

FReadyPass: Creating Digital Passports to Track the Location of Data in the Cloud

Submitted in fulfillment of the requirements of the degree
Magister Scientiae Computer Science

in the faculty of
Engineering, the Built Environment and Information Technology
of the
University of Pretoria

by
Philip Trenwith

Supervised by
Prof H.S. Venter
Department of Computer Science

Tuesday 20th February, 2018

Abstract

The cloud has exacerbated the difficulty of digital forensic investigations because data may be spread over an ever-changing set of hosts and data centers. The normal search and seizure approach digital forensic investigators tend to follow does not scale well in the cloud because it is difficult to identify the physical device's data resides on. The location of these devices is often unknown or unreachable. A solution to identifying the physical devices can be found in data provenance. Similar to the tags included in an email header indicating where the email originated from, a tag added to data as it is accessed both in and out of the cloud identifies where the data has been. If such a trace can be provided for data in the cloud, it may ease the investigating process by indicating where the data can be found to be able to continue with the investigation. In this research, the author proposes the development of a Digital Passport that aims to provide a detailed history of where user data came from and where it has been since it came under the control of a cloud service provider. The Digital Passport further provides the user with access control, allowing the user to choose the location from where the digital passport may be accessed. The ability to control access to the digital passport regarding its location holds many advantages for cloud service providers, users and digital forensic investigators alike. The digital passport provides digital forensic investigators with a clear audit log of where the data has been. The digital passport furthermore holds the advantage that digital forensic investigating teams are not limited by jurisdictional issues that grant them only limited access to the data they need. Because

of location-based access control, it is possible to ensure the data does not move into a jurisdiction that is outside the control of the cloud service provider.

Keywords. Digital Passports, Data Provenance, Access Control, Jurisdiction, Digital Forensics

Acknowledgement

I wish to thank the following people who provided me with assistance during my studies:

- My parents for providing me with this great opportunity to further my education and for the sacrifices that they had to endure during this time.
- My friends that provided support and encouragement during the time of my studies.
- My supervisor for his guidance and support that he imparted throughout my studies.
- The University of Pretoria, the National Research Foundation (NRF) and my employer GEW Technologies that provided financial assistance during my studies.

Table of Contents

Abstract	i
Acknowledgement	iii
Part I Introduction and Problem Statement	1
1 Introduction	2
1.1 Introduction	2
1.2 Problem Statement	3
1.3 Methodology	5
1.4 Objectives	6
1.5 Layout	7
Part II Background Literature	11
2 Background on Digital Forensics and Data Provenance in the Cloud	12
2.1 Introduction	12
2.2 Digital Forensics	12
2.3 The Digital Forensic Investigation Process	14
2.3.1 Identification	14
2.3.2 Collection	14
2.3.3 Transportation	15
2.3.4 Storage	15
2.3.5 Examination	15

2.3.6	Presentation	17
2.3.7	Destruction	17
2.4	Digital Forensic Readiness	18
2.5	Cloud Computing	21
2.6	Cloud Forensics	22
2.7	Data Provenance	25
2.7.1	Embedded storage technique	26
2.7.2	Separate provenance storage technique	27
2.8	Conclusion	28
3	Information Security	29
3.1	Introduction	29
3.2	Protection of sensitive data and information in the cloud	30
3.2.1	Data Protection: European Union	30
3.2.2	Data Protection: United States of America	31
3.2.3	Data Protection: South Africa	31
3.3	Determining the location of a user accessing the cloud . .	32
3.4	Dangers of IP address spoofing and tunneling services . .	34
3.5	Guarding against IP address spoofing and tunneling services	37
3.6	IP address routing	38
3.7	Cryptography and verification of data integrity	40
3.8	Asymmetric Key Exchange Algorithm	41
3.9	Access Control	45
3.9.1	Password based access control	45
3.9.2	Biometric based access control	46
3.9.3	Physical device based access control	46
3.9.4	Location-based access control	46
3.9.5	Multiple techniques based access control	47
3.10	Conclusion	48

Part III A Model for FReadyPass: Creating Digital Passports to Track the Location of Data in the Cloud	49
4 Location-based access control	50
4.1 Introduction	50
4.2 Access Control Rules	51
4.3 Guarding against IP address spoofing and tunneling services	54
4.4 Conclusion	58
5 Architectural requirements for FReadyPass	59
5.1 Introduction	59
5.2 Functional Requirements	60
5.2.1 Single passport single file	60
5.2.2 Encapsulate any data file	61
5.2.3 Standardized file structure	61
5.2.4 Standard protocols	61
5.3 Non-Functional Requirements	62
5.3.1 Minimum overhead	62
5.3.2 Scalability	62
5.3.3 Reliability	62
5.3.4 Availability	63
5.3.5 Security Requirements	63
5.4 Conclusion	69
6 Architectural Design of FReadyPass	70
6.1 Introduction	70
6.2 Designing a FReadyPass	70
6.2.1 Part 1: Identifier	71
6.2.2 Part-2: Historic Provenance Data	72
6.2.3 Part-3: Payload	72
6.3 Overview of constructing a FReadyPass	73
6.4 FReadyPass file structure	77
6.5 Conclusion	81

7	Provenance Data	82
7.1	Introduction	82
7.2	Capturing Provenance Data	82
7.2.1	Meeting R1. A provenance record needs to be unforgeable.	83
7.2.2	Meeting R2. A provenance record needs to be kept confidential.	83
7.2.3	Meeting R3. The integrity of a provenance record should be maintained by the system.	84
7.2.4	Meeting R4. A provenance record should show who is responsible for a modification to the payload.	85
7.2.5	Meeting R5. A provenance record should show what was modified.	86
7.2.6	Meeting R6. A provenance record should show the time of a modification.	86
7.2.7	Meeting R7. A provenance record should show the FReadyPasses location in the cloud at the time of a modification.	87
7.3	Provenance Record Design	87
7.4	Maintaining cryptographic keys for verifying provenance data	92
7.5	Verify the integrity of provenance data	95
7.5.1	Verify the integrity of a subset of provenance records	96
7.5.2	Verifying the integrity of the entire set of provenance records	103
7.6	Conclusion	104
8	Accessing a FReadyPass in and out of the cloud	106
8.1	Introduction	106
8.2	Securing a FReadyPass at rest and in transit	106
8.3	Client Application	107
8.4	Maintaining RSA Keys for signature and verification	108
8.5	FReadyPass life-cycle	109

8.5.1	Downloading a FReadyPass	110
8.5.2	Decrypting a FReadyPass	112
8.6	Optimal Cloud Service Provider Network	113
8.7	Conclusion	116
 Part IV Prototype		117
 9 Prototype		118
9.1	Introduction	118
9.2	Prototype Server Setup	119
9.3	Location Based Access Control Mechanism	119
9.4	Development of the client application	121
9.5	Downloading a FReadyPass	123
9.6	Uploading a FReadyPass	127
9.7	Verify the integrity of provenance records	130
9.8	FReadyPass provenance viewing application	131
9.9	Conclusion	132
 10 Related Work		134
10.1	Introduction	134
10.2	Related work on the application of data provenance systems in the cloud	134
10.2.1	PASSv2	134
10.2.2	Xen Hypervisor modified for PASSv2	136
10.2.3	Windows log files stored on a central log server	137
10.2.4	Network Infrastructure used to determine location	137
10.2.5	DataPROVE, TrustCloud and Flogger	138
10.2.6	CloudDT	139
10.2.7	Blockchain Technology	142
10.3	Proposed solution compared to other solutions	145
10.3.1	FReadyPass system vs. CloudDT	146
10.3.2	FReadyPass system vs. ProvChain	148
10.4	Conclusion	149

11 Critical Evaluation	150
11.1 Introduction	150
11.2 Detailed review of the proposed model and prototype . .	150
11.3 Value added to community	156
11.4 Conclusion	158
Part V Summary	159
12 Conclusion	160
12.1 Introduction	160
12.2 Review the research question	161
12.3 Research Contribution	162
12.4 Future Work	162
12.5 Conclusion	164
Part VI Bibliography and Appendices	165
Bibliography	166
Appendix A Client Code	179
Appendix B Provenance Data	182

Part I

Introduction and Problem Statement

Chapter 1

Introduction

1.1 Introduction

Ciborra [1987] states, the traditional role of computers is to provide support for the human decision-making process. Computers were developed during the Second World War to help solve mathematical problems. As time passed and technology progressed, the internet was developed. The internet is the product of research funded by the US Department of Defence in the early 1960's[Mowery and Simcoe, 2002]. The goal of the research was to design a computer network for the United States military to communicate with their military bases. In the early days of computing the notion of "the cloud" did not exist. Over the years, as technology progressed, the cloud was born. The primary use of a computer changed from that of storage and processing device to communication and information gathering device[Johnson, 1996]. With access to the cloud, users of computing devices are no longer limited to use a specific device for everyday tasks like reading emails or updating documents. Even replacing one's mobile device has become a trivial task with cloud services such as Google's Sync service used by many users to synchronize calendar entries and contacts to the cloud. Verio [2012] discusses more benefits of the cloud. However, the cloud does have its disadvantages.

In traditional computing systems, a computer was used as storage and processing device. If suspected that the computer was involved in illegal activities, the computer could be seized for investigation [Trenwith and Venter, 2013]. Such an investigation is referred to as a digital forensic investigation (DFI). A DFI is undertaken whenever there is a suspicion of a severe crime requiring further investigation into digital space. Search, and seizure is the typical approach followed during a DFI. However, this approach does not scale well in the cloud because access to physical devices in the cloud is often restricted and the location of these devices unknown. Gartner [2008] states that investigating cloud services are especially difficult for several reasons, among others, the fact that data may be spread over an ever-changing set of hosts and data centers, leaving one with little control over one's data [Gartner, 2008]. In this work, DFIs conducted in the cloud is referred to as cloud forensics. Reilly et al. [2010] refers the biggest challenge regarding cloud forensics as the inability to gain physical access to devices in the cloud. The aforementioned makes the collection of data for an investigation a difficult task to perform. However, collecting data in the cloud for investigative purposes is not the only problem that needs to be addressed. Barbara [2009] discuss more of the challenges that the cloud presents for DFIs. In the remainder of this chapter, the author looks at some of these problems in more detail, determine which methodologies to use to best address these issues and then establish objectives that he aims to accomplish to address the issues identified.

1.2 Problem Statement

With the introduction of the cloud into the computing world, conducting a DFI became increasingly difficult to do. Especially regarding the challenges, it presents for digital forensic examiners [Gartner, 2008] [Barbara, 2009]. The digital domain is often used to orchestrate and commit crimes, both in the digital and in the physical world. If the suspicion of

a crime exists, an investigation is required to determine the facts. When the outcome of such an investigation leads to criminal proceedings, the evidence gathered during the investigation needs to be presented in a court of law. The chain of custody needs to be maintained to prove the evidence presented is original and has not been altered or forged. The U.S. National Institute of Justice(NIJ) defines chain of custody as “a process used to maintain and document the chronological history of the evidence.”[Giova, 2011] data provenance should be used as a source of the chain of custody for digital evidence. Muniswamy-Reddy and Seltzer [2010] defines data provenance as the history of a digital object. A digital object can be defined as sequence of bits/bytes[Cross, 2014] This include any software element that forms part of a computer system, which can include anything from a website on the internet to video or audio streams or files stored on a computer system. For the purpose of this work, the author defines a digital object as any flat file irrespective of its location or purpose in a computer system. The history of a digital object refers to metadata information such as when an object was accessed, where it was accessed and who accessed it. Data provenance is not always available to cloud forensic investigators because not all cloud service providers capture or store data provenance. Another problem that investigators are faced with is accessing data in another jurisdiction. If a Cloud Service Provider(CSP) within another jurisdiction is not willing to give investigators access to possible sources of evidence, the investigators may be able to legally compel the CSP to grant access by obtaining a court order referred to as a warrant. However, such a warrant can only be obtained from the legal entity governing the jurisdiction in which the CSP is located. If the jurisdiction of the investigating team is different from that of the CSP’s jurisdiction, and the legal entities governing these jurisdictions are different, the investigating team may not be able to obtain a warrant, due to such jurisdictional, political and legal reasons.

In physical space when a person travels from one country to another, that person requires a passport from his/her country allowing him/her

to travel outside the borders of that country. In most cases, a Visa is required from the country the person intends to visit, for him/her to enter that country. This model provides sovereign nations with the ability to choose who is allowed to enter their country, as well as prohibit some of their citizens from leaving the country if they are deemed unfit. The problem that this dissertation addresses is that, currently no such system exists to govern digital space and communications. Such a system for digital space would provide many advantages to investigating teams, and users alike. Therefore, the research question this dissertation will address is stated as follows: “How can the implementation of a digital passport and visa system address the cross-jurisdictional issues faced by digital forensic investigation teams performing investigations in the cloud?”

1.3 Methodology

To address the research question that is asked in this work, the author performs a literature survey to find similar systems available for monitoring the flow of information across jurisdictional boundaries. The author identifies the advantages and disadvantages of these systems and the areas of interest that are not addressed. The author identifies requirements for a new system. After that, a model is designed that implements the requirements to address the research question. The model aims to address the issues identified by first providing a theoretical basis for implementing a prototype. The final methodology in this research is, therefore, the implementation of a proof-of-concept prototype. The prototype translates the theoretical model into a practical working system in the digital domain.

Based on the problems identified in the literature survey the author forms objectives that the proposed model should achieve. There are primary and secondary objectives. These objectives are discussed further in the next section.

1.4 Objectives

Three objectives are identified. First, capture provenance data and store it in a digital forensic ready manner, second, how to use physical location during access control, third, implement the first two objectives in current cloud infrastructure.

The first two objectives have a general goal and aim to provide a solution to a problem in theory. This theory can then be implemented in any practical system based on the workings of that system. The third objective follows a practical approach, taking into account the specific layout of the cloud at present.

1.4.1 Objective One: Digital Object History

During a cloud forensic investigation, the history of digital objects may be required to serve as the chain of custody for digital evidence. The author investigates how the history of a digital object can be obtained from cloud computing environments and stored in a digital forensic ready manner. Digital Forensic Readiness(DFR) is a concept whereby an organization captures information that may be useful in an investigation, ahead of any such investigation, to be prepared for the investigation ahead of time. DFR is discussed in more detail later in this dissertation.

1.4.2 Objective Two: Location Based Access Control

Some organizations may require the ability to limit access to data based on the physical location of the requester. The author investigates the implementation of an access control mechanism to restrict access to data based on the boundary of a physical jurisdiction and the physical location

of the requester.

1.4.3 Objective Three: Practical Implementation

The author investigates the implementation of a system to capture data provenance and provide location-based access control regarding current cloud infrastructure. This objective serves to provide a practical implementation that can be used by existing CSPs with minimal modification to their current system.

Having stated the problem statement, the methodologies that will be used to address the problem statement and the objectives the author aims to achieve. The layout of the remainder of this dissertation is discussed in the next section.

1.5 Layout

The rest of this dissertation is structured in parts as follows:

Part one is the introduction. Part two contains the digital forensics and cloud computing background that serves as the foundation for the proposed model. Chapter 2 and 3 forms part of Part two. Part three encapsulates the design of the model. Including the access control mechanism, the architectural requirements and design of the digital passport and the data provenance and access to digital passports. Chapters 4 through to 8 forms part three. Part four contains the Prototype and Related Work chapters. This part includes a critical evaluation chapter and the concluding chapter that includes future work to be done.

In Chapter 2 the author investigates the domains of Digital Forensics and Cloud Computing. Understanding the domains of digital forensics and cloud computing provides essential information, necessary when de-

signing a system that interacts with the cloud. The author further investigates digital forensic readiness and data provenance. Considering the objectives of the system proposed in this work requires the capturing of provenance data. It is crucial that the process of capturing provenance data adhere to digital forensic standards, should the data be needed for the proceedings of a digital forensic investigation.

Having looked at the building blocks of cloud computing and digital forensics, the author investigates cloud forensics and its challenges. These challenges form the base of the requirements of this dissertation. From the investigation of known challenges, it is possible to determine the requirements for successfully tracking digital objects through the cloud while adhering to digital forensic standards.

In Chapter 3 the author investigates information security. The author investigates the laws and legislation put in place by various countries regarding the protection of sensitive information. The author further addresses the requirements needed to develop a location-based access control mechanism and the challenges associated with such a mechanism. The author also discusses cryptography and the verification of data integrity in this chapter.

In Chapter 4 location-based access control is discussed. The author investigates how a user's IP address can be used to determine the user's location. Having determined how the IP address can be used, the author proposes a technique that will be used in this work to determine the user's location as well as use the location as part of an access control mechanism to accomplish geo-fencing for cloud data. Geo-fencing is a feature in a software program that uses the global positioning system (GPS) or radio frequency identification (RFID) to define geographical boundaries[TechTarget.com, 2017b]. As discussed in the objectives section of this chapter one of the goals of this work is to provide geographical boundaries for cloud data in an attempt to eliminate jurisdictional issues

during a digital forensic investigation.

In Chapter 5 the functional and non-functional requirements of the model is identified and summarized. It is necessary to have clear, measurable requirements to develop the model accordingly and to be able to determine the success or failure of the model.

The requirements of this model have been discussed, and the information security concerns were taken into account. In Chapter 6 the author discusses the design of the digital passport. The design takes all of the requirements into consideration and shows how the provenance data is stored with the digital passport.

Chapter 7 discusses the process of capturing provenance data as well as the requirements for provenance data and how these requirements are addressed. The author further discusses how the provenance data is stored with the digital passport and how the integrity of the provenance data is maintained and verified.

In Chapter 8 the author takes a more detailed look at how the digital passport is accessed in and out of the cloud. The author discusses the processes related to the handling of a digital passport as well as how the passport is secured. The author looks at the layout of a CSP's internal network and how the proposed model fits into that network.

In Chapter 9 the proof of concept(POC) prototype is designed and discussed. The POC is the practical implementation of the proposed model. It is implemented into existing cloud computing infrastructure. The POC serves to prove that the model works and how it adds value to the community.

In Chapter 10 the author conducts a literature survey on related work. The goal of this survey is to identify similar research, as well as

alternate solutions possibly addressing the same or similar issues. One of the critical goals of the literature survey is to investigate if gaps exist in existing solutions that have not been addressed. The proposed solution is compared to existing solutions related to this work, and the value of the author's contribution to the scientific community is briefly discussed. The author explains to what degree the proposed solution improves on existing solutions.

Chapter 11 is a critical evaluation of the work that was done. In this chapter, the author evaluates his work, specifically looking at specific design decisions that were made during the work and the reasoning behind these decisions. The author then evaluates the value that this work adds to the domain.

Chapter 12 concludes the dissertation. The author revisits the research question proposed; indicating to what extent the research question has been addressed. This chapter concludes with a suggestion on future work to be done.

Part II

Background Literature

Chapter 2

Background on Digital Forensics and Data Provenance in the Cloud

2.1 Introduction

This chapter provides information regarding digital forensics, cloud computing, and data provenance. The information in this chapter provides the necessary background to understand the foundation on which the work proposed in this dissertation is modeled. In the next section, the background of digital forensics is examined.

2.2 Digital Forensics

Digital Forensics is the scientific approach followed to investigate and potentially solve crimes committed using computing devices. Many of the techniques used by digital forensic investigators were initially developed with a different goal, as data recovery techniques [Garfinkel, 2010]. These include techniques such as live analysis, file carving and the recovery of deleted files. Initially, computer professionals worked with law enforcement on an ad-hoc basis performing digital forensic investiga-

tions[Garfinkel, 2010]. The police forces in the United Kingdom has a guideline document for investigators which specifies the steps and procedures that officers should take when dealing with situations associated with digital evidence[Wilkinson and Haagman, 2010]. This guideline aims to uphold the chain of custody. The chain of custody is important in an investigation because normally it is one of the main areas targeted by a defense team to break down the credibility of evidence in a court of law[Giannelli, 1982]. The years 1999 to 2007 defines the golden age for digital forensics. It is during this time that much of the research leading to the development of digital forensics took place and universities started offering courses in digital forensics. The development of digital forensic tools was eased by the failure of the market to adopt encryption techniques for data at rest.

A definition of Digital Forensic Science as defined by Palmer [2001] is: “the use of scientifically derived and proven methods toward the preservation, collection, validation, identification, analysis, interpretation, documentation and presentation of digital evidence derived from digital sources for the purpose of facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations.” From this definition, we gather that the identification and collection of possible evidence is the key step in any investigation. Investigators can only collect what is identified, and to successfully collect data, the investigators need to be able to determine the location of data.

In the next section of this chapter, the author discusses the traditional digital forensic investigation process, briefly discussing each phase in detail.

2.3 The Digital Forensic Investigation Process

Stephenson [2003] defines the digital forensic investigation process having six phases; they are identification, collection, transportation, storage, examination, and presentation. In this work, a destruction phase is added as defined by Cohen [2011, p.44-51]. Each of these phases is discussed next.

2.3.1 Identification

In the normal course of a digital forensic investigation if a computing device is suspected to be involved in a crime, the traditional approach, after identifying and locating the suspected device, warrants the investigating team to seize the device for imaging and analysis. Imaging can produce an enormous amount of data available for analysis by the investigators. Potential evidence needs to be identified for the investigators to build a case. Potential evidence does not only include data on a specific device, in a network environment potential evidence may include data on all connected network devices. The identification process is critical because the collection of evidence is driven by what is identified in this phase. If data cannot be identified as potentially relevant evidence, it may never be collected or processed as evidence.

2.3.2 Collection

For evidence to be admissible in a court of law, it needs to be collected in a manner that does not dilute the integrity of the data. During the collection of data, the chain of evidence must be upheld. During the collection phase, investigators have to adhere to the proportionality rule, which states that, only evidence deemed to be legal and useful in building a case should be collected for analysis. It further states that only with good cause can computerized data be brought into discovery[Moss, 2009].

The collection of evidence is normally done by preserving a copy of the original evidence. A copy is made so that the actual media does not need to be preserved. The common practice in digital forensic investigations is to make an image of a device. The storage media in use today are often very large and the process of imaging such big devices can take a long time. This process can be disruptive to normal business. It is for this reason that digital forensic readiness plays a vital role if an investigation is required in the cloud.

2.3.3 Transportation

Digital evidence is transported by making an exact copy of the data at the bit level. Transportation of digital evidence can be done either by transporting the physical device to the required destination or by transmitting only the digital data to the required destination. It is important during this phase to maintain the chain of custody and the integrity of evidence. Evidence has to be properly packaged during transportation to avoid spoliation, a condition caused by bad weather that can degrade evidence if it is not properly transported.

2.3.4 Storage

Digital evidence must be maintained properly during an investigation. The storage area must be secured to maintain a proper chain of custody. Many things can go wrong during storage such as natural environmental conditions such as flood or fire, decay over time, the loss of power or decay of media preserving mechanisms[Cohen, 2011, p.44-51].

2.3.5 Examination

The examination of evidence is based on the principle of exchange defined by Edmond Locard stating that when two objects come in contact

with one another, a trace of one object is left behind on the other object [Horswell and Fowler, 2004, p.45]. This principle holds true in the digital world as well, although it is a little bit different, a trace in the digital world is the change in bits that indicates the change in state when the digital object came in contact with the device. The examination phase involves processes that the investigators use to find and in some cases reconstruct these traces. The examination phase consists of four phases known as Analysis, Interpretation, Attribution, and Reconstruction. These four phases are briefly discussed next.

2.3.5.1 Analysis

The goal and focus of the analysis phase are to find sequences of events that could have produced the traces of evidence. Only a finite number of possibilities can produce a trace in the digital world. However, this finite number can still be extremely large.

