used to commit even more complex kinds of cybercrime. In fact, technology nowadays can be used to commit extremely serious crimes such as human trafficking and drug trafficking. For this reason, perpetrators of cybercrime have to be brought to book. This is where the role of digital forensics comes into play. Due to the developments in technology, digital forensics as a research fields is itself facing challenges. In this chapter a background on cloud computing, digital forensics and the challenges faced by digital forensics in general and also while investigating cloud environments therefore presented.

The chapter is organised as follows: Section 2.2 contains a background on cloud computing and its architectures. Section 2.3 depicts the attack vectors which can be utilised by attackers to gain access and do malicious damage in the cloud. It is these damages that prompt a need for digital forensic investigation in cloud environments. Once attackers have compromised the cloud, investigations need to be carried out and Section 2.4 is dedicated to the unique attributes of the cloud that make investigating it an uphill battle.

Section 2.5 is devoted to a discussion of live forensics, network forensics and digital forensic readiness. Challenges in digital forensics are presented in Section 2.6, and Section 2.7 concludes the chapter.

2.2 BACKGROUND ON CLOUD COMPUTING

Cloud computing provides computing resources on a pay-per-use basis. It is based on five principles: on-demand self-service; broad network access; resource pooling; rapid elasticity; and measured service [98, p.49]. On-demand self-service means that a cloud user can create (for example) a virtual machine or virtual instance and pay for it for the duration of its use, after which the virtual machine or instance can be terminated if it is no longer needed. Such services are referred to as measured services, because users are billed per usage. Cloud computing can be defined as highly scalable computing resources provided as an external service via the Internet on a pay-as-you-go basis [84]. This means that service consumers in the cloud pay

for services as they use them. Cloud resources need to be accessible to customers irrespective of geographical location, hence the requirement for broad network access. Resource pooling refers to computational resources that are published in a cluster for consumption by customers on demand. When a hardware resource is no longer in use, it is made available to other users.

Resources in the cloud can be scaled up and down according to user needs. This process is referred to as rapid elasticity. In a cloud environment, cloud service providers (CSPs) offer infrastructure or hardware, a platform and software as services that can be accessed by consumers over the Internet [138]. The availability of such services eases the burden on vendors as they no longer need to own physical infrastructures (such as servers) for computational needs. A cloud model can offer a solution to resource-constrained SMMEs in developing countries.

The cloud architecture can be viewed as a pyramid consisting of the three CSP services, namely cloud application, cloud platform and cloud infrastructure from top to bottom. These respective layers are referred to as Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) [84].

Cloud users participate as consumers or as providers of these services in the cloud. The responsibilities of the service providers and service consumers differ as they move up the cloud stack. This is important for digital forensic investigators to understand, as it helps to limit the scope of an investigation in the case of an incident scene. An IaaS provider is responsible for the vulnerability patches and configurations management of the systems, networks, hosts and application owned by the service provider and helps to manage the infrastructure [70, 88]. A vulnerability is defined as a “is a mistake in software that can be directly used by a hacker to gain access to a system or network”[34]. The IaaS provider is therefore responsible for patching the vulnerabilities on any of the components that they manage.

PaaS providers are IaaS customers and thus also responsible for the management of vulnerabilities, patches and configurations of the virtual resources allocated to them by the IaaS providers. They are likewise responsible for the management of the platforms that they provide as services, such as operating systems, application servers, web servers, databases, etc. SaaS providers are responsible for managing the applications that they offer as services in the cloud.

The cloud can be deployed in four different forms, namely the private cloud, community cloud, public cloud and hybrid cloud [55]. A private cloud is deployed to be used within an organisation and the entire cloud infrastructure, including hardware resources, is owned by the organisation. A community cloud is a cloud shared by

organisations with a common business interest. A public cloud is a cloud deployed for the purpose of being used by both public and private organisations, regardless of their business interests. A hybrid cloud is a combination of any of the previous cloud models.

In the cloud, hardware, platforms and software that were traditionally installed in the vicinity of the user are now offered as services by a third party [26, 84]. These services include storage and processing hardware, software platforms such as Java Virtual Machines (JVM), and software platforms such as human resource management systems. A third party may be another company within the national borders or a company outside the national borders. In all of these scenarios, the effort that would be required to carry out a digital forensic investigation differs. The costs, for example, will differ when a need arises to collaborate with international law enforcement agencies versus when collaboration is not required. This is one of the challenges faced by digital forensic investigators in a cloud environment.

Virtual machines as one of the services hosted in the cloud can be used to commit cybercrime in the cloud in the same way that a criminal can use a physical desktop. In [109], the authors define cloud crime as *“any crime that involves cloud computing where the cloud can be the object, subject or tool of crimes”*. It is when such crimes are committed in the cloud that the services of a forensic expert will be required.

Encryption is a method used to address data security in the cloud and it is used widely by cloud consumers and cloud service providers to address confidentiality. However, encryption itself faces challenges as adversaries use the same computing power of the cloud to decrypt data. The other aspects of security, namely availability and integrity of the data in the cloud, still depend on the IaaS provider. In most cases the IaaS is hosted by a third party. For incidences requiring digital forensic investigation, collaboration with an IaaS provider would be required in most cases.

The next section presents cloud attack vectors that can be utilised by adversaries to compromise the cloud.

2.3 CLOUD INFRASTRUCTURE ATTACK VECTORS

Depending on whether a cloud service provider (CSP) offers IaaS, PaaS or SaaS, or whether a cloud service consumer (CSC) consumes IaaS, PaaS or SaaS; there are different attack vectors that can be exploited by attackers. Attack vectors are defined as *“a path or means by which a hacker (or cracker) can gain access to a computer or network server in order to deliver a payload or malicious outcome”*[112]. This section

explores a high-level view of the common attacks vectors to which cloud users can fall victim.

2.3.1 IAAS ATTACK VECTORS

In a cloud environment, IaaS providers make use of hypervisors to expose logical partitions of their hardware as services. Management and access to these services is through Application Program Interfaces (APIs), which expose web consoles or Secure Shell (SSH) consoles. SSH consoles however require more enhanced technical skill, which most cloud administrators and clients may not have. SSH consoles are vulnerable as pre-shared keys are subject to being stolen if an adversary were to gain access to the client machine. This is so because private key management — generation, storage and distribution — is also an open research issue[159]. The more user-friendly interface is the web console that is used in most cloud platforms. Web consoles therefore are vulnerable to most web application attacks such as SQL injections, cross-site scripting, distributed denial of service, broken authentication and session management[96]. IaaS comprises physical servers that run hypervisors on which client virtual machines and cloud management virtual machines. IaaS is therefore vulnerable to attacks that exploit virtualisation technologies such as through memory leaks and CPU caches as proved by researchers in [158].

In the next section, PaaS attack vectors are presented.

2.3.2 PAAS ATTACK VECTORS

Platform as a Service embraces computing software environments that enable service consumers to deploy and run their applications on the Internet. The platform as a service can be a virtual machine instance running selected applications such as database servers and compilers.

To sign in so as to deploy applications in these virtual machines, clients use SSH tokens with pre-shared keys as one method to connect to the remote cloud service. Web consoles are also used for the purpose. SSH becomes an attack vector as hackers may steal these tokens and insert malicious code into the deployed application or the deployed image instance. If software platforms running on the remote virtual machine are not properly updated, they can be vulnerable to attacks. If for example, the platform consumed by the client runs a publicly available application server, it may have published vulnerabilities of which attackers can take advantage.

In the next section, attack vectors on IaaS vectors are presented.

2.3.3 SAAS ATTACK VECTORS

Software as a Service in the cloud is an extension of Web applications. Web applications have known threats as published and updated by the Open Web Application Security Project (OWASP) [96] and other authoritative sources such as Common Vulnerabilities and Exposures (CVE)[34]. These threats take advantage of the vulnerabilities in Web applications. The published threats include among others, SQL injection, cross-site request forgery and cross-site scripting. As an extension of Web applications, SaaS is vulnerable to comprise from these attacks.

In the next section, unique attributes of the cloud that lead to difficulty in investigating it are presented.

2.4 UNIQUE CLOUD ATTRIBUTES

This section presents unique cloud attributes that contribute towards the difficulty in conducting digital forensics in cloud environments. The attributes discussed here include the distributed nature of the cloud, encrypted data, multi-tenancy, fragmented data and volatile data.

2.4.1 DISTRIBUTED

Physical resources of the cloud are often distributed geographically. For reasons of high data availability, IaaS providers have their infrastructure data centres located in different regions. Having data centres in different regions ensures that if a single data centre should be subjected to natural or unnatural disasters, services would still be available through the unaffected datacentre. The distributed nature means that when an investigation has to be conducted, law enforcement agencies from the jurisdictions concerned will have to be involved. Such collaborations would extend the investigation time and costs. In addition to the administrative issues of time and costs, the involvement of multiple jurisdictions also introduces legal challenges. Some countries such as the European Union countries have strict policies on migration of electronic data to countries that are perceived to have inadequate data protection measures [14]. A cloud based investigation may therefore be hindered if it involves these European Union jurisdictions.

For example, Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 [43] prohibits transfer of data from the European Union (EU) member countries to non-EU member countries that are perceived to have inadequate

level of data protection. If a digital forensic investigation is carried out by a country that is perceived to have inadequate level of data protection, this would be a challenge for investigators if the data involved in the investigation is hosted in the EU zone if cloud data that can be obtained through the cloud user's client device is insufficient. Examples of such data are audit logs that may only be accessible to a cloud service provider. In such a case a digital forensic investigation would be hindered.

2.4.2 ENCRYPTION

Encryption is the main security measure that is implemented by cloud vendors to address security issues[15, 101, 113, 143]. There are a number of encryption layers in the cloud and the basic ones are used by the cloud service client and the cloud service provider. At the first layer, a cloud service client encrypts its data before deploying it in the cloud. At the second layer a cloud service provider further encrypts the client's data before storing it. For this reason, even if investigators would be able to obtain evidence data, it would still be difficult to process such data in its encrypted state unless the warrant asked for the decryption of the same data.

2.4.3 MULTI-TENANCY

On the cloud service provider's side, data from multiple clients are co-hosted on a single physical storage server. In conventional digital forensic processes where an incident scene host is required to be powered off (such as [94]), this becomes a challenge. Powering off the host would disrupt services belonging to tenants that are not concerned with the investigation. Isolating tenant data in such a scenario is also a research issue brought to light by Decker in [35].

2.4.4 FRAGMENTED DATA

In cloud environments, hardware resources are utilised according to availability. If data belonging to a cloud service is stored on a specific storage server, and the server runs out of space, the remaining data are stored on an additional server that has available space. In this way data obviously becomes fragmented. Data fragmentation can however also be implemented as a security measure or for easier processing [13, 42]. When only fragments of the data are obtained by investigators during an investigation, no sense can be made of the obtained data and thus this is a challenge to investigators.

2.4.5 VOLATILE

The basic principles of the cloud involve self-service and service on demand [1]. This means that cloud service consumers create data or service instances in the cloud as and whenever they need to utilise the services. When the service instances or data is no longer needed, cloud service consumers delete the service instance or data from the cloud. This gives investigators limited time to conduct their investigations as by the time they have obtained authorisation to acquire data from the cloud, the perpetrator may have deleted it already.

In the next section, digital forensic background is presented.

2.5 BACKGROUND ON DIGITAL FORENSICS

Digital forensics can be defined as a discipline that combines elements of law and computer science to collect and analyse data from computer systems, networks, wireless communications, and storage devices in a way that makes this data admissible as evidence in a court of law [103]. According to Zimmerman and Glavach [160], a digital forensic process can be divided into four distinct phases:

- Collection of artefacts (both digital evidence and supporting material) that are considered of potential value
- Preservation of original artefacts in a way that is reliable, complete, accurate and verifiable
- Filtering analysis of artefacts for the removal or inclusion of items that are considered of value
- Presentation phase in which evidence is presented to support the investigation

This is a basic process that is facing compounded challenges due to the advent of new technologies. More detailed digital forensic processes and their evaluation are presented in Chapter 3.

This thesis's ultimate goal is to study digital forensics in a cloud computing environment. Investigating cloud environments requires a focus on live forensics and network forensics which are dealt with in the subsections that follow. Digital forensic readiness is one important aspect of digital forensics.

2.5.1 LIVE FORENSICS

Traditionally, two categories of digital forensics existed, namely static digital forensics and live forensics. In [160] the authors argue that the two categories existed as a result of forensic evolution to recreate and document sophisticated incidences. Static forensics, also referred to as “dead forensics” or “non-live forensics”, involves the analysis of static data such as hard drives obtained by using conventional formalised acquisition procedures. Live forensics on the other hand involves the analysis of the system memory and any other relevant data while the system being analysed is running. As opposed to “dead forensics”, live forensics does not involve powering of the host at the incident scene. Evidence is collected or analysed from a live system. This type of digital forensics is ideal for cloud environments due to the multi-tenancy property of the cloud. Powering off a physical or virtual host in a cloud environment would disrupt not only the perpetrator, but also the other client whose services are co-hosted on the physical or virtual server.

Live forensics has advantages, since volatile digital evidence from locations such as Random Access Memory (RAM), which often gets lost when the host is powered off, can be analysed. While investigating the RAM, an investigator can choose to capture the RAM for off-line analysis or analyse it while it is still attached to the running host. Evidence that can be obtained during live forensics for the host includes, among others, running processes, open ports, open files, running script files and logged-on users.

2.5.2 NETWORK FORENSICS

As cloud computing architectures are networked in nature, it is worthwhile to discuss digital forensics with a specific focus on the network. This thesis contributes by providing a cloud forensic solution in the form of digital forensic procedures for the cloud and a framework for conducting the cloud investigation using such procedures. Network-specific forensic procedures form part of the procedures that need to be carried out in a cloud environment and such procedures were published by the researcher in [122] and [119].

A number of different network types exist, but they all originate from two basic ones: Local Area Networks (LANs) and Wide Area Networks (WANs) [47]. These networks can also be deployed as wireless networks (e.g. WLAN) and wired networks (e.g. Ethernet). In the context of cloud forensics, the network layer is a fertile ground from which digital evidence can be collected as all communications with cloud services

occur via a local network or a wide area network (e.g. the Internet). Digital evidence or data that can be obtained from the network may include the source and destination address of any communication, as well as the data (Internet Protocol (IP) packets) that is transmitted during the communication session itself.

There are various key locations in the network from which such data can be captured and stored. These locations include network routers, firewalls and even workstations or network-accessing devices. Information collected from such locations can be used for subsequent digital forensic purposes, if a case should arise.

Traditional computer forensics generally involves data acquisition from a storage medium such as a hard drive, while network forensics encompasses the capture, recording and analysis of network traffic that can be used for digital forensics purposes [47]. Networks in the cloud are often made up of virtual devices that may be added and removed dynamically in the virtual network. An important contribution of this thesis involves the digital forensic procedures that are applicable in cloud environments. This will be covered in details in the latter chapters.

In the next section, digital forensic readiness is covered.

2.5.3 DIGITAL FORENSIC READINESS

Digital forensic readiness is an approach that is used to minimise the effort required and (hence) minimise costs when an investigation has to be carried out [93, 105]. Digital forensics readiness makes data that may be used as evidence readily available throughout the lifetime of a live system or ICT infrastructure. This approach reduces the effort needed to conduct the investigation, as evidence is readily available. The investigation can move quickly to the advanced phases of an investigation process. Since the human resources required during an investigation are minimised, this also reduces the costs. Various scholars such as Dykstra and Sherman in [40] have proposed technologies and techniques for digital forensic readiness, which includes those meant for cloud environments. The technique by Dykstra and Sherman involves extending a cloud platform software, in this case OpenStack, which does not have digital forensic capability mechanisms by default. In this way, an investigator is able to access potential evidence instance data in real time whenever the need arises. Other techniques, such as that by Sang in [110], involve running agents on the client side and the server side of a cloud service. The agents ensure that consistent logs are kept securely on both sides of the service. Although there are benefits in incorporating digital forensic readiness into an infrastructure, such readiness does not completely solve problems in digital forensics as not all infrastructures will do so. However, a

cost-effective investigation still needs to be conducted when an environment that is without forensic readiness mechanisms is compromised.

In the next section, digital forensic challenges are presented.

2.6 DIGITAL FORENSIC CHALLENGES

In this section, challenges associated with digital forensics are at issue. Since the focus of this thesis is on conducting a digital forensic investigation in cloud environments, the focus of the discussion will be on challenges related to cloud computing. Identification of evidence is of key importance to an investigation since collection cannot take place unless evidence has been identified. If evidence cannot be collected, it simply means that the hypothesis cannot be validated in a court of law or at a hearing. This is the case with new technologies such as the cloud where data is often geographically distributed and physical access is limited.

These challenges lead to difficulty in proving in a hearing or during court proceedings the forensic validity of evidence collected from such an environment. Research efforts that seek to standardise digital forensic processes and procedures are on course, but more work still needs to be done. The constant developments in technology oblige digital forensics to develop at the same pace (if not faster). Digital forensics is a relatively new field of research in computer science and therefore still poses many research challenges. One of the challenges in digital forensics is the lack of standardised forensic processes and procedures [20]. Standardised digital forensic procedures would go a long way in addressing challenges associated with conducting an investigation into unconventional computing platforms such as the cloud.

Should digital evidence be altered by any investigator actions, the evidence may well be rendered unusable [153]. When an investigation has to be conducted on live systems such as in live forensics, alteration of the evidence can unfortunately not be avoided. If however documented procedures and processes were available with known commands that may be executed on live incident scenes (with documented effects), evidence collected from such scenes would still be usable. This is an example of a role that can be played by digital forensic standards.

The advent of cloud computing has brought in a new set of challenges in digital forensics. One aspect of such challenges is the legal aspect. In cloud environments, digital forensics is a different ball game as evidence is not always physically accessible and hence can no longer be an acquisition for dead forensic analysis. Cloud environments often span multiple jurisdictions and therefore require collaboration from in-

ternational agencies in an investigation. Obtaining multi-jurisdictional collaboration is, however, difficult and may take longer and delay prosecutions. During the process of seeking international search warrants and collaboration, vital evidence in the cloud may also be disappearing due to its volatile and dynamic nature. Comprehensive legal challenges, particularly with regard to the United States' legal frameworks, are discussed by Dykstra in [39].

The multi-tenant nature of the cloud prevents the applicability of the conventional “dead forensic” procedures as hosts in the cloud cannot be powered off or captured in whole. The researcher shares the sentiment that a digital forensic investigation in a cloud environment needs to be live. That is, evidence acquisition or evidence analysis needs to be carried out on live hosts without powering them off[108]. Fortunately, live evidence cannot be obtained from a live host only, but also from the network to which the host is attached. However, while investigating the cloud, live forensics can be carried out in combination with dead forensics. e.g. If evidence is copied from the live cloud instance and analysed outside of the instance, it is then dead forensics.

In Section 2.3, cloud attack vectors were presented with respect to the three service models of the cloud. When an attacker has utilised any of the attack vectors, a forensic investigation may need to be carried out. The attack vectors provide leads on where evidence can be gathered during an investigation given an incident. Some traditional digital forensic investigation tools can still be used to obtain evidence from these leads cloud. However, currently standards for use in investigations conducted in cloud environments are lacking. There are endeavours to provide such standards such as by Birk et. al. in [21] and Maras in [86]. The investigation process conducted by the Federal Bureau of Investigation (FBI) in the United States of America v. Ulbricht case [64] also paves a way towards realisation of standards that can be utilised in investigating the cloud. Research works on standardisation of cloud forensic processes still have shortcomings and these are discussed in Chapter 3.

A standard process has to be followed as cloud data is often distributed and multi-jurisdictional. For law enforcement agencies from the different countries to collaborate, standardised processes are required, since using standards can increase the chance of successful convictions. By following internationally agreed upon forensic investigation procedures, evidence collected in this way can be acceptable to both the defence and the investigators in a hearing, i.e. there would be no dispute about the digital forensic process followed. This can then lead to a successful conviction. In the draft report by the United Nations on the Comprehensive Study on Cybercrime in 2013, it was enforced by countries that “*Close working relationships on the pros-*

ecution team between the prosecutor and investigator that result in collection of all relevant properly authenticated evidence are essential to success in prosecution”[85, p.169]. In this thesis the researcher argues that the close working relationship can be enhanced through the use of standards in the investigation.

Other factors that pose a challenge to digital forensics in a cloud environment include privacy and security measures in the cloud. The extent to which data is protected and the privacy policies are enforced, directly affects the procedures and techniques that can be applied in investigating the cloud. Encrypted data for example requires specialised tools for decryption before any analysis can take place. Some jurisdictions, such as countries that subscribe to the laws of the European Union, are very strict on privacy and getting data from such countries for investigation purposes may be a challenging task. An example is if a user’s cloud based data is distributed among two jurisdictions where one is less strict on privacy and the other is stricter. Investigators would be likely to obtain incomplete evidence . Specific procedures need to be adhered to in a multi-jurisdictional investigation that involves such countries.

2.7 CONCLUSION

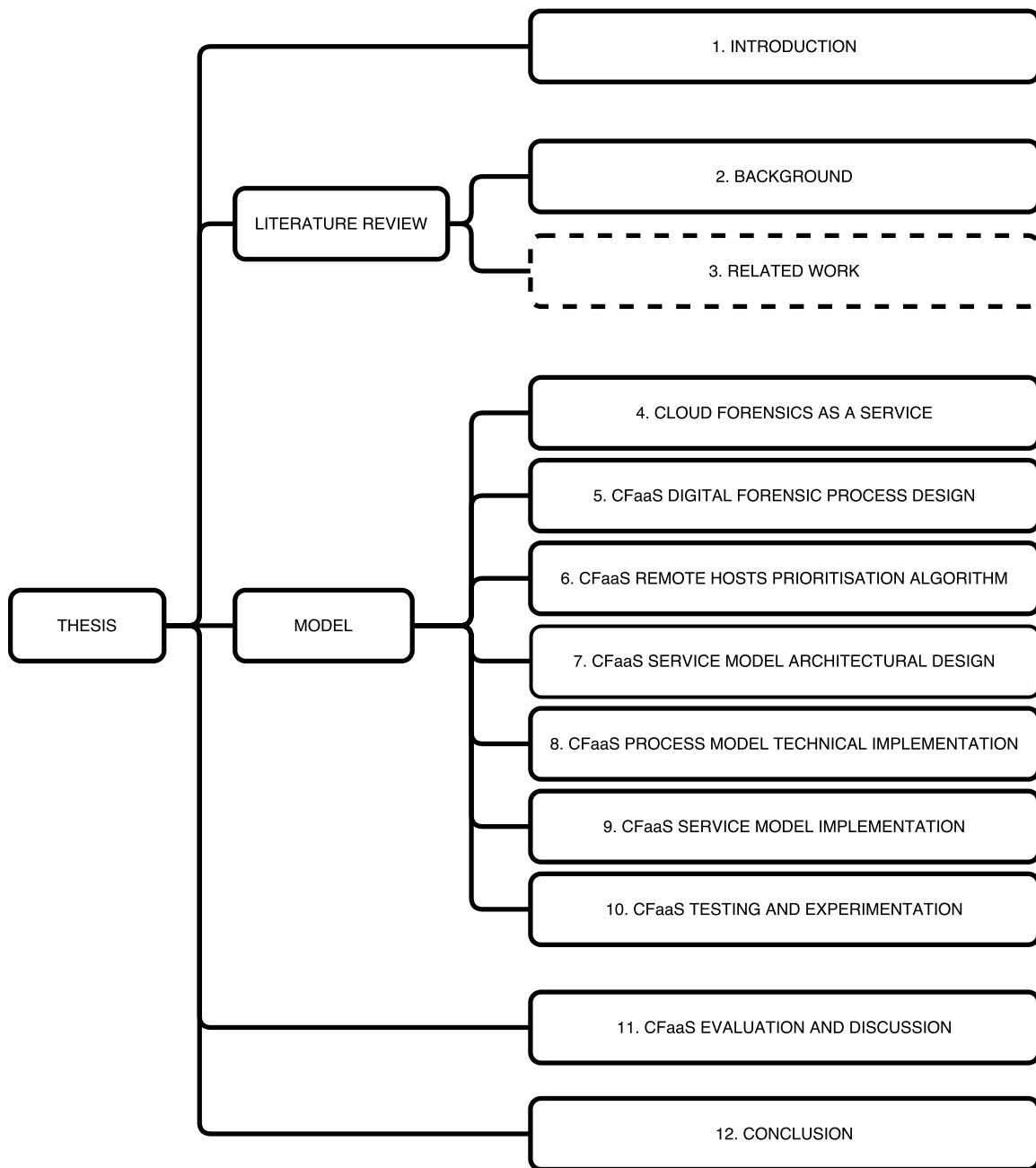
Cloud computing is a new technology that is posing serious challenges to digital forensic investigators. In this chapter, a background on cloud computing was presented. Attack vectors that may be exploited by cyber criminals in the cloud were presented, as well as attempts by other researchers to secure the cloud. Since the cloud is not fool proof against compromise by cyber criminals, digital forensics may always be required to be carried out in the cloud.

This thesis is aimed at addressing issues encountered when conduction digital forensic investigations in cloud environments. The issue emanates from the virtualised and distributed nature of the cloud and the immaturity of digital forensics as a research field. This chapter presented some background on digital forensic investigation. Including challenges faced by digital forensic investigators.

In Chapter 3 that follows, related research endeavours on digital forensic processes and digital forensic systems are at issue.

3

Related Work



3.1 INTRODUCTION

Endeavours by other researchers to address digital forensic challenges prevail both in the form of digital forensic processes and in the form of digital forensic system architectures. In this chapter, a review of these endeavours is presented. The chapter is organised as follows: Section 3.2 contains a review on existing digital forensics processes. Section 3.3, deals with the criteria that can be used to evaluate digital

forensic systems. Section 3.4 is dedicated to a discussion on the existing architectures and their evaluation. As costs may accumulate while conducting an investigation into cloud environments, Section 3.5 discusses cost minimisation techniques proposed by other researchers. Section 3.6 concludes the chapter.

3.2 DIGITAL FORENSIC PROCESSES REVIEW

This section summarises digital forensic processes proposed by other researchers, after which a comparative analysis of the summarised processes is provided. The research works are summarised according to the steps/procedures/phases/stages proposed in each. Steps, phases, processes and stages are different terms used by researchers to refer to a state or activity or task within digital forensic processes. The digital forensic processes evaluation presented in this section has also been published by the researcher in [125].

Digital forensics is an area in the computer security field that focuses on performing forensic investigations in electronic environments for trial or troubleshooting purposes. Advanced progress has been made with research on investigating digital environments and digital media. Traditional forensic processes such as the research by Casey, Katz and Lewthwaite in [29] generally comprise potential evidence identification, evidence acquisition, evidence analysis and presentation. This section presents other related processes and an analytical comparison of them all.

In their investigation of wireless environments, Lin, Yen and Chan[83] use a process that involves preparation, operation and report. The operation phase is further broken down into collection, analysis and forensics, which are either actions performed on the crime scene or evidence found in the investigation laboratory.

Leigland in [80] formalises the digital forensic process. The formalisation framework presented by Leigland allows portability of digital forensic procedures to a given incident, such as a type of operating system and all types of attacks. However, the presented framework only addresses the analysis process in a digital forensic process. The digital forensic processes are summarised by listing their respective phases.

Firstly, the guide for first responders by Mukasey, Sedgwick and Hagy in [94] that addresses guidelines for first responders in a digital forensic investigation is presented. The guidelines cover securing and evaluating the scene, documenting the scene, evidence collection, packaging, transportation and storage of digital evidence. The guide also provides potential sources of evidence, based on the category of crime.

Shin in [116] proposes a digital forensic process that comprises of phases which

they name: investigation preparation, classification of cybercrime and deciding about investigation priority, investigating a damaged digital crime scene, criminal profiling and analysis, tracking suspects, investigating injurer digital crime scene (perpetrator's workstation), summoning suspect(s), preparing profiling and writing a report.

Valjarevic in [147] presents an aggregation of published digital forensic processes that resulted in incident detection, first response, planning, preparation, incident scene documentation, potential evidence identification, potential evidence collection, potential evidence transportation, potential evidence storage, potential evidence analysis, presentation and conclusion. Concurrent phases that run throughout the investigation process in [147] include obtaining authorisation, documentation, information flow, preserving the chain of evidence, preserving evidence and interaction with physical investigation.

Bulbul in [23] presents a digital forensic process model that encompasses managerial tasks, crime scene examination, system assurance, evidence search, evidence acquisition, hypothesis and validation, organisation of evidence, physical management of evidence and system, and service restoration. Managerial tasks involve checking with the legal authorities concerned, obtaining digital forensic tools and securing the crime scene. Crime scene examination involves physical scene investigation, surveys and interviews, as well as digital crime scene investigation. System assurance involves the installation of activity-monitoring agents, system preservation and crime scene communication shielding.

Evidence acquisition involves the collection of evidence, acquisition of evidence, duplicating and copying of evidence, evaluation of data integrity and data analysis.

Baryamureeba and Tushabe in [18] propose a process that groups digital forensic investigation phases into five types of phase groups: readiness, deployment, trace-back, dynamic and review. The readiness phase group encompasses an operational readiness phase and an infrastructure readiness phase which respectively have to do with human capacity and selecting equipment. The deployment phase group involves detection and notification, physical crime scene investigation, digital crime scene investigation, as well as confirmation and submission phases. The confirmation refers to the validity of the incident, thus warranting further investigation, while the submission phase refers to the presentation of the evidence (according to Baryamureeba and Tushabe).

Quick and Choo in [100] present a digital forensic process model that is tailored for cloud based storage systems. The process model comprises of commence (scope), preparation, evidence identification and preservation, collection, examination and

analysis, presentation and complete. The trace-back phase group covers digital crime scene investigation and authorisation. These phases attempt to track down a perpetrator by investigating the crime scene and authorisation to obtain evidence from a perpetrator's local legal authority.

The dynamic phase group involves physical crime scene investigation, digital crime scene investigation, reconstruction and communication phases. Unlike submission, the communication phase is the presentation of the case in a court or hearing.

Watson and Jones in [151] cover wide aspects of digital forensics that include incident response, case processing, case management, evidence presentation, digital forensic laboratory IT infrastructure among others. For the purpose of this chapter focuses on four aspects, which are: incident response; case processing; case management and evidence presentation. These are the aspects that have procedures for initialisation of an investigation (case opening) through to investigation closure.

The incident response process covers the receipt of requests for first response presence, creation of a new case, advising the forensic laboratory manager of the request, gathering of information by the first response team leader, securing of the scene, identification of required evidence, determination of jurisdiction, preservation of evidence, documenting of the scene, serving the correct documents on the suspect, recovery of evidence to forensic lab, completion of the initial part of the evidence seizure summary, receipt of instructions and finally, agreement on the reporting points, communication plan and escalation procedures (see Figure 8.1 in Watson and Jones [151]). Case processing involves identification (recognition that an incident has taken place), preparation of equipment and search warrants, defining of an approach strategy, evidence preservation, evidence collection, evidence examination, validation of image's correctness by hashing evaluation, evidence analysis, evaluation, presentation of evidence, and the return of seized items (see Figure 9.1 in Watson and Jones [151]).

In case management, Watson and Jones not necessarily present procedures that need to be followed in case management. They present a MARS (management and reporting tool) that is used throughout a lifetime of a case or multiple cases. Watson and Jones [151] also addresses the presentation phase of a digital forensic process where they cover rules of evidence to be presented, report formats and report quality through to testimonies in court.

Kent in [68] mentions four phases - collection, examination, analysis and reporting. In addition to these phases, there is an action that runs throughout the investigation process, namely the preservation and documentation of the evidence.

Sindhu and Meshram [127] use a digital forensic process that proceeds as follows:

assess the crime scene, create a forensic image, calculate hash, hand over image to investigator, check integrity of the forensic image, start investigation, recover deleted files, search for hidden evidence, analyse the evidence, create a chain of custody and prepare a report. As a step towards addressing the lack of a standardised digital forensic process, the ISO/IEC27043 [61] standard presents a harmonised digital forensic process model as shown in Figure 3.1. Throughout the harmonised digital forensic process, accompanying parallel processes are being carried out. These processes are the following: obtaining authorisation, documentation, defining an information flow, preserving the chain of evidence, preserving evidence, and interaction with the physical investigation.

Three of these processes — obtaining authorisation, documentation, and managing information flow — are carried out during the course of the entire harmonised digital forensic process, while preservation of the chain of evidence, preservation of evidence and interaction with the physical investigation only start after the incident has been detected. The parallel actions performed are discussed in detail in [61]. The incident documentation phase depends on whether investigators have physical access to the incident scene or not. In a virtual environment such as the cloud, a crime scene may not be physically accessible; hence documented information may differ. Instead of photographs associated with physically accessible incident scenes, screen shots of the incident scene may be the only available documentation.

The other sub-processes are grouped into readiness process, initialisation process, acquisition process, and investigative process. For the purpose of this thesis the researcher considers the initialisation process, acquisition process and investigative process groups. These three groups of processes are chosen as they are compulsory, regardless of whether the digital forensic readiness mechanisms are in place or not. The harmonised digital forensic process that exist in these groups are - in their sequence - incident detection, first response, planning, preparation, potential digital evidence collection, potential digital evidence acquisition, potential digital evidence transportation, potential digital evidence storage, potential digital evidence acquisition, digital evidence examination and analysis, digital evidence interpretation, reporting, presentation and investigation closure.

It is worth noting that other literature and standards on digital forensic processes are not considered in the analysis provided here, as they were considered in constructing the standards and literature by other researchers that are included in the analysis. Related standards that are not included in the analysis include the following among others:

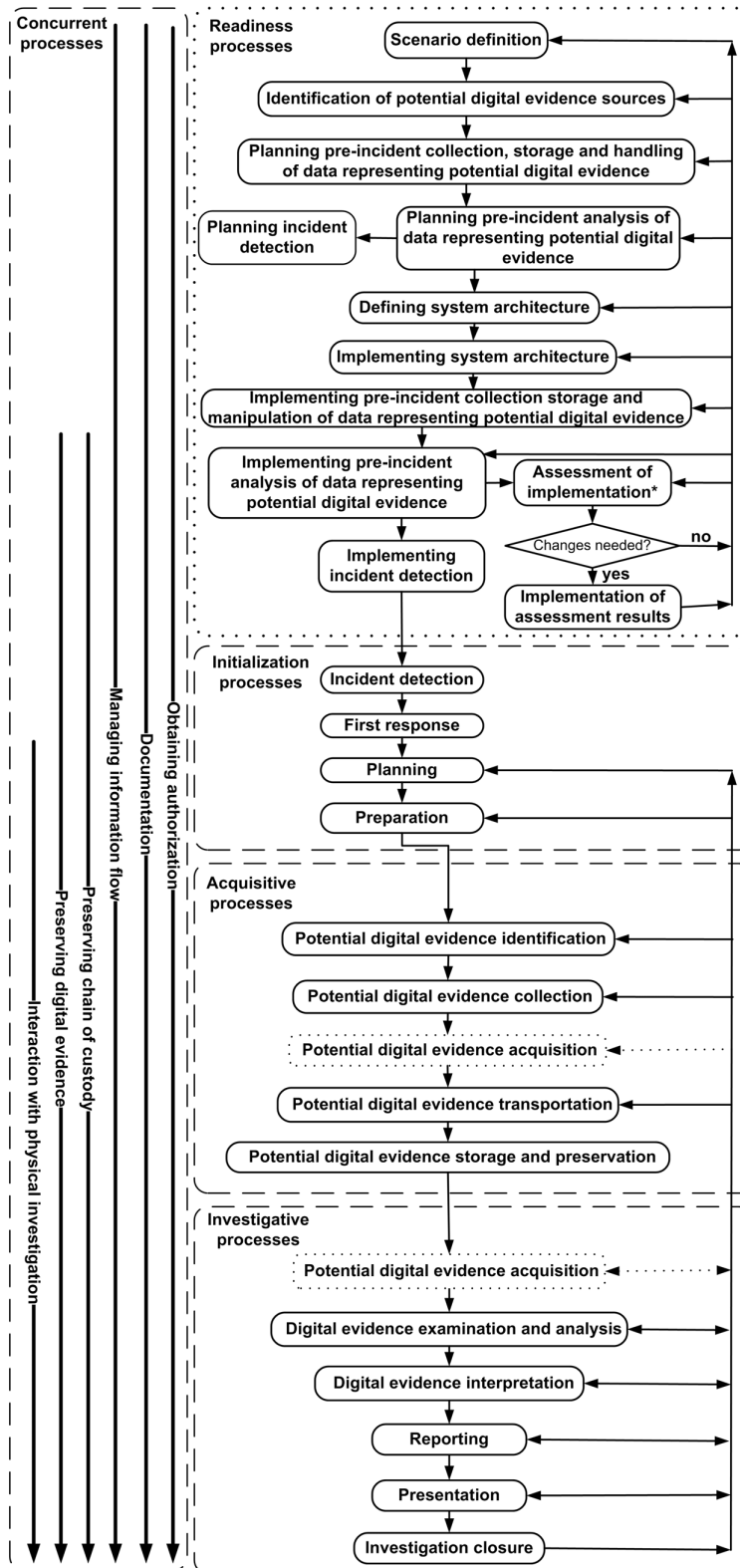


Figure 3.1: Standard Digital Forensic Process[61]

- ISO/IEC27037 - Information technology – Security techniques – Guidelines for identification, collection, acquisition and preservation of digital evidence,
- ISO/IEC27035 - Information technology – Security techniques – Information security incident management,
- ISO/IEC27040 - Information technology – Security techniques – Storage security,
- ISO/IEC27041 - Information technology – Security techniques – Guidance on assuring suitability and adequacy of incident investigative method,
- ISO/IEC27042 - Information technology – Security techniques – Guidelines for the analysis and interpretation of digital evidence,
- ISO/IEC27044 - Guidelines for Security Information and Event Management (SIEM),
- ISO/IEC27050 - Information technology – Security techniques – Electronic discovery
- ISO/IEC30121 - Information technology – Governance of digital forensic risk framework

All these standards are published or in a process of being published by the International Organisation of Standards (ISO) [58]. These standards were considered while constructing the processes in ISO/IEC27043 and also in Bulbul [23].

The processes are discussed in detail in [61], as well as in Chapter 5, where the proposed digital forensics process model is presented.

The digital forensic processes by other researchers presented in this section are analysed and visualised through Figure 3.2 in the form of a bar chart. Analysis of the 13 processes presented in this section is based on 81 phrases used by researchers in the 13 articles to name investigation processes/phases/stages/stages. The phrases that are used can be seen in Table A.1 (see Appendix A).

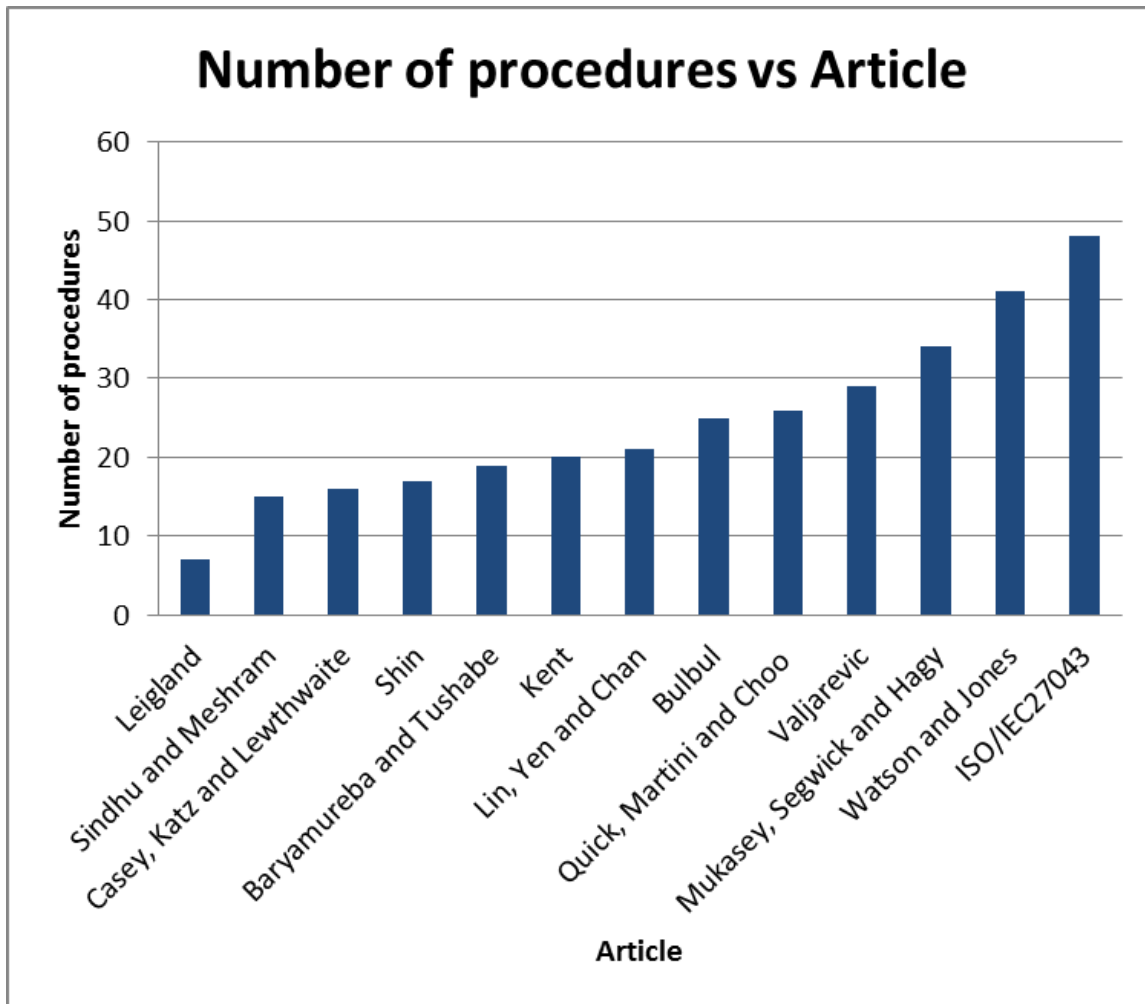


Figure 3.2: Digital Forensic Procedures Literature Analysis

The analysis above is based on the number of phrases each article in Figure 3.2 covers in the digital forensic processes that it presents. There is a disparity in the level of granularity of the digital forensic processes and standards presented in Figure 3.2. The ISO/IEC27043 is the most granular and comprehensive of them all yet it covers width as opposed to depth. This makes it the most suitable candidate for an implementation of a cloud forensic process as the cloud is dynamic and heterogeneous in nature.

3.3 CLOUD FORENSIC SYSTEM DESIGN REQUIREMENTS

In order to be used in an effective investigation into cloud environments, certain basic requirements need to be met by digital forensic systems aimed at the cloud. To formulate digital forensic system evaluation criteria this section is devoted to a summary

of survey literature that looks at issues being addressed by the existing digital forensic systems, as well as at their shortcomings. Based on these shortcomings, subsequently deduces requirements that need to be met by digital forensic systems that are aimed at addressing digital forensic investigation in the cloud are subsequently deduced.

3.3.1 DIGITAL FORENSIC SYSTEMS SURVEY LITERATURE SUMMARY

Having investigated and presented background on cloud computing, prevalent issues associated with conducting digital forensics in the cloud were attended to. This was done by considering the surveys conducted by other researchers about issues of digital forensics. The research works considered were those by Fahdi [11], Henry et al. [53] and Quick and Choo [99]. To be selected for study, the specific research works had to be survey articles exploring digital forensic issues and, more specifically, issues about digital forensics in cloud computing. They had to discuss challenges facing digital forensics, but were not supposed to propose technologies, techniques or procedures that can be used to investigate cloud environments. Instead existing technologies and procedures were discussed and remaining gaps were brought to light.

Fahdi et al. [11] selected respondents to their surveys from digital forensic practitioners and from digital forensic researchers. In Fahdi et al. [11], the top six limitations of digital forensics as rated by digital forensic practitioners in order of importance are the following: Volume of data, Legal Aspect, Time, Tool Capacity, Visualisation and Automation of Forensic Analysis. On the other hand, the top six limitations according to researchers that were respondents in the survey are Time, Volume of Data, Automation of Forensic Analysis, Tool Capacity, Visualisation, Forensic Training and Legal Aspect. Further in Fahdi et al., cloud computing is highly rated among the technologies, which is a cause for concern for digital forensics, both currently and in future.

Results reported in Henry et al. [53] show that investigations into cloud-based systems and services constitute only a fraction of all investigations carried out by investigators. This is a reflection of the challenges attributed to cloud-based investigations. These challenges oblige investigations to end on client devices owned by perpetrators and not to extend to cloud-based data. Henry et al. [53] further point out that, according to respondents, a lack of standards and tools constitutes most of the challenges associated with investigating cloud environments. This is followed by a lack of skills and certifications, followed by legal issues. The respondents also rate the imaging of incident scenes as the highest cause of complications with regard to investigations in virtualised environments. This is followed by disk acquisition, legal

processes, live response, monitoring/scanning for events and evidence analysis.

Quick and Choo [99] address the issue of the large volumes of data that investigators often have to deal with. They point out that chances of missing crucial evidence data is high when searching for evidence in large volumes of data, and this also increases the time required in conducting an investigation. They furthermore concede that tools to automate some investigation tasks are required. The main solutions that they [99] propose to deal with large volumes of data are data mining, data reduction and subsets, triage and intelligence analysis, and digital intelligence. Data mining, a process of gathering information from a large volume of data, is a research area in its own right. By using data-mining techniques, intelligence can be gathered by analysing patterns in the data without processing the actual data. Data reduction is a technique applied during the data collection process where only parts of the evidence data are collected. Triage is a *“process of sorting enquiries into groups based on the need for a likely benefit from examination”* (Parsonage in Choo and Quick [99]).

In the next subsection, the requirements that are used as criteria to evaluate architectures are presented, including the titles of survey articles that they were taken from.

3.3.2 DEDUCED DIGITAL FORENSIC SYSTEM EVALUATION CRITERIA

From the works of Fahdi [11] and Quick and Choo [99] it can be noted that the large volumes of data that investigators often have to deal with are cited as an issue in both surveys. Large volumes of data need scalable systems to process it. In this thesis the researcher views scalability as a requirement for a digital forensic system that is aimed at investigating cloud environments. Investigations by both the authors mentioned reveal the importance of automation during a digital forensic process. It is in the researcher’s view that standards that were also found to be lacking by Henry et al. [53] can play a role as they can help to specify tasks that can be automated in a digital forensic process and thus result in an efficient digital forensic system. An efficient system can lead into less time spent on an investigation - and time is regarded as one of the major issues by Fahdi [11] and Quick and Choo [99].

Analysis of the results in Quick and Choo [99] reveals that most digital forensics investigations do not get extended to cloud-based data. It is in the researcher’s view also that standards can solve this challenge as standards can assist investigators with processes and procedures that need to be carried out while investigating cloud-based incidences. In this way, standards can therefore also help facilitate cooperation from cloud service providers.

Based on the analysis of the survey, evaluation criteria are proposed for cloud forensic systems. The evaluation criteria presented are not standardised, but aimed at inspiring research in this direction. The criteria are as follows:

1. **Live Cloud Forensic Investigation** - With the ability to analyse a live cloud based system in a standardised manner would help eliminate the need to image data and real time crucial evidence can be obtained from the incident scene. Live forensic response has also emerged as a one of the digital forensic challenges in the survey by Henry et al. in [53].
2. **Implements a standardised digital forensic process** - There are a number of digital forensic processes proposed by researchers some of which are not standardised. There are however emerging efforts towards standardisation of the digital forensic processes such as the ISO/IEC27037 [59] and ISO/IEC27043 [61]. Given the number of existing ones, it would be impractical for an investigator to know them all even after training. A digital forensic system that can lead an investigator through a chosen standard process would help can contribute in efficiency as an investigator would not always need to consult with documentation of the chosen standard every time an investigation is carried out.
3. **Aid and Lead Investigator through a Standard digital forensic Process** - With this requirement satisfied, a digital forensic system can assist investigators go through a complex digital forensic process efficiently. This means that some of the investigation tasks need to be automated. Automation of investigation tasks is emerged as one of the challenges that face digital forensic investigators in the survey by Fahdi et al. in [11]. To determine if a digital forensic system is capable of aiding in an investigation, automating some investigation tasks can then be used as a criterion. That is, e digital forensic system needs to address the automation challenge.
4. **Cloud Based (i.e. runs on the cloud)** - The volume of data that has to be dealt with in an investigation emerged as one of the challenges in Fahdi et al. [11] and Henry et al. [53] surveys. Deploying a digital forensic system on the cloud would not only help address the challenge of dealing with large data by utilise unlimited processing and storage resources provided by the cloud but, the self service and collaborative environment in the cloud would help enable digital forensic investigations to commence timorously and be executed efficiently.

5. **Aimed at investigating Cloud environments** - Existing digital forensic tools and processes are either meant for general purposes or specific for other electronic environment but the cloud. Though the tools can be used in combination to investigate the cloud, there is need for a specialised investigation platforms meant for the cloud. A specialised platform would ensure that all unique aspects of the cloud are adequately covered during a digital forensic investigation.
6. **Digital Forensics Readiness** - Monitoring or scanning for events in a cloud has emerged as one of the challenges in the Survey by Henry et al. [53]. The researcher regards monitoring as digital forensic readiness as it is mechanism that is always on standby to trigger an investigation and also makes data available to initiate the investigation. In the case of the cloud, digital forensic readiness is an additional mechanism incorporated in cloud services that makes digital forensic to be readily available for investigation. Maintaining digital forensic readiness mechanisms come at a cost on the cloud service provider side or on the customer side if the service providers transfer the costs to the customer.

A driving force of businesses is the delivery of their product and digital forensic readiness does not always add value to a business. Businesses therefore do not always invest on digital forensic readiness on their cloud services. If a cloud forensic system would offer this functionality and deployed as a cloud service, the readiness feature can be utilised by cloud service providers as and when needed. Digital forensic readiness therefore needs to be used as a criterion to evaluate a digital forensic system.

A driving force of businesses is the delivery of their product and digital forensic readiness does not always add value to a business. Businesses therefore do not always invest on digital forensic readiness on their cloud services. If a cloud forensic system would offer this functionality and be deployed as a cloud service, the readiness feature can be utilised by cloud service providers as and when needed. Digital forensic readiness therefore needs to be used as a criterion to evaluate a digital forensic system.

3.4 DIGITAL FORENSIC ARCHITECTURES

Digital forensics as a service has been proposed by many researchers over the last four years. Most of the researchers, however, simply presents the concept and do not

provide details on how such a forensic service can be implemented in practice. This section discusses some of the research works that are aimed at investigating cloud environments and/or deployed cloud. The research works are subsequently evaluated based on the requirements presented in Section 3.3.

Some studies such as Sang et al. [110], Thorpe et al. [139] and Thorpe et al. [140] focus on the issue of log-based digital forensics in the cloud for instance securing the logs, and collecting and storing forensically sound logs. Logs form the basis of most digital forensic evidence data. However, there are types of attacks that do not leave a trail of logs when carried out, such as cross-VM side-channel attacks [158]. Logs are also subject to being tampered with by attackers through a logging-process system called hijacking and log files modification. Digital forensics in the cloud may therefore not rely solely on log files as evidence, and more improvements should be made in respect of these approaches.

Other studies including Duykstra et al. [40] focus on implementing tools that can aid the digital forensic process in the cloud, while studies such as [156] keep track of progress and new developments regarding digital forensic investigations conducted in cloud environments. There are however, a limited number of studies that work towards standardising the digital forensic process in cloud environments. As a result, and since the standards themselves are limited, there are no tools that can aid investigators to follow a standard digital forensic process. The research work by Simmons and Chi in [126] does indeed contribute towards addressing issues associated with investigating cloud environments, but more work still needs to be done.

The next section presents the evaluation of a digital forensic system that aims to address research issues associated with investigating cloud environments.

3.4.1 APPROACH ON APPLYING THE EVALUATION CRITERIA

This section evaluates the suitability of a digital forensic system for live forensics based on its ability to interact with a live system and gather evidence from it in a standardised manner. In cloud environments it is not possible to power off a virtual machine or physical server for imaging, as it may be hosting essential services that belong to fellow tenants, hence, the need for the ability to do live forensics.

Assessing whether a digital forensic system implements a standardised digital forensic process is based on whether it follows defined digital forensic processes and procedures, whether the defined processes are standardised (i.e. based on published standards), and whether the processes are incorporated in the implementation of the digital forensic system.

The ability to lead an investigator through a standardised digital forensic process is based on the system's ability to guide the investigator through investigation tasks and to render the tasks to investigators in their standard order. Any web application or service can be developed and deployed in a cloud environment without necessarily implementing the features found in cloud services. To evaluate a digital forensic system on its deployment in the cloud, the system is checked whether it offers a collaborative environment, whether investigators have access to the same version of the service and whether the digital forensic system has resources that scale on demand.

To evaluate if a digital forensic system is aimed at investigating cloud environments, the system is checked if it is designed and implemented with remote access to incident scene capability. To evaluate digital forensic readiness in a forensic system, the system is checked if the system adopts a proactive approach to digital forensic investigations.

3.4.2 EVALUATED ARCHITECTURES

A summary of the architectures is provided in Table 3.1. A signature detection framework is presented in Hegarty et al. [52]. The framework is aimed at performing digital forensic analysis of cloud-based storage services. Using the framework, investigators can perform an analysis of a client's data stored in the cloud. This framework takes advantage of the seamlessly limitless computing resources in the cloud to analyse and detect malicious files from cloud-based systems or storage systems. The article by Hegarty et al. in [52], however, does not emphasise procedures that need to be followed in analysing the data.

Houmansar, Zunouz and Berthier [56] propose a cloud-based system that detects intrusion on mobiles phones linked to the cloud. Instead of having the intrusion detection system running on the mobile device, the system is cloud based and monitors an exact copy of the device, an emulator based on the cloud. The system also takes advantage of the resources available on the cloud and implements an incident detection mechanism that can initiate a digital forensic investigation. Again, no digital forensic procedures are presented in [56].

The digital forensic system by Wen et al. [152], FaaS is concerned with scalable evidence data processing using the cloud. These authors use capabilities of the cloud to analyse large volumes of evidence data. In FaaS, there is no emphasis on standardised procedures that can be followed by the investigator and it does not address investigations on cloud-based incident scenes. Sang [110] proposes an approach whereby a Software-as-a-Service provider and a Platform-as-a-Service provider provide log modules that will be executed on the client's side in addition to the services that they

offer. This approach partially removes cloud service providers from the equation as they may sometime not be willing or have no capacity to participate in an investigation. A weakness in this approach is that it requires that a client installs a CSP's custom module on its devices, and therefore the client may well tamper with it. In this way, responsibilities are shifted to the client, and a client may be the perpetrator being investigated.

Forensic OpenStack Tools (FROST) by Dykstra et al. [40] involves the modification of cloud software - in this case OpenStack - to integrate digital forensic capabilities. In digital forensic terms, this process is referred to as digital forensic readiness.

Shende [110] presents a cloud Forensic as a Service (FRaaS) which was also published by Shende as a chapter in [39]. The cloud-based digital forensic service deployment concept presented in the system is required, as was proposed by many other researchers over the last few years (including Wen et al. [152], Lee et al. [78], Marty et al. [87] and Ruan et al. [109]). There is a need to also focus on the investigator's interaction with a digital forensic service in investigating cloud-based incident scenes where the investigation is following a standardised process. This aspect is receiving less attention as seen in Section 3.2. The system by Lee [77] involves deployment of a digital forensic service in a scalable resources platform such as the cloud. Lee goes further and offers the capability to investigate an incident remotely. The approach by Lee however is based on the physical accessibility of the incident scene so that investigation components can be attached to the incident scene or the evidence device. Physical accessibility is not always possible in the cloud. However, incident scenes in the cloud are often physically inaccessible.

The system proposed by Deepak et al. [36] is a cloud service that utilises cloud resources to analyse videos in order to detect illegal content. Its aim is not to investigate cloud-based incident scenes, which in this thesis is viewed as a requirement.

The technologies proposed in Zeng [157], Lee et al. [79], Lee and Un [78], Baar et al. [148] and Ting and Yang [141] implement cloud-based services to carry out computing tasks such as data analysis and performing searches on big evidence data. The technologies can be referred to as cloud-based forensic labs designed to process data obtained from any storage medium. There is, however, a need for system designs aimed at conducting an investigation by interacting with a live cloud-based incident scene in a standardised manner.

Thorpe [140] and Shirkherdkar and Patil [117] present digital forensic readiness frameworks for cloud computing. There is however a need to also focus on investigating an incident after it has occurred, regardless of whether forensic readiness

mechanisms were in place or not. Thorpe presents a detailed framework on how digital forensic readiness mechanisms can be incorporated in a cloud service stack. Shirkherdkar and Patil present an approach through which communications between the incident scene and a potential attacker can be monitored.

The literature discussed in this section is summarised in Table 3.1. In this table, the sign ✓ means that the feature is supported by the digital forensic system. An ✗ means that the feature is not supported by the digital forensic system. A third sign, –, indicates that no information could be found on the system about the feature. The literature represented in the table is discussed below. The data presented in the table is illustrated in visual form in Figure 3.3.

Table 3.1: Digital Forensic Service Architectures

	Live cloud forensic investigation	Implements a standardised digital forensic process			Aids and leads an investigator through a standard digital forensic process	Cloud Based (i.e. runs on the cloud)			Aimed at investigating Cloud environments	Digital Forensics Readiness
	Retrieve data from live system	Has defined forensic processes and procedures	Processes Standardised	Incorporates standard process in implementation	System renders tasks to investigator based on their standard sequence	Collaborating Investigators have access to same Service Instance	Investigators run similar service version at all times	Hardware resources can scale on demand	Designed with remote access to incident scene capability	Proactive Forensic Approach
Wen et al. [152]	✗	✗	✗	✗	✗	✓	✓	✓	✗	✗
Sang [110]	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓
Dykstra et al. [40]	✓	✗	✗	✗	✗	✓	✓	✓	✓	✓
Shende [115]	✗	–	–	✗	✗	✓	✓	✓	✗	✓
Lee [77] A1	✓	✗	✗	✓	✓	✓	✓	✓	✗	✓
Deepak et al. [36]	✗	✗	✗	✗	✗	✗	✓	✓	✗	✗
Zeng [157]	–	✓	–	–	✗	✓	✓	✓	✓	✗
Lee et al. [79]	✗	✓	✓	✗	✗	✓	✗	✓	✗	✗
Lee and Un [78]	✗	✗	✗	✗	✗	✓	✗	✓	✗	✗
Baar et al. [148]	✗	✓	✗	✓	–	✓	✓	✓	✗	✗
Ting and Yang [141]	✓	✗	✗	✗	✗	✓	✓	✓	✓	✗
Thorpe [140]	✗	✗	✗	✗	–	✓	✗	✓	✓	✓
Shirkherdkar and Patil [117]	✓	✗	✗	✗	✗	–	–	✓	✓	✓
Hegarty et al. [52]	✗	✗	✗	✗	✗	✓	✓	✓	✓	✓

Table 3.1: Digital Forensic Service Architectures

	Live cloud forensic investigation	Implements a standardised digital forensic process			Aids and leads an investigator through a standard digital forensic process	Cloud Based (i.e. runs on the cloud)			Aimed at investigating Cloud environments	Digital Forensics Readiness
	Retrieve data from live system	Has defined forensic processes and procedures	Processes Standardised	Incorporates standard process in implementation	System renders tasks to investigator based on their standard sequence	Collaborating Investigators have access to same Service Instance	Investigators run similar service version at all times	Hardware resources can scale on demand	Designed with remote access to incident scene capability	Proactive Forensic Approach
Houmansar, Zunouz and Berthier [56]	×	×	×	×	×	✓	✓	✓	✓	✓

In the next section, cost minimisation approaches are presented.

3.5 COST MINIMISATION APPROACHES

Shin in [116] attempts to reduce the costs of a digital forensic investigation by proposing a forensic procedure model. Another model that takes the cost of fraud detection into account is the one proposed by Stolfo, Fan, Lee, Prodromidis and Chan in [132]. In contrast to most Intrusion Detection models that concentrate on model accuracy, Stolfo et al. take into consideration the cost implications of an undetected fraudulent activity.

Most of the research that addresses issues of digital forensics either focuses on intrusion detection [56], [91], [132] which corresponds to an incident detection phase of a digital forensic process, or on the latter phases of the process, namely evidence collection and evidence analysis. This is a shortcoming of traditional digital forensics as data in the cloud is huge, distributed and hence, very hard to collect. It is the researcher's view that the evidence identification process can play an important role in reducing the scope of data that needs to be collected or analysed as evidence.

3.6 CONCLUSION

This thesis is aimed at addressing issues encountered when conducting digital forensic investigations in cloud environments. The issues emanate from the virtualised and

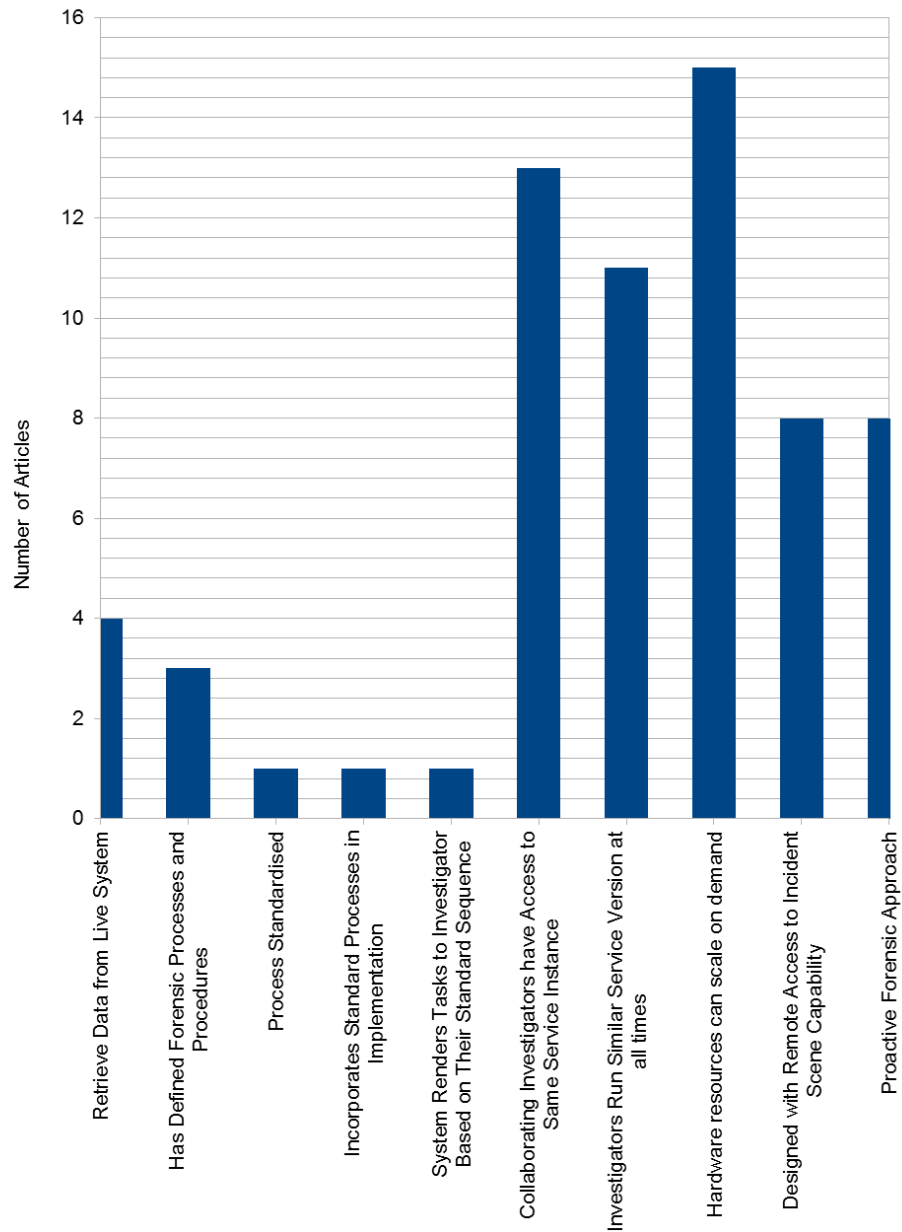


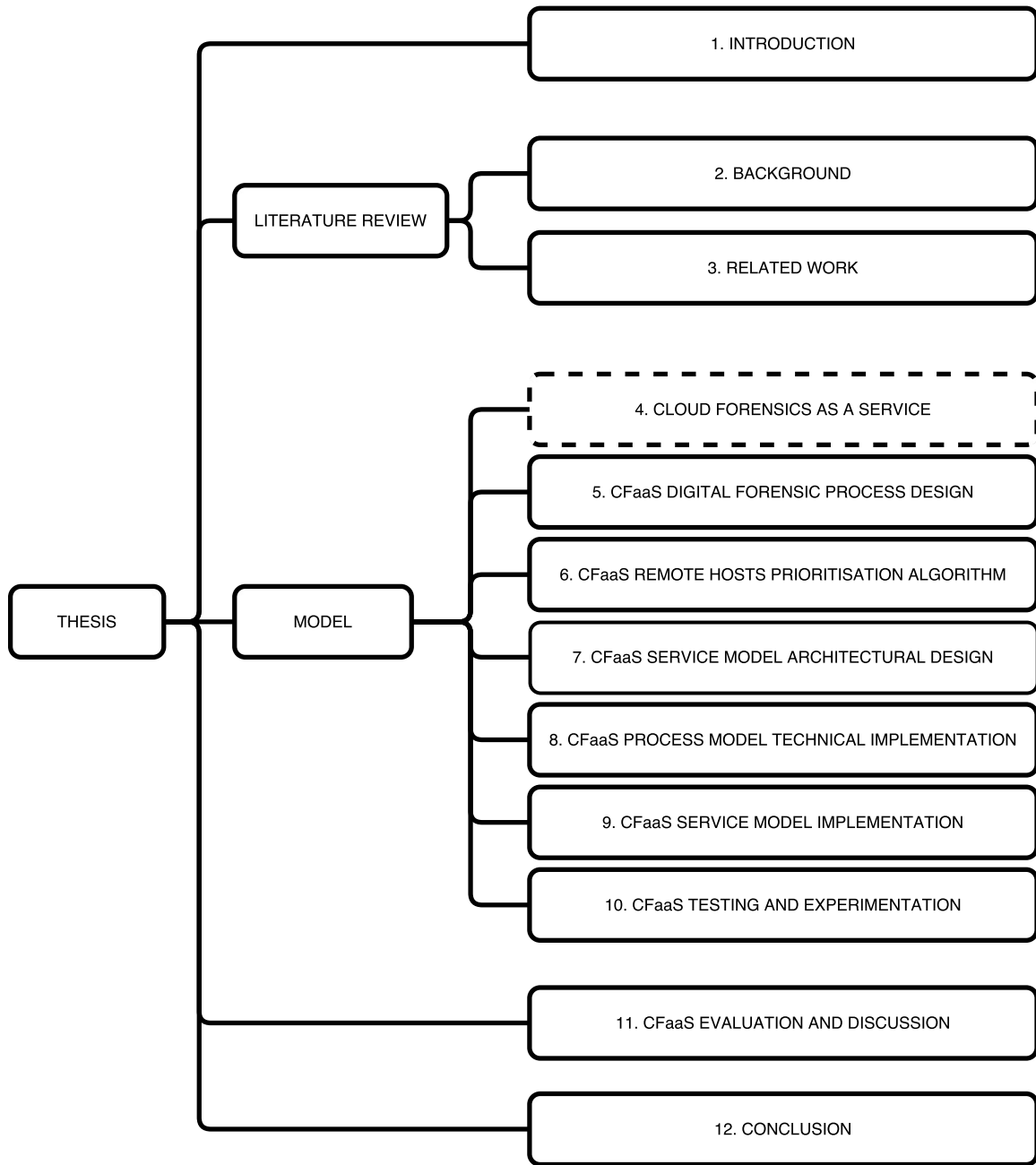
Figure 3.3: Digital Forensic Service Literature Visualisation

distributed nature of the cloud, as well as from the immaturity of digital forensics as a research field. This chapter analysed existing digital forensic processes and standards. Existing gaps on systems that address digital forensics in the cloud were discussed and an evaluation criterion for such systems was developed. The criterion was then used to evaluate and analyse existing digital forensic systems. Cost minimisation

approaches were also covered in Chapter 3. Based on the findings in this chapter, the next chapter proposes a solution to digital forensic challenges in the cloud.

4

Cloud Forensics as a Service



4.1 INTRODUCTION

In this chapter the researcher proposes a solution associated with conducting an investigation in the cloud environments. The proposed solution allows an investigator to conduct cloud investigations in a standardised manner that requires no prior knowledge of the investigation standard. In the previous chapters, digital forensic standards and tools were cited as the main contributors to the challenges associated with digital

forensics in cloud environments. The proposed solution contributes towards addressing the lack of standards and tools associated with cloud investigation.

This chapter is organised as follows: Section 4.2 recaps on the research questions that were raised while discussing the problem statement in Chapter 1. Still in the section, for each research question, the proposed solution approaches to each question is presented. The research questions are nonetheless fully tackled throughout the remainder of this thesis. In general, this chapter serves as a high-level overview of the solution approach, and a summary of the proposed solution is presented in Section 4.3. (Further details of the proposed solution are presented in the subsequent chapters). Section 4.4 presents the scope and key assumptions of the study while Section 4.5 concludes the chapter.

4.2 APPROACHES IN ADDRESSING THE RESEARCH QUESTIONS

This section recaps the discussion on the research questions addressed by this research and briefly explains how each of them is addressed in this research. Expansions of the four research questions are covered from Sections 4.2.1 through 4.2.4.

4.2.1 WHAT ARE THE STANDARDISED PROCEDURES THAT CAN BE CARRIED OUT IN CLOUD ENVIRONMENTS WHILE CARRYING OUT DIGITAL FORENSICS INVESTIGATION?

The above research question sought to address the issue of digital forensic standards. A lack of international collaboration while conducting an investigation in the cloud was cited as one of the problems by Ruan in [107]. For such collaboration to be realised, standardised procedures need to be followed. In this research the researcher proposes that standardised cloud forensic procedures can play a major role in enabling inter-jurisdiction law enforcement collaboration. This question is addressed by proposing a cloud forensic process that is compliant with the ISO/IEC 27043 [61] international standard. The cloud forensic process is discussed in more detail in Chapter 5.

4.2.2 WHAT ARE THE REQUIREMENTS FOR A CLOUD FORENSIC SYSTEM?

This question sought to address the lack of tools to be used in investigating cloud environments. Traditional digital forensic tools such as SANS SIFT [111], Volatility [150], FTK Imager [9], COFEE [90] are meant to address to address only parts of

an investigation process and mainly evidence acquisition, evidence analysis and reporting. These tools however can be incorporated in a cloud forensic when evidence acquisition, evidence analysis and reporting are carried out.

Investigations by the researcher led to the conclusion that a cloud forensic system aimed to investigate the cloud needs to meet the requirements presented in the sections that follow. The conclusion on the requirements was also based on the evaluation criteria that were presented and applied in the previous chapter (see Section 3.3). The requirements are divided into four functional requirements¹ and six non-functional² requirements.

4.2.2.1 FUNCTIONAL REQUIREMENTS

The four functional requirements are as follows:

1. **Implements a standard digital forensic process in the cloud.**

This requirement has been raised by Ruan et al. in [108]. There are many factors that cause the investigating of cloud environments to differ from investigating traditional environments. Data in the cloud cannot always be imaged for off-line analysis due to multi-tenancy, data's size that is often large and its distributed nature. Collecting evidence and the ability to recreate the sequence of events that occurred on the incident scene used to form the basis of traditional digital forensic processes. Since techniques such as these are no longer always possible in cloud environments, evidence obtained from the cloud may be challenged in a hearing such in the case of the Silk Road trial[64]. In the Silk Road trial, the defence challenged evidence where one of the basis was a missed step of preserving packet logs. In the cloud however, missing a step may not be because of an oversight, but because of impracticality to conduct such a step or procedure due to factors discussion in Section 2.4. In order to better defend evidence for acceptability in a court of law or at a hearing, a standardised procedures and techniques need to be followed as per Daubert's rules in [134] and this also applies for investigating a cloud incident. Any digital forensic solution that is aimed for the cloud therefore needs to implement a standard digital forensic process.

A cloud forensic process needs to comply with international standards for the sake of easier multi-jurisdictional collaboration, which is often required when

¹A functional requirement specifies a function that a system must be capable of performing.[22]

²Non-functional requirements state characteristics of the system to be achieved that are not related to functionality.[22]

cloud environments are investigated. It may not always be possible for investigators in country A to access data in a foreign country B. Hence, collaborating authorities in country B can carry out investigation tasks on behalf of country A investigators in the same way investigators in country A would carry it out by following the standard. [28]

While research efforts are ongoing in providing forensic standards, there is a need for a platform on which such standards can be put into practice. This is essential, specifically for cloud environments, as investigation techniques are still lacking.

2. Aids an investigator through the standard digital forensic process in the cloud.

Conducting a digital forensic process in a cloud environment can become a lengthy task that extends over several days and months. Even if investigators were familiar with a provided digital forensic standard, they may find it a daunting task to apply the finer detail of such standard. A cloud forensic system should be able to lead investigators in a transparent manner by means of a standard forensic process during the course of an investigation.

Often, investigators from multiple jurisdictions will need to collaborate in an investigation. This requirement is addressed next.

3. Allow collaboration among multi-jurisdictional law enforcement agencies in a cloud investigation.

The distributed nature of the cloud makes it essential for investigators to collaborate while investigating the cloud. To minimise costs and save time, it is important that the investigation system used should provide a collaborative environment for internationally distributed investigators. Privacy laws in other countries may hinder the progress of an investigation. For example, the European Union's data protection directive is very strict in respect of cross-border movements of data [43]. Procedures that may exist for moving data across borders can be costly and time consuming, and they may often require investigators to travel to and from the data-hosting country.

It is furthermore not feasible to expect investigators to be familiar with all the privacy regulations of foreign countries, as the regulations are of a dynamic and varying nature. This requires that foreign agencies that are better positioned in terms of knowledge of their own country's privacy laws be involved in the

investigation process. Involving the foreign agencies in an investigation can also eliminate the need to move data across borders. Establishing a collaborative environment can dispense with the need to travel, which reduces costs and the time that would be spent travelling. The solution proposed in this thesis provides a collaborative environment by utilising the cloud services.

4. **Semi-automated**

Applying a digital forensic process can be a daunting task on its own. Therefore the system that implements a digital forensic standard needs to be semi-automated in its adherence to procedures specified on the standard. Semi-automated means that some of the forensic tasks can be automated, for instance through scripts that are implemented as such, while those that requires intervention by investigators need to be carried out manually. The system also needs to render processes in their sequence as prescribed by the standard.

4.2.2.2 NON-FUNCTIONAL REQUIREMENTS

The seven non-functional requirements are as follows;

1. **Flexibility**

Stay [129] is one of the researchers who also views flexibility as an important requirement for digital forensic systems. As research matures towards the development of digital forensic standards, there is a likelihood that a number of digital forensic standards will emerge in the near future. These may include international, regional or per-country standards. The cloud forensic system must provide an easy way to incorporate a forensic standard that investigators may choose to use.

2. **Ease of use and efficiency**

The members of the investigation team should be able to perform their tasks without the need for specific additional training on the system. It is likely that bottlenecks may occur when a member of the investigation team delays the process while trying to understand an investigation system. The prevalence of large data combined with a complicated investigation system can prolong the investigation. Having a self-explanatory system available that minimises navigation would contribute towards making the system easy to use.

Although the investigation process must be exhaustive to obtain as much evidence as possible, it must also be completed within a reasonable time. During

an investigation, the system must be able to process the large amount of data in the cloud within a reasonable time, hence allowing the investigators to complete their tasks efficiently irrespective of the amount of data.

Ease of use and speedy processing constitute one of the important requirements for digital forensic systems and this sentiment is also shared by Craiger et al. in [33].

3. Scalability

A digital forensic investigation process can be quite extensive, given the often voluminous nature of evidence data. The system therefore must be able to scale up on demand. If a system is unable to do this, it can delay the investigation and provide room for errors. An inability to expand its capacity can jeopardise the case being investigated - hence the need for a scalable system.

4. Security

The importance of the requirement of security is emphasised by researchers who propose digital forensic systems, including Craiger et al. in [33]. The digital forensic service needs to be protected against attacks so as to prevent attacks from adversaries who intend to disrupt an investigation or tamper with evidence. Evidence data on its way to and from the investigation system, data at rest in the investigation system, as well as data in transit need to be protected. To avoid tampering with evidence and the investigation process, only authorised users must be able to carry out tasks in an investigation.

5. Auditability

An audit may need to be carried out after completion of a digital forensic investigation or during the presentation of the evidence. The system must provide the necessary means for a concluded investigation to be audited. Logging all events that occur during the investigation is one strategy that will support such auditing.

6. Maximum control of the service stack by investigating agency

Criminals often attempt to compromise an investigation system with the aim of tainting or deleting evidence. For this reason, maximum security of the investigation system is essential. Having maximum control or access to the service stack and the infrastructure on which it is deployed can help to enforce essential security measures. This can also minimise external attack vectors on the investigation system.

The next section presents a research question that deals with the design of a system that satisfies the requirements discussed above.

4.2.3 HOW CAN A CLOUD FORENSIC SYSTEM BE DESIGNED TO MEET THE REQUIREMENTS OF THE CLOUD?

The market is saturated with digital forensic tools that were tailored to work perfectly in non-cloud environments. These tools however do not adequately meet the requirements of the cloud. This raised the question of how a cloud forensic system can be designed to meet the requirements of the cloud. To address this research question in this thesis, an architecture of a cloud forensic service model that addresses the requirements presented in Section 4.3 is proposed. The service model including its design is presented in details in Chapter 7.

4.2.4 HOW CAN AN INVESTIGATION IN A DISTRIBUTED CLOUD ENVIRONMENT BE OPTIMISED?

In a cloud environment, an investigation will often expand to include more remote hosts in addition to the incident scene host. This is particular with the cloud as in most cases, applications and platforms are deployed as services in the cloud and are expected to always have a large number of connections from consumers. Identification of remote hosts for further investigation can be time consuming. As a measure to optimise a digital forensic process and minimise costs in return, in this thesis does present an algorithm which automatically prioritises the remote hosts for further investigator. This is a procedure within an entire digital forensic process mentioned in Section 4.2.1 and the procedure is introduced in Section 5.2.5 in the next chapter.

After revisiting the research sub-questions and presenting their respective solution approaches, the framework that is solutions to the main research question in this thesis is presented. The framework is presented next in Section 4.3.

4.3 CLOUD FORENSIC AS A SERVICE (CFAAS)

This section presents the framework that addresses the main research question in this study which is:

On what framework can a cloud forensic solution be based for a cost-effective digital forensic investigation in the cloud?

In this thesis, digital forensic standards are viewed as having a major role to play in addressing the admissibility in a hearing of digital forensic evidence collected from

the cloud. This requirement also applies to general electronic evidence as in Insa [57]. Digital forensic procedures that are compliant with these standards are therefore required. The procedures followed in this research work implement the ISO/IEC 27043 international standard and have been published by the researcher in [120, 122]. Each sub-process from the standard has detailed procedures specific for cloud environments. ISO/IEC27043 was chosen as it is one of the comprehensive processes according to the evaluation provided in the previous chapter (see Section 3.2).

For the proposed standardised procedures to be executed, an environment is required in which they can be deployed for investigators to execute them collaboratively. For this purpose, a Cloud Forensic-as-a-Service (CFaaS) framework to accomplish this aim is proposed. CFaaS provides components through which the procedures can be deployed as a template. Investigators execute each of the procedures based on their authorisation. CFaaS performs or helps investigators perform all the essential tasks required in an investigation, which range from performing automated tasks to conducting manual tasks such as generating reports. Figure 4.1 represents a high-level view of the proposed CFaaS framework.

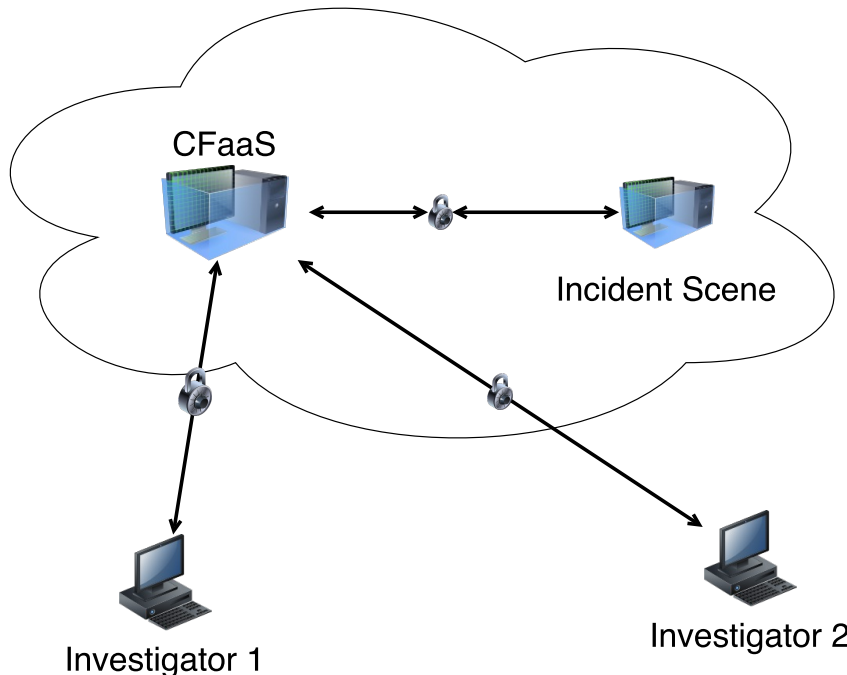


Figure 4.1: CFaaS Conceptual Framework.

The researcher first published this framework in [121] and [124]. The figure shows how investigators can utilise the service (CFaaS) to investigate a cloud-based incident

scene, as well as how the investigators in this framework are able to collaborate from different jurisdictions. Collaboration among investigators from different jurisdictions can eliminate cross-border evidence data movement, which often complicates multi-jurisdictional investigations. Their connections to the service are through secure channels to avoid middle-man attacks on communications. Since the communication between the service and the incident scene is the one most likely to be detected by an attacker. It is the most vulnerable and needs to be protected at all cost. If attackers were to successfully compromise this communication, they can hijack the traffic and feed the forensic service with bogus data.

Within the framework represented in Figure 4.1, is the Cloud Forensic-as-a-Service model (represented by CFaaS in Figure 4.1) which is the major contribution by this thesis. The model is hereafter referred to as CFaaS service model. More details on the design and deployment of CFaaS service model are presented in Chapter 7.

4.4 SCOPE AND KEY ASSUMPTIONS OF THE STUDY

In this section, the scope and key assumptions of the study are presented.

4.4.1 SCOPE

There are two main contributions of this thesis which are the framework and its implementation. The implementation in the thesis serves as a proof of concept. A comprehensive evaluation of the implementation which includes testing it in practice is planned for future work. The implementation which is ultimately a prototype is evaluated through investigating a simulated criminal case. Algorithms and architectures are evaluated using formal analysis.

The focus of this study is on a cloud based incident scene as virtual machine instance. The study therefore does not go deep into investigating the hardware components of an IaaS provider nor does it go deep into investigating applications hosted by SaaS providers.

Due to the cloud's multi-jurisdictional nature, legal frameworks from different jurisdictions become important when an investigation is conducted in the cloud. This study, however, does not extensively examine legal frameworks from those jurisdictions that have implications on cloud investigations.

The implementation of the framework is based and only tested on Ubuntu based cloud based incident scenes. The heterogeneity of the cloud environment however does require a cloud forensic solution to be able to investigate all state-of-the-art operating

system based cloud instances. The implementation however can be adapted to work with any operating system that supports remote access.

Cloud services often comprise of networked services or instances. An incident scene therefore may comprise of just one instance or comprise of multiple networked instances. In this study however, the investigation focuses on just one cloud instance from the beginning until the end. Also in this case, the implementation can be adapted to work with multiple cloud instances simultaneously.

4.4.2 KEY ASSUMPTIONS

In this thesis, it is assumed that not all cloud-based services will have digital forensic readiness mechanisms in place. For this reason, from the standard digital forensic process in the ISO/IEC27043[61] standard on which this study is based, the “Readiness Process” group is not considered. The study is only based on the “Initialisation Process”, “Acquisition Process” and the “Investigative Process” groups.

In this study it is also assumed that the digital forensic investigation team have remote access and credentials — with root or privileged access— to the incident scene(s). These are the credentials that the investigators use to interact with the incident scene while doing the investigation and the CFaaS system also use the same credentials while executing automated tasks.

It is also assumed in this study that the cloud-based instance that is being investigated remains on-line for the duration of the investigation. In practice this will not be the case. Instances may be brought down for maintenance, to save costs or be terminated while the investigation is ongoing.

Installing third-party software on an operating system can modify contents of a RAM and the file system. For this reason, installing software on the incident scene is likely to modify or contaminate evidence. In this study however, it is assumed that installing additional software on the incident scene will have negligible effect on the evidence. This aspect however does need to be studied further.

An investigation that involves multiple jurisdictions in this study is assumed to be costly and time consuming when investigators have to travel in between the jurisdictions involved. A platform that can allow the investigators from the jurisdictions concerned to collaborate without having to travel is assumed to be cost effective. This aspect also does require further investigations to provide monetary value comparisons.

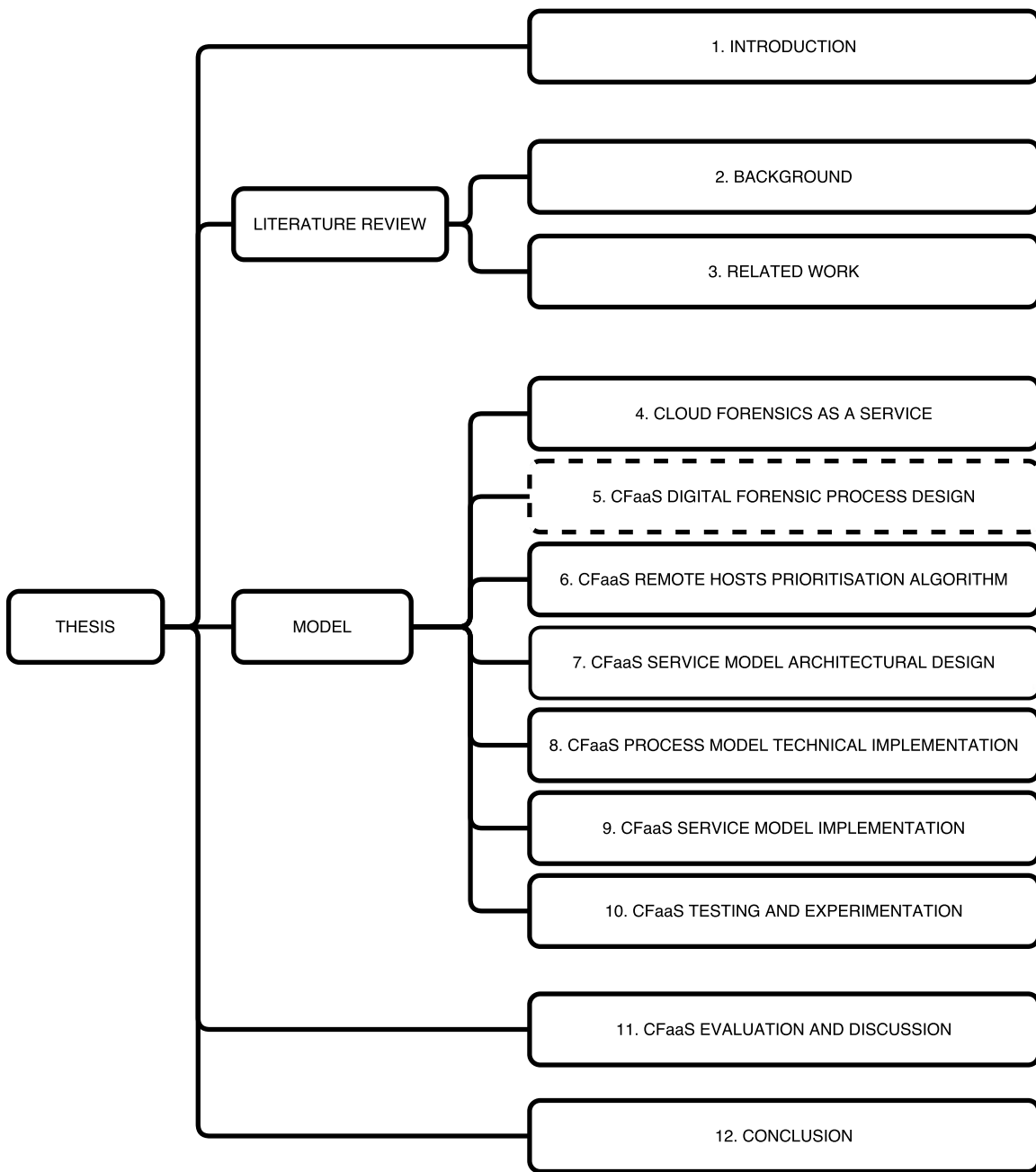
4.5 CONCLUSION

This chapter presented CFaaS framework, which is a cloud-based digital forensic service that can be used to conduct an investigation on cloud environments in a standardised manner. The service minimises the need to travel and for evidence data to be moved across borders by providing a collaboration environment for investigators. This is ideal for any investigation that involves multiple jurisdictions as the cross-border movement of data may conflict with privacy laws in other countries. In this chapter, author also discussed the fine points of the system requirements and proposed a framework that addresses these challenges when an investigation has to be conducted. The scope covered by the framework and the study as a whole has also been presented.

Chapter 5 presents more details on the standardised cloud forensic procedures.

5

CFaaS Digital Forensic Process Design



5.1 INTRODUCTION

Characteristics of the cloud that were discussed in the previous chapters pose challenges for digital forensic investigators, regardless of whether the investigation needs to be performed on a victim's data or on a perpetrator's data. This thesis presents a live forensic framework that can be used to conduct a forensic investigation in a cloud environment in a semi-automated manner. Some procedures to be conducted during

a digital forensic investigation can be automated, while some can only be carried out manually by humans. The proposed solution is of a semi-automated nature in a way that procedures that can be automated are implemented as such and investigators are guided through manual tasks. In this chapter, “tasks” will be used to refer to procedures that are carried out automatically by script or manually followed or undertaken by investigators during an investigation process.

This chapter focuses on the design of a CFaaS digital forensic process model that was introduced in the previous chapter (see Section 4.2.1) as part of a solution to address these challenges. Section 5.2 presents the proposed cloud forensic process model that is compliant with the digital forensic process standard in [61]. Section 5.3 concludes the chapter.

5.2 CLOUD FORENSIC PROCESS MODEL

This section presents the design of the proposed digital forensic process model that can be used in a cloud environment and that is compliant with the ISO/IEC 27043 international standard. The standard groups digital forensic investigation processes into four categories: readiness process, initialisation process, acquisition process and investigative process. In this thesis the readiness process is omitted and the initialisation, acquisition and investigative process groups are considered as they are the processes in which digital forensic investigators are actively involved after an incident has occurred. The assumption in this thesis is that not all cloud environments that would require an investigation will have digital forensic readiness mechanisms in place and readiness is an optional component in ISO/IEC27043. Sub-processes extracted from the three process groups of the ISO/IEC27043 standard are as depicted in Figure 5.1. The sub-processes will also be referred to as “processes” in this chapter as it is also the case in the ISO/IEC27043 standard.

In this thesis, *Evidence Acquisition* process is considered and the *Evidence Collection* process from ISO/IEC27043 is not considered. The reason for the considering *Evidence Acquisition* only is because *Evidence Collection* may not be possible in some investigation cases due to the fact that cloud-based data is not always physically accessible. This problem has also been raised by the Cloud Security Alliance (CSA)’s incident management and forensic working group when mapping of ISO/IEC27037 to cloud computing was carried out [21].



In addition to the thirteen sub-processes presented in Figure 5.1, the *Documentation*, obtaining authorisation and information flow management sub-processes run

concurrently with these processes from the beginning to the end of the investigation. The concurrent processes are depicted in Figure 5.1 as parallel lines. The lines indicate where the processes get activated.

The processes discussed in this section are *Incident Detection*, *First Response*, *Planning Process*, *Preparation*, *Potential Evidence Identification*, *Evidence Acquisition*, *Evidence Transportation*, *Evidence Storage*, *Evidence Examination and Analysis*, *Reporting*, *Presentation*, and *Investigation Closure*.

The following sub-sections presents the procedures involved in each process — from *Incident Detection* through to the *Investigation Closure* process. Concurrent processes are also discussed in their separate sub-section, Section 5.2.12. The procedures presented in this chapter were first published by the researcher in [119] and [122]. The procedures are presented in summary in this chapter and discussed in even more details in the implementation of the process in Chapter 8.

5.2.1 INCIDENT DETECTION

With digital forensic readiness in place, *Incident Detection* is the process that triggers the whole investigation process. The sub-processes involved in it include the steps *Register Case* and *Incident Description* as illustrated in Figure 5.2. From Figure 5.2 onwards, tasks marked with the icon  represent manual tasks, while the icon  represents automated tasks or script tasks. In this chapter here after, script task and automated task are used interchangeably when referring to automated tasks. Case registration in a *Incident Detection* process is a manual task often performed by administrators (not system administrators) who are part of the investigation team. The registration initiates the investigation process. The *Incident Description* process may also be performed by administrators or the incident scene owner. During this process a comprehensive description of the incident is provided. This information constitutes a reference point for all the other subsequent investigation sub-processes.

Incident Detection is followed by the *First Response* process, which is dealt with in the next section.

5.2.2 FIRST RESPONSE

The *First Response* in a forensic investigation is meant to contain the incident scene. In general, if the compromise on a system puts people's lives in danger, they are evacuated. If there has been system failure during the compromise, attempts are made to restore the systems to their operational state. For a cloud environment

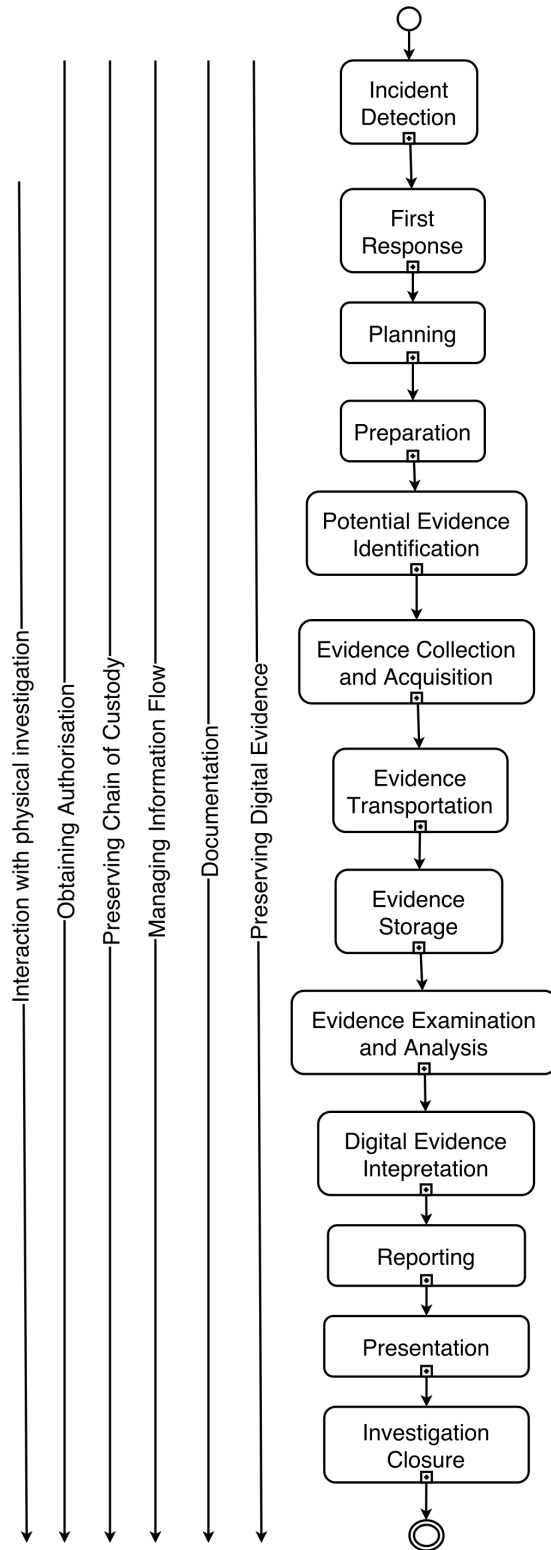


Figure 5.1: Cloud Forensic Process (Adapted from [61], see Figure 3.1)

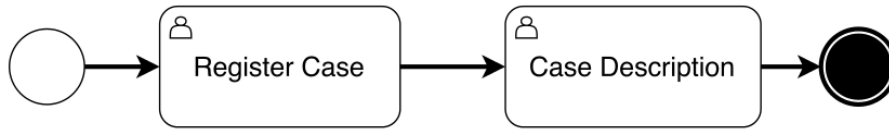


Figure 5.2: Incident Detection

the sub-process illustrated in Figure 5.3 is considered as a *First Response* process specific to the cloud environment. Cloud computing comprises physical hosts and virtual machines accessed remotely through the Internet or Local Area Network. An incident scene is therefore assumed to be a remote physical host, virtual host or a network.

The first action to be performed in the *First Response* process is establishing a secure connection to the remote host, which is the incident scene. It is important that a connection to the incident scene be through a secure channel to avoid eavesdropping on the traffic during the investigation process. The task used to achieve this is called *Establish Secure Connection* in Figure 5.3. If a secure channel can be established, a connection is initiated and this action is referred to as *Remote Connection*. The action *Start Servers* is responsible for initialising software servers and agents that will be used in the investigation process. The action *Enable Secure Logging* enables logs to log activities on the incident scene both by investigators and system users or perpetrators.

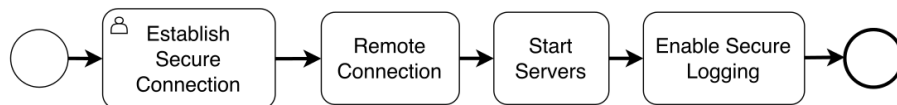


Figure 5.3: First Response

After the *First Response* process has been concluded, the process to plan how the entire investigation will be carried out is conducted and this process is discussed in the next section.

5.2.3 PLANNING PROCESS

Proper planning is essential for a cost-effective digital forensic investigation. The process presented in Figure 5.4 begins with organising a digital forensic team to handle the investigation. Different types of network exist. In each network type, evidence may be located in a different location compared to another network type. In a physical network, potential evidence may be located in physical routers and hardware

information such as media access control (MAC) addresses are static whereas, in virtual environments it is not the case. In virtual networks - as it is a common case with cloud computing - an investigation may need to be handled differently as it characterised with virtual hardware resources that can be added and removed dynamically.

All of this will inform the requirements for the tools that will need to be acquired for investigation purposes. The last step entails listing potential evidence sources in different scenarios, as identified in the preceding procedures under the *Planning Process*.

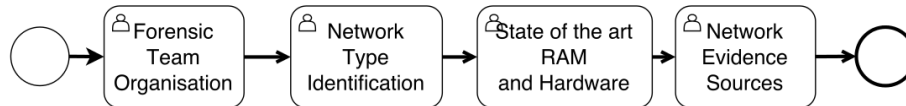


Figure 5.4: Planning Process

The *Preparation Process* is presented in the next section. Section 5.2.4 presents procedures required to prepare before any investigative interaction with the incident can commence.

5.2.4 PREPARATION PROCESS

The *Preparation Process* represented in Figure 5.5 covers three sub-processes, namely task assignments, the specifications for evidence analysis tools and collection tools, and tools acquisition. The task assignment sub-process involves assigning tasks to the forensic team members that was assembled as part of the *Planning Process*. This is important as, for example, not all forensic team members may have clearance to handle certain organisational data. Given the scenario outlined in the *Planning Process* which are the state-of-the-art network type and potential evidence sources, digital forensic tools specifications capable of handling such technologies are defined. Once the tools specifications have been outlined, the digital forensic tools are selected. An example of tools to be selected may include on-line file scanning services.



Figure 5.5: Preparation Process

The next section presents the *Potential Evidence Identification* process. It is during this process that interactions with the incident scene commence.

5.2.5 POTENTIAL EVIDENCE IDENTIFICATION

During the process of *Potential Evidence Identification* illustrated in Figure 5.6, data that may be used as evidence in the incident scene is identified. The process includes four sub-processes, namely locating paging file, RAM classification, corrupted data identification and determining remote hosts connected to the scene. Operating systems can use a file stored on the hard disk as an extension to existing RAM. This file may or may not exist and this depends on whether the user chooses to create it. If the file does not exist, the sub-process can be skipped. The RAM classification sub-process involves categorising the RAM on the incident scene, which involves obtaining information found in system BIOS about the RAM hardware. Information obtained includes RAM type such as DDR2, DDR3, etc., size, configured clock speed and others. However, this information becomes relevant only when an incident requires low level investigation has to be conducted on the physical infrastructure(usually IaaS). An example of such incidences is the cross-VM side channel attacks [158]. . These will help in selecting appropriate tools to be used during the investigation.

An incident is usually characterised by file creation, modification or deletion. Files are considered to be corrupted if they are modified, but not by the owner. The corrupt data identification sub-process deals with identifying files that were recently modified. Identifying these files can provide leads to new evidence such as users that are logged in or were recently logged in. The last action to be carried out involves determining the remote hosts that have active or recent connections to the incident scene. Further investigations can be carried out on the remote host. In this paper, however, the investigation is restricted to the incident scene.

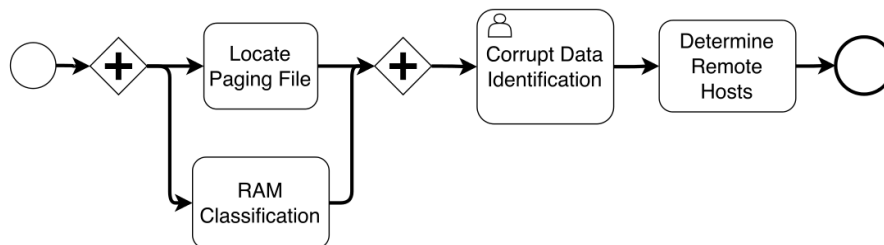


Figure 5.6: Evidence Identification

The identification of potential evidence in the incident scene is followed by the collection process, which is presented in the next section.

5.2.6 EVIDENCE ACQUISITION

The ISO/IEC27043 defines collection as a “*process of gathering the physical items that contain potential digital evidence*” and acquisition as a “*process of creating a copy of data within a defined set*”. In cases whereby physical access to the incident scene is possible, the data considered as potential evidence discovered on the incident scene is collected for preservation, storage and dead forensic analysis. In cloud environments however, physical access to the incident scene is limited or not possible.

For this reason, only acquisition is considered in this chapter as this thesis is aimed at investigating cloud environments. The acquisition process is as represented in Figure 5.7 and it begins with the installation or activation of a data acquisition agent. Data to be acquisition using the agents includes corrupted files, created files, and recently modifies files (or any other files that may be related to the case being investigated). The next datasets to be collected are network dumps. The RAM from the incident scene is also captured for dead forensic analysis. The last procedure to be carried out involves the preservation of the collected data or potential evidence.

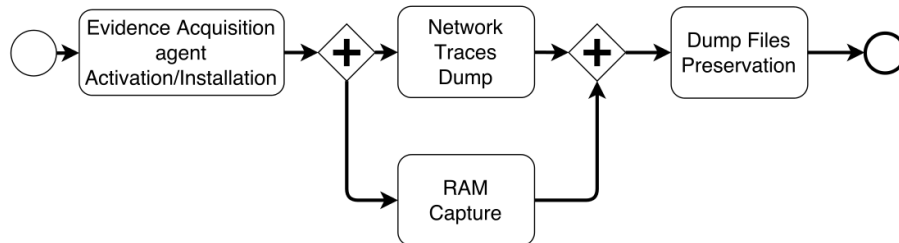


Figure 5.7: Evidence Acquisition

The next section explains the *Evidence Transportation* process. Measures taken while carrying out this process are discussed in the section.

5.2.7 EVIDENCE TRANSPORTATION

The *Evidence Transportation* process in Figure 5.8 deals with moving evidence from the incident scene where it was collected to a secure location under the control of the investigation team. Evidence can be transported in a physical removable drive or via a secure network link. To transport evidence through a physical drive, the traditional procedures of transporting evidence may be followed such as not transporting the drives in electromagnetic field resistant containers. If the evidence is transmitted via

a network link, the link needs to be secured first. The destination needs to be under the control of the investigating team and it must be secure.

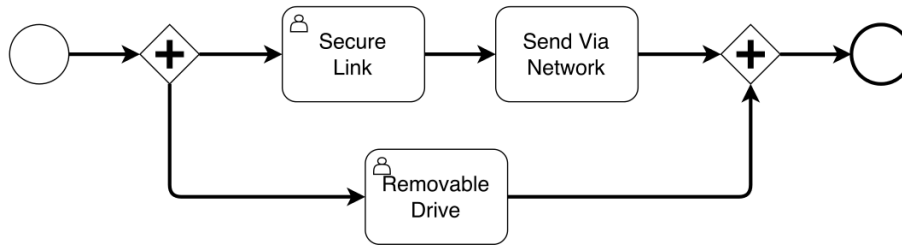


Figure 5.8: Evidence Transportation

Potential evidence that has been collected and transported by means of the previous processes has to be stored securely. *Evidence Storage* is therefore at issue in the next section.

5.2.8 EVIDENCE STORAGE

Evidence is transported with the purpose of being stored in a secure environment. The *Evidence Storage* Process in Figure 5.9 consists of three activities, the first of which involves verification of the integrity of the evidence. The *Evidence Acquisition* Process is always concluded by preservation of the evidence and one of the methods of preserving is the calculation of hash codes. The integrity of the evidence is subsequently verified by comparing the hash recalculated from the evidence and the hash code provided through *Documentation*. The two options for store evidence are online storage and off-line storage on physical drives respectively.

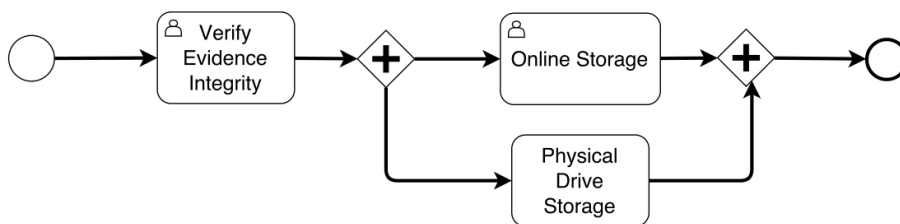


Figure 5.9: Evidence Storage

The stored evidence needs to be analysed. The analysis can either take place on a live cloud incident scene or be conducted based on potential evidence data collected from the incident scene. The analysis process is presented in the next section.

5.2.9 EVIDENCE EXAMINATION AND ANALYSIS

The most comprehensive and most essential process in a digital forensic investigation is the evidence analysis process represented in Figure 5.10. Analysis of the incident scene can be separated into two host network based analysis and host based analysis. These are however conducted in parallel. The analysis process incorporates procedures by Burdach in [24] and [25]. The evidence analysis discussion begins with a network analysis adapted from [65].

5.2.9.1 NETWORK BASED ANALYSIS

These network analysis procedures are applicable to both off-line network dump files and connections on a live system. The network connections analysis is as follows: Copy MAC address and the route kernel cache tables [24] - These cache tables are copied first, as the information on the tables is volatile but helpful to an investigator to determine the networks that the incident host is connected to.

List current and pending TCP/UDP connections - A connection that belongs to an attacker is likely to be among these connections if the attack is still ongoing. If a rootkit loaded into the kernel hides an open port, it can be detected by using information from this action [24].

Decode Connections - Decoding a network connection protocol involves determining and decoding a connection's protocol such as TCP, HTTP, or UDP [146]. Decoding a network connection helps in extracting packet attributes relevant to an investigation.

Extract Packet Attributes - The procedure that follows decoding network connections is the extraction of packet attributes from each connection.

Convert Connection Attributes into Database - The packet attributes are then converted into a database format [44]. This is done for the sake of easier visualisation of the packet attributes using existing data visualisation tools.

Network flow reconstruction - The next procedure involves reconstruction of network flows [19], [49]. This action can be skipped if the analysis is performed on a live network stream and not on network dump files. The reconstructed flow can help to describe and profile a network-based attack or incident [37]. If it is a case that involves media transmission, packets that are used to reconstruct the media can be identified and marked from the traffic.

Visualise Reconstructed network flow - The next action comprises a statistical visualisation of the reconstructed traffic flow [73, 130]. This also includes visualising packet attributes that were converted into the database format. Some visualisation

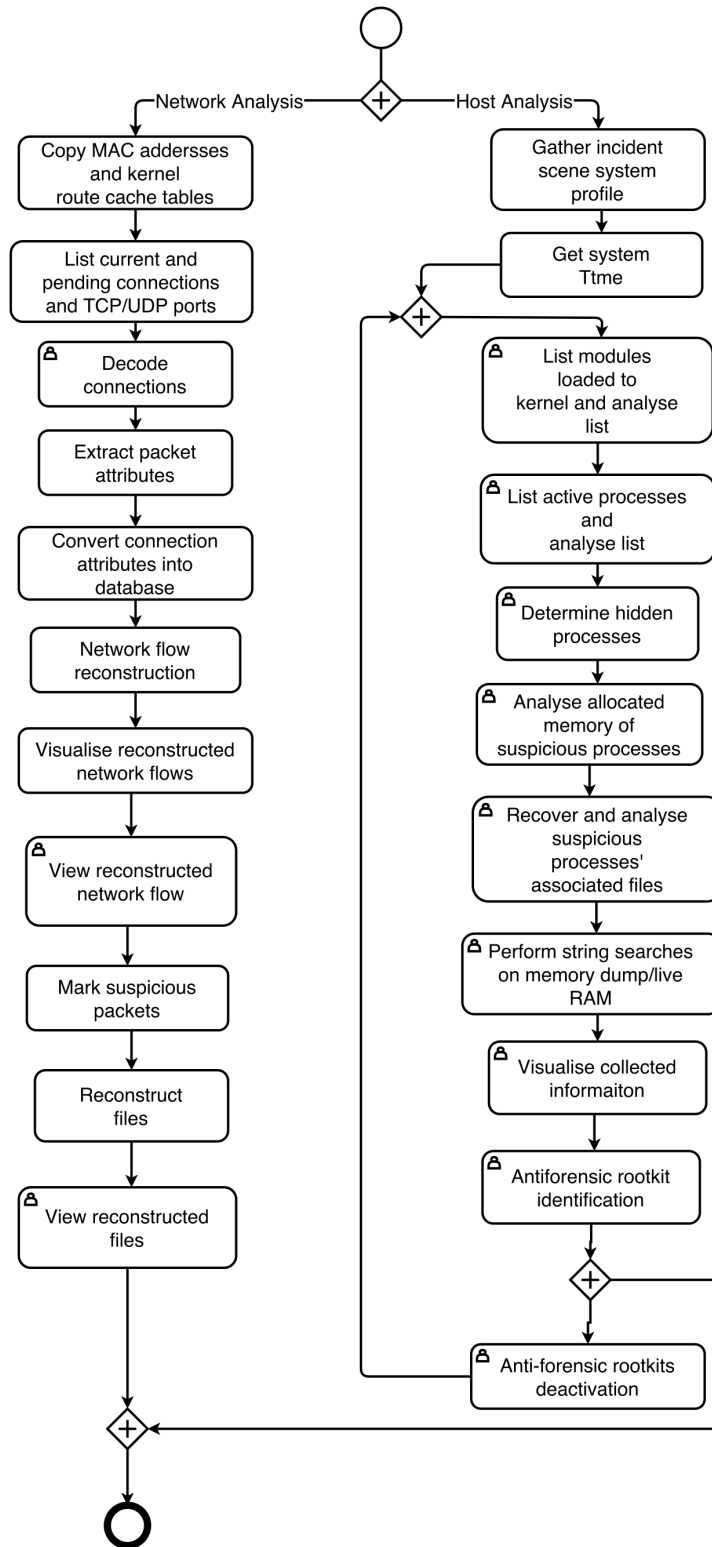


Figure 5.10: Evidence Analysis

tools require human intervention. Hence, this action may be carried out in conjunction with viewing and analysing the generated visual statistics.

View reconstructed network flow - Viewing of the visualised flow is followed by marking suspicious packets on the network flow.

Mark suspicious packets - Marked packets are isolated from the main traffic and therefore reduce the data to be handled in an investigation.

Reconstruct files - A subsequent procedure to be carried out is reconstruction of files, such as media files, if they exist in the reduced network traffic. Reconstructing files after the network traffic has been reduced results in an effective analysis process. The process would not be effective if the reconstruction was done on the original network traffic. The reconstructed files may also include scripts that an attacker may be transferring to or from the incident host.

View reconstructed files - In the process of viewing the reconstructed files, the investigator may also run and analyse the recovered scripts in an isolated and controlled environment. Viewing reconstructed files concludes network-specific analysis procedures.

Results from network analysis are then combined with results from the host based analysis.

5.2.9.2 HOST BASED ANALYSIS

Analysis of the host begins with gathering the profile of the incident scene host system. The information gathered to profile the system includes the following: host operating system version, host name, domain name, hardware information, swap partitions, local file systems, mounted file systems and system uptime [25]. The information gathered here is essential for decision making by the investigator while analysing using his/her intelligence or intuition.

Though it is recommended in this thesis that the procedures be performed in the sequence as depicted in Figure 5.10, a system that implements these procedures needs to be flexible to allow investigators to carry out the some procedures concurrently or in another order that they deem fit given a context of an incident being investigated. For example, as shown under the "Host Analysis" part in 5.10, "Get System Time" may potentially be carried out before "Gather Incident Scene System Profile". However, in another example, "Analyse allocated memory of suspicious processes" may not be carried out before "Determine hidden processes". The order of the procedures as chosen in Figure 5.10 has been set simply so that results obtained from earlier procedures can be utilised in other consecutive procedures.

If the procedures are carried out in their order, the next procedure performed by a script is aimed at getting the system time. The consecutive procedures that follow are recursive and can be repeated if anti-forensic processes and/or hidden processes and kernel modules were discovered during the first iteration. The first of those procedures is retrieving the list of loaded kernel modules and analysing the list. If malicious modules are discovered, each module is investigated further. This procedure is followed by listing running processes on the incident host. The process list is then analysed to identify any suspicious process names. One way to identify suspicious processes is by unfamiliar process names which do not ship with the operating system and/or are never installed by the system user. Process names are at a high level. A deeper inspection of each process is required which can be to identify systems files that it has opened or any ports that are open and are related with it.

The memory space allocated to each suspicious process is analysed. Files such as scripts that started the process can be recovered from the memory space. During an attack, the attacker loads a script or scripts into the victim host, runs it and then deletes it in an attempt to hide any traces. If the process started by the script or program is still running, these scripts can be recovered from memory. There is a need to perform a string search from the memory dump or live memory.

String-based information that can be obtained from the memory includes IP addresses, domain names and commands history (among others). Technical details on how string search is performed on RAM are discussed in Chapter 8 where the implementation of the digital forensic process is presented. Some of the information collected under host analysis can be visualised statistically. Such information is visualised in the Visualise Collected Information procedure.

To hide their traces, attackers use anti-forensic techniques, one of which is data-hiding and process-hiding rootkits [133]. The next procedure is therefore to try and discover such rootkits. If any rootkits are discovered, they are deactivated and the host analysis process is restarted to uncover new evidence. If no rootkits were discovered, the process continues as this concludes analysis of the host. Technical skills are required to both be able to identify rootkits and deactivate them. Since investigators are assumed to have remote access to the cloud based incident scene in this thesis, anti-virus software can also be used to deactivate or remove root kits and then restart the analysis process.

In the next section the research interprets the evidence analysis.

5.2.10 DIGITAL EVIDENCE INTERPRETATION

After examination and analysis of the digital evidence, a decision will have to be made on whether the evidence is strong enough to substitute the existence of the case. This is done through interpretation of the analysis performed on the evidence data, which includes among others viewing the statistical analysis of the data. The steps in the process of evidence interpretation are reviewing statistical analysis of the evidence and validating the case based on the statistical analysis. This process is depicted in Figure 5.11.

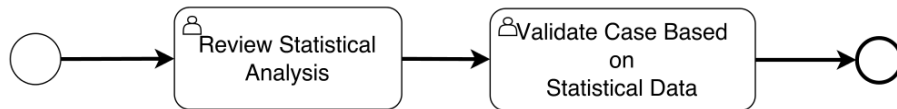


Figure 5.11: Evidence Interpretation

5.2.11 REPORTING, PRESENTATION AND CLOSURE

In the cloud environments, the *Reporting*, *Presentation* and closure of the case are as described in the ISO/IEC27043 standard.

5.2.12 CONCURRENT PROCESSES

The concurrent processes of *Obtaining Authorisation*, *Documentation* and *Managing Information Flow* commence in the *Scenario definition* process according to ISO/IEC27043. The *Preserve Chain of Custody* and the *Preserve Digital Evidence* processes commence within the readiness processes group. Since standard process (see Figure 3.1) is considered from the *Incident Detection* process onwards in this thesis, afore mentioned four processes run throughout the cloud investigation processes presented in this chapter (see Figure 5.1).

Obtaining Authorisation involves obtaining incident scene credentials from the system owner

In the cloud investigation process presented in this thesis, the *Documentation*, *Managing Information Flow*, *Preserve Chain of Custody* and the *Preserve Digital Evidence* concurrent processes are regarded as automated processes. The details of the personnel who undertook the manual investigation task are documented. In this thesis, this is also regarded as being part of both the *Managing Information Flow* and *Preserve Chain of Custody*. The documented information also includes the outcomes of an investigation procedure or process.

The *Preserve Digital Evidence* is carried out every time potential evidence is obtained through acquisition from the incident scene.

Concurrent process that is initialised in the *First Response* process according to ISO/IEC 27043 is the *Interaction With Physical Investigation* process. In the case of cloud, there is often no physical access to the incident scene hence, this process refers to the moment when investigators start connecting and communicating with the incident scene through communication protocols at their disposal such as SSH or web consoles. The interaction with the incident scene is always through automated tasks within the digital forensic process in CFaaS as well as remotely from investigators through SSH or web consoles (HTTP(S)). Details on how investigators interact with the incident scene are presented in Chapter 8.

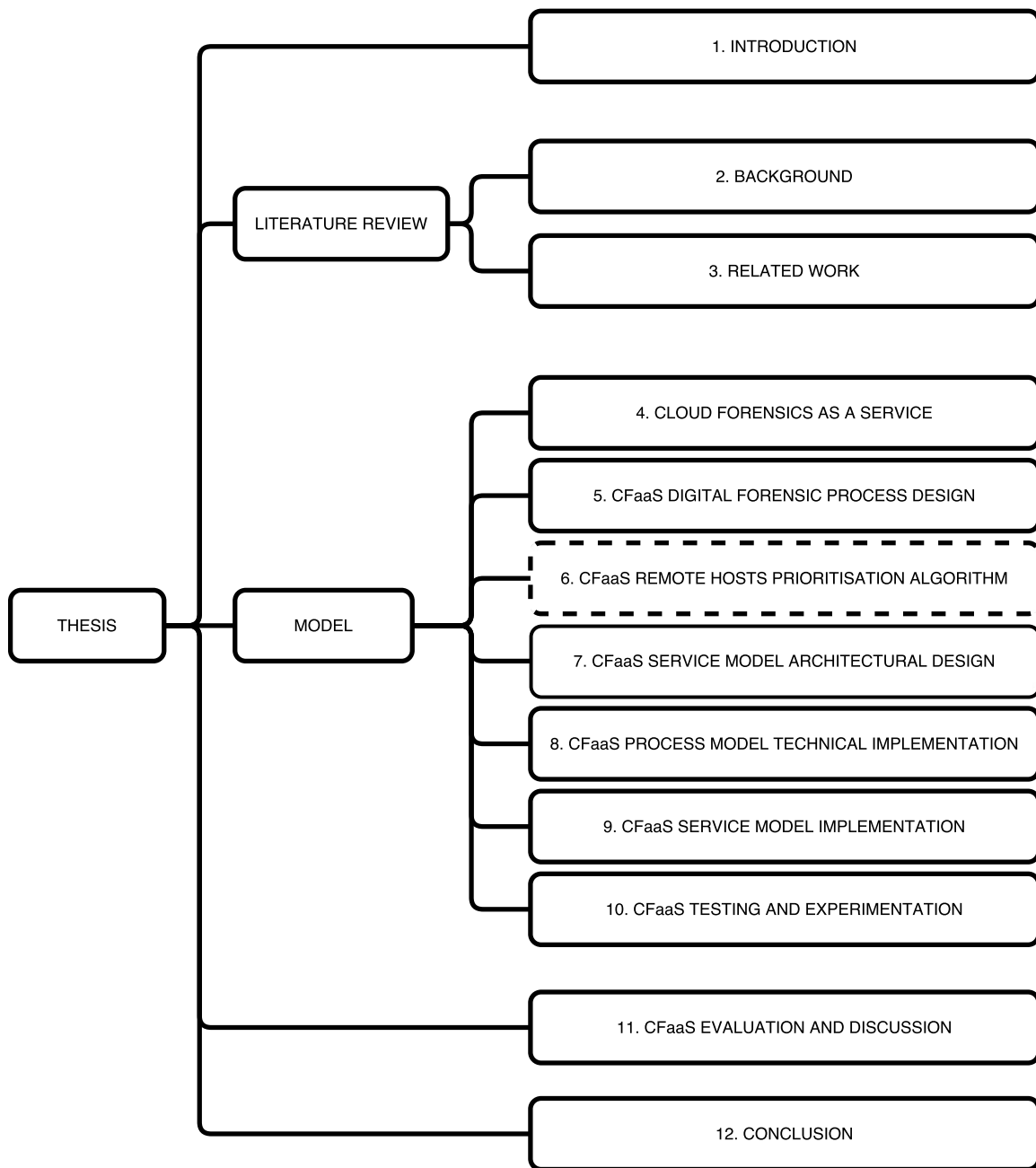
5.3 CONCLUSION

The cloud environment is unique due to its distributed and virtualised nature. Conventional procedures applicable in traditional environments cannot be applied directly in cloud environments. Specialised procedures are required for cloud environments. On addressing the research question on standardised digital forensic procedures for the cloud, the digital forensic process model in this chapter that constitutes the standardised procedures was presented. These are the procedures that are implemented in the proposed digital forensic service, CFaaS. Details on how the procedures can be implemented are presented in Chapter 8.

In cloud environments, an investigation need not be confined to a single incident scene. While investigating an incident scene, leads may be discovered that would require an investigation to be extended to additional cloud virtual instances. Chapter 6 presents the design of the algorithm implemented by the *Determine Remote Hosts* script task. The task is part of the digital forensic process and is performed inside the *Potential Evidence Identification* sub-process. The process deals with determining the additional cloud instances and remote hosts that would be investigated. Chapter 6 therefore addresses the research question on optimisation.

6

CFaaS Remote Hosts Prioritisation Algorithm



6.1 INTRODUCTION

In cloud environments, an attack often originates from a remote host. The remote hosts access the victim host - hereafter referred to as incident scene - via the network. An investigation on the incident scene often leads the investigator to those remote hosts. As the number of network connections to a cloud-based host is normally large, it is important for selection of the more relevant remote host for further investigation. This is part of evidence data reduction, which in turn contributes by addressing the

research question on optimising an investigation in a distributed cloud environment. The algorithm optimises the investigation process by automating the prioritisation of remote hosts that can be considered for further investigation. This chapter presents a formal representation of an incident scene and the **SeL**ection **Of rEmote** hosts (SLOE) algorithm that selects and prioritises remote hosts for further investigation. The algorithm presented in the current chapter was published in by the researcher in [123]. In Section 6.2 the formal mathematical model of the scene is presented. An algorithm that determines connected hosts to be investigated on the basis of the presented formal model is given in Section 6.3. The evaluation of the algorithm is presented in Chapter 10 together with evaluation of the entire CFaaS framework.