2.3.5.2 Interpretation

The goal of interpretation is to try to determine what caused the traces that were found. Cohen [2011, p.44-51] list some conditions that can cause the presence of traces that is not valid. These include a trace produced by a different application that looks similar, a trace that was produced by a failed transaction, a Trojan program, or the trace may have been produced and placed in the system maliciously. Investigators should consider these possibilities when performing analysis to determine what caused a trace. The possibility of malicious activities should be considered when making assertions regarding events that took place. The assertions made might carry more weight if it is possible, with the evidence available to explore alternative possibilities and show that they are unlikely or not possible given the evidence.

2.3.5.3 Attribution

The attribution phase looks at factors that could attribute to the evidence and provide some clarity. Including what is known as, anchor events. An anchor event is something that can link digital space to physical space. Consider an example, evidence suggest that a crime was committed with computer A. The owner of computer A cannot be charged with the crime unless it can be proven with an anchor event and non-repudiation that the owner was present at his/her computer at the time the crime was committed and that the owner is, in fact, responsible for committing the crime.

2.3.5.4 Reconstruction

In some cases, when a little more certainty is required to make reasonable assumptions regarding the traces of evidence, it is required to reconstruct the events that took place to produce the events. Through the reconstruction phase, the level of certainty can be reasonably established. One of the conditions under which reconstruction could be necessary is if it becomes known that some evidence is of importance, but the evidence is no longer available.

2.3.6 Presentation

The presentation of evidence is not discussed to great length in literature, but it is a vital step. If the evidence and the investigation that led to the discovery cannot be presented and explained to a jury or a court of law in such a way that they understand the case, they cannot make a reasonable decision regarding the guilt or innocence of the accused.

2.3.7 Destruction

In some occasions, there exists evidence that is of a confidential nature; this may be due to military classification, copyrights, trade secrets or

other cases. When this is true, the evidence should be destroyed at the conclusion of an investigation. The destruction of evidence should be conducted in a way that does not break the chain of custody or violate the availability of classified information. The digital forensic process as described here follows a rather reactive approach. There exists a proactive approach to digital forensics as well, known as Digital Forensic Readiness(DFR). In the next section, the author discusses Digital Forensic Readiness.

2.4 Digital Forensic Readiness

Tan [2001] defines DFR as the ability of an organization to maximize its potential to use digital evidence while minimizing the cost of an investigation. Data that may be required can actively be collected by a CSP and stored for a predetermined period in preparation for a possible investigation. Rowlingson [2004] defines a ten-step process to help organizations prepare for possible investigations by capturing potential evidence. This ten-step process addresses some of the areas of interest to the author.

Step 1 - Define the business scenarios that require digital evidence. For an organization to determine, which of their business scenarios may require digital evidence the organization has to implement a risk assessment. This assessment determines the impact that various risks may have on the organization. From this assessment, the organization can set up protocols to capture potential evidence in a digital forensic ready manner, thus reducing the time required to perform an investigation. If this is done correctly, the risk can be mitigated or reduced.

Step 2 - Identify available sources and different types of potential evidence. After conducting the risk assessment, it is required to

identify all of the available sources producing digital data that can serve as potential evidence in an investigation.

Step 3 - Determine the evidence collection requirement. Taking the various risks into consideration, an organization should decide which of the available sources should be used to collect data to reduce the risks identified. A cost-benefit analysis identifying the cost associated with collecting evidence from certain sources greatly aids this process.

Step 4 - Establish the capability for securely gathering legally admissible evidence to meet the requirement. Consider the requirement as the mitigation and reduction of identified risks. Collected evidence has to be stored and maintained as authentic records. An organization has to collect potential evidence in such a manner that it adheres to digital forensic standards and requirements. Digital evidence is only accepted into a court of law with good cause, and as such, the data collected should be relevant to the requirement.

Step 5 - Establish a policy for secure storage and handling of potential evidence. The focus of this step is the long-term storage and retrieval of collected evidence. Data integrity has to be maintained during storage. The policy for storing collected data should contain security measures to ensure the authenticity of evidence, including methods to show that the integrity of evidence is maintained when analyzed.

Step 6 - Ensure monitoring is targeted to detect and deter major incidents. Some attacks may have serious implications for an organization. Intrusion Detection Systems should be set up to monitor data sources, to detect and deter possible attacks.

Step 7 - Specify circumstances when escalation to a full formal investigation should be launched. Suspicious events should

be investigated; some events could be escalated if serious enough. Circumstances requiring escalation are events that pose a serious threat to an organization. A policy should be set up for the organization to indicate when an event should be classified as a serious incident that requires escalation.

Step 8 - Train staff in incident awareness, so that all those involved understand the role they have in the digital evidence process and the legal sensitivities of evidence. This step prepares staff for the responsibilities that may fall on them should an event occur that requires evidence collection or monitoring of sources.

Step 9 - Document an evidence-based case describing the incident and its impact. The aim of an investigation is to find answers to relevant questions. Answering questions such as who, what, where, how and why. The goal of this step is to establish a policy for handling an evidence-based case. The case file compiled should also indicate how evidence should be presented for non-technical people to understand it.

Step 10 - Ensure legal review to facilitate action in response to the incident. Legal advice is required at some point in the investigation to determine if the evidence collected is strong enough to be used in a court of law. If not, further investigation needs to be undertaken in the attempt to capture a culprit. When a formal action is taken after an event, the action taken needs to be justified, and often the decisions made needs to withstand scrutiny in a court of law. Any legal advisers consulted, needs to have the necessary training and experience in Cyber laws and other aspects of the case at hand.

Digital Forensic Readiness allows an organization to prepare for possible investigations in advance. There are many threats that if realized justify the undertaking of a digital forensic investigation. In this work, the author is interested in investigating the possibility of tracking data

through the cloud to identify where the data is and where it has been. Tracking the location of data in the cloud aids digital forensic readiness. Before the author can discuss the tracking of data through the cloud, it is required to determine what the cloud is and how it works.

2.5 Cloud Computing

Cloud computing is defined by Heiser [2009] as: “a style of computing where massively scalable IT-enabled capabilities are delivered ‘as a service’ to external customers using Internet technologies”. The National Institute of Standards and Technology(NIST) defines cloud computing as: “a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g. networks, servers, storage applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction”. This work adopts the NIST definition of cloud computing.

The main goal of cloud computing is to be able to solve large-scale computation problems and maximize throughput by making use of distributed services. These services include Infrastructure as a Service(IaaS), Platform as a Service(PaaS), Software as a Service(SaaS) and Backend as a Service(BaaS).

IaaS offers hosting and storage service to cloud users. PaaS offers platform support for software development. In addition, SaaS offers fully functional applications to cloud users. BaaS offers cloud storage and services to Web and Mobile App Developers using Application Programming Interfaces. These cloud services can be deployed in one of four deployment models. They are; private cloud, public cloud, hybrid cloud and community cloud[Mell and Grance, 2011].

A private cloud belongs to an organization and is utilized by the

members of that organization. This type of cloud is not accessible to the public or any other organizations. A public cloud is available to any member of the public, and each user is billed for the service they utilize. This form of cloud is the least secure. A hybrid cloud is a combination of the first two types. It is owned by an organization but typically has access to resources in the public cloud. A community cloud is not publicly accessible, nor is it owned by a single organization. A community cloud is defined as “A multi-tenant infrastructure that is shared among several organizations from a specific group with common computing concerns”[Marinos and Briscoe, 2009].

Resources in the cloud are scalable and provisioned according to the requirements of a user or system. The scalable and virtualized nature of the cloud can be of great advantage to users but holds many challenges for digital forensic examiners. DFI in cloud computing environments is referred to as cloud forensics and is discussed in more detail next.

2.6 Cloud Forensics

In the early days of digital forensics, it was possible for an investigator to pull the plug on a machine that needed to be investigated, after which the device could be analyzed[Adelstein, 2006]. This approach does not scale well in cloud computing environments due to the virtualized nature of the cloud.

Cloud forensic investigations can in most cases not be performed independently without the cooperation of one or multiple CSPs. CSPs have full control over the computing environment and, thus, the sources of evidence that is of interest to investigating teams. In most cases, it is impossible for investigators to gain physical access to devices in the cloud, which makes the traditional investigating process impossible to perform[Reilly et al., 2010]. In the cloud, seizing a device for investi-

gation is not possible; the location of the device may be outside the jurisdiction of the investigating team, or simply in a location impossible to reach.

If data is required in a court of law, the investigators are required to retrieve the data from the cloud. However, the location of data in the cloud is often unknown due to the virtualized nature of the cloud. When investigating cloud devices, it is necessary to isolate the device from the rest of the cloud. Delpont et al. [2011] discuss the necessity of isolating a crime scene as well as how to isolate a virtual crime scene in the cloud. Isolating an instance under investigation protects the data from contamination. Delpont et al. [2011] strengthens the argument that devices in the cloud cannot be seized for investigation because multiple virtual cloud instances can exist on a single physical device. When isolating a specific instance for investigation, it is necessary to remove other virtual instances from the physical device instead of removing the instance under investigation from the device. Moving the instance under investigation might compromise the integrity of the data once the instance is restarted and recognizes new hardware. Isolating an instance for investigation has the advantage that once isolated; the instance can be investigated without disrupting the business of the CSP. However, a disadvantage is that isolating the instance is still a post-event driven action. Action is only taken after a malicious event has occurred.

Birk and Wegener [2011a] discusses a tool that allows investigators to take a snapshot of virtual machines. However, a snapshot is only useful in an investigation if the snapshot was made at the exact time of the event requiring the investigation. From the discussion so far, it is clear that the traditional post-event investigation model does not scale well to the cloud. Cloud forensics requires a different approach, a proactive approach known as Digital Forensic Readiness. CSPs have to capture log data and other potential evidence proactively and store this data for a period of time in preparation for a DFI.

Another challenge where log data is crucial is identified by Barbara [2009]. Barbara [2009] states that the biggest challenge investigators are faced with in the cloud is to determine who, what, when, where, how, and why of cloud-based criminal activity. Investigators require log data to answer these questions. Logs provide crucial information regarding access and modification to data in any computer system. In a cloud-computing environment, having log data collected in a central location, potentially decrease the complexity of a DFI during the acquisition phase. Tan [2001] state that, having a central point of storage for log data allows easier access for acquisitions. If the log data is accompanied by sufficient data provenance, it can greatly aid the investigators in identifying and collecting potential evidence. Tan states that centralized logging is the key to efficient forensic strategies.

A survey on the missing capabilities and challenges of cloud forensics is done by Keyun Ruan [2012], this article discuss the biggest challenges associated with cloud forensic investigations. Another survey conducted on cloud forensics by Baggili et al. [2011] identifies five major threats that digital forensic investigators face in cloud computing environments. These threats are:

1. Jurisdictional issues. Jurisdiction in cloud environments is an issue whenever an international incident occurs.
2. External chain of dependencies. This refers to a CSP utilizing services provided by other CSPs.
3. Lack of international collaboration and legislative mechanisms for the exchange and accessing of data.
4. Lack of law and regulation. There is uncertainty regarding the determination of laws and the government of these laws regarding services offered to an international customer base.
5. Decreased control over data and decreased access to forensic data from a client side.

Although similar to a traditional DFI, a cloud forensic investigation is exceedingly more difficult to perform. It is clear, having looked at the major challenges associated with cloud forensics, that a post-event driven investigation is inadequate for cloud forensics. Barbara [2009] states that the biggest challenge digital forensic investigators are faced with in the cloud is to determine the who, what, when, where, how, and why of cloud-based criminal activity. The author believes most of these questions can be answered if the history of data can be provided. The history of a digital object is referred to as data provenance and is discussed in the next section. Data provenance is very important in any digital forensic investigation. Birk and Wegener [2011b] states, the history of a digital object, combined with a suitable authentication scheme is crucial information for a digital forensic investigation.

2.7 Data Provenance

Muniswamy-Reddy and Seltzer [2010] defines data provenance as the history of a digital object. Data provenance reveals valuable information about a digital object, such as when the object was modified, who accessed it and sometimes how it was changed. This information is incredibly valuable in a digital forensic investigation. Proactively capturing and storing data provenance is an excellent example of digital forensic readiness in a cloud environment. Some researchers investigate DFR and data provenance for use in cloud computing environments and cloud forensics[Zhang et al., 2011][Rowlingson, 2004][Muniswamy-Reddy and Seltzer, 2010][Lu et al., 2010][Trenwith and Venter, 2014].

Lu et al. [2010] state that provenance is required in the cloud, because of anonymity. The cloud offers anonymous authentication to users. Anonymous authentication is an authentication scheme allowing users to gain access to a service based on group authentication instead of individ-

ual authentication. Each user in the group is assigned credentials that are calculated based on a mathematical inverse function. The function is stored by the system instead of the user's credentials. Thus, the credentials cannot be used to identify an individual. Different techniques for providing anonymous authentication are discussed in great detail by Lindell [2007]. Lu et al. [2010] states that provenance data will aid in the wider acceptance of cloud computing by the general public, primarily because the public wants to know how their data is being used.

Data provenance can be captured from many different sources, capturing provenance data is looked at later in this dissertation, after identifying specific requirements. Data provenance can be stored in one of two techniques, either embedded within the object it describes or stored separately from the object resulting in a second digital object. These two techniques are briefly discussed emphasizing their advantages and disadvantages.

2.7.1 Embedded storage technique

The embedded storage technique embeds the data provenance inside the digital object, typically in the header of the file. When this technique is used, the integrity of the provenance data is more easily maintained because the provenance data can easily be verified, which is a great advantage. A disadvantage, however, is that the provenance data is difficult to search[Simmhan et al., 2005]. There are two well-known file formats that use this technique, the Flexible Image Transport System used by NASA[Hanisch et al., 2000], and the Spatial Data Transfer Standard[Altheide, 2008] used to reference spatial data by Geographic Information Systems.

2.7.2 Separate provenance storage technique

The provenance data can also be stored separately from the data object. This technique produces a separate data object, which can be a disadvantage from a maintenance point of view. However, it can also help to keep track of a digital object if the provenance records are centrally available. Trenwith and Venter [2014] utilizes separate provenance storage together with a central log server to store and keep track of provenance data.

2.8 Conclusion

In this chapter, the author looked at the background of digital forensics and cloud computing to determine how cloud computing has affected digital forensic investigations. The goal of this chapter was to identify the areas of interest to this work to determine the challenges related to conducting cloud forensic investigations. Having identified these challenges, the author has a better understanding of the domain and can identify the requirements for conducting cloud forensic investigations. Although the main focus of this work is not specifically on cloud forensic investigations, the requirements for conducting cloud forensic investigations should be taken into consideration while developing the digital passport's architecture, if the digital passport is going to succeed in providing value to the digital forensic domain. The author further identified the different levels of granularity when capturing provenance data, some of these levels will not be looked at in this work while others will receive great focus. In the next chapter, the author investigates information security.

Chapter 3

Information Security

3.1 Introduction

Information security relates to the protection of data and information in computer systems. Some information may be sensitive, relating to state or trade secrets and should only be visible to authorized persons, while other information may simply be of a private nature and the owner may not want that information shared with others. Regardless of the reasoning, the need exists to protect information and control access to it. In this chapter, the author investigates the techniques available to provide information security. The author investigates some of the laws and legislation regarding the protection of information in computer systems. These laws state that sensitive data must always be under the control of the controlling authority and may not leave the jurisdiction of the controlling authority. In the next section, the author investigates these laws in more detail.

3.2 Protection of sensitive data and information in the cloud

Government institutes and global organizations often need to protect sensitive information. To address these needs, laws, and legislation has been put in place to regulate the protection of personal and sensitive information. In this section, the data protection laws of the European Union, the United States and South Africa are briefly discussed. These jurisdictions are just some of the jurisdictions that have legislation in place that specifically addresses data protection needs. The author specifically looks at the legislation of the United States and the European Union because these are two of the major jurisdictions in world politics in the author's personal opinion. The author also looks at South Africa because it is the author's home country.

3.2.1 Data Protection: European Union

The European Union has legislation in place to control the protection of information. The one legislative protocol is the Data Protection Directive 1995/46/EC and the second is the e-Privacy directive 2002/58/EC [InfoSec_Institute, 2015].

The Data Protection Directive(DPD) applies to the automated processing of personal data, by computer systems. This directive states that no sensitive data may leave the jurisdiction of the European Union. This directive defines sensitive data as any information that can identify a natural person. However, it does not apply to information regarding natural persons involved in illegal activities, security or defense.

The e-Privacy directive served the purpose of protecting personal data specifically related to the field of telecommunication. This directive regulates location data and data required for communications specifically. It also regulates unsolicited communications such as spam. If an organi-

zation is in breach of this directive, the organization may be reported to national authorities.

3.2.2 Data Protection: United States of America

The United States has privacy laws and legislation both at state and federal level. One of these federal data protection laws is the Health Insurance Portability and Accountability Act(HIPAA). HIPAA is similar to the European Union’s Data Protection Directive. It states that information that can physically identify a natural person may only be used by healthcare professionals for treatment and care coordination [InfoSec_Institute, 2015].

3.2.3 Data Protection: South Africa

South Africa has the Protection of Personal Information(POPI) act. The purpose of the POPI act is to regulate the processing of Personal Information in South Africa[Compliance, 2015]. Personal Information is defined as “any information relating to an identifiable, living natural person or juristic person.”

From what has been discussed in this dissertation so far it is clear that there exists a need to protect personal and sensitive information in the cloud. However, laws and legislation stating that data should be kept secure are not enough. Systems are required to enforce the law. The European Union’s DPD requires sensitive data to stay within the jurisdiction of the European Union. Thus, requiring a system to determine the jurisdiction of users before granting access to sensitive data. In the next section, the author discusses how a user’s jurisdiction can be determined when accessing a CSP.

3.3 Determining the location of a user accessing the cloud

Jaeger et al. [2009] suggests that data should be able to flow easily between the cloud and the user, and makes the statement “Cloud computing only works if the cloud is massive and contiguous.” This statement is true for user data, but as discussed earlier, there are certain cases where the flow of information should be limited or stopped at certain boundaries. With regard to the European Union’s DPD, this boundary is the physical boundary of the European Union. This presents a particular challenge: access control based on physical location.

A goal of this work is to produce a location-based access control mechanism. Access control based on a person’s location refers to the user’s physical location. An example of a system that takes a user’s physical location into account when accessing digital content is the region encoding on a digital versatile disc(DVD). When a region controlled DVD is compiled, it is encoded for a specific region[Sun, 2005]. This encoding is used to control certain aspects of the digital content based on the user’s location, such as the selling price. The region-locked DVD players will only play content from a DVD if that DVD is encoded for the same region as the DVD player.

One of the techniques available to control access to digital content in a cloud computing environment is to use the IP address assigned to the user by the Internet Service Provider(ISP). An IP address identifies a system’s location on a network in the same way a street address identifies a house on the street. A street address must identify a unique residence; similarly, an IP address must be globally unique to the internet[Fall and Stevens, 2011]. Most IP addresses are available for use by public services on the internet, but some address spaces have been reserved for private use in office or home networks. IP addresses on these

private networks are not visible to other networks and are referred to as private IP addresses. The IP addresses assigned to public services by ISPs are referred to as public IP addresses. A reverse lookup of a public IP address against an IP address database can determine where in the world that IP address is assigned. In short, this allows the CSP to control the location from where digital content under its control may be accessed.

A DNS lookup may be necessary to get the IP Address of a connection accessing a service. The Domain Name Service(DNS) is used to translate domain names such as website names to IP addresses[Mockapetris and Dunlap, 1988]. A DNS lookup returns an IP address for a given name, and a reverse DNS lookup does the reverse, given an IP address, a reverse DNS returns the domain name of the IP address. Another service that provides meaningful information when determining the location of an IP address is the WHOIS protocol. WHOIS provides information regarding parties responsible for internet resources[ICANN, 2017].

The Internet Assigned Numbers Authority(IANA) is responsible for the distribution of IP addresses to ISPs around the world[IANA, 2015]. IANA maintains IP addresses in three database groups namely, Regional Internet Registries(RIRs), National Internet Registries(NIRs), and Local Internet Registries(LIRs). The highest level is the five RIRs, which are shown in Figure 3.3.1.



Figure 3.3.1: Regional Internet Registries (Image adopted from: whatismyipaddress.com [2017])

The RIRs are split into NIRs and LIRs. LIRs are mostly ISPs, and they assign IP addresses to their clients[ARIN, 2015]. A DNS lookup determines the IP address of the client and a lookup against the IP address databases can indicate which ISP owns the IP address. There is a challenge though. With this approach, a user can spoof his/her IP address to make it look like his/her internet activity is coming from someone else’s IP address. The Oxford Dictionary defines the word spoof as “Hoax or trick.”[Oxford, 2016]. Spoofing is a real danger when using IP addresses to determine a user’s location. The dangers of spoofing are discussed in the next section.

3.4 Dangers of IP address spoofing and tunneling services

Bishop and Heberlein [1996] discuss the technique of using IP address spoofing as an attack, which highlights the importance of guarding against IP address spoofing. It is possible that the IP address presented to a ser-

vice is not the user's true IP address. In this section, the author discusses some of the techniques used to hide one's IP address.

A common technique used to provide anonymity when browsing the web is using a proxy server, also referred to as a gateway. A proxy server is a server used as an intermediate server between one computer requesting a service and the computer hosting the service.[Brothers and Smith, 2004]. A proxy hides the identity of the requester from the service provider. Proxies are not always used with the primary goal of hiding a user's identity. Large private networks often make use of a proxy to allow all devices on the network to access the internet without requiring each device on the network to have a unique public IP address. Only the proxy is assigned a globally unique public IP address from the ISP. A proxy is often used hand in hand with a firewall. A firewall uses rules to control incoming and outgoing network traffic[Cheswick and Whitten, 2001].

Another common service that provides anonymity is The Onion Router(TOR) project. The Onion Router protects a user's privacy by relaying a user's requests within the TOR network for an unknown number of relays before completing the request by forwarding it to the intended service provider[Dingledine et al., 2004]. At this point, the device establishing the connection to the requested service provider is not the requester's device but another device in the TOR network. The response is sent back in a similar manner, relaying the response from device to device until it reaches the requesting user's device. Figure 3.4.1 shows the danger this network pose when relying on an IP address to determine a user's location.

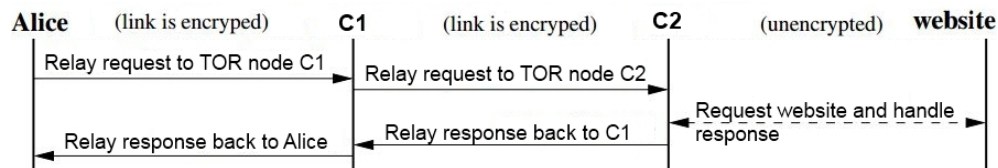


Figure 3.4.1: Simplified TOR Network example

The example shows Alice requesting a website. Two nodes, C1 and C2, relay the request. The node connecting to the website is C2. Thus, as far as the website knows, the request is coming from C2, instead of Alice’s device. The response is also routed back from the website, through nodes C2 and C1 before arriving at Alice. If the access control mechanism uses only the detected IP address, a user using TOR may be granted access to the website. If a user’s location is used to determine access, a user should not be granted access when traffic is routed through external networks outside of the originating host network before being directed to the service provider. The TOR project is one example of a service that routes requests through external networks.

Another service that allows the users to spoof their physical location is a Virtual Private Network(VPN). A VPN is a service that allows a user to connect to the internet via a server run by a VPN provider[Crawford, 2016-01-20]. The VPN further allows a user to connect to the VPN provider’s private network granting the user access to content on the private network that would otherwise be inaccessible from locations outside the network. A VPN provides a user with two primary advantages.

1. The data sent from the VPN provider to the user is encrypted, thus, making it impossible to censor the data.
2. The VPN grants the user access to the internet from a location other than the user’s location, thus, allowing the user to gain access to services that may not be available to the user due to location-based access control.

Because of the services and techniques available to the users as dis-

cussed in this section, location-based access control cannot be established by relying on IP address information alone. In the next section, the author discusses a technique that guards against the dangers discussed in this section.

3.5 Guarding against IP address spoofing and tunneling services

The Challenge-Handshake Authentication Protocol(CHAP) periodically verifies the identity of the peer using a 3-way handshake[Simpson, 1996]. The 3-way handshake synchronizes a sequence number and acknowledgment number of both sides of the connection[Fall and Stevens, 2011]. The sequence number is assigned to every message sent by the Transmission Control Protocol(TCP), and the acknowledgment number is used to acknowledge that the message was received. The use of the sequence and acknowledge numbers ensure reliability by the protocol. TCP initiates a 3-way handshake to establish a communication link between a client and a server[Postel, 2003] [Braden, 1989] [Gont and Bellovin, 2012]. Figure 3.5.1 shows the sequence of a 3-way handshake. The authenticator opens a port on the device capable of receiving connections from initiators[Oracle, 2016]. This port is known as a “server socket” or socket for short. Sockets make use of the Transmission Control Protocol or User Datagram Protocol(UDP). UDP is not discussed in this dissertation, but UDP is defined by RFC768[Postel, 1980].

During the establishment of a TCP handshake, the initiator connects to the authenticator sending a Synchronize packet referred to as *SYN*. The authenticator responds that it received the *SYN*, by replying with a Synchronize-Acknowledgement referred to as a *SYN + ACK*. The initiator responds that it received the *SYN + ACK* by sending an Acknowledgement packet referred to as a *ACK*, which completes the

handshake[Chappell, 2000]. In Section 4.3 the author discusses in great detail how the 3-way handshake is used to verify a client's IP address.

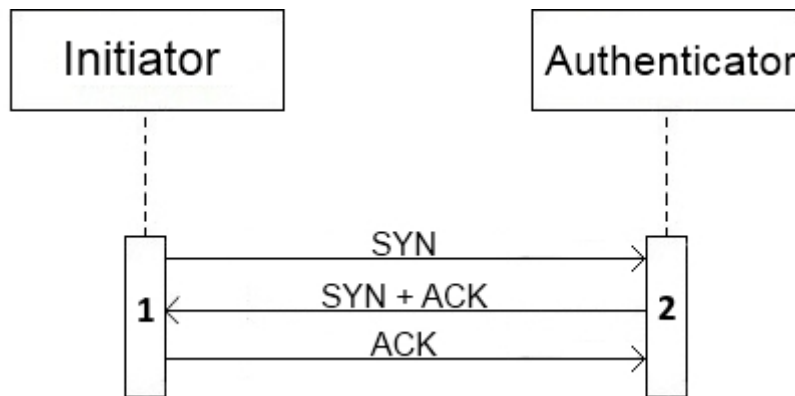


Figure 3.5.1: 3-way handshake

TCP is a reliable, connection-oriented delivery service[Xue and Zhu, 2009]. When an authenticator opens a server socket, it is possible for any device to initiate a 3-way handshake to that socket. Devices that are not on the same network as the authenticator relies on IP address routing to connect to the authenticator. In the next section, some necessary background information is provided regarding IP Address routing.

3.6 IP address routing

A router is a device that forwards packets from one network to another[Fall and Stevens, 2011]. For a router to determine how to forward packets the router consults a routing table stored in memory. Devices connected to the internet can send data to each other by relying on the routing tables that are hosted by routers. These routing tables are created by default during the initialization of TCP/IP. The Internet protocol(IP) is the principle protocol used for relaying data across networks. IP is defined in Postel [1981]. Additional routes can be added manually by a system administrator or automatically through communication with routers. Routing tables are found on any device that initializes the

TCP/IP protocol.

Devices on a private network are assigned private IP addresses by the router[Rekhter et al., 1996]. The router makes use of Network Address Translation(NAT) to remap data packets from the private network to the internet and back[Egevang and Francis, 1994].

For a client-side application running on a device in a private network to receive a TCP socket connection from outside the network requires a port-forwarding rule to be set up on the router. Port forwarding enables the router to forward a connection request received to a specific device on a specific port inside the private network[Cheshire and Krochmal, 2013]. It is also required that the client-side application open a server socket on the specified port to receive connection requests. The Universal Plug and Play(UPnP) protocol allow for the automatic setup of port forwarding rules on routers[Presser et al., 2008]. Using the UPnP protocol on a router allows a device outside the private network to connect to a device inside the private network.

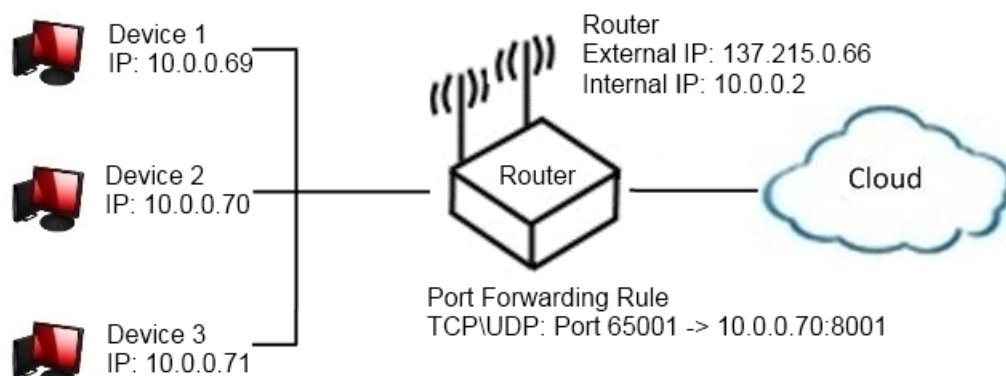


Figure 3.6.1: Port forwarding

Figure 3.6.1 shows the router's two IP addresses. The public IP address is visible from the internet, while the private IP address is only

visible to the devices inside the LAN, i.e., Device 1 to 3. A port-forwarding rule is set up to forward any TCP or UDP protocol communication received by the router on port 65001 to the device with IP address 10.0.0.70(Device-2) on port 8001. Thus, for a device in the cloud referred to as Device-X, to establish a connection to an application on Device-2 requires Device-X to open a TCP or UDP socket to IP address and port: 137.215.0.66:65001. All packets sent from Device-X to Device-2 is routed through the internet by making use of routing tables on IP routers[Fall and Stevens, 2011] Eventually, the packets should reach the destination router which is attached to the destination network. The destination router receives the connection request and through the use of port forwarding, forwards the connection to Device-2 on port 8001.

During the process of transferring data from a CSP to a client, it is important to ensure that others cannot view or modify the data. The objective of an access control mechanism is to ensure that only authorized entities gains access to regulated data[Sandhu et al., 1996a]. Encryption is a well-known technique that ensures that unauthorized parties cannot read sensitive information. Therefore, in the next section, the author briefly discusses cryptography and cryptographic verification of the integrity of data, loosely referred to as hashing or hash codes. While encryption protects the confidentiality of data, the process of cryptographic verification proves that an unauthorized third party did not modify the data.

3.7 Cryptography and verification of data integrity

Whenever data is transferred between a client and a server, and the requirement exists that those devices relaying this data should not be able to read the information, it is necessary to encrypt the data. This ensures

that the confidentiality of the data is maintained. The most common solution to provide this service is referred to as end-to-end encryption [Lu and Sundareshan, 1989]. End-to-end encryption is an encryption technique that encrypts a data packet at the source, and it is only decrypted at the destination. Hence, any device in between the source, and destination cannot read the contents of the data packet.

Symmetric key encryption is most commonly used to transfer larger data files because the performance of a symmetric key encryption algorithm is much better than that of an asymmetric encryption algorithm [Pardo, 2013] [Pfleeger and Pfleeger, 2006, p.767-774]. Symmetric key encryption algorithms are encryption algorithms that use one key for both encryption and decryption of data, which means that the sender and the receiver share the same key. The most common symmetric key encryption algorithm in use today is the Rijndael algorithm, also referred to as the Advanced Encryption Algorithm or AES for short [Jamil, 2004]. The U.S National Institute of Standards and Technology (NIST) believes that the AES algorithm has the potential to remain secure for the next couple of decades. NIST states that hash functions with a key strength of at least 112-bits should be strong enough to withstand attacks until the end of 2030 [Burr, 2005]. To prove the integrity of data the author implements the SHA-256 hash algorithm.

The AES-256 algorithm is a variation of the AES algorithm that uses a 256-bit encryption key. A key exchange algorithm is used to communicate the encryption key from the server to the client to decrypt the data. The key exchange algorithm is discussed in the next section.

3.8 Asymmetric Key Exchange Algorithm

The key exchange algorithm implemented in this work is RSA asymmetric encryption algorithm [Zimmermann, 1986]. The RSA algorithm uses

two keys, the private key is kept secret, and the other is publicly available. The RSA private key and public key are linked through a mathematical inverse [Al Hasib and Haque, 2008].

Rivest et al. [1978] discuss the RSA algorithm in more detail. Consider Bob wants to send an encrypted message M to Alice. Bob obtains Alice's public key: $A^{pub}(n, e)$ which he uses to encrypt the message. The encrypted message C is produced with the following mathematical formula:

$$M^e \bmod(n) = C$$

When Alice receives the encrypted message C from Bob, she decrypts it using her private key $A^{priv}(d, n)$ as follows:

$$C^d \bmod(n) = M$$

The RSA key exchange algorithm does not only provide encryption; it provides authentication as well. Consider the following instruction: $cipher = encrypt(message, key)$ meaning the ciphertext is produced by encrypting the message with the key. Similarly, $message = decrypt(cipher, key)$ means the message can be retrieved by decrypting the ciphertext with the given key.

After Bob encrypts the message M with Alice's public key, he also signs the message with his private key. Anyone can verify the signature using Bob's public key. However, only Bob can produce the signature because only Bob has access to his private key. The message is encrypted using Alice's public key, thus, only Alice can retrieve the original message because her private key is needed to decrypt the message. Thus, any message signed using Bob's private key and encrypted with Alice's public key can only be read by Alice, and after verifying the signature, Alice has an assurance that Bob produced the message. The following formula shows these processes in mathematical form, cypher is produced

by Bob:

$$cypher = sign((encrypt(message, A^{pub}), B^{priv}))$$

Alice can retrieve the original message using the following formula:

$$message = decrypt(cypher, A^{priv})$$

A data object is digitally signed to ensure that any modification to the object does not go unnoticed. To digitally sign a data object requires an RSA private key, as well as the data object's hash code [RSA][Elgamal, 1985]. This process is secure, as long as the RSA private key remains secure. If an attacker attempts to forge the digitally signed data object, the forged data object will not pass a verification process because the attacker does not have access to the original private key.

Figure 3.8.1 shows the process of digitally signing a data object. An RSA private-public key pair is required to produce a digital signature. The private key is used to encrypt the hash code of the data object, producing the digital signature. The digital signature is decrypted with the public key to verify its integrity.

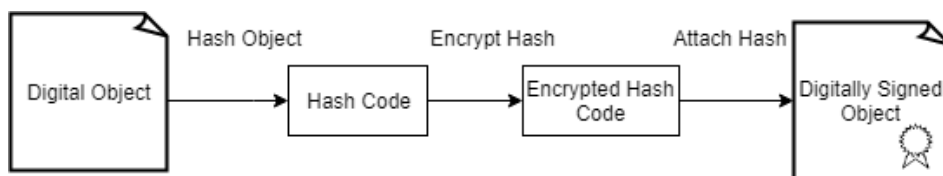


Figure 3.8.1: Digitally Sign Data

Figure 3.8.2 shows the process of verifying the digital signature. The digital signature is extracted from the digitally signed object and decrypted using the signer's public key, which produces a hash. The digital object is also hashed separately and the two hash codes are compared to

each other to determine if the digital object has been altered since it was signed.

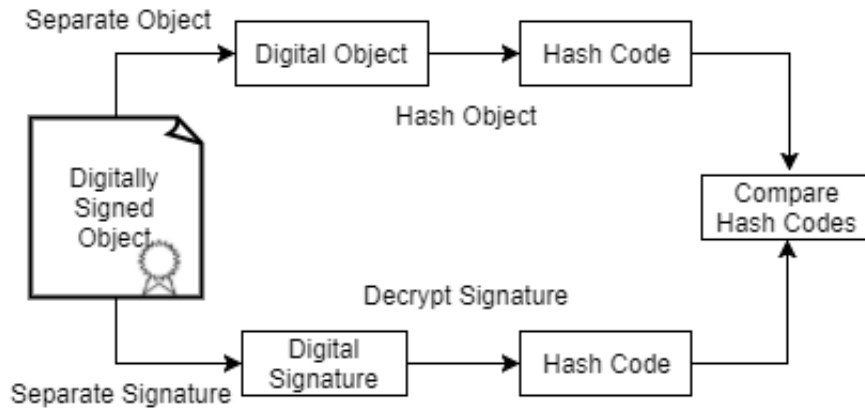


Figure 3.8.2: Process of verifying a Digital Signature

Figure 3.8.3 shows a real-world example of how the RSA algorithm is implemented. The algorithm is used to encrypt an AES encryption key and send it to a client. The server knows only the client can decrypt the key and the client knows the key came from the server.



Figure 3.8.3: Sending RSA encrypted AES Key

Figure 3.8.4 shows a visual representation indicating, after the AES key has been sent to the client, the sensitive information can be encrypted with the AES encryption algorithm and sent to the client who can decrypt the sensitive information.

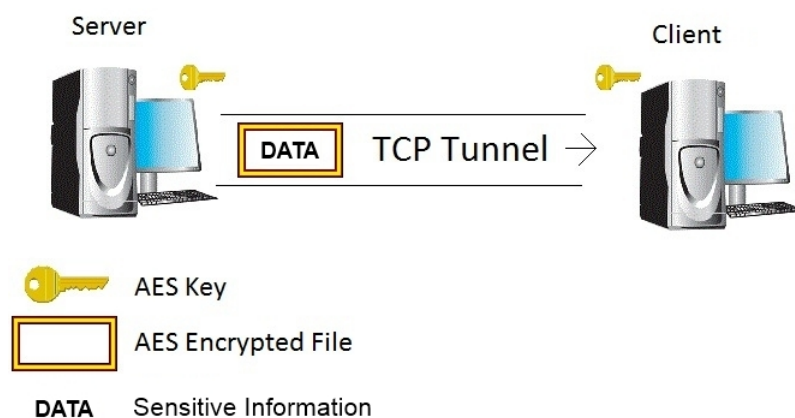


Figure 3.8.4: Sending AES encrypted sensitive information

So far in this chapter, the author has discussed the techniques of digitally signing data objects and how to ensure that a client connecting to a server is not connecting to the server from a proxy. The purpose of implementing techniques such as these is to control access to the data. Access control in digital space is the implementation of algorithms and protocols to control and regulate access to digital resources [Ferraiolo and Kuhn, 2009] [Goyal et al., 2006] [Ferraiolo et al., 2001] [Sandhu et al., 1996b]. Access control is discussed in more detail in the next section.

3.9 Access Control

There exist many different techniques available to control access to digital data. This section discusses a few of these techniques.

3.9.1 Password based access control

Probably the most well-known access control mechanism is the password based mechanism [Angelo et al., 2002] [Eldridge and Kaufman, 2000a]. This technique requires the user to enter his/her username and password. The username may be an alias chosen by the user but sometimes can be the user's email address. The password is an alpha-numeric key

that should be kept secret. If the username and password becomes known to a third party, the third party may access the digital data that should be private. The password based access control mechanism relies on something the user knows[Pfleeger and Pfleeger, 2006, p.221].

3.9.2 Biometric based access control

Biometric access control relies on something the user is, referred to as biometrics[Pfleeger and Pfleeger, 2006, p.219-221]. Biometrics is based on a physical characteristic of the user. These characteristics include but are not limited to a person's fingerprints, a person's voice or facial features that are used in voice or face recognition respectively[Schmitt and Setlak, 1999].

3.9.3 Physical device based access control

Another form of access control is physical based. Physical device access control relies on the user having access to a physical device or "key"[Pfleeger and Pfleeger, 2006, p.219]. This form of access control requires the user to have a physical device that is presented to the access control mechanism when access to a service is requested[Song, 2010].

3.9.4 Location-based access control

Location-based access control enables the service provider to limit access to a service to specific geographical locations, thus only granting users located inside that geographical area access to that service[Ardagna et al., 2006] [Ray and Kumar, 2006] [Denning and MacDoran, 1996]. One example of location-based access control is the implementation of region encoded DVDs and DVD players[Sun, 2005]. A region coded DVD player is designed to play DVD's that are encoded for the same region. It is necessary to note that the design of these region code's is intended to limit access to specific geographical areas, however, the implementation of this design is device specific and not location-based. Therefore, it is possible

to buy a DVD player in Europe along with a DVD encoded for Europe and transfer both to South Africa and playback will work because the device recognizes the encoding.

3.9.5 Multiple techniques based access control

Some of the more secure access control systems require more than one access control mechanism [Yang et al., 2007][Eldridge and Kaufman, 2000b]. These systems rely on more than one technique to verify a user's identity. Thus, the system is more secure because compromising a single mechanism does not compromise the entire system. If a system requires both a user's password and fingerprint, an attacker cannot gain access to a system simply by guessing a user's password. The user's fingerprint is also required, which is much harder to acquire.

3.10 Conclusion

In this chapter, the author discussed information security and its challenges in detail, but without looking at the technical aspects in too much detail. It is necessary to have a thorough understanding of the problem at hand before looking at technical implementations.

Having identified the major threats and challenges associated with cloud forensic investigations, the author compiles a list of requirements necessary to track data through the cloud while providing digital forensic investigators with the information they require in the case of an investigation. In the next chapter, the author develops the architecture for digital passports that focus on protecting user information and capturing data provenance to track a digital passport through the cloud.

Part III

A Model for FReadyPass: Creating Digital Passports to Track the Location of Data in the Cloud

Chapter 4

Location-based access control

4.1 Introduction

In the physical world when a natural person is traveling from one country to another, that person requires a passport, and in some cases a Visa. A passport, issued by an individual's home country, indicates the home country's approval to allow the individual to leave the country. A Visa indicates the visiting country's willingness to allow an individual to enter its borders and remain in the country for a predetermined period. The passport and Visa system provides the ability, to a certain extent to track a person's location to within a specific country. Each country is responsible for the implementation of its Visa system as well as the requirements for procuring a Visa. A Visa acts as an access control mechanism while a passport provides the history of a person's travels, showing where that person has been. In the case that a Visa is not required, the passport is sufficient. The author proposes the design of a forensic ready digital passport referred to as FReadyPass to encapsulate user data and its provenance. The author further proposes the design of a software system to manage the creation of and access to FReadyPasses. This system, referred to as the FReadyPass system, aims to provide both an access control mechanism(ACM) for jurisdictional access, as well as a tracking system, using provenance data describing encapsulated digital

data files to keep track of where the FReadyPass has been. The goal of the ACM is location-based access control referring to a user's physical location and the granularity of the physical location is at the jurisdictional level. The creator of a FReadyPass may choose the jurisdictions from where a FReadyPass may be accessed.

In the remainder of this chapter, the author discusses the access control mechanism, indicating how it works. The author further explains how the ACM guards against IP address spoofing.

4.2 Access Control Rules

Consider Figure 4.2.1. A South African CSP has a server *S* located in Cape Town. The CSP's policy allows only local access; meaning only requests originating from within South African jurisdiction is allowed access to data under the CSP's control. A computer within the borders of South Africa, referred to as local site *L*, should be granted access to *S*. However, a computer outside the borders of South Africa, referred to as remote site *R*, should not be allowed access to *S*.

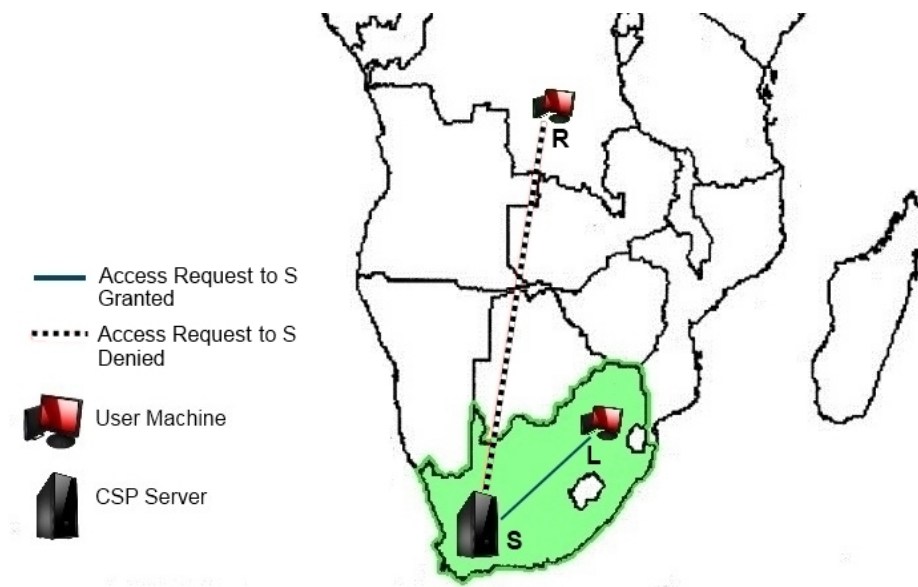


Figure 4.2.1: Location based access

From this scenario, it is possible for the author to establish a set of Access Control Rules(ACRs) for the ACM. The ACM requires a set of rules to determine which access requests to grant and which to deny. The list below is the rules the author finds necessary for the given scenario. The \rightarrow indicates a connection from one machine to another. $L \rightarrow S$ means a connection to S is requested, originating from L. The server S determines which requests should be granted access(Allowed) or denied access(Not Allowed).

ACR1: $L \rightarrow S = \text{Allowed}$

Description: L request access to S and the request is allowed.

ACR2: $R \rightarrow S = \text{Not Allowed}$

Description: R request access to S and the request is not allowed. This access request should not be allowed because it originates from a site outside the jurisdiction of South Africa and in this scenario; the CSP's policy only allows local access requests

ACR3: $R \rightarrow L \rightarrow S =$ Not Allowed

Description: R request access to S and the request is not allowed for the same reason as ACR2. This access request is different from ACR2 because the access request is routed through L. The access control mechanism has to be able to detect such cases.

ACR4: $L \rightarrow R \rightarrow S =$ Allowed

Description: L request access to S and the request is allowed. Even though the request is routed through the remote machine R, the request may be permitted because the communication channel from L to S is encrypted. Thus, the data cannot be read as it is relayed by R.

In short, any access request originating at a remote site, such as ACR2 and ACR3 should not be allowed because the policy of the CSP only allow access when the request originates from within the South African jurisdiction. Therefore, an access request originating from a local site should be granted, even if it is routed through a remote site.