6.2 INCIDENT SCENE MODELLING

Consider a live system host that hosts cloud services in a cloud environment where an incident has been reported. As the host is residing in the cloud, a large number of connections from remote hosts that consume the hosted cloud service are expected. A set of such remote hosts — both connected and recently disconnected — is represented by set H as follows:

$$H = \{h_i | h_i \text{ is a remote host}, i \in N\} \quad (6.1)$$

Take for example, $H = \{h_0, h_1, h_2, \dots, h_9\}$. In this example, the host has 10 remote hosts that have active or inactive connections with the incident scene. In this thesis, active connections refer to network connections where a communication is in an active state and data is being transmitted between the local and the remote hosts. Inactive connections refer to network connections where a connection request has been sent and connection is not yet established, or where a connection has been closed from the remote host but may still be in a waiting state locally. From the set of hosts, set H , that have active connections or inactive connections with a victim, remote hosts need to be selected and prioritised, for a cost effective investigation. A remote host can be associated with at least one active or inactive connection in the victim host. The set of hosts with active connections (H_A) and the set of hosts with inactive connections to the victim host are referred to as H_D . H_A and H_D are covering subsets of set H , i.e.

$$H = H_A \cup H_D \quad (6.2)$$

Incident types that can be detected in a computer environment are from a finite set defined in the computer security domain (such as defined by the United States CERT [5]). As an example, the categories as defined by the US CERT, which are referred to as types in this chapter, include (among others) unauthorised access, denial of service and malicious code. In this chapter a set I of incident types is represented as follows:

$$I = \{i_k | i_k, \text{ is a type of an incident, } k \in N\} \quad (6.3)$$

Each incident type can be associated with a subset of network connection attributes. Network connection attributes include the source *source port*, *destination port* and *protocol*. For instance, network connections related to a denial-of-service incident can be characterised by ICMP network protocol. The set of attributes, A , is represented as follows:

$$A = \{a_i | a_i, \text{ is a connection attribute, } i \in N\} \quad (6.4)$$

Each attribute from set A can take any value from a set of values, set V , in Equation 6.5. The values can either be discrete values or continuous values. For example, distance in meters would comprise continuous values (e.g. 4.345 meters), while if a number of hops from incident scene to the remote host are used to represent distance, the distance values would be discrete (e.g. 10 hops). A union of the sets of attribute values is represented by Equation 6.5.

$$V = \bigcup_i^n V_i = \{x | x \in V_i, i \in N\} \quad (6.5)$$

For instance, let $V_1 \in V$ represent a set of possible values for a distance between hosts. If the distance between hosts as were represented as a number of hops, V_1 would be a set of discrete natural numbers e.g. $V_1 = \{0, 1, 2, 3, \dots, n\}$. The last set used in modelling the incident scene is the set of connections, C . The set is represented in Equation 6.6:

$$C = \{c_i | c_i, \text{ is a network connection and } i \in N\} \quad (6.6)$$

Each connection will have a subset of attributes from set A in Equation 6.4. Following the formal representation of the incident scene by equations 6.1 through 6.6, the composition of functions that compute a set of prioritised hosts to be investigated can now be presented. Details on determining such hosts are given by the functions and presented in the next section.

6.3 ALGORITHM

The SLOE algorithm starts with a function that takes the incident type as an input and provides a subset of attributes relevant to the incident type reported. Usually when an incident is detected and reported during the preliminary examination of the scene (such as the incident response process), the incident type is also identified. Building the attributes set is the step that follows immediately after the incident type has been determined. The incident type helps to determine relevant connection attributes to search for in the victim host, which is the role of the function f in Equation 6.7.

$$f : I \mapsto A \quad (6.7)$$

For example, if the type of attack (I) that is being investigated is the denial of service performed through the classic ping of death [136], an attribute that would be searched for in the system would be that of protocol type from set A , in Equation 6.4. This attribute would ideally have a symbolic value of Internet Control Messaging Protocol (ICMP) from set V in Equation 6.5, i.e. $ICMP \in V$. Other attributes may also be associated with a ping of death and hence such attributes would also be considered and included in set A .

Next, to associate the attributes set A in Equation 6.4 and the connections in set C in Equation 6.6, only relations (and not functions)[46, p.101] can be used. The reason for this is that a single connection can be associated with multiple attributes of interest in the attributes set A . For example, a destination port or source port and/or connection protocol may all be attributes of interest on a single connection, given an incident type. Similarly, an attribute and its value may be associated with multiple connections in the connections set C . i.e., $f(a) = x$ and $f(a) = y$ with $x \neq y$. Since relations are not functional, they can be used in this scenario. The association with a symmetric relation R , a subset of the Cartesian product of sets A and C , can therefore be represented as follows:

$$R \subseteq A \times C \quad (6.8)$$

Two functions that move from the relation R to produce the set of ranked hosts to be investigated are then defined, namely set H . These functions are g and h . Function g has the relation R defined in Equation 6.8 as its domain and its range is

set C , i.e.

$$g : R \mapsto C \quad (6.9)$$

Furthermore, function h is a function with its domain in set C and the range in set H , i.e.

$$h : C \mapsto H \quad (6.10)$$

It is worth noting that $\forall c \in C \exists ! h \in H$. This means that there is no connection that that can be found on the incident scene that is without a remote source host or a remote destination host. The function h above simply determines remote hosts from the provided set of connections in the incident scene. The two functions in Equation 6.9 and Equation 6.10 constitute a composite function, namely:

$$h \circ g = h(g(R)) \quad (6.11)$$

Functions h and g utilise the characteristic functions k and l in Equation 6.12 and Equation 6.13 respectively in building the subsets. The characteristic functions are defined as follows:

Let $C_j \subseteq C$ where C_j is the j th subset of set C with connections relevant to an attack type. Furthermore, let set H_j be a set of remote hosts that can be considered for further investigation, i.e. $H_j \subseteq H$ and $j \in N$. Characteristic functions that determine whether a host or a connection is an element of the subsets C_j or H_j respectively can be applied. These functions are denoted by $k_{C_j}(c)$ and $l_{H_j}(h)$.

$$k_{C_j}(c) = \begin{cases} 1, & \text{if } c \in C_j \\ 0, & \text{if } c \notin C_j \end{cases} \quad (6.12)$$

Similarly,

$$l_{H_j}(h) = \begin{cases} 1, & \text{if } h \in H_j \\ 0, & \text{if } h \notin H_j \end{cases} \quad (6.13)$$

What the characteristic functions basically do is to determine whether a connection and/or a host is worth investigating further. If a connection is assigned a value of 1 in Equation 6.12, it means the connection is included in the connections of interest (set C_j) for further examination. If this is not the case, the connection is not included. Equation 6.13 then assigns the value 0.

Finally, there is a need to assign weights to each host based on factors such as the distance of the remote host away from the incident scene (locality). A remote host can be from within an organisation, from between organisations, from within the same country or from a foreign country.

Values 0 through 1 are assigned as weights to each remote host represented in set H . These weights reflect the effort (or cost) required to investigate a host, i.e.

$$D = \{d | 0 \leq d \leq 1, d \in Z\} \quad (6.14)$$

The weights are used to determine the priority ranking of a host being investigated, hence they further reduce the set of remote hosts produced by the composite function in Equation 6.11. Different techniques can be used to assign weights to a host. Based on the threshold that has been set on the farthest host that can be investigated, the weight can be determined. For example, if the farthest host that can be investigated is set to be 20 hops away, a host that is 5 hops away from the incident scene can be assigned a weight of $d = 5/20 = 0.25$. If more attributes than just distance are considered in a given incident type, a function can be used to determine the aggregated weight of a remote host. A set of such attribute value pairs constitute a profile of the remote host. Set P_r in Equation 6.15 is used to represent the profile of a remote host.

$$P_r = \{z | z = xRy\} \quad (6.15)$$

Equation 6.15 means that P_r is a set of z where z is an attribute value pair. Attribute x has a value y . Set P_r therefore becomes a domain of function j that computes the weight of a host (See Equation 6.16). w in Equation 6.16 represents the computed weight and its value is in D in Equation 6.14:

$$j : P_r \mapsto w \quad (6.16)$$

where,

$$w \in D \quad (6.17)$$

With the incident scene and the SLOE algorithm as presented above, the model can be implemented and the implementation is dealt with in Chapter 8.

The most critical part of the SLOE algorithm is the final stage where weights are assigned to remote hosts. In this thesis weights are assigned based on the remote

host's location (distance) and number of connections the remote host has with the incident scene. It is assumed to be more costly to investigate a host that is too far according to Equation 6.14 than it is to investigate a host that is closer. On the other hand, a remote host that has more multiple connections to a victim host has a higher probability to be an attacker. In the selection or ranking process of the host, the SLOE algorithm needs to find a balance between distance and the number of connections from a remote host.

To summarise the SLOE algorithm, the flow chart illustrated in Figure 6.1 is used.

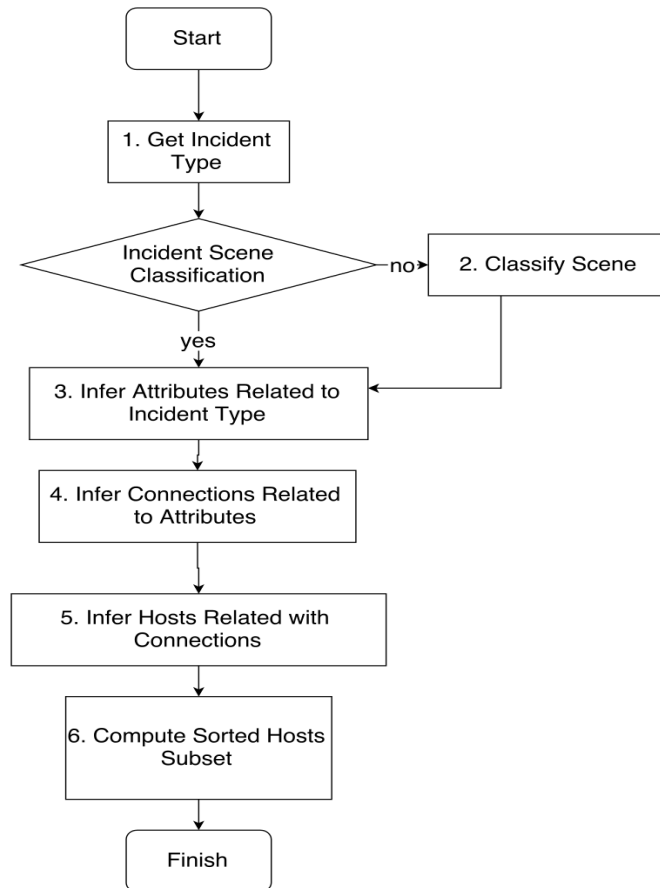


Figure 6.1: Hosts Prioritisation algorithm

The SLOE algorithm starts with an input which is an incident type. The incident type is provided during the Incident Reporting process discussed in Chapter 5. If the incident type could not be determined, an investigator is prompted to classify the incident, which means providing the type of the incident. If the incident is already classified, the SLOE algorithm proceeds. From the incident type provided, associated attributes are inferred. This process is formally represented by Equation 6.7.

Based on the attributes related to the incident scene, connections that have these attributes, with values of interest, are determined. Remote hosts that are associated with the connections are then included in set H to be ranked.

The last step is where the weights are assigned to remote hosts based on their distance from the incident scene and their number of connections with the incident scene.

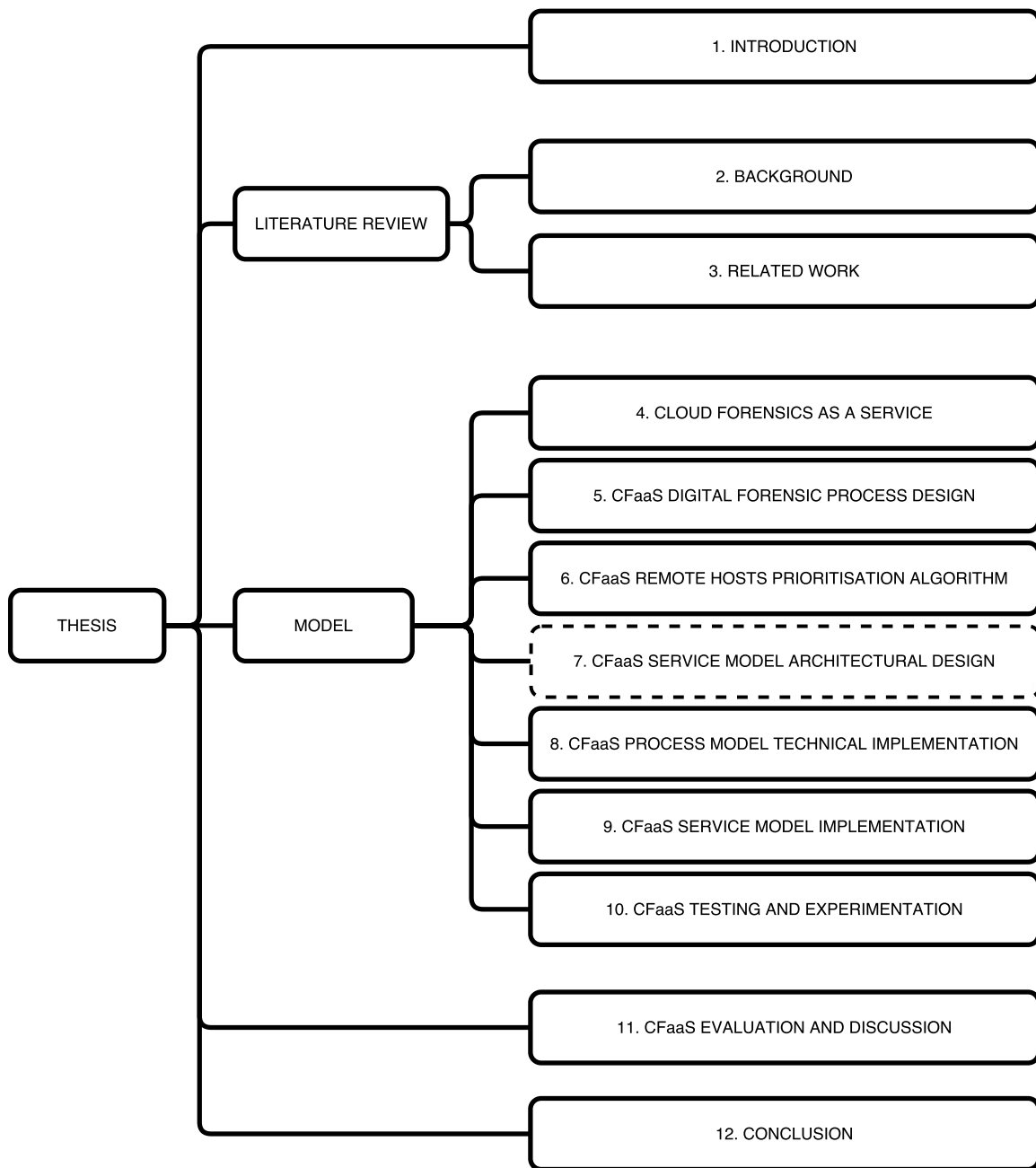
6.4 CONCLUSION

This chapter presented an approach through which remote hosts connected to an incident scene can be prioritised for further investigation. This approach can minimise the amount of time that would be required to identify remote hosts that are connected to the incident scene for further investigation. The algorithm is part of the digital forensic processes in Chapter 5, and is within the *Potential Evidence Identification* process.

In Chapter 7, the design of the CFaaS service model is presented.

7

CFaaS Service Model Architectural Design



7.1 INTRODUCTION

The advent of cloud computing has been mired by security concerns. Many research efforts on cloud computing have exposed security holes in the cloud and others proposed attempts to patch up those security holes. Patching security holes, however, does not completely eliminate security threats, as new holes are continuously discovered. The cloud is indeed insecure, and more work needs to be done for a secure cloud to be realised. It is for this reason that digital forensic investigations will always be

required. As traditional digital forensic tools and processes cannot be applied directly in the cloud, there is a need for new solutions. This chapter presents the architectural design of the CFaaS service model that is a solution introduced in Chapter 4.

Section 4.2.2 presented the requirements of a cloud forensic service that contributes towards addressing digital forensic issues in cloud environments. By addressing these requirements, a way is paved for successful convictions in criminal cases that occur in cloud environments. Chapter 7 explains how CFaaS is designed in a way that addresses the requirements presented in Section 4.2.2. Section 7.2 is devoted to the CFaaS service model architectural design. Section 7.3 discusses how the architecture addresses digital forensic system requirements and Section 7.4 concludes the chapter.

7.2 CFAAS SERVICE MODEL ARCHITECTURE

Chapter 7 addresses the research question in Chapter 1 (see Section 1.3) dealing with the design of a cloud forensic system, namely:

“How can a cloud forensic system be designed?”

This section presents the visual design of the CFaaS service model architecture. For this purpose the 4+1 model [69], a commonly used standard way of visually presenting a system design is used in a way that can be understood by all stakeholders (including end users, system analysts, sponsors and developers). The four views of the 4+1 view model are the logical view, physical view, development view and process view. The ‘plus one’ view is the scenario view. A logical view depicts the functionality that a system provides to end users and includes class diagrams, data flow diagrams and sequence diagrams.

The physical view includes a deployment diagram and is concerned with the depiction of a system from a system engineer’s perspective. The development view provides a view as perceived by the developers and includes component diagrams and the package diagrams. The process view addresses the dynamic aspects of a system such as communications among components and it consists of an activity diagram. The scenario view comprises a use case diagram.

Each of these views are discussed in detail in the subsequent subsections based on CFaaS, with the exception of the logical view which is discussed in Section B.1 of the appendix to this thesis. The other four views of the architecture presented in this section provide sufficient CFaaS service model architecture information required for the purpose of the thesis.

7.2.1 LOGICAL VIEW

The logical view diagrams presented in this section provide a logical view of the processes that are executed by CFaaS and messages that are transmitted among the processes. There is a direct mapping between the digital forensic processes that are presented in Chapter 5 and the CFaaS system processes that are represented in the logical views portrayed in this section. These diagrams partly show how CFaaS implements a standard digital forensic process as a requirement.

For the logical view this section discusses a context diagram, Diagram 0 and its child diagrams. A context diagram is “the highest level in a data flow diagram and contains only one process representing the entire system” [67]. The other diagrams that will be discussed in this section are descendants of the context diagram, namely Diagram 0 and child diagrams.

Figure 7.1 represents the context diagram of CFaaS. In the diagram there are four entities that interact, i.e. investigator, CFaaS (cloud forensic service), *Cloud-Based Instance* and the *Evidence Storage Server*. The investigator in this case is the user of the digital forensic service (CFaaS).

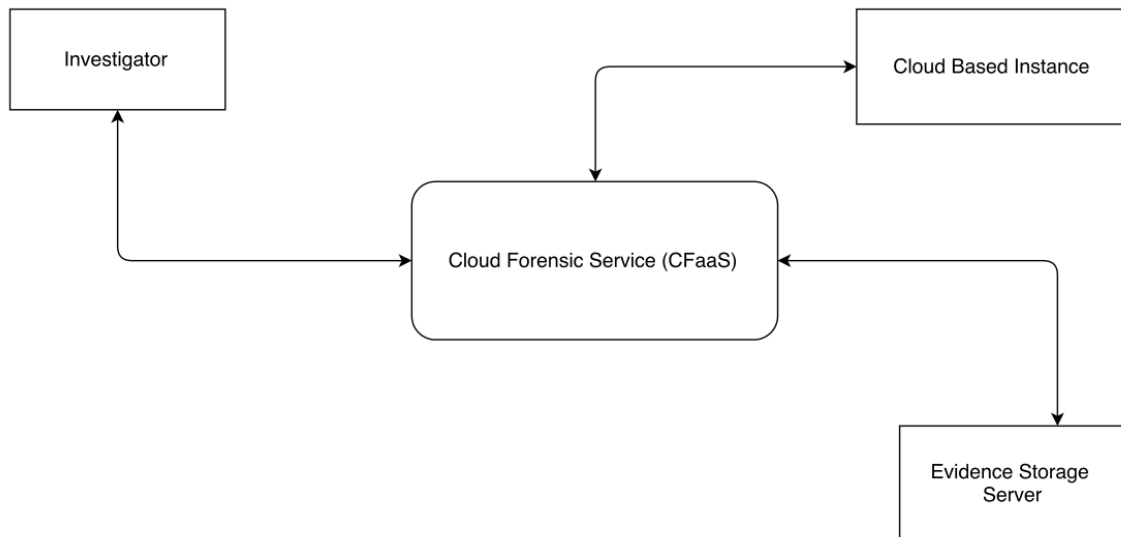


Figure 7.1: Context Diagram

Messages exchanged between the *CFaaS Service* and the cloud-based instance (incident scene) include evidence data and investigative commands. Messages between the *CFaaS Service* and the *Evidence Storage Server* are evidence data as well as the evidence retrieval and evidence storage commands. A context diagram in essence is a high-level view of the data flow diagram and it gets exploded into Diagram 0, and

further into child diagrams. The child diagram represents levels of granularity of the data flow diagram. To explain the ancestry phenomenon in these diagrams Figure 7.2 is used in which diagram relationships from Diagram 0 (level 0) to two descendants (level 2) are depicted.

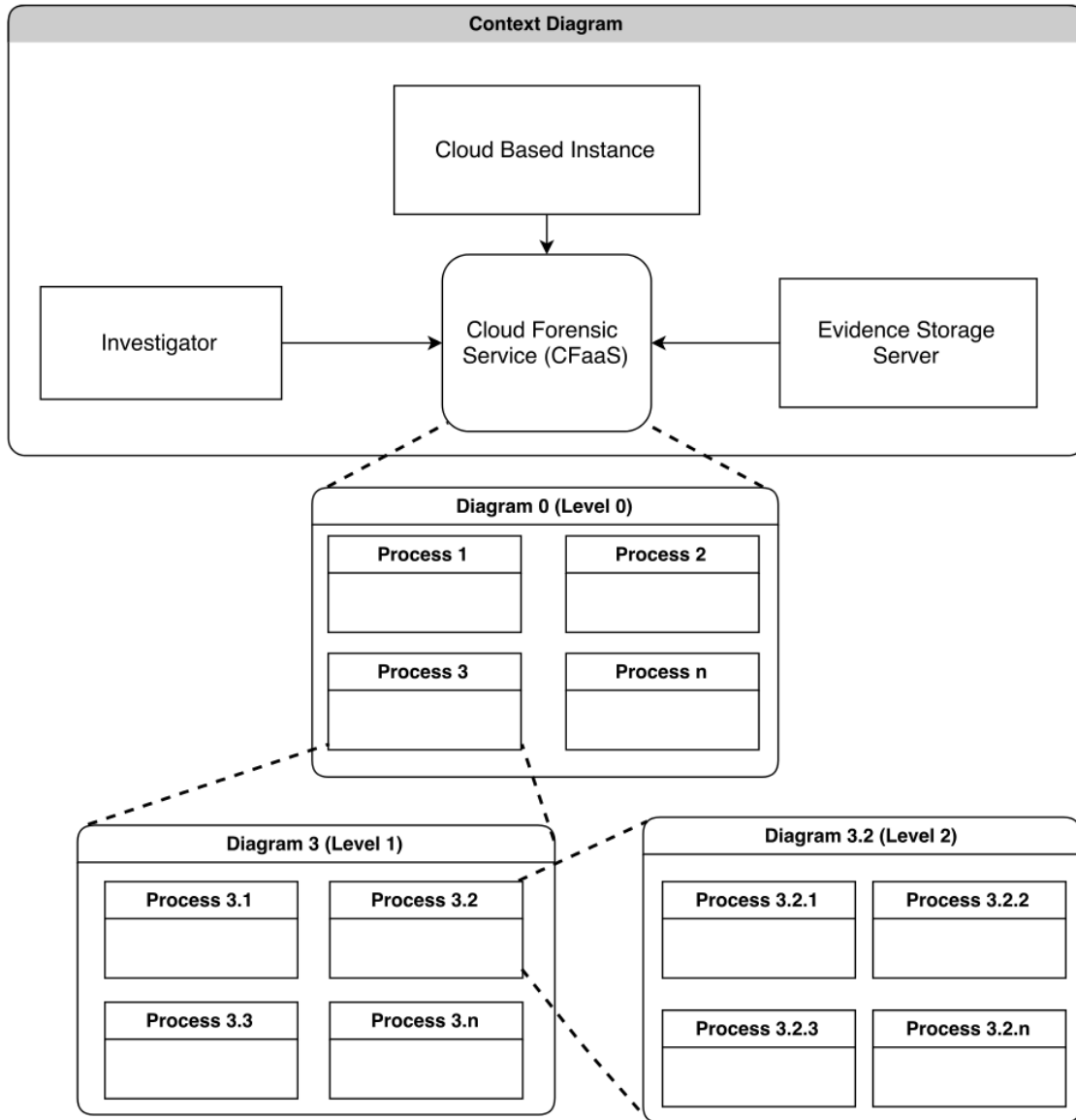


Figure 7.2: CFaaS logical diagrams relationships

Diagram 0 contains major processes in the *CFaaS service*, namely Process 1 to Process n . Each major process has internal processes. In Figure 7.2, for example, Process 3 is a major process of CFaaS contained in Diagram 0. Process 3 has internal processes which are contained in Process 3's corresponding diagram named Diagram 3 on level 1. The processes in Diagram 3 also have their respective internal processes,

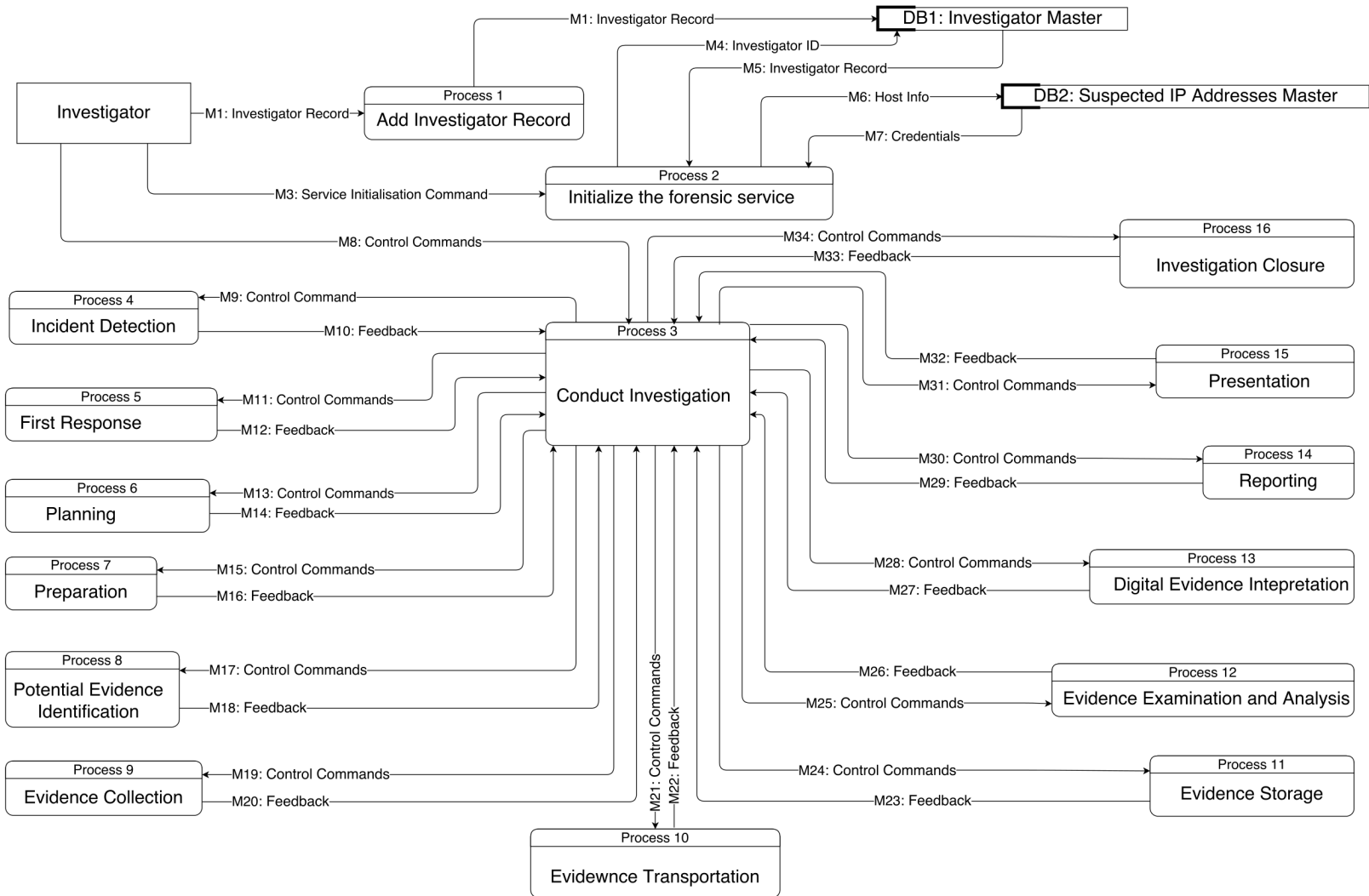
namely Process 3.1 through Process 3.n. An example is Process 3.2 whose corresponding diagram is Diagram 3.2 (on level 2). Inside Diagram 3.2, internal processes of Process 3.2 are represented, namely Process 3.2.1 through Process 3.2.n. All other diagrams that will be discussed in this section will be interpreted in a similar way.

For the purpose of this thesis, however, child diagrams are presented only up to level 1. All level 1 child diagrams can be found in the appendix of this thesis (see Section B.1) and the reader can safely choose to skip them since diagrams presented in this section are sufficient for the understanding of the child diagram concept specific to CFaaS. A single child diagram based on level 2 is however also presented in this section. This diagram corresponds with the *Determine Remote Hosts* and it is presented due to the significance of the *Determine Remote Hosts* process in this thesis.

In the next two subsections, Diagram 0 and the *Determine Remote Hosts* child diagram are discussed.

7.2.1.1 DIAGRAM ZERO

Figure 7.3 represents Diagram 0 of CFaaS, which is an exploded form of the context diagram in Figure 7.1. Diagram 0 is also a data flow diagram. Figure 7.3 in this case depicts processes that emits and consume messages as depicted in the figure.



85

Figure 7.3: CFaaS data flow diagram: Diagram 0

Sixteen main processes of CFaaS are depicted in Diagram 0. Among these processes, a number of messages are transmitted. The messages are labelled M1 through M34 in their transmission order. DB1, DB2, DB3, etc. denoted with the symbol, DB: name, represent databases of the items stated on the labels. Processes represented in Figure 7.3 are the following:

- Process 1: *Add Investigator Record*,
- Process 2: *Initialise the forensic service*,
- Process 3: *Conduct Investigation*,
- Process 4: *Incident Detection*,
- Process 5: *First Response*,
- Process 6: *Planning*,
- Process 7: *Preparation*,
- Process 8: *Potential Evidence Identification*,
- Process 9: *Evidence Collection*,
- Process 10: *Evidence Transportation*,
- Process 11: *Evidence Storage*,
- Process 12: *Evidence Examination and Analysis*,
- Process 13: *Digital Evidence Interpretation*,
- Process 14: *Reporting*,
- Process 15: *Presentation*,
- Process 16: *Investigation Closure*

Adding Investigator Record (Process 1 in Figure 7.3) is the process involving the investigator's registration to the digital forensic service. Investigator registration is a requirement, as investigators need to be authenticated to use the service for accountability purposes. Initialising a digital forensic service is the actual initiation of an investigation. Creating an account on the digital forensic service does not necessarily launch an investigation process. It is the *Initialise the Forensic Service*

(Process 2) that starts an investigation process and it includes registering a case to be investigated, which is always prompted by the detection of an incident.

Process 3 (*Conduct Investigation*) involves the actual carrying out of the digital forensic investigation. This process takes place after the investigator has registered with the digital forensic service, the digital forensic service has been initialised and an incident has occurred.

The remainder of the processes in Figure 7.3 labelled Process 4 (*Incident Detection*) through Process 16 (Investigation Closure) are implementations of the standard thirteen processes that were presented in Chapter 5 and hence they are not discussed again in this chapter. Instead, further discussion on them is covered when their implementation is discussed in Chapter 8.

In the next subsection, an example of a level 2 child diagram is presented.

7.2.1.2 LEVEL 2 CHILD DIAGRAM - DETERMINE REMOTE HOSTS

One of the important contributions of this thesis is the SLOE algorithm to determine remote hosts for further investigation (see Chapter 6). This task was referred to in the proposed standardised process model as the *Determine Remote Hosts* process. Due to its significance in this thesis, the *Determine Remote Hosts* process is used as an example for level 2 child diagrams. The process was introduced in Chapter 5 (Figure 5.6) and the corresponding algorithm was discussed in detail in Chapter 6. To illustrate level two granularity of Diagram 0, a diagram that corresponds with this process is presented — namely Diagram 8.5 derived from Process 8.5 in Figure B.6. The data flow diagram for this process is subsequently presented in Figure 7.4.

Figure 7.4 is an example of a Level 2 granularity of Process 8 of the Diagram 0 in Figure 7.3. For the convenience of the reader to see where this figure fits in in Level 1 granularity, a Level 1 granularity diagram (an exploded view of Process 8) can be seen in Figure B.6 (see Appendix B). The processes in Figure 7.4 use three data sources labelled DB1, DB2 and DB3. These data sources contain *Connection Attributes*, *Incident Type classification* and *IP Address-credentials* pairs respectively. The connection attributes in DB1 are known attribute types that are used to characterise a connection, such as connection protocol, server error rate, etc. [3]. Incident types in DB2 include unauthorised remote access to local host, unauthorised access to escalated privileges, denial of service and probing or surveillance [3]. The two data sources are utilised by Process 8.5.1 (Infer Incident Connection Attributes) in the form of Message M1 and Message M2 respectively. Given the incident type communicated through Message M2, Process 8.5.1 determines a sub-set of attributes from

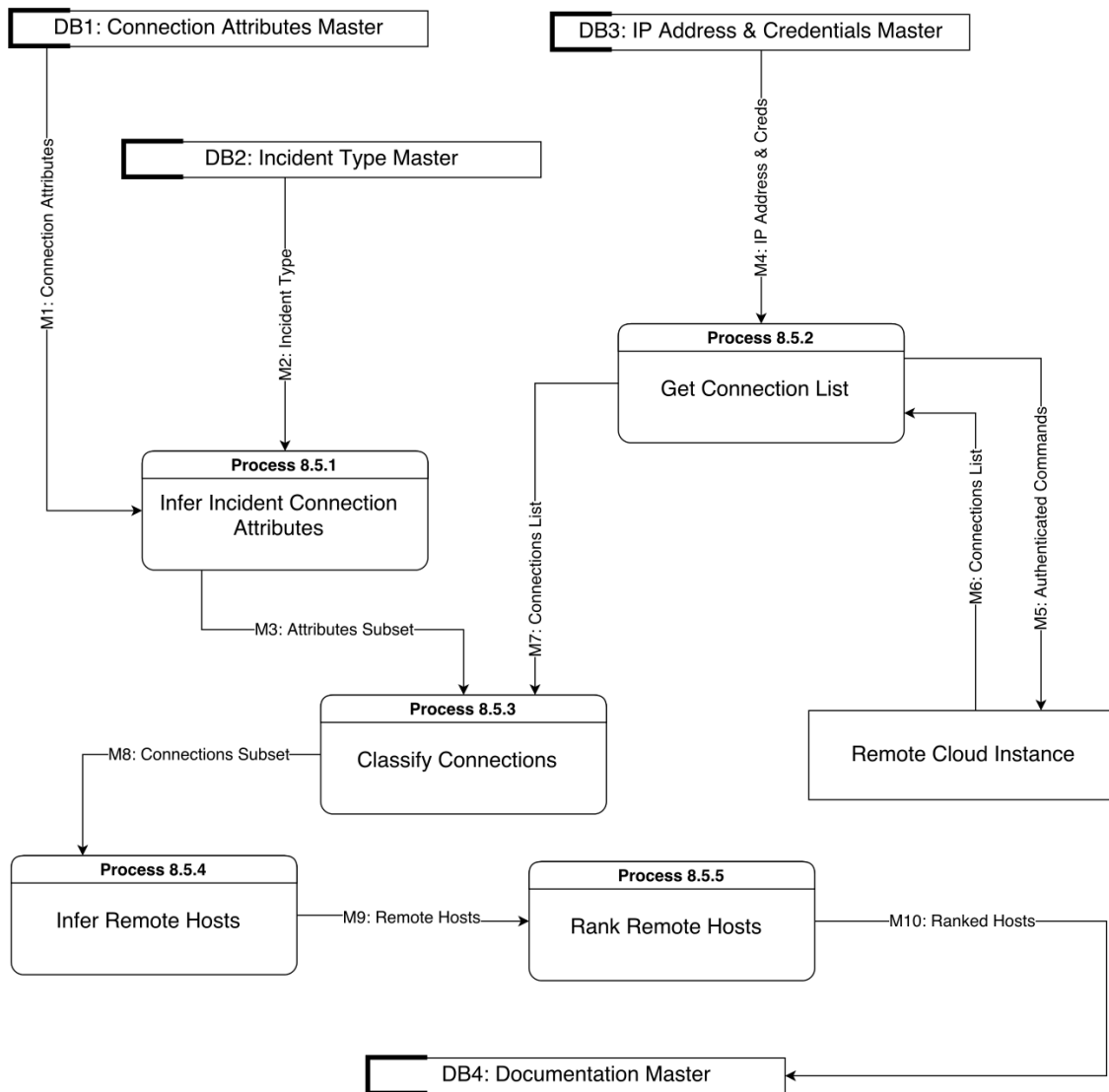


Figure 7.4: CFaaS Data Flow Diagram: Child Diagram 8.5 - Determine Remote Hosts.

Message M1, which characterises the incident type as discussed in Chapter 6. This sub-set of attributes is sent to Process 8.5.3 as Message M3.

Before Process 8.5.3 can proceed, it requires Message M3 and Message M7. Message M4 is an IP Address-credentials pair which is to be utilised by Process 8.5.2. Process 8.5.2 uses the provided credentials to connect to the provided IP Address of the incident scene and lists network connections on the host through the command in Message M5. The list of connections (Message M6) is forwarded by Process 8.5.2 onwards to Process 8.5.3 as Message M7. Process 8.5.3 filters the process list based on attributes in Message M3 and their values in each of the connections in Message M7. The filtered connections are then sent to Process 8.5.4 as Message M8 a subset

of network connections as Message M7.

In Process 8.5.4, remote hosts associated with the connections in Message M8 are extracted from each network connection. The researcher refers to this process as inference of the hosts. The hosts are subsequently ranked in Process 8.5.5 and sent to persistent storage as Message M10.

The next section presents the physical view of the CFaaS design, as well as components of CFaaS that implement the processes presented in this section.

7.2.2 CFAAS PHYSICAL VIEW

A conceptual architecture representing the proposed digital forensic service model for cloud environments, CFaaS, is at issue in this section. Unlike the logical view that provides logical views on processes, the physical view has a view or representation of the components and their arrangement in the CFaaS system. The different components address different requirements that were presented in Chapter 4. A discussion that maps the presented components with attributes that they address is provided later in Table 7.1 (see Section 7.3).

In the subsequent subsections, the components of the conceptual architecture depicted in Figure 7.5 are discussed. These components are arranged in such a way that each component is placed adjacent to components that it directly communicates with. The communication among the components is covered during the discussion and also later in the Process View discussion in Section 7.2.4. The communication can again be seen in the Logical view discussion in Section B.1 of the appendix.

In the discussion, the components are grouped into main components of CFaaS, which are *Hardware*, *Hypervisor*, *Cloud Software*, *Platform*, *Identity and Security Management*, *CFaaS Task Server* and *CFaaS Service*. The discussion of the conceptual model in Figure 7.5 takes a bottom-up approach, starting from *Hardware* through to *CFaaS Service*. There are three components that can optionally be hosted off-site by trusted IaaS providers. The components are the *Hardware* component, *Hypervisor* component and *Cloud Software* component which are represented with broken line in Figure 7.5. The three components form the Infrastructure-as-a-Service (IaaS) part of the *CFaaS Service*. These components are not necessarily unique to CFaaS. They are, however, made part of the architecture for completeness of CFaaS as a cloud service. The next section discusses the *Hardware* component, which is the bottom part of the architecture on which all other virtual components are hosted.

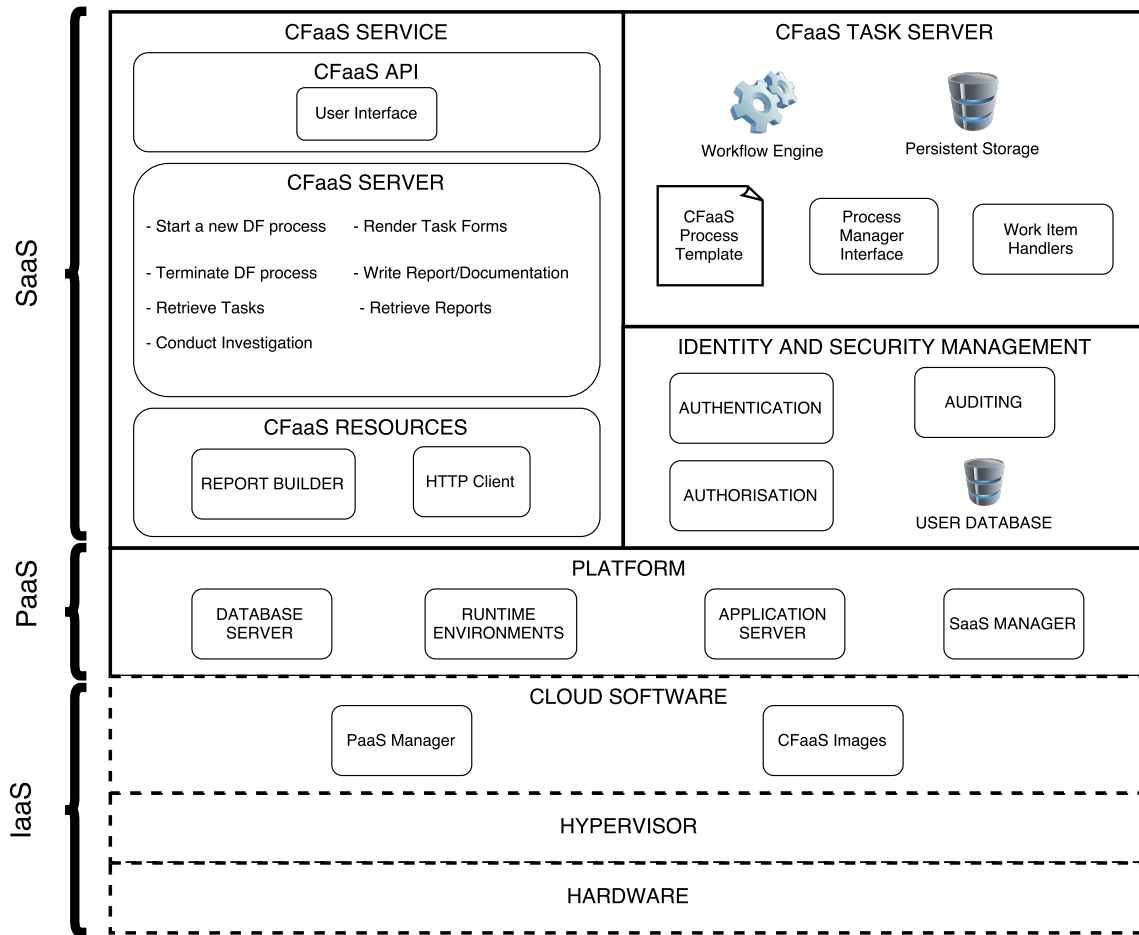


Figure 7.5: CFaaS Conceptual Model

7.2.2.1 HARDWARE

The *Hardware* component represents the data centre that hosts the digital forensic service and includes hardware resources such as storage devices, RAM, CPU, physical network devices, etc. This infrastructure needs to be hosted by the investigation agency or by a trusted IaaS provider. Having complete control of the *Hardware* component partially addresses the security requirement in the digital forensic service. However, this approach only provides certain layers of security, namely the physical, perimeter, internal network and host security layers [16, 102]. Other security layers - application and data security - are addressed by additional components discussed in this section such as *Identity and Security Management*. Storage servers constitute one of the crucial components of the hardware layer as this is where evidence and investigation reports are physically kept. It is recommended that the physical infrastructure be under the control of the investigation agency that can ensure its security. This can be achieved by means of a private cloud.

The next section deals with the *Hypervisor* component that manages and exposes the hardware resources (from a single host) as presented in this section to other virtual components.

7.2.2.2 HYPERVISOR

The *Hypervisor* component enables virtualisation and all functions of the cloud computing software installed in it for cloud services provision. A hypervisor runs on a host-operating system that runs directly on the hardware. Each physical host and even virtual hosts in a data centre have an instance of this component running on it. The hypervisor is then used to manage or virtualise hardware resources and to create virtual machines that utilise the virtual hardware resources [135]. Some attacks target this part of a cloud infrastructure for instance by redirecting data flows using firewall ports and hooking system library calls [135, 144]. In the case of CFaaS and any other cloud infrastructure, adversaries could as well take control of the entire cloud service stack if they were to gain control of the *Hypervisor* component. It is therefore essential that security in this component be prioritised.

In the next section, the *Cloud Software* component enables integration and sharing of hardware resources exposed by individual hypervisor instances from different hosts.

7.2.2.3 CLOUD SOFTWARE

This is the highest component on the component stack that is part of an IaaS and as a result, the highest component that can optionally be managed by a third party. Beyond this component, the components have to be hosted on-site or in a private cloud infrastructure. The *Cloud Software* component is used to manage IaaS nodes and to allocate virtual instances to computing nodes. The cloud software communicates directly with hypervisors running on each computing node. It comprises a built-in *PaaS manager* that handles service orchestration and starts new instances on demand from the CFaaS images that are uploaded into *PaaS manager*.

The *Hypervisor* Component that includes the virtual machines utilising the hardware resources made available by the *Cloud Software* for sharing is presented in the next section.

7.2.2.4 PLATFORM

The platform in the architecture refers to a virtual machine with required software components installed in it. In this architecture the platform needs to at least com-

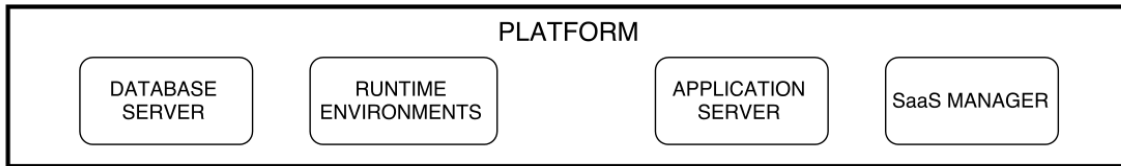


Figure 7.6: CFaaS Conceptual Model: Platform

prise a database server, a runtime environment such as Java virtual machines (JVMs), script execution environments (e.g. JavaScript), application servers and a *SaaS manager*. A digital investigation agency manages the platform. The database server is used to implement persistent storage for investigation processes. It is also used by the *Identity and Security Management* to manage users. The JVM component is used to run Java-based services that are required by the forensic services. The SaaS manager is used for the orchestration of the higher CFaaS components deployed on the platform.

The Identity and Security Management component - which provides a main security layer for the Platform, the software components adjacent and above it, and the CFaaS data - is presented next.

7.2.2.5 IDENTITY AND SECURITY MANAGEMENT

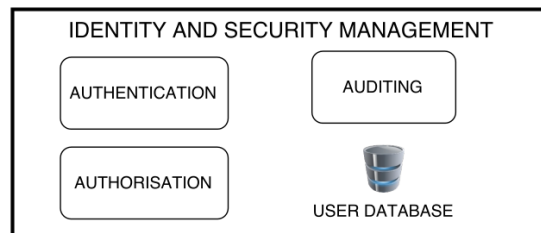


Figure 7.7: CFaaS Conceptual Model: Identity and Security Management

In Section 4.2.2.2 where non-functional requirements for a digital forensic system aimed for the cloud, security was also presented. In the description of the security requirement, security constraints were also outlined which are:

- C1: The digital forensic system needs to be protected from attacks aimed at disrupting the investigation process or tampering with evidence.
- C2: Investigation data at rest and in transit needs to be protected to avoid tampering.

C3: Only authorised users can access the investigation system and only authorised users can carry out tasks in an investigation to avoid tampering with evidence and/or an investigation process.

In order to satisfy the constraints above, the following security requirements[51] need to be met:

R1: The digital forensic system can be protected through hosting it in adequate physically protected systems.

R2: Forensic data at rest and in transit can be protected by encrypting data itself and the transmission channel such using SSL/TLS.

R3: Authorisation of a user to access the digital forensic system and also to carry out investigation tasks can be verified through authentication.

Based on the security requirements above, trust assumptions [51] of CFaaS are summarised and are listed as A1 through A8 below. Each assumption is discussed within the list in terms of its identification of the affected domain, the effect of the assumptions, description of the assumption, preconditions of the assumption, justification and the requirement that is partially satisfied by the assumption.

A1: Servers hosting the CFaaS system are placed in secure premises. This assumption affects people in general. If the premises are not secure, any person including people with malicious intent can access the physical infrastructure steal data using mountable devices and through other means. The precondition for this assumption to hold is that in must behind locked and/or with manned security. The assumption partially addresses the security requirement, R1.

A2: Private keys are managed by a competent IT administrator. This assumption affects IT administrators. Before the assumption, all IT administrators including one that do not have adequate skills to private keys can be assigned to manage the IT infrastructure. A competent IT administrator will be able to maintain best practices in all his/her activities of managing the CFaaS systems security such as IT infrastructure key management. This assumption partially addresses the protection of data in transit and at rest, requirement R2.

A3: IT administrator implements strong encryption controls for both the communication channels and the investigation servers. If this assumption does not

hold, weak encryption usage on securing data at rest and in transit would result the data being easily compromised. This assumption can only hold if assumption A2 holds. Strong encryptions lower the chances of the security of the data being breached through cryptanalysis attacks. This assumption also partially addresses requirement R2.

A4: Investigators do not share their authentication credentials with other investigators or other people involved and not involved in the investigation. Before this assumption, any investigator can access the digital forensic system by obtaining credentials from other investigators. The precondition for this assumption is that investigators are trust worthy in that they will not share their credentials with other investigators. Due to the fact that investigators are chosen based on their credibility and track record, it can be safely assumed that they will not share their credentials. This assumption ensures that only authorised investigators take part in the in the investigation and it partially addresses requirement R3.

A5: Investigators are authorised only to view data related to task they are authorised to perform. This assumption is important for audit trail. Before this assumption, investigators can access other part of the investigation that they are not assigned to. The assumption partially addresses requirement R2 and R3.

A6: Physical access if any to the IT infrastructure that hosts the CFaaS system is restricted only to authorised administrators. This assumption is dependent on assumption A1. Before this assumption, unauthorised persons can access the IT infrastructure and do malicious damage. The security personnel that control access to the infrastructure are assumed to be selected on merit. It can therefore be safely assumed that they will only grant access to the IT infrastructure premises to only authorised people. The assumption also partially addresses requirement R3.

A7: Administrators do not leak user credentials and private keys. This assumption partially addresses requirements R2 and R3. Before this assumption instead of investigators sharing their credentials, administrators would leak credentials to unauthorised users. The leaked credentials would then be used to interrupt the investigation or contaminate evidence.

A8: IT administrators restrict access to cases being investigated only to authenticated users and assigns access rights to investigators. This assumption also partially addresses requirements R2 and R3.

These assumptions to hold, a component and subsequently the satisfaction of the security requirements, a component is required that be used to implement and enforce the restrictions presented within the assumptions. The implementation and enforcement of the restrictions are enabled by the *Identity and Security Management* component implies that only authenticated and authorised persons can access a portion of the system or digital forensic evidence and the evidence data is available when needed. Evidence under the custody of an investigating agency needs to be protected at all costs from both inside and outside attacks. How data can be protected from inside attacks is, however, beyond the scope of this thesis. International security standards, such as ISO/IEC27001 (Information security management) [60] and other relevant standards need to be enforced and this is also the role of the Identity and Security Management component. The latter is implemented and treated as the trusted component [30, 71] of CFaaS. A trusted component in this context is a component that among other things enforces authentication, authorisation and accountability.

In the next section, the focus is on the *CFaaS Task Server* that enables digital forensic tasks to be executed in their standard sequence.

7.2.2.6 CFAAS TASK SERVER

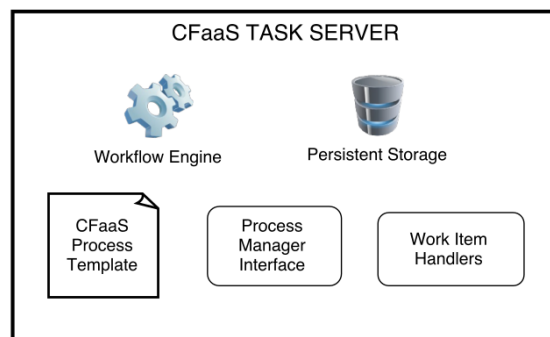


Figure 7.8: CFaaS Conceptual Model: CFaaS Task Server

A standardised forensic process in CFaaS is implemented using a business process template [6]. Procedures that are carried out are viewed as tasks to be completed. Some of the tasks need to be completed by humans while some can be performed by scripts. The *CFaaS Tasks Server* server manages both the tasks that need to

be carried out manually by investigators and those that are executed automatically by services. It comprises the *Workflow Engine*, *Persistent Storage*, *CFaaS Process Template*, *Process Manager Interface* and the *Work Item Handlers*.

The *Workflow Engine* executes script tasks and hands over manual tasks to relevant digital forensic team members. The Persistent Storage component stores the outcomes of each task executed during the investigation process. The *CFaaS Process Template* is a process definition of the workflow tasks that will be executed during an investigation.

The *CFaaS Task Server* receives calls from the *CFaaS Service*. Calls include requests to start up a new digital forensic process instance, terminating an instance, and restarting a process instance. The process manager also sends asynchronous event notifications to the *CFaaS Service* during a process execution. The events include among others entering and exiting a process instance task node, starting a process and completion of a process execution. The communication is enabled by a Representation State Transfer (REST) API¹, represented by the *Process Manager Interface* component in Figure 7.5 and Figure 7.8. Finally, the *Work Item Handlers* is a software module that implements the services that handle automated tasks in the investigation process. The interaction between an investigator and the task server component is enabled by the *CFaaS Service* component which is presented in the next section.

7.2.2.7 CFAAS SERVICE

The *CFaaS Service* component forms an integral part of the CFaaS service model architecture and is broken down into three components, namely *CFaaS API*, *CFaaS Server* and *CFaaS Resources*. The *CFaaS API* provides a means of interaction with the system for the users. It generates and renders web consoles for investigators to interact with the entire forensic service system.

The *CFaaS Server* implements the core functions of the digital forensic service. The component can make a call to the CFaaS Process Manager that creates a new process instance from a *CFaaS Process Template*. A process instance is unique to a case being investigated. While an investigation is in progress, a need may arise for it to be terminated. The server provides a means to make a call to the investigation process manager to terminate the investigation process with a provided case identification. On termination of an investigation process, the data specific to the terminated process

¹A REST API exposes functions that can be invoked remotely by HTTP clients to post data into or retrieve data from a server.

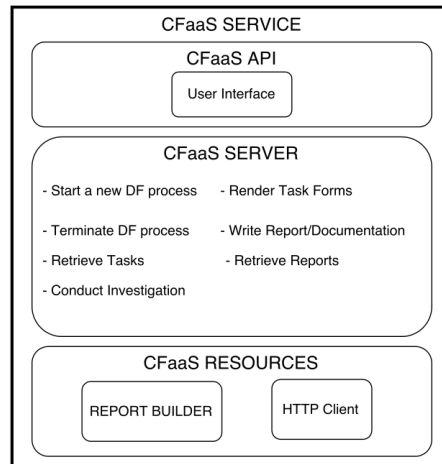


Figure 7.9: CFaaS Conceptual Model: CFaaS Service

is not cleared from the digital forensic service storage. The length of the period for which data related to a terminated investigation process of a specific case needs to be kept, would be in compliance with relevant digital forensic laws.

The server also provides a means to communicate with the CFaaS task service to retrieve tasks to be rendered to an investigator. Finally, the server implements a function to write documentation and reports and also a function to retrieve those reports for viewing and printing.

The *CFaaS Resources* component provides communication between the *CFaaS Service* component of CFaaS with both local services and remote services. It handles the transfer of concrete files such as evidence files and reports and encompasses a report builder and HTTP client components. The report builder compiles reports from all events and outcomes that were logged by the *CFaaS Task Server* in the database. The HTTP client implements the remote calls functionality to remote Transmission Control Protocol (TCP) services.

The next section deals with the development view of CFaaS design.

7.2.3 SCENARIO VIEW

The scenario view of CFaaS, also known as the use case view, is at issue here. The scenario view shows the different needs of an investigator that can be satisfied by CFaaS. Most of the use cases are the standardised investigation processes presented in Chapter 5. The scenario view in this case shows how CFaaS implements a standardised process and how it can assist an investigator throughout the investigation process. The presentation of the view is through a use case diagram in Figure 7.10

where the diagram depicts three participants in the system, namely the Investigator, *Persistent Storage Server* and the Remote Instance that is being investigated.

Three use cases for an investigator are presented in Figure 7.10, namely *Conduct Investigation*, Register and Initialise Forensic Service. The *Conduct Investigation* use case includes 14 other use cases which can be mapped to the 13 processes from ISO/IEC27043 presented in Figure 5.1 (with the exception of the Retrieve Evidence use case). These processes are the use cases in which an investigator will interact with the system while conducting an investigation.

One of the processes presented as a use case in the use case diagram is Analyse Evidence, which is short for the Evidence Examination and Analysis process as named in Figure 5.1. The name is shortened for clarity of the use case diagram in Figure 7.10. The Analyse Evidence use case is further extended by twenty four other use cases as shown in Figure 7.10. These use cases include processes or procedures carried out under the Evidence Examination and Analysis process as shown in Figure 5.10.

The remote instance represents the cloud-based incident scene or the perpetrator's cloud-based instance that is or was being used to conduct an attack. Evidence can still be acquired from the remote instance for non-live forensics, as CFaaS allows a combination of the digital forensic approaches. Even in a live digital forensics investigation, artefacts that support the investigation may still be retrieved from the instance and saved in the secure storage server for future reference.

The *Evidence Storage Server* is used to manage digital forensic data. Data includes digital evidence and other data relevant for the investigation.

The next section presents the process view of the CFaaS design. The view presented in the next section will show a sequence of interactions among the components during the communications presented in this section.

7.2.4 PROCESS VIEW

As part of process view, an activity diagram is presented that covers the use cases in Figure 7.10 (see Section 7.2.3). The process view provides a graphical view of the sequence of activities and the components that become involved while CFaaS fulfils an investigator's use case scenario. The activity diagram in Figure 7.11 is for the *Conduct Investigation* use case and the components that interact in the activity are those presented under physical view in Section 7.2.2. The researcher acknowledges that, in principle, each use case in Figure 7.10 needs to have its own activity diagram. However, this is the only activity diagram that is presented in this section due to a couple of reasons. First, it serves as an example that shows interactions of major

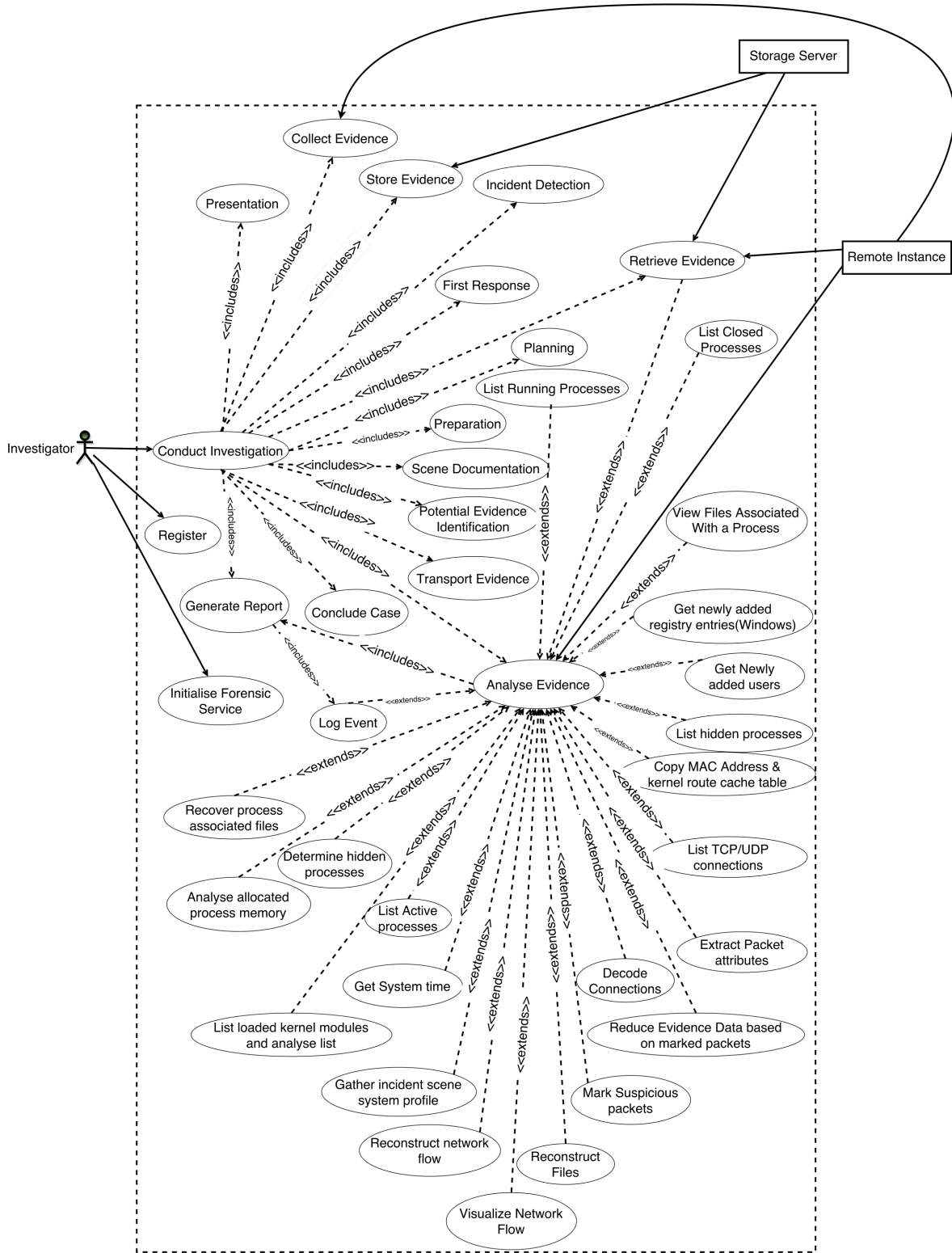


Figure 7.10: Scenario View: CFaaS Use Cases

components in CFaaS in totality. Secondly, all other use cases are trivial as they all involve automated tasks and manual tasks. How automated tasks and manual tasks are handled by CFaaS is covered during the discussion of Figure 7.11.