As discussed in Section 3.3 it is possible to determine a user's location with regard to a jurisdictional area by implementing a reverse IP address lookup. A reverse lookup of the user's IP address should point to the user's ISP. The most common way of accessing services provided by a CSP is using the web browser on a device with internet access. Although, other means can be used to access a CSP, such as utilizing tools like Putty [2016]. A web browser connects to the internet making use of protocols like HTTP and FTP automatically, shielding the user from the technical complexities. Regardless of the method employed, the connection still relies on the IP protocol. Thus, any client connection made to a CSP's server contains an IP address. This address can be used to trace where the user connection is being established. In the next section, the author proposes a technique to guard against IP address spoofing by implementing the 3-way handshake technique discussed in section 3.5.

4.3 Guarding against IP address spoofing and tunneling services

In this section, the author discusses and explain how he uses a 3-way handshake to counteract IP address spoofing and tunneling services. When a client connects to a CSP's server through the internet, the server detects the IP address of the device connecting to the server. This may or may not be the IP address assigned to the client device. However, it is possible to prove that the connecting device is the client by treating the client as the authenticator, and the CSP's server as the initiator, as shown by Figure 4.3.1. For the client device to receive an incoming TCP connection requires a running application on the client able to receive TCP connection requests. Figure 4.3.1 shows how a 3-way handshake is initiated by the server. The steps in the figure is numbered in the order in which it is executed.

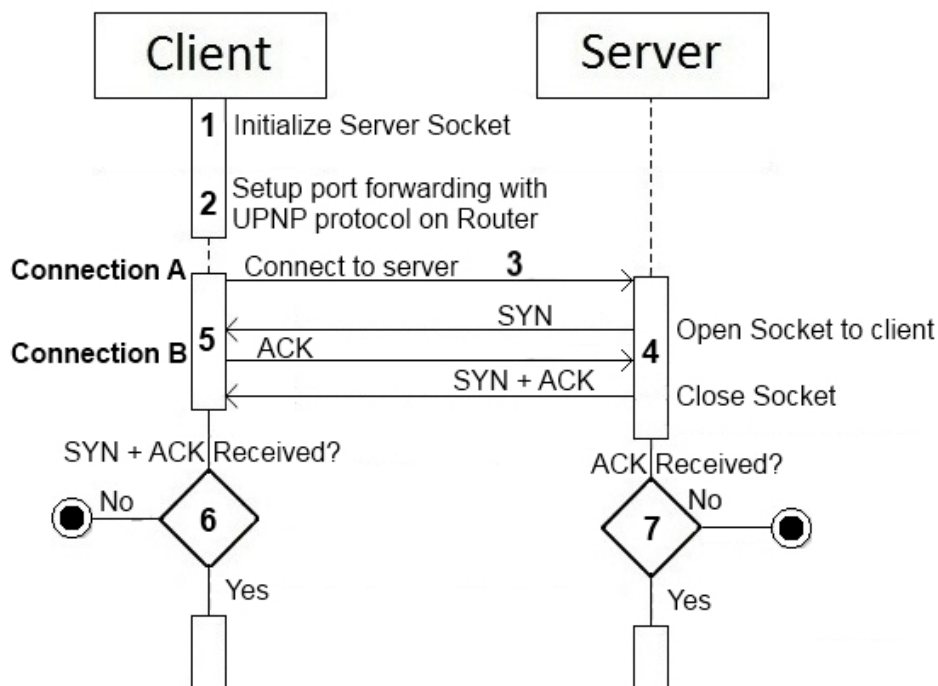


Figure 4.3.1: Authenticate a User Connection

Step 1: The client-side application opens a TCP server socket, awaiting a connection from the server.

Step 2: The client sets up port forwarding on the router to allow an incoming TCP socket connection to reach the client.

Step 3: The client connects to the server. This connection is Connection A.

Step 4: The server performs a reverse lookup of the IP address from connection A established by the client. To determine if the connection is being established using a tunneling service or a spoofed IP address. The server initiates a 3-way handshake to the client on a predefined port establishing a new connection, referred to as Connection B. If the client's IP address is not spoofed, the 3-way handshake will successfully reach the client. If the connection attempt fails, it is likely that the device connected to the server is not the same device that the user is connecting from, either because of IP address spoofing or because the user is using a tunneling service.

Step 5: The server awaits the establishment of a 3-way handshake.

Step 6 and 7: If the 3-way handshake is unsuccessful both the client and server application will end, else the applications will continue to the next step, which is discussed later in this chapter.

When the CSP's access control mechanism receives a connection request from a client(connection A, step 3), the access control mechanism attempts to establish a new communication link (TCP connection) with the client(connection B, step 5), initiating a 3-way handshake to the client after the client's IP address has been determined(step4). Connection B is routed independently from Connection A. Connection B is routed through internet routers using routing tables outside the control of the client. Thus, irrespective of how Connection A was routed from the client's private network to the internet, the client cannot control how Connection B is routed.

Initiating Connection B from the server instead of the client grants

more control to the CSP regarding the access to data. If a client's IP address is spoofed as discussed in Section 3.4, the server's *SYN + ACK* will be sent to the spoofed IP address and will, thus, never reach the client. Thus, the handshake will never be completed. Therefore, the client cannot route its outgoing traffic through a tunneling service. If Connection B cannot be successfully established with the client, the access control mechanism cannot determine the client's location with reasonable certainty. Therefore the request to access data is denied by the access control mechanism if Connection B fails.

Figure 4.3.2, 4.3.3, and 4.3.4 shows three examples of a server initiating a 3-way handshake to three different clients. One client is spoofing an IP address. Another has established a connection to the server routed through a tunneling service. The third device is not spoofing an IP address or making use of a tunneling service.

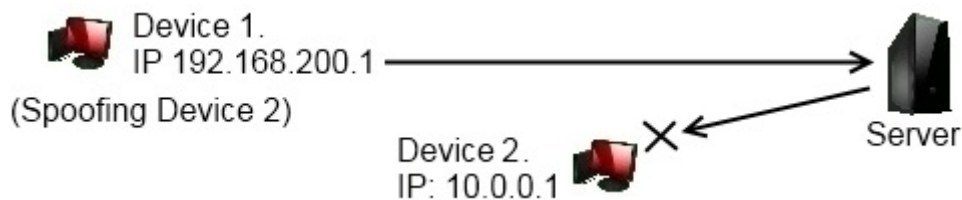


Figure 4.3.2: 3-way handshake initiated with a spoofed IP address

Figure 4.3.2 shows a 3-way handshake initiated from the server, but the client device connecting to the server is spoofing its IP address to make it look like it is another device. A device that is connected to a VPN present these characteristics. The IP address that the server is presented with in Figure 4.3.2 is not that of Device 1, which is the client, but that of Device 2 with IP address 10.0.0.1. Thus, when the server attempts to open a TCP socket connection, the server attempts to establish a handshake with Device 2, which is not where the client application is running. Thus, there is no listening TCP server socket to

receive the handshake, and the handshake attempt fails.



Figure 4.3.3: 3-way handshake initiated with a connection routed through a tunneling service

Figure 4.3.3 shows an example of a tunneling service, similar to what is shown in Figure 3.4.1. Device 1 is communicating with the server, but the connection is routed through Device 2. Thus, the connection established with the server is established from Device 2. Thus, the server has no knowledge of Device 1. When the server attempts to establish a 3-way handshake to verify the client's IP address, the handshake is sent to Device 2, which does not have a listening TCP server socket awaiting the handshake. Thus, the handshake attempt fails.

Figure 4.3.4 shows a connection established with the server from a device that is not spoofing its IP address or making use of a tunneling service. When the server verifies the client's IP address by performing a 3-way handshake, the handshake is successful.

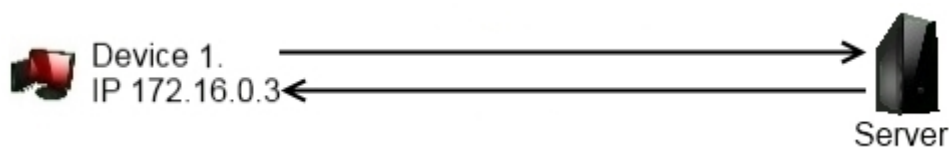


Figure 4.3.4: 3-way handshake on a device not using any tunneling service or IP address spoofing

The use of a client-side application provides additional security benefits including, client-side authentication and the availability of a secure communication channel between the client and the server. Establishing

a 3-way handshake between the client and the server, and initiating this handshake from the server, eliminates the danger of IP address spoofing or tunneling services from being used when communicating with the server.

4.4 Conclusion

By making use of the techniques discussed so far, it is possible to ensure that a client connecting to a CSP is, in fact, connecting from where they appear to be. The CSP can ensure that content under its control can only be downloaded within a jurisdiction that is allowed access to the content. This is done by determining the user's location based on his/her IP address, and ensuring that the IP address is not spoofed, as discussed in this section. However, even if it is possible to counteract the use of IP address spoofing and tunneling services, many routers and other devices will still relay the data packets transmitted from the server to the client. To secure the data and ensure that it is not read or modified by devices relaying the data packets, more commonly referred to as a man-in-the-middle, it is necessary to encrypt the data packets. The architectural requirements that a FReadyPass should aim to achieve is discussed in the next chapter.

Chapter 5

Architectural requirements for FReadyPass

5.1 Introduction

The purpose of this dissertation is to propose an architecture that satisfies three objectives as identified in Section 1.4. The primary objective is to develop a forensically ready digital passport (FReadyPass) to track the location of user data. The author intends to accomplish this objective while providing the ability to control access to the data by using the physical location as one of the access control requirements. In this work, the author takes a close look at the provenance data of user data. The author does not investigate provenance regarding the CSP network, physical machines, and virtual machines. The provenance data collected needs to provide meaningful information about the history of the user's data, specifically relating to its location in the cloud. The captured provenance data should be stored in a forensic ready manner, to adhere to digital forensic standards. In this chapter, the author summarizes the requirements that the FReadyPass should aim to address.

In the next two sections, the author defines functional and non-functional requirements. The functional requirements define the func-

tionality the software needs to achieve. Functional requirements focus on what a software system should do while non-functional requirements focus on how the software system should achieve its objectives[Rainardi, 2008]. From the literacy study conducted in the earlier chapters, some requirements, as well as shortcomings, have been identified. The requirements are discussed in this chapter.

5.2 Functional Requirements

The requirements discussed in this section focus on the functionality the FReadyPass system should provide. The remainder of this section summarize the functional requirements that the architecture should be able to provide. These requirements are deduced from the literacy study conducted earlier in this dissertation.

5.2.1 Single passport single file

In the problem statement of this dissertation, the term digital object was defined for the purpose of this work as “any flat file irrespective of its location or purpose in a computer system”. It is possible to further explain this definition, considering that any digital content streamed from the internet is in some way cached or buffered on secondary storage resulting in temporary files created on a secondary storage device used to playback the content. In a similar manner records in a database is stored in some way in a file object on a secondary storage device to allow for data to persist power outages or other disruptions that could otherwise result in a loss of data. It is based on the knowledge that the FReadyPass system is designed to focus on capturing provenance data for files.

The captured provenance data for a file should be stored. In section 2.7 the author discussed the various techniques available to store provenance data. Considering the legions of files that could be under a CSP’s control the embedded storage technique holds more advantages

than the separate storage technique. Should a CSP implement a separate storage technique as discussed in section 2.7.2, the CSP also has to maintain a database to identify the data provenance files associated with user data files. This database may potentially grow to contain millions or even billions of records. Thus, the maintenance required to service a separate-storage data provenance technique is in itself a challenge that the embedded storage technique does not have to address. To make the FReadyPass system less complicated, easy to use and easier to maintain, all the provenance data associated with a single data file should be stored in the data file, encapsulated into a single FReadyPass, resulting in a single digital object.

5.2.2 Encapsulate any data file

CSPs offer various services and may contain various files. Thus, the type of data that a CSP requires a FReadyPass to encapsulate may vary. It is for this reason that a FReadyPass should be able to encapsulate any data file. Furthermore, the type of data encapsulated in the FReadyPass should not impede how the FReadyPass works.

5.2.3 Standardized file structure

The provenance data should be stored in a standardized file structure to allow for easy access and verification of data. Using a standardized file structure does not only simplify the implementation of a FReadyPass, but it also allows the use of third-party tools to read and verify data provenance contained in a FReadyPass.

5.2.4 Standard protocols

From a practical system design perspective, it is necessary to design a system such as the FReadyPass system in such a way that it uses standard protocols as far as possible. If it does not, the system developer is required to develop specific functionality from scratch. Thus, increasing

the size of the code-base, the time and financial cost necessary to implement and test new functionality, as well as the amount of maintenance required on the system. Furthermore, with regard to system security, one of the tests used to determine if a protocol is secure is the test of time. For a protocol to withstand the test of time requires the protocol to remain secure for a long time. Therefore, it is of utmost importance that the development of FReadyPass focus on the use of standard protocols.

In this section, the requirements are listed that indicates what the FReadyPass system should achieve. In the next section, the requirements are listed that address how the system should accomplish its goals.

5.3 Non-Functional Requirements

5.3.1 Minimum overhead

The architecture should be efficient in its implementation. The processing time required for the FReadyPass system should be as minimal as possible to allow the CSP resources to be allocated towards the services that the CSP aims to provide.

5.3.2 Scalability

As the CSP data grows in size and more FReadyPasses is added to the system it may be necessary to increase resources allocated towards servicing the FReadyPass system to maintain an acceptable quality of service(QoS). Resource allocation should be adjustable with minimal downtime.

5.3.3 Reliability

The FReadyPass system and the FReadyPasses must be reliable. User data encapsulated in a FReadyPass should not be corrupted or lost. A FReadyPass should provide accurate data provenance.

5.3.4 Availability

User data should be available to the owner whenever the owner requests it. The FReadyPass system should not hamper data availability in any way.

5.3.5 Security Requirements

Security requirements has many different elements, which is discussed in the remainder of this section.

5.3.5.1 CIA principles

The FReadyPass system should adhere to the CIA principles: confidentiality, integrity, and availability. The use of encryption and access control ensure confidentiality. Encryption ensures that unauthorized entities cannot access data. Access control ensures that only authorized entities gain access to data. The encryption algorithms used should be fast and efficient to minimize the processing required for encryption and decryption. The encryption algorithms chosen should be standard encryption algorithms that is acceptable to both the academic and commercial industries. The integrity of data is maintained by the implementation of one-way-encryption known as hashing. The final part of CIA is the availability of data. Although this is of utmost importance for CSPs, the CSP infrastructure will already account for availability, and the FReadyPass system need not be concerned with the duplication of data to account for its availability. However, the FReadyPass system should be secure and reliable to ensure that it does not prohibit availability of data for any reason other than unauthorized access.

5.3.5.2 Digital Forensic Requirements

Cloud forensic investigations require a proactive approach; therefore, CSPs need to prepare for potential investigations by applying Digital

Forensic Readiness. Log data and data provenance is crucial information in a cloud forensic investigation. The FReadyPass system must capture sufficient provenance data to aid digital forensic readiness. The provenance data captured must be stored in a forensic ready manner to eliminate any doubt as to its authenticity and integrity. Jurisdictional disputes need to be reduced or eliminated, and legislation such as the European Union’s Data Protection Directive requires the ability to control the access to information across jurisdictional boundaries. Therefore, the access control mechanism is required to consider a user’s physical location when determining access. Access control based on a user’s physical location provides users and Cloud Service Providers with the ability to choose where data may be accessed. Given this ability, the risk of jurisdictional disputes regarding the access of user data may be greatly reduced if not eliminated. However, considering the choice may be to provide All-Access to all jurisdictions, it is not possible to guarantee a reduction or elimination of risk.

5.3.5.3 Access Control Requirements

As discussed in Section 4.2 there are a few access control rules specifically related to location-based access control that the architecture should address. The access control rules can be represented in a formal language. Parkes [2008] defines a formal language as “any (proper or non-proper) subset of the set of all strings, which can be formed using zero or more symbols of the alphabet A ”. Consider the binary number system, the symbols of the binary alphabet is 0 and 1. The alphabet is written in the following manner:

$$\Sigma = [0, 1]$$

To determine if a sentence conforms to the formal language requires the use of formal grammar. Grammar is a set of rules for generating

strings[Parkes, 2008]. The grammar used in this work is known as phrase structure grammars (PSGs). The grammar consists of:

Start symbol S: The start symbol represents the start of a sentence.

Terminal symbols: This is an alphabet containing symbols that can be contained in a sentence. These symbols are always written in lowercase.

Non-terminal symbols: This is an alphabet containing symbols that cannot be contained in a sentence. These symbols are written in uppercase.

Production rules: The production rules define the order in which terminal symbols may be present in a string.

Each production rule defines a valid sequence of symbols that can appear in a sentence. A valid sentence does not contain any non-terminal symbols, therefore whenever a non-terminal symbol appears in a sentence, it must be replaced by another sequence of symbols as defined by the production rules, until the sentence contain no more non-terminal symbols.

The production rules for the binary number system is listed below:

$$S \rightarrow 1$$

A valid sentence in the binary number system is a single symbol 1.

$$S \rightarrow 0$$

A valid sentence in the binary number system is a single symbol 0.

$$S \rightarrow 1S$$

A sentence can start with the symbol 1 followed by other symbols in the alphabet.

$$S \rightarrow 0S$$

A sentence can start with the symbol 0 followed by other symbols in the alphabet.

The start symbol S used in a production rule means it is a placeholder

for another production rule. Meaning, a sentence is not complete as long as it contains a start symbol.

For easy reference, the access control rules as defined in Section 4.2, are listed again before it is represented in a formal language.

ACR1: $L \rightarrow S = \text{Allowed}$

ACR2: $R \rightarrow S = \text{Not Allowed}$

ACR3: $R \rightarrow L \rightarrow S = \text{Not Allowed}$

ACR4: $L \rightarrow R \rightarrow S = \text{Allowed}$

ACR2 and ACR3 are not allowed and therefore is not defined as part of the production rules because they are not considered valid. The production rules are defined to state ACR1 and ACR4 formally.

Consider the following alphabet to represent the access control rules:

$$\Sigma = [r, l, s]$$

In this alphabet, the symbol r represents a remote device, the symbol l represents a local device, and the symbol s represents a server.

The formal language contains the following non-terminal symbols:

$$N = [B, S]$$

S represents the start symbol indicating this is the start of a sentence. B represents a sequence of symbols to follow as defined by the production rules. The non-terminal symbols are used to define the production rules that determine the order in which symbols in the alphabet may appear. All non-terminal symbols are written in uppercase, and all terminal symbols are written in lowercase. Capitalization is used to distinguish non-terminal symbols from terminal symbols. The access control rules are represented by the following production rules:

$$B \rightarrow r$$

B can be a single symbol r.

$$B \rightarrow l$$

B can be a single symbol l.

$$B \rightarrow rB$$

B can start with a symbol r followed by other symbols as defined by the production rules of B.

$$B \rightarrow lB$$

B can start with a symbol l followed by other symbols as defined by the production rules of B.

$$S \rightarrow ls$$

A valid and complete sentence is ls. This sentence is complete because it does not contain non-terminal symbols.

$$S \rightarrow lBs$$

A sentence can start with the symbol l followed by any one of the sequences of symbols as defined by the production rules of B.

The above-mentioned production rules state that the connection must originate at l. This is shown by the two production rules that define the start symbol S, these represent the start of a sentence or as it is used in this work, the start of a connection. Both production rules defining the start symbol S has as its very first symbol the terminal symbol l. The next symbol in the production rule can either be the terminal symbol s or the non-terminal symbol B followed by the terminal symbol s. Because both production rules for the non-terminal symbol S ends on the terminal symbol s, it indicates that the sentence or connection has to terminate at s, the server. Whenever the non-terminal symbol B is present in a sentence one has to replace that symbol by any sequence of symbols defined by the production rules for the symbol B. This sequence of symbols is either: r, l, rB, or lB as defined above. A complete sentence is a sentence that contains only terminal symbols such as the sentence $S = lrrlls$. This sentence defines the path that a connection traversed before arriving at the server s, the connection was started at a local site l,

and routed through three remote sites and two local sites before arriving at the server s.

Next, the author addresses the requirements that the data provenance is expected to achieve.

5.3.5.4 Data Provenance Requirements

Lu et al. [2010] identified three requirements that a provenance record should have, similar to the CIA principles; these requirements are listed as R1 to R3 below. In the work of Shi et al. [2010] and Juels and Kaliski Jr [2007], a challenge was raised; a requirement exists to provide cryptographic proofs for verifying data integrity within the cloud. Trenwith and Venter [2014] addresses the challenges digital forensics investigators are faced within the cloud. These challenges are addressed as requirement R4 to R7 in the proposed model. The requirements are listed below:

- R1:** A provenance record needs to be unforgeable.
- R2:** A provenance record needs to be kept confidential.
- R3:** The integrity of a provenance record should be maintained by the system.
- R4:** A provenance record should show who is responsible for a modification.
- R5:** A provenance record should show what was modified.
- R6:** A provenance record should show the time of a modification.
- R7:** A provenance record should show the object's location in the cloud at the time of a modification.

The FReadyPass system should provide non-repudiation and attribution. Non-repudiation refers to the ability of the system to prove that an entity is responsible for an action taking place. Consider R4, the system requires the ability to provide proof that cannot be disputed that an entity is responsible for the modification to a FReadyPass resulting in

a provenance record being produced. Attribution, as discussed in the background section of this work, refers to the ability of the FReadyPass system to link a digital event to a physical person. A way of achieving attribution is discussed in greater detail in section 7.2.4 of this dissertation.

If all provenance records adhere to the requirements laid out in this section, it should contain sufficient information to answer the important questions digital forensic investigators ask during the process of an investigation.

5.4 Conclusion

The goal of this chapter is to highlight and summarize the requirements that the FReadyPass system should meet. The requirements are organized in this chapter according to the category that they belong too. In the next chapter, the design of the FReadyPass is discussed, the requirements listed in this chapter is taken into consideration during the design of the FReadyPass system. The author also discusses the cryptographic processes implemented to secure a FReadyPass.

Chapter 6

Architectural Design of FReadyPass

6.1 Introduction

The author investigates the optimal design of FReadyPass that best fits the requirements of the model. The author continues to investigate security mechanisms that are required to keep a FReadyPass secure, at the cloud service provider and the client. The author also investigates the security protocols required to secure a FReadyPass during the process of uploading and downloading the FReadyPass. The FReadyPass is intended to control access to user data; therefore, it is necessary to ensure that data cannot be leaked from the FReadyPass. If the data is leaked and becomes accessible outside the FReadyPass, it is no longer possible to control access to it.

6.2 Designing a FReadyPass

The author proposes the design of a forensic ready digital passport (FReadyPass), shown in figure 6.2.1. The FReadyPass stores both the user's uploaded data as well as the provenance data of the user data since it

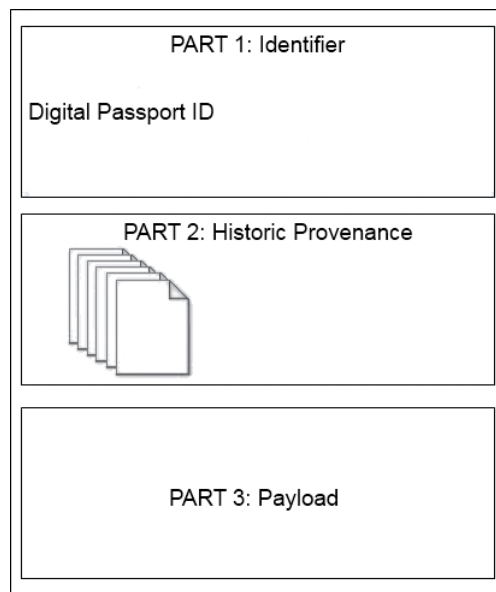


Figure 6.2.1: FReadyPass design

came under the control of the CSP. The data is stored in a single digital object. The FReadyPass is encrypted to protect the confidentiality and integrity of sensitive data. The FReadyPasses under the CSP's control may be legion; therefore, a unique ID is required to distinguish between FReadyPasses. The design of the FReadyPass is discussed in the next three sections.