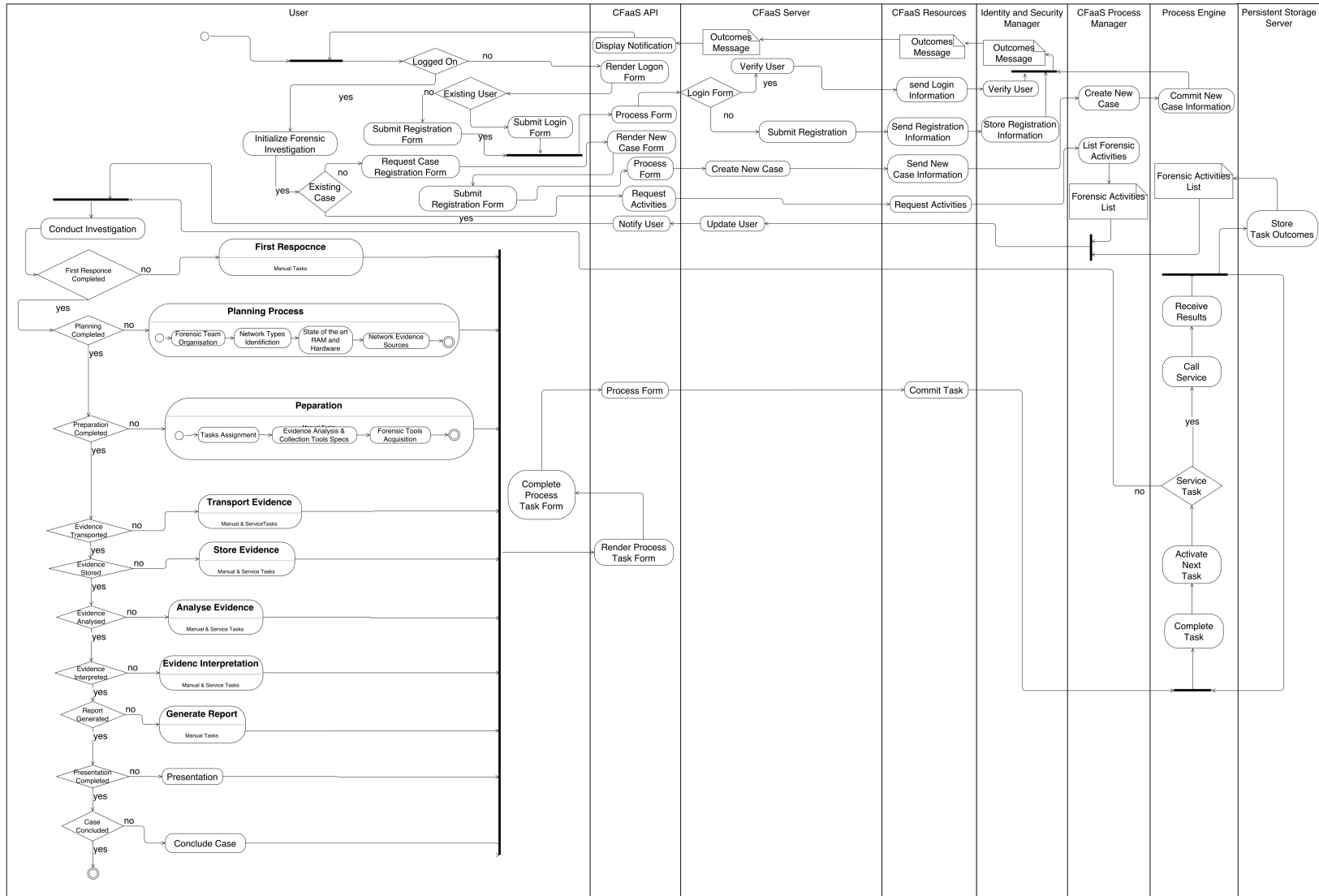


Figure 7.11: Process View: Conduct Investigation Activity Diagram

To start the *Conduct Investigation* use case, there are two scenarios that can be handled by the system. In the first scenario an incident has been detected and an investigator wants to utilise CFaaS in conducting the investigation, but he/she is not registered with the service. The second scenario is where an investigator is already registered with CFaaS and there is a new incident that needs to be investigated. The activity diagram in Figure 7.11 accommodates both scenarios by allowing an investigator to go through the registration process even if he/she does not already own an account with CFaaS.

The initiator of the system activity in Figure 7.11 is the user (investigator). The user is an investigator who intends to conduct a cloud forensic investigation using CFaaS. An investigator initiates the process by visiting the CFaaS web page. If the investigator is already signed in, they can Initialise Forensic Investigation. If the investigator is not already signed in, the system performs the function Render Login Form through the *CFaaS API*.

If an investigator is not registered in the system, he/she requests a registration form, completes and submits it, after which the *CFaaS API* processes the submitted form. If the information being submitted involves login credentials, the credentials are verified inside the *CFaaS Server*. The verification process involves sending the credentials to the *Identity and Security Management* component via the *CFaaS Resources* component from the *CFaaS Server* component. The *Identity and Security Management* verifies the user credentials and sends the outcomes back to the user. If the information being submitted is registration information, it is sent to the *Identity and Security Management* component via the *CFaaS Resources* component. The *Identity and Security Management* component stores the registration information and sends the outcomes back to the user. The outcomes messages are routed via the *CFaaS Resources* and the *CFaaS Server* and they are displayed by the *CFaaS API*.

If the investigator is already signed in, he/she can perform the function Initialise Forensic Investigation, and if he/she wants to start a new investigation case, the next step is Request New Case Registration Form. Using the Render New Case Form function, the *CFaaS API* component renders the case registration form to the investigator. The investigator completes and submits Case Registration Form, which is subsequently processed by the *CFaaS API*. The *CFaaS Server* sends the new case information via the *CFaaS Resources* to the CFaaS Process Manager which creates a new case process instance. The Process Engine executes the new case process. To investigate an existing case, the investigator selects a case that they intend to begin investigating. The *CFaaS API* queries a list of current tasks that the investigator is

authorised to view or execute. The *CFaaS Resources* component sends the request for the list of tasks to the CFaaS Process Manager who responds with the list of tasks. If an investigation into the selected case has not commenced, the Register Case manual task will be at the top of the task list.

To complete the Register Case task which is a manual task, a task form is rendered to the investigator by the *CFaaS API* component. On submission of the completed task form, the *CFaaS API* component processes it and commits the investigator's inputs by using the *CFaaS Resources* component. The latter calls the Process Engine component to complete the task, and after successful completion, the Process Engine component activates the next activity in line on the investigation process.

Inside the Process Engine component, if the activated process is a script task, the Process Engine component calls the script task's corresponding service. On receipt of the service outcomes, the Process Engine component sends the results to the *CFaaS Server* component and notifies the current user. The Process Engine then completes the current script task. If the activated task is a manual task, the task gets listed on the tasks that an investigator can execute manually.

The system iterates through all the investigation processes and their sub-tasks where manual tasks are executed inside the User component and script tasks are executed within the Process Engine component. Once all investigation processes have been completed, the system exits.

This concludes the five different views of the CFaaS service model architecture. In the next section, a summary is presented on how the architecture addresses the digital forensic system requirements.

7.3 HOW THE ARCHITECTURE ADDRESSES THE REQUIREMENTS

The purpose of the CFaaS service model architecture presented in this chapter is to comply with the functional and non-functional requirements presented in Chapter 4. This section provides a summary (in the form of Table 7.1) of how the architecture addresses such requirements. The first column in Table 7.1 represents a specific requirement that is being addressed. The second column represents a component in the CFaaS service model architecture that addresses an attribute specified in the first column. The third column provides a description of the component that addresses the corresponding requirement.

Table 7.1: Requirements satisfaction

REQUIREMENTS (in Chapter 4)	COMPONENT(S)	DESCRIPTION
Functional Requirements		
Implements a standard digital forensic process in the cloud	CFaaS Task Server	The implementation of a standardised digital forensic process is enabled by the <i>CFaaS Task Server</i> component where a standard process is implemented as a work-flow.
Aid an investigator through the standard digital forensic process in cloud	CFaaS Service & CFaaS Task Server	The workflow implementation of the standard allows investigation tasks/ processes to be rendered in their sequence as per the standard's requirement. The <i>CFaaS Service</i> component renders appropriate task forms according to their sequence to an investigator. Each task form is self-explanatory and if necessary, explicit task explanations are given on the task forms.
Allow collaboration among multi-jurisdictional law enforcement agencies in a cloud investigation	CFaaS Service	The <i>CFaaS Service</i> component exposes a similar version of an investigation window to collaborating investigators. The preliminary reports on all activities performed by investigators are entered and updated on a single report.
Semi-automated	CFaaS Task Server	The CFaaS Task Server executes automated tasks, in other words script tasks and service tasks. All the automated tasks utilise data inputs provided by investigators while performing other manual tasks.
Flexibility	CFaaS Task Server	The CFaaS Task Server allows replacements of the process templates with updated versions or completely new investigation process definition templates.
Ease of Use and Efficiency	CFaaS Service	The user interfaces exposed by the CFaaS Service component's API are self-explanatory and subsequent tasks are rendered to an authorised investigator automatically.
Non-functional requirements		
Scalability	Hardware, Cloud Software and Platform	The auto-scaling nature of hardware resources in the cloud partly addresses scalability in CFaaS.
Security	Identity and Security Management	The Identity and Security Manager implements the Auditability, Authentication and Authorisation mechanisms of CFaaS.
Auditability	CFaaS Service and Identity and Security Management	This component enables auditing by maintaining an audit trail of investigators who perform specific tasks during the entire investigation.
Maximum Control of the Service Stack by Investigation Agency	Hardware, Hypervisor and Platform	This is enabled by placing the three components that constitute the IaaS section under an investigation agency's complete control or making them part of a private cloud.

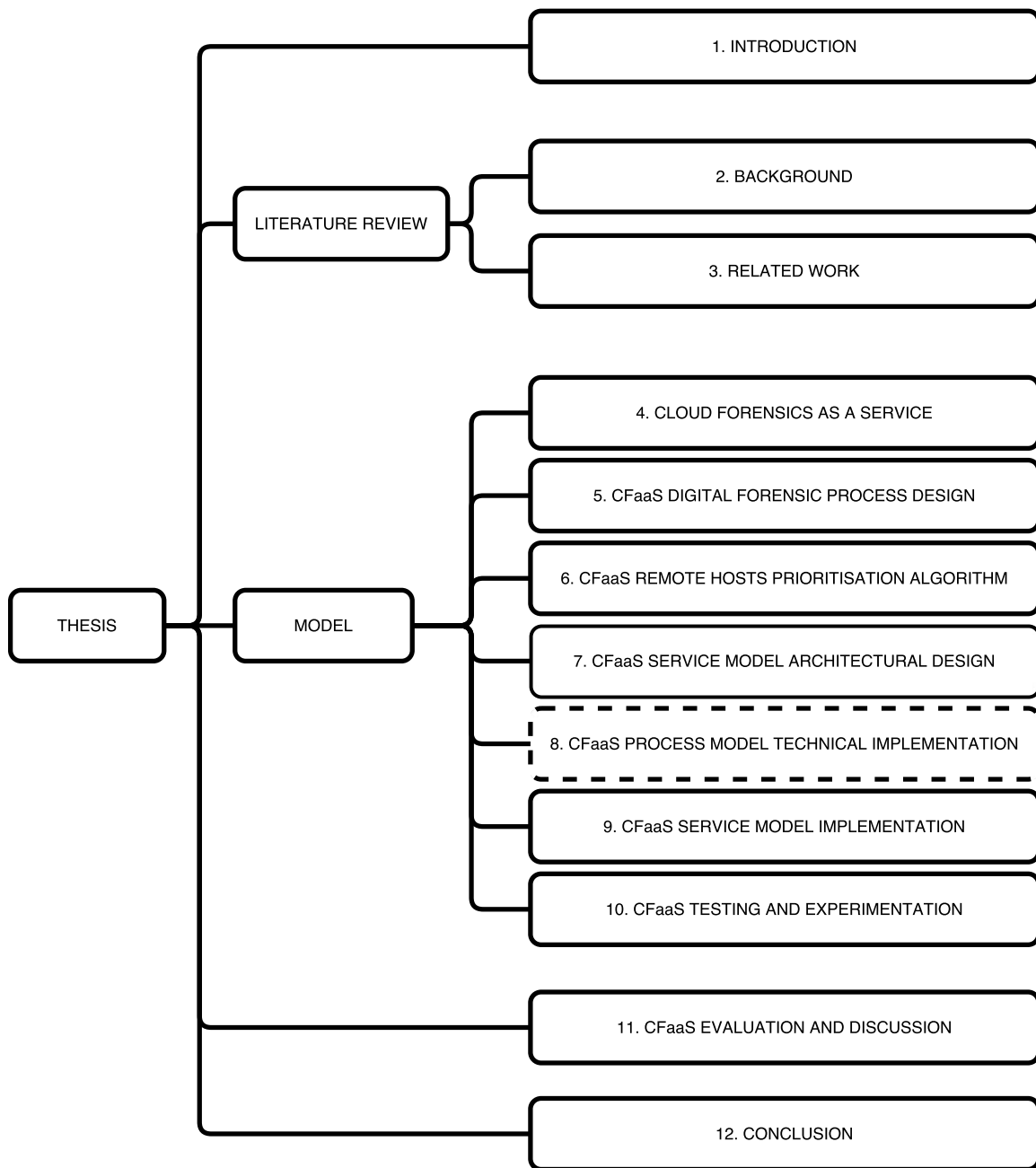
The above is a summary of how the architecture addresses the requirements presented in Chapter 4. The next section concludes the chapter.

7.4 CONCLUSION

The design of CFaaS service model architecture was discussed in this chapter. The system design addresses the functional and non-function requirements presented in Chapter 4. The design views of CFaaS were presented by using the 4+1 model view. Components of CFaaS are represented in the design view, and it is shown how the components interact to aid an investigator in conducting an investigation in a cloud environment. In Chapter 8 the researcher focuses on the implementation details of the digital forensic process model presented in Chapter 5.

8

CFaaS Process Model Technical Implementation



8.1 INTRODUCTION

In Chapter 5 the focus was on the design of a standardised digital forensic process. Concurrent processes, namely *Documentation*, Obtain Authorisation, Preserving Digital Evidence, Managing Information Flow and Interaction with Physical Investigation are also presented. In the current chapter, the implementation details of the process that was designed in Chapter 5 are presented. The implemented processes are as shown in Figure 8.1 in their sequence. The processes in Figure 8.1 are however with-

out the concurrent processes. Figure 8.1 is a more comprehensive version of Figure 5.1 and the concurrent processes are depicted in Figure 5.1.

The rest of chapter is organised as follows. Section 8.2 presents jBPM [6], a Business Process Management suite that was used to implement our digital forensic process model. Section 8.3 focuses on the individual processes and their implementation details in the sequence as they appear in Figure 8.1. Section 8.4 concludes the chapter.

8.2 JBoss jBPM

In this section, jBPM, “*a flexible Business Process Management (jBPM) Suite*” [6] is used to implement the proposed standardised digital forensic process. jBPM “*makes the bridge between business analysts and developers*” [6] and allows the implementation of a business process in a way that combines both the manual tasks referred to as human tasks and other tasks that can be executed by third party web services or custom scripts as automated tasks.

jBPM process implementation also allows for the enforcement of human task execution based on roles. These features of jBPM have made it ideal for the implementation of the standard digital forensic process presented in Chapter 5. In a digital forensic investigation team, team members are assigned to different roles. For example, the task of the handling of digital forensic evidence can be assigned to a set of individuals for the sake of the proper management of the chain of custody. This is essential in a digital forensic process; hence, jBPM is ideal for it.

Section 8.3 presents details on how each of the investigation processes presented in Chapter 5 is implemented, using jBPM.

8.3 PROCESSES IMPLEMENTATION

This section presents the implementation of the digital forensic processes and focuses on the business process tasks that are instantiations of the sub-processes tasks in Figure 8.1. For the purpose of this chapter, any staff member who becomes involved in carrying out any of the investigation processes presented hereafter will be referred to as an investigator, just with different roles. For example, the role of an investigator can be that of an office administrator who may be tasked with undertaking the registration process of a case, while individuals with technical expertise may only have to perform the evidence acquisition and evidence preservation process. In the same way, there could be many other roles that can be assumed by personnel taking

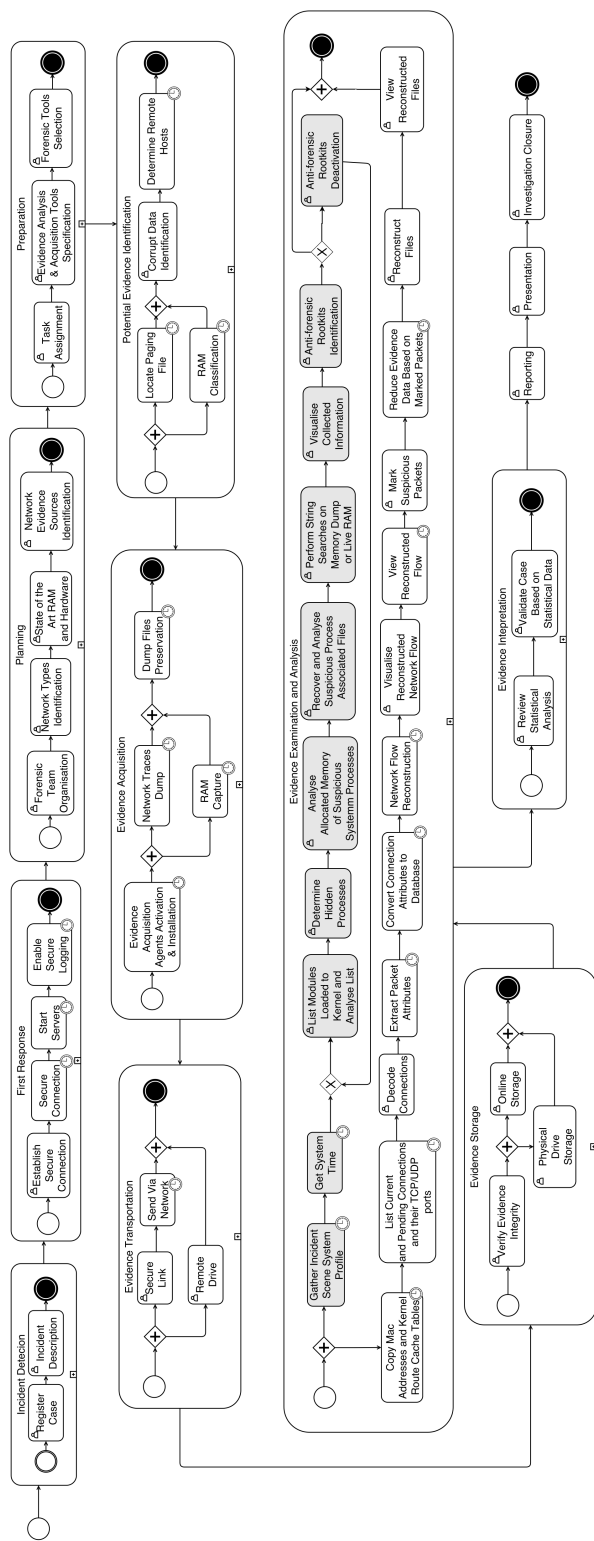


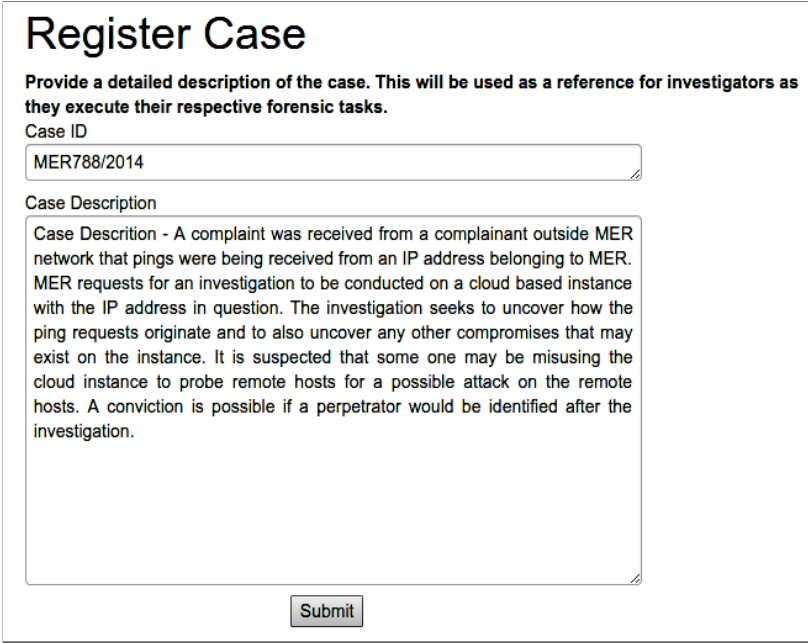
Figure 8.1: Digital Forensic Process Implementation

part in the investigation. In this chapter they are all referred to as investigators. All processes and sub-processes referred to in italics in this section are those represented in Figure 8.1.

In the subsections that follow, the implementation of each process is discussed.

8.3.1 INCIDENT DETECTION PROCESS

As can be seen from Figure 8.3, there are two sub-processes within the *Incident Detection* process namely *Register Case* and *Incident Description*. The processes are both implemented as human tasks (i.e. manual tasks) in jBPM. Figures 8.2 and 8.3 contain the task forms used by investigators to complete these human tasks. The *Register Case* process implementation takes no input variables. Its corresponding task form comprises two fields - Case ID and the Case Description. The Case ID field represents a unique human readable case identification. In the description field, the investigator enters the description of the case. The two fields become outputs of this process and are used in the documentation process that runs concurrently with all the investigation processes.



Register Case

Provide a detailed description of the case. This will be used as a reference for investigators as they execute their respective forensic tasks.

Case ID

MER788/2014

Case Description

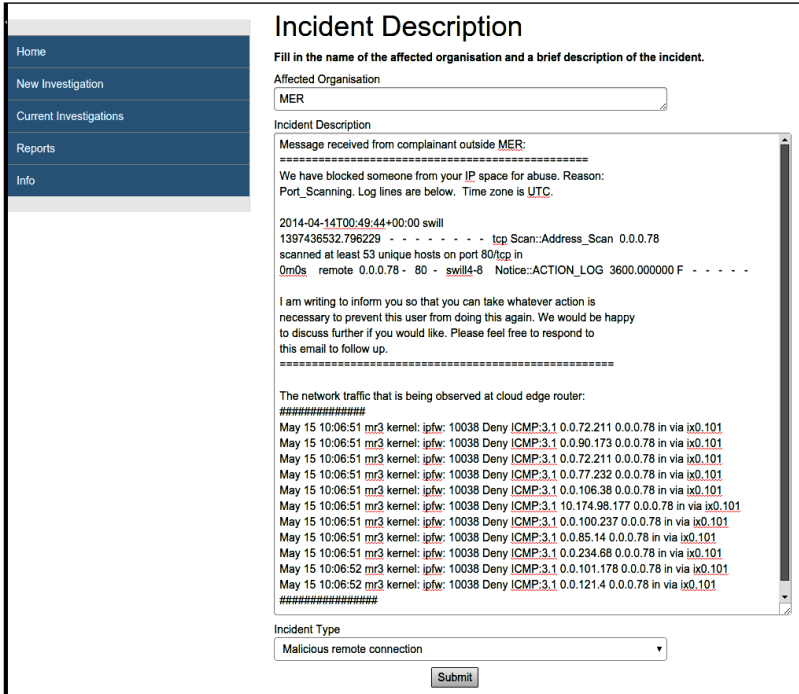
Case Description - A complaint was received from a complainant outside MER network that pings were being received from an IP address belonging to MER. MER requests for an investigation to be conducted on a cloud based instance with the IP address in question. The investigation seeks to uncover how the ping requests originate and to also uncover any other compromises that may exist on the instance. It is suspected that some one may be misusing the cloud instance to probe remote hosts for a possible attack on the remote hosts. A conviction is possible if a perpetrator would be identified after the investigation.

Submit

Figure 8.2: Case registration

The *Incident Description* process in Figure 8.3 comprises the fields used to enter the name of the organisation affected by the incident, as well as a description of the incident. The affected organisation is the organisation that owns the infrastructure affected by the incident and the organisation may be the cloud service provider that

hosts the service instance (or an owner of the instance). In this case, Figure 8.3, the incident description, includes what is being observed in the incident scene, namely abnormal activities that triggered the investigation. A description of the incident is also provided in words. These manual text inputs from investigators become outputs of the process and they are written to persistent storage to be utilised in later processes.



Incident Description

Fill in the name of the affected organisation and a brief description of the incident.

Affected Organisation

Incident Description

Message received from complainant outside MER:
 =====
 We have blocked someone from your IP space for abuse. Reason:
 Port_Scanning. Log lines are below. Time zone is UTC.

2014-04-14T00:49:44+00:00 swill
 1397436532.796229 - - - - - tcp Scan:Address_Scan 0.0.0.78
 scanned at least 53 unique hosts on port 80/tcp in
 0m0s remote 0.0.0.78 - 80 - swill4-8 Notice::ACTION_LOG 3600.000000 F - - - -

I am writing to inform you so that you can take whatever action is necessary to prevent this user from doing this again. We would be happy to discuss further if you would like. Please feel free to respond to this email to follow up.
 =====

The network traffic that is being observed at cloud edge router:
 #####
 May 15 10:06:51 m3 kernel: iptfw: 10038 Deny ICMP:3.1 0.0.72.211 0.0.0.78 in via ix0.101
 May 15 10:06:51 m3 kernel: iptfw: 10038 Deny ICMP:3.1 0.0.90.173 0.0.0.78 in via ix0.101
 May 15 10:06:51 m3 kernel: iptfw: 10038 Deny ICMP:3.1 0.0.72.211 0.0.0.78 in via ix0.101
 May 15 10:06:51 m3 kernel: iptfw: 10038 Deny ICMP:3.1 0.0.77.232 0.0.0.78 in via ix0.101
 May 15 10:06:51 m3 kernel: iptfw: 10038 Deny ICMP:3.1 0.0.106.38 0.0.0.78 in via ix0.101
 May 15 10:06:51 m3 kernel: iptfw: 10038 Deny ICMP:3.1 0.174.98.177 0.0.0.78 in via ix0.101
 May 15 10:06:51 m3 kernel: iptfw: 10038 Deny ICMP:3.1 0.0.100.237 0.0.0.78 in via ix0.101
 May 15 10:06:51 m3 kernel: iptfw: 10038 Deny ICMP:3.1 0.0.85.14 0.0.0.78 in via ix0.101
 May 15 10:06:51 m3 kernel: iptfw: 10038 Deny ICMP:3.1 0.0.234.68 0.0.0.78 in via ix0.101
 May 15 10:06:52 m3 kernel: iptfw: 10038 Deny ICMP:3.1 0.0.101.178 0.0.0.78 in via ix0.101
 May 15 10:06:52 m3 kernel: iptfw: 10038 Deny ICMP:3.1 0.0.121.4 0.0.0.78 in via ix0.101
 #####

Incident Type

Figure 8.3: Incident description

The next section discusses the process that follows the *Incident Detection* process in the sequence, namely the *First Response* process.

8.3.2 FIRST RESPONSE PROCESS

The first response process is implemented as a sub-process with three processes as can be seen in Figure 8.3: *Establish Secure Connection*, *Secure Connection*, *Start Servers* and *Enable Secure Logging*. *Establish Secure Connection* is the only process implemented as a human task and the task form that is used to complete it is shown in Figure C.1 in the Appendix. From hence forth, only a selection of task form images will be included in the chapter text. For the rest of the task form images, the reader will be referred to Appendix C at the end of the thesis. The *Establish Secure Connection* task only provides a way by means of which an investigator can provide details to be used to connect to the incident scene¹, namely IP address, username and

password. These details are stored in process variables and can be used by subsequent automated tasks to connect to the remote cloud incident scene.

The *Secure Connection* process is a script task that takes as an input the IP address, username and password that the investigator supplied during the *Establish Secure Connection* process. In the remainder of this chapter, script task and automated task are used interchangeably when referring to an investigation task that does not require human intervention. An alternative authentication method that can be used by the investigation scripts is the public key authentication method, which involves the transfer of public keys to the incident scene. The transfer of the public key to the incident scene enables subsequent investigation script tasks to authenticate with the incident scene without having to exchange keys every time the script tasks send connection requests to the incident scene.

The *Start Servers* process is implemented as a script task. The process starts up investigative software servers on the investigation cloud service instance. While starting up the servers, this process also transfers files (e.g. agents' installation files) to the incident scene that would be used for digital forensic investigation purposes. These files are to be utilised by the *Evidence Acquisition* process. In the implementation of CFaaS, the NodeJs [95], a JavaScript execution environment is used to implement Transport Layer Security (TLS) servers. If for example, the incident scene already runs NodeJs, a NodeJs TLS server source file can be transferred to and executed in the remote cloud instance. This server can then be used in transferring or acquiring evidence from the incident scene. The script checks the incident scene environment for its suitability to run specific digital forensic applications. If the environment already supports an intended application such as NodeJs, the required files are transferred to the incident scene and executed.

Installing and running custom investigation files alters or may alter evidence on the incident scene. In this research, however, the researcher is of the view that files installed by investigators have known and documented effects on a target system. With the effects of the installed files documented, evidence acquired from the scene is still admissible. An extract from a USA vs Safavein case [41] reads as follows:

“The possibility of alteration does not and cannot be the basis for excluding e-mails as unidentified or unauthenticated as a matter of course any more than it can be the rationale for excluding paper documents (and copies of those documents). We live in an age of technology and computer use where e-mail communication now is a normal and frequent fact for the majority of this nation’s population, and is of particular

importance in the professional world. The defendant is free to raise this issue with the jury and put on evidence that e-mails are capable of being altered before they are passed on. Absent specific evidence showing alteration, however, the Court will not exclude any embedded e-mails because of the mere possibility that it can be done.”

From this extract of the case it can be deduced that installing software, and hence altering files on the incident scene, would not necessarily form the basis for the rejection of evidence acquired from it. Moreover, an act of installing digital forensic software would not alter user owned files but system files instead. This applies to both the *Start Servers* and the *Enable Secure Logging* processes which are the initial processes to interact with the incident scene. These actions are likely to alter the contents of the RAM in the incident scene.

The *Enable Secure Logging* process is also implemented as a script. The task takes as inputs the IP address and the authentication credentials for the incident scene that were provided by an investigator during the *First Response* process. After successfully authenticating with the remote cloud instance, the script task copies the remote cloud instance system and application logs to preserve and store them in a secure directory. This forms part of the evidence in the Evidence Preservation process that runs concurrently with the investigation.

The next section presents the *Planning* process, which follows after the *First Response* process has been concluded.

8.3.3 PLANNING PROCESS

The *Planning* process involves only manual tasks that are implemented as human tasks (see Figure 8.3) and are as follows: *Forensic Team Organisation*, *Network Types Identification*, *State-of-the-art RAM and Hardware* and *Network Evidence Sources Identification*. The *Planning* process kick-starts with the *Forensic Team Organisation* process (see Figure C.2). The task form used to execute this task has two fields, namely the Lead Investigator and the Team Members. The list of names provided in this process will be used in the *Preparation* process when investigation tasks are to be assigned to individuals.

A *Network Types Identification* process execution task form (see Figure C.3) comprises fields in which the identified network types and fields that can be entered to provide a description of each. The listed networks include network types associated with the current cloud incident scene, as well as networks that are likely to be encountered during the investigation as the investigation expands to additional cloud-based

instances. The list of network types as output from this process becomes an input in the *Network Evidence Sources Identification* process to be discussed later.

The form used to complete the *State-of-the-art RAM and Hardware* process (Figure C.3) comprises fields that are used to provide specific Hardware technologies and RAM technologies. A description of each is also provided by an investigator through the provided fields. The information provided in this process is used in the *Potential Evidence Identification* process and can also inform the techniques applied by an investigator in undertaking subsequent processes during the investigation.

The *Network Evidence Sources Identification* process is implemented as a manual task and completed by means of a task form (see Figure C.5). In carrying out this process, an investigator investigates existing network types as listed in the *Network Types Identification* process and provides potential sources of evidence. For example, if a host being investigated is on a local area network (LAN) or wide area network (WAN) setting, pieces of evidence are obtainable from the routers. If a host being investigated accesses the Internet via a mobile network data service provider, evidence can also be obtained from that mobile data service provider. The output of this process is useful in the *Potential Evidence Identification* process.

8.3.4 PREPARATION PROCESS

The *Preparation* process consists of three processes: *Task Assignment*, *Evidence Analysis & Acquisition Tools Specification* and *Forensic Tools Selection*. The process starts with the *Task Assignment* process (see Figure C.6) which is implemented as a manual task. To complete the process, an investigator who performs this task makes use of the list of available personnel provided during the *Forensic Team Organisation* process. The tasks that are assigned to investors in this process can be other, subsequent processes to be carried out as part of the investigation or they can be other sub-tasks that support other processes. Using the task form, an investigator can enter a task name to be performed and use the drop-down list to select an investigator that will undertake the specified task.

The *Evidence Analysis & Acquisition Tools Specification* process (Figure C.7) is also implemented as a manual task. The specific tools and their specifications, which can be provided through the task form, are among the tools to be acquired in the *Forensic Tools Selection* process. The output from this process will then serve as input into the *Forensic Tools Selection* process.

The *Forensic Tools Selection* process is implemented as a manual task and can be completed by means of a task form (Figure C.8). A list and a description of each of

the acquired tools can be provided by an investigator. The tools may include both software forensic tools and hardware forensic tools.

The next section focuses on the *Potential Evidence Identification* process.

8.3.5 POTENTIAL EVIDENCE IDENTIFICATION PROCESS

The *Potential Evidence Identification* process consists of five processes, namely the *Locate Paging File*, *RAM Classification*, *Corrupt Data Identification* and *Determine Remote Hosts* processes, which are all implemented as script tasks. The *Locate Paging File* process executes the command in Listing 8.1 to obtain information about a swap space on Linux hosts, and the command in Listing 8.2 in the case of Windows-based hosts.

Listing 8.1: Swap space on Linux based hosts

```
ssh user@host swapon -s
```

Listing 8.2: Paging file on Windows Hosts

```
C:\> wmic pagefile list /format:list
```

The commands that are executed by automated tasks in this sub-section and throughout the chapter are not extensive in nature. More complex automated tasks can be implemented to carry out investigation tasks thoroughly and more efficiently. The commands provided in this chapter are for demonstration purposes and mostly Linux based. An example of a detailed procedure of how a Linux-based host can be investigated is given by Dittrich in [38] though the example does not focus solely on a live digital forensics investigation.

The *RAM Classification* process executes the command in Listing 8.3 to obtain information about the type of RAM in the system. The “dmidecode” command dumps the Desktop Management Interface table information[32]. dmidecode is not a standard Linux tool. It is therefore installed in the cloud instance being investigated along with other investigation tools that would be required for the investigation. This is done in the *Evidence Acquisition Agents Activation & Installation* process. In this thesis it is assumed that installing additional software in the cloud instance should have a very small documented memory footprint within the cloud instance so as to cause minimal contamination of potential evidence.

Listing 8.3: Obtaining RAM classification in Linux hosts

```
ssh user@host ‘‘sudo dmidecode —type memory’’
```

The *Corrupt Data Identification* process is implemented as a manual task. In this process, an investigator uses different techniques such as identifying and inspecting recently modified system files. Recently modified files are likely to include system files replaced by an attacker. For example, say an attacker has replaced the “/bin/ls” binary file in Linux with a malicious one. An investigator may use techniques to identify recently modified or created files. The “/bin/ls” file is likely to be listed among the files. Such a file can be regarded as a rootkit by an investigator. The techniques applied by the investigator are provided by the investigator by means of a task completion form (Figure C.9).

The closing process under the *Potential Evidence Identification* process is the *Determine Remote Hosts* process that was discussed thoroughly in Chapter 6. Remote hosts that have live connections to the incident scene may contain evidence. The hosts connected with the incident scene can include an attacker’s host or an enterprise’s cloud instances that are infected and may be used to conduct an attack such as the denial of service. This process applies criteria as presented in Chapter 6 to prioritise hosts for further investigation. It is implemented as a script and a sample command that is executed by the script to classify and rank network connections is as shown in Listing 8.4.

Listing 8.4: Classifying and ranking network connections

```
ssh user@host ‘‘netstat —numeric-hosts’’
```

The network connection list obtained by executing the command provided as an example is utilised in the *Determine Remote Hosts* process with the algorithm presented in Chapter 6 as an input Set C in Equation 6.6.

8.3.6 EVIDENCE ACQUISITION PROCESS

The *Evidence Acquisition* process comprises four processes: *Evidence Acquisition Agents Activation & Installation*; *Network Traces Dump*; *RAM Capture* and *Dump Files Preservation*. The process starts with the *Evidence Acquisition Agents Activation & Installation* component which is implemented as a service or script task. The service installs required investigation packages listed in a text file and initialises

TLS servers in the forensic investigation servers by executing the command in Listing 8.5. It is worth noting that the command in Listing 8.5 is unique to an Ubuntu based operating system which is used as a proof of concept in this thesis. The TLS server to be used for file transfers is implemented with NodeJs [95] and it is used to start the server in the incident scene host, as shown in Listing 8.5. This assumes that the incident scene runs NodeJs environment and the TLS server source file is already transferred to the cloud based incident scene.

Listing 8.5: Evidence Acquisition Agents Activation & Installation

```
ssh user@host ‘‘cat tools-list.txt | xargs sudo apt-get install -  
↪ y && node tlserver.js &’’
```

The *Network Traces Dump* process executes the command in Listing 8.6 on the remote cloud incident scene. From Listing 8.6 onwards, the red hook right arrow (↪) indicates that a line is broken to fit the text width of the page. The network dump files are stored in a file for later analysis, should a need arise after completion of the live analysis of the network. During the *Evidence Examination and Analysis* process, the analysis of the dump files can still be used in conjunction with the live network analysis to make better informed deductions.

Listing 8.6: TCP Dump Command

```
ssh user@host ‘‘sudo -s 0 -U -n -w - -i eth0 not port 22’’ > .  
↪ localinvestigationdirectory/packet_capture
```

The *RAM Capture* process executes a command on the remote cloud incident scene. The executed command as shown in Listing 8.7 captures the RAM in the incident scene. The captured RAM is to be used during the analysis in support of the live analysis. Evidence obtained from analysing the live incident scene can be compared with the off-line analysis done on the RAM dump for more informed conclusions.

Listing 8.7: RAM Capture command

```
ssh user@host ‘‘sudo dd if=/dev/mem of=/.investigationdirectory/  
↪ memoru_dump’’
```

Listing 8.8 presents a command executed in the *Dump Files Preservation* process on the remote cloud instance. The executed command calculates the checksums of the network and RAM dump files and writes the checksums on a text file. The text

file constitutes part of the documentation that accompanies evidence data. Storing the checksum is part of the Evidence Preservation that runs concurrently with the investigation process.

Listing 8.8: Evidence Files Preservation

```
ssh user@host ‘‘md5sum /home/user /.investigationdirectory/tcpdump  
↪ && md5sum /home/user /.investigationdirectory/memory_dump >>  
↪ /home/user /.investigationdirectory/checksums.txt’’
```

The next section presents the *Evidence Transportation* process.

8.3.7 EVIDENCE TRANSPORTATION PROCESS

The *Evidence Transportation* Process comprises three processes: *Secure Link*, *Send Via Network* and *Removable Drive*. A *Secure Link* process is implemented as a manual task. The completion task form (Figure C.10) contains fields for an IP address and the credentials of the secure forensic storage server both to be provided by an investigator.

The *Send Via Network* process which utilises the results from the *Secure Link* process is implemented as a script task. The task takes the storage server credentials and the evidence item that need to be transferred to the storage server as an input. The script uses system commands such as "scp" on a Linux host to transfer evidence files between the incident scene and the secure storage server. The command is executed on the investigation server hosting CFaaS to retrieve evidence from the incident scene. This one way of retrieving the evidence. Another way is through making TLS calls to the TLS server/agent running on the incident scene. The complete command executed by the process to transmit evidence is as shown in Listing 8.9.

Listing 8.9: Secure Link Transmission

```
scp -r user@host:~/ .investigationdirectory/ .  
↪ localinvestigationdirectory
```

The *Removable Drive* transportation process is implemented as a manual task. Through its task form (Figure C.11), an investigator can provide evidence ID being transported with the removable drive and the drive identification. A brief description of the mode through which the removable drive is transported may also be provided.

8.3.8 EVIDENCE STORAGE PROCESS

The *Evidence Storage* process involves three processes which are: *Verify Evidence Integrity*, *Physical Drive Storage* and *Online Storage*. The first one, the *Verify Evidence Integrity* process, is accomplished through computation of a hash key and comparing it with the HASH key that was computed during the *Evidence Acquisition* process. The form used to undertake this process can be viewed in Figure C.12. The process is performed on data stored on removable drives as well as data transported through the wire.

Figure C.13 shows a form that is used to provide details of the storage server and the evidence ID for the *Online Storage*. The form contains a field to provide evidence identification and a field to provide details about the server on which the evidence was stored.

If the evidence is stored using a removable drive i.e. *Physical Drive Storage*, a task form (see Figure C.14) allows an investigator to provide the evidence storage drive ID and the description of where the storage drive is stored. This forms part of the *Documentation* process that is running concurrently with investigation processes in Figure 8.1.

8.3.9 EVIDENCE EXAMINATION AND ANALYSIS PROCESS

The implementation of the *Evidence Examination and Analysis* process is divided into two separate groups of processes, namely *Incident Host Based Analysis* and *Host Network Based Analysis*. The first group focuses on analysing the incident scene host itself, while the second group focuses on analysing the incident host network. The host-based analysis processes are shaded in Figure 8.1. The reason for the different routes is that, depending on an incident type being investigated (such as a network-based denial-of-service attack), a network analysis may be sufficient. If the case being investigated involves remote access to the incident scene with escalated privileges, analysing the cloud incident scene host system may be sufficient. Both analyses can however be carried out in parallel as they complement each other.

8.3.9.1 INCIDENT HOST BASED ANALYSIS

The host-based analysis process starts with the *Gather Incident Scene System Profile* component which is implemented as a script task. The system profile includes operating system and other information discussed in Section 5.2.9.2. This process is followed by the *Get System Time* component, which is also a script task.

Figure C.15 shows a task form used to execute the manual *List Modules Loaded to Kernel and Analyse List* process. Through the task form an investigator provides commands used to discover modules that are loaded in the kernel of the live incident scene. The list of modules and the details of each module discovered are provided through the task form. An investigator needs to provide analysis of these modules while providing the description.

Figure C.16 contains a form for the manual *List Active Processes and Analyse List* process performed on the incident scene. Commands or techniques that are used by an investigator to list the processes are provided through the task form. Abnormal or malicious processes can also be identified by an investigator as part of this process.

Figure C.17 contains a form for executing the manual *Determine Hidden Processes* process. While executing this task, investigators exercise their own intelligence in executing the process. The techniques that they use to unhide processes are provided through the task form. The assumption in this thesis is that login details to the incident scene are available and are as provided through the *Establish Secure Connection* process in Section 8.3.2. Using the credentials, an investigator signs into the incident scene and executes system commands that can reveal the hidden process. These commands are submitted as part of the techniques used to uncover the processes.

The *Analyse Allocated Memory of Suspicious Processes* process (see Figure C.18) is implemented as a manual task. For all suspicious processes uncovered during the *List Active Processes and Analyse List* and the *Determine Hidden Processes* process, the allocated memory space for each is analysed. The commands executed to analyse the memory space, as well as the process IDs and summary of what was observed in each process memory space are provided by an investigator by means of a task form (see Figure C.18).

Figure C.19 contains a task form used to complete the manual *Recover and Analyse Suspicious Processes Associated Files* process. One command that can be used by the investigator to list open files in a Linux host is the command in Listing 8.10. As part of this process, suspicious system processes are analysed further. This process involves recovering files associated with the suspicious system processes from the live memory. Techniques used to recover the files are supplied by the investigator and for each system process that was investigated, a list of the files is provided, using the task form.

Listing 8.10: Listing Open Files

```
ssh user@host ‘‘sudo lsof’’
```

The *Perform String Searches on Memory Dump or Live RAM* process is implemented as a manual task and can be completed through a task form (Figure C.20). Using the form, an investigator supplies the techniques or commands used to perform the string search. For each search term used in this process, a term that yields valuable outcomes is provided on the task form, together with the outcomes of that search. In Listing 8.11, an example of commands that can be executed to extract human readable data from the live RAM is presented.

Listing 8.11: RAM String Search Example [10].

```
ssh user@host ‘‘sudo dd if=/dev/mem | strings | grep ‘Search Key’  
↪’’
```

This command would then be entered in the ‘techniques used’ field, together with the outputs in their respective fields.

Figure C.21 shows the manual *Visualise Acquired Information* process completion task form. The process task form has fields to provide techniques and tools that are used to visualise the information obtained from the incident scene. It also has a field to be used to complete a summary of the visualisation of the information. The investigator can furthermore upload files generated from the investigation process.

In Figure C.22, a task form for completing the manual *Anti-forensic Rootkits Identification* process can be viewed. After performing the rootkit discovery or identification process, an investigator indicates whether any rootkits were found. This indication (which forms part of the outputs of this process) is to be used to determine if the host analysis process has to be restarted or not.

The anti-forensic rootkits identified in the *Anti-forensic Rootkits Identification* process can be deactivated in the *Anti-forensic Rootkits Deactivation* process using the task form in Figure C.23. After anti-forensic rootkits have been deactivated, the analysis process is restarted on the host. Deactivated anti-forensic rootkits are likely to reveal hidden processes and files. By restarting the host system analysis process, more potential evidence may be obtained. If no rootkits were discovered, the process continues to be combined with network analysis and then exits the *Evidence Examination and Analysis* process.

8.3.9.2 HOST NETWORK BASED ANALYSIS

Three basic manual tasks that can be performed by an investigator while analysing the network are *Decode Connection Protocols*, *View Reconstructed Flow* and *View Reconstructed Files*. The task forms for completing these tasks can be viewed in Figure C.24, Figure C.25 and Figure C.26 respectively. The *Evidence Examination and Analysis* process on the *Host Network Based Analysis* starts with the *Copy MAC Address and Kernel Route Cache Tables* process, which is a script or automated task. Inputs required for this task are incident scene IP address, the username and the password. It establishes a connection to the incident scene and executes a system command shown in Listing 8.12. The command in Listing 8.12 assumes that the remote cloud instance being investigated is a Linux-based host.

In this case, the command being executed by the script copies the details of the remote cloud instance network and routing tables, and stores them in a local file on the investigation host.

Listing 8.12: Copying MAC address and Routing tables

```
ssh user@host ‘‘ifconfig -a && route’’ >> .  
↪ localinvestigationdirectory/mac_routes.txt
```

The *List Current and Pending Connections and Their TCP/UDP Ports* process is the second process under network analysis to be performed and it is also implemented as a script task. The script makes use of the provided remote cloud instance credentials to connect to the host. It then executes the command as shown in Listing 8.13. The list of connections obtained from this process is written to a persistent local file in the cloud-based investigation instance as part of the *Documentation* process.

Listing 8.13: Listing Network Connections

```
ssh user@host ‘‘netstat --numeric-hosts’’ >> .  
↪ localinvestigationdirectory/connections.txt’’
```

Figure C.24 shows a task completion form for the manual *Decode Connection Protocols* process. The connection to be decoded in this process is the connections listed and given as output in the *List Current and Pending Connections and their TCP/UDP Ports* process. For each decoded connection, a description of the payload is presented through the task form. To accomplish this script task, network sniffing scripts by Silver Moon in [92] is made use of.

Extract Packet Attributes process is implemented as a manual task and it follows the manual *Decode Connection Protocols* in the standardised sequence of process. The input required for this process is a list of connections that are supplied by an investigator in the *Decode Connection Protocols* process. For each of these connections, the packet attributes are extracted and the results are used for statistical visualisation, e.g. number of packets versus protocol violation.

Data acquired from the network analysis needs to be visualised to allow for it to be better analysed[45]. In order to visualise the network data, the data is quantified first. The approach to quantify the attributes may involve converting them into a database which is the approach taken in this implementation of the digital forensic process chapter. After the network packet data have been converted to a database, queries can then be executed on against the data and results be fed into visualisation tools.

The process, *Convert Connection Attributes into Database*, which performs such conversion is implemented as a script task. The script utilises the technique presented by Tsaturyan in [142]. The script converts live traffic in an XML format using the command in Listing 8.14. This XML file is then further processed to comply with any database format. Alternatively, an investigator can use specialised tools to visualise data directly from the XML file without first converting it to the SQL database. This takes place as part of the *Visualise Reconstructed Network Flow* process. If this process involves visualising a Network traffic TCP dump file, e.g. *.pcap file, the same tool, "tshark" can be used to read data from the dump file into XML format.

Listing 8.14: Packet Capturing and into XM

```
ssh user@host "sudo tshark -i vbr1001 -T pdml" >> .  
↪ localinvestigationdirectory/livetraffic.xml"
```

After extraction of the network packets, storing them in the database and analysing the attributes, specific network flow can be reconstructed with the aim to recover files of interest in the network traffic. The reconstruction process is called Network Flow Reconstruction and it is implemented as a manual task.

Visualise Reconstructed Network Flow is the process that follows next. It uses the database information created in the *Convert Connection Attributes into Database* process where graphs can be plotted manually by an investigator. For this task, tools such as Wireshark [154] can be used to visualise the network flow directly from a live network.

Figure C.25 in the appendix presents a task form for completing the *View Reconstructed Flow* process, through which an investigator views the constructed network flow from a network dump. The data is next converted into XML in the *Convert Connection Attributes into Database* process. One technique that can be used to accomplish this task is to use ordinary spreadsheet tools such MS Excel that can accept data in XML format. The information is then provided through the task form. The input to this process is the output from the *Visualise Reconstructed Network Flow* process.

Mark Suspicious Packets is implemented as a manual task and can be carried out by investigators using tools such as Shorewall [118]. Suspicious packets are marked and enable isolation of those packets from the main traffic. In this way, packets with a suspicious profile can be filtered from the network traffic and thus reduce the volume of network data that needs to be analysed by investigators. This exercise is implemented as a manual task and is known as the *Reduce Evidence Data Based on Marked Packets*.

Files can be reconstructed from the network flow by using tools such as Wireshark [154]. A task form that is used to carry out the *View Reconstructed Files* process is represented in Figure C.26. An investigator can upload the files viewed while also providing their description.

8.3.10 EVIDENCE INTERPRETATION PROCESS

The *Evidence Interpretation* process comprises the *Review Statistical Analysis* and the *Validate Case Based on Statistical Data* processes.

Figure 8.4 contains a task form that is used to complete the *Review Statistical Analysis* process. Reviewing the statistics will be used in the *Validate Case Based on Statistical Data* process, and validation of the case process is completed by using the task completion form represented in Figure 8.5.

An investigator also provides his/her opinion on the validity of the case being investigated. This can be used to decide whether the case can be concluded or has to be restarted.

In the task form, the investigator provides a motivation for his/her decision on the validity of the case. The motivation can include reference to parts of the preliminary report that are always visible to investigators during the entire investigation.

View Statistics

Review statistical information obtained from the investigation.

Stats ID

EXIBIT56

Stats Summary

This image is a bar chart representing a number of connections per remote host over a period of 5 hours. Host __.28.44 appears to have constituted the largest number of connections in total. This host is followed by host __.28.56 and then by host __.1.55. On preview of results from previous processes of the in vestigation, these IP addresses appear to have been among the prioritised list of hosts discovered during the evidence identification processes.

Stats ID

EXIBIT57

Stats Summary

This is a pie providing a different view of the statistics in EXIBIT56

Figure 8.4: Statistics Viewing

8.3.11 REPORTING PROCESS

The *Reporting* process embraces the one process represented in Figure 8.6. The task form allows an investigator to provide a case ID that the report is generated from. The investigator also provides a summary of the case, which will accompany the report that has been generated incrementally during the entire investigation.

8.3.12 PRESENTATION PROCESS

Figure 8.7 represents a task completion form that is used to execute the *Presentation* process of the case. The form allows the investigator to provide the date on which the presentation took place. The summary field may include information such as outcomes from the presentation and recommendations that were made during presentation of the case. An example of a recommendation may be that the investigation has to be restarted from scratch or as from a particular step onwards.

Validate Case

Provide own judgement on the validity of the case based on the available information. Even if the case would be invalid, it would still go through the reporting and Presentation processes before it can be concluded.

Case ID

MER789/2014

Case Valid



Motivation

On analysing data that has been collected throughout the investigation process and most importantly the analysis phase, it can be concluded that this is a valid case that can be presented in a court of law and possibly lead to prosecution.

The visualised statistics of the data collected during the investigation show that there has been a large number of connections from different hosts originating from a same network. The connections from the hosts are using different ports that the expected ports with hosted services on the cloud instance.

It has also been established that a user with privileged access to the instance in question, appears to have been accessing the instance outside of expected office hours. The legitimate user with the allegedly compromised user ID acknowledges that they never accessed the instance during those hours.

Submit

Figure 8.5: Case Validation

Reporting

Printout outcomes from each task of the investigation process and also provide a summary of the investigation.

Case ID

MER789/2014

Investigation Summary

The investigation processes have been concluded. Based on the comprehensive investigation process carried out, the suspected case of an unauthorised access to a cloud based instance (incident scene) of IP address _._.55.34 is considered valid. There are malicious IP addresses have been found to be connecting to the incident scene without necessarily accessing services hosted on the instance which have been listed on the generated report particularly in the Evidence Interpretation process.

Submit

Figure 8.6: Reporting

8.3.13 INVESTIGATION CLOSURE PROCESS

Figure 8.8 represents a task completion form used to execute the *Investigation Closure* processes. The form allows an investigator to provide a case ID and a summary of

Presentation

Actual presentation of the report from the investigation. This presentation may be to the investigation panel or a hearing in a court of law.

Presentation Date
12/03/2014

Presentation Summary

The report on this investigation was presented in a panel of investigators in the presence of the victim. The aim was to decide on whether to carry on with the investigation. Taking the investigation further would include investigating additional hosts that have been listed and prioritised as potential evidence sources.

The investigation panel was advised to take the investigation further

Submit

Figure 8.7: Presentation

the case. The latter includes information such as regarding convictions.

Investigation Closure

Case ID
MER789/2014

Case Summary

The case has not been closed yet. This is just the end of a preliminary investigation. It has been concluded that more evidence is required from additional evidence sources.

Submit

Figure 8.8: Investigation Closure

8.4 CONCLUSION

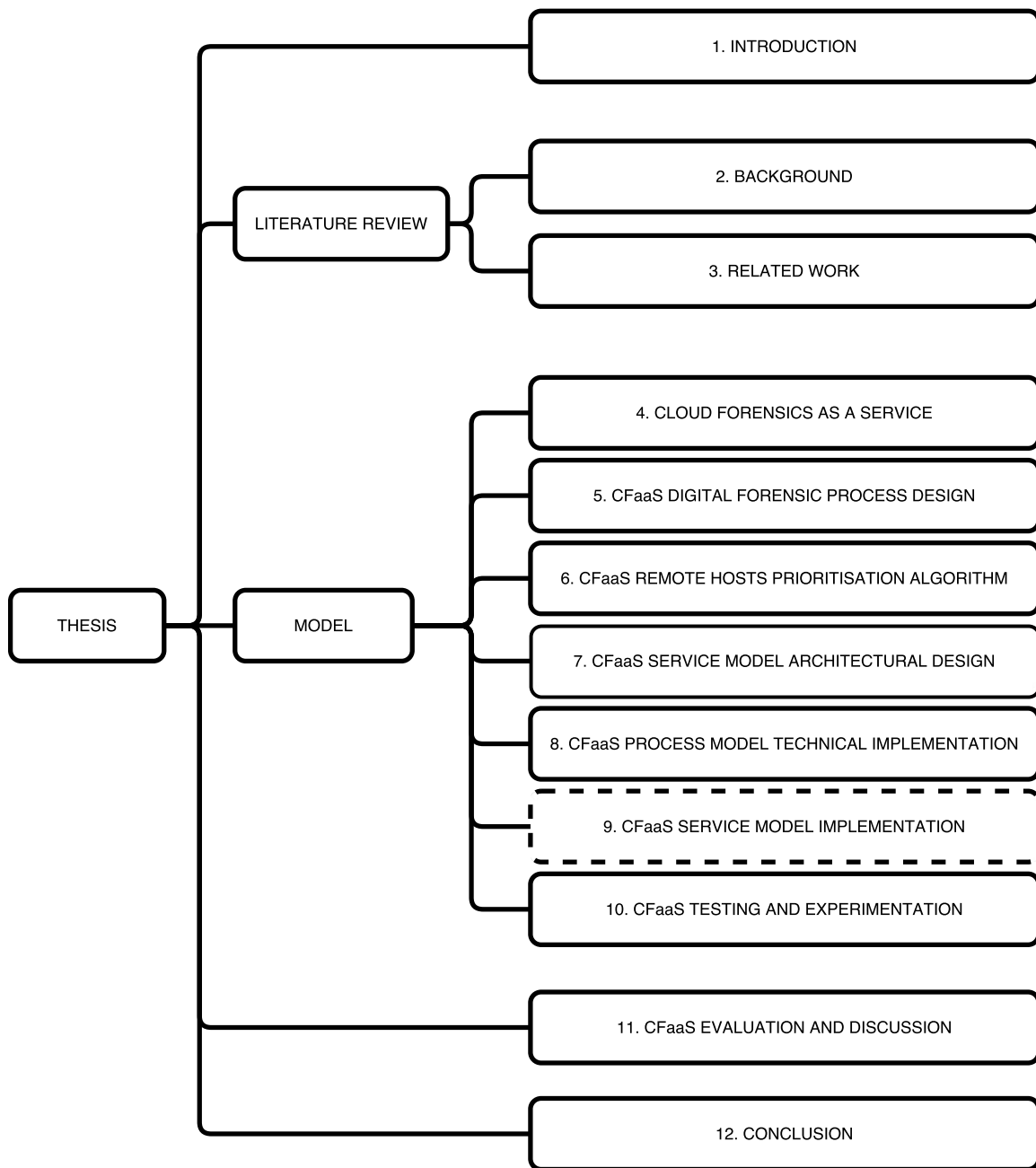
This chapter presented the implementation of the proposed standardised cloud forensic investigation process in the cloud and discussed how each process depicted in Figure 8.1 was implemented. Examples of techniques that could be used by an inves-

tigator in executing the processes were provided. This implemented process requires a suitable environment that would enable the investigator to interact with the process.

Chapter 9 presents the implementation of such an environment. In the chapter, a focus is put on the implementation of the CFaaS service that is used to aid investigators in executing the cloud forensic process presented in this chapter.

9

CFaaS Service Model Implementation



9.1 INTRODUCTION

One of the main contributions in this thesis is a standardised cloud forensic process. The design of the cloud forensic process model was discussed in Chapter 5, and its implementation followed in Chapter 8. This chapter deals with the technical implementation of the CFaaS service model that executes the process implemented in Chapter 8. The design of the CFaaS service model was presented in Chapter 7. Chapter 9 also covers the hardware environment and software implementations in

Section 9.2. In Section 9.2, necessary screen-shots while utilising the process. Section 9.3 concludes the chapter.

9.2 CFAAS SERVICE IMPLEMENTATION

This section presents implementation details of the cloud forensic service, CFaaS, proposed in this thesis as well as the technologies used. The service was developed and deployed as part of a private cloud environment. The details on each component are presented in the subsequent sections.

9.2.1 HARDWARE COMPONENT

The hardware components of the digital forensic system were deployed as shown in Figure 9.1.