6.2.1 Part 1: Identifier

Part 1, contains the ID of the FReadyPass. The ID is a unique key that is assigned to the FReadyPass when it is constructed. The ID is required to distinguish between the many FReadyPasses and is used to lookup metadata stored in a central database that is associated with the specific FReadyPass.

6.2.2 Part-2: Historic Provenance Data

This part contains the provenance data of the payload. The payload refers to the third part, which is discussed shortly. The provenance data describes the current state of the payload as well as modifications made to the payload since it came under the control of the CSP. The provenance data aims to address requirements R4 to R7. The author discusses exactly how the requirements are addressed in Section 7.2. The provenance data captured needs to describe the time of a modification, the FReadyPass's location, and the process, or user account that modified the FReadyPass. A provenance record is captured when the FReadyPass is downloaded to a client or uploaded to the server. A provenance record is also captured at the client when the FReadyPass is modified. The provenance records are chained together to ensure that the integrity of the provenance records is maintained. The design of the provenance records, as well as the process of verifying the integrity of provenance records, are discussed in detail later in this dissertation.

6.2.3 Part-3: Payload

The payload is the user's data file. By design, this can be any file. This architecture aims to allow the CSP or the owner of the data file to encapsulate any file in a FReadyPass. The policy of the CSP determines if user data should be encapsulated in a FReadyPass. The policy may dictate that Microsoft Word Documents has to be encapsulated, but the encapsulation of PDF documents are optional. In this manner, the CSP may keep provenance data on all files associated with client accounts or may keep provenance data on only specific types of files associated with sensitive user data. Irrespective of the policy, provenance data cannot be stored without a FReadyPass. The policy established by the CSP is enforced when a user uploads a file to the CSP. The process of constructing a FReadyPass is discussed in the next section.

6.3 Overview of constructing a FReadyPass

When a file is uploaded to a CSP that requires the capturing of provenance data, that file is encapsulated into a FReadyPass. The jurisdictional areas that may access the FReadyPass are determined and stored. A FReadyPass may be accessed from within a certain jurisdiction or group of jurisdictions. As discussed in Section 3.2 certain documents deemed sensitive might not leave certain jurisdictional areas. Unfortunately, it is not possible to state with certainty that all files of a specific type such as a Word document or a PDF document contain sensitive information. Therefore, the CSP has a choice, it may elect to allow access to data under its control to a certain subset of jurisdictions, or it may provide the user with the freedom to choose where his/her data may be accessed. However, the latter choice may violate legislation discussed in Section 3.2, depending on the nature and sensitivity of the user's data files. However, the design of the FReadyPass allows for both cases, having the CSP enforce the jurisdictional areas in which data may be accessed, based on policy, as well as allowing the user to assign allowed jurisdictions for his/her data manually.

The author proposes a group access policy to govern user accounts. Consider the following explanation: depending on the nature of the business of a user, certain users may require similar access and privileges. Therefore, user accounts are assigned to groups with similar privileges based on what is required.

Consider the following scenario with regard to the European Union's Data Protection Directive: Two users, Alice and Bob, both European citizens, each creates a user account with the CSP. Alice is a private citizen. Bob is a government official. The CSP employs a group access policy where one group has "Grant-All" access, meaning, FReadyPasses of members of this group may be accessed from anywhere in the world. The other group has "European Union Only" access, meaning FReady-

Passes of members in this group may only be accessed from within the European Union. Alice, the private citizen, is assigned the privileges of the “Grant-All” access group, while Bob, the government official is given the privileges of the “European Union Only” access group. Thus, Alice may access her content from anywhere in the world, but Bob can only access his data from within the European Union. This fulfills the requirements of legislation such as the European Union’s DPD.

Jurisdictional access to a FReadyPass is determined and assigned during the process of creating the FReadyPass. The FReadyPass can be created by uploading a data file to the CSP’s server and having the server construct the FReadyPass, or by making use of a client-side application to construct the FReadyPass after which the FReadyPass is uploaded to the server. Figure 6.3.1 shows the process of uploading a data file to the server and having the server construct the FReadyPass.

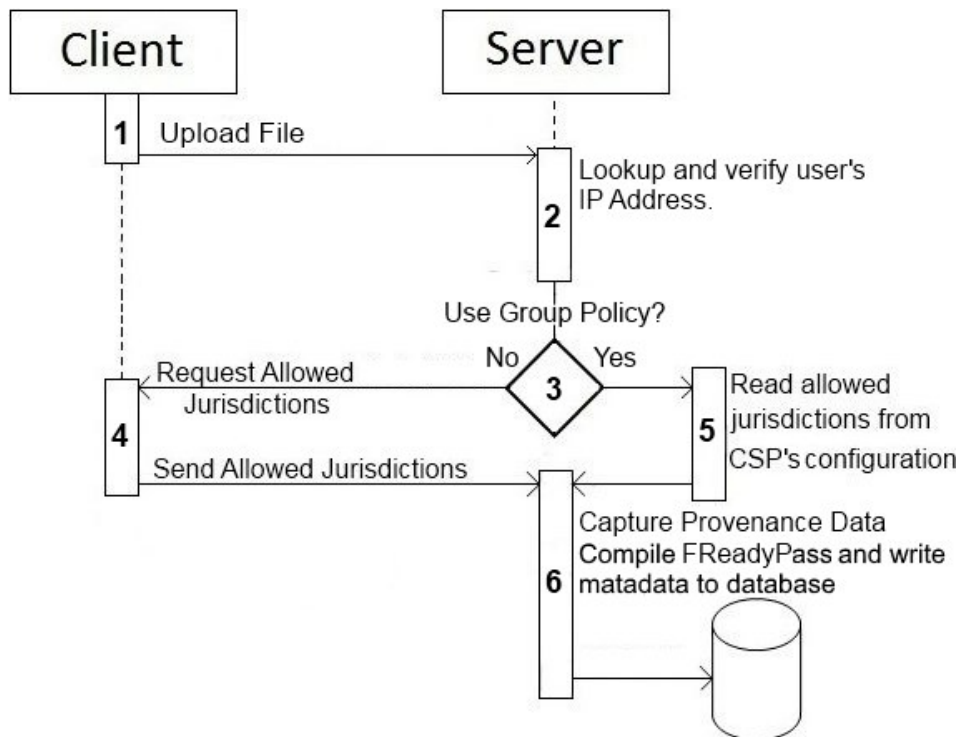


Figure 6.3.1: Constructing the FReadyPass at the server

Each step in the process is explained below:

Step 1: The user uploads a data file to the CSP.

Step 2: The CSP determines the location of the user uploading the file. The user's location is essential in the process of constructing the initial provenance record, indicating where the file came from.

Step 3: The system determines whether, for the specific user, the group access policy should be followed or whether the user should be prompted to determine the allowed jurisdictional areas for the uploaded file.

Step 4: This step is only executed if the user has the right to decide the permitted judicial regions from where the FReadyPass may be accessed.

Step 5: If the group access policy is followed, this step is executed. The CSP determines which jurisdictional areas shall have access to the created FReadyPass, based on the access rights of the group the user belongs to.

Step 6: Finally, provenance data is captured for the data file, and the FReadyPass is constructed, and the metadata of the constructed FReadyPass with its access privileges is written to the database.

Figure 6.3.2 shows the process of constructing a FReadyPass at the client and uploading it to the server after its construction is complete.

Each step in the process is explained below:

Step 1: The client application connects to the server

Step 2: The server verifies the client's IP Address by executing the 3-way handshake as shown in Figure 3.5.1.

Step 3: The client requests a unique ID from the server to assign to and identify the FReadyPass with.

Step 4: The unique ID generated by the server is a Universally Unique Identifier (UUID) as defined in RFC4122[Leach et al., 2005]. The UUID is assigned to the new FReadyPass.

Step 5: Provenance data is captured describing the user data file, the FReadyPass is constructed, and the FReadyPass is uploaded to the server.

Step 6: The server captures a location-update provenance record. The

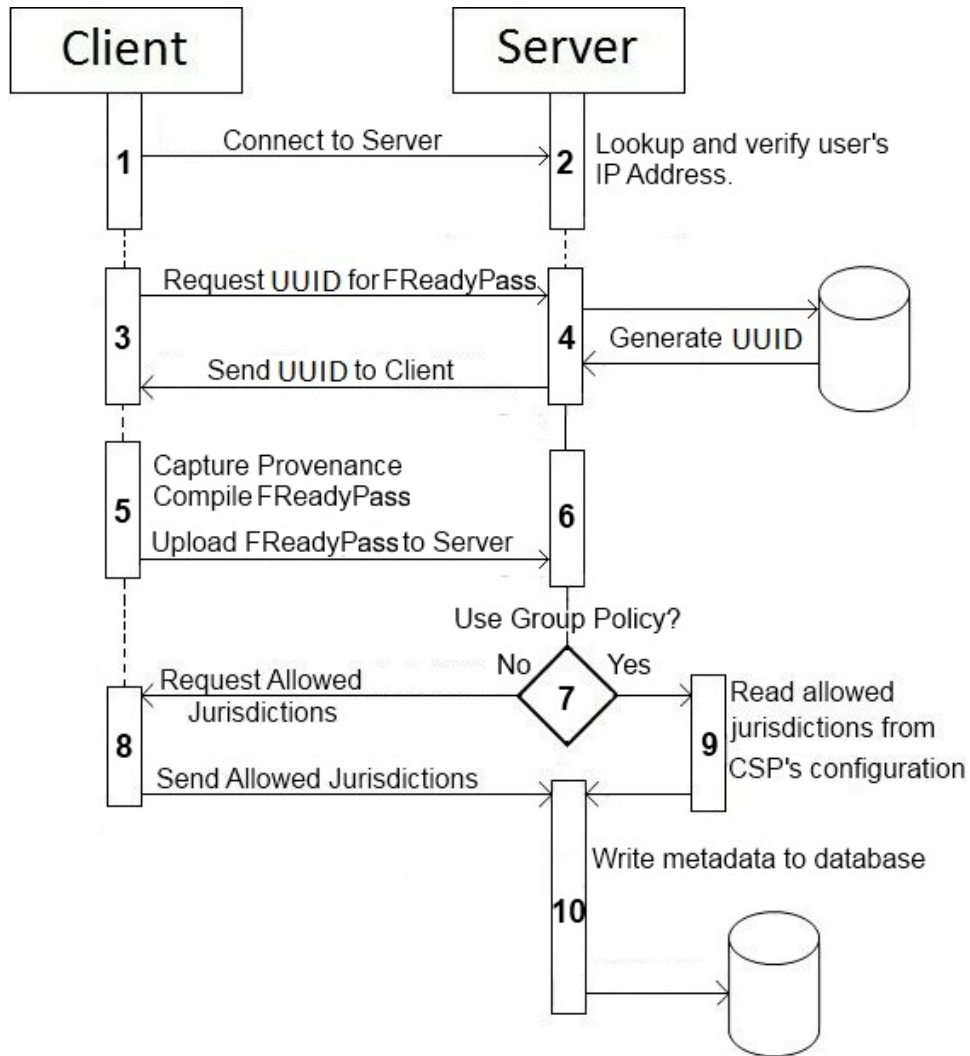


Figure 6.3.2: Constructing the FReadyPass at the client

different types of provenance records are discussed in detail in Chapter 7.

Step 7: The system determines whether, for the specific user, the group access policy is followed, or whether the user should be prompted to determine the allowed jurisdictional areas for the uploaded file.

Step 8: This step is only executed if the user has the right to decide the permitted judicial regions from where the FReadyPass may be accessed. The user is prompted to provide the allowed jurisdiction from where the FReadyPass may be accessed.

Step 9: If the group access policy is followed this step is executed. The CSP determines which jurisdictional areas shall have access to the created FReadyPass, based on the access rights of the group the user belongs to.

Step 10: Finally, the metadata of the constructed FReadyPass with its access privileges is written to the database.

A FReadyPass, just like any other digital object is initially created, and after that, it can be modified. When a user uploads a file to a CSP for the very first time, a newly created FReadyPass is required to store that file. After that, every time the user downloads this file the FReadyPass is accessed to present the user with his/her file. The user reads and/or modifies the file content, and upon saving the possible changes, provenance data is captured, and the file together with the provenance data is stored in the FReadyPass, resulting in the FReadyPass being modified. The next section presents the technical breakdown of the FReadyPass, showing how the different parts fit together.

6.4 FReadyPass file structure

The header of a FReadyPass is 32 bytes long and is shown in table 6.1.

16-bytes (128-bits) are required to store a UUID. The remaining 16-bytes of the header is used to store the length of the provenance data and the length of the payload respectively. The header allows the pay-

UUID	Provenance Length	Payload Length
16 bytes	8 bytes	8 bytes

Table 6.1: FReadyPass header architecture

load length to be indicated using up to 8-bytes (64-bits). Similarly, the length of the provenance data is indicated using 8-bytes. Therefore, in theory, the FReadyPass can encapsulate a file that is up to 2^{64} bytes (18.4 exabytes) in size. The FReadyPass supports provenance data of the same length.

A value of “1” in the Payload Length field indicates that the payload has a size of 1 byte. The data in the header is formatted and stored in a byte array using two’s complement representation[Finley, 2000]. Two’s complement representation is used due to its popularity in computer hardware to represent signed numbers[Salonen, 2013][Ostrovsky, 2010]. The Java programming language exposes functionality to get the binary representation of a UUID, and this binary representation is in two’s complement. Therefore, the rest of the FReadyPass header is also stored as two’s complement. A single byte has a decimal range of 0 to 255. Table 6.2 shows how a decimal number is represented using a signed two’s complement value. From 0 to 127 the signed two’s complement representation is the same as the original decimal value. From 128 to 255 the signed two’s complement becomes -128 down to -1. If the number represented is greater than 255 the value rolls over and a second number is required as is explained shortly.

As an example used to explain how the header is constructed consider a FReadyPass with a UUID of ed42e942-46be-4bbe-9faf-9752d55224d6. Remember this value is generated as indicated by RFC4122[Leach et al., 2005]. The provenance data in this example consist of 1211-bytes, and the payload, in this case, has a length of 2,072,275-bytes. The UUID, Provenance Length and Payload Length are converted to respective byte arrays

8-bit Binary	Decimal Number	Signed Two's Compliment
00000000	0	0
00000001	1	1
00000010	2	2
...
01111111	127	127
10000000	128	-128
10000001	129	-127
...
10111011	187	-69
...
11111110	254	-2
11111111	255	-1

Table 6.2: Number to Two's compliment conversion

and stored in the FReadyPass header. The header for this FReadyPass is shown in Table 6.3.

Consider, for example, the Provenance Length field shown in Table 6.3, the length is 1211. Remember that two's complement can represent 256 numbers using only a single byte, which in turn, can represent only numbers 0 to 255. Therefore, to convert 1211 to signed two's complement, we divide by 256.

$$1211/256 = 4 \text{ remainder } 187.$$

The remainder 187 is less than 256.

Referring back to table 6.2 the Two's complement representation of 4 remains 4 and the Two's complement representation of 187 is -69. Thus, the decimal number 1211 is equal to the signed two's complement number 00000100 11000101 which is also represented as 4,-69 since the two's complement binary number 00000100 is equal to the decimal number 4 and the two's complement binary number 11000101 is equal to the decimal number -69.

Thus, to save the provenance length in the FReadyPass header the value 4,-69 is stored in the 8-byte Provenance Length field as is shown in table 6.3. The values of the UUID and the Payload Length fields are calculated similarly.

Field	Value
UUID	137,66,233,66,70,190,75,-66,159,175,151,82,213,82,36,-42
Provenance Length	0,0,0,0,0,0,4,-69
Payload Length	0,0,0,0,0,31,-98,-45

Table 6.3: FReadyPass sample header

6.5 Conclusion

Figure 6.3.1 and Figure 6.3.2 showed the processes of constructing and uploading a FReadyPass that tracks and protects a user's data file. One step that has been left out of this process is the capturing of provenance data. The FReadyPass is designed to encapsulate user data as well as the provenance data and associated metadata required to identify the FReadyPass for quick and easy access. The author stated that the provenance data is stored inside the FReadyPass, but not how the data is stored or verified. In the next chapter, the author discusses the design of the provenance data storage inside the FReadyPass.

Chapter 7

Provenance Data

7.1 Introduction

In this chapter, the author provides a detailed discussion about how the provenance data is stored in the FReadyPass as well as how the process of verifying the integrity of the provenance data works. The provenance data contains information about the state of the user's data. Therefore, the process of verifying the integrity of the provenance data also verifies the integrity of the user's data. In the next section, the author discusses the requirements that a provenance record should meet and how the FReadyPass design address these requirements.

7.2 Capturing Provenance Data

This section is a discussion section regarding the process of capturing provenance data. In Chapter 5 the author discussed what is required from provenance data. Requirements R1 to R3 are requirements concerning the system capturing and maintaining provenance data. Requirements R4 to R7 states what is expected from provenance data. The model proposed by the author can capture provenance data at the server and the client. In the remainder of this section, the author discusses each of these requirements and how they are met.

7.2.1 Meeting R1. A provenance record needs to be unforgeable.

To ensure that a provenance record cannot be forged requires two things. First, only the FReadyPass system should be able to produce a provenance record. Therefore, the process of creating a provenance record should contain a step that authenticates that it is either the FReadyPass client or the FReadyPass server that is creating the provenance record. The author wants the identity of the entity producing the provenance record to be known to enhance the integrity of the provenance records.

Second, a verification mechanism is required to verify a provenance record to prove its authenticity as well as prove that the records integrity is intact.

A mechanism that provides the capability to accomplish both of these steps is the RSA sign and verify functions, which are discussed in Section 3.8. Each provenance record produced by the FReadyPass system is signed by either the client or server depending on which one created the provenance record. Provenance records that are signed by a client that is in possession of a FReadyPass are verified by the server when the FReadyPass is uploaded back to the server after the client has finished with the FReadyPass. The process is discussed in more detail later in this chapter.

7.2.2 Meeting R2. A provenance record needs to be kept confidential.

It is necessary to ensure that only authorized persons can read the provenance records to ensure confidentiality. Encryption provides a good foundation to achieve confidentiality. Provenance data needs to be made available to digital forensic investigators upon request. Furthermore, Lu et al. [2010] referred that cloud users want to know how their data is

being used. Thus, the argument can be made that the user that created a FReadyPass should have the ability to view the provenance data, but should not have the ability to alter it. Thus, provenance data should be viewable to the owner of the FReadyPass, and the CSP requires the ability to view provenance data of all FReadyPasses under its control.

Provenance records can be viewed using a Provenance viewing application that is discussed in Section 9.8. The Provenance viewing application may be used when provenance data has to be made available to a Digital Forensic Investigation team, or on request, the owner of a payload encapsulated in a FReadyPass. The owner of a payload may want to see the provenance data to determine how his/her data has been accessed in and out of the cloud and where it has been.

The process of making provenance data available is discussed in greater detail in Section 9.8.

7.2.3 Meeting R3. The integrity of a provenance record should be maintained by the system.

The process of verifying the digital signature of a provenance record prove the integrity of the provenance record. The signature is produced by creating a cryptographic hash code of the provenance record, which is known as the original hash code. The hash code is calculated by the entity that is producing the provenance record. The process of verifying the signature can be done by any entity. The FReadyPass server verifies digital signatures produced by FReadyPass clients.

To verify a digital signature requires that the provenance record is hashed again, which is referred to as a verification hash. Therefore, if the original provenance record were modified, the verification hash code would differ from the original hash code. Thus, if a provenance record has been altered, invalidating its integrity, the verification of the signa-

ture would fail, exposing the alteration. Therefore, a provenance record cannot be altered without the alteration being detected. This process can be extended to chain together all the digitally-signed provenance records ensuring that a provenance record cannot be removed from the FReadyPass or a false provenance record inserted. The processes of chaining provenance records are different from blockchain technology in its implementation. Blockchain technology and the differences between blockchain and this work is discussed in detail in chapter 10. Many researchers are doing work on Blockchain[Crosby et al., 2016a][Nelson et al., 2016][Liang et al., 2017].

7.2.4 Meeting R4. A provenance record should show who is responsible for a modification to the payload.

The user is authenticated before reading or modifying a FReadyPass to determine if the user has access to the FReadyPass. Therefore, the system knows which user account is accessing the FReadyPass.

Unfortunately, what cannot be determined is whom the attributed person is using the account. [Cohen, 2011, p.44-51] states that it is only possible to link a physical person to a digital event given an anchor event and non-repudiation. An anchor event is an event that ties an event in digital space to a person in physical space, such as a video recording showing a person accessing a computer terminal at the exact time of the digital event taking place on that computer terminal. If the user has a webcam connected to his or her computer, it is possible to request a photo showing the user's face, when the user attempts to access a FReadyPass.

Such an attribution step will, however, not be implemented in this work, because facial recognition is not a new concept in the access control domain and adding such a process to this work offers little scientific

value to this work.

7.2.5 Meeting R5. A provenance record should show what was modified.

The granularity of indicating how a FReadyPass was modified can vary. From merely keeping track of the current state of the FReadyPass by making use of the implementation of cryptographic hash functions, to indicating which areas of the FReadyPass have been modified. The Flogger tool discussed in Section 10.2.5 can indicate which blocks of data in a file has been modified. In this work, the author keeps track of the state of a FReadyPass by making use of the SHA-256 cryptographic hash algorithm to capture the current state of a FReadyPass.

7.2.6 Meeting R6. A provenance record should show the time of a modification.

Capturing the time of a modification to a payload is of vital importance to construct a timeline indicating the location of a FReadyPass at any one point in time. However, considering the widespread nature of the cloud and the fact that clients may be anywhere in the world, all captured timestamps should be stored as Coordinated Universal Time(UTC)[TimeAndDate.com, 2017]. UTC does not change to accommodate Daylight Saving Time.

The clock on the device accessing a FReadyPass should be synced to avoid overlapping timestamps that can cause confusion due to a device's clock being incorrect. The Network Time Protocol(NTP) is an acceptable protocol to implement on both the client and server side applications to ensure all the clocks are synced[Mills et al., 2010]. Tan [2001] suggests that the NTP is the most efficient protocol for achieving time synchro-

nization on IP based systems. The NTP connects to one or more NTP servers to synchronize the clock. The NTP servers have access to highly precise atomic clocks and GPS clocks[TechTarget.com, 2017a]. The network time protocol is a protocol, which is used to synchronize time clocks. The time provided by NTP servers is formatted in UTC[Chirico, 2017]

7.2.7 Meeting R7. A provenance record should show the FReadyPasses location in the cloud at the time of a modification.

As discussed in section 4, an IP address is captured and verified to be truthful before allowing access to any FReadyPass. Along with logging the IP address of the device accessing the FReadyPass, the jurisdiction is also recorded. The jurisdiction and IP address are logged together because the possibility exists that an ISP goes out of business for any number of reasons. If this happens the range of IP addresses that the ISP is responsible for managing has to be reassigned to another ISP, possibly in a different jurisdiction.

In the next section, the author designs a provenance record to address all the requirements.

7.3 Provenance Record Design

Listing 7.1 shows an example of the design of a provenance record that captures all the requirements mentioned above. The XML-file structure is used to store provenance records. This complies with the Standardized File Structure requirement stated in Section 5.3.

```

1 <ProvenanceRecord>
2   <ProvenanceData>
3     <Chain>
4       08861eb7321747b910570677b4de92da9181097059c370def97bca39f9f3bcfa
5     </Chain>
6     <Number>2</Number>
7     <RecordType>Modification</RecordType>
8     <Identity>
9       <OSDomain></OSDomain>
10      <OSUsername>Philip</OSUsername>
11      <UserAccount>ptrenwith@gmail.com</UserAccount>
12      <MachineName>PhilipT-PC</MachineName>
13    </Identity>
14    <Location>
15      <IPAddress>137.215.0.66</IPAddress>
16      <Jurisdiction>ZA</Jurisdiction>
17    </Location>
18    <Timestamp>1430251065774</Timestamp>
19    <HashCode>
20      78c0c177eeaf438cccbda3aa3226fc4a18a98c2e672800a7ab781efb8e6c485
21    </HashCode>
22  </ProvenanceData>
23  <DigitalSignature>
24    B1i+WDjJtu5Bdu7XxOYGWCVPztyMHccvr0Nu8b0V0Y/mxBu68FJaTN
25    mZBup4uVQ14pbGUruZvFHLk8VbHVHF+Ltc8sGguNWBOQFSn+buWG
26    apNiqawwtiWuc7nO8BcdQOEIUqwH3xHslO500ZajoMETiw1qTqkpd7txZd
27    cB+8vA1Lpi9riIAxaWe8Ka79h4iVLwsGjAV9pXicOEO4gy4rwT58MLn4V3
28    TPQIwtXYghZsaOvP6E/DDtZbPKTy4GLCpTut94v4ZZZeDXCZpQwhJJ1
29    TFFf3tZ8avM2Yp3krNWkEE6FzSI3MPS/OVai1oQhCsoSC/q824PxjYyWIB
30    H7WEi1A==
31  </DigitalSignature>
32 </ProvenanceRecord>

```

Listing 7.1: Provenance Record

The provenance record is shown in listing 7.1, the provenance record starts with the XML ProvenanceRecord tag shown on line 1. Each opening tag in an XML file must have a corresponding closing tag, the ProvenanceRecord closing tag is shown on line 32. The provenance record exists

of two main sections, the ProvenanceData section, visible in Listing 7.1 from line 2 to 22, and the DigitalSignature section, visible from line 23 to 31. Figure 7.3.1 shows the events that trigger the capturing of the different types of provenance records.

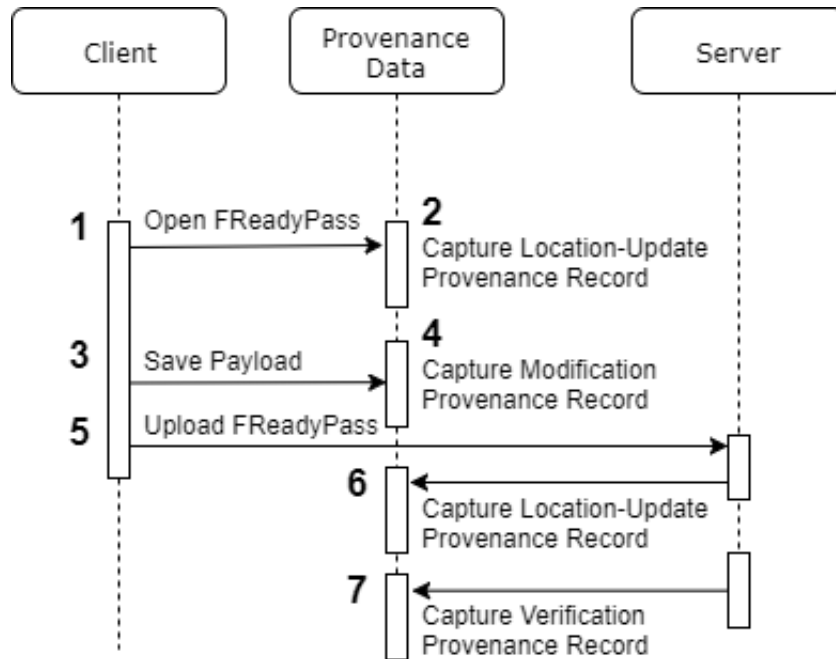


Figure 7.3.1: Capturing of different types of provenance records.

Step 1 and 2: A location-update provenance record is created when the client application opens a FReadyPass.

Step 3 and 4: A modification provenance record is created when the client application saves a payload after a modification has been made.

Step 5: After a client has finished with a FReadyPass, the client uploads the FReadyPass to the server.

Step 6: After a FReadyPass is uploaded to the server, the server creates a location-update provenance record.

Step 7: Finally, after a FReadyPass is uploaded to the server and a location-update provenance record has been captured, the server creates a verification provenance record.

Listing 7.1 is dissected in the discussions to follow in order to explain each section in the provenance record.

```
3     <Chain>
4         08861eb7321747b910570677b4de92da9181097059c370def97bca39f9f3bcfa
5     </Chain>
```

Listing 7.2: Provenance Record Chain Section

Provenance records are chained together to ensure the integrity of the provenance data as a whole. The process of chaining the records use cryptographic hashing and is discussed in greater detail in Section 7.5.1. The value of the chain field as shown in Listing 7.2 is the hash code of the previous provenance record. The chain value is used in conjunction with the digital signature to verify the integrity of the provenance data.

```
6     <Number>2</Number>
```

Listing 7.3: Provenance Record Number Section

Each provenance record is numbered, as shown in Listing 7.3. The number is used to indicate how many provenance records have been produced as well as the order in which the provenance records were produced. The timestamp can also be used to determine the order; however, using the timestamp requires more complex calculations, whereas a natural number makes it less complicated to determine the order of the provenance records.

```
7     <RecordType>Modification</RecordType>
```

Listing 7.4: Provenance Record Type Section

Listing 7.4 shows the provenance record type. A provenance record type may be either a modification, a location-update or a verification record. Each type of provenance record is discussed in more detail later in this chapter.

```
8      <Identity>
9      <OSDomain></OSDomain>
10     <OSUsername>Philip</OSUsername>
11     <UserAccount>ptrenwith@gmail.com</UserAccount>
12     <MachineName>PhilipT-PC</MachineName>
13     </Identity>
```

Listing 7.5: Provenance Record Identity Section

The user is already signed in before he/she is granted access to a FReadyPass. Thus, it is possible to log the user's account name. The client application also captures the name of the device being used to access the FReadyPass, as well as the domain name and user account of the Operating System being used. If the user account is not associated with a domain the OSDomain field remains empty as shown in Listing 7.5

```
14     <Location>
15     <IPAddress>137.215.0.66</IPAddress>
16     <Jurisdiction>ZA</Jurisdiction>
17     </Location>
```

Listing 7.6: Provenance Record Location Section

The user's location refers to the public IP address assigned to the user by the user's ISP. The user's IP address is determined and verified by the location-based access control mechanism. The IP address and the user's jurisdiction is logged as part of the provenance record as shown in Listing 7.6.

```
18     <Timestamp>1430251065774</Timestamp>
```

Listing 7.7: Provenance Record Timestamp Section

The time at which the provenance record was created is logged as a number and stored in milliseconds and shown in Listing 7.7.

```
19     <HashCode>
20     78c0c177eeaf438cccbda3aa3226fc4a18a98c2e672800a7ab781efb8e6c485
```

```
21      </HashCode>
```

Listing 7.8: Provenance Record Hash Code Section

The hash code as shown in Listing 7.8 is the SHA-256 hash code of the payload of the FReadyPass at the time the provenance record was created.

```
23      <DigitalSignature>
24      B1i+WDjJtu5Bdu7XxOYGWCVPztyMHccvr0Nu8b0V0Y/mxBu68FJaTN
25      mZBup4uVQ14pbGUruZvFHLk8VbHVHF+Ltc8sGguNWBOQFSn+buWG
26      apNiqaWwtiWuc7nO8BcdQOEIUqwH3xHslO500ZajoMETiw1qTqkpd7txZd
27      cB+8vA1Lpi9riIAxaWe8Ka79h4iVLwsGjAV9pXicOEO4gy4rwT58MLn4V3
28      TPQIwtXYghZsaOvP6E/DDtZbPKTy4GLCpTut94v4ZZZeDXCZpQwhJJ1
29      TFFf3tZ8avM2Yp3krNWkEE6FzSI3MPS/OVai1oQhCsoSC/q824PxjYyWIB
30      H7WEi1A==
31      </DigitalSignature>
```

Listing 7.9: Provenance Record Digital Signature Section

Finally, the digital signature shown in Listing 7.9 is the RSA signature of the data in the ProvenanceData section of the provenance record. The ProvenanceData section is visible from line 2 to 22 in Listing 7.1 and the digital signature of this section is visible from line 23 to 31 in Listing 7.1. Signing the provenance records require the storing and maintenance of the cryptographic keys used to sign and verify the digital signatures. Management of these cryptographic keys is discussed in the next section.

7.4 Maintaining cryptographic keys for verifying provenance data

The public key of an RSA key pair is used to verify any digital signature that has been produced using the private key of that key pair. Verifying a digital signature produced by a client serves the goal of verifying that the signature was produced by the entity that the server expects. Therefore, the verification process has little to no value if the server verifying the

signature does not know the identity of the client. To clarify, consider Alice uploads a FReadyPass to the server, but the server does not know Alice and does not have her public key. Thus, to verify the digital signature, the server would have to request Alice's public key. If this process is followed, what would stop a malicious agent from pretending to be Alice, producing forged provenance records or even altering legitimate provenance records and then uploading the altered FReadyPass to the server supplying with it the public key that would successfully verify the signatures of the altered provenance records? There would be no way for the server to determine if the FReadyPass came from Alice, or if it came from a malicious agent who has modified and re-signed the provenance records using another RSA key pair.

Therefore, it is important that Alice register her client application with the server and activates her public key. Activating a key entails marking the key as in-use and setting the expiration date to some point in the future, this can be six months, or a year, or even an indefinite period. How long a key is valid is determined by CSP policy. After a key is activated, it is stored on the server and linked to Alice's user account. However, Alice may have more than one device such as a laptop/desktop/mobile device, etc. Each device requires a private-public key pair for the signing and verification of provenance records. Private keys should not be shared between devices as it compromises security and heightens the risk of having the private key compromised. Thus, there should be one key pair per device.

The key pair is generated at the client, after which the private key is stored in an encrypted key-store on the client's device. The public key is uploaded to the server after Alice has been properly authenticated and a secure connection set up with the server. Once the public key is uploaded to the server, it has to be activated and linked to Alice's user account. A unique id is assigned to the device and stored locally on the device as well as with the server to identify the device. If the integrity of a key

pair ever comes into question, the key can be deactivated by setting the expiration date to a date in the past. Deactivating a key forces the client to generate a new key pair before being allowed to continue accessing FReadyPasses. Figure 7.4.1 illustrates the lifecycle of a key pair.

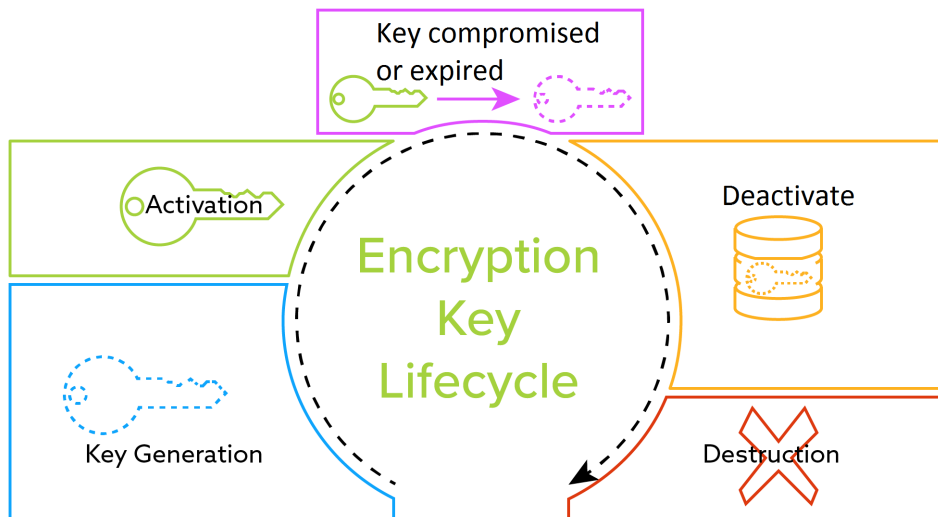


Figure 7.4.1: RSA Key pair lifecycle adopted from [Townsend, 2017]

The key lifecycle shown in Figure 7.4.1 consists of four stages and one condition. The four stages are:

1. Key Generation
2. Activation
3. Deactivation
4. Destruction

The condition: key compromised or expired triggers the deactivation stage.

The key generation takes place at the client as mentioned earlier. After the key pair has been generated, the public key is uploaded to the server at which point it is activated as a valid key. The key may remain

activated for a long period. The deactivation stage will only be executed once the condition is met that the key pair is either compromised or suspected to be compromised, or has reached its expiration date. At this point, the key pair is deactivated. The private key should be destroyed once it has been deactivated, considering that, it holds no further value. However, the public key may or may not be destroyed. Even though the key pair is no longer active and will not be used to sign any future provenance records, the public key can still be used to verify the digital signatures of provenance records that were signed using the corresponding historic private key.

By making use of digital signatures, this model provides cryptographic proof of the integrity of provenance data. Chaining provenance records together using a cryptographic hash, as well as using digital signatures makes this possible. The process is discussed in greater detail next.

7.5 Verify the integrity of provenance data

The process of verifying provenance data is quite simple. The digital signatures are used to confirm that the provenance records were not modified. Sometimes it may be necessary to verify the entire set of provenance records, and other times it could be sufficient to verify only a subset of provenance records.

A scenario that could require the verification of the entire set of provenance records is when a client of the CSP or a digital forensic investigation team request to see the entire history of the FReadyPass or its payload. In this scenario, the entire history of the FReadyPass has to be verifiable. Another scenario requiring the verification of a subset of provenance records is when a FReadyPass was downloaded to a client and is later uploaded to the server. The server only has to verify new provenance

records produced since the FReadyPass was downloaded. The next section discusses the verification of a subset of provenance records.

7.5.1 Verify the integrity of a subset of provenance records

Consider Alice (the client) downloads a FReadyPass from Bob (the server). At the time of the download, the FReadyPass contains 12 provenance records. Alice opens the FReadyPass to access and modify its payload multiple times, and when Alice uploads the FReadyPass back to Bob, the FReadyPass contains 16 provenance records. At this point, Bob only has to verify the integrity of provenance records 13,14,15 and 16.

There exist two concerns regarding the integrity of the provenance data.

- C1. A provenance record may have been altered.
- C2. A provenance record may have been deleted.

An attacker may alter or delete provenance records in an attempt to conceal access or modification to a FReadyPass payload.

To address concern C1, a provenance record is digitally signed to verify its integrity and either prove that the provenance record has not been altered, or detect if it was altered. The provenance record is signed by the entity that produces it. Thus, the client will sign provenance records it creates, and the server will sign provenance records that it creates. Formally stated:

$$\text{DigitalSignature}(n) \rightarrow \text{Record}(n)$$

Meaning the signature of provenance record n denoted as

DigitalSignature(*n*) verifies provenance record *n* denoted as *Record*(*n*). Listing 7.10 shows a visual representation of a signed provenance record. The *DigitalSignature* visible from line 23 to 31 is the digital signature computed over the data in the *ProvenanceData* section, which is visible from line 2 to 22.

```

1  <ProvenanceRecord>
2    <ProvenanceData>
3      <Chain>
4        08861eb7321747b910570677b4de92da9181097059c370def97bca39f9f3bcfa
5      </Chain>
6      <Number>2</Number>
7      <RecordType>Modification</RecordType>
8      <Identity>
9        <OSDomain></OSDomain>
10       <OSUsername>Philip</OSUsername>
11       <UserAccount>ptrenwith@gmail.com</UserAccount>
12       <MachineName>PhilipT-PC</MachineName>
13     </Identity>
14     <Location>
15       <IPAddress>137.215.0.66</IPAddress>
16       <Jurisdiction>ZA</Jurisdiction>
17     </Location>
18     <Timestamp>1430251065774</Timestamp>
19     <HashCode>
20       78c0c177eeaf438cccbda3aa3226fc4a18a98c2e672800a7ab781efb8e6c485
21     </HashCode>
22   </ProvenanceData>
23   <DigitalSignature>
24     B1i+WDjJtu5Bdu7XxOYGWCVpztyMHccvr0Nu8b0V0Y/mxBu68FJaTN
25     mZBup4uVQ14pbGUruZvFHLk8VbHVHF+Ltc8sGguNWBOQFSn+buWG
26     apNiqawwtiWuc7nO8BcdQOElUqwH3xHslO500ZajoMETiw1qTqkpd7txZd
27     cB+8vA1Lpi9riIAxaWe8Ka79h4iVLwsGjAV9pXicOEO4gy4rwT58MLn4V3
28     TPQIwtXYghZsaOvP6E/DDtZbPKTy4GLCpTut94v4ZZZeDXCZpQwhJJ1
29     TFFf3tZ8avM2Yp3krNWkEE6FzSI3MPS/OVai1oQhCsoSC/q824PxjYyWIB
30     H7WEi1A==
31   </DigitalSignature>

```

32 `</ProvenanceRecord>`

Listing 7.10: Signed Provenance Record

To address concern C2, the system needs to provide cryptographic proof of the integrity of all provenance data, provenance records are chained together to detect if a provenance record has been deleted. When a new provenance record denoted as record n is being created, a SHA-256 hash of the ProvenanceRecord section of provenance record $n - 1$ is computed. The hash code of provenance record $n - 1$ is included as the value of the Chain field of provenance record n , shown in Listing 7.11. Line 1 to 31 is record $n-1$, which is hashed using SHA-256 and the hash value is included as the chain value visible on line 35.

Formally stated:

$$Chain(n) = Hash(Record(n - 1) \text{ AND } DigitalSignature(n - 1))$$

```

1  <ProvenanceRecord>
2    <ProvenanceData>
3      <Chain>
4      </Chain>
5      <Number>1</Number>
6      <RecordType>Location-Update</RecordType>
7      <Identity>
8        <OSDomain></OSDomain>
9        <OSUsername>Philip</OSUsername>
10       <UserAccount>ptrenwith@gmail.com</UserAccount>
11       <MachineName>PhilipT-PC</MachineName>
12     </Identity>
13     <Location>
14       <IPAddress>137.215.0.66</IPAddress>
15       <Jurisdiction>ZA</Jurisdiction>
16     </Location>
17     <Timestamp>1430239541326</Timestamp>
18     <HashCode>
19       98f9f37910af438csdsfbra3ga32g6841gha18a98c2e6720a7ab7dff8e6ch653

```

```
20     </HashCode>
21 </ProvenanceData>
22 <DigitalSignature>
23     bkY7LyXMcRTrIZg+20yMZL4LGGqA85o1EF4IHsaEZWw7tbsY26OT4r
24     QyFhueFtgLO933m8slb/Fcw+EHuZl/5Wt7ewJIY7DJ9NkCrJ8T+3YWh
25     nBN3dAKVOaib1U5FeFcicyEJwtkukx/ZqC86JHyX5XmzfeooAnuHf4Sx
26     AWpoAuMdBahil60yfe5ZaTEcsWg/EbA8UVPvFZlqcIEC84U0OswJ5ID
27     /5zFO5VtIehYOS9GuPov0ffhDiYC0aXmFv8xstXM2zVAkIodFkEC7Ry
28     1qPQvK6rtvy4A8SoHvu9uW44AfXGCMK0IU1L3PrZI6gJ64vb2NpZhS3
29     FiMF/N3swHg==
30 </DigitalSignature>
31 </ProvenanceRecord>
32 <ProvenanceRecord>
33     <ProvenanceData>
34         <Chain>
35             08861eb7321747b910570677b4de92da9181097059c370def97bca39f9f3bcfa
36         </Chain>
37         <Number>2</Number>
38         <RecordType>Modification</RecordType>
39         <Identity>
40             <OSDomain></OSDomain>
41             <OSUsername>Philip</OSUsername>
42             <UserAccount>ptrenwith@gmail.com</UserAccount>
43             <MachineName>PhilipT-PC</MachineName>
44         </Identity>
45         <Location>
46             <IPAddress>137.215.0.66</IPAddress>
47             <Jurisdiction>ZA</Jurisdiction>
48         </Location>
49         <Timestamp>1430251065774</Timestamp>
50         <HashCode>
51             78c0c177eeaf438cccbda3aa3226fc4a18a98c2e672800a7ab781efb8e6c485
52         </HashCode>
53     </ProvenanceData>
54 </DigitalSignature>
55     B1i+WDjJtu5Bdu7XxOYGWCVPztyMHccvr0Nu8b0V0Y/mxBu68FJaTN
56     mZBup4uVQ14pbGUruZvFHLk8VbHVHF+Ltc8sGguNWBOQFSn+buWG
57     apNiqawwtiWuc7nO8BcdQOEIUqwH3xHslO500ZajoMETiw1qTqkpd7txZd
58     cB+8vA1Lpi9riIAxaWe8Ka79h4iVLwsGjAV9pXicOEO4gy4rwT58MLn4V3
```

```

59     TPQIwtXYghZsaOvP6E/DDtZbPKTy4GLCpTut94v4ZZZeDXCZpQwhJJ1
60     TFFf3tZ8avM2Yp3krNWkEE6FzSI3MPS/OVai1oQhCsoSC/q824PxjYyWIB
61     H7WEi1A==
62     </DigitalSignature>
63 </ProvenanceRecord>

```

Listing 7.11: Chained Provenance Records

Thus, chain value n is equal to the SHA-256 Hash code of provenance record $n - 1$.

Chaining the provenance records together ensures that a forged provenance record cannot be inserted in the chain, a valid provenance record cannot be removed, and no provenance records can be modified without being detected.

Consider a malicious agent modifies provenance record number 2 shown in Listing 7.11. When the system verifies the integrity of the provenance records, the modification will be detected because verifying the signature of provenance record 2, $DigitalSignature(2) \rightarrow Record(2)$ will fail. Furthermore, consider the malicious agent removes provenance record number 2 altogether. The removal will be detected upon verification because the chain value in provenance record 3 does not equal the hash of provenance record 1:

$$Chain(3) \neq Hash(Record(1) \text{ AND } DigitalSignature(1)).$$

Let us assume there exists a provenance record number 1, upon verification of the chain, the SHA-256 hash of provenance record 1: $Hash(Record(1) \text{ AND } DigitalSignature(1))$ will not match the hash code stored in $Chain(3)$. If the malicious agent removes or modifies $Chain(3)$, the signature verification: $DigitalSignature(3) \rightarrow Record(3)$ fails, therefore these verification rules satisfies concerns C1 and C2.