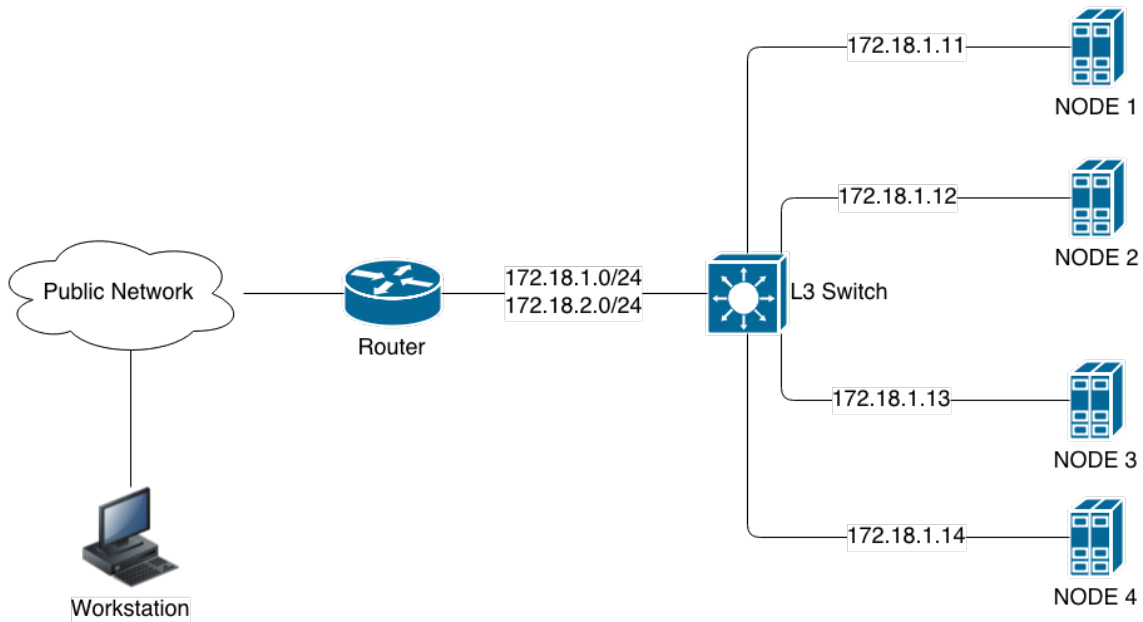


Figure 9.1: CFaaS service deployment environment

The deployment involves four physical nodes, a network switch and a router. The router is a configured Linux host that runs Ubuntu 12.04 and has two network interface controllers (NIC) cards - an external and an internal NIC. One of the interfaces, the external interface, is connected to the public network and the internal interface is connected to the private network. The router is configured to forward 172.18.2.0/24 and 172.18.1.0/24 networks through the internal interface. The internal NIC is physically connected to the 172.18.1.0/24 network, the same network as the nodes network.

The switch used in this set-up is an unmanaged network switch (i.e. it cannot be configured). The hardware specifications of the nodes are as represented in Table 9.1. The infrastructure therefore has a total of 16 CPU cores and 28GB of RAM. The nodes are all configured with static IP addresses from the 172.18.1.0/24 network and they are running Ubuntu Server 12.04.

Table 9.1: Private cloud infrastructure hardware specifications.

Node	CPU	RAM	Quantity
Node type 1	Intel(R) Core(TM) i5-3550 CPU @ 3.30GHz	8GB	3
Node type 2	Intel(R) Core(TM) i5 CPU 750 @ 2.67GHz	4GB	1
Total	16	28GB	4

9.2.2 HYPERVISOR COMPONENT

The hypervisor used in the deployment of CFaaS is the kernel-based virtual machine (KVM) [72]. The choice of hypervisor was based on it being open source and one of the best performing hypervisors [63, 82] and it comes bundled with the cloud software used in the implementation to be discussed next.

9.2.3 CLOUD SOFTWARE COMPONENT

For the cloud software component, OpenStack is used. One of the nodes presented in Section 9.2 is used as a management node while the other three nodes are used as computing nodes. The 172.18.2.0/24 network is configured to be a public IP address pool and it is assigned to new instances when they are launched. Two components in the cloud software are singled out which are, the PaaS manager and the CFaaS images or image manager. The CFaaS images component comprises of pre-built CFaaS images that are uploaded into the cloud software. OpenStack provides two methods of booting up instances from the images. The first method involves manually selecting an uploaded virtual machine image or a snapshot and starting a specific number of instances at once. The second method is through using the orchestration service that ships with OpenStack, named Heat. To launch an instance using Heat, a template specifying an image to be used and the hardware requirements of the instance is launched into the Heat service. The Heat service, together with a load balancer, automatically launches new instances on demand and removes unnecessary

instances when they are no longer needed - thus demonstrating the elasticity of the cloud.

Listing 9.1: Heat Service Template

```
heat_template_version: 2013-05-23
description: Simple template to deploy a single
CFaaS service instance
resources:
  cfaas_instance:
    type: OS::Nova::Server
    properties:
      key_name: heat_key
      image: Forensic_Service
      flavor: m1.medium
```

9.2.4 PLATFORM COMPONENT

The platform component is a collection of tools required to run the CFaaS service. It has internal components, a database server, runtime environment, script execution environment, application server and SaaS manager. Mongo DB [8] is used to implement the persistent storage. The CFaaS service is developed using Java™ technologies, and the runtime environment used on the platform is Java Runtime Environment (JRE) 1.7.0.45. During a digital forensic investigation, servers such as file transfer servers may need to be started on the fly. Some automated tasks such as script tasks need to execute local investigating system commands to complete their tasks. This is achieved through Java.

While preparing the incident for an investigation during the *Start Servers* process, some of the files that would be used to start-up investigation agents and SSL/TLS servers are transferred to the incident scene. These files are implemented in JavaScript and node specifically NodeJs. The choice of JavaScript over other technologies such as Java is its relatively lower overhead. A comparison between JavaScript and Java can be seen in [4]. In the deployment, Java Script is used in the environment known as Node.js [95]. A code snippet of an SSL/TLS server implemented with NodeJs (which implements the function in Listing 9.2 gets a list of files in a directory provided as a parameter.

Listing 9.2: File Listing Function

```
function listfiles(response , request){
    var directory = require('url').parse(request.url).search.
        ↪ substring(11);
    var contents = "empty";
    exec("ls "+directory " -lah", function(error , stdout ,
        ↪ stderr){
        contents = stdout;
        if(error){
            contents = stderr;
        }
        response.write(contents);
        response.end();
    });
}
```

This server can be used as an agent on a remote host being investigated to assist with the live analysis of the remote host. In this thesis, it is assumed that installing additional software in the incident scene has documented effect on the scene. This issue has also been addressed in Section 8.3.2. SSL/TLS mutual authentication [27, 74] is enforced in the communication between CFaaS and the server agents running on the cloud incident scene. This ensures that the agents running on the incident communicate only with the CFaaS service with which it has exchanged certificates. In this manner, evidence is protected against malicious users.

9.2.5 IDENTITY AND SECURITY MANAGEMENT COMPONENT

This component is currently implemented using the Meteor framework [7]. The built in accounts management module in the Meteor can be configured to work the LDAP server used in the design of CFaaS. The *Identity and security management* component comprises Authentication, Authorisation, Auditing and user database modules. Currently, in the Authentication module a basic username-password pair or “*a what you know based authentication*” method is used. More advanced authentication methods can still be incorporated. The Authorisation module uses a role-based method. The activities in the investigation process that can be carried out only by qualified digital forensic investigators, while other tasks such as logging or registering a case can be performed by administrators, are further enforced by the jBPM process engine which

makes manual tasks to be only visible to users based on their roles specified in the process definition. The auditing capability is enabled through writing all access logs on log files and in the database. The details form part of the *Documentation* parallel process and the user database used in the deployment is MongoDB.

9.2.6 CFAAS PROCESS MANAGER COMPONENT

The digital forensic process implemented and executed in CFaaS service is compliant with the ISO/IEC 27043 draft standard [60]. Cloud forensics combines network forensics and RAM or Live forensics. Standard-based forensic procedures specific to network and live forensics are presented by Sibiyi et al. in [120] and [124] respectively. As was thoroughly discussed in Chapter 8, the cloud forensic process is implemented as a jBPM [6] business process. The API, *CFaaS Process Management* exposes functions that are called by the CFaaS service to start up a process instance, terminate a process instance or to query the status of a process instance.

9.2.7 CFAAS TASKS SERVER COMPONENT

A standardised forensic process in CFaaS service is implemented using a business process template in jBPM. Procedures that are carried out are viewed as tasks to be completed either by investigators or automatically by scripts. Some of the tasks need to be completed by humans while some can be performed by scripts. The human task server manages tasks that need to be manually carried out by investigators. It contains the process engine that executes script tasks and hands over manual tasks to relevant digital forensic team members. The task server also has a persistent storage A. Hardware to keep track of the status of each task and storing each task outcomes.

9.2.8 CFAAS SERVICE COMPONENT

This section discusses the *CFaaS Service* components which houses other major components of the service. All components within the *CFaaS Service* component are implemented using the Meteor Framework. The *CFaaS API* (application programming interface) component implements all the web interfaces that an investigator uses to interact with CFaaS. Examples of the interfaces are the task forms. A sample task form is shown in Listing 9.3. The listing shows an implementation of the *Establish Secure Connection* process task form that was discussed in Section 8.3.2.

Listing 9.3: Establish Secure Connection Task Form

```

<template name="EstablishSecureConnection">
<div class="col-md-12"><h1>Establish Secure Connection</h1></div
  ↪ >
  ...
<div class="row"><div>Incident Scene IP Address </div><div>
  ↪ textarea id="ipaddress" style=""></textarea></div></div>
<div class="row"><div>User Name </div><div>textarea id="
  ↪ username" style=""></textarea></div></div>
<div class="row"><div>Password </div><div><input type="password
  ↪ " id="password" style=""/></div></div>
  ...
</template>

```

The main window that is used by investigators during the course of the investigation is depicted in Figure 9.2. The page shows two investigations that are under way. The MER567/2014 case is selected and it shows completed tasks and tasks that are activated and being carried out ("Reserved" state in the figure). The far right-hand column in Figure 9.2 shows progress of a selected case.

The screenshot displays the 'Investigation Page' in the CFaaS interface. The page is divided into several sections:

- Navigation Bar:** Includes 'CFaaS', a search field, and links for 'Dashboard', 'Settings', 'Profile', 'Help', and 'Logout'.
- Left Sidebar:** Contains navigation links: 'Home', 'New Investigation', 'Current Investigations', 'Reports', and 'Info'.
- Investigation Page Header:** Features a blue header with the title 'Investigation Page'.
- Case Overview:**
 - Case ID:** MER567/2014
 - Description:** The case involves disruptions of cloud services belonging to the MER Inc. Several complaints have been received from customers with regards to intermittent accessibility of the service for a period of 5 days now. It is suspected that services may be under a denial of services attack.
 - Case ID:** TTMR645/2014
 - Description:** The case involves suspected unauthorized access to cloud services that belong to TTMR.
- Case Management:**
 - MER567/2014:** Includes buttons for 'Register Case', 'Claim', 'Start', and 'Completed'.
 - Incident Description:** Includes buttons for 'Claim', 'Start', and 'Reserved'.
- Task List (Right Panel):** A hierarchical tree of tasks under the heading 'Cloud Forensics'. The 'Remote Files' task is highlighted in green. The tasks include:
 - Incident Detection
 - Incident Description
 - First Response
 - Establish Secure Connection
 - Remote Connection
 - Start Servers
 - Enabling Secure Logging
 - Planning
 - Forensic Team Organisation
 - Network Types Identification
 - State of the art RAM and Hardware
 - Network Evidence Sources Identification
 - Preparation
 - Task Assignment
 - Evidence Analysis & Collection Tools Spec
 - Forensic Tools Acquisition
 - Potential Evidence Identification
 - Locate Paging File
 - RAM Classification
 - Corrupt Data Identification
 - Determine Remote Hosts
 - Evidence Collection
 - Evidence Collection Agents Activation & In
 - Network Traces Dump
 - RAM Capture
 - Dump Files Preservation
 - Evidence Transportation
 - Removable Drive
 - Secure Link
 - Send Via Network
 - Evidence Storage
 - Verify Evidence Integrity
 - Online Storage
 - Physical Drive Storage
 - Evidence Examination and Analysis
 - Gather Incident Scene System Profile
 - Get System Time
 - List Modules Loaded to Kernel and Analyse
 - List Active Processes and Analyse List
 - Determine Hidden Processes
 - Analyse Allocated Memory of Suspicious P
 - Recover and Analyse Suspicious Process
 - Perform String Searches on Memory Dumps
 - Visualize Collected Information
 - Anti-forensic Rootkits Identification
 - Anti-forensic Rootkits Deactivation

Figure 9.2: Remote files list request output

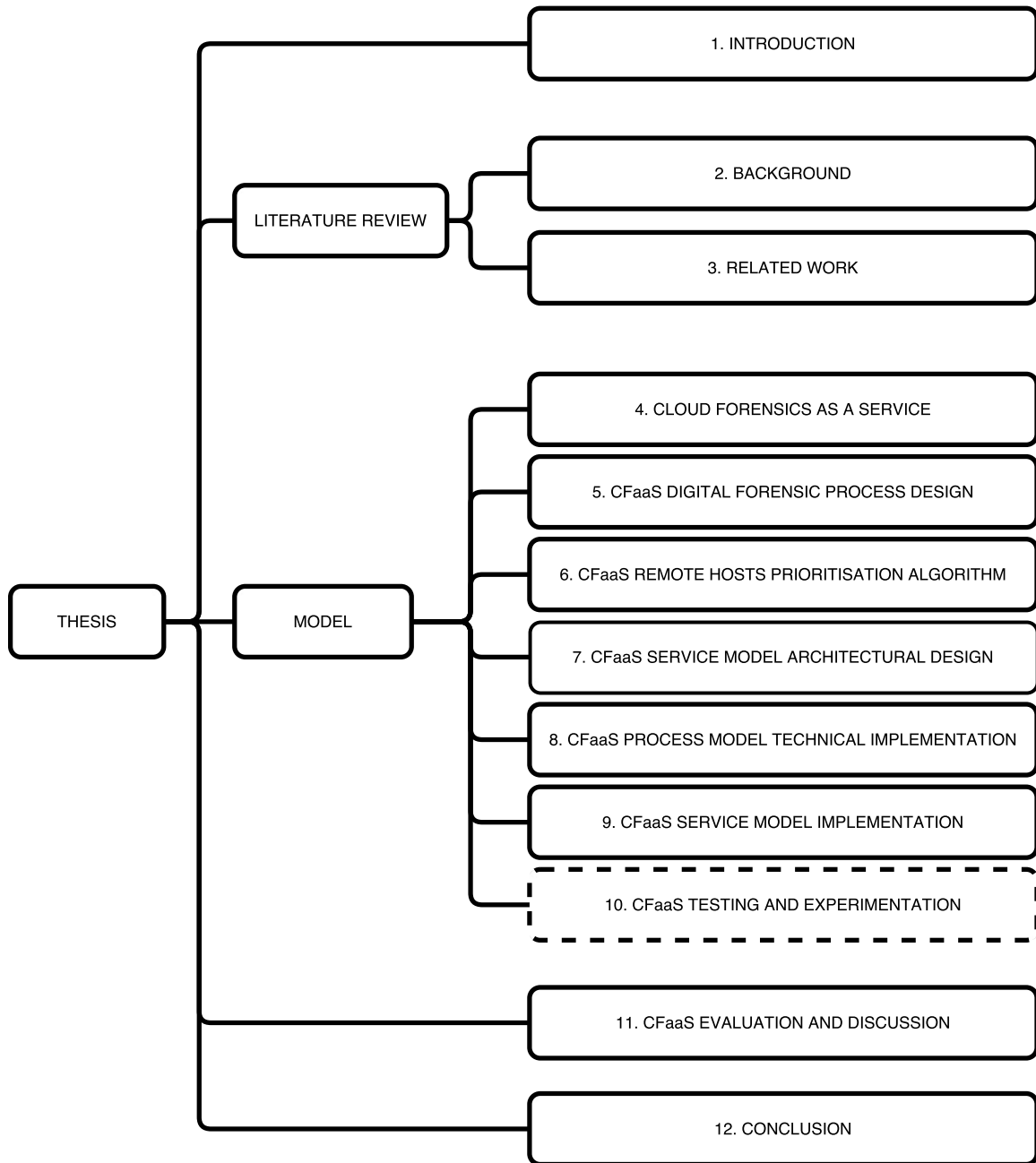
9.3 CONCLUSION

This chapter presented the technical implementation details of CFaaS, which ranged from the hardware cloud infrastructure, on which CFaaS was deployed, to the service front end of CFaaS. The chapter therefore focused on implementation of the service referred to in this case as the “environment” or “infrastructure” on which the process model implemented in Chapter 8 is executed.

In Chapter 10, the CFaaS framework is evaluated. The evaluation will focus on the design of the digital forensic process model, the design of CFaaS service model, the SLOE algorithm presented in Chapter 6 and the implementation of CFaaS presented in Chapter 9.

10

CFaaS Evaluation and Testing



10.1 INTRODUCTION

The main purpose of this research was to formulate a framework to be used to conduct a digital forensic investigation in cloud environments. The framework addresses issues that are associated with cloud environments that are volatile in nature, hence making it hard for the applicability of conventional procedures and techniques of conducting an investigation in cloud environments. The main research questions that

are addressed by the framework proposed in this thesis were presented in Chapter 1 and the framework in Chapter 4.

This chapter presents an evaluation that demonstrates the extent to which the proposed framework addresses these research questions. A design science research method [54, 97] was adopted in this thesis. According to Hevner et al. in [54], design science “addresses research through the building and evaluation of artefacts designed to meet the identified business need.” In addressing the research questions (in Section 4.2 in Chapter 1) as business needs in cloud forensics, four artefacts emerged from this research. The artefacts are cloud forensic procedures presented in Chapter 5, a remote hosts selection algorithm (SLOE algorithm) aimed at minimising costs presented in Chapter 6 and a cloud forensics system architecture presented in Chapter 7 and finally, a prototype that implements the digital forensic procedures and the architecture.

All these artefacts are evaluated in Chapter 10. The chapter is organised as follows: Section 10.2 presents the evaluation of the framework based on the proposed digital forensic process. Section 10.3 presents the evaluation of the remote hosts prioritisation algorithm that is aimed at minimising the costs of an investigation. Section 10.4 presents the evaluation of the design of CFaaS and how the design addresses the system requirements, and Section 10.5 evaluates CFaaS prototype implementation and the extent to which it addresses functional requirements. Section 10.6 concludes the chapter.

10.2 PROCESS EVALUATION

To the researcher’s best knowledge, at the time of writing there were no standardised digital forensic processes tailored for the cloud that were available for comparative analysis against the processes proposed in this chapter. Criteria for evaluating digital forensic processes are also lacking. To evaluate the digital forensic processes proposed in this chapter, the process is subjected to the cohesion and coupling matrix proposed by Vanderfeesten et al. in [149] that was designed to evaluate work flows. There was no specific reason for the choice of this evaluation criterion, and the aim of the evaluation was to provide a benchmark for other digital forensic process designs intended for cloud environments. Other evaluation methods can still be used, such as through process simulation to obtain estimates such as those on costs and time [62, 106].

Vanderfeesten et al. define a process cohesion¹ through Equation 10.1. ch in

¹Cohesion is a measure of the relationships of the elements within a module and is sometimes

Equation 10.1 is an average of individual cohesions for all individual tasks. Details on determining cohesion of individual tasks are discussed thoroughly in [149].

$$ch = \frac{\sum_{t \in \bar{S}} c(t)}{|\bar{S}|} \quad (10.1)$$

Equation 10.2 [149] represents the coupling² of a process. T represents an activity which is a set of operations with a resource class or role that is authorised to execute the task. An operation refers to the act of transforming a given set of information items into a new information item. \bar{T} represents an activity represented without specifying resources such as human resources that will carry out the task. \tilde{T} represents the task in a format that specifies input-output relations on information elements. \hat{T} represents a set of information elements processed as inputs to an activity. \bar{S} is a set of activities without references to resource classes or roles.

$$cp = \begin{cases} \frac{|\{(T_1, T_2) \in S \times S | \bar{T}_1 \neq \bar{T}_2 \wedge (\hat{T}_1 \cap \hat{T}_2) \neq \emptyset\}|}{|S| \cdot (|S| - 1)}, & \text{for } |S| > 1 \\ 0, & \text{for } |S| \leq 1 \end{cases} \quad (10.2)$$

Equation 10.3 represents a ratio between coupling and cohesion in the entire process. A lower value of the ratio is desired as cohesion in a workflow process is more preferable than coupling [149].

$$\rho = \frac{cp}{ch} \quad (10.3)$$

Equation 10.4 presents a list of digital forensic process information items. Details on the information items are provided in Table 10.1. After summarising the evaluation criteria, the process information elements are modelled as follows in the procedure in [149] and then apply equations 10.1, 10.2 and 10.3 to evaluate the process design. In Equation 10.4, Ω represents a set of information items that are also discussed in

referred to as module strength [149]

²Coupling is measured by the number of interconnections among modules [149]

detail in Table 10.4.

$$\Omega = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56\} \quad (10.4)$$

W in equation 10.5 [149] represents a set of resource classes or roles capable of carrying out an operation on data items. h represents human resources and s represents a script or service resource class. h can still be broken down into further roles such as investigator or technician. For the purposes of this chapter, however, it is kept it at its general level, h .

$$W = \{h, s\} \quad (10.5)$$

In Equations 10.6 through Equation 10.18, processes presented in Section 5.2 through Section 8.8 as task A through task M are represented. The tasks are represented as sets of operations on information items that are executed in each task. For example, the operation $(2, h, 1, 4, 10, 11)$ in Equation 10.6, information items 1, 4, 10 and 11 are used by a human resource to obtain the information item, 2.

$$A = \{(2, h, \{1, 4, 10, 11\})\} \quad (10.6)$$

$$B = \{(4, s, \{3, 11\}), (5, s, \{3\}), (6, s, \{11\})\} \quad (10.7)$$

$$C = \{(8, h, \{11\}), (9, h, \{11\}), (10, h, \{2, 11\})\} \quad (10.8)$$

$$D = \{(11, h, \{7\}), (12, h, \{2, 11\}), (13, h, \{11\})\} \quad (10.9)$$

$$E = \{(14, s, \{3, 13\}), (15, s, \{3, 13\}), (16, s, \{3, 13\}), (17, s, \{2, 3, 13\})\} \quad (10.10)$$

$$F = \{(18, s, \{3\}), (19, s, \{3, 18\}), (20, s, \{3, 14, 18\}), (21, s, \{3, 19, 20\})\} \quad (10.11)$$

$$G = \{(22, h, \{11, 19, 20\}), (23, h, \{11\}), (24, s, \{3, 23\})\} \quad (10.12)$$

$$H = \{(25, h, \{11, 21, 22, 24\}), (26, h, \{11, 22, 25, 24\}), (27, h, \{11, 22, 25\})\} \quad (10.13)$$

$$I = \{(28, s, \{3, 18\}), (29, s, \{3, 18\}), (30, h, \{2, 3, 20, 11\}), (31, h, \{2, 3, 11, 20, 38\}), (32, h, \{3, 11, 20\}), (33, h, \{2, 11, 32, 30, 31\}), (34, h, \{2, 11, 33\}), (35, h, \{2, 11, 26, 27\}), (36, h, \{11, 34, 35\}), (37, h, \{11, 13\}), (38, h, \{11, 37\}), (40, s, \{39\}), (41, h, \{11, 40\}), (42, s, \{3, 41\}), (43, s, \{42\}), (44, s, \{3, 26, 27\}), (45, h, \{3, 11, 43, 44\}), (46, h, \{11, 45\}), (47, h, \{3, 11, 44, 46\}), (48, s, \{47\}), (49, h, \{3, 11, 48\})\} \quad (10.14)$$

$$J = \{(50, h, \{11, 49\}), (51, h, \{11, 17, 45, 50\}), (52, h, \{11, 51\})\} \quad (10.15)$$

$$K = \{(53, h, \{11, 52, 55\})\} \quad (10.16)$$

$$L = \{(54, h, \{53\}), (55, s, \{4, 5, 6, 8, 9, 10, 21, 28, 29\})\} \quad (10.17)$$

$$M = \{(56, h, \{54\})\} \quad (10.18)$$

Table 10.1: forensic Information Elements

1	Register Case - Case Information
2	Incident Description - Case description
3	Establish Secure Connection - Incident IP Address and credentials
4	Remote Connection - Connection attempt outcome
5	Start Servers - Outcomes that may include errors
6	Enabling Secure Logging - Enabling logs outcomes including log directory
7	forensic Team Organisation - Team members list
8	Network Types Identification - Network types information
9	State-of-the-art RAM and Hardware - RAM/Hardware information
10	Network Evidence Sources Identification - Potential network evidence sources locations
11	Task Assignment - Tasks with assigned members
12	Evidence Analysis & Collection Tools Specification - forensic tools specifications
13	forensic Tools Acquisition - forensic Tools
14	Locate Paging file - Page file directory
15	RAM Classification - RAM Description
16	Corrupt Data Identification - Corrupted files list
17	Determine Remote Hosts - List of remote IP addresses
18	Evidence Collection Agents Activation & Installation - Installation/Activation outcomes
19	Network Traces Dump - TCP Dump file(s)
20	RAM Capture - RAM dump file
21	Dump files Preservation - HASH keys
22	Removable Drive - Drive transportation mode description
23	Secure Link - forensic storage server IP address and credentials
24	Send Via Network - Sending outcomes
25	Verify Evidence Integrity - Integrity Verification results
26	Online Storage - Storage outcomes including evidence ID
27	Physical Drive Storage - Description of drive storage location
28	Gather Incident Scene System Profile - Incident scene system profile such as CPU and RAM info
29	Get System Time - System time
30	List Modules Loaded to Kernel and Analyse List - Module list and list analysis summary
31	List Active Processes and Analyse List - Process list and list analysis summary
32	Determine Hidden Processes - Process list
33	Analyse Allocated Memory of Suspicious Processes - Memory analysis summary
34	Recover and Analyse Suspicious Processes Associated files - file analysis summary
35	Perform String Searches on Memory Dump or Live RAM - String search results and summary
36	Visualise Collected Information - Visualisation images
37	Anti-forensic Rootkits Identification - Identified rootkits
38	Anti-forensic Rootkits Deactivation - deactivated rootkits
39	Copy MAC Address and Kernel Route Cache Tables - MAC addresses and routing tables

40	List Current and Pending Connections and Their TCP/UDP Ports - Network connection list
41	Decode Connection Protocols - Decoded connections
42	Extract Packet Attributes - Packet attribute-value pairs
43	Convert Connection Attributes into Database - DB Info
44	Network flow Reconstruction - Reconstructed network flow
45	Visualise Reconstructed Network flow - Visualisation images
46	View Reconstructed flow - flow summary
47	Mark Suspicious Packets
48	Reduce Evidence Data Based on Marked Packets - Reduced data
49	Reconstruct files - Reconstructed files
50	View Reconstructed files - files viewing summary
51	Review Statistical Analysis - Stats summary
52	Validate Case Based on Statistical Data - Validation results and summary
53	Reporting - Report
54	Presentation - Presentation summary
55	Documentation - Documentation

The evaluation to determine cohesion and coupling of the entire process was performed using Equation 10.1 and Equation 10.2 respectively, namely:

$$\begin{aligned}
 ch &= \frac{ch\{A\} + ch\{B\} + \dots + ch\{M\}}{13} \\
 &= \frac{0 + 0.266667 + \dots + 0}{13} \\
 &= 0.400403
 \end{aligned} \tag{10.19}$$

$$\begin{aligned}
 cp &= \frac{|\{(A, B), (A, C), \dots, (M, L)\}|}{13 * 12} \\
 &= \frac{2 * 81}{13 * 12} \\
 &= 1.038461538
 \end{aligned} \tag{10.20}$$

The coupling-to-cohesion ratio in Equation 10.3 therefore yields:

$$\begin{aligned}
 \rho &= \frac{1.038461538}{0.400403} \\
 &= 2.593540854
 \end{aligned} \tag{10.21}$$

A similar evaluation procedure is carried out on a digital forensic process by Al-Fedaghi and Al-Babtain [12]. Al-Fedaghi and Al-Babtain followed the digital forensic process by NIST [68] which consists of Collection, Examination, Analysis and Reporting stages. They however consider Reporting to be the Presentation stage, which

generates reports. They then break down data that flows through the investigation process into Evidence, Information and Reality groups. Each of these groupings gets transformed in each stage of the investigation process. Each of the transformation states is then considered as a unique information item by in this thesis. The Report, which is unique to the Presentation stage as shown in Figure 10.1, is considered an information item.

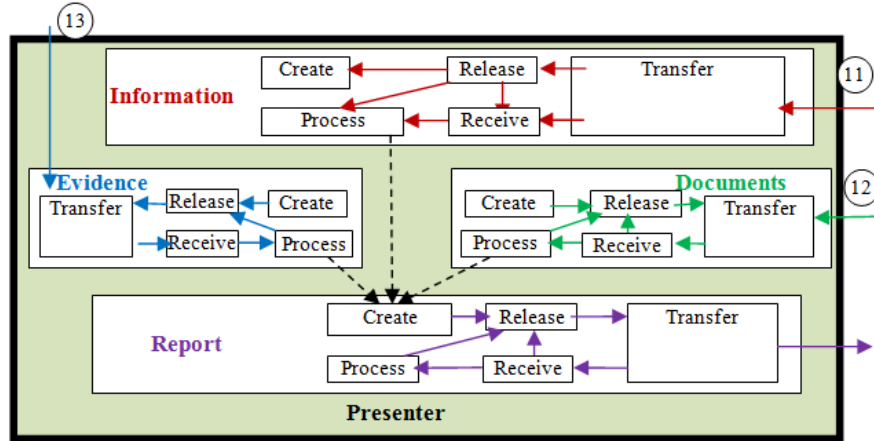


Figure 10.1: Presentation Sphere [12]

The author of the current thesis regards each of the four stages of the process as activities. In [12], these activities are carried out by a Collector, Examiner, Analyser and a Presenter respectively - all of which are humans, i.e. h in Equation 10.5. The equations (10.1 through to 10.3) were then applied to obtain the results as shown in Table 10.2.

Table 10.2: Results Table

	Cohesion (ch)	Coupling (cp)	Ratio (ρ)
CFaaS Process	0.400403	1.038461538	2.5935408
Al-Fedaghi and Al-Babtain [12]	0.05	1	20

In the next section, the evaluation of the second artefact, the Remote Hosts List Optimisation Algorithm is presented.

10.3 REMOTE HOSTS PRIORITISATION (SLOE) ALGORITHM EVALUATION

This section evaluates the cost minimisation algorithm presented in Chapter 6. The SLOE (SeLection Of rEmote hosts) algorithm is evaluated based on correctness and

time efficiency in the next subsections.

10.3.1 CORRECTNESS VERIFICATION

A correctness verification of an algorithm is aimed at verifying that the SLOE algorithm terminates and always produces desired output. Correctness means that “..*certain desirable program properties hold*” [17]. For example, if it is a sorting algorithm that is being analysed, one of the properties that need to be satisfied would be that an element in the list being sorted would be expected to be greater than the element that precedes it in ascending order (and smaller if in descending order). The steps involved in correctness evaluation as summarised by Skiena in [128, p.11] are first, a clear description of what is to be proved, assumptions and lastly, a chain of reasoning. Algorithm correctness verification procedures are also discussed in detail by Apt in [17].

The clear statement of what is to be proved is a description of the problem as well as the properties that need to be fulfilled by an algorithm. Assumptions are what are taken to be true and those facts are used during the process of proving correctness of the algorithm. The chain of reasoning involves the logical step-by-step mathematical description of the algorithm on how it satisfies properties of a solution to the problem.

In the next subsection, the problem to be solved is described and what needs to be archived by the SLOE³ algorithm that solves the problem in the form of postconditions is provided.

10.3.1.1 DESCRIPTION OF WHAT IS TO BE PROVED

The problem to be solved by the SLOE algorithm is stated by means of providing the inputs and the description of the problem.

The inputs of the SLOE algorithm are a set of predefined attributes represented by set A from Equation 6.4⁴, as well as an IP address and an incident categorisation at the incident type i_k (i.e. $i_k \in I$) where I is from Equation 6.3⁵.

A set of connections in the incident scene represented by set C in Equation 6.6⁶ is populated immediately after the SLOE algorithm has started execution. The set is populated by obtaining a list of connections from the host with the provided IP address. The network attributes in set A and the incident type obtained from

³SeLection Of rEmote hosts (SLOE) algorithm

⁴Equation 6.4: $A = \{a_i | a_i, \text{ is a connection attribute, } i \in N\}$

⁵Equation 6.3: $I = \{i_k | i_k, \text{ is a type of an incident, } k \in N\}$

⁶Equation 6.6: $C = \{c_i | c_i, \text{ is a network connection and } i \in N\}$

set I are used by the SLOE algorithm to filter network connections in set C . The filtered connections are those that may be associated with the incident type. This will become clearer as the SLOE algorithm is discussed.

Given a known set of network connection attributes, a subset of attributes needs to be obtained that can best describe or profile a given incident type. From the knowledge base, obtain the attribute-value pairs of these attributes that profile the incident type and use this information to filter connections and sort resulting remote hosts for prioritisation in an investigation.

Given a set of hosts (H) that are connected to the incident scene, the SLOE algorithm computes a subset of hosts (H') that can be prioritised for further investigation after an incident. The cost minimisation algorithm presented in Chapter 6 can be broken down into two parts: the first part is when relevant attributes are inferred based on the incident type. The second part is when the inferred attributes are used to match connections and hence hosts that are connected to the incident scene. The preconditions of the SLOE algorithm with predicate P are represented as follows:

$$P(A, C, I_i) \tag{10.22}$$

where A, C and I_i respectively represent a set of attribute names, a set of connections and a set of incident types ($I_i \in I$).

The postcondition of the SLOE algorithm is a minimised and sorted set of hosts H'_s where:

$$H'_s \subseteq H \tag{10.23}$$

,

and H is a set of all remote hosts that have network connections with the incident scene represented by Equation 6.1⁷. The SLOE algorithm postcondition can then be represented with predicate Q as follows:

$$Q(H'_s \subseteq H) \tag{10.24}$$

To prove the correctness of the SLOE algorithm, this was demonstrated by applying the SLOE algorithm on P in Equation 10.22 leads to Q in Equation 10.24:

$$\{P(A, C, I)\} \text{SLOE} \{Q(H'_s \subseteq H)\} \tag{10.25}$$

⁷Equation 6.1: $H = \{h_i | h_i \text{ is a remote host, } i \in N\}$

Equation 10.25 shows the state of the SLOE algorithm before the SLOE algorithm executes (P) and the state after the SLOE algorithm has terminated (Q), where ‘SLOE’ represents the SLOE algorithm being applied on P to produce Q . States, P and Q represent sets of variables whose values are affected by the SLOE algorithm resulting in consecutive states of P (e.g. $P_0 \dots P_n$) that eventually lead to state Q .

In the next sub-subsection, the assumptions of the proof are presented.

10.3.1.2 ASSUMPTIONS

For the SLOE algorithm, it is assumed that the list of attribute names (x) that can be associated with a network connection exists in the knowledge base, K_B . In other words

$$\forall x \in K_B \exists x \text{ such that } x \subseteq A \wedge x \neq \emptyset \quad (10.26)$$

It is also assumed that information already exists in the knowledge base where incident types (y) are mapped with network connection attribute-values pairs and hence are used to profile an incident or attack type, namely

$$\forall y \in K_B \exists i_k \text{ such that } y \subset I \wedge y \times x \times V \neq \emptyset \quad (10.27)$$

where V is a set of values that can be assumed by each connection attribute defined in Equation 6.5⁸ in Chapter 6. For example in $protocol = 'UDP'$ attribute-value pair, attribute name $protocol \in x$, hence $protocol \in A$ and the value $'UDP' \in V$.

If an incident involves an unauthorised remote access to a local machine instance, attribute name “*username*” would be of interest and it is assumed that the *user* (i.e. $user \in A$) attribute of a connection would already be in the knowledge base with the username used to access the instance. If it is a denial-of-service incident, information would already be in the knowledge base. For instance, the ‘*protocol*’ attribute of a connection with a value of ‘*ICMP*’ is associated with such kind of attacks.

In the next sub-subsection, the chain of reasoning of the proof is presented.

⁸Equation 6.5: $V = \bigcup_i^n V_i = \{x | x \in V_i, i \in N\}$

10.3.1.3 CHAIN OF REASONING

During the reasoning process of the SLOE algorithm, different control structures will be encountered. Statement types and algorithm components are used interchangeably in this chapter to refer to these control structures. These algorithm control structures are sequential statements, conditional statements and loop statements. For this reason, rules that govern each of these statements types will be observed every time such a statement is encountered during the analysis. The rules are as discussed thoroughly in Zaharie [155], Apt et al. [17], Cormen et al. [31] and Levitin [81]. For the convenience of the reader, a summary of these rules is included in Appendix D of this thesis (see Section D.1). Reference to the rules will be made while the chain of reasoning is being presented.

To aid in the presentation of the chain of reasoning, pseudo code is utilised to model the SLOE algorithm and analyse a step-by-step execution of the SLOE algorithm. The aim is to demonstrate how the SLOE algorithm reaches the conclusion with desired results i.e. sorted set of hosts (H'_s). These pseudo codes are presented next.

Algorithm 1, represents a pseudo code that covers process 1 (Get Incident Type) through process 6 (Compute Sorted Hosts Subset) that were discussed in Chapter 6 (see Figure 6.1). The SLOE algorithm takes an incident type t , an IP Address and a set of predefined network connection attribute names, set A as inputs. An incident type is provided by an investigator in the earlier processes of the investigation. The SLOE algorithm lists network connections on the incident scene by calling the function in line 4 in Algorithm 1. A list of attributes is then filtered by the function called in line 6, which corresponds with process 3 (Infer Attributes Related with Incident Type) in Figure 6.1 (Chapter 6).

The loops starting from line 7 through line 26 obtain a set of network connections from the incident scene and for each connection, its set of connection attribute value-pairs is compared with the value-pairs of known attack types in the knowledge base. If the connection has matching attributes, it is included in the set of connections to be investigated further. After those loops are exited, the function “RankHosts” in line 28 is called. The function in line 28 corresponds with processes 5 and 6 in Figure 6.1.

The pseudo code in Algorithm 2 describes the function called in line 6 in the previous algorithm, Algorithm 1. All that the function does is to use the provided suspected attack type as a key to infer relevant attributes and their values from the knowledge base. The output from Algorithm 2 is a reduced set of attributes with their values corresponding to the attack type.

Algorithm 1: Remote hosts list optimisation

Input : Incident type t , incident scene IP address ip and a set of connection attribute A ;

Output: A sorted set of remote hosts IP Addresses H'_s ;

```

1 begin
2    $m \leftarrow |A|$ ;
3    $A', C' \leftarrow \emptyset$ ;
4    $C \leftarrow \text{ListNetworkConnections}(ip)$ ;
5    $l \leftarrow |C|$ ;
6    $A' \leftarrow \text{FilterAttributes}(t, A)$ ;
7   for  $i \leftarrow 0$  to  $l - 1$  do
8      $c \leftarrow C[i]$ ;
9     /*  $A_{match}$  is a list of matching attributes per connection.
10      To be used when computing host relevance weight. */
11     $A_{match} \leftarrow \emptyset$ ;
12     $A_c \leftarrow \text{GetAttributes}(c)$ ;
13    foreach  $key$  in  $A'$  do
14      /* Comparing attribute value in  $A_{match}$  against the value
15       of the same attribute (as  $key$ ) in  $A_c$ . */
16      if  $(key \in (A' \cap A_c)) \wedge (A'[key] = A_c[key])$  then
17         $A_{match} \cup \{key\}$ ;
18      end
19      else
20         $A_{match} \cup \{\}$ ;
21      end
22    end
23    if  $|A_{match}| > 0$  then
24       $\text{SetMatchingAttributes}(c, A_{match})$ ;
25       $C' \cup \{c\}$ ;
26    end
27    else
28       $C' \cup \{\}$ ;
29    end
30  end
31  if  $C' \neq \emptyset$  then
32     $H'_s \leftarrow \text{RankHosts}(C')$ ;
33  end
34  else
35     $H'_s = \emptyset$ 
36  end
37  return  $H'_s$ ;
38 end

```

Algorithm 2: Incident Type Attributes filter

Input : Incident type i_k , Set of connection attributes A and knowledge base K_B ;

Output: A' where A' is attributes key-value pair List;

```

1 begin
2    $A' \leftarrow \emptyset$ ;
   /* map contains a Map of known attributes mapped with values
   corresponding with attack type  $i_k$  obtained from knowledge
   base,  $K_B$ , i.e.  $i_k \in I$  and  $I \times A \times V \in K_B$ . */
3    $map \leftarrow K_B[i_k]$ ;
4   foreach  $a$  in  $A$  do
   /* Check if the current attribute  $a$  exists in  $map$  as a key.
   */
5     if  $a \in map$  then
6        $A' \cup \{a\}$ ;
7     end
8     else
9        $A' \cup \{\}$ ;
10    end
11  end
12  return  $A'$ ;
13 end

```

The pseudo code in Algorithm 3 describes the function “RankHosts” called in line 28 in Algorithm 1. Algorithm 3 determines the list of attributes whose values matched with the suspected incident from a connection c in line 6.

The distance to a host associated with each connection can be determined by utilising IP address geolocation mechanisms proposed by researchers such as Shavitt and Zilberman in [114] and Maziku et al. in [89]. In this thesis, a host that is further away from the incident scene is regarded as being more costly to investigate. This argument is based on the fact that if a remote host is further away from the incident scene, there is a higher probability that the remote host is in a different jurisdiction than the jurisdiction of the incident scene. The fact that an adversary’s host can be the furthest from the incident scene is addressed by attributing weights assigned to its connections before distance is considered. It is for this reason that the distance is inversely proportional to the weight assigned to a host which represents a level of relevance for further investigation. The weight based on distance, ω_1 , is therefore computed using the formula in line 9 in Algorithm 3. The constant k in the formula is used to adjust the threshold on the furthest host that can be investigated. If the

Algorithm 3: Remote Hosts Ranking

```

Input : A reduced Set of network connections  $C'$ ;
Output: A ranked set of remote hosts IP Addresses  $H'$ ;
1 begin
2    $ma \leftarrow \emptyset$ ;
3    $\omega, \omega_1, \omega_2 \leftarrow 0$ ;
4    $H' \leftarrow \emptyset$ ;
5   foreach  $c$  in  $C'$  do
6      $A_{match} \leftarrow \text{GetMatchingAttributeList}(c)$ ;
7      $h \leftarrow \text{GetRemoteIP}(c)$ ;
8      $distance \leftarrow \text{GeographicalDistance}(h)$ ;
9     /* The distance from the host is inversely proportional to
10      the weight assigned to hosts. It is used to compute  $\omega_1$ 
11      together with a constant  $k$  which can be used to increase
12      or decrease a threshold on furthest hosts that can be
13      investigated. */
14      $\omega_1 \leftarrow \frac{k}{distance}$ ;
15     foreach  $a$  in  $A_{match}$  do
16        $\omega_2 \leftarrow \omega_2 + \text{ComputeWeigtValue}(a)$ ;
17     end
18      $\omega \leftarrow \omega_1 + \omega_2$ ;
19      $h[\omega] = \omega$ ;
20     if  $\exists i : H'[i] = h$  and  $H'[i][\omega] < h[\omega]$  then
21       /* Replace existing record of host in  $H'$ ,  $H'[i]$  with
22        current record,  $h$  */
23        $H'[i] \leftarrow h$ ;
24     end
25     else if  $h \notin H'$  then
26        $H' \cup \{h\}$ ;
27     end
28     else
29        $H' \cup \{\}$ ;
30     end
31   end
32    $H'_s \leftarrow \text{Sort}(H')$ ;
33   return  $H'_s$ ;
34 end

```

constant is large, even further hosts are assigned more weight and if it is smaller, it is only closer hosts that are assigned more weight.

The SLOE algorithm being evaluated comprises sequential, conditional and loop statements, as can be seen in Algorithms (1-3). The SLOE algorithm is then analysed

accordingly, while following these statements' respective rules in Section D.1. For the sake of simplicity, in the properties that are presented next, SLOE will be used in reference to the SLOE algorithm.

To prove correctness of the SLOE algorithm, the SLOE algorithm should satisfy the correctness formula:

$$\{true\} \text{SLOE} \{H'_s \subseteq H \wedge (1 \leq i \leq |H'_s| - 1 : H'_s[i][\omega] \geq H'_s[i+1][\omega])\}, \quad (10.28)$$

To prove correctness of the SLOE algorithm, the SLOE algorithm should satisfy the correctness formula (Equation 10.28) in the sense of total correctness. This means that it has to be proved that at the end of its execution, the SLOE algorithm produces a set of remote hosts (set H'_s sorted in a descending order of priority) and that the algorithm terminates. Partial correctness of an algorithm means that it has been proven that the algorithm produces desired results but it has not been proven that it terminates [17, p.70]. On the other hand, total correctness means that both correctness of the algorithm output and the algorithm's ability to terminate have been proven.

Now, going back to the SLOE algorithm, in Algorithm 1 it can be observed that the initial statements are sequential statements (line 2 to line 7). These statements satisfy the following property:

$$P_0 = \{true\} \text{SLOE} \{m = |A|, A' = \emptyset, C' = \emptyset, C = \emptyset, H = \emptyset\}$$

P_0 represents the state of the SLOE algorithm before it starts with its execution. At this stage, all other lists are empty except for the attributes list. The property

$$P_1 \{C = \emptyset\} \text{SLOE} \{|C| \geq 0, l = |C|\}$$

is an updated state of the SLOE algorithm after executing line 4 in Algorithm 1.

$$P_2 \{true\} \text{SLOE} \{l = |C|\}$$

$$P_3 \{A \neq \emptyset \wedge A' = \emptyset\} \text{SLOE} \{A' \subseteq A\}$$

To prove that the property P_3 holds in the SLOE algorithm, an analysis of Al-

gorithm 2 will be carried out as this property involves calling the ‘FilterAttributes’ function as seen on line 6 in Algorithm 1.

Algorithm 2 comprises a loop starting from line 4. The loop is made up of a conditional statement. The precondition of the loop is $\{a \in map \vee a \notin map\}$ and the postcondition is $\{A' \subset A\}$. By applying the conditional statements rule in Section D.1, this can be formalised as follows:

$$\{a \in map\} \text{ SLOE } \{A' \cup \{a\}\} \text{ and } \{\neg(a \in A')\} \text{ SLOE } \{A' \cup \{\}\} \quad (10.29)$$

Since $\forall a \in A', \nexists a : a \notin A$, then,

$$(\{A' \cup \{a\}\} \cup \{A' \cup \{\}\}) \subseteq A \quad (10.30)$$

This means that since only elements from set A are included only once into set A' or not at all, it follows that set A' will always be a subset of set A . The condition inside the loop in line 5 of Algorithm 2 is correct in the sense of partial correctness. This satisfies the postcondition in property P_3 .

Next, the correctness of the loop in line 5 of Algorithm 2 is proved. To prove correctness of the loop, a loop invariant has to be identified first. The precondition of the loop is $\{A' = \emptyset\}$ and the postcondition is $\{A' \subset A \wedge A' \neq \emptyset\}$. After the first iteration of the loop, $(|A'| = 0) \vee (|A'| = 1)$, and after the second iteration of the loop, $(|A'| = 0) \vee (|A'| = 1) \vee (|A'| = 2)$ and so on - until the last element of set A is reached.

If an iteration counter i is maintained, the stopping condition would be when $i = |A|$ as the loop executes for each element of A . Once the stopping condition is reached, it can be seen that the first part of the postcondition ($A' \subset A$) holds. $A' \subset A$ is the loop invariant as it is true before, during and after the execution of the loop. From the assumption in Equation 10.26, it can deduced that $A' \neq \emptyset$. In other words, at some point during the loop iterations, attribute name(s) related to the incident type will be encountered, which will then be included in A' . At this stage, property P_3 still holds - which means that Algorithm 2 is still partially correct. The termination function of the loop is $t(i) = |A| - i$, where $|A|$ is the size of set A and i is the iteration counter. Clearly, $t(i)$ is a decreasing function since $|A|$ is constant, i.e. $t(i+1) < t(i)$, which satisfies the loop statement rule number 4 in Section D.1 (see Appendix D). By considering the sequential statements rule for the entirety of Algorithm 2, the loop leads to the state in line 12 where the subset of attributes is returned. This is the point where Algorithm 2 terminates.

Given that all components of Algorithm 2 terminate, it follows that Algorithm 2 eventually terminates. Algorithm 2 is therefore correct in the sense of total correctness.

As Algorithm 2 terminates, the outcomes from it are assigned to set A in Algorithm 1, line 6. By sequential rule, the set A' assignment is followed by the loop starting from line 7 where the set A' is utilised. To prove correctness of the loop in line 7 in Algorithm 1, the proof starts by proving the correctness of the inner loop starting from line 11. The loop in line 11 has a conditional statement that starts from line 12 and ends at line 17. The conditional statements' rule applies in this case. The precondition of the conditional statement is $\{A_{match} = \emptyset \wedge A' \neq \emptyset\}$ and the postcondition is $\{A_{match} \subseteq A' \wedge A_{match} \supseteq \emptyset\}$.

As seen in the derivation of Equation 10.29, the two conditions in line 12 through line 17 of Algorithm 1 (in this case the following property) holds, namely:

$$\begin{aligned} & \{(key \in (A' \cap A_c)) \wedge (A'[key] = A_c[key])\} \text{ SLOE } \{A_{match} \cup \{key\}\} \text{ and} \\ & \{(\neg(key \in (A' \cap A_c)) \wedge (A'[key] = A_c[key]))\} \text{ SLOE } \{A_{match} \cup \{\}\} \end{aligned} \quad (10.31)$$

and $((A_{match} \cup \{key\}) \cup (A_{match} \cup \{\})) \subseteq A'$ where $A'[key], A[key] \in V$ and V is a set of values from Equation 6.5⁹. Therefore the conditional statement in line 12 through line 17 of Algorithm 1 satisfies the postcondition and is therefore correct in the sense of partial correctness. Next, the correctness of the loop starting from line 11 through line 18 in Algorithm 1 is proved. The loop statements rule in Section D.1 applies in this case.

To prove the correctness of a loop, a loop invariant and a termination function is required. In each iteration of the loop, set A match is the only updated variable and set A' does not change. The precondition of the loop is $\{A_{match} = \emptyset \wedge A' \neq \emptyset\}$ and the postcondition is $\{A_{match} \supseteq \emptyset \wedge A' \neq \emptyset\}$. Since $\forall a \in A_{match}, a \in A'$. Therefore, $A_{match} \subseteq A'$ is the invariant of the loop as it is evaluated to be true before, during and after the execution of the loop.

Now - to determine the termination function of the loop, an execution counter for the loop is set, i.e. $i = 0$. The integer i before the loop executes for the first time is set to 0. The loop iterates until the elements of set A are exhausted, which result in $i = |A'|$ when the loop terminates. Therefore, the termination function is $t(i) = |A'| - i$ and it is clearly a decreasing function as shown in Figure 10.2 with

⁹Equation 6.5: $V = \bigcup_i^n V_i = \{x | x \in V_i, i \in N\}$

$|A'| = 1000000$. i.e., $t(i+1) < t(i)$. The loop from line 11 through line 18 in Algorithm 1 is correct in the sense of total correctness as its invariant has been identified and it has a decreasing termination function.

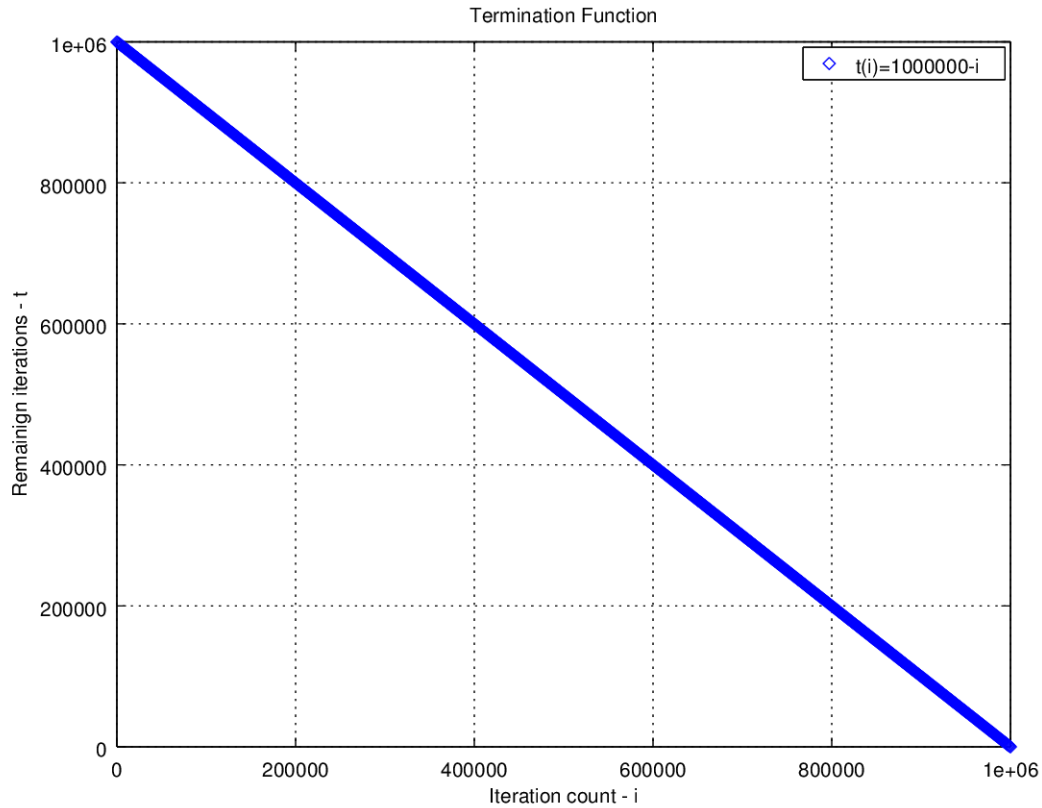


Figure 10.2: Termination function

Another component within the loop from line 7 through line 26 of Algorithm 1 is the conditional statement from line 19 through line 25 of Algorithm 1. This conditional statement corresponds with Equation 6.12¹⁰ in Chapter 6. By following the procedures to derive Equation 10.29 and Equation 10.31, it can be shown that the property:

$$\{C' = \emptyset \wedge C \neq \emptyset\} \text{ SLOE } \{C' \cup \{c\}\} \text{ and } \{\neg(A_{match} = \emptyset)\} \text{ SLOE } \{C' \cup \{\}\} \quad (10.32)$$

holds for the condition in line 19 of Algorithm 1 where the precondition of the condition statement is $\{C' = \emptyset \wedge C \neq \emptyset\}$ and the postcondition is $\{C' \subseteq C\}$. The

¹⁰Equation 6.12: $k_{C_j}(c) = \begin{cases} 1, & \text{if } c \in C_j \\ 0, & \text{if } c \notin C_j \end{cases}$

conditional statement is therefore correct in the sense of partial correctness, since both conditions lead to the postcondition $\{C' \subseteq C\}$. Now that the components of the loop from line 7 through line 26 of Algorithm 1 have been proved to be correct, It remains to be proved that the loop itself is correct and this is attempted next. The precondition of the loop in line 7 is $\{C \neq \emptyset \wedge C' = \emptyset \wedge A' \neq \emptyset\}$ and the postcondition is $\{C' \subseteq C\}$. Clearly, $C' \subset C \wedge A' \neq \emptyset$ is the loop invariant and the termination function is $t(i) = l - i$ where i is the iteration counter and l is the size of set of connections, set C . Since $t(i)$ is a decreasing function and the postcondition holds when the loop terminates at $t(i) = 0$, the loop is correct in the sense of partial correctness. The termination of the loop leads to the conditional statement in line 27 through line 32 of Algorithm 1, which follows the loop in the sequence. The first condition of the conditional statement calls an external function in line 28 of Algorithm 1. The function being called is Algorithm 3. The correctness of Algorithm 3 will be proved first before completing the proof of the conditional statement in line 27 of Algorithm 1. In Algorithm 3, leading up to the loop in line 5, there are sequential statements that are declarations. To prove the correctness of the loop in line 5, the nested loop inside it in line 10 needs to be proved first. The precondition of the nested loop is $\{\omega_2 > 0\}$ and the postcondition is $\{\omega_2 > 0\}$. The loop invariant is:

$$\omega_2 = \sum_{i=1}^{|A_{match}|} ComputeWeightValue(a_i) \quad (10.33)$$

where $a_i \in A_{match}$. The termination function of the loop in line 10 is $t(i) = |A_{match}| - i$ and it is a decreasing function. The loop invariant in Equation 10.33 holds at the termination of the loop when $t(i) = 0$. Since it is given that $A_{match} \neq \emptyset$ this means that $\omega_2 > 0$ as ω_2 values are computed from the elements of A_{match} . Therefore, the loop in line 10 in Algorithm 3 is correct in the sense of total correctness.

The termination of the loop leads to sequential statements in line 13, line 14 and eventually the conditional statement in line 15 - all in Algorithm 3. The conditional statement in line 15 corresponds with Equation 6.13¹¹ in Chapter 6. The correctness of the three sequential statements is clear as each sequential statement leads to the next, as its postcondition becomes a precondition on the next statement:

$\{\omega_1 \geq 0 \wedge \omega_2 > 0\} \text{SLOE} \{\omega > 0\}$ in line 13 and,
 $\{h[\omega] = 0\} \{h[\omega] = \omega\} \text{SLOE} \{h[\omega] > 0\}$ in line 14

¹¹Equation 6.13: $l_{H_j}(h) = \begin{cases} 1, & \text{if } h \in H_j \\ 0, & \text{if } h \notin H_j \end{cases}$

The precondition of the conditional statement is therefore $H' \supseteq \emptyset$ and the postcondition is $h \in H'$. In other words, in either of the conditions, at the end, host h becomes part of set H . The three conditional statements from the condition in line 15 through line 20 hold as follows:

$$\begin{aligned} & \{(H' \supseteq \emptyset) \wedge (\exists i : H'[i] = h \wedge H'[i][\omega] < h[\omega])\} \text{SLOE}\{H[i][\omega] = h[\omega]\}, \\ & \{(H' \supseteq \emptyset) \wedge (h \notin H')\} \text{SLOE}\{H' \cup \{h\}\} \text{ and} \\ & \{(H' \supseteq \emptyset) \wedge (\neg(\exists i : H'[i] = h \wedge H'[i][\omega] < h[\omega]) \wedge (h \notin H'))\} \text{SLOE}\{H' \cup \{h\}\} \end{aligned} \quad (10.34)$$

In all three cases, the postcondition holds, namely $\{h \in H'\}$. This is because in the three cases the host h gets updated if it exists in H' . Lesser weight gets added if it is not in H' or nothing is added if it is already in H' . The rule on conditional statements is therefore satisfied and it be concluded that the conditional statement from line 15 through line 20 of Algorithm 3 is correct in the sense of partial correctness.

Now the correctness of the loop from line 5 through line 24 is proved. The precondition of the loop is $\{H' = \emptyset \wedge C' \neq \emptyset\}$. The postcondition is $\{H' \supseteq \emptyset\}$ and the loop invariant is:

$$H' = \bigcup_{i=1}^{|C'|} h_i = \{h_i | c_i \implies h_i \text{ and } H'[i] \neq H'[j], i, j \in N\} \quad (10.35)$$

Equation 10.35 means that in each iteration, H' remains a set of all hosts inferred from each connection in C' with no repeats of a host h which may result from a host having multiple connections in the cloud incident scene, i.e. $|H'| \leq |C'|$. The termination function is $t(i) = |C'| - i$ where i is the loop iteration counter. $t(i)$ is clearly a decreasing function. The invariant in Equation 10.35 holds as the loop terminates at $t(i) = 0$, in other words when $i = |C'|$. The loop from line 10.35 through line 24 is correct in the sense of total correctness.

On termination of the loop that ends in line 24, the loop's postcondition $\{H' \supseteq \emptyset\}$ becomes a precondition in line 25 of Algorithm 3. Therefore, the property

$$\{H' \supseteq \emptyset\} \text{Sort}(H') \{H'_s : H'[i][\omega] = r \wedge H'[i+1][\omega] = r \wedge r \geq z \wedge H' \equiv H'_s\} \quad (10.36)$$

where ω is a weight assigned to a remote host $H'[i]$ from Equation 6.17¹² (see Chapter 6) in line 25 of Algorithm 1 holds. The property means that set H'_s is such that the hosts are arranged in ascending order of the respective weight (ω) values. H'_s is equivalent to H of H' and vice versa.

After assigning the outcome from the Sort function to set H'_s , Algorithm 3 terminates and returns the sorted set H'_s and terminates in line 26. Since all components of Algorithm 3 terminate, it follows that Algorithm 3 is correct in the sense of total correctness.

Algorithm 3 is called from line 28 of Algorithm 1. That is, it is called from the conditional statement from line 27 through line 32 of Algorithm 1. This conditional statement has to be proved. The precondition of the conditional statement is $H'_s = \emptyset$ and the postcondition is $H'_s \supseteq \emptyset$. The conditional statement has the property

$$\begin{aligned} &\{H'_s = \emptyset \wedge C' \neq \emptyset\}SLOE\{H'_s \supseteq \emptyset\} \text{ and} \\ &\{H'_s = \emptyset \wedge \neg(C' \neq \emptyset)\}SLOE\{H'_s = \emptyset\} \end{aligned} \quad (10.37)$$

which satisfies the conditional statements rule in Section D.1. In both scenarios of the conditional statement, $H'_s \supseteq \emptyset$, which satisfies the postcondition. The conditional statement is therefore correct in the sense of partial correctness. This concludes the proofs of correctness of all the components of Algorithm 1. Algorithm 1 terminates in line 33 with $H'_s \subseteq H$.

The SLOE algorithm is therefore correct in the sense of total correctness. The SLOE algorithm always produces expected outcomes; since it terminates, it is guaranteed not to prevent the entire cloud investigation process from continuing. In the next section, the evaluation of the SLOE algorithm continues, but now the focus is on its time efficiency.

10.3.2 TIME EFFICIENCY EVALUATION

The SLOE algorithm is utilised in the CFaaS cloud forensics process during Evidence Identification as part of the Determine Remote Hosts process. Looking at the cloud forensic process diagram in Figure 8.3 (see Chapter 8), it can be observed that if the SLOE algorithm is not scalable, the entire cloud forensic process would be blocked. In this section, scalability of the SLOE algorithm is evaluated by means of time efficiency evaluation.

¹²Equation 6.17: $w \in D$

The analysis of the SLOE algorithm begins with the initial part of the algorithm. The steps that are used to evaluate the time efficiency of the algorithm are as summarised by Levitin in [81, p.62]:

Step 1: Deciding on the parameter indicating the input size to the algorithm.

Step 2: Identifying the basic operation of the algorithm.

Step 3: Checking if the algorithm's basic operation is executed and depends in the algorithm's inputs only. Otherwise best case, average case and worst case evaluations have to be considered.

Step 4: Setting up the sum expressing the number of times the algorithm's basic operation is executed.