However, there is one risk that has not yet been addressed. The most

recent provenance record, provenance record n can be deleted and will go undetected. If one deletes provenance record n , it becomes possible to delete provenance record $n - 1$ and again it will go undetected. To ensure that the most recent provenance record, provenance record n cannot be deleted without being detected, the author creates an index of the provenance chain. It is not necessary to create an index for all of the provenance records considering the provenance records is chained together. The index has merely to indicate how many provenance records there are, and what the value of n is. The author proposes the addition of an index record referred to as Provenance Record Zero shown in Listing 7.12.

```

1 <ProvenanceRecord>
2   <Index>
3     <LastRecordNumber>2</LastRecordNumber>
4     <LastRecordHashCode>
5       78c0c177eeaf438cccbda3aa3226fc4a18a98c2e672800a7ab781efb8e6c485
6     </LastRecordHashCode>
7   </Index>
8   <DigitalSignature>
9     HBMO+PflA1g2M+taApyQbGtnaM8gepMssB6ZcnOjmM0kZSw1V9Pa
10    BemN7PhpM/JOQ/v5fzKO+IA74U/50AgsVf65xMrEXPQujNLvrIznDn
11    b3WVGbptP4VTEJgE2KmDcuINifb70MZwBrVG3Zrkel5OZFR+/1PB
12    rU9OP3XuAKdiY==
13  </DigitalSignature>
14 </ProvenanceRecord>

```

Listing 7.12: Provenance Record Zero

The LastRecordNumber section shown in Listing 7.12 on line 3 refers to the value in the Number section of the most recent provenance record, Provenance Record n .

```

4   <LastRecordHashCode>
5     78c0c177eeaf438cccbda3aa3226fc4a18a98c2e672800a7ab781efb8e6c485
6   </LastRecordHashCode>

```

Listing 7.13: Provenance Record Zero

The value of `LastRecordHashCode` is the hash code of Provenance Record n . Adding Provenance Record Zero to the system in addition to what has already been developed allows the system to know the length of the chain of provenance records that has to be verified. The signature of Provenance Record n can be used to verify the integrity of the Record section of provenance record n , and the `LastRecordHashCode` as shown in Listing 7.13 can be used to verify the integrity of the digital signature of Provenance Record n .

The digital signature of Provenance Record Zero shown in Listing 7.12 from line 8 to 13 is the signature of record zero. Thus, the integrity of provenance record zero can be verified. If a malicious attacker deletes Provenance Record n , the system can detect the deletion because the hash code of Record $n - 1$ will not match the hash code saved in Provenance Record Zero. The hash code of one provenance record will never match that of another provenance record in its chain because of the Record Number field that will never have the same value. The timestamp field should never match either, however, if the network time protocol fails and the time on a machine is incorrectly set it is possible for a timestamp to match a previously produced provenance record's timestamp.

If a malicious attacker deletes Provenance Record Zero, the integrity of the Provenance data is compromised, and the system can automatically detect the compromise because Provenance Record Zero should always be present. In the next section, the author discusses the verification of an entire chain of provenance data, taking into consideration that the keys used to sign the provenance records may have changed.

7.5.2 Verifying the integrity of the entire set of provenance records

Because each client application has its own individual set of RSA keys, it becomes a challenge whenever the integrity of historic provenance data has to be verified, because the client's keys may have changed over time. To address this issue, the server verifies provenance records as soon as a `FReadyPass` is uploaded to the server. The server verifies all the provenance records generated by the client using the public key associated with that client's device. Once all the provenance records produced by the client has been verified, the server creates a location update provenance record, followed by a verification-type provenance record, referred to as a verification record shown in Listing 7.14. The verification record's position in the chain indicates that all provenance records preceding it has been verified by the server. The verification record contains a verification field shown in Listing 7.14 on line 8, indicating the validity of all the provenance records it has verified. The server signs the verification record with its own signature before the `FReadyPass` is stored.

```

1 <VerificationRecord>
2   <Record>
3     <Chain>
4       08861eb7321747b910570677b4de92da9181097059c370def97bca39f9f3bcfa
5     </Chain>
6     <Number>6</Number>
7     <Timestamp>1430251065774</Timestamp>
8     <Verification>All Records Valid</Verification>
9   </Record>
10  <DigitalSignature>
11    HBMO+PflA1g2M+taApyQbGtnaM8gepMssB6ZcnOjmM0kZSw1V9PaBe
12    mN7PhpM/JOQ/v5fzKO+IA74U/50AgsVf65xMrEXPQujNLvrIznDmb3W
13    VGbptP4VTEJgE2KmDcuINifb70MZwBrVG3Zrkel5OZFR+/1PBrU9OP3
14    XuAKdiY==
15  </DigitalSignature>
16 </VerificationRecord>

```

Listing 7.14: Verification Record

To verify historic provenance data, the server verifies the value of the chain:

$$Chain(n) = Hash(Record(n - 1) \text{ AND } DigitalSignature(n - 1))$$

The server has to verify that the hash value stored in $Chain(n)$ is equal to the hash value of provenance record $n - 1$.

The server also verifies the signatures of the verification records using the server's public key. The possibility exists that the signature of provenance records signed by a client device cannot be verified after the client device key has expired and been changed, however, it is still possible to verify the integrity of provenance data because the provenance records are chained. It is simply impractical to store all the public keys ever used by all the client devices because these may be legion. Therefore, only active keys are stored.

If an attacker modifies a single provenance record, it can be detected. If the attacker also modifies the hash codes in the provenance records following that provenance record, it is possible to conceal exactly where the falsification starts, but it is possible to determine whether or not the section of provenance records from one verification record to the next is valid.

7.6 Conclusion

The process of signing and chaining provenance records together provides cryptographic proof of the integrity of the provenance data. Thus, this system provides cryptographic proof of the integrity of a FReadyPass and its history.

The solution to verifying provenance data presented in this chapter

is secure as long as the private keys used to sign provenance records remain secure. Therefore, it is essential that proper security protocols are implemented and adhered to, in order to maintain encryption keys both at the server and at the client to ensure that the integrity of the system cannot be called into question.

Having discussed the design and implementation of provenance data and shown the process of verifying the integrity of provenance data, the author can now proceed to discuss the process of uploading and downloading `FReadyPasses`.

Chapter 8

Accessing a FReadyPass in and out of the cloud

8.1 Introduction

In the previous chapters, the author discussed the design of the FReadyPass, its provenance data, and the access control mechanism. In this chapter, the author discusses the process of uploading and downloading a FReadyPass by making use of a hypothetical scenario. In this chapter, the author specifically looks at how the different building blocks fit together. Finally, the author looks at the optimal CSP network layout required to service this model.

8.2 Securing a FReadyPass at rest and in transit

Consider ACR4 stated in Section 5.3.5. If a user's connection to the server is being routed through a remote site R by making use of a tunneling service, and the FReadyPass is not encrypted, it is possible for the encapsulated data stored within the FReadyPass to be read at the remote site, as it is passed from the source to the destination. Therefore,

the FReadyPass should be secured to ensure that R in ACR4 does not get access to the data. One possible solution is to use end-to-end encryption[Lu and Sundareshan, 1989].

In this work, the author implements the AES-256 encryption algorithm for encrypting FReadyPasses at rest. “at Rest” refers to a FReadyPass that is stored on a secondary storage device and not being copied or transmitted over a network at the time. If a FReadyPass is being downloaded or uploaded, it is referred to as a FReadyPass “in transit”. The FReadyPass remains encrypted while in transit and is only decrypted at the client after the client has been authenticated. The FReadyPass is also decrypted at the server when the FReadyPass is uploaded for the server to create a location-update record. After the location-update record has been written the FReadyPass is re-encrypted before it is stored. The RSA key exchange algorithm is used to communicate the encryption key from the server to the client to decrypt the FReadyPass. The next section discusses the client application design.

8.3 Client Application

The client application is digitally signed before it is made available to the CSP’s clients for use. This is accepted practice in application development to assure the users of an application that the application is safe and has not been modified after being released by the original host[Comodo, 2017]. Signing the application provides the user with a certain level of comfort regarding the client application’s integrity.

Throughout this dissertation, it has been discussed how the client application has to establish a connection with the server application to verify the client’s IP address and determine the client’s location. Thus, ensuring that only users within an allowed jurisdiction may gain access to a FReadyPass. It is also a requirement that the FReadyPass remains

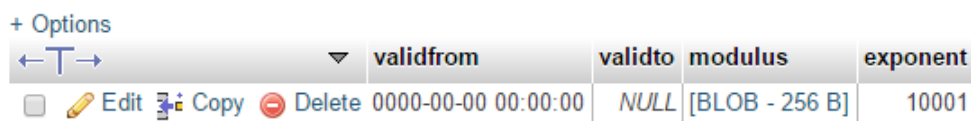
encrypted when it is transferred to the client, thus requiring the exchange of encryption keys.

As discussed in section 7.4 the client application has its own RSA keys, and the public key has to be activated and stored on the server. The RSA private key should be kept secure in a key-store. The client application uses its private key stored in the key-store to sign provenance records. If a client's keys are changed, the new public key has to be activated with the server. The client's public key is used to verify the data digitally signed by the client. The maintenance of RSA keys is discussed in the next section.

8.4 Maintaining RSA Keys for signature and verification

It is necessary to keep the public keys used by the server even after the keys have changed because the server keys are used to sign verification records. If the need arises to verify the full set of provenance records all the server keys used to sign these provenance records will be required to verify the provenance records. When it becomes necessary for the server's RSA keys to be changed, the server saves its old public key in the database. It is not necessary to keep every public key used by client applications, however, by keeping all the server's public keys, it is possible to verify the signatures of verification records. It is not absolutely necessary to have the ability to verify the signatures of verification records because the integrity of the chain can be verified by simply hashing each provenance record and verifying that the hash matches the chain value of the next provenance record. However, if that chain is compromised, it is only possible to determine the section between verification records that was compromised by verifying signatures of the verification records. Therefore, the server's public keys are kept on record even after a key

has expired. The database table used to store the public keys are shown in Figure 8.4.1.



	validfrom	validto	modulus	exponent
<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	0000-00-00 00:00:00	NULL	[BLOB - 256 B]	10001

Figure 8.4.1: Server RSA public key history table

An expired public key can be used to verify any of the digital signatures generated by the server during the time that the key was active. The duration that a key is active is indicated in the database by the “validfrom” and “validto” fields. If the value of the “validto” field is “NULL” it means the key is still valid. In the next section, the author discusses the life-cycle of a FReadyPass by making use of scenario describing a hypothetical use-case.

8.5 FReadyPass life-cycle

The life-cycle of a FReadyPass is discussed through the use of a use-case. In this use case, there exists 3 actors: Alice, Bob, and the CSP. In this use case, the following events take place:

Step 1: Alice compiles a document describing a business plan.

Step 2: The document is encapsulated in a FReadyPass and stored with a CSP.

Step 3: Alice asks Bob to review the business plan and provide feedback.

Step 4: Bob downloads the FReadyPass to review the business plan and provide his feedback.

Step 5: After Bob finishes his review of the document he saves the changes to the document, the document is encapsulated in the FReadyPass, and the FReadyPass is recompiled and uploaded back to the CSP where Alice can download it to see Bob’s feedback.

The process of constructing a FReadyPass has already been discussed. Therefore, the focus of this use-case is on step 4 and step 5. The process of downloading the FReadyPass is discussed next.

8.5.1 Downloading a FReadyPass

When Bob downloads the FReadyPass to review Alice's business plan and provide feedback, it is possible for that FReadyPass to be taken outside the jurisdiction of the controlling CSP. The FReadyPass can be downloaded at a location that is allowed to access the FReadyPass and thereafter copied to a secondary storage device and transferred. It is also possible that a local site that is allowed access to the FReadyPass is being controlled remotely, making it possible to copy the FReadyPass over a network connection to another device, possibly outside the allowed jurisdiction. Hence, it is necessary to determine Bob's jurisdiction when he downloads the FReadyPass, as well as when he attempts to open the FReadyPass. If Bob attempts to open the FReadyPass in the same session he used to download the FReadyPass, it is not necessary to determine his jurisdiction again as it has already been done in that session.

Figure 8.5.1 shows the process where Bob attempts to open the FReadyPass in a newly established session.

Step 1: Bob's client application sends a request to the server to open a FReadyPass. The client provides the FReadyPass's ID, and this is used to lookup the allowed jurisdiction where the FReadyPass may be opened from. The client also provides its application ID that is necessary to look up the client's public key that is used to encrypt the AES key when it is sent to the client.

Step 2: The server looks up Bob's jurisdiction, using Bob's IP address. The system then determines if Bob may access the requested FReadyPass from his jurisdiction.

Step 3: The client awaits the notification from the server.

Step 4 and 5: The system notifies the client whether or not it has access to open the FReadyPass.

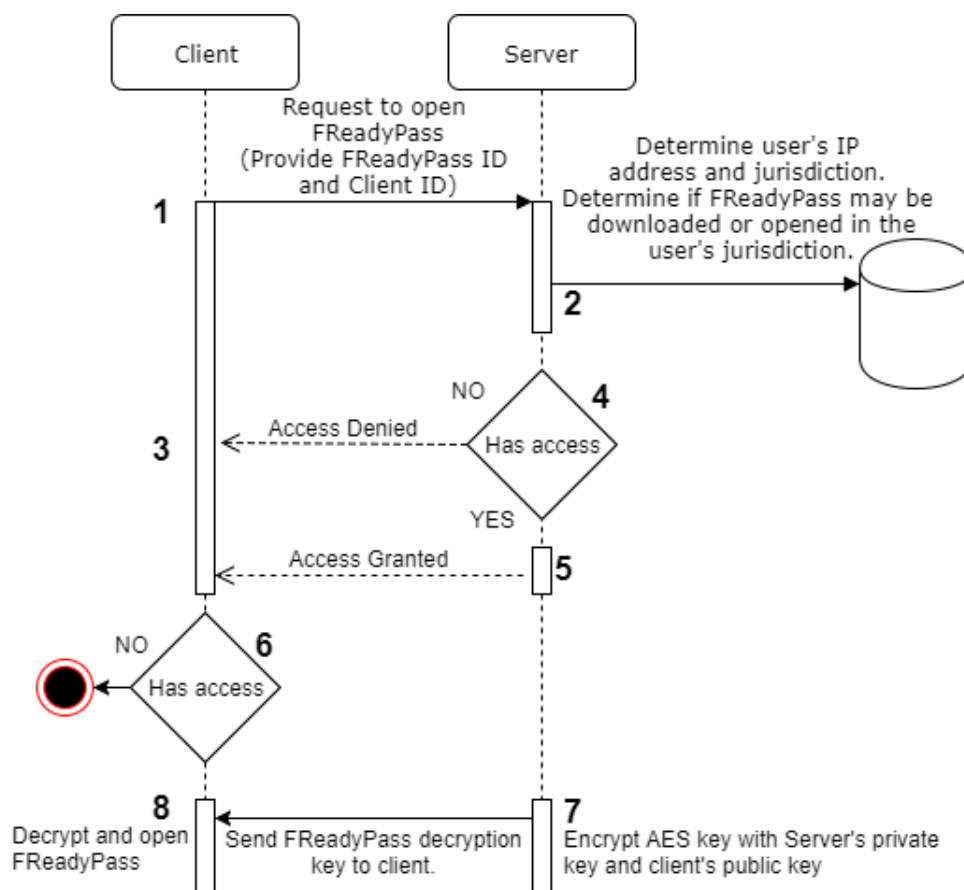


Figure 8.5.1: FReadyPass Download Process

Step 6: If the access request is denied, step 8 will not be executed by the client. If the access request is granted, the server executes step 7 after which the client executes step 8.

Step 7: The server encrypts the AES key needed to decrypt the FReadyPass and sends the key to the client.

Step 8: The client awaits the FReadyPass's decryption key to be downloaded, and then the client decrypts the AES encryption key required to decrypt the FReadyPass, after which the client decrypts the FReadyPass

and opens it for Bob to view and modify.

The FReadyPass decryption process is discussed in greater detail in the next section.

8.5.2 Decrypting a FReadyPass

The FReadyPass that was just downloaded by Bob is encrypted with the AES-256 cryptographic algorithm. The decryption key that is required to decrypt the FReadyPass is exchanged between the server and the client using the RSA asymmetric encryption scheme discussed in Section 3.8.

It is possible for anyone to copy the FReadyPass after it has been downloaded. While it is encrypted, it does not matter that it can be copied. Once the data is decrypted, if it is copied, it can be moved out of the jurisdiction, defeating the purpose of this model.

To ensure that the decrypted payload is not copied by conventional manners the client application creates a secure folder on disc from where data cannot be copied. To implement this, the client program monitors the Windows clipboard. If any event written to the clipboard originates from within the secure folder, the event is removed from the clipboard. The implementation is discussed in detail in chapter 9.

The implementation of a secure folder ensures that Bob cannot copy the decrypted file in a conventional manner. However, it does not prevent Bob from opening the file and saving a second copy of the data to another location. Even though this results in a second digital object and the original digital object is still being tracked through the FReadyPass, it presents the possibility of sensitive information to be compromised. Hence, there exists a need for a trusted relationship between the owner of the digital file, in this case, Alice, the CSP and those users that are granted access to the FReadyPass, such as Bob. One possible solution to

mitigate the risk of data being leaked is forensic marking. When forensic marking is applied the unique ID of a user, in this case, Bob's account name is added to the data file as an invisible watermark when the data is made available to the user. Hence, if "illegal" copies of data are made, it is possible upon discovering the data to determine where the data leak occurred. Thus, appropriate action can be taken against the guilty party.

Once the FReadyPass has been decrypted, the client application opens the FReadyPass and captures a location-update provenance record as shown in Figure 7.3.1. It is now possible for the payload to be extracted into the secure folder and for Bob to access the payload. Once he is done and saves the changes to the payload a modification provenance record is captured by the client application indicating that the payload has been altered, after which the modification provenance record is added to the provenance chain and the FReadyPass is re-encrypted. If Bob so chooses he can now upload the FReadyPass to the server or elect to do that later in the event he wishes to reassess the payload at a later stage. The process of uploading a FReadyPass is discussed in greater detail in Chapter 9. After Bob has uploaded the payload to the server, the server opens it up and verifies the integrity of the newly produced provenance records, after which the server captures a location update provenance record followed by a verification record. In the next section, the author discusses the layout of the Cloud Service Provider's network.

8.6 Optimal Cloud Service Provider Network

This section shows a suggested layout of a CSP network, taking the FReadyPass system into account.

A CSP's internal network is not much different from any organizations network that provides a public service accessible through the internet. The network contains some servers providing some services and some

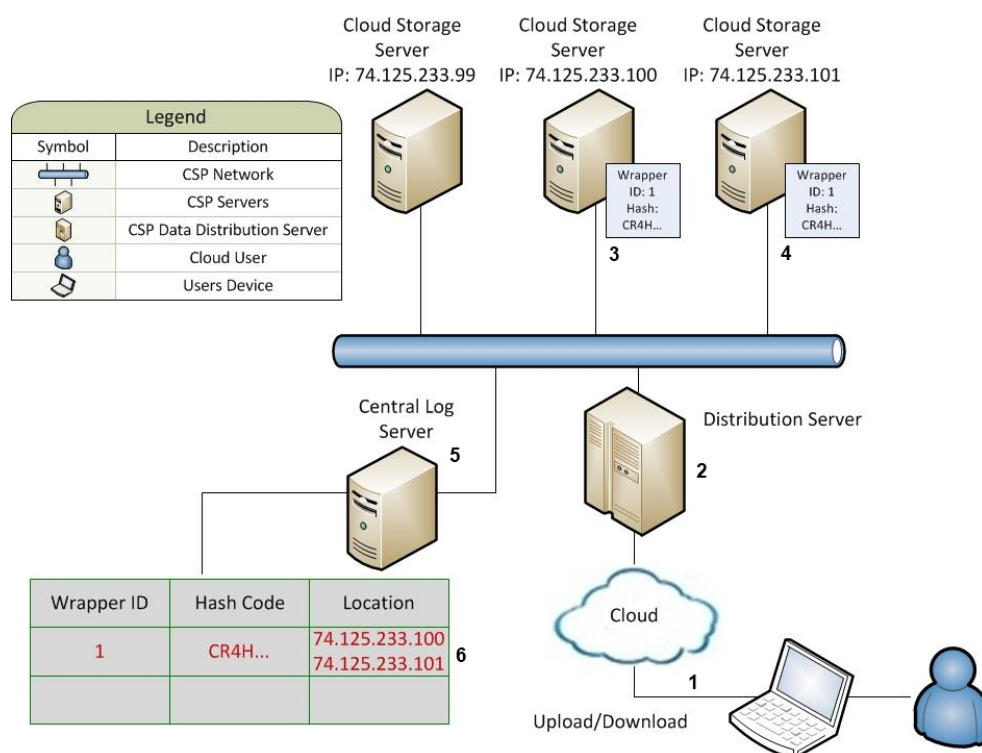


Figure 8.6.1: Example CSP network

routers and other network devices directing traffic. This section discusses the addition of a central log server to a CSP’s network to manage the location information of FReadyPasses. Figure 8.6.1 shows an example of a network to demonstrate how the FReadyPasses are stored and made available to clients. Certain areas of Figure 8.6.1 are numbered to allow the author to describe the components in the figure in more detail. The numbers in the figure refer either to a step taken by the user or the CSP or to a component of the CSP’s network. Because not all the numbers are steps, the author refers to them as numbers in the discussion rather than steps.

Number 1: A cloud user accesses the CSP using any device with an internet connection. To access any FReadyPass, the user needs to download the client application.

Number 2: To simplify the example the author introduces a Distribution Server. This server retrieves a requested FReadyPass from the storage servers and also manages the uploading of new payloads that should be encapsulated as well as existing FReadyPasses.

Number 3 and 4: Shows a FReadyPass stored on the cloud storage servers. There is a copy of the FReadyPass on another storage server. For this example, the assumption is made that the backup and copies of data are managed by the Distribution Server.

Number 5: The Central log server is added by the author to manage the lookup of location data and other related metadata. Thus, filling the role of an index server and search engine.

Number 6: A database record for each FReadyPass is written to the central database. The record contains the unique ID of the FReadyPass as well as the hash-code of the FReadyPass and the IP addresses of the servers where the FReadyPass can be found. When the FReadyPass is downloaded to a client, a database record is written to the database indicating the location and identity of the client that downloaded the FReadyPass. There may be more than one server containing the FReadyPass considering cloud service providers often distribute content to multiple servers to improve the availability of data and provide backups. However, it is not necessary for the proposed FReadyPass system to keep track of multiple copies of a FReadyPass considering backup of data is not a new concept for a CSP, and these processes should already be in place.

8.7 Conclusion

In this chapter, the author presented the proposal for tracking data through the cloud and securing the data. The proposal includes the design of a FReadyPass, a server-side application, a client-side application and a secure communication protocol between the two applications. The FReadyPass is encrypted using the AES-256 cryptographic algorithm and the key exchange algorithm used is the RSA asymmetric cryptographic algorithm.

The author proposes the use of a location based access control mechanism and propose the design of an access control mechanism, relying on reverse IP address lookups and an anti-spoofing technique to determine the user's location.

Both the server and client applications capture provenance data for the FReadyPass. In addition to capturing provenance data, the server application monitors the location of FReadyPasses and logs the location and metadata in a central database. The central database provides easy access to digital forensic investigators to determine the location of a FReadyPass as well as to retrieve any relevant log data required in an investigation.

This chapter only discusses the theoretical design of the model without taking the practical implementation into account. In the next chapter, the practical implementation is taken into account. The author develops a proof of concept prototype and discusses the techniques used during this implementation.

Part IV

Prototype

Chapter 9

Prototype

9.1 Introduction

In this chapter, the practical implementation of the model developed in part III is discussed. This chapter is intended for software developers and technical personnel. The discussion in this chapter considers technical challenges. The prototype proposed by the author in this chapter is not the only possible technique for implementing the model. It is one possibility that the author uses to demonstrate the successful implementation and execution of the model.

This chapter discusses six main building blocks of the model.