Step 5: finding the closed form formula for the count or the order of growth using standard formulas.

A time efficiency analysis of the SLOE algorithm can be performed easily with the pseudo codes presented from Algorithm 1 through Algorithm 3. The time efficiency analysis has to do with determining the order of growth of the execution time of the algorithm, given an increase in the size of input. For instance, if the execution time of an algorithm grows exponentially (e.g. $t = 2^n$) as the size of its input grows, and the execution time of the second algorithm grows in a logarithmic order (e.g. $t = \log_2 n$). With an increase in input size n , the latter grows slower and hence is regarded as being more scalable. The shapes of these functions can be seen in Figure 10.3 where $f(n) = 600\log_2 n$, $g(n) = 250n$ and $h(n) = \frac{1}{100}2^n$. The constants on the curves were chosen in such a way that all three graphs are visible on the chosen scale of Figure 10.3. Otherwise they have no significant meaning, and they are only for demonstration purposes. Whichever choices of constants are made, at some point(s) $h(n)$ will start to grow dramatically and go far above the rest - as can be seen in Figure 10.3.

The time efficiency analysis also known as Big-O analysis is discussed thoroughly in the available literature, including by Levitin in [81] and Skiena in [128].

For the convenience of the reader, a Big-O analysis summary is also provided in Section D.2 of Appendix D. Now, following the analysis steps provided above, the analysis is presented as follows.

Step 1: *Deciding on the parameter indicating the input size to the algorithm*

The first step is to identify parameters to the SLOE algorithm and the input size.

inside another loop that is indented to the level of the line is in level 2. The loops that are at level 2 are the loops starting from line 11 in Algorithm 1 and starting from line 10 in Algorithm 3. Algorithm 2 and Algorithm 3 are called by Algorithm 1 from line 6 and line 28 of Algorithm 1 respectively. If the two algorithms were to be incorporated into Algorithm 1, the loops in the respective algorithms would still be on the same level as they are.

Since Algorithm 2 and Algorithm 3 are called from within Algorithm 1, if they would be substituted back into Algorithm 1, the loops in the respective algorithms would be on the same levels as they appear in each algorithm. Therefore, the operations in line 12 of Algorithm 1 and line 11 of Algorithm 3 are both innermost operations.

The execution times of the innermost operation in Algorithm 1 is proportional to the size of set A and the size of set C . The execution time is therefore approximately $(l \times m)$ where l and m are respective sizes of set C and set A as defined in Algorithm 1. This operation will then be compared with the operation in line 11 of Algorithm 3 to determine the basic operation of the entire algorithm.

It has already been shown that the operation in line 12 of Algorithm 1 executes $(l \times m)$ times. To determine the number of times that the operation in line 11 of Algorithm 3 executes, the sizes of set C and A_{match} have to be used. Set A_{match} has a list of attributes that have matched the profile of the incident type. But $A_{match} \subseteq A' \subseteq A$. Therefore, the maximum size that set A_{match} can possibly have is the same as the size of set A . The operation in line 11 of Algorithm 3 therefore also executes approximately $(l \times m)$ times.

Since the two operations in line 11 of Algorithm 3 have a similar maximum number of times that they can execute, either of them can be considered a basic operation.

Step 3: Checking if the algorithm's basic operation is executed, depends on the algorithm's inputs only

In Step 2 it was shown that the number of times the basic operation executes is $l \times m$ time. l and m are both input sizes to the SLOE algorithm. For this reason, the three different cases (best-case, average-case and worst-case analysis) will not be considered in the analysis.

Step 4: Setting up the sum expressing the number of times the algorithm's basic operation is executed

The fourth step involves expressing the number of times that the basic operation is executed in a summation form. The base operation of the SLOE algorithm has been

shown to be running $(l \times m)$ times,

$$\begin{aligned}
 T(l, m) &= \sum_{i=0}^{l-1} \sum_{i=0}^{m-1} 1 \\
 &= \sum_{i=0}^{l-1} m \\
 &= lm
 \end{aligned} \tag{10.38}$$

where $T(l, m)$ is the time complexity required for an input size of attributes m and input size of connections l . from the time complexity obtained in this step, a closed form formula for the order of growth is next determined in Step 5.

Step 5: *Finding the closed form formula for the count or the order of growth, using standard formulas* The number of connection properties can only be up to a limited number, k ,

$$\begin{aligned}
 \lim_{m \rightarrow k} (\lim_{l \rightarrow \infty} T(l, m)) &= \lim_{m \rightarrow k} (\lim_{l \rightarrow \infty} lm) \\
 &= \lim_{l \rightarrow \infty} kl \\
 &= \infty
 \end{aligned} \tag{10.39}$$

where $m \leq k$ and k is a constant.

The significant variable in this algorithm is therefore l , the number of connection attributes in the incident scene. The cost function can be represented by

$$T(l, m) \approx l \tag{10.40}$$

Given the insignificance of m when l is too large, T can be considered a function of l only. For example, if $m = 3$ and $l = 1$ -million, this means that the innermost loop of the SLOE algorithm will be executed $l \times m = 3$ -million times. Even though l can take small values, it is unbounded above, i.e. $0 \leq l < \infty, l \in N$. For m on the other hand, if the network connection attributes to be considered are the 41 attributes in KDDCUP [50], the maximum value that m can have is 41, which is a number of attributes extracted from the headers of network packets. For this reason, function

$T(l, m)$ can safely be represented as follows:

$$T(l) = l \quad (10.41)$$

This represents the order of growth of the execution time with respect to input size of the SLOE algorithm. In this case, the SLOE algorithm time efficiency is directly proportional to the number of connections in the incident scene. Therefore,

$$g(l) = l \implies f(l) \in O(l) \quad (10.42)$$

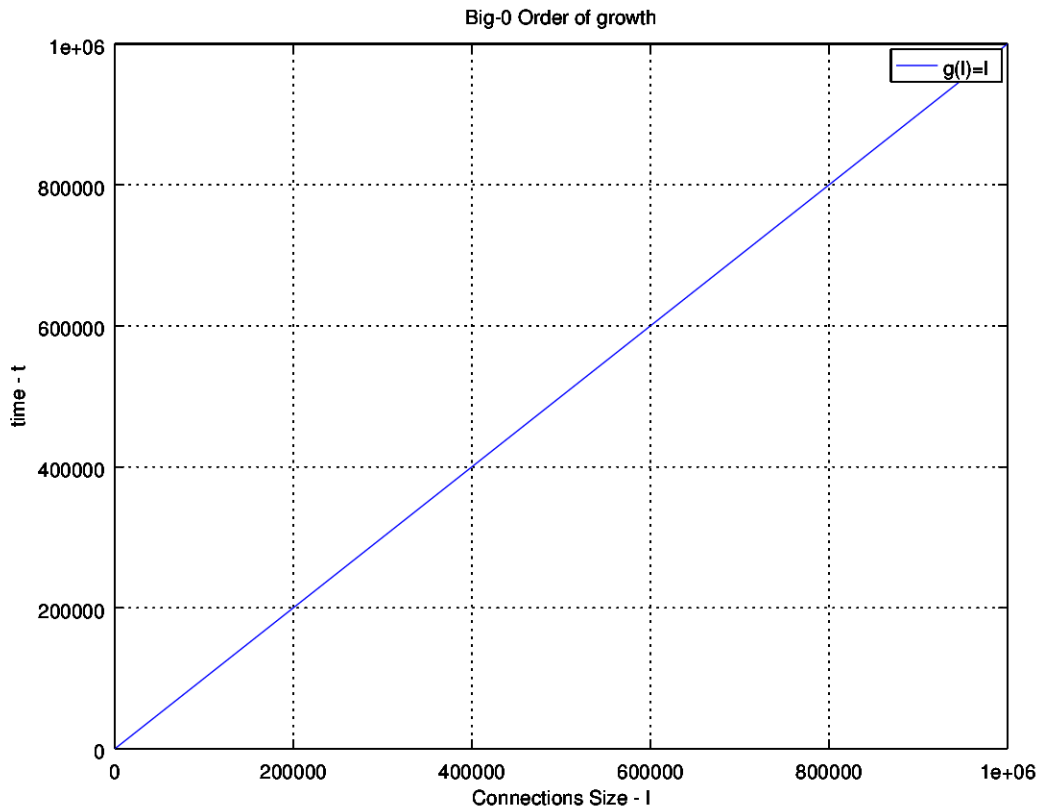


Figure 10.4: SLOE algorithm Time Complexity

where $f(l)$ represents a time complexity function of the SLOE algorithm. The different complexity functions in Big-O analysis where examples include $O(n)$, $O(\log n)$, $O(n^2)$, $O(n!)$ and others where in this case, the different notations respectively correspond with linear, logarithmic, quadratic and factorial functions. n represents an algorithm's input size. The rate at which these respective functions grow with the size of an input can be seen in Section D.2. From Equation 10.42, it is clear that $f(l)$ corresponds with the $O(l)$, i.e. it is a linear complexity function.

According to this analysis, the SLOE algorithm presented in Chapter 6 is scalable as the algorithms' time complexity ($O(l)$) grows at the same proportion as the input size (see Figure 10.4).

The third artefact provided in this research is the CFaaS service model architecture. In the next section, an evaluation of this artefact is provided.

10.4 ARCHITECTURAL DESIGN EVALUATION

This section evaluates the architectural design of the CFaaS service model based on the non-functional requirements by using an Architecture Trade-off Analysis Method (ATAM)[66], a standard framework used to evaluate architectural design in software engineering. In this case, evaluating the CFaaS service model architecture helps identify the extent to which the design approaches adopted in CFaaS affect the requirements presented in Chapter 4. The ATAM comprises the following eight steps that culminate in the presentation of results:

- Step 1: Presentation of ATAM;
- Step 2: Presentation of the Business Drivers;
- Step 3: Presentation of the Architecture;
- Step 4: Identification of Architecture Approaches;
- Step 5: Generating Quality Attribute Utility Tree;
- Step 6: Analysis of Architecture Approaches;
- Step 7: Brainstorming and Prioritisation of Scenarios;
- Step 8: Analysis of Architecture Approaches and
- Step 9: Presentation of the Results.

Step 1 of the ATAM evaluation involved the presentation of ATAM to the research team. This involved one-on-one sessions with respective research team members during which the ATAM evaluation process steps were discussed.

As part of step 2, which is the presentation of business drivers, research was carried out to investigate existing digital forensic tools that are meant for cloud environments. Many of the tools and techniques were found to be suitable for conventional computing environments and could not be applied directly in cloud environments. Step 3 involves

the presentation of the architecture as was done in Chapter 7 where all relevant diagrams and views of the architecture were presented. Chapter 7 therefore concludes step 3 of ATAM evaluation.

Step 4 deals with the identification of architectural approaches. The hierarchical pattern is chosen and used to allow maintainability with low effort. The component configuration pattern is applied to allow decoupling (independence of individual components). Deploying a service in the cloud mainly addresses scalability issues and availability of the service. Meteor [7] is used to develop the CFaaS service and the front-end. The choice of this JavaScript-based technology was due to its ability to separate server-side and client-side components in CFaaS. The jBPM (Java Business Process Management) workflow process engine is used to implement and execute a standard digital forensic process template. jBPM enforces role-based access to cloud forensic investigation tasks as discussed in Chapter 8. This aspect is important for the security requirement in CFaaS.

Step 5 concerns generating a quality attributes utility tree. The attributes or non-functional requirements that were used to generate the quality attribute utility tree are as presented in Section 4.2.2.2. The quality attribute utility tree provides a way to express in context the quality attributes of an architecture, including the scenarios, and it is as represented in Figure 10.5 and explained in Table 10.3. In the table, Column 1 represents the attribute and the second column provides identifiers for the refinement of that attribute. The fourth column is a scenario associated with an attribute refinement. H, M and L in the importance and difficulty columns stand for High, Medium and Low respectively.

As an example, flexibility, the first attribute in Table 10.3, is refined and the first refinement is assigned ID, F1. The refinement is Ability to add a new digital forensic standard process. The scenario associated with this refinement is when an investigator needs to deploy a new digital forensic standard process template within a shortest time possible. The importance of this task is M (medium) as digital forensic process standards do not evolve frequently. The effort to implement it is H (high) because though it is easy to import a new process template in the CFaaS task server, in the current implementation of CFaaS, scripts and task forms have to be developed from scratch for tasks that did not previously exist in the CFaaS task server. The rest of the table is interpreted in the same way.

Step 6 involves analysing the architectural approaches. Analysis tables are presented for all the considered attributes. An example of the analysis tables is in Table 10.4 where R, NR, S and T stand for Risk, Non-risk, Sensitivity and Trade-off re-

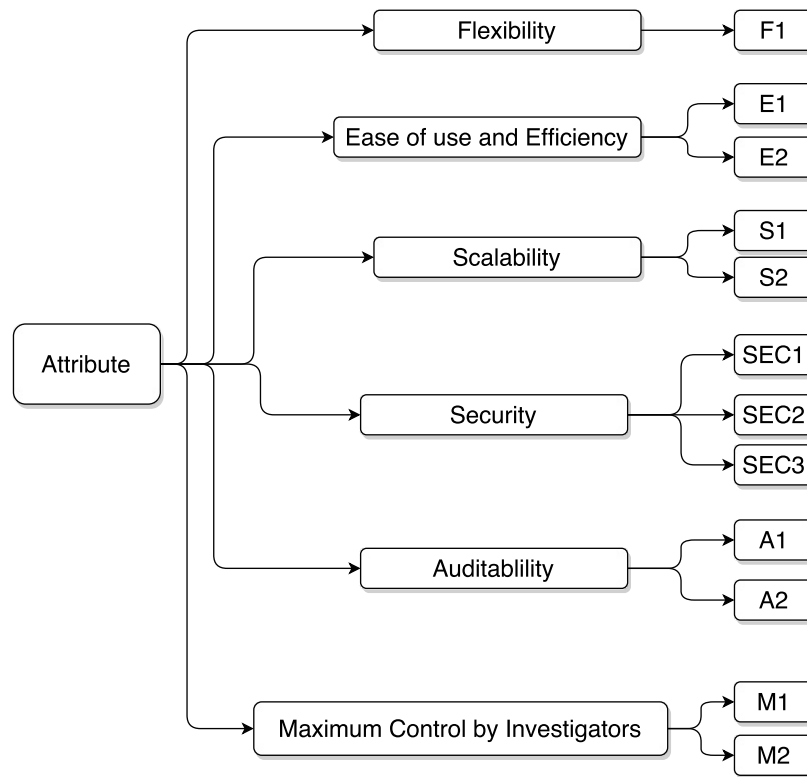


Figure 10.5: Quality Attributes Tree

spectively. R in the table is the risk for the system that is associated with a certain architectural approach. Non-risks are the benefits or advantages of using a certain architectural approach. Sensitivity is when an architectural approach directly affects other attributes of the architecture. Tradeoffs are those attributes that are sensitive to an architectural approach and are compromised as a result of the adoption of the architectural approach. Table 10.4 represents an analysis of the flexibility attribute and focuses on plugging in a new standard digital forensic process template. The table is interpreted as follows:

R1 - communication between the CFaaS service component and the Tasks server takes place through HTTPS. As a risk, if communication would break down between the two components, CFaaS will not be usable. NR1 - having the task server as a separate entity allows the replacement of the tasks server component without affecting the whole digital forensic service system. R2 - JBPM, the workflow engine used as task server expects all manual tasks to have corresponding task forms to be loaded on the system for a successful process execution. If a corresponding task form is not available, JBPM renders a blank form and the task cannot be completed.

Table 10.3: Quality attribute utility tree

Attribute	ID	Attribute Refinement	Attribute Scenario	Importance	Difficulty
flexibility	F1	Ability to add a new digital forensic standard process	Deploy a new digital forensic standard process template in minutes	M	H
Ease of Use and Efficiency	E1	User Understands all features	The graphic user interface windows are clear for an investigator to use their intuition to use and navigate	H	M
	E2	Minimised number of button clicks to execute a specific task	An investigator restart investigation process from a specific sub-process through a single click of a button	M	M
Scalability	S1	Consistent data transfer rate	Transferring 50GB of an evidence data takes 1ms, it should take time proportional to data size to transfer 1TB of evidence data	H	H
	S2	faster processing.	When performing a string search on evidence data, results should be returned within 2 seconds	H	H
Security	SE1	Authentication	Investigators using the digital forensic service instance must be authenticated	H	M
	SE2	Authorisation	An investigator only carries out tasks they are authorised to perform.	H	M
	SE3	Availability	Information residing in the cloud is volatile, the service must be available 99.99% for real-time investigation.	H	H
Audit-ability	A1	Logging	All actions performed in the service during the life-time of an investigation are logged.	M	M
	A2	Log retrieval	Logs for auditing are retrieved through click of a button	M	M
Maximum Control by investigators	M1	Access to any component	Investigator technician can access and configure platform components of the service.	M	L
	M2	Access to logs as low as possible in the service stack.	The auditor must be able to access application log from the platform.	M	L

More analysis tables that correspond with the attributes in Figure 10.5 and Table 10.3 are presented in Section D.3 (see Appendix D). The analysis revealed that the CFaaS service model architecture has trade-offs - which are discussed next.

The architecture addresses the third research question listed in Section 11.1. During the evaluation of the architecture using ATAM, risks and benefits in the architecture were identified. The risks identified during the analysis of the architecture are breaks in communications among service components. The LDAP directory service and JBPM task server run as remote components and are accessed via TCP. A break in the HTTPS communication or limited bandwidth may disrupt the whole digital forensic service functioning. This trade-off is, however, necessary as it is less critical than if a LDAP [75] would run on the CFaaS cloud instance. A compromise on the CFaaS cloud instance would affect the security component (LDAP) as well.

Table 10.4: flexibility: F1 - Analysis

Attribute	flexibility		
Scenario #	F1 - Deploy a new digital forensic standard process template in lesser than 5 seconds.		
Scenario	Design time		
Stimulus	Need to use an alternative or updated digital standard forensic process		
Response	Deploy the standard process template with no side effects		
Architectural Decisions	Risk/Non-Risk	Sensitivity	Trade-off
Component architecture	R1,NR1		
jBPM	R2		
Reasoning: The standard digital forensic service templates need not to be tightly coupled with the whole forensic service system. The need to replace the template is more frequent than the need to upgrade the whole system. Updating the whole system would require more time than replacing a process template. The component architecture allows the use of a generic work-flow process engine such as the JBPM process engine.			

Storing digital evidence in the cloud increases the attack vectors through which adversaries can compromise evidence data. Reduction of the attack surface is traded for performance and scalability in the cloud due to the large amount of data that is expected to be processed in the cloud investigation. This problem can, however, be addressed by adopting a private cloud deployment in CFaaS as is the case with CFaaS current deployment.

By deploying the service in the cloud, logs become distributed, which adversely affect the auditability of the digital forensic service. This is also traded-off for scalability of CFaaS.

In conclusion, the preliminary evaluation of the service revealed strengths and weaknesses of the architecture that can still be improved. The remaining steps of ATAM that can still be carried out are expected to reveal more strengths or weaknesses of CFaaS. For the purposes of this thesis, however, the current steps carried out in ATAM are sufficient and the rest of the steps will be conducted as the research proceeds.

10.5 IMPLEMENTATION EVALUATION

This section presents the testing procedures carried out on the CFaaS service. To use the service to investigate a cloud-based incident, a compromised virtual machine was deployed in a cloud environment. The compromised virtual machine was obtained

from a company network and used to create a cloud image. The virtual machine image was then deployed in a CloudStack environment and a virtual instance was booted up from the compromised virtual machine image. In practice, this can be regarded as an off-line analysis of an incident scene, but here it was carried out with CFaaS for demonstration purposes.

The incident on the compromised company virtual machine was discovered when a complaint was received alleging that their network was being flooded by ping requests from the virtual machine. The initial suspicion was that an adversary has gained access to the virtual machine and is using it to probe other potential victims.

Given this scenario, CFaaS was used to investigate the incident. The purpose of the investigation was to uncover how the attacker managed to gain access to the virtual machine, what components of the virtual machine they compromised (if any), to uncover back-doors used (if any) and to identify the attacker or the attack-originating host.

To demonstrate how the investigation was conducted on the cloud instance, the forms that were required as part of each manual task during the investigation were completed. On tasks that were carried out by services, snippets from the outputs of the services were presented. Snippets of the report prepared once the investigation had been concluded are also presented in the form of a task input summary.

After successfully registering with a CFaaS service, an investigator lands on a welcoming page in Figure 10.6, from where they can start performing a task for which they purpose to use the service.

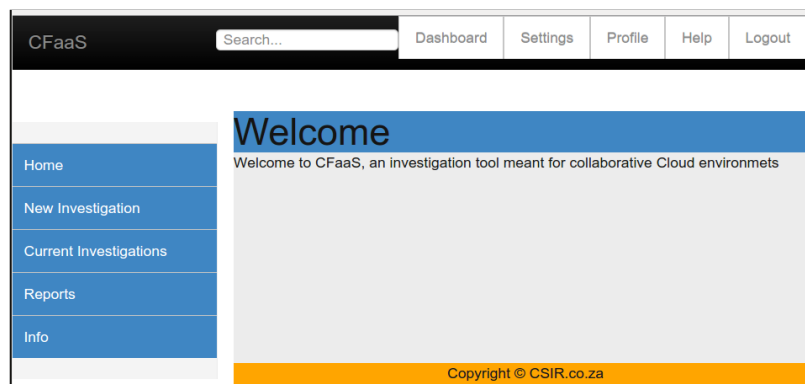


Figure 10.6: Welcome Page

Figure 10.7, a completed form used to initialise an investigation is presented.

Figure 10.7 presents a completed form used to initialise an investigation. In this figure it is assumed that an investigator has already been registered with CFaaS, (as discussed in Section 7.2.4 and summarised by Figure 7.11 where a process view of

Home

New Investigation

Current Investigations

Reports

Info

Investigation Initialization Form

Case ID

Case Description

A complaint was received from a complainant outside MER network that pings were being received from an IP address belonging to MER. MER requests for an investigation to be conducted on a cloud based instance with the IP address in question. The investigation seeks to uncover how the ping requests originate and to also uncover any other compromises that may exist on the instance.

It is suspected that some one may be misusing the cloud instance to probe remote hosts for a possible attack on the remote hosts. A conviction is possible if a perpetrator would be identified after the investigation.

Figure 10.7: New Investigation Initialisation

CFaaS was discussed). On the investigation window, an investigator selects “New Investigation” menu item, which opens the form in Figure 10.7. After completing the form as shown, the investigator submits it and CFaaS creates an investigation process instance with a list of consecutive tasks to be completed by authorised investigators and by services.

Once the case has been initialised, the first task in the investigation process gets activated and it becomes available to all investigators who are authorised to execute it. The first task, Register Case, was executed by Hein Venter and a summary of the task outcomes is shown in Figure 10.8. It should be noted that the information provided in the Register Case task form is the same as the information provided in the case initialisation form. While completing the registration task, an investigator can provide a description of the case being investigated (to the best of their knowledge) and according to stipulated investigation agency rules.

Figure 10.9 represents outcomes from a task where an investigator provides a description of the incident. In this case, the description includes contents of the notification to be sent to investigators once malicious activities have been observed. The description also includes what is being observed by an individual who currently has access to the incident scene. In this case, a snapshot is provided of what was observed at the edge router of the incident scene.

Figure 10.10 contains a summary of the details provided by an investigator while completing the Establish Secure Connection task. These are credentials to be used to connect to the incident scene. In this case it is assumed that an incident is occurring in only one cloud-based instance. In reality, an incident may involve multiple

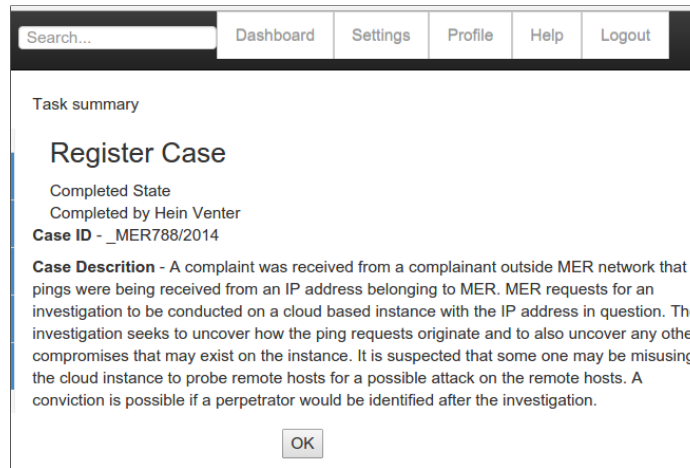


Figure 10.8: Case Registration Summary

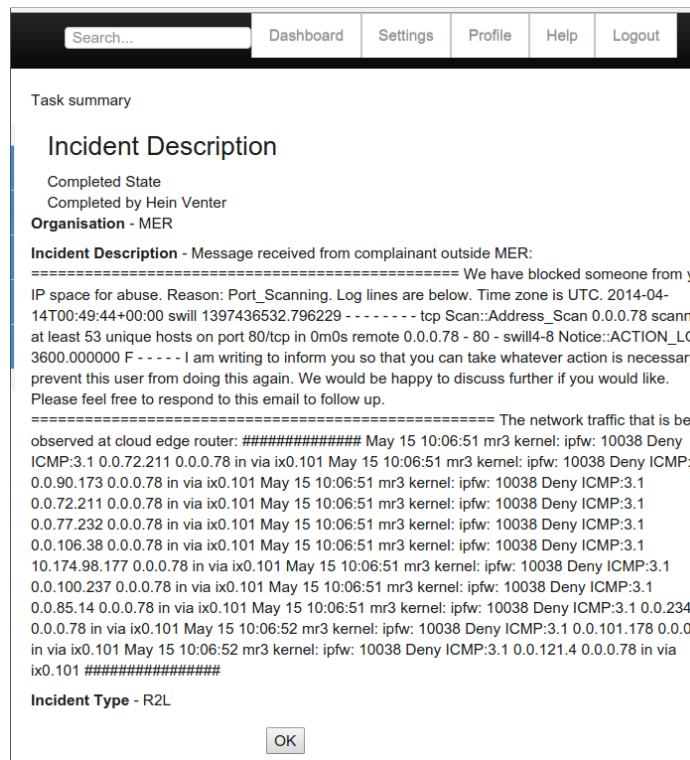


Figure 10.9: Incident Description Summary

cloud instances. Credentials of such additional instances would be provided by an investigator through this task.

Figure 10.11 summarises the task outcomes after a service task was carried out as they would appear on the investigation report. The Secure Connection executed in this case connects to the incident scene. Ideally, this task connects to the incident scene and exchanges SSH Keys for connections by other subsequent investigation

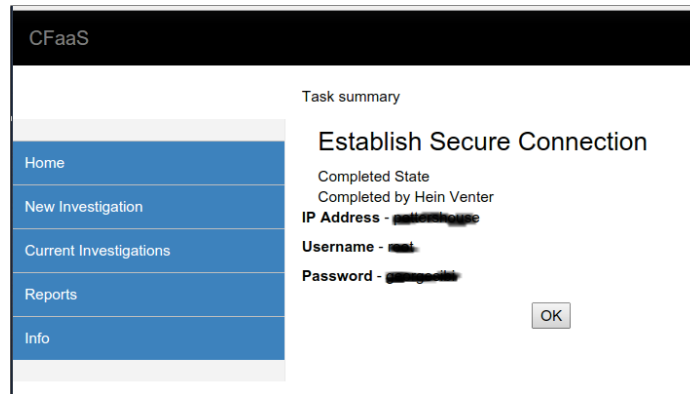


Figure 10.10: Establish Secure Connection Summary

tasks. In this case, however, the task tests the connection and returns true if the connection to the remote host was successful and an appropriate error message is displayed otherwise.

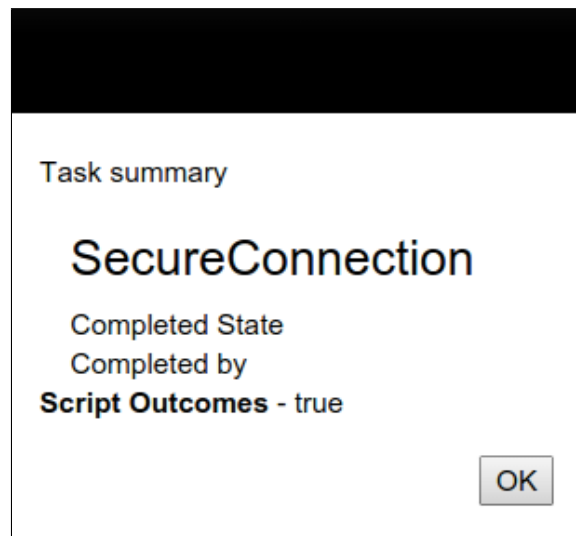


Figure 10.11: Secure Connection Summary

Figure 10.12 shows outcomes from the *Enabling Secure Logging* service task. The task creates a temporary file directory `"/root/work"` on the incident scene to be used to temporarily keep files collected from the scene. files that can be temporarily kept in the directory are system logs, RAM dumps and network traces dumps. The files are kept briefly in this directory before they are sent to the secure evidence storage server and access to the directory is restricted. In this case, creation of the directory failed as the directory already exists. files created by investigative tasks in the directory are still hidden and permission for access is restricted to the owner who is an investigator in this case.

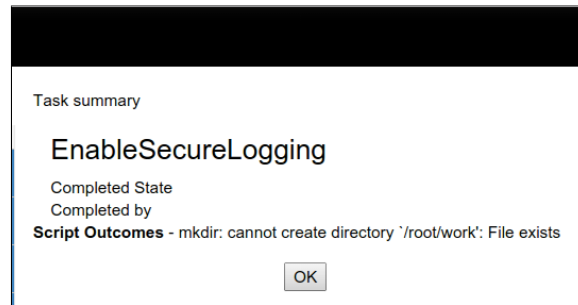


Figure 10.12: Enable Secure Logging Summary

Figure 10.13 represents outcomes from the forensic Team Organisation task. In this case, a leader of the investigation team is specified and members who will take part in the investigation are provided. These members will later be assigned to tasks in the Task Assignment process.

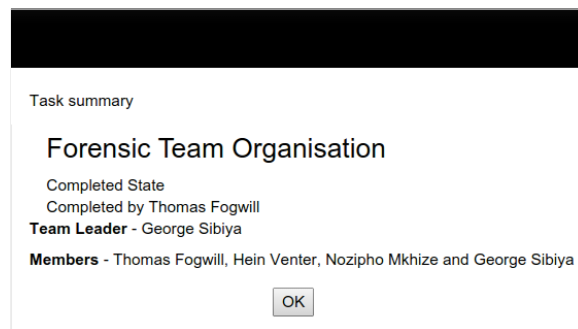


Figure 10.13: Forensic Team Organisation Summary

Figure 10.14 presents a snippet from the digital forensic investigation report that shows the first four tasks of the investigation, namely *Register Case*, *Incident Description*, *Establish Secure Connection* and *Secure Connection*. The report is made up of each task outcome as was indicated in this section. This report can be used during court proceedings or at a hearing, together with exhibitions that may include images and videos recovered or recreated during the course of the investigation.

10.6 CONCLUSION

In this chapter, the CFaaS service model architecture evaluation was presented. The evaluation was based on three aspects of CFaaS: firstly, the design of the standardised digital forensic process presented in Chapter 5; secondly, the architecture of CFaaS that enables the execution of the standardised digital forensic process; and thirdly, the implementation of CFaaS. These aspects focused on how CFaaS satisfied the

CFaaS

[Dashboard](#) | [Settings](#) | [Profile](#) | [Help](#) | [Logout](#)

Process - Register Case
Completed By - Hein Venter
Case ID - _MER788/2014
Case Description - A complaint was received from a complainant outside MER network that pings were being received from an IP address belonging to MER. MER requests for an investigation to be conducted on a cloud based instance with the IP address in question. The investigation seeks to uncover how the ping requests originate and to also uncover any other compromises that may exist on the instance. It is suspected that some one may be misusing the cloud instance to probe remote hosts for a possible attack on the remote hosts. A conviction is possible if a perpetrator would be identified after the investigation.

Process - Incident Description
Completed By - Hein Venter
Organisation - MER
Incident Description - Message received from complainant outside MER:
 ===== We have blocked someone from your IP space for abuse. Reason: Port_Scanning. Log lines are below. Time zone is UTC, 2014-04-14T00:49:44+00:00 swill 1397436532.796229 - - - - - tcp Scan::Address_Scan 0.0.0.78 scanned at least 53 unique hosts on port 80/tcp in 0m0s remote 0.0.0.78 - 80 - swill4-8 Notice::ACTION_LOG 3600.000000 F - - - - I am writing to inform you so that you can take whatever action is necessary to prevent this user from doing this again. We would be happy to discuss further if you would like. Please feel free to respond to this email to follow up.
 ===== The network traffic that is being observed at cloud edge router: ##### May 15 10:06:51 mr3 kernel: ipfw: 10038 Deny ICMP:3.1 0.0.72.211 0.0.0.78 in via ix0.101 May 15 10:06:51 mr3 kernel: ipfw: 10038 Deny ICMP:3.1 0.0.90.173 0.0.0.78 in via ix0.101 May 15 10:06:51 mr3 kernel: ipfw: 10038 Deny ICMP:3.1 0.0.72.211 0.0.0.78 in via ix0.101 May 15 10:06:51 mr3 kernel: ipfw: 10038 Deny ICMP:3.1 0.0.77.232 0.0.0.78 in via ix0.101 May 15 10:06:51 mr3 kernel: ipfw: 10038 Deny ICMP:3.1 0.106.38 0.0.0.78 in via ix0.101 May 15 10:06:51 mr3 kernel: ipfw: 10038 Deny ICMP:3.1 10.174.98.177 0.0.0.78 in via ix0.101 May 15 10:06:51 mr3 kernel: ipfw: 10038 Deny ICMP:3.1 0.0.100.237 0.0.0.78 in via ix0.101 May 15 10:06:51 mr3 kernel: ipfw: 10038 Deny ICMP:3.1 0.0.85.14 0.0.0.78 in via ix0.101 May 15 10:06:51 mr3 kernel: ipfw: 10038 Deny ICMP:3.1 0.0.234.68 0.0.0.78 in via ix0.101 May 15 10:06:52 mr3 kernel: ipfw: 10038 Deny ICMP:3.1 0.0.101.178 0.0.0.78 in via ix0.101 May 15 10:06:52 mr3 kernel: ipfw: 10038 Deny ICMP:3.1 0.0.121.4 0.0.0.78 in via ix0.101 #####
Incident Type - R2L

Process - Establish Secure Connection
Completed By - Hein Venter
IP Address - 172.18.2.4
Username - root
Password - this is my password

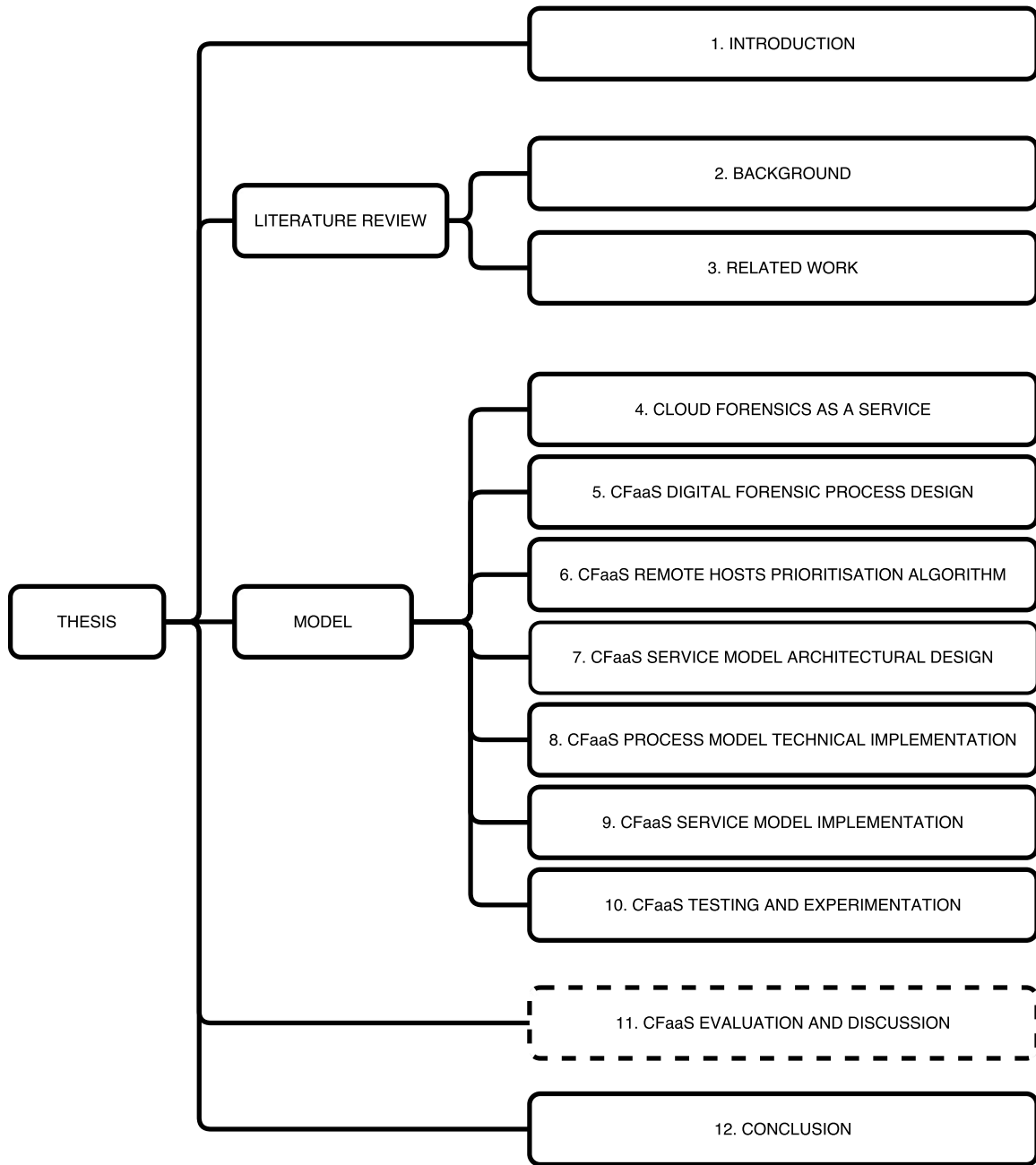
Process - SecureConnection
Script Outcomes - true

Figure 10.14: Investigation Report Snippet

investigator needs as presented in Chapter 4. The outcomes of the evaluation carried out in this chapter are subjected to critical discussion in the next chapter.

11

CFaaS Evaluation Results Discussion



11.1 INTRODUCTION

This research was carried out to address issues associated with conducting digital forensics in a volatile and often multi-jurisdictional distributed cloud computing environment. The following was the basic research question that this research had to answer in addressing the research problem:

On what framework can a cloud forensic solution be based for a cost-effective digital

forensic investigation in the cloud?

In order to address this research question, research sub-questions had to be addressed, which are:

1. What are the requirements for a Cloud forensic system?
2. What are standardised procedures that can be carried out in a Cloud environment while conducting a digital forensic investigation?
3. How can a Cloud forensic system be designed?
4. How can an investigation in a distributed cloud environment be optimised?

The objectives that were set to guide the research towards addressing these research questions are as follows which were to:

1. Uncover shortcomings in existing digital forensic investigation frameworks.
2. Propose a framework on which digital forensic systems aimed for the cloud can be based.
3. Propose a cloud forensic service model for the cloud that is based on the proposed framework.
4. Demonstrate the effectiveness of the framework in a practical environment by implementing and testing a prototype of the model.

The shortcomings of existing endeavours were discussed thoroughly in Chapters 2 and in Chapter 3, where a qualitative analysis on the endeavours was also provided. Analysis of these shortcomings led to requirements for a cloud forensic system and these requirements were presented in Chapter 4 where the CFaaS framework was also presented. Required procedures that can be carried out in investigating cloud environments were presented in Chapter 5. A design of CFaaS service model was presented in Chapter 7 and an optimisation of an investigation process in cloud environments in Chapter 6. Both were presented to fulfil the objectives that were set for addressing the research questions.

The proposed digital forensic process, CFaaS service model architecture and the implementation of CFaaS service model were evaluated in Chapter 10. This penultimate chapter presented discussions on the results obtained during the CFaaS evaluation. The discussion made a critical analysis of the extent to which the results obtained

support the solutions that were provided in this thesis (i.e. a digital forensic process, cloud forensic system architecture and its implementation) in a form of prototype to address the research questions.

11.2 PROCESS EVALUATION RESULTS DISCUSSION

This section presents a critical discussion and analysis of the results obtained while evaluating the CFaaS cloud forensic process model proposed in Section 10.2 (see Chapter 10). As one of the contributions made by this thesis, a cloud forensic process model that can be applied in investigating cloud environments is presented. Because a standardised digital forensic process is required in the cloud, the digital forensic process model presented in this thesis is based on the ISO/IEC27043 standard. The purpose of following such a standard is to enhance the chances of the admissibility in court of evidence obtained through investigating cloud environments.

However, for the proposed cloud forensic process to work, assumptions are made in this thesis that the IaaS provider has access to all instances running on their infrastructure and that the IaaS providers are willing and able to collaborate with investigators. It is also assumed that the incident scene remains accessible in the entire duration of the investigation. Unfortunately these assumptions may not always be true in a real world. Therefore, there still remains the need to test the digital forensic process model proposed in this thesis in real practice.

For the purposes of this thesis, an analytic evaluation of the process is sufficient. The process was subjected to the cohesion-coupling matrix evaluation by Vanderfeesten et al. in [149] because the cloud forensic process was implemented as a workflow. In Vanderfeesten et al. [149], it is argued that a weakly coupled and strongly cohesive workflow design results in fewer errors during the information exchanges. This is the stance that is also taken in this thesis and hence the CFaaS cloud forensic process was evaluated based on its coupling and cohesiveness. The cohesion-coupling evaluation of the CFaaS cloud forensic process model yielded a coupling-to-cohesion ratio, ρ , of 2.59.

The obtained result is slightly larger than an ideal ratio as a ratio below 1 is desired. This would indicate that such a process or workflow is strongly cohesive and weakly coupled. The results obtained for the CFaaS cloud forensic process model resulted from the information elements, 2 (Case Description), 3 (Incident IP Address and Credentials) and 11 (Tasks with assigned members), see Table 10.1, which are reused in many other consecutive processes. In the design of CFaaS, this trade-off is

necessary as an investigator would, for example, not be expected to re-enter the case description, incident scene credentials and task assignments every time a subsequent activity has to be carried out. Alternative designs of the process that would yield a lower ratio would nevertheless still be desired.

The results obtained on the evaluation of the CFaaS process model indicate that the process model design is still among the best when compared with other digital forensic processes (see Table 10.2). This means that the CFaaS cloud forensic process has sufficiently low coupling and sufficiently high cohesion to minimise errors during information exchange among individual digital forensic processes or workflow activities.

The next section presents a discussion on the remote hosts prioritisation algorithm evaluation.

11.3 REMOTE HOSTS PRIORITISATION (SLOE) ALGORITHM EVALUATION DISCUSSION

This section, discusses the outcomes of the evaluation performed on the SLOE (Selection Of rEmote hosts) algorithm in Section 10.3 (see Chapter 10). The SLOE algorithm is an algorithm meant to optimise a list of remote hosts that can be considered for further investigation. The algorithm helps to address the research question on the optimisation of an investigation in the cloud environment.

Since the main purpose of deploying cloud virtual instances is to host cloud services, such instances are expected to have a large number of network connections originating from service consumers. In case of an incident, it would be unrealistic for investigators to probe or investigate all remote hosts connected in an attempt to identify a perpetrator.

Cloud-based instances are meant to host cloud services. If such an instance were to be compromised, both legitimate and illegitimate connections would be expected in the cloud incident scene. So, if investigators would attempt to make a manual selection of remote hosts and investigate them, a significant amount of time may be spent investigating legitimate remote hosts in a case where a malicious connection's host would be listed last. This algorithm therefore contributes to optimising an investigation by ranking remote hosts in descending order of their probability of being malicious hosts. The researcher argues that by selecting the most relevant remote hosts as part of the *Potential Evidence Identification* process, it can save time and hence also costs – time can be translated to costs (e.g. in a case where

investigators bill per hour).

The SLOE algorithm is implemented in the *Determine Remote Hosts* task inside the Potential Evidence Identification process. Since the Determine Remote Hosts task is placed early in the cloud forensic process (see Figure 8.1), it is essential that this task be scalable and error free, so that it does not block the entire investigation process. The SLOE algorithm inside this task is then evaluated based on correctness and time efficiency. The SLOE algorithm's correctness has been proved in Section 10.3.1 and its time efficiency has been proved in Section 10.3.2. The correctness aspect checks whether the algorithm delivers expected outcomes (hosts in their order of priority) and whether it terminates. The time efficiency evaluation aspect reflects on the scalability of the algorithm, as it can be inferred from the algorithm's time complexity function, $F(l) = O(l)$ that was obtained as can be seen in Figure 10.42. Where $F(l)$ is the time complexity of the algorithm and $O(l)$ is the Big-O notation of the order of growth (see Section D.2 in Appendix D).

How this algorithm addresses the question of optimisation firstly concerns its position in the entire cloud investigation process. As the cloud incident scene is to be investigated live, the identification of connections and prioritisation of hosts in the early stages of the investigation process increase chances to also capture malicious connections before a perpetrator realises that the incident scene is being investigated and therefore terminates its connections.

Secondly, before investigators start investigative interactions with the incident scene, the SLOE algorithm would already have listed and ranked remote hosts that are likely to render additional evidence, hence saving time. By using manual approaches to list remote hosts, investigators would only be complementing the list already provided by the SLOE algorithm.

The evaluation has verified that the algorithm delivers desired results, terminates and is scalable.

In the next section, the evaluation of the CFaaS service model architecture is discussed.

11.4 ARCHITECTURE EVALUATION RESULTS DISCUSSION

This section discusses the results of the evaluation performed on the designed CFaaS service model architecture (see Section 10.4). The evaluation was done to determine the extent to which the architecture addresses the research problem on how a cloud forensic system can be designed.

The CFaaS service model architecture was evaluated by using an analysis method, ATAM (see Section 10.4). The aim of the evaluation with ATAM was to identify risks and benefits as a result of adopted design approaches and technology choices. The process of architecture evaluation also reflects the levels of importance and difficulty to implement attributes that meet the requirements of the architecture.

One trade-off that has been identified in the CFaaS service model architecture evaluated is confidence on the level of the CFaaS service security that is traded over scalability. The CFaaS service is designed to be deployed in cloud environments. Once a service is deployed in a cloud environment, the service owner cannot guarantee the security of the service. Security can only be guaranteed by the cloud service provider that hosts the service, unless the service is hosted in a private cloud environment. The unlimited resources available in cloud environments are ideal for the scalability aspect of the CFaaS service. Scalability of CFaaS therefore takes precedence over confidence about security, as the CFaaS service model architecture addresses the service's security in one of its components, namely Identity and Security Management (discussed in Chapter 7).

The presented CFaaS architectural evaluation shows that the research question has been addressed adequately in this study. Requirements were proposed for cloud forensic systems in Chapter 4. The evaluation of the CFaaS service model architecture indicates a level of importance and levels of difficulty to implement features of the CFaaS service model architecture that address the proposed requirements. Features that address highly rated non-functional¹ requirements in terms of importance (i.e. Ease of Use and Efficiency, Scalability and Security) were found to have medium difficulty when being implemented in the CFaaS service model architecture. The evaluation has shown that it is possible to design a cloud forensic service architecture in a manner that addresses both non-functional and functional² requirements.

How both functional and non-functional requirements were addressed by the architecture will be further discussed in the next section where the experimental evaluation results on the CFaaS service architecture implementation are at issue.

11.5 EXPERIMENTAL EVALUATION RESULTS DISCUSSION

The CFaaS implementation was aimed at evaluating the effectiveness of CFaaS in guiding investigators through the investigation of cloud environments. The criteria

¹Non-functional requirements state characteristics of the system to be achieved that are not related to functionality.[22]

²A functional requirement specifies a function that a system must be capable of performing.[22]

used to determine CFaaS effectiveness are based on CFaaS satisfying investigator needs that have been presented as functional requirements (see Chapter 4):

1. Implement a standard digital forensic process in the cloud.
2. Aid an investigator through the standard digital forensic process in the cloud.
3. Allow collaboration among multi-jurisdictional law enforcement agencies in a cloud investigation.
4. Semi-automated

The implementation of CFaaS has demonstrated how a standardised digital forensic process can be implemented in practice whereby ISO/IEC27043 compliant processes were implemented. During experimental evaluation of the implementation, an investigation could be conducted using CFaaS and investigation tasks were rendered to investigators in their standard sequence. The most important product of an investigation is a report which is what has to be presented in a hearing. In CFaaS, preliminary reports on completed tasks may be viewed at any stage of the investigation. On conclusion of the investigation, a report with aggregated outcomes from individual tasks is generated. This demonstrates that a digital forensic process can be implemented in practice in a way that can easily be followed by investigators while investigating cloud environments.

During the evaluation of the CFaaS implementation, an investigator could be guided through the investigation process. As much as investigators need to be trained on the ISO/IEC27043 standard implemented by CFaaS, CFaaS does not require an investigator to know the sequence of processes and procedures to carry out an investigation. CFaaS seamlessly ensures that processes and procedures in the standard are carried out.

A shortcoming while following the steps of the investigation process, however, is that an investigator cannot explicitly request to redo an already completed specific process within the entire process. To redo a specific process, the entire investigation process would need to be restarted. In principle, it should be possible to redo a specific process or procedure without having to terminate the entire ongoing investigation process. This is clearly a weakness in implementation itself but not in the concept presented in this thesis. Moreover, although the inability to redo a specific process can be viewed as a weakness of this particular implementation, it is on the other hand necessary, as processes in CFaaS are interdependent. In other words, outputs from

one process are utilised in the execution of other consecutive processes. Processes that utilise outputs from a revised process would need to be revised as well. It would still be ideal, however, for CFaaS to allow revision of only affected processes.

In allowing multi-jurisdictional agencies to collaborate, CFaaS implementation provides a common platform through which investigators may work together in investigating a case. Such collaboration is made possible through a cloud-based deployment of CFaaS. Some investigation tasks in CFaaS are trivial and others require investigators to apply their intelligence. Hence, some of the tasks in an investigation could well be automated, while other tasks that require investigator intervention should be implemented as such. The CFaaS implementation evaluation demonstrated that all tasks or processes are executed by investigators. Automated service tasks were also executed and their outcomes formed part of the report that was prepared on conclusion of the investigation.

For the current implementation, a cloud investigation with CFaaS depends on the cloud incident scene remaining accessible throughout, in other words the cloud incident scene remains on-line and the credentials do not change for the duration of the investigation. This, however, would not be the situation in all investigation cases. Handling cases in which incident scene instances go off-line or credentials change, needs to be investigated further. For the purpose of this thesis, it was assumed that an instance will remain accessible throughout the investigation process. This assumption is most likely to hold true where incident scenes are cloud service hosts. A virtual instance hosting a cloud service is expected to be running for the entire lifetime of the cloud service being offered. The evaluation of the implementation was aimed at assessing how CFaaS best satisfies user needs. This aspect focused on the non-functional requirements which are:

1. Flexibility.
2. Ease of use and Efficiency.
3. Scalability.
4. Security.
5. Maximum Control of the Service Stack by Investigating Agency.
6. Auditability.

The flexibility aspect had to do with the ability of CFaaS to cater for updates in digital forensic standards. Though updates in international standards are less

frequent, more digital forensic standards are expected in the near future and some jurisdictions may prefer different digital forensic standards than ISO/IEC27043. CFaaS therefore has to take this into account. One of the key components of CFaaS is the CFaaS Task Server. This component comprises a process that executes the processes template that implements the standardised digital forensic processes. Flexibility evaluation therefore had to evaluate the effort required to replace the standard process template.

In respect of evaluating the ability to replace a cloud forensic process template, CFaaS is found to require less effort. However, on the current implementation of CFaaS, the ability to replace a process template is only possible if executed by a CFaaS service administrator. Moreover, a new process template needs to contain only predefined sub-processes and procedures in CFaaS; this implies that tasks forms and script task implementations are not created dynamically. If a replacement process template would contain a task named Case Registration instead of Register Case that already exists in CFaaS, a task form required to complete Case Registration would not be available. A task form for a task named Register Case would however be available. Task forms are rendered based on syntactic task names rather than semantic task names. This is an aspect that can still be improved on in later implementations of CFaaS.

With regard to ease of use and efficiency, the evaluation of the CFaaS implementation considered Learnability, Efficiency, Memorability, Errors and Satisfaction. On evaluation of the implementation, the CFaaS user interface was found to be self-explanatory, hence, easy to learn (as can be seen in Figure 10.6). To use the system, an investigator needs only minimal training and the training would emphasise only the standard digital forensic process as given in ISO/IEC27043. Regarding efficiency, CFaaS activates consecutive tasks as soon as each task is completed. These tasks get listed on the investigation page for the investigator to execute. The efficiency of CFaaS can only be affected by expertise and experience of an investigator in carrying out manual investigative tasks. For instance, if it takes an investigator 10 working hours to carry out Network Types Identification, the next task, namely State-of-the-art RAM and Hardware will wait 10 working hours before it can be activated. This likely effect on efficiency is therefore a human factor outside the CFaaS system.

In CFaaS, processes that require a scalable system are Evidence Collection, Evidence Transportation, Evidence Storage and Evidence Examination and Analysis. Depending on the volume of data being collected, transported, stored or analysed, the time taken by automated tasks in these processes needs to be proportional to the

data size. With CFaaS deployed on a system with static resources, CFaaS scalability would be affected. CFaaS however, is deployed on the cloud hence taking advantage of the available resources in the cloud.

In the current implementation of CFaaS, security employs the basic username-password authentication. Investigators are only allowed to utilise the service after they have been authenticated. Manual tasks to be executed are rendered only to authorised investigators. The security of the entire forensic system can further be enforced by Maximum Control of the Service Stack by Investigating Agency, i.e. deploying the CFaaS service on a private IaaS.

CFaaS is deployed in a private cloud environment. The entire service deployment stack is under the control of an organisation. In such a scenario, restriction of access to the CFaaS's virtual instances can easily be enforced by service administrators through firewall rules.

To enable auditability, all investigative commands executed by service tasks and manual tasks executed by investigators are logged and become part of the documentation. While auditing the digital forensic system, an auditor can execute those commands on the incident scene to see if there is any correspondence between documented results and what they are able to gather. This is however dependent on the availability of the virtual incident scene instance. In either way, auditing capability of a digital forensic system is required as the need for auditing may arise at any instance during a litigation process.

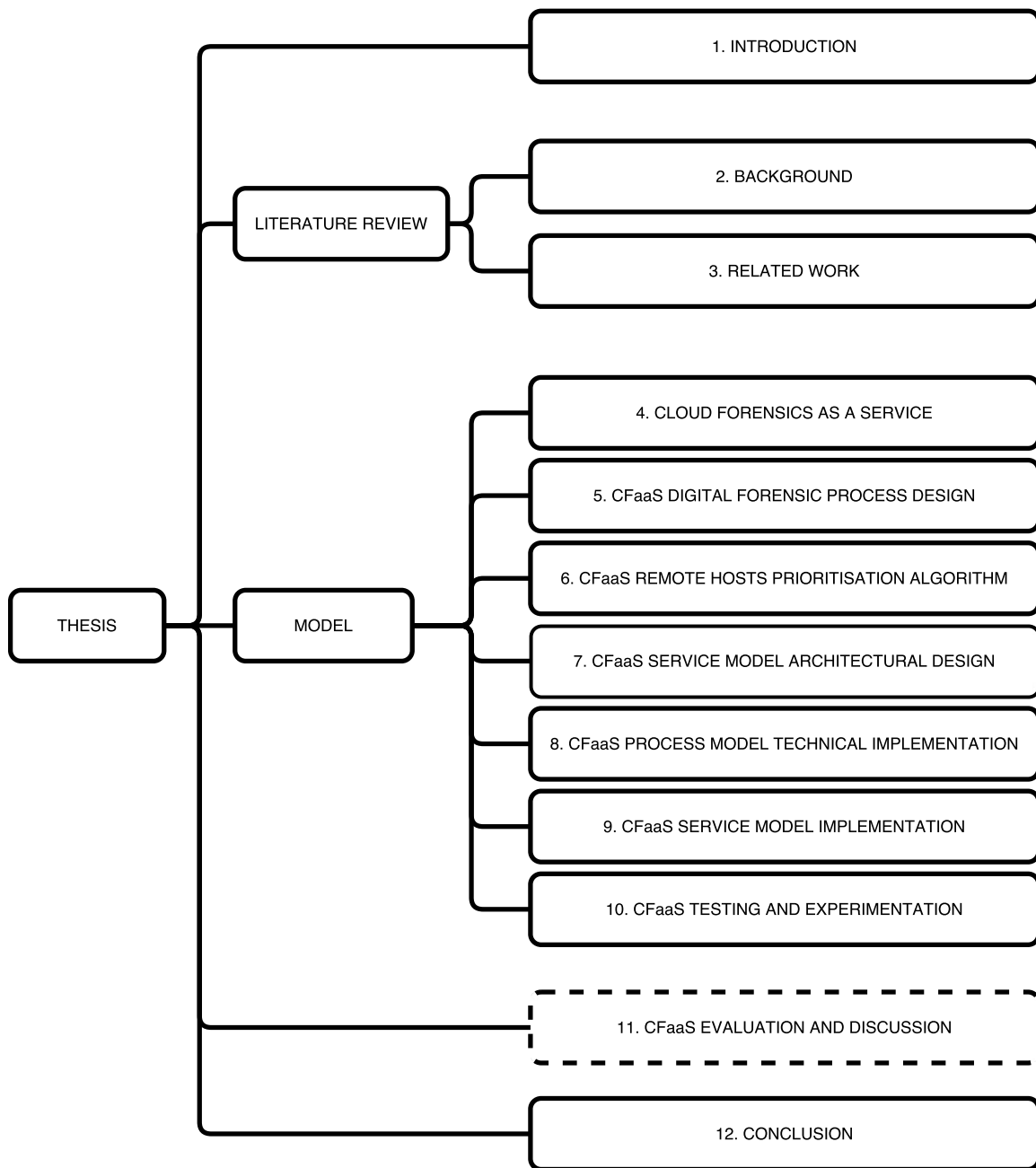
11.6 CONCLUSION

In this chapter, the evaluation results obtained in Chapter 10 were discussed. The discussion presented the strengths of CFaaS based on the evaluation results, as well as areas that can still be improved in CFaaS.

The next chapter, Chapter 12, concludes the thesis.

12

Conclusion



12.1 INTRODUCTION

The first and major contribution made by this thesis involves the standardised digital forensic process model that can be applied while investigating cloud environments. Secondly, the thesis proposed an infrastructure in which the standardised process can be executed and followed by investigators. This chapter provides a concluding discussion of the thesis and is organised as follows: Section 13.2 contains a summary

of the problem being addressed by this thesis. Section 13.3 summarises the solution to the problem presented in this thesis and Section 13.4 is devoted to future work. Section 13.5 provides a final conclusion of the chapter and the research.

12.2 SUMMARY OF RESEARCH PROBLEM ADDRESSED

The research problem is summarised as follows: Cloud computing is a relatively new computing paradigm that makes digital forensics a different and challenging ball game. There are key attributes of the cloud that make the environment to be unique and therefore require novel techniques and processes in conducting digital forensic investigations in the cloud. The key attributes of cloud computing include the following:

1. Distributed Nature
2. Encrypted Data
3. Multi-Tenancy
4. Fragmented Data
5. Volatile Nature

These attributes of the cloud were discussed in detail in Chapter 2 (see Section 2.4). Clearly, to investigate an environment with such attributes requires collaboration between stakeholders in an investigation process. The stakeholders include investigators, cloud data owners (the victim(s) and even the perpetrator), cloud service providers that have physical access to cloud data and law enforcement agencies from the jurisdictions concerned. Standards and standardised collaborative environments that can enable stakeholder collaboration were found to be lacking at the time of this research.

To aid in providing a solution to this problem, the following main research question were formulated and had to be addressed:

On what framework can a cloud forensic solution be based for a cost-effective digital forensic investigation in the cloud?

To address the above research question, the following research sub-questions were formulated:

1. What are the requirements for a cloud forensic system?

2. What are standardised procedures that can be carried out in a cloud environment while conducting a digital forensic investigation?
3. How can a cloud forensic system be designed?
4. How can an investigation in a distributed cloud environment be optimised?

12.3 RESEARCH PROBLEM SOLUTION

In this section, a summary on how each of the research questions was addressed in this research is provided.

12.3.1 ADDRESSING: WHAT ARE THE REQUIREMENTS FOR A CLOUD FORENSIC SYSTEM?

To provide a way to address the main research question, the research sub-questions had to be addressed. To address the question on requirements for a cloud forensics system, existing research efforts that address digital forensic challenges were reviewed. The reviewed literature included articles from cloud computing, digital forensics, literature that attempts to address the digital forensic issues in the cloud and survey articles that look at developments in addressing digital forensics challenges — more specifically, digital forensics in the cloud. In this thesis, the review of existing literature spanned over chapters namely Chapters 2 and 3. In Chapter 3, a critical analysis was made of existing literature. From this analysis of the literature, the following requirements were deduced and grouped under functional and non-functional requirements.

Functional requirements:

1. Implement a standard digital forensic process in the cloud.
2. Aid an investigator through the standard digital forensic process in the cloud.
3. Allow collaboration among multi-jurisdictional law enforcement agencies in a cloud investigation.
4. Semi-automated - Manual investigative tasks are carried out by investigators while being assisted by the cloud forensic system. Trivial investigative tasks that can be implemented as scripts are implemented as such and carried out automatically the cloud forensic system.

Non-functional requirements:

1. Flexibility.
2. Ease of use and Efficiency.
3. Scalability.
4. Security.
5. Maximum Control of the Service Stack by Investigating Agency.
6. Auditability.

12.3.2 ADDRESSING: WHAT ARE STANDARDISED PROCEDURES THAT CAN BE CARRIED OUT IN A CLOUD ENVIRONMENT WHILE CONDUCTING A DIGITAL FORENSIC INVESTIGATION?

In Section 3.2 (see Chapter 3), a critical analysis was made of the existing digital forensic processes. A comprehensive and standardised digital forensic process was found to be the ISO/IEC 27043 (Information technology - Security techniques - Incident investigation principles and processes) [61]. The ISO/IEC27043 was therefore tailored for the cloud by specifying and implementing procedures that can be carried out in cloud environments. The design of the cloud forensic process model with these procedures was presented in Chapter 5 and the implementation of the process model was discussed in Chapter 8.

The process model was evaluated in Chapter 10 and its design was shown to have minimal errors compared to other process models (see the discussion on its evaluation in Chapter 11).

12.3.3 ADDRESSING: HOW CAN A CLOUD FORENSIC SYSTEM BE DESIGNED?

In Section 3.4, a critical analysis of existing digital forensic architectures was carried out. Based on the outcomes of the critical analysis of the architectures that addressed the requirements in Section 12.3.1 above, an architecture was designed. The design of this architecture was presented in Chapter 7 while the implementation of the architecture was presented in Chapter 9. The evaluation of the architecture was of a twofold nature: first, it involved the design of the architecture itself, and second, it involved the implementation of the architecture through a prototype.

These evaluations were discussed in Section 10.4 and Section 10.5 respectively. The results of the respective evaluations showed that the architecture and its implementation addressed the requirements listed in Section 12.3.1. This is according to the discussion of the evaluations presented in Sections 11.5 and 11.

12.3.4 ADDRESSING: HOW CAN AN INVESTIGATION IN A DISTRIBUTED CLOUD ENVIRONMENT BE OPTIMISED?

The goal of optimising the digital forensic process was to minimise the costs involved in it. The researcher shared the sentiment that if the investigation process were optimised, i.e. a reasonable amount of time was spent on an investigation and the outcomes of an investigation were proportional to the resources invested in it, the cost would be minimised. In Section 3.5, approaches used by other researchers to minimise costs in a digital forensic investigation were reviewed. Although the approaches were valid, the researcher focused on one aspect that could contribute to the accumulation of unnecessary costs, especially in a cloud environment. This aspect was based on the fact that a large number of network connections could be expected on an Internet-based cloud incident scene, and these could also include a perpetrator's network connection. An investigator would be expected to analyse each and every network connection until the malicious one was identified. This might not always be feasible and could be a daunting task, even with the support of network analysis tools.

In this thesis, network connections which can be translated to remote hosts that are connected with a cloud incident scene were prioritised. The prioritised remote hosts can then be considered for further investigation. Automatically prioritising hosts saves time on the part of investigators and hence, it contributes towards optimising a digital forensic process in the cloud. The **SeL**ection **Of rEmote** hosts (SLOE) algorithm that prioritises the remote hosts was presented in Chapter 10 and it was evaluated through formal analysis in Section 10.3. The SLOE algorithm was furthermore found to be correct (it returned expected results) and scalable in accordance with the discussion of the evaluation results in Section 11.3.

12.3.5 THE FRAMEWORK

All the artefacts presented as solutions to the research sub-questions constitute a framework that addresses the main research question in this thesis. The artefacts that were presented and evaluated in this thesis are the following:

1. A Cloud Forensic as a Service (CFaaS) architecture in Chapter 7,

2. A Standardised Digital Forensic Process Model in Chapter 5,
3. A Remote Hosts Prioritisation Algorithm (SLOE) in Chapter 6 and
4. An implementation of the CFaaS service model through a prototype in Chapter 9.

Through this framework, the thesis in hand therefore contributes towards addressing digital forensic challenges in cloud environments. The framework provides a standardised process of conducting an investigation in cloud environments and an infrastructure to carry out the investigation in a collaborative manner.

12.4 FUTURE WORK

This research has addressed the research questions presented in Section 1.3. This was achieved through meeting the objectives presented in Section 1.4. The research, however, has limitations which open opportunities for further research and they are as follows:

1. Most of the evaluations that were performed in this thesis resulted from formal, theoretical analysis. As part of suggested future work, the framework therefore has to be tested in practice. This thesis is based on multiple assumptions, namely that an investigation is restricted to a single cloud-based incident scene; the incident scene remains on-line for the duration of the investigation; the investigators have credentials to sign in the incident scene; and the credentials do not change for the duration of the investigation. These assumptions are not likely to hold in practice, and therefore have to be addressed when future research work is carried out.
2. The implementation of the standardised digital forensic process model in Chapter 8 and the implementation of the CFaaS service model architecture as a prototype in Chapter 9 are intended for the proof of concept and for evaluation purposes. The implementations are therefore basic. For example, the script task implementations in Chapter 8 utilise the cloud incident scene system commands and the incident scene operating system is assumed to be a Linux-based system. Adversaries who are smart enough will be able to detect if such investigative commands are executed on the system that they are attacking. A furtive way of interacting with the incident scene is required in future implementations of a digital forensic service that follows the CFaaS approach.

3. The cost effectiveness in this thesis was based on an assumption that a cross-border investigation is costly due to the fact that more digital forensic practitioners have to be involved from different jurisdictions involved. The costs however, have not been quantified in this research. A cost model that can be used to quantify costs is required. The model can then be used to quantify costs when an investigation is carried out conventionally and when the investigation is carried out using a multi-jurisdictional collaborative tool such as CFaaS.

12.5 CONCLUSION

The architecture of cloud computing requires that there be changes in the way digital forensics is carried out from the way it currently is in conventional non-cloud environments. New procedures and techniques specifically for the cloud need to be developed. In this research, challenges faced by digital forensic practitioners when investigating cloud environments were presented. Endeavours by other researchers that address the challenges through new digital forensic procedures and digital forensic systems were presented. The shortcomings of those endeavours were brought to light and based on the shortcomings, in this research, new digital forensic procedures or processes and a digital forensic system specific for the cloud were developed and implemented.

Through this, it was demonstrated that, a framework based on the cloud can be used to investigate cloud environments in a standardised manner that can be admissible in a hearing or court of law.

Appendices

A

Chapter 3 Tables

Table A.1, presents an analysis of published digital forensic processes. Each article is assessed based on whether it contains a particular procedure as a process, sub-process, procedure, or an action and/or whether article recognises that a particular process/sub-process/procedure/action is a step that has to be taken during a digital forensic investigation. If the article supports the procedure, it is indicated by a check mark (✓) and if not, a cross (✗) indicates otherwise. In Table A.1, phrases used to name processes or procedures have been retained as they are from their source articles even when they are semantically similar. Only syntactically similar procedures or processes have been unified. For example, if “Reporting” appears in multiple articles a procedure name, it is represented only once in the table. But “Examining crime scene” and “Investigating crime” are represented separately in the table though they may be semantically similar. This was done to preserve the different process or procedure naming by different scholars.

Table A.1: Digital forensic processes evaluation

	Lin, Yen and Chan [83]	Leigland [80]	Shin [116]	Vajjarevic [147]	Bulbul [23]	Baryamureeba and Tushabe [18]	Watson and Jones [151]	Kent [68]	Sindhu and Meshram [127]	ISO/IEC27043 [61]	Casey, Katz and Lewthwaite [29]	Mukasey, Sedgwick and Hagy [94]	Quick, Martini and Choo [100]
Preparation	✓	✗	✓	✗	✗	✗	✓	✗	✗	✓	✓	✗	✓
Reporting	✓	✗	✓	✗	✗	✗	✓	✗	✓	✓	✓	✗	✓
Evidence collection	✓	✗	✓	✓	✓	✗	✓	✓	✓	✓	✗	✓	✓
Evidence analysis	✓	✓	✓	✓	✓	✗	✓	✗	✓	✓	✓	✗	✓
Securing incident scene	✗	✗	✓	✗	✓	✗	✓	✓	✗	✓	✗	✓	✗
Evaluating incident scene	✗	✓	✗	✗	✓	✗	✗	✓	✓	✗	✗	✓	✗
Documenting incident scene,	✓	✗	✗	✓	✗	✗	✓	✓	✗	✓	✗	✓	✗
Evidence packaging	✓	✗	✗	✓	✓	✗	✗	✓	✗	✓	✗	✓	✓
Evidence transportation	✓	✗	✗	✓	✗	✗	✗	✓	✗	✓	✗	✓	✓
Classifying cybercrime	✓	✗	✓	✗	✗	✗	✗	✓	✗	✗	✗	✓	✗
Deciding investigation priority	✗	✗	✓	✗	✗	✗	✗	✗	✗	✓	✗	✓	✗
Investigating victim digital crime scene	✗	✗	✓	✗	✓	✓	✗	✗	✗	✓	✗	✓	✓
Criminal/Perpetrator profiling	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Tracking suspects	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Investigating perpetrator digital crime scene	✗	✗	✓	✗	✗	✗	✗	✗	✗	✓	✗	✓	✓
Summoning suspect	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Incident detection	✗	✗	✓	✓	✗	✗	✗	✗	✗	✓	✗	✗	✗
First response	✗	✗	✗	✓	✗	✗	✓	✓	✗	✓	✗	✓	✗
Planning	✗	✗	✗	✓	✗	✗	✗	✗	✗	✓	✗	✗	✗
Potential evidence identification	✗	✓	✗	✓	✓	✓	✓	✗	✗	✓	✗	✓	✓
Potential evidence storage	✓	✗	✗	✓	✓	✗	✓	✓	✗	✓	✓	✓	✓
Presentation	✓	✗	✗	✓	✗	✓	✓	✗	✗	✓	✗	✗	✓
Conclusion	✗	✗	✗	✓	✗	✗	✗	✗	✗	✓	✗	✗	✗
Obtaining authorisation,	✗	✗	✗	✓	✓	✓	✗	✗	✗	✓	✗	✗	✓
Documentation	✗	✗	✗	✓	✗	✗	✓	✓	✗	✓	✗	✓	✗
Information flow	✗	✗	✗	✓	✗	✗	✓	✗	✓	✓	✗	✗	✗
Preserving chain of evidence	✗	✗	✗	✓	✗	✗	✓	✗	✓	✓	✓	✗	✗

Table A.1: Digital forensic processes evaluation

	Lin, Yen and Chan [83]	Leigland [80]	Shin [116]	Valjarevic [147]	Bulbul [23]	Baryamureeba and Tushabe [18]	Watson and Jones [151]	Kent [68]	Sindhu and Meshram [127]	ISO/IEC27043 [61]	Casey, Katz and Lewthwaite [29]	Mukasey, Sedgwick and Hagy [94]	Quick, Martini and Choo [100]
Preserving evidence	✓	✗	✗	✓	✓	✗	✓	✓	✗	✓	✓	✓	✓
Interaction with physical investigation	✗	✗	✓	✓	✗	✓	✓	✓	✗	✓	✓	✓	✓
Crime scene examination	✓	✓	✓	✗	✓	✓	✗	✓	✓	✓	✗	✓	✓
System assurance	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✓	✓
Evidence search	✓	✓	✗	✗	✓	✗	✗	✗	✓	✗	✓	✓	✓
Evidence acquisition	✗	✗	✗	✓	✓	✗	✗	✗	✗	✓	✗	✓	✓
Hypothesis validation	✗	✗	✗	✗	✓	✗	✗	✗	✗	✓	✗	✗	✗
Organisation of evidence	✓	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✓	✗
Physical management of evidence	✗	✗	✗	✗	✓	✗	✗	✓	✗	✗	✗	✓	✓
System and service restoration	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗
Checking with legal authority	✓	✗	✗	✓	✓	✓	✗	✗	✗	✓	✗	✗	✗
Obtaining digital forensic tools	✓	✗	✗	✓	✓	✓	✓	✓	✗	✓	✗	✓	✗
Physical scene investigation	✗	✗	✗	✗	✓	✓	✗	✓	✗	✓	✗	✓	✗
Surveys and interviews	✗	✗	✓	✗	✓	✗	✓	✓	✗	✗	✗	✓	✗
Digital crime scene investigation	✓	✓	✓	✓	✓	✓	✓	✗	✓	✓	✓	✓	✓
Installing activity monitoring agents	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✓
System preservation	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✓	✓	✓
Crime scene communication shielding	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗
Duplicating and copying of evidence	✓	✗	✗	✗	✓	✗	✗	✗	✗	✓	✓	✗	✓
Evaluating data integrity	✗	✗	✗	✗	✓	✗	✓	✗	✓	✗	✓	✗	✗
Review phase	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗	✗
Operational readiness phase	✗	✗	✗	✗	✗	✓	✗	✗	✗	✓	✗	✗	✗
Infrastructure readiness phase	✗	✗	✗	✗	✗	✓	✗	✗	✗	✓	✗	✗	✗
Incident notification	✗	✗	✗	✓	✗	✓	✓	✗	✗	✓	✗	✗	✗
Reconstruction	✓	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗	✗
Receipt of incident notification	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
New case creation	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
Forensic management notification	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
First response team gathers incident scene information	✗	✗	✗	✗	✓	✗	✓	✓	✗	✓	✗	✓	✗
Determine jurisdiction	✓	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
Serve correct documents on suspect	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
Recover evidence to forensic lab	✗	✗	✗	✓	✗	✗	✓	✓	✗	✓	✗	✗	✓
Complete initial part of evidence seizure summary	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
Receive instructions from law enforcement/client	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
Agree on reporting points	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
Communication plan	✗	✗	✗	✓	✗	✗	✓	✗	✗	✓	✗	✗	✗
Notification and escalation procedures	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
Incident Validation (recognise that an incident has taken place)	✗	✗	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗
Defining approach strategy	✗	✗	✓	✓	✗	✗	✓	✗	✗	✓	✗	✗	✗
Evidence examination	✗	✗	✗	✗	✗	✗	✓	✗	✗	✗	✓	✗	✓
Evidence evaluation	✗	✗	✗	✗	✗	✗	✓	✗	✗	✓	✗	✗	✗

Table A.1: Digital forensic processes evaluation

	Lin, Yen and Chan [83]	Leigland [80]	Shin [116]	Valjarevic [147]	Bulbul [23]	Baryamureeba and Tushabe [18]	Watson and Jones [151]	Kent [68]	Sindhu and Meshram [127]	ISO/IEC27043 [61]	Casey, Katz and Lewthwaite [29]	Mukasey, Sedgwick and Hagy [94]	Quick, Martini and Choo [100]
Presentation of evidence (to a hearing/court)	✓	×	×	✓	×	✓	✓	×	×	✓	×	×	×
Returning of seized items	×	×	×	×	×	×	✓	×	×	×	×	×	×
Documentation of the evidence	×	×	×	✓	×	×	×	✓	×	✓	✓	✓	×
Assessing the crime scene (virtually/physically)	×	×	×	×	×	×	×	×	×	×	×	✓	✓
Create forensic image	×	×	×	×	×	×	×	×	✓	✓	×	✓	✓
Calculate forensic image hash	×	×	×	×	×	×	✓	×	✓	✓	✓	✓	×
Hand over image to investigators	×	×	×	×	✓	×	×	×	✓	×	×	✓	×
Check integrity of forensic image	×	×	×	×	×	×	×	×	×	×	×	×	×
Start investigating image	×	×	×	×	×	×	×	×	×	×	×	×	×
Recover deleted files	×	✓	×	×	✓	×	✓	×	✓	×	×	×	×
Create chain of custody	×	×	×	✓	×	×	✓	×	✓	✓	✓	×	×
Digital evidence interpretation	×	×	×	×	✓	✓	×	×	×	✓	×	×	×
Investigation closure	×	×	×	✓	×	✓	×	×	×	✓	×	×	✓

B

CFaaS Data Flow Diagrams

B.1 LOGICAL VIEW

B.1.1 DIAGRAM 12 - EVIDENCE EXAMINATION AND ANALYSIS.

In order to illustrate at least one detailed discussion example of a Level 1 granularity child diagram, Process 12 (*Analyse Evidence* process) in Diagram 0 is used and is presented as Diagram 12 in Figure B.1. The *Analyse Evidence* process implements the corresponding *Evidence Examination and Analysis* process that was presented in Section 5.2.9 and as represented by Process 12 in Figure 7.3. For the sake of clarity, the process names in Figure 7.3 were shortened compared to the process naming in Chapter 5. Full names of the process that are shortened in the diagram are provided in brackets in the process list on page 83 of this thesis.

Analysis of evidence can be done on a live host or on previously collected evidence acquired from a dead forensics investigation. The focus of this thesis, however, is on live forensics, as the researcher considered it to be the most appropriate for volatile cloud environments.

In Figure B.1, processes that carry the icon are manual tasks. This means that in addition to the input messages indicated in each of these processes, they also accept manual inputs from human investigators. These manual input messages could not be indicated as they would obscure the clarity of the figure.

The total list of included processes in Figure B.1 is as follows:

Process 12.1: Gather Incident Scene System Profile

Process 12.2: Get System Time

Process 12.3: List Modules Loaded to Kernel and Analyse List

Process 12.4: List Active Processes and Analyse List

Process 12.5: Determine Hidden Processes

Process 12.6: Analyse Allocated Memory of Suspicious Processes

Process 12.7: Recover and Analyse Suspicious Processes Associated Files

Process 12.8: Perform String Searches on Memory Dump or Live RAM

Process 12.9: Visualise Collected Information

Process 12.10: Anti-forensic Rootkits Identification

Process 12.11: Anti-forensic Rootkits Deactivation

Process 12.12: Copy MAC Address and Kernel Route Cache Tables

Process 12.13: List Current and Pending Connections and their TCP/UDP Ports

Process 12.14: Decode Connection Protocols

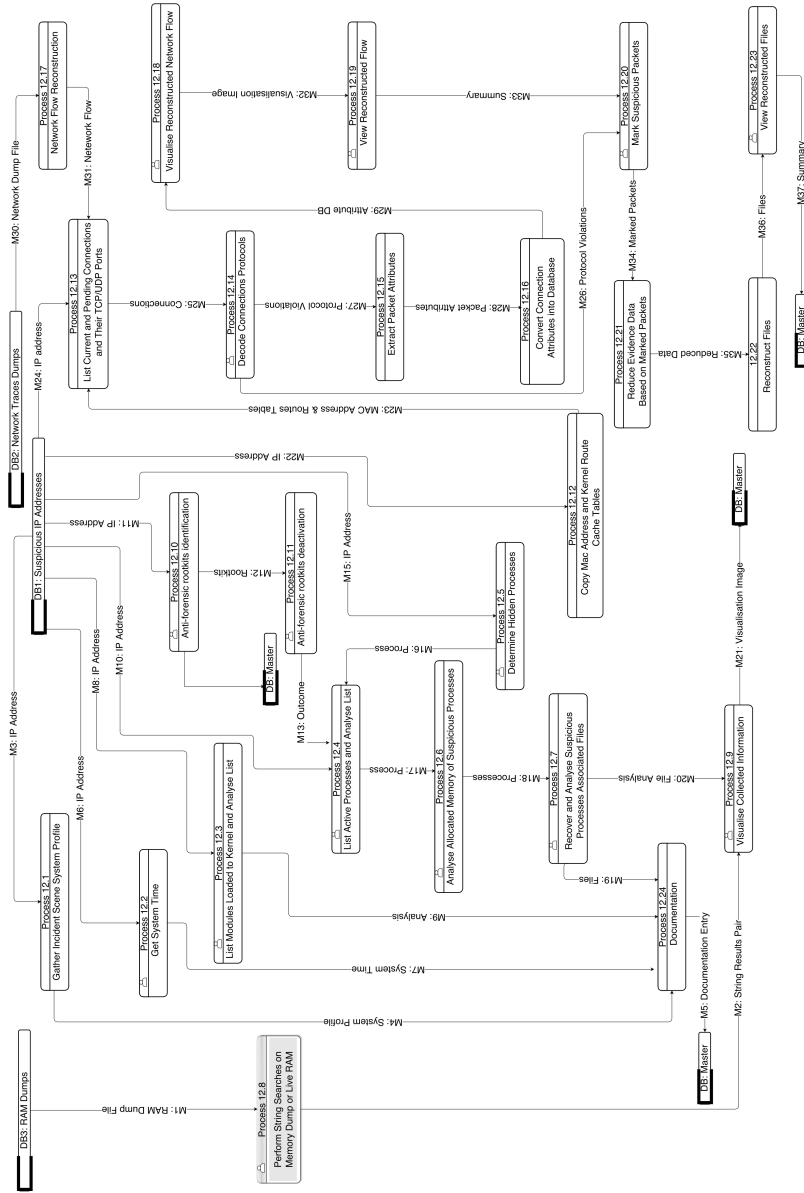


Figure B.1: CFaaS data flow diagram: Child Diagram 12-Analyse Evidence

- Process 12.15: Extract Packet Attributes
- Process 12.16: Convert Connection Attributes into Database
- Process 12.17: Network Flow Reconstruction
- Process 12.18: Visualise Reconstructed Network Flow
- Process 12.19: View Reconstructed Flow
- Process 12.20: : Mark Suspicious Packets
- Process 12.21: Reduce Evidence Data Based on Marked Packets
- Process 12.22: Reconstruct Files
- Process 12.23: View Reconstructed Files
- Process 12.24: Documentation

These CFaaS system processes are server-side implementations of the digital forensic processes discussed in Section 5.2.9. *DB1*, *DB2* and *DB3* in Figure B.1 respectively represent database sources, including IP addresses, Network Traces Dumps and RAM Dumps - all obtained from the secure storage server as in Figure B.1.

For easier understanding of the discussion of the diagram in Figure B.1, the following has to be noted:

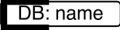
Note 1: The processes and messages are not numbered in any particular order.

Note 2: The arrangement of processes in the figure is based on dependency among the processes, i.e. a process that utilises output from another and the process that produces the utilised output are placed in close vicinity.

Note 3: Before a process is discussed, all paths that lead to the messages that are utilised by the process are discussed first.

Note 4: Explanations of the processes are brief as they have been discussed in detail in Section 5.2.9.

Taking Process 12.8 (shaded) as a start in discussing the processes in Figure B.1, Process 12.8 receives a RAM Dump indicated by Message M1. The reason for starting with this process is merely because it is the leftmost process in the figure and it does not depend on any of the other processes before it can execute. In Process 12.8, an investigator performs a string search on the RAM dump as discussed in Chapter 5. The results obtained from this process, Message M2, go into Process 12.9 where the collected search string results are visualised. Visualisation is a statistical view of the number of matches that have been found in the RAM dump versus the term that has been searched and it can be done with a bar chart, histogram, pie chart, etc. The statistical visualisation images as Message M21 are written to a persistent storage

space for later use in other, subsequent investigation processes such as Reporting (Process 14 in Figure 7.3). Persistent storage is represented by the symbol  in Figure B.1 and in other figures in this section.

Process 12.24 utilises messages M4, M7, M9 and M19. The paths that lead up to these messages are discussed next.

Starting from Process 12.1, Process 12.1 takes an IP address, Message M3, as an input and collects the system profile from the host with the provided IP address. The system profile, Message M4, is sent to Process 12.24 (*Documentation*) to be part of the documentation. As discussed in Chapter 5, Process 12.1 is implemented by a service, and there is no human involvement during the execution of this process as it is an automated task. Process 12.24 writes the information to persistent storage as a *Documentation* entry.

Process 12.2 (*Get System Time*) takes the *IP Address*, Message M6, as an input. The output from this process is Message M7, which also gets sent to Process 12.24 to be written to persistent storage. Process 12.10 (*Anti-forensic Rootkits Identification*) is a manual process in which an investigator makes use of Message M11 (*IP Address*) to identify anti-forensic rootkits that may exist in the host. The output from this process is Message M12, which the investigator uses to execute Process 12.11 (*Anti-forensic Rootkits Deactivation*). The deactivation outcomes from this process, Message M13, are utilised by Process 12.4 (*List Current and Pending Connections and Their TCP/UDP Ports*) and carried out manually by an investigator.

An investigator also uses the *IP Address* in Message M15 to determine hidden processes in Process 12.5 (*Determine Hidden Processes*). Message M16 is an output from Process 12.5 and an input into Process 12.4 (*List Network Connections*). Message M17, a list of system processes obtained from Process 12.4, serves as an input into Process 12.6, i.e. for an investigator to analyse the system processes' allocated memory. After analysing the process memory, an investigator provides a list of suspicious processes that needs to be analysed further. The list of processes is in Message M18, which is to be an input into Process 12.7. The recovered files in process 12.7 and their analysis constitute outputs as Message M19 and Message M20.

The latter two messages become inputs into Process 12.24 and Process 12.9 respectively, where in Process 12.9 an investigator performs a statistical visualisation of the file analysis. The visualisation images in Message M21 are written to persistent storage.

Process 12.12 takes Message M22 (*IP Address*) as an input and copies Media Access Control (MAC) addresses and routing tables from the provided IP address host. The MAC addresses and routing tables information is sent to Process 12.13 (*List Network Connections*) as Message M23. The details in Message M23 are used by Process 12.13 together with the *IP Address* in Message M24. Process 12.13 also accepts a network traces dump file in Message M31 as input. Furthermore, Message M31 is a reconstructed network flow from which network analysis processes are performed. Process 12.13 (*List Network Connections*) lists network connections in the provided IP address host, which is the same host from which details in Message M23 were obtained.

The list of connections constitutes Process 12.13, as Message M25 is used as an input in Process 12.14. In this process, Process 12.14, the connections are decoded by an investigator to see if there are any connections that have network protocol violations. Outputs from this Process are Message M26 and Message M28. The packet attributes in Message M28 are converted into database format in Process 12.16. The attribute values from the database in Message M29 are used in Process 12.18 to construct a statistical visualisation of the network traffic. In Process 12.19, an investigator analyses the visualisation images created in Process 12.18, after which a summary of the images is sent to Process 12.19 as Message M33.

In a cloud-based instance, both legitimate and illegitimate traffic are expected. Process 12.20 deals with marking packets that are deemed to be malicious and they are routed accordingly. Marked packets are used to reduce network evidence data in Process 12.21. The reduced network evidence traffic or packets are sent as Message M35 to Process 12.22. In Process 12.22, files are reconstructed from the

reduced network traffic. Constructed files are sent to Process 12.33 as Message M36. An investigator analyses these reconstructed files and then makes a summary, which is sent to persistent storage as Message M37. Level 1 child diagrams that represent other processes - namely Process 3 through Process 16 in Figure 7.3 - are presented in summary in the next sections.

B.1.2 DIAGRAM FOUR - INCIDENT DETECTION

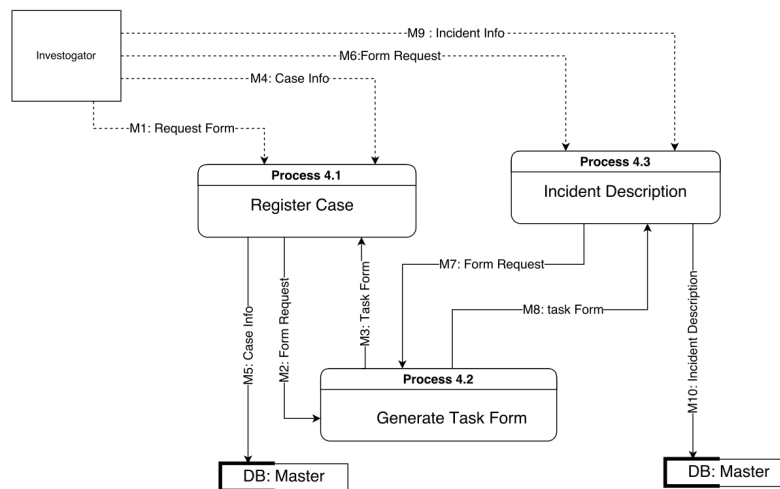


Figure B.2: CFaaS Data Flow Diagram: Diagram 4

B.1.3 PROCESSES

Process 4.1: Register Case - Case registration process by an investigator.

Process 4.2: Generate Task Form - A process that generates a form to be used by an investigator to complete a specific task.

Process 4.3: Incident Detection - A process that involves the detection of an incident that would require investigation.

B.1.4 MESSAGES

Message M1: This is an HTTP request sent by an investigator to the CFaaS API.

Message M2: The request for a Register Case task completion form propagated to a process that generates it.

Message M3: Requested task form is returned after being generated.

Message M4: Information about the case to be investigated, e.g. case identification, case description.

Message M5: Case information to be written to persistent storage.

Message M6: Request for the Incident Description task form.

Message M7: Request for a task form forwarded to a process that is responsible for the generation of task completion forms.

Message M8: Task form as a response message.

Message M9: Information provided by an investigator to be populated on the task form.

Message M10: Information from the investigator regarding the incident to be written to persistent storage.

B.1.5 DIAGRAM FIVE - FIRST RESPONSE

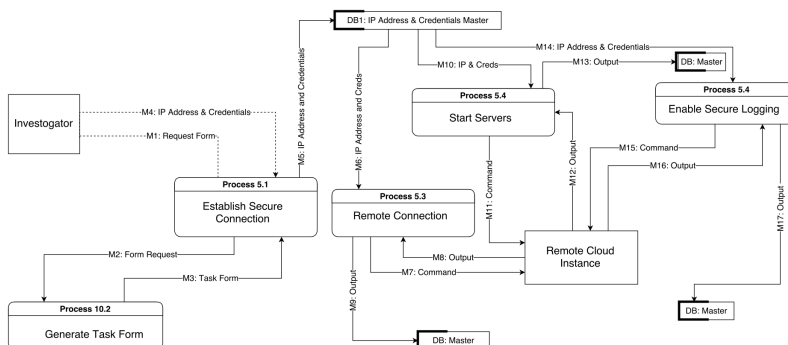


Figure B.3: CFaaS Data Flow Diagram: Diagram 5

B.1.6 PROCESSES

Process 5.1: Establish Secure Connection - A manual task through which an investigator provides credentials to an incident scene.

Process 5.2: Generate Task Form - A process that generates task forms based on the task provided while the request for a form is adhered to.

Process 5.3: Remote Connection - A process that utilises the credentials provided to establish SSH connections with the incident scene.

Process 5.4: Start Servers - A process that starts relevant forensic software servers and clients on both the incident scene and the digital forensic server.

Process 5.5: Enabling Secure Logging - A process that transfers incident scene system logs to a secure location or directory where they cannot be accessed by an adversary.

B.1.7 MESSAGES

Message M1: Request for a task form to complete the Establish Secure Connection process.

Message M2: Request for the form forwarded to the process that generates forms.

Message M3: The generated task form sent to the process that requested it.

Message M4: Incident credentials provided by an investigator to populate the task form.

Message M5: The provided credentials written to working memory for later use by subsequent processes.

Message M6: An SSH command sent to the incident scene as a connection request.

Message M7: Outcomes from the connection attempt sent back to the process that requested it.

Message M8: Connection attempt outcome is written to persistent storage.

B.1.8 DIAGRAM SIX - PLANNING

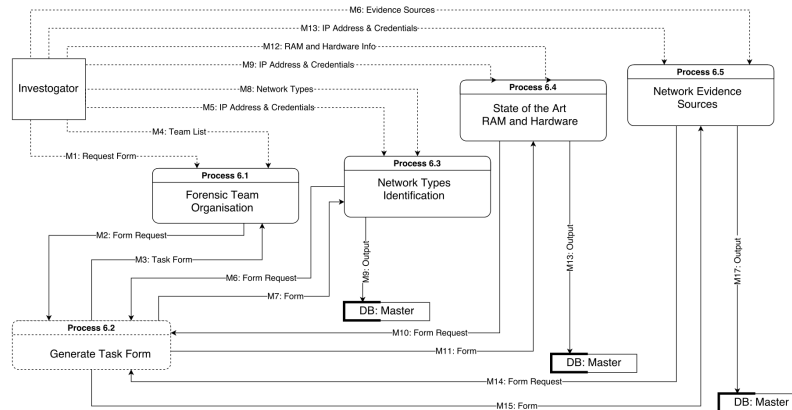


Figure B.4: CFaaS Data Flow Diagram: Diagram 6

B.1.9 PROCESSES

Process 6.1: Forensic Team Organisation - A process that involves compiling a list of investigators that can be involved in an investigation, including an international one in a case that involves multi-jurisdictions.

Process 6.2: Generate Task Form – A process that generates task completion forms relevant to a task.

Process 6.3: Network Types Identification – A manual task by investigators that identify state-of-the-art network types.

Process 6.4: State-of-the-art RAM and Hardware - Manual task process by investigators that identify state-of-the-art RAM and Hardware.

Process 6.5: Network Evidence Sources - Manual task by investigators that involves identifying evidence sources, given the network types identified in Process 6.3.

B.1.10 MESSAGES

Message M1: Request for a task form from an investigator.

Message M2: Request for a form forwarded to a process that generates task forms.

Message M3: Sending a task form as a response to the process that requested it.

Message M4: List of investigation team members that may participate in the investigation.

Message M5: Request for task form from investigator.

Message M6: Request forwarded to a process that generates task forms. Message

- Message M7: Task form as a response to the process that requested it.
- Message M8: State-of-the-art network information provided by investigators through populating the task form.
- Message M9: Information on network types written to persistent storage.
- Message M10: Request for task completion form.
- Message M11: Request for a form forwarded to a process that generates it.
- Message M12: Sending a task form as a response to Process 6.4 that requested it.
- Message M13: State-of-the-art RAM & Hardware/CPU information provided by the investigator.
- Message M14: RAM & Hardware/CPU information to be written to persistent storage.
- Message M15: Task form request.
- Message M16: Request for a task form forwarded to Process 6.2
- Message M17: Sending task form as a response to Process 6.5 that requested it.
- Message M18: Information on network evidence sources to be populated on the task completion form.
- Message M19: Network evidence sources written to persistent storage.

B.1.11 DIAGRAM SEVEN - PREPARATION

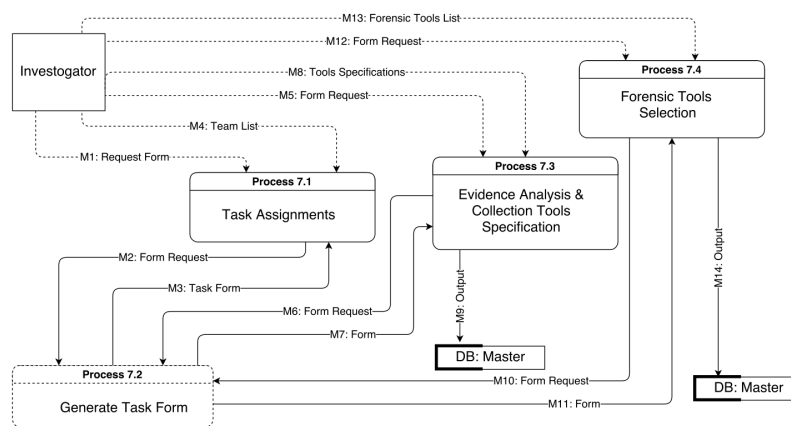


Figure B.5: CFaaS Data Flow Diagram: Diagram 7

B.1.12 PROCESSES

Process 7.1: Task Assignments - A process that involves assigning digital forensic process tasks to investigators who were identified in the planning process.

Process 7.2: Generate Task Forms – A process that generates task completion forms at run time.

Process 7.3: Evidence Analysis & Collection Tools Specification - A manual task process in which digital forensic tools are specified, carried out by investigators.

B.1.13 MESSAGES

Message M1: Request for a task form corresponding to that manual cloud forensic task at hand.

Message M2: Request for a task form is forwarded to a process that generates task forms.

Message M3: Generated task form is sent back to an investigator via Process 7.1.

Message M4: List of investigators, including the tasks that each investigator is eligible to carry out.

Message M5: Request for a task form to carry out Process 7.3.

Message M6: Request for the form is forwarded to the relevant process that generates it, i.e. Process 7.2.

Message M7: Process 7.2 responds with the generated task form to carry out Process 7.3.

Message M8: Specifying the tools that would be required for the investigation.

Message M9: Tools specification information is written to persistent storage to be utilised in later stages of the investigation process.

Message M10: Request for a task completion form for Process 7.4.

Message M11: Request for the task form is forwarded to Process 7.2.

Message M12: Process 7.2 responds with the generated task form.

Message M13: Investigator submits completed task with information on selected digital forensic tools to Process 7.4.

Message M14: Information on selected digital forensic tools is written to persistent storage by Process 7.4

B.1.14 DIAGRAM EIGHT - POTENTIAL EVIDENCE IDENTIFICATION

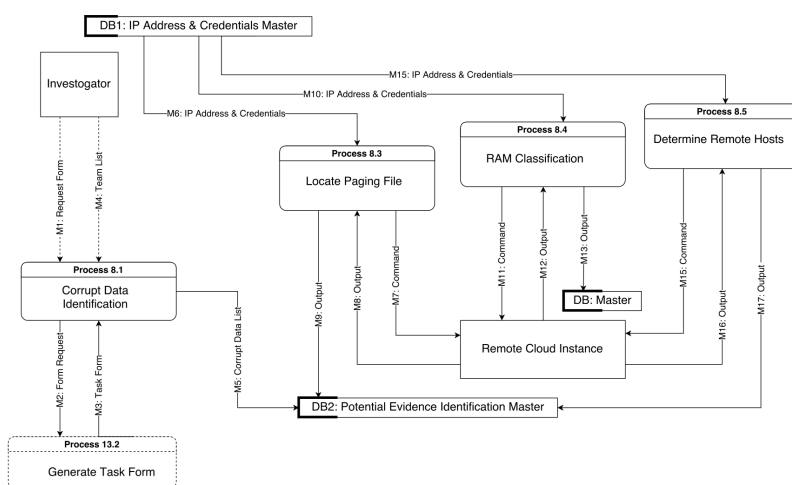


Figure B.6: CFaaS Data Flow Diagram: Diagram 8

B.1.15 PROCESSES

Process 8.1: Corrupt Data Identification - A manual process executed by an investigator by providing a list of files that has been modified recently or that is suspected to be corrupt.

Process 8.2: Generate Task Form - A process that generates a task completion form based on the task at hand.

Process 8.3: Locate Paging File - An automated task that locates a paging file directory.

Process 8.4: RAM Classification - An automated process that gathers information about the RAM from the cloud incident scene.

Process 8.5: Determine Remote Hosts - An automated task process that implements the SLOE algorithm presented in Chapter 6.

B.1.16 MESSAGES

Message M1: Request for a task form by an investigator.

Message M2: Request for a task form forwarded to a relevant Process, Process 8.1.

Message M3: The requested task form is generated and returned to the investigator via Process 8.1.

Message M4: A list of corrupted files provided by the investigator by completing the task completion form.

- Message M5: Corrupted data information written to persistent storage, DB2, by Process 8.1.
- Message M6: Incident scene IP address and credentials obtained from persistent storage to be utilised by Process 8.3 to connect to the incident scene.
- Message M7: An authenticated command sent to the remote cloud instance (cloud incident scene).
- Message M8: Paging file information returned as results from executing the command on the cloud incident scene.
- Message M9: The paging file information is written to persistent storage by Process 8.3.
- Message M10: IP address and credentials of the cloud incident scene are obtained from persistent storage to be utilised by Process 8.4.
- Message M11: An authenticated command is sent to the cloud-based incident scene (Remote cloud instance).
- Message M12: RAM information as output from the command sent to incident scene is sent back to Process 8.4.
- Message M13: The RAM information obtained from the cloud incident scene is written to persistent storage.
- Message M14: IP address and credentials of the cloud incident scene are obtained from persistent storage to be utilised by Process 8.5.
- Message M15: The command to obtain a list of connections is sent to the cloud-based incident scene.
- Message M16: The results, which are a list of connections obtained after executing the command on the remote host, are sent as a response to Process 8.5.
- Message M17: From the list of connections obtained from the cloud incident scene, Process 8.5 computes and prioritises a list of hosts-to-be and the list of hosts is written to persistent storage.

B.1.17 DIAGRAM NINE - EVIDENCE ACQUISITION

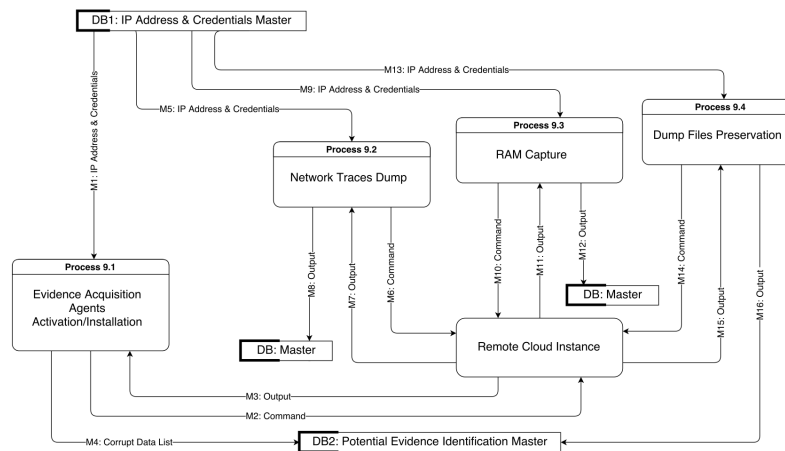


Figure B.7: CFaaS Data Flow Diagram: Diagram 9

B.1.18 PROCESSES

Process 9.1: Evidence acquisition agents and servers installation or activation process. The software agents and/or software servers are installed on the cloud-based incident scene.

Process 9.2: Network Traces Dump - A process that captures network packets on the cloud-based incident scene.

Process 9.3: RAM Capture - A process that copies the RAM from the incident scene for off-line analysis. Off-line analysis is used to complement the live analysis of the cloud-based incident scene.

Process 9.4: Dump Files Preservation - A process that preserves the network traces dump file and the RAM dump files.

B.1.19 MESSAGES

Message M1: Incident scene IP address and credentials obtained from persistent storage to be utilised by Process 9.1.

Message M2: The authenticated command to install/activate investigative software agents is sent to the cloud-based incident scene.

Message M3: Outcomes from executing the command are written to persistent storage.

Message M4: Information about the agents that were installed or activated, and outcomes from executing the command are written to persistent storage.

Message M5: Incident scene IP address and credentials obtained from persistent storage to be utilised by Process 9.2.

Message M6: The command to dump network traces is sent to the cloud-based incident scene.

Message M7: Outcomes from an attempt to execute the network traces dump are received as a response from the remote cloud incident scene.

Message M8: The outcomes resulting from execution of the network traces dump command on the incident scene are written to persistent storage by Process 9.2.

Message M9: Incident scene IP address and credentials obtained from persistent storage to be utilised by Process 9.3.

Message M10: The command to acquire the RAM on the cloud incident scene is sent to the incident scene.

Message M11: The command execution outcomes are sent as a response to Process 9.3.

Message M12: The RAM dump command execution outcomes are written to persistent storage by Process 9.3.

Message M13: Incident scene IP address and credentials obtained from persistent storage are to be utilised by Process 9.4.

Message M14: The command to preserve the dumps files that are temporarily kept on the incident scene is sent to the cloud-based incident scene.

Message M15: The dump files preservation information (HASH sums) is sent to Process 9.4 as a response to execution of the files preservation command.

Message M16: The dump files preservation information is written to persistent storage.

B.1.20 DIAGRAM TEN - EVIDENCE TRANSPORTATION

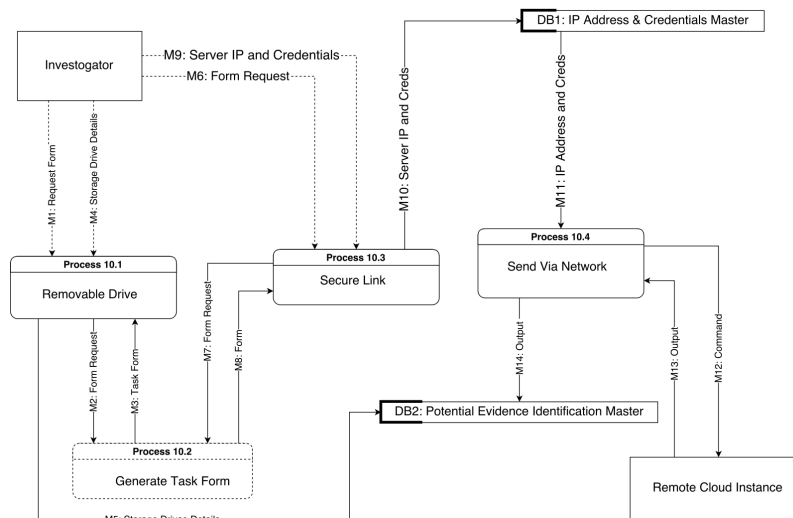


Figure B.8: CFaaS Data Flow Diagram: Diagram 10

B.1.21 PROCESSES

Process 10.1: Removable Drive - Digital evidence may be transported physically from the incident scene through removable storage drives. This may be carried out if investigators have physical access to the incident scene. It is therefore a manual task process carried out by investigators.

Process 10.2: Secure Link - Also a manual task process carried out by investigators. In this task, investigators provide information about the secure storage server on which acquired digital forensic evidence will be stored.

Process 10.3: An automated task that transmits digital evidence to the storage server specified in Process 10.4.

B.1.22 MESSAGES

Message M1: Request for a task form to complete Process 10.1.

Message M2: The request for a task form is forwarded to a relevant process that generates it, i.e. Process 10.2.

Message M3: The generated task form is sent back to Process 10.1 as a response.

Message M4: The evidence storage drive details submitted to Process 10.1 as provided by the investigator.

Message M5: The storage drive details written to persistent storage by Process 10.1.

Message M6: Request for a task form to complete Process 10.3.

Message M7: The request for a task form is forwarded to a relevant process that generates it, i.e. Process 10.3.

Message M8: The generated task form is sent back to Process 10.3 as a response.

Message M9: The evidence storage server IP address and credentials submitted to Process 10.3 as provided by the investigator.

Message M10: The storage server IP address and credentials written to persistent storage by Process 10.3.

Message M11: IP address and credentials to be utilised by Process 10.4.

Message M12: The command to transmit evidence to an online secure storage server is sent to the remote cloud incident scene.

Message M13: The outcomes from executing the command to transmit evidence are sent to Process 10.4 as a response.

Message M14: The outcomes are written to persistent storage.

B.1.23 DIAGRAM ELEVEN - EVIDENCE STORAGE

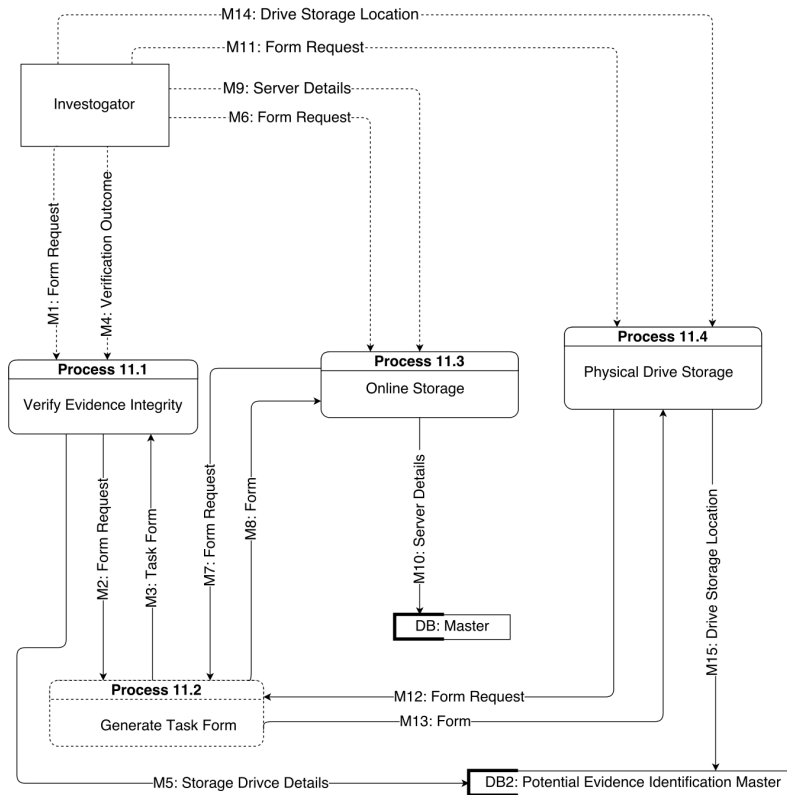


Figure B.9: CFaaS Data Flow Diagram: Diagram 11

B.1.24 PROCESSES

Process 11.1: Verify Evidence Integrity - It is a process that determines whether the evidence files have been altered in transit. This may be done by comparing hash sums calculated before the evidence was transported and the hash sums calculated before storage.

Process 11.2: Online Storage - Storing the evidence on the secure storage server.

Process 11.3: Physical Drive Storage - A manual task carried out by investigators where a description of the storage of the physical drives is presented.

B.1.25 MESSAGES

Message M1: Request for a task form to complete Process 11.1.

Message M2: The request for a task form is forwarded to a relevant process that generates it, i.e. Process 11.2.

Message M3: The generated task form is sent back to Process 11.1 as a response.

Message M4: The outcomes of the evidence integrity verification are submitted to Process 11.1 as provided by the investigator.

Message M5: The evidence integrity verification outcomes are written to persistent storage by Process 11.1.

Message M6: Request for a task form to complete Process 11.3.

Message M7: The request for a task form is forwarded to a relevant process that generates it, i.e. Process 11.2.

Message M8: The generated task form is sent back to Process 11.3 as a response.

Message M9: The secure storage server details submitted to Process 11.3 as provided by the investigator.

Message M10: The secure storage server details are written to persistent storage by Process 11.1.

Message M11: Request for a task form to complete Process 11.4.

Message M12: The request for a task form is forwarded to a relevant process that generates it, i.e. Process 11.2.

Message M13: The generated task form is sent back to Process 11.4 as a response.

Message M14: The evidence storage drive and storage location details are submitted to Process 11.4 as provided by the investigator.

Message M15: The storage drive and storage location details are written to persistent storage by Process 11.4.

B.1.26 DIAGRAM THIRTEEN- ANALYSE EVIDENCE

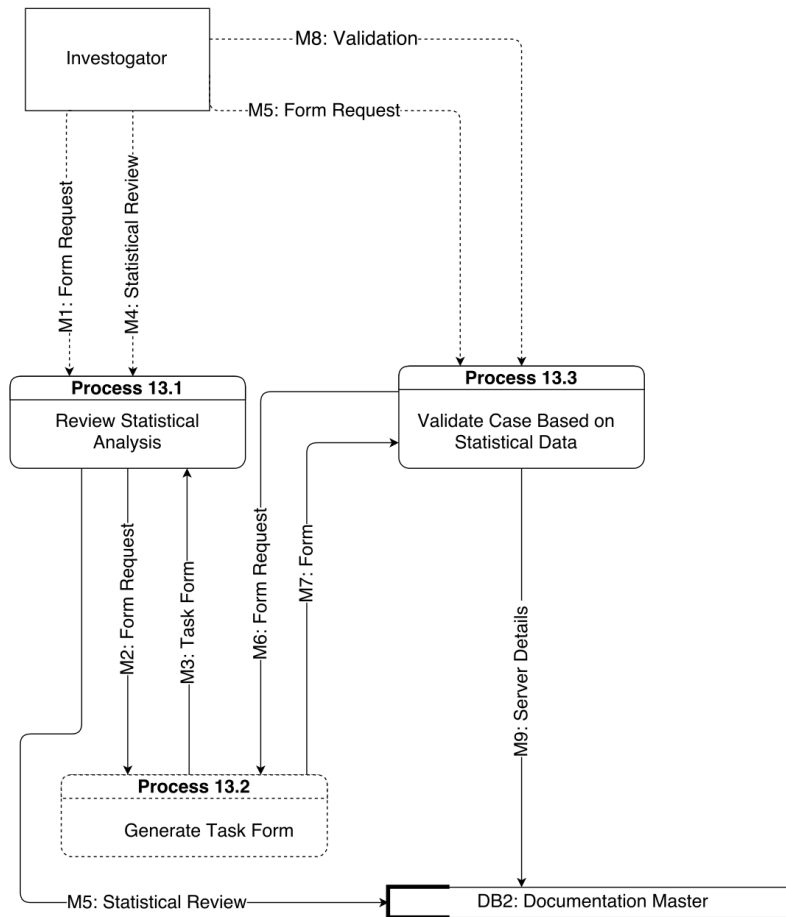


Figure B.10: CFaaS Data Flow Diagram: Diagram 13

B.1.27 PROCESSES

Process 13.1: Review Statistical Analysis - A manual process carried out by investigators where statistical visualisation images are reviewed.

Process 13.2: Generate Tasks Form - A process responsible for the generation of task forms relevant to a manual task being carried out.

Process 13.3: Validate Case Based on Statistical Data - A manual task executed by investigators to confirm the validity of the case.

B.1.28 MESSAGES

Message M1: Request for a task form to complete Process 13.1.

Message M2: The request for a task form is forwarded to a relevant process that generates it, i.e. Process 13.2.

- Message M3: The generated task form is sent back to Process 13.1 as a response.
- Message M4: The statistical review summary submitted to Process 13.1 as provided by the investigator.
- Message M5: The statistical review summary written to persistent storage by Process 13.1.
- Message M6: Request for a task form to complete Process 13.1.
- Message M7: The request for a task form is forwarded to a relevant process that generates it, i.e. Process 13.2.
- Message M8: The generated task form is sent back to Process 13.1 as a response.
- Message M9: The statistical review summary is submitted to Process 13.1 as provided by the investigator.
- Message M10: The statistical review summary is written to persistent storage by Process 13.1.

B.1.29 DIAGRAM FOURTEEN - REPORTING

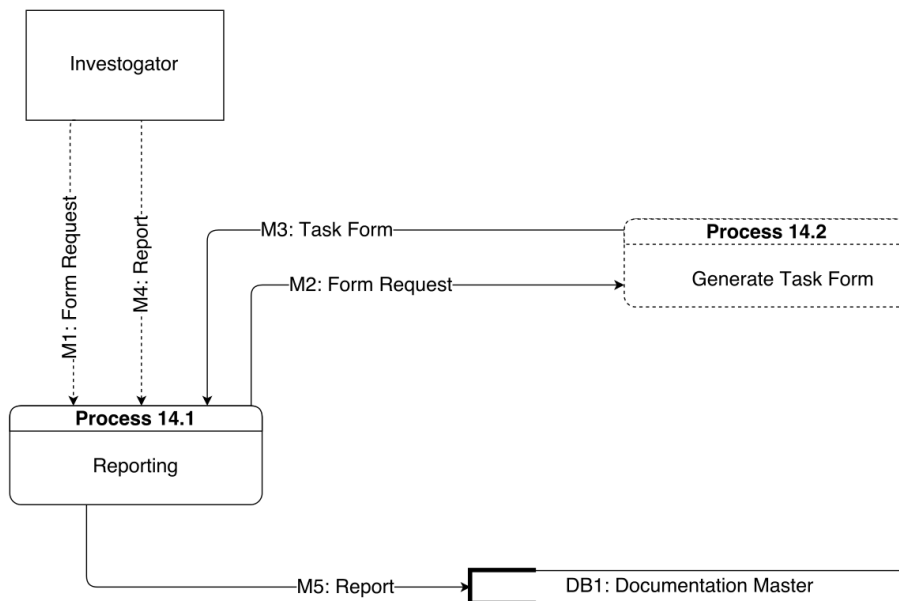


Figure B.11: CFaaS Data Flow Diagram: Diagram 14

B.1.30 PROCESSES

Process 14.1: Reporting - A manual task carried out by investigators where a report is written on the entire investigation.

Process 14.2: Generate Task Form - A process that generates task forms relevant to the manual tasks being executed.

B.1.31 MESSAGES

Message M1: Request for a task form to complete Process 14.1.

Message M2: The request for a task form is forwarded to a relevant process that generates it, i.e. Process 14.2.

Message M3: The generated task form is sent back to Process 14.1 as a response.

Message M4: The case report is submitted to Process 14.1 as provided by an investigator.

Message M5: The case report is written to persistent storage by Process 14.1.

B.1.32 DIAGRAM FIFTEEN - PRESENTATION

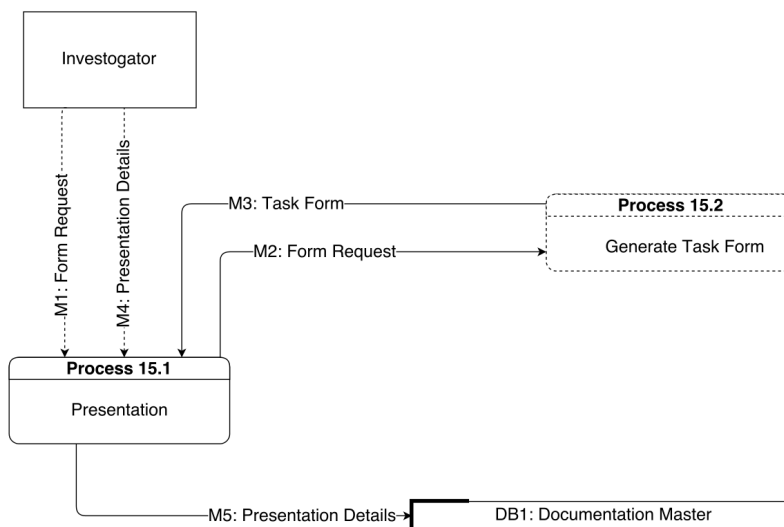


Figure B.12: CFaaS Data Flow Diagram: Diagram 15

B.1.33 PROCESSES

Process 15.1: Presentation - This is a manual process carried out by investigators where the investigation is presented at a hearing or in a court of law.

Process 15.2: Generate Task Form - An automated process responsible for generating task forms corresponding to the manual tasks being carried out.

B.1.34 MESSAGES

Message M1: Request for a task form to complete Process 15.1.

Message M2: The request for a task form is forwarded to a relevant process that generates it, i.e. Process 15.2.

Message M3: The generated task form is sent back to Process 15.1 as a response.

Message M4: The case presentation summary is submitted to Process 15.1 as provided by the investigator.

Message M5: The case presentation summary is written to persistent storage by Process 15.1.

B.1.35 DIAGRAM SIXTEEN - INVESTIGATION CLOSURE

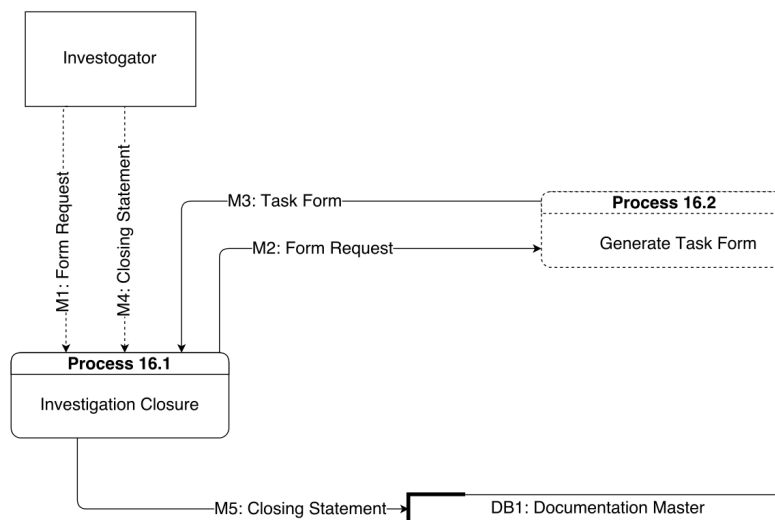


Figure B.13: CFaaS Data Flow Diagram: Diagram 16

B.1.36 PROCESSES

Process 16.1: Investigation Closure - A manual task executed by investigators to conclude the case being investigated.

Process 16.2: Generate Task Form - A process that generates task completion forms relevant to the manual tasks being carried out.

B.1.37 MESSAGES

Message M1: Request for a task form to complete Process 16.1.

Message M2: The request for a task form is forwarded to a relevant process that generates it, i.e. Process 16.2.

Message M3: The generated task form is sent back to Process 16.1 as a response.

Message M4: : The investigation closing statement submitted to Process 16.1 as provided by the investigator.

Message M5: The investigation closing statement is written to persistent storage by Process 16.1.

B.2 DEVELOPMENT VIEW

This section presents the development view of CFaaS. This view provides a graphical representation of the communications among the components presented in the physical view. It shows which component depends on which component for information. The development view comprises of the component diagram depicted in Figure B.14, which contains five main components depicted in Figure B.14, namely Cloud Software, Platform, *Identity and Security Manager*, *CFaaS Server* and the *CFaaS Task Server*. The *Cloud Software* component involves internal components, namely *CFaaS Images* and the *CFaaS Manager* component. The Cloud Software component provides an interface through which new images of the *CFaaS images* can be created. The *CFaaS Manager* is responsible for handling scalability and boots up new instances according to demand, while the *CFaaS Images* component manages the CFaaS virtual machine images.



Figure B.14: CFaaS Component Diagram

The Platform component is the environment provided by an instance booted from *CFaaS images*. It comprises of the database servers, runtime environments (e.g. Java RE), application servers, *SaaS Manager* and *Java Script platform*. The platform furthermore provides interfaces to the internal components through which TLS/HTTPS servers can be launched via commands from higher-level components such as the CFaaS Service component.

The *Identity and Security Manager* handles all user authentication actions in all interactions with service components where authentication is required. The *Identity and Security Manager* is a service and manages user information in a database. The *CFaaS Service* component, *CFaaS Task Server* and the *CFaaS Process Manager* require user authentication throughout their utilisation. Each of these authentications occurs through the Identity and Security Manager component, which exposes an interface through which other components can supply credentials or a request to authenticate. It returns an authentication token for each request.

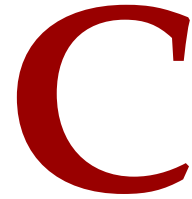
The *CFaaS Process Manager* component is responsible for starting and stopping an investigation process as requested by the *CFaaS Service*. After starting an investigation process, the component hands it over to the *CFaaS Task Server*. Since the

CFaaS Process component is also used to request the status of a running process, it exposes an interface to request the status of a process, start a new process and stop a running process.

The *CFaaS Task Server* component comprises of the Process Engine and the *Persistent Storage* components. The *CFaaS Task Server* component manages an investigation process task from the beginning of the investigation process until the end. The component accepts a digital forensic process template from the *CFaaS Process Manager*, as well as requests to start or stop processes from the *CFaaS Process Manager* component. The process template is loaded on the Process Engine and the persistent storage component stores the statuses and outcomes of each task carried out in a process in a persistent database. This component exposes interfaces to communicate with the *CFaaS Process Manager* component only.

The *CFaaS Service* component is the direct link between an investigator and the whole digital forensic service stack. Through it, investigators can register as required by the digital forensic service before they can utilise it. The *CFaaS Service* component includes other components such as *CFaaS API*, *CFaaS Server* and the *CFaaS Resources* components. The *CFaaS API* comprises of the interfaces that are exposed to an investigator for utilisation and are mainly web interfaces. The *CFaaS Server* implements the business logic of the digital forensic service, while the *CFaaS Resources* component includes a Report Builder and an HTTP client. The functionalities of these components were discussed in detail in Section 7.2.4.

The next section presents the scenario view of the CFaaS design, which will involve use cases of CFaaS.



Processes Task Completion Forms

Establish Secure Connection

Provide the IP address and the credentials to be used in connecting to the incident scene. These credentials will be used in the entire investigation. Scripts that will be interacting with the incident scene will use these credentials to authenticate.

Incident Scene IP Address

User Name

Password

Submit

Figure C.1: Incident Scene Connection Establishment

Forensic Team Organisation

Provide the names of available personnel that will be assigned to investigation tasks.

Lead Investigator

Team Members

Submit

Figure C.2: Forensic Team Organisation

Network Types Identification

Describe the state of the art network types that the incident scene instance is likely to have access to.

Network Type
Local Area Network (LAN)

Network Description
It comprises of servers and network devices usually within the same building.

Network Type
Metropolitan Network (MAN)

Network Description
A network linking multiple sites of an organisation within a city. The link is usually enabled by utilising telecommunications provider services.

Submit

Figure C.3: Network Types Identification

State of the Art RAM and Hardware

Provide state of the art RAM and Hardware that would be relevant for the investigation process

Hardware
Multi-core processors: Include Intel Pentium Dual Core, Intel i3, Intel i5, Intel i5, Intel i7, AMD A4, AMDA6, AMD A8, AMD A10

Hardware Description
The hardware types listed do not have implications on the case being investigated. i.e. The attack is not based at hardware level.

RAM
SDRAM; DDR; RDRAM

RAM Description

Submit

Figure C.4: State of the art RAM and Hardware Identification

Network Evidence Sources

Provide possible sources of evidence in the identified network types.

Network Type
LAN

Evidence Source and Description
Routers, Managed switches (if available),

Network Type
MAN

Evidence Source and Description
Routers, Managed switches and telecommunications providers' network devices.

Submit

Figure C.5: Evidence Sources

Task Assignment

Assign team members to available tasks in the entire investigation.

Task
Physical Drive: Transportation; Physical Drive Transportation

Assignees
Thomas Fogwill
Nozipho Mkhize
Hein Venter

Task
List Active Processes and Analyse List; List Modules Loaded to Kernel and

Assignees
Thomas Fogwill
Nozipho Mkhize
Hein Venter

Task
Perform String Searches on Memory Dump or Live RAM; Anti-forensic

Assignees
Nozipho Mkhize
Hein Venter
George Sibiyi

Submit

Figure C.6: Task Assignments

Evidence Analysis and Acquisition Tools Specification

Provide the specifications for the tools that will be utilised during the investigation both evidence collection tools and evidence analysis tools. Provide a description of each.

Acquisition Tools

Specifications

Analysis Tools

Tools Specifications

Submit

Figure C.7: Tools specifications

Evidence Analysis and Acquisition Tools Selection

List and provide description of acquired tools

Tools Acquired

Tools Description

WireShark and Octave are obtained seperately and are not installed by default.

Submit

Figure C.8: Tools acquisition

Corrupt Data Identification

Use system tools and techniques to list files that were modified after the compromise

Techniques Used

```
lsdf - list open files  
find / -mtime -2  
find / -ctime -2  
find / -user root -perm -4000 -print  
find / -user root -perm -2000 -print
```

Corrupted Files List

/usr/lib/x86_64-linux-gnu/liblcms2.so.2.0.2					
gnome-set 1312	gsib mem	REG	252,0	64288	1324895
/usr/lib/x86_64-linux-gnu/libcanberra.so.0.2.5					
gnome-set 1312	gsib mem	REG	252,0	18784	1324901
/usr/lib/x86_64-linux-gnu/libcanberra-gtk3.so.0.1.8					
gnome-set 1312	gsib mem	REG	252,0	26258	1321585
/usr/lib/x86_64-linux-gnu/gconv/gconv-modules.cache					

Files Description

The files listed are files that were open/loaded in memory at the time of the investigation.

Submit

Figure C.9: Corrupted data identification

Secure Transportation Link

Provide an IP address and credentials of the secure storage server where investigation related data will be stored and managed.

Storage Server IP Address

User Name

Password

Submit

Figure C.10: Securing Online Transportation Link

Removable Drive

If there is evidence stored on a physical removable drive, provide the drive details and the evidence identification stored in the drive.

Evidence ID

Drive ID

Figure C.11: Removable Drive Transportation

Verify Evidence Integrity

As part of the preservation process, a hash key is generated in each creation of evidence. Before evidence can be stored, the keys have to be verified. The keys will be compared internally by the forensic service.

Evidence ID

HASH Key 1

HASH Key 2

Figure C.12: Integrity Verification

Online Storage

Provide details on the storage server and the evidence being stored.

Evidence ID(s)

Storage Server IP Address

Figure C.13: Online Storage

Physical Drive Storage

Provide details of the removable drive being stored and the evidence being stored in the drive.

Evidence ID(s)

Storage Location

Figure C.14: Removable Drive Storage

List Modules

List commands that were run or details of the techniques used to list loaded modules in the incident scene. A list of such modules is also need to be provided.

Commands

Modules List

```
ddm_crypt 23125 1 - Live 0xffffffffa021f000
dvboxvideo 12575 1 - Live 0xffffffffa01c9000 (O)
dsnd_intel8x0 38570 2 - Live 0xffffffffa01d2000
dsnd_ac97_codec 134869 1 snd_intel8x0, Live 0xffffffffa01a7000
dac97_bus 12730 1 snd_ac97_codec, Live 0xffffffffa0170000
dsnd_pcm 97275 2 snd_intel8x0,snd_ac97_codec, Live 0xffffffffa018e000
dsnd_seq_midi 13324 0 - Live 0xffffffffa0189000
```

Figure C.15: Loaded modules identification

List Active Processes

List commands executed to list the processes and the outcomes of each. Techniques used to list the modules can also be provided.

Commands Executed

```
ps auxwww; pstree; top
```

Process List

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME
root	1	0.0	0.0	24568	1148	?	Ss	13:32	0:00
root	2	0.0	0.0	0	0	?	S	13:32	0:00
root	3	0.0	0.0	0	0	?	S	13:32	0:02
root	5	0.0	0.0	0	0	?	S	13:32	0:00
root	6	0.0	0.0	0	0	?	S	13:32	0:00

COMMAND

```
/sbin/init  
[kthreadd]  
[ksoftirqd/0]  
[kworker/u:0]  
[migration/0]
```

Submit

Figure C.16: Active processes listing

Determine Hidden Processes

List uncovered hidden process, analyse and provide their description. This may include processes that are disguised with legitimate process names.

Techniques Applied

```
The following commands were installed in the incident scene:  
  
unhide sys -  
unhide proc -  
unhide brute -
```

Process

```
HIDDEN Processes Found: 1 sysinfo.procs = 442 ps_count = 444
```

Process Description

```
Attempts to analyse the process by "cd /proc/442" were unsuccessful. It is likely that the process is false positive.
```

Process 2

```
HIDDEN Processes Found: 1 sysinfo.procs = 436 ps_count = 438
```

Process 2 Description

```
Attempts to analyse the process by "cd /proc/436" were unsuccessful. It is likely that the process is false positive.
```

Submit

Figure C.17: Hidden processes identification

Allocated Process Memory Space Analysis

List commands that were executed in analysing the memory space. Provide a description of the outcomes.

Sequence of Commands Executed

```
ls -al /proc/PID/fd/ ; pidstat -p PID 2
```

Summary of results

```
lrwx----- 1 root root 64 Mar 24 08:14 8 -> anon_inode:inotify
lrwx----- 1 root root 64 Mar 24 08:14 9 -> anon_inode:[eventfd]
```

pidstat

```
#####
13:12:11 1979 0.00 0.00 0.00 deja-dup-monito
13:12:13 1979 0.00 0.00 0.00 deja-dup-monito
```

Figure C.18: Suspicious processes memory space analysis

Recover Suspicious Process Files

Provide a list of files that are associated with suspicious processes.

Techniques Used

```
list files inside the "/proc/PID/" directory
```

Process ID

```
HIDDEN Processes Found: 1 sysinfo.procs = 442 ps_count = 444
```

File List

```
Navigating into the process directory or listing files in the pricocess directory was not possible. No files could therefore be found.
```

Process ID

```
HIDDEN Processes Found: 1 sysinfo.procs = 436 ps_count = 438
```

File List

```
Navigating into the process directory or listing files in the pricocess directory was not possible. No files could therefore be found.
```

Figure C.19: Suspicious process files identification

String Searching

Provide strings that are being performed on memory dumps.

Technique Used

1. Physical memory dumped with "dd if=/proc/kcore of= ../mem_dump.dd" command
2. Strings extracted from the Memory dump with "strings -t d mem_dump.dd > memdumpstrings"
3. Regular expression search performed on text file

Search String

"j#"; ":-\$"

Results

The regular expression was searched to identify any commands executed in the system. Nothing could be found on in memory that matches the regular expressions.

Search String

"\\.pl"; "\\.py"; "\\.tgz"

Results

The regular expression were searched to find any malicious perl or python script presence in memory. Nothing could be found on in memory that matches the regular expressions.

Submit

Figure C.20: Memory String Searching

Visualize Collected Host Information

Use statistical tools to visualize any quantitative information obtained from string searches. After visualizing provide a brief summary of you observations.

Tools and Techniques Used

Tool - Octave Technique - a scatter plot of number of hits/matches versus a memory search regular expression. Modules loaded to kernel and processes could not be visualised as no suspicious processes were and modules were found.

Visualisation Summary

The strings serched during the "Perform String searches on memory Dump/Live RAM" had no hits. The points on the scatter are all on zero.

Investigation Files Upload

Files to upload:

Choose Files | exhib23.jpg

Status Messages

Submit

Figure C.21: Host Information Visualisation

Anti-forensic rootkit identification

Identify rootkits and provide a description of each.

Techniques Used

rkhunter -c

Rootkit 1

/usr/sbin/adduser

Rootkit Description

[17:54:01] [01:31m]KWarning[m]K: The command '/usr/sbin/adduser' has been replaced by a script: /usr/sbin/adduser: a /usr/bin/perl script, ASCII text executable

Rootkit 2

/usr/bin/lld

Rootkit 2 Description

[01:31] Warning: The command '/usr/bin/lld' has been replaced by a script: /usr/bin/lld: Bourne-Again shell script, ASCII text executable

Rootkits Found

Submit

Figure C.22: Anti-forensic Rootkits Identification

Anti-forensic rootkit deactivation

Deactivate any identified root kits

Techniques Used

The files that were identified as potential rootkits which are /usr/sbin/adduser, /usr/bin/lld, /bin/which, /run/initramfs were analysed and found to be legitigate. Had that not have been the case, the files would have been replaced with legitgmate ones

Outcomes

No deactivation action was taken

Rootkits Succesfully Stopped

Evidence that it stopped

No deactivation action was taken

Submit

Figure C.23: Anti-forensic Rootkits Deactivation

Decode Network Connections

Decode the identified network connections and view any files in the network traffic.

Techniques Used

Wireshark network analysis tool was used.

Connection

Source-00.00.00.78 Dest-00.00.00.99 Protocol-SSH

Observed Payload Description

Encrypted response packet

Message Contents

.Q....Ad.....!711...f...%^.8J.....v...r...ma.S.....li
...7...t...Ol...+.....!...j...>..
.QG...-...+..
.6....Q1...l...A<"...C....Q5v?...vEm.P.2
.v,O.<J..o'S.wU.X...o.o.....V.../v.....1.u.X...5..z.a.....{....

Connection 2

Source - 146.64.8.78 Dest -37.187.236.23 Protocol - TCP

Observed Payload Description

54 50643 > ariel2 [RST] Seq=1 Win=0 Len=0

Submit

Figure C.24: Decoding Network Connections

View Reconstructed Flow

View reconstructed flow from the network connections. This task is carried out if the analysis is performed on a network dump.

Flow ID

FLOW_DUMP1

Flow Description

From this network flow dump, no malicious network traffic could be identified for the network flow dump.

Flow ID

N/A

Flow Description

N/A

Submit

Figure C.25: Network Flow Viewing

View Reconstructed Files

View files obtained from the network flow.

File ID

N/A

File Description

At the time of analysis, no malicious network connections were identified hence, no files were reconstructed. This could have been due to the fact that the "compromised" virtual host was copied and launched in the cloud where the investigation was conducted on it. It therefore has a new IP address unknown to the "attacker".

File ID

N/A

File Description

N/A

Investigation Files Upload

Files to upload:

No file chosen

Status Messages

Figure C.26: Network Files Viewing

D

Evaluations Background Theory

D.1 ALGORITHM CORRECTNESS ANALYSIS RULES

The rules presented in this section are adapted from Zaharie in [155].

Sequential Statements:

If $P \Rightarrow P_0, \{P_{i-1}\}S\{P_i\}$ for $1 \leq i \leq n$ and $P_n \Rightarrow Q$, then $\{P\}S\{Q\}$,

where $\{P_{i-1}\}S\{P_i\}$ represents a transition of the state P from state P_{i-1} to state P_i after the execution of the algorithm S . Q represents the final state or postcondition after the algorithm has exited. The rule means that if all intermediate states (P_0, \dots, P_n) together are such that the final state P_n leads to or implies the postcondition Q , then it can be concluded that the execution of the algorithm S on P leads to the postcondition Q .

Conditional Statements:

If c is well defined,

$\{P \wedge c\}S\{Q\}$ and $\{P \wedge \neg c\}S\{Q\}$ then $\{P\}S\{Q\}$.

This rule means that if the algorithm S would be applied to precondition P with a true value of boolean c leading to postcondition Q , and S is applied to P with a false value of c leading to postcondition Q , then it can be concluded that applying algorithm S on P leads to Q .

Loop Statement:

It is correct if there exists assertion or loop invariant I concerning the state of the algorithm and a function $t : N \rightarrow N$ for which the following rules hold:

1. The assertion I is true at the beginning of the algorithm.
2. I is an invariant property if I is true before executing the algorithm S and c is also true then I remains true after the execution of $A(\{I \wedge c\}S\{I\})$
3. At the end of the loop, the postcondition Q can be inferred from I i.e. $(I \wedge \neg c \Rightarrow Q)$
4. After each execution of A , the value of t decreases, in other words $\{c \wedge (t(p) = k)\}S\{t(p+1) < k\}$, where p can be interpreted as a counting variable associated with each loop execution.
5. If condition c is true, then $t(p) \geq 1$ and when t becomes 0, then the condition c becomes false.

D.2 BIG-O ANALYSIS THEORY

The orders of magnitude are denoted by O-notation (O), Theta-notation (Θ) and Omega-notation (Ω) [81, 128].

The notations are formally defined as follows [81, 128] :

A function $f(n)$ is said to be in $O(g(n))$ i.e., $f(n) \in O(g(n))$ if:

$$f(n) \leq cg(n), \forall n \geq n_0 \quad (\text{D.1})$$

A function $f(n)$ is said to be in $\Theta(g(n))$ i.e., $f(n) \in \Theta(g(n))$ if:

$$f(n) \geq cg(n), \forall n \geq n_0 \quad (\text{D.2})$$

A function $f(n)$ is said to be in $\Omega(g(n))$ i.e., $f(n) \in \Omega(g(n))$ if:

$$c_1f(n) \leq f(n) \leq c_2g(n), \forall n \geq n_0 \quad (\text{D.3})$$

where c is a constant and n_0 is a non-negative integer. The orders of growth $g(n)$ can be seen in Table D.1. From this table it can be observed that the rate of $g(n) = \log_2 n$ is lower than the rest and it is the worst if $g(n) = n!$. If $g(n)$ is of the order 2^n and of the order $n!$. An algorithm with such a growth rate cannot process an input size of more than 10000. Therefore, algorithms with a lower degree order of growth are more scalable than algorithms with higher orders of growth.

Table D.1: $g(n)$ orders of growth [81, p.46].

n	$\log_2 n$	n	$n \log_2 n$	n^2	n^3	2^n	$n!$
10	3.3	10	3.3×10^1	10^2	10^3	10^3	3.6×10^6
10^2	6.6	10^2	6.6×10^2	10^4	10^6	1.3×10^{30}	9.3×10^{157}
10^3	10	10^3	1.0×10^4	10^6	10^9		
10^4	13	10^4	1.3×10^5	10^8	10^{12}		
10^5	17	10^5	1.7×10^6	10^{10}	10^{15}		
10^6	20	10^6	2.0×10^7	10^{12}	10^{18}		

D.3 ATAM ANALYSIS TABLES

Table D.2 contains an analysis of an ease-of-use attribute focusing on clarity of the system. NR1 - In CFaaS, server side and client side are implemented with Meteor which has a separation of duties between the front-end and the back-end service stack. For this reason the front-end processes do not affect the business processes in the service. NR2 - Applications built by using Meteor and other JavaScript applications cannot be indexed by search engines as content is generated dynamically. In CFaaS this reduces the attack risks as adversaries cannot discover the service through search engines.

In Table D.3 the author analysed the scalability of the service while transferring evidence data.

R1 - storing evidence in the cloud expands the attack surface as adversaries would collude with the storage service provider or compromise the third-party storage service. R2- log files are more distributed and it is more difficult to audit the forensic system. NR1 - in this work it is proposed that the forensic service stack be under the control of the investigators, in other words that all IaaS, PaaS and SaaS be managed

Table D.2: Ease of Use : E1 - Analysis

Attribute	Ease of Use and Efficiency		
Scenario #	E1 - The windows are clear for an investigator to use his/her intuition to use and navigate through the CFaaS system.		
Scenario	Normal operation		
Stimulus	Investigator needs to execute a specific investigation task independently		
Response	After logging in, the interface is self-explanatory and the investigator can follow instructions easily.		
Architectural Decisions	Risk/Non-Risk	Sensitivity	Trade-off
Reasoning: Ensures efficient manual tasks execution. Meteor is based on JavaScript and for this reason, these displays are handled on the client side with no server involvement as it affects performance on the CFaaS service.			

internally. This minimises the attack surface on the service and therefore minimises risks.

Table D.3: Scalability: S1 - Analysis

Attribute	Scalability		
Scenario #	S1 - Under normal circumstances, it takes $T(d)$ seconds to transfer 10GB of evidence data; thus it should take $T(d + i)$ seconds to transfer 1TB of data where d is the data size.		
Scenario	Change in input parameters		
Stimulus	An investigator transfers evidence data to a secure storage server.		
Response	System transfers data in time proportional to data size.		
Architectural Decisions	Risk/Non-Risk	Sensitivity	Trade-off
Cloud Storage	R1,R2,NR1		
Reasoning: Data transfer rate is important in cloud environments given that such environments are volatile.			

In Table D.4 the author analyses the scalability of the cloud forensic system based on carrying out process intensive tasks. During the evidence analysis process, tasks such as string searching through evidence data can cost a lot in terms of processing on the system side. Data sizes that an investigator deals with may vary. Since there is a need for a system that can scale up to meet the demand, a software-as-a-service approach is adopted. R1 - The risk involved with software as a service is the distribution of logs, which makes auditing a challenge. NR1 – A SaaS approach

will increase the availability of the service regardless of the failure of any node. S1 - SaaS will affect availability of logs for auditing. T1 - By adopting SaaS, the costs of auditing are increased. NR2 - The orchestration service enables auto-scaling in the digital forensic service.

Table D.4: Scalability: S2 - Analysis

Attribute	Scalability		
Scenario #	S2 - Under normal circumstances, the system returns results in $T(d)$ when string searching 10GB of data. When performing a string search on evidence on 40GB of evidence, results must be returned in $T(d)$.		
Scenario	Change in input parameters		
Stimulus	An investigator		
Response	System returns data in time proportional to data size.		
Architectural Decisions	Risk/Non-Risk	Sensitivity	Trade-off
Software as a Service	R1,NR1	S1	T1
Orchestration Service	NR2		
Reasoning: A SaaS approach enables horizontal and vertical scaling to handle a demand, for instance when more storage resources for evidence data or processing are required. When enabled, the orchestration service handles auto-scaling. This service can scale resources both vertically and horizontally.			

Table 4.5 provides an analysis on security, focusing on the authentication aspect. R1 - If communication breaks down between the LDAP server and the CFaaS service, the service is not usable. NR1 - LDAP enables a single sign-on functionality for the CFaaS service and the task server. In this way, an investigator does not need to login the jBPM service every time he/she needs to execute a task. By signing in to the CFaaS service, the session becomes active in the jBPM service as well.

Table D.6 analyses the authorisation aspect of the security. In an investigation, some members of the investigation team are granted access to perform certain tasks based on their roles. NR1 - JBPM enforces authorisation by showing only the list of tasks that an investigator is authorised to carry out, based on his/her role. The roles that can execute that task are specified in the process template. NR2 – LDAP implements the roles that are specified in a process template when defining a manual forensics task.

Table D.7 analyses the availability aspect of security. NR1 - The deployment of the service in the cloud enables redundant deployment and increases availability. T1 - The use of redundant deployments distributes log files and auditing becomes costly. Efficient audibility is traded.

Table D.5: Security: SE1 - Analysis

Attribute	Security		
Scenario #	SE1 - Investigators using the digital forensic service instance must be authenticated		
Scenario	Under attack		
Stimulus	Need to conduct an cloud forensic investigation investigation		
Response	Prompt user for credentials		
Architectural Decisions	Risk/Non-Risk	Sensitivity	Trade-off
LDAP	R1,NR1		
Reasoning: LDAP is a widely adopted authentication service and it can be used to authenticate different components of the CFaaS service.			

Table D.6: Security: SE2 - Analysis

Attribute	Security		
Scenario #	SE2 - SE2 - An investigator carries out only those tasks he/she is authorised to perform.		
Scenario	Normal operation		
Stimulus	An investigator needs to carry out an investigation task assigned to his/her.		
Response	Only show activities that a user is allowed to execute.		
Architectural Decisions	Risk/Non-Risk	Sensitivity	Trade-off
JBPM	NR1		
LDAP	NR2		
Reasoning: JBPM implements a role-based execution of tasks and LDAP defines and implements those roles.			

Table D.7: Security: SE3 - Analysis

Attribute	Security.		
Scenario #	SE3 - SE3 – Since information residing in the cloud is volatile, the service must be available 99.99% for real-time investigation..		
Scenario	Degraded operation.		
Stimulus	Hardware failure.		
Response	System continues to function without interruption.		
Architectural Decisions	Risk/Non-Risk	Sensitivity	Trade-off
Cloud deployment	NR1		T1
Reasoning: In cloud environments, availability of the service is critical as data is volatile. Analysis or evidence data retrieval needs to be done in real time. The decision to deploy in a cloud environment is informed by the need, and the ability of the cloud provides redundant deployments of the service and hence increases availability.			

References

- [1] Cloud forensics - lab systems solutions. URL http://www.labsystems.co.in/images/Cloud_Forensics_-_A_Lab_systems_approach.pdf.
- [2] Norse - IPViking Live. URL <http://map.ipviking.com/>.
- [3] KDD Cup 1999: Computer network intrusion detection | Sig KDD. URL <http://www.sigkdd.org/kdd-cup-1999-computer-network-intrusion-detection>.
- [4] Computer Language Benchmarks Game. URL <http://benchmarksgame.alioth.debian.org/u32/javascript.php>.
- [5] Incident Reporting System | US-CERT, 2014. URL <https://www.us-cert.gov/government-users/reporting-requirements>.
- [6] jBPM - JBoss Community, 2014. URL <http://www.jboss.org/jbpm>.
- [7] Meteor, 2014. URL <https://www.meteor.com/>.
- [8] MongoDB, 2014. URL <http://www.mongodb.org/>.
- [9] AccessData. Computer Forensics Software for Digital Investigations-FTK, 2014. URL <http://www.accessdata.com/products/digital-forensics/ftk>.
- [10] Rares Aioanei. Learning Linux commands: dd - LinuxCareer's Documentation, 2011. URL <http://how-to.linuxcareer.com/learning-linux-commands-dd>.
- [11] M. Al Fahdi, N.L. Clarke, and S.M. Furnell. Challenges to digital forensics: A survey of researchers & practitioners attitudes and opinions. In *2013 Information Security for South Africa*, pages 1–8. IEEE, August 2013. ISBN 978-1-4799-0808-0. doi: 10.1109/ISSA.2013.6641058. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6641058>.
- [12] Sabah Al-fedaghi and Bashayer Al-babtain. Modeling the Forensics Process. 6 (4):97–108, 2012.
- [13] Alessandro Aldini, Javier Lopez, and Fabio Martinelli, editors. *Foundations of Security Analysis and Design VII*, volume 8604 of *Lecture Notes in Computer*

- Science*. Springer International Publishing, Cham, 2014. ISBN 978-3-319-10081-4. doi: 10.1007/978-3-319-10082-1. URL <http://link.springer.com/10.1007/978-3-319-10082-1>.
- [14] M. ALLIOT-MARIE. Council Framework Decision 2008/977/JHA of 27 November 2008 on the protection of personal data processed in the framework of police and judicial cooperation in criminal matters, 2008.
- [15] S. Alshehri, S.P. Radziszowski, and R.K. Raj. Secure access for healthcare data in the cloud using ciphertext-policy attribute-based encryption. In *Data Engineering Workshops (ICDEW), 2012 IEEE 28th International Conference on*, pages 143–146, April 2012. doi: 10.1109/ICDEW.2012.68.
- [16] J. Andress. *The Basics of Information Security: Understanding the Fundamentals of InfoSec in Theory and Practice*. The Basics. Elsevier Science, 2014. ISBN 9780128008126. URL <https://books.google.co.za/books?id=9NIOAwAAQBAJ>.
- [17] Krzysztof R. Apt, Frank S. de Boer, and Ernst-Rüdiger Olderog. *Verification of Sequential and Concurrent Programs*. Springer London, London, 2009. ISBN 978-1-84882-744-8. doi: 10.1007/978-1-84882-745-5. URL <http://www.springer.com/computer/theoretical+computer+science/book/978-1-84882-744-8>.
- [18] Model Venansius Baryamureeba and Florence Tushabe. The enhanced digital investigation process. In *Digital Forensic Research Workshop*, 2004.
- [19] Robert Beverly, Simson Garfinkel, and Greg Cardwell. Forensic carving of network packets and associated data structures. *Digital Investigation*, 8:S78–S89, August 2011. ISSN 17422876. doi: 10.1016/j.diin.2011.05.010. URL <http://linkinghub.elsevier.com/retrieve/pii/S174228761100034X>.
- [20] Dominik Birk. Technical Challenges of Forensic Investigations in Cloud Computing Environments. January 2011. URL www.zurich.ibm.com/~cca/csc2011/submissions/birk.pdf.
- [21] Dominik Birk, Michael Panico, Aaron Alva, Bernd Jaeger, Dominik Birk, Josiah Dykstra, Keyun Ruan, Michael Panico, and Richard Austin. Mapping the Forensic Standard ISO/IEC 27037 to Cloud Computing, 2013. URL <https://downloads.cloudsecurityalliance.org/initiatives/imf/Mapping-the-Forensic-Standard-ISO-IEC-27037-to-Cloud-Computing.pdf>.
- [22] John W. Brackett. Software Requirements, 1990. URL http://resources.sei.cmu.edu/asset_files/curriculummodule/1990_007_001_15809.pdf.
- [23] Halil Ibrahim Bulbul, H Guclu Yavuzcan, and Mesut Ozel. Digital forensics: an analytical crime scene procedure model (ACSPM). *Forensic science*

- international*, 233(1-3):244–56, December 2013. ISSN 1872-6283. doi: 10.1016/j.forsciint.2013.09.007. URL <http://www.sciencedirect.com/science/article/pii/S0379073813004155>.
- [24] Mariusz Burdach. Forensic Analysis of a Live Linux System, Pt. 1 | Symantec Connect Community, 2010. URL <http://www.symantec.com/connect/articles/forensic-analysis-live-linux-system-pt-1>.
- [25] Mariusz Burdach. Forensic Analysis of a Live Linux System, Pt. 2 | Symantec Connect Community, 2010. URL <http://www.symantec.com/connect/articles/forensic-analysis-live-linux-system-pt-2>.
- [26] Rajkumar Buyya, Chee Shin, Srikumar Venugopal, James Broberg, and Ivona Brandic. Cloud computing and emerging IT platforms : Vision , hype , and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, 25(6):599–616, 2009. ISSN 0167-739X. doi: 10.1016/j.future.2008.12.001. URL <http://dx.doi.org/10.1016/j.future.2008.12.001>.
- [27] Cafésoft. SSL/TLS Mutual Authentication Primer. URL <http://www.cafesoft.com/products/cams/ps/docs31/admin/SSLTLSPrimer.html>.
- [28] Eoghan Casey. Attacks against forensic analysis. *Digital Investigation*, 4(3-4):105–106, September 2007. ISSN 17422876. doi: 10.1016/j.diin.2008.01.001. URL <http://www.sciencedirect.com/science/article/pii/S1742287608000029>.
- [29] Eoghan Casey, Gary Katz, and Joe Lewthwaite. Honing digital forensic processes. *Digital Investigation*, 10(2):138–147, September 2013. ISSN 17422876. doi: 10.1016/j.diin.2013.07.002. URL <http://www.sciencedirect.com/science/article/pii/S1742287613000753>.
- [30] Chen Chen, Himanshu Raj, and Stefan Saroiu. cTPM: A Cloud TPM for Cross-Device Trusted Applications. In *11th USENIX Symposium on Networked Systems Design and Implementation*, pages 187–201, 2014. ISBN 9781931971096.
- [31] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein. *Introduction To Algorithms*. MIT Press, 2001. ISBN 9780262032933. URL https://books.google.co.za/books?id=NLngYyWFl_YC.
- [32] Alan Cox, Jean Delvare, and Anton Arapov. dmidecode, 2015. URL <http://www.nongnu.org/dmidecode/>.
- [33] Philip Craiger, Paul Burke, Christopher Marberry, and Mark Pollitt. A virtual digital forensics laboratory. In Indrajit Ray and Sujeet Shenoj, editors, *Advances in Digital Forensics IV*, volume 285 of *IFIP — The International Federation for Information Processing*, pages 357–365. Springer US, 2008. ISBN 978-0-387-84926-3. doi: 10.1007/978-0-387-84927-0_28. URL http://dx.doi.org/10.1007/978-0-387-84927-0_28.

- [34] Cve. CVE - Common Vulnerabilities and Exposures (CVE). URL <https://cve.mitre.org/>.
- [35] Gero Decker and Mathias Weske. Instance Isolation Analysis for Service-Oriented Architectures. In *2008 IEEE International Conference on Services Computing*, volume 1, pages 249–256. IEEE, July 2008. ISBN 978-0-7695-3283-7. doi: 10.1109/SCC.2008.44. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4578470>.
- [36] N Deepak, B Arunkumar, and T.V.P Sundararajan. Digital forensics service platform for internet videos. 2:456–464, 2013.
- [37] Omer Demir. *A scalable agent-based system for network flow reconstruction with applications to determining the structure and dynamics of distributed denial of service attacks*. Doctoral, City University of New York, 2010.
- [38] Dave Dittrich. Basic Steps in Forensic Analysis of Unix Systems, 2014. URL <http://staff.washington.edu/dittrich/misc/forensics/>.
- [39] Josiah Dykstra. Seizing Electronic Evidence from Cloud Computing Environments. In Keyun Ruan, editor, *Cybercrime and Cloud Forensics: Applications for Investigation Processes*, pages 156–185. IGI Global, Hershey, PA, USA, 2013. ISBN 9781466626621. doi: 10.4018/978-1-4666-2662-1.ch007. URL <http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/978-1-4666-2662-1.ch007>.
- [40] Josiah Dykstra and Alan T. Sherman. Design and implementation of FROST: Digital forensic tools for the OpenStack cloud computing platform. *Digital Investigation*, 10:S87–S95, 2013. URL <http://www.sciencedirect.com/science/article/pii/S174228761300056X>.
- [41] Nathaniel B. Edmonds and Peter Robert Zeidenberg. United States of America, v. David Hossein Safavein, Defendant., 2006. URL <https://casetext.com/case/us-v-safavian-4>.
- [42] Abdelaziz Ettaoufik and Mohamed Ouzzif. Query’s optimization in data warehouse on the cloud using fragmentation. In *2014 International Conference on Next Generation Networks and Services (NGNS)*, pages 145–148. IEEE, May 2014. ISBN 978-1-4799-6937-1. doi: 10.1109/NGNS.2014.6990243. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6990243>.
- [43] European Commission. DIRECTIVE 95/46/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data, 1995. URL <http://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:31995L0046&from=EN>.

- [44] Wenfei Fan and Wu. *Advances in Web-Age Information Management, 6th International Conference, WAIM 2005, Hangzhou, China, October 11-13, 2005, Proceedings*, volume 3739 of *Lecture Notes in Computer Science*. Springer, 2005. ISBN 3540292276.
- [45] B. Fei, J. Eloff, H. Venter, and M. Olivier. Exploring forensic data with self-organizing maps. In Mark Pollitt and Sujeet Shenoj, editors, *Advances in Digital Forensics*, volume 194 of *IFIP — The International Federation for Information Processing*, pages 113–123. Springer US, 2005. ISBN 978-0-387-30012-2. doi: 10.1007/0-387-31163-7_10. URL http://dx.doi.org/10.1007/0-387-31163-7_10.
- [46] Marcel B. Finan. *Lecture Notes in Discrete Mathematics*. Arkansas Tech University, 2001.
- [47] B.A. Forouzan and S.C. Fegan. *Data Communications and Networking*. McGraw-Hill Forouzan networking series. McGraw-Hill Higher Education, 3rd edition, 2003. ISBN 9780072923544. URL <https://books.google.co.za/books?id=U3Gcf65Pu9IC>.
- [48] Great. Stuxnet: Zero Victims - Securelist, 2014. URL <http://securelist.com/analysis/publications/67483/stuxnet-zero-victims/>.
- [49] Guowu Xie, M. Iliofotou, T. Karagiannis, M. Faloutsos, and Yaohui Jin. ReSurf: Reconstructing web-surfing activity from network traffic, 2013.
- [50] Gure KDD CUP. GureKddcup database description, 2014. URL <http://www.sc.ehu.es/acwaldap/gureKddcup/README.pdf>.
- [51] Charles B. Haley, Robin C. Laney, Jonathan D. Moffett, and Bashar Nuseibeh. Using trust assumptions with security requirements. *Requirements Engineering*, 11(2):138–151, 2006. ISSN 09473602. doi: 10.1007/s00766-005-0023-4.
- [52] Robert Hegarty, Madjid Merabti, Qi Shi, and Bob Askwith. forensic analysis of distributed Service oriented comtuting platforms. *Sciences-New York*, 2011.
- [53] Paul Henry, Jacob Williams, and Benjamin Wright. The SANS Survey of Digital Forensics and Incident Response. Technical Report July, SANS, 2013. URL https://blogs.sans.org/computer-forensics/files/2013/07/sans_dfir_survey_2013.pdf.
- [54] Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Sudha Ram. Design science in information systems research. *MIS Quarterly*, 28(1):75–105, March 2004. ISSN 0276-7783. URL <http://dl.acm.org/citation.cfm?id=2017212.2017217>.
- [55] Emma Webb Hobson. QinetiQ White Paper: Digital Investigations in the cloud, 2010.

- [56] Amir Houmansadr, Saman A Zonouz, and Robin Berthier. A cloud-based intrusion detection and response system for mobile phones. *Security*, 2011.
- [57] Fredesvinda Insa. The admissibility of electronic evidence in court (a.e.e.c.): Fighting against high-tech crime—results of a european study. *Journal of Digital Forensic Practice*, 1(4):285–289, 2007. doi: 10.1080/15567280701418049. URL <http://dx.doi.org/10.1080/15567280701418049>.
- [58] International Organisation for Standardization. ISO Standards - ISO. URL <http://www.iso.org/iso/home/standards.htm>.
- [59] International Organisation for Standardization. ISO/IEC 27037:2012 - Information technology – Security techniques – Guidelines for identification, collection, acquisition and preservation of digital evidence, 2012. URL http://www.iso.org/iso/catalogue_detail?csnumber=44381.
- [60] International Organisation for Standardization. ISO/IEC 27001:2013 - Information technology – Security techniques – Information security management systems – Requirements, 2013. URL http://www.iso.org/iso/catalogue_detail?csnumber=54534.
- [61] International Organisation for Standardization. ISO/IEC 27043:2015 - Information technology - Security techniques - Incident investigation principles and processes. 2015.
- [62] M. H. Jansen-vullers and M. Netjes. Business process simulation – a tool survey. In *In Workshop and Tutorial on Practical Use of Coloured Petri Nets and the CPN*, 2006.
- [63] Jinho Hwang, Sai Zeng, F.Y. Wu, and T. Wood. A component-based performance comparison of four hypervisors, 2013.
- [64] JUSTIA Dockets & Filings. United States of America v. Ulbricht et al :: Justia Dockets & Filings, 2013. URL <https://dockets.justia.com/docket/new-york/nysdce/1:2013cv06919/418116>.
- [65] Atul Kant Kaushik and R. C. Joshi. Network Forensic System for ICMP Attacks. *International Journal of Computer Applications*, 2(3):14–21, May 2010. ISSN 09758887. doi: 10.5120/649-906. URL <http://www.ijcaonline.org/volume2/number3/pxc387906.pdf>.
- [66] Rick Kazman, Mark Klein, Mario Barbacci, Tom Longstaff, Howard Lipson, and Jeromy Carriere. The architecture tradeoff analysis method. In *Proceedings of the Fourth IEEE International Conference on Engineering of Complex Computer Systems (ICECCS)*, 1998.
- [67] Kenneth E. Kendall and Julie E. Kendall. *System analysis and design*. Prentice Hall, 8 edition, 2011. ISBN 9780136089162.

- [68] Karen Kent, Suzanne Chevalier, Tim Grance, and Hung Dang. Guide to Integrating Forensic Techniques into Incident Response. *Nist Special Publication*, 2006. URL <http://csrc.nist.gov/publications/nistpubs/800-86/SP800-86.pdf>.
- [69] Philippe Kruchten. The 4+1 view model of architecture. *IEEE Softw.*, 12(6):42–50, November 1995. ISSN 0740-7459. doi: 10.1109/52.469759. URL <http://dx.doi.org/10.1109/52.469759>.
- [70] Gurudatt Kulkarni, Ramesh Sutar, and Jayant Gambhir. Cloud computing-infrastructure as service- Amazon ec2. 2(1):117–125, 2012.
- [71] Ihor Kuz, Liming Zhu, Len Bass, Mark Staples, and Xiwei Xu. An Architectural Approach for Cost Effective Trustworthy Systems. In *2012 Joint Working IEEE/IFIP Conference on Software Architecture and European Conference on Software Architecture*, pages 325–328. IEEE, August 2012. ISBN 978-1-4673-2809-8. doi: 10.1109/WICSA-ECSA.212.54. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6337744>.
- [72] KVM. Kernel Based Virtual Machine. URL http://www.linux-kvm.org/page/Main_Page.
- [73] K. Lakkaraju, W. Yurcik, R. Bearavolu, and A.J. Lee. NVisionIP: an interactive network flow visualization tool for security. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*, volume 3, pages 2675–2680. IEEE, 2004. ISBN 0-7803-8567-5. doi: 10.1109/ICSMC.2004.1400735. URL <http://ieeexplore.ieee.org/articleDetails.jsp?arnumber=1400735>.
- [74] C. Latze, U. Ultes-Nitsche, and F. Baumgartner. Strong mutual authentication in a user-friendly way in eap-tls. In *Software, Telecommunications and Computer Networks, 2007. SoftCOM 2007. 15th International Conference on*, pages 1–5, Sept 2007. doi: 10.1109/SOFTCOM.2007.4446137.
- [75] LDAP. OpenLDAP, Main Page. URL <http://www.openldap.org/>.
- [76] N Leavitt. Mobile Security: Finally a serious problem?, June 2011. URL http://www.leavcom.com/ieee_jun11.php.
- [77] J Y Lee, S K Un, Y S Kim, G W Kim, S S Lee, S H Jo, Y H Gil, W Y Choi, D W Hong, and H S Cho. Remote forensics system based on network, 2011. URL <https://www.google.com/patents/US20110153748>.
- [78] Jooyoung Lee and Sungyong Un. Digital forensics as a service: A case study of forensic indexed search. In *2012 International Conference on ICT Convergence (ICTC)*, pages 499–503. IEEE, October 2012. ISBN 978-1-4673-4828-7. doi: 10.1109/ICTC.2012.6387185. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6387185>.

- [79] Taerim Lee, Hyejoo Lee, Kyung-Hyune Rhee, and Uk Shin. The efficient implementation of distributed indexing with Hadoop for digital investigations on Big Data. *Computer Science and Information Systems*, 11(3):1037–1054, 2014. ISSN 1820-0214. doi: 10.2298/CSIS130920063L. URL <http://www.doiserbia.nb.rs/Article.aspx?ID=1820-02141400063L>.
- [80] Ryan Leigland. A Formalization of Digital Forensics. *International Journal of Digital Evidence*, 3(2):1–32, 2004.
- [81] Anany Levitin. *Introduction to the Design and Analysis of Algorithms*. 3rd edition, 2011. ISBN 0132316811.
- [82] Jack Li, Qingyang Wang, Deepal Jayasinghe, Junhee Park, Tao Zhu, and Calton Pu. Performance Overhead among Three Hypervisors: An Experimental Study Using Hadoop Benchmarks. *2013 IEEE International Congress on Big Data*, pages 9–16, June 2013. doi: 10.1109/BigData.Congress.2013.11. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6597113>.
- [83] I-Long Lin, Yun-Sheng Yen, and Annie Chang. A Study on Digital Forensics Standard Operation Procedure for Wireless Cybercrime. In *2011 Fifth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, pages 543–548. IEEE, June 2011. ISBN 978-1-61284-733-7. doi: 10.1109/IMIS.2011.58. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5976271>.
- [84] Rob Lovell. White Paper: Introduction to cloud computing, 2011. URL www.thinkgrid.com/docs/computing-whitepaper.pdf-UnitedStates.
- [85] Steven Malby, Robyn Mace, Anika Holterhof, Cameron Brown, Stefan Kascherus, and Eva Ignatuschtschenko. Comprehensive Study on Cybercrime. Technical report, Viena, 2013. URL https://www.unodc.org/documents/organized-crime/UNODC_CCPCJ_EG.4_2013/CYBERCRIME_STUDY_210213.pdf.
- [86] Marie-Helen Maras. Inside Darknet: the takedown of Silk Road. *Criminal Justice Matters*, 98(1):22–23, November 2014. ISSN 0962-7251. doi: 10.1080/09627251.2014.984541. URL <http://www.tandfonline.com/doi/abs/10.1080/09627251.2014.984541#.ValadnU4Z4s>.
- [87] Raffael Marty. Cloud application logging for forensics. In *Business*, Taichung, Taiwan, March 2011. SAC, ACM. ISBN 9781450301138.
- [88] Tim Mather, Subra Kumaraswamy, and Shahed Latif. *Cloud Security and Privacy: An Enterprise Perspective on Risks and Compliance*. O'REILLY, 2009.
- [89] H. Maziku, S. Shetty, K. Han, and T. Rogers. Enhancing the classification accuracy of ip geolocation. In *Military communications conference, 2012 - Milcom 2012*, pages 1–6, Oct 2012. doi: 10.1109/MILCOM.2012.6415842.

- [90] Microsoft. NW3C - Microsoft COFEE, 2014. URL <https://cofee.nw3c.org/>.
- [91] Aikaterini Mitrokotsa and Christos Dimitrakakis. Intrusion detection in MANET using classification algorithms: The effects of cost and model selection. *Ad Hoc Networks*, 11(1):226–237, January 2013. ISSN 15708705. doi: 10.1016/j.adhoc.2012.05.006. URL <http://linkinghub.elsevier.com/retrieve/pii/S1570870512001011>.
- [92] Silver Moon. Code a network packet sniffer in python for Linux, 2011. URL <http://www.binarytides.com/python-packet-sniffer-code-linux/>.
- [93] Antonis Mouhtaropoulos, Marthie Grobler, and Chang-Tsun Li. Digital Forensic Readiness: An Insight into Governmental and Academic Initiatives. *2011 European Intelligence and Security Informatics Conference*, pages 191–196, September 2011. doi: 10.1109/EISIC.2011.30. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6061177>.
- [94] Michael B Mukasey, Jeffrey L Sedgwick, and David W Hagy. Electronic Crime Scene Investigation : A Guide for First Responders , Second Edition. Technical report, US, Department of Justice, Washinton, DC, 2008.
- [95] Nodejs. Nodejs, 2014. URL <http://nodejs.org>.
- [96] OWASP. OWASP Top 10 - 2012: THE Ten Most Critical Web Application Security Risks, 2013. URL https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project.
- [97] Nicolas Prat, Isabelle Comyn-Wattiau, and Jacky Akoka. Artifact evaluation in information systems: Design science research - a holistic view, 2014. URL <http://aisel.aisnet.org/pacis2014/23>.
- [98] N. Preve. *Computational and Data Grids: Principles, Applications and Design: Principles, Applications and Design*. Premier reference source. Information Science Reference, 2011. ISBN 9781613501146. URL <https://books.google.co.za/books?id=8JKReC0yxKEC>.
- [99] Darren Quick and Kim-Kwang Raymond Choo. Impacts of increasing volume of digital forensic data: A survey and future research challenges. *Digital Investigation*, 11(4):273–294, December 2014. ISSN 17422876. doi: 10.1016/j.diin.2014.09.002. URL <http://www.sciencedirect.com/science/article/pii/S1742287614001066>.
- [100] Darren Quick, Ben Martini, and Kim-Kwang Raymond Choo. Chapter 2 - cloud storage forensic framework. In Darren Quick, Ben Martini, and Kim-Kwang Raymond Choo, editors, *Cloud Storage Forensics*, pages 13 – 21. Syngress, Boston, 2014. ISBN 978-0-12-419970-5. doi: <http://dx.doi.org/10.1016/B978-0-12-419970-5.00002-8>. URL <http://www.sciencedirect.com/science/article/pii/B9780124199705000028>.

- [101] A.M.G. Rani and A. Marimuthu. Key insertion and splay tree encryption algorithm for secure data outsourcing in cloud. In *Computing and Communication Technologies (WCCCT), 2014 World Congress on*, pages 92–97, Feb 2014. doi: 10.1109/WCCCT.2014.14.
- [102] Archie Reed, Chris Rezek, and Paul Simmonds. Security guidance for critical areas of focus in cloud computing V3.0. Technical report, CSA, 2011.
- [103] Irma Resendez, Pablo Martinez, and John Abraham. An Introduction to Digital Forensics. 2008. URL http://acetweb.org/journal/ACETJournal_Vol6/An%20Introduction%20to%20Digital%20Forensics.pdf.
- [104] Beth Rowen. Cyberwar Timeline: The roots of this increasingly menacing challenge facing nations and businesses. URL <http://www.infoplease.com/world/events/cyberwar-timeline.html>.
- [105] Robert Rowlingson. A Ten Step Process for Forensic Readiness. 2(3):1–28, 2004.
- [106] Anne Rozinat, Moe Wynn, Wil van der Aalst, Arthur ter Hofstede, and Colin Fidge. Workflow simulation for operational decision support using design, historic and state information. In Marlon Dumas, Manfred Reichert, and Ming-Chien Shan, editors, *Business Process Management*, volume 5240 of *Lecture Notes in Computer Science*, pages 196–211. Springer Berlin Heidelberg, 2008. ISBN 978-3-540-85757-0.
- [107] Keyun Ruan, Joe Carthy, Tahar Kechadi, Mark Crosbie, Ibrahim Baggili, Prof Joe Carthy, and Prof Tahar Kechadi. Survey on cloud forensics and critical criteria for cloud forensic capability: A preliminary analysis. 2010.
- [108] Keyun Ruan, Joe Carthy, Tahar Kechadi, and Mark Crosbie. Cloud forensics. In Gilbert Peterson and Sujeet Sheno, editors, *Advances in Digital Forensics VII*, volume 361 of *IFIP Advances in Information and Communication Technology*, pages 35–46. Springer Berlin Heidelberg, 2011. ISBN 978-3-642-24211-3. doi: 10.1007/978-3-642-24212-0_3. URL http://dx.doi.org/10.1007/978-3-642-24212-0_3.
- [109] Keyun Ruan, Prof Joe Carthy, Prof Tahar Kechadi, Mark Crosbie, Joe Carthy, and Tahar Kechadi. Cloud forensics: An overview. 2011. URL http://www.google.com/url?sa=t&source=web&cd=2&ved=0CCEQFjAB&url=http://cloudforensicsresearch.org/publication/Cloud_Forensics_An_Overview_7th_IFIP.pdf&rct=j&q=cloudforensics:anoverview&ei=r00fTub5Ls04hAe2jo3RAw&usg=AFQjCNE1scUVGct6IoNCS7jfqrnJ00iAsQ&sig2=Jkq0KiXoM7khR0vL1UxDfA&cad=rja.
- [110] Ting Sang. A Log Based Approach to Make Digital Forensics Easier on Cloud Computing. In *2013 Third International Conference on Intelligent System Design and Engineering Applications*, pages 91–94. IEEE, January

2013. ISBN 978-1-4673-4893-5. doi: 10.1109/ISDEA.2012.29. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6454779>.
- [111] SANS. SANS SIFT Kit/Workstation: Investigative Forensic Toolkit, 2014. URL <http://digital-forensics.sans.org/community/downloads>.
- [112] Searchsec. What is attack vector? - Definition from WhatIs.com. URL <http://searchsecurity.techtarget.com/definition/attack-vector>.
- [113] Seung-Hyun Seo, M. Nabeel, Xiaoyu Ding, and E. Bertino. An efficient certificateless encryption for secure data sharing in public clouds. *Knowledge and Data Engineering, IEEE Transactions on*, 26(9):2107–2119, Sept 2014. ISSN 1041-4347. doi: 10.1109/TKDE.2013.138.
- [114] Y. Shavitt and N. Zilberman. A geolocation databases study. *Selected Areas in Communications, IEEE Journal on*, 29(10):2044–2056, December 2011. ISSN 0733-8716. doi: 10.1109/JSAC.2011.111214.
- [115] J R G SHENDE. Cloud forensics, 2014. URL <http://www.google.com/patents/W02014145626A1?cl=en>.
- [116] Yong-Dal Shin. New Digital Forensics Investigation Procedure Model. *2008 Fourth International Conference on Networked Computing and Advanced Information Management*, pages 528–531, September 2008. doi: 10.1109/NCM.2008.116. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4624063>.
- [117] Deoyani Shirkhedkar and Sulabha Patil. Design of digital forensic technique for cloud computing. 2(6):192–194, 2014.
- [118] Shorewall. Shoreline Firewall (Shorewall), 2015. URL <http://shorewall.net/>.
- [119] G. Sibiyi, H.S. Venter, S. Ngobeni, and T. Fogwill. Guidelines for procedures of a harmonised digital forensic process in network forensics. In *Information Security for South Africa (ISSA), 2012*, pages 1–7, Aug 2012. doi: 10.1109/ISSA.2012.6320451.
- [120] George Sibiyi, Hein S Venter, and Thomas Fogwill. Digital Forensic Framework for a Cloud Environment. In Paul Cunningham and Miriam Cunningham, editors, *IST-Africa 2012*, pages 1–8, Dar es Salam, 2012. IIMC. ISBN 9781905824342.
- [121] George Sibiyi, Hein S Venter, and Thomas Fogwill. Digital Forensic Framework for a Cloud Environment. In Paul Cunningham and Miriam Cunningham, editors, *IST-Africa 2012*, pages 1–8, Dar es Salam, 2012. IIMC. ISBN 9781905824342.

- [122] George Sibiya, Hein S Venter, and Thomas Fogwill. Procedures for a Harmonized Digital Forensic Process in Live Forensics. In *SATNAC 2012*. SATNAC, 2012.
- [123] George Sibiya, Thomas Fogwill, and H S Venter. Selection and ranking of remote hosts for digital forensic investigation in a Cloud environment. In *Information Security for South Africa, 2013*, pages 1–5, 2013. doi: 10.1109/ISSA.2013.6641044.
- [124] George Sibiya, Thomas Fogwill, H.S. Venter, and Siphon Ngobeni. Digital Forensic Readiness in a Cloud Environment. In *AFRICON2013*, volume 1, Mauritius, 2013.
- [125] George Sibiya, Hein S Venter, and Thomas Fogwill. Digital forensics in the cloud: The state of the art. In *IST-Africa 2015 Conference*, Lilongwe, 2015.
- [126] Michael Simmons and Hongmei Chi. Designing and implementing cloud-based digital forensics hands-on labs. In *Proceedings of the 2012 Information Security Curriculum Development Conference*, InfoSecCD '12, pages 69–74, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1538-8. doi: 10.1145/2390317.2390329. URL <http://doi.acm.org/10.1145/2390317.2390329>.
- [127] K. K. Sindhu and Dr. B. B. Meshram. Digital Forensic Investigation Tools and Procedures, 2012. URL <http://www.mecs-press.org/ijcnis/ijcnis-v4-n4/IJCNIS-V4N4-5.pdf>.
- [128] Steven S. Skiena. *The Algorithm Design Manual*. Springer-Verlag, London, 2nd edition, 2008. ISBN 978-1-84800-069-8.
- [129] Jill Slay, Yi-Chi Lin, Benjamin Turnbull, Jason Beckett, and Paul Lin. Towards a formalization of digital forensics. In Gilbert Peterson and Sujeet Shenoi, editors, *Advances in Digital Forensics V*, volume 306 of *IFIP Advances in Information and Communication Technology*, pages 37–47. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-04154-9. doi: 10.1007/978-3-642-04155-6_3. URL http://dx.doi.org/10.1007/978-3-642-04155-6_3.
- [130] C L Sowmya, C D Guruprakash, and M Siddappa. Visualization of network traffic. *International Journal of Smart Sensors and Ad Hoc Network*, 2(3):19–22, 2012. URL http://www.interscience.in/IJSSAN_Vol2Iss3,4/19-22.pdf.
- [131] Internet World Stats. World internet usage statistics: News and world population stats. In Internet World Stats. March 2011. URL <http://internetworldstats.com/stats.htm>.
- [132] Salvatore J Stolfo, Wei Fan, Wenke Lee, Andreas Prodromidis, and Philip K Chan. Cost-based Modeling for Fraud and Intrusion Detection : Results from the JAM Project.

- [133] Johannes Stüttgen and Michael Cohen. Anti-forensic resilient memory acquisition. *Digital Investigation*, 10:S105–S115, 2013. URL <http://www.sciencedirect.com/science/article/pii/S1742287613000583>.
- [134] Syllabus. *Daubert v. Merrell Dow Pharmaceuticals, Inc.* :: 509 U.S. 579 (1993) :: Justia U.S. Supreme Court Center, 1993. URL <https://supreme.justia.com/cases/federal/us/509/579/case.html>.
- [135] Jakub Szefer, Eric Keller, Ruby B. Lee, and Jennifer Rexford. Eliminating the hypervisor attack surface for a more secure cloud. In *Proceedings of the 18th ACM conference on Computer and communications security - CCS '11*, page 401, New York, New York, USA, October 2011. ACM Press. ISBN 9781450309486. doi: 10.1145/2046707.2046754. URL <http://dl.acm.org/citation.cfm?id=2046707.2046754>.
- [136] Mark Taber. *Maximum Security : A Hacker ' s Guide to Protecting Your Internet Site and Network*, 1997. URL <https://docs.com/93LK>.
- [137] M Taylor, J Haggerty, D Gresty, and R Hegarty. Digital evidence in cloud computing systems. *Computer Law & Security Review*, 26(3):304–308, 2010. ISSN 0267-3649. doi: 10.1016/j.clsr.2010.03.002. URL <http://dx.doi.org/10.1016/j.clsr.2010.03.002>.
- [138] Mark Taylor, John Haggerty, David Gresty, and David Lamb. Forensic investigation of cloud computing systems. *Network Security*, 2011(3):4–10, March 2011. ISSN 1353-4858. doi: 10.1016/S1353-4858(11)70024-1. URL [http://dx.doi.org/10.1016/S1353-4858\(11\)70024-1](http://dx.doi.org/10.1016/S1353-4858(11)70024-1).
- [139] Sean Thorpe, Tyrone Grandison, and Indrajit Ray. Cloud computing log evidence forensic examination analysis, 2011. URL <http://www.proficiencylabs.com/uploads/1/8/8/1/18817082/cyberforensics2012.pdf>.
- [140] Sean Thorpe, Tyrone Grandison, Arnett Campbell, Janet Williams, Khalilah Burrell, and Indrajit Ray. Towards a Forensic-Based Service Oriented Architecture Framework for Auditing of Cloud Logs. In *2013 IEEE Ninth World Congress on Services*, pages 75–83. IEEE, June 2013. ISBN 978-0-7695-5024-4. doi: 10.1109/SERVICES.2013.76. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6655678>.
- [141] Yi-hsiung Ting and Chung-huang Yang. *Design and Implementation of a Cloud Digital Forensic Laboratory*, 2013.
- [142] Tigran Tsaturyan. *Capturing Network Traffic into Database*, 2013. URL <http://www.slideshare.net/tigrantsaturyan58/network-packetstodatabase-v1>.

- [143] Fu-Kuo Tseng, Rong-Jaye Chen, and B.-S.P. Lin. ipeks: Fast and secure cloud data retrieval from the public-key encryption with keyword search. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on*, pages 452–458, July 2013. doi: 10.1109/TrustCom.2013.57.
- [144] Louis Turnbull and Jordan Shropshire. Breakpoints: An analysis of potential hypervisor attack vectors. In *2013 Proceedings of IEEE Southeastcon*, pages 1–6. IEEE, April 2013. ISBN 978-1-4799-0053-4. doi: 10.1109/SECON.2013.6567516. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6567516>.
- [145] Gene Tyler. Cloud Computing : Silver Lining or Storm Ahead? *IATAC*, 13(2), 2010.
- [146] John R Vacca. *Computer and Information Security Handbook (The Morgan Kaufmann Series in Computer Security)*. Morgan Kaufmann, 2009. ISBN 0123743540. URL <http://www.amazon.com/Computer-Information-Security-Handbook-Kaufmann/dp/0123743540>.
- [147] Aleksandar Valjarevic and Hein S. Venter. Harmonised digital forensic investigation process model. In *2012 Information Security for South Africa*, pages 1–10. IEEE, August 2012. ISBN 978-1-4673-2159-4. doi: 10.1109/ISSA.2012.6320441. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6320441>.
- [148] R.B. van Baar, H.M.A. van Beek, and E.J. van Eijk. Digital Forensics as a Service: A game changer. *Digital Investigation*, 11:S54–S62, May 2014. ISSN 17422876. doi: 10.1016/j.diin.2014.03.007. URL <http://www.sciencedirect.com/science/article/pii/S1742287614000127>.
- [149] Irene Vanderfeesten, Hajo A. Reijers, and Wil M. P. van der Aalst. Evaluating workflow process designs using cohesion and coupling metrics. *Comput. Ind.*, 59(5):420–437, May 2008. ISSN 0166-3615. doi: 10.1016/j.compind.2007.12.007. URL <http://dx.doi.org/10.1016/j.compind.2007.12.007>.
- [150] Volatility. VolatilityIntroduction - volatility - Introduction to Volatility - An advanced memory forensics framework, 2014. URL <https://code.google.com/p/volatility/wiki/VolatilityIntroduction>.
- [151] David Watson and Andrew Jones. *Digital Forensics Processing and Procedures: Meeting the Requirements of ISO 17020, ISO 17025, ISO 27001 and Best Practice Requirements [Paperback]*. Syngress; 1 edition, 2013. ISBN 1597497428. URL <http://www.amazon.com/Digital-Forensics-Processing-Procedures-Requirements/dp/1597497428>.

- [152] Yuanfeng Wen, Xiaoxi Man, Khoa Le, and Weidong Shi. Forensics-as-a-Service (FaaS): Computer Forensic Workflow Management and Processing Using Cloud. (c):208–214, 2013.
- [153] Janet Williams. Good Practice Guide for Computer-Based Electronic Evidence, 2012. URL <http://www.acpo.police.uk/documents/crime/2011/201110-cba-digital-evidence-v5.pdf>.
- [154] WireShark. WireShark: The world’s foremost network protocol analyzer, 2015. URL <http://www.wireshark.org/>.
- [155] Daniela Zaharie. Lecture 3: Verification of algorithms correctness, 2015. URL <http://web.info.uvt.ro/~dzaharie/algeng/lecture3.pdf>.
- [156] Shams Zawoad and Ragib Hasan. Cloud Forensics: A Meta-Study of Challenges, Approaches, and Open Problems. February 2013. URL <http://arxiv.org/abs/1302.6312>.
- [157] Gang Zeng. Research on Digital Forensics Based on Private Cloud Computing. 2(9):24–29, 2014.
- [158] Yinqian Zhang, Ari Juels, Michael K. Reiter, and Thomas Ristenpart. Cross-VM side channels and their use to extract private keys. *Proceedings of the 2012 ACM conference on Computer and communications security - CCS '12*, page 305, 2012. doi: 10.1145/2382196.2382230. URL <http://dl.acm.org/citation.cfm?doid=2382196.2382230>.
- [159] Liping Zheng and Hong Yi. A management scheme of user private keys in an IBE system. In *2012 IEEE Fifth International Conference on Advanced Computational Intelligence (ICACI)*, pages 1078–1080. IEEE, October 2012. ISBN 978-1-4673-1744-3. doi: 10.1109/ICACI.2012.6463338. URL <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6463338>.
- [160] Scott Zimmerman and Dominick Glavach. Cyber Forensics in the Cloud. *IAnewsletter*, 14(1):4–7, 2011.