1. Prototype Server Setup.
2. The location-based access control mechanism.
3. The client application.
4. Downloading a FReadyPass.
5. Uploading a FReadyPass.
6. Chaining provenance records and verifying its integrity.

The prototype server setup is discussed in the next section.

9.2 Prototype Server Setup

In this section, the author briefly discusses the software configuration that is used to support the prototype server.

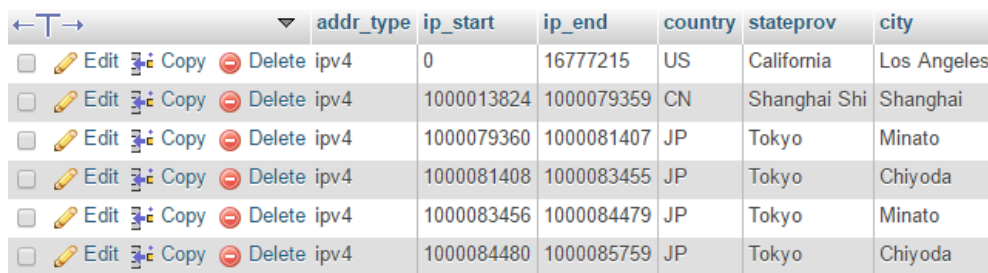
The server used to support the prototype is running on a Digital Ocean droplet server running the Ubuntu 14.04 operating system supported by a MySQL 5.5 Database server. To install additional software such as Java, etc. the author connects to the Ubuntu virtual server using an SSH client. The article at [DigitalOcean, 2016] discuss how to setup SSH on a Linux Server. The server application is developed with the Java programming language.

In the next section of this chapter, the author discusses the implementation of location-based access control.

9.3 Location Based Access Control Mechanism

In the design of the model, the author suggests the use of the IANA registrar databases to determine the jurisdiction of an IP address. IANA is not the only source for Geolocation lookups. DB-IP [2016] provides Geolocation lookups of IP addresses to cities. The IP address database is saved to the MySQL Database on the Linux Server. The IP address table looks as follows:

The data in the database indicates the type of address, IP version 4 or version 6. The ISP is assigned a block of IP addresses. The database shows the first IP address in the block and the last IP address in the block represented by a natural number. The country, state, and city where the ISP is located to whom this block is assigned is also provided. The IP address information is converted to natural numbers before it



	addr_type	ip_start	ip_end	country	stateprov	city
<input type="checkbox"/> Edit Copy Delete	ipv4	0	16777215	US	California	Los Angeles
<input type="checkbox"/> Edit Copy Delete	ipv4	1000013824	1000079359	CN	Shanghai Shi	Shanghai
<input type="checkbox"/> Edit Copy Delete	ipv4	1000079360	1000081407	JP	Tokyo	Minato
<input type="checkbox"/> Edit Copy Delete	ipv4	1000081408	1000083455	JP	Tokyo	Chiyoda
<input type="checkbox"/> Edit Copy Delete	ipv4	1000083456	1000084479	JP	Tokyo	Minato
<input type="checkbox"/> Edit Copy Delete	ipv4	1000084480	1000085759	JP	Tokyo	Chiyoda

Figure 9.3.1: IP Address Lookup Table

is stored in the database. Converting IP addresses to natural numbers makes address lookups much faster and easier to execute.

When a user connects to the server using the client application, the user has to log in using his/her credentials. The client application proceeds to initialize port forwarding on the router to receive the pending TCP socket connection established from the server to the client application. The author makes use of the Port Mapper application available at: <https://sourceforge.net/projects/upnp-portmapper/> [2016] to setup port forwarding. The Port Mapper application allows the client application to dynamically add and remove port forwarding rules on the router without requiring credentials to access the router.

Next, the client determines the IP address assigned to it by the ISP by making use of a lookup service such as <http://checkip.dyndns.org> [2016]. The client sends the assigned IP address to the server's REST-full interface to initiate the socket connection from the Server. Figure 9.3.2 shows this process. The client determines its IP address and transmits it to the server to perform a Geolocation lookup. The server responds by informing the client that it is in Randburg, South-Africa. Finally, the client requests the server to initiate a 3-way handshake with the REST call to <http://178.62.89.26:8010/auth/authenticate>.

Upon receiving the socket connection request from the server, the

```
Activity Log:
Starting Server...
looking up public ip address...
Public IP-Address: 169.0.100.253
GetIPInformation: json: {"ipaddress":"169.0.100.253"}
IP information: IP-Address: 169.0.100.253, City:
Randburg, Country: ZA
Authenticating...
Setting Upnp port forwarding router 60000 >> 65000...
>> URI: http://178.62.89.26:8010/auth/authenticate
params: {"ipaddress":"169.0.100.253"}
Connection received...
Generating Client RSA Key Pairs
Secure folder added: F:\STUDIES\Prototype
\SecureFolder
```

Figure 9.3.2: Client Application Log

server knows that the client's IP address is what it says it is, and from the Lookup database shown in Figure 9.3.1, the server knows where the client is, thus completing the location-based access control. The client may now proceed to request a FReadyPass for downloading or opening an already downloaded FReadyPass. In the next section, the development of the client application is discussed.

9.4 Development of the client application

The client application written for the prototype is written using both the Java and C# programming languages and is developed to run on a Windows Operating System. The client application can be downloaded from the server. Figure 9.4.1 shows a screenshot of part of the client application. From this application, the user can create a new FReadyPass, open an existing FReadyPass located on the local machine, and upload a FReadyPass to the server.

The client and server communicate with each other using REST calls.

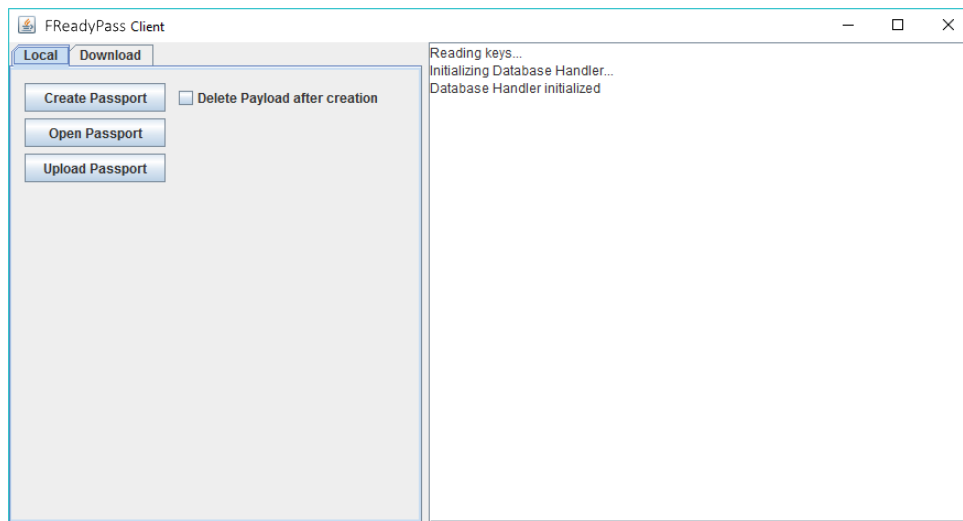


Figure 9.4.1: Client Application

These calls are unencrypted however, the uploading and downloading of FReadyPasses are done using the SFTP protocol. Furthermore, the FReadyPass is encrypted using AES, and the AES key is encrypted as well.

The client stores its RSA private key and certificate including the public key in a PKCS12 file format. The passphrase to access this file is hard-coded into the client application source code. To protect the source code from being reversed engineered or decompiled it is obfuscated and after that converted to native code. The identity of the client remains secure as long as the Keystore file remains secure. If the integrity of the client comes into question, the client can simply generate a new RSA key pair.

On the download tab, the user can search the server for a specific FReadyPass. The search results for any user query does not filter based on the user's location. However, when the user requests a FReadyPass for download, the client and server will negotiate through the process of location-based access control and determine whether or not the client

may access the FReadyPass from his/her current location. The user's location is not the only parameter that is taken into consideration when accessing a FReadyPass. The user or his/her user group also needs to be granted permission to access a FReadyPass from the FReadyPass's owner.

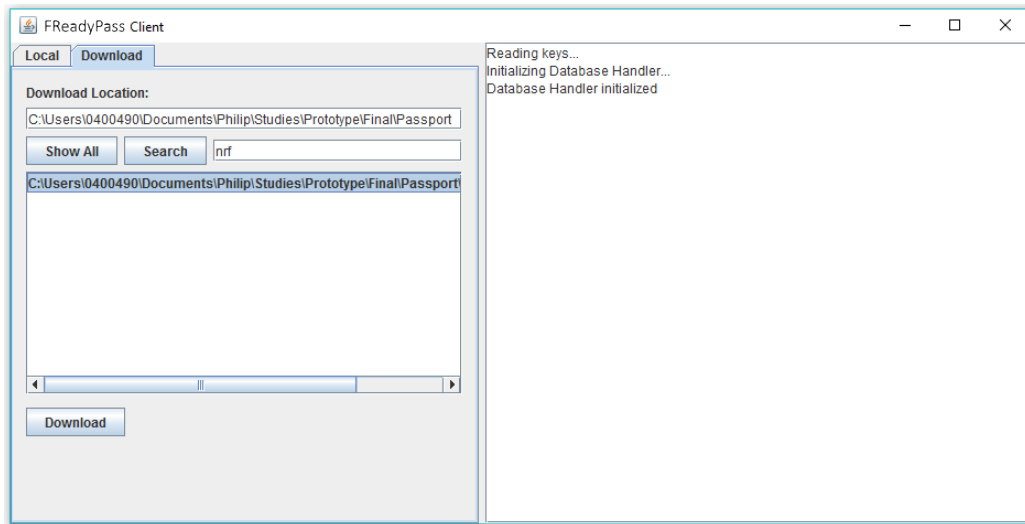


Figure 9.4.2: Searching for a FReadyPass

9.5 Downloading a FReadyPass

Before a FReadyPass is downloaded, the client application creates a secure folder into which the FReadyPass is downloaded. The secure folder has a few characteristics common folders don't have. Files located in the secure folder, cannot be copied, cut or sent to other locations. This functionality is necessary to make it harder for agents to copy sensitive data once the FReadyPass has been decrypted. Knowing the location of the decrypted data gives the client application the ability to control access to the information.

To stop the data in the secure folder from being copied or cut on windows based operating systems requires the application to monitor the

clipboard; this part of the client is developed as a C# DLL. If any file in the secure folder shows up on the clipboard, it is immediately removed from the clipboard thus making it impossible to copy the file by conventional means.

To monitor the clipboard requires the client application to import the User32.dll and override the WndProc method to interact with the clipboard. The source code to do this is shown in Appendix A, Figure A.0.1 and Figure A.0.2. Other techniques can be used to copy data; these techniques are discussed in chapter 11.

Once the user has successfully logged in, and the location-based access request has been completed. The server sends its RSA public key to the client application.

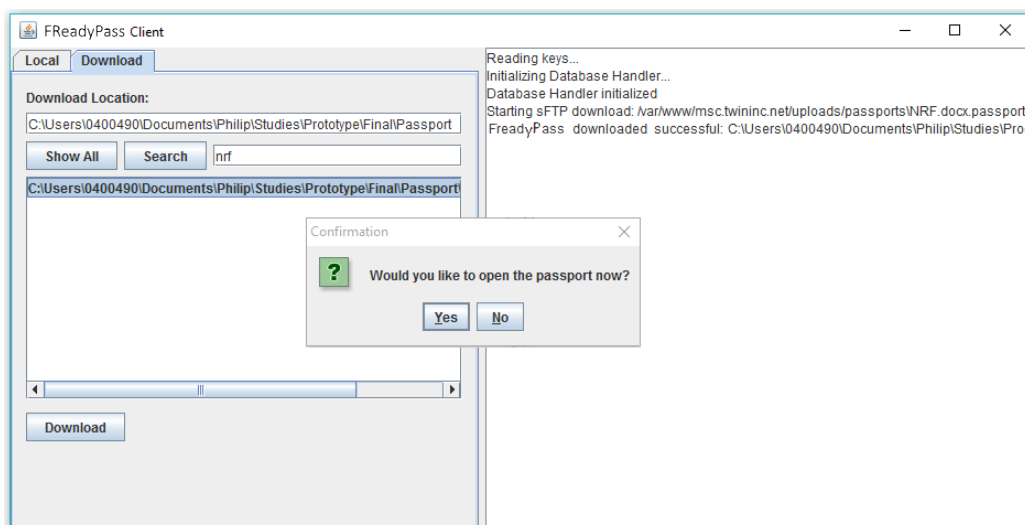


Figure 9.5.1: Downloading a FReadyPass

Figure 9.5.1 shows an example of a user downloading a FReadyPass and Figure 9.5.2 shows the complete flow diagram of everything that happens in the backend to complete the process of downloading and decrypting the FReadyPass.

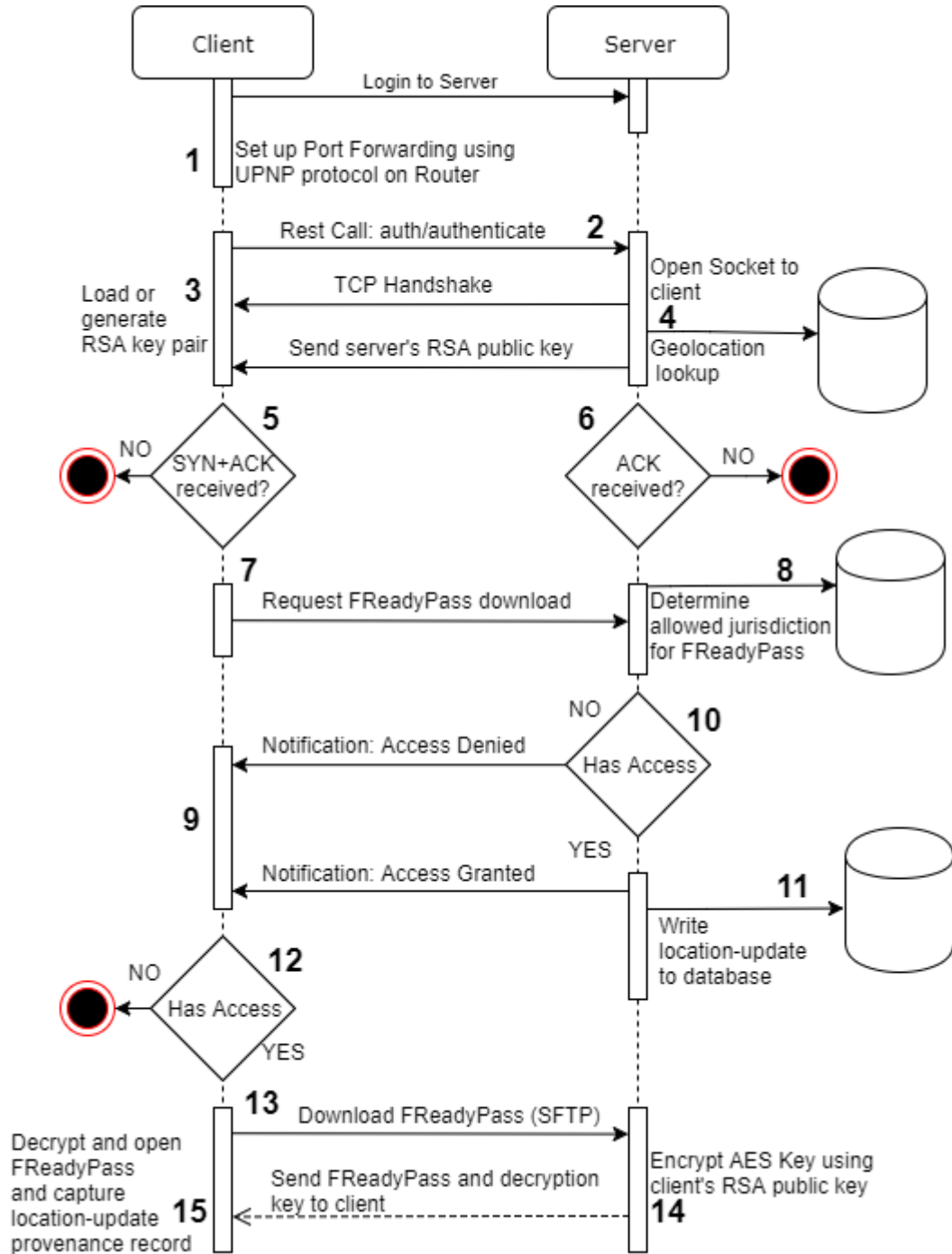


Figure 9.5.2: Secure Download process

Step 1: The user login to the server and the client application establish a port forwarding rule on the local router to allow for the incoming TCP handshake to reach the client application.

Step 2: The client sends a REST call to the server to authenticate the client.

Step 3: The client loads its RSA key pair from the local key-store. If the client has to generate a new RSA key pair, it is done at this point.

Step 4: The server initiates a TCP handshake to the client to verify the client's IP address. Once the IP address has been verified, the server uses the IP address to do a Geolocation lookup against the database to determine the client's jurisdiction. Finally, the server sends its RSA public key to the client to use when decrypting the AES key.

Step 5 and 6: The client and server applications respectively determine if the 3-way handshake was successful.

Step 7: The client requests a FReadyPass for download.

Step 8: The server determines if the client is allowed to access the requested FReadyPass at its present location.

Step 9: The client waits for the notification response from the server indicating it the client is allowed to download the FReadyPass. If the client is granted access, the server sends the FReadyPass's physical location on disc to the client to allow the client to download the FReadyPass using the SFTP protocol.

Step 10: The server notifies the client whether or not it is allowed to download the FReadyPass.

Step 11: The server updates the database to indicate the location of the FReadyPass. The database allows for easy lookup of a FReadyPass's location.

Step 12: If the client is not allowed to download the FReadyPass, the process ends, else step 13 is executed.

Step 13: The client downloads the FReadyPass from the server making use of the secure SSH FTP protocol SFTP.

Step 14: The server encrypts the AES key with which the FReadyPass is encrypted with the RSA algorithm using the server's private key and the client's public key. The server sends the RSA encrypted AES key to the client for the client to decrypt the FReadyPass.

Step 15: The client decrypts the RSA encrypted AES key after which the client decrypts the FReadyPass. Next, the client captures a location-update provenance record.

9.6 Uploading a FReadyPass

Once the user is finished with the modification to the payload, the client application saves the payload into the FReadyPass's payload section. A modification provenance record is captured, and the FReadyPass should be uploaded to the server if the user so chooses. The user may work on a payload for any amount of time. Therefore, one cannot assume that the session that was created when the FReadyPass was downloaded is still active. Therefore, the upload process requires certain steps that formed part of the download process to be retaken. Figure 9.6.1 shows the process of uploading a modified FReadyPass.

Step 1 through 6 is the same as for the download process.

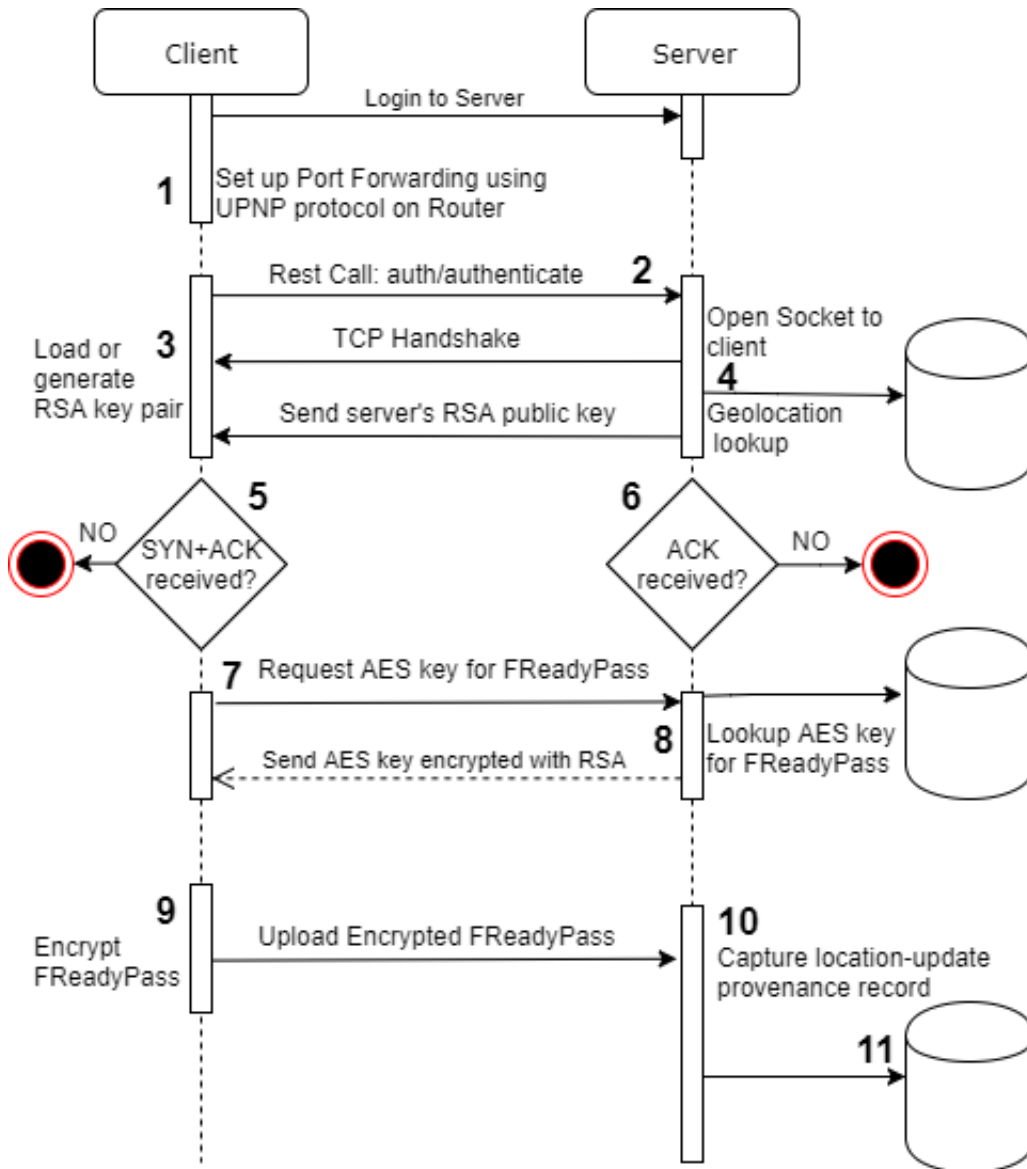


Figure 9.6.1: Secure Upload process

Step 7: After the client has been authenticated, the client requests the AES key with which to encrypt the FReadyPass. The client provides the FReadyPass's ID and the client's RSA public key as parameters when requesting the AES key to encrypt the FReadyPass.

Step 8: The server looks up the AES encryption key from the database using the FReadyPass's ID provided in step 7. The server encrypts the AES key using the server's RSA private key and the client's RSA public key before sending the key to the client.

Step 9: The client encrypts the FReadyPass using the AES key provided in step 8. It should be noted that a modification provenance record has already been captured when the payload was saved into the FReadyPass and this process is complete before the upload process start.

Step 10: Once the server receives the FReadyPass, the server decrypts the FReadyPass to verify that the provenance data's integrity is in tack. The server then updates the provenance data by adding a location-update provenance record indicating that the FReadyPass is back at the server. After the location-update record is captured, the server writes a verification provenance record to the FReadyPass indicating the validity of the provenance data.

Step 11: Finally, the server re-encrypts the FReadyPass and stores it after which the server updates the central database to indicate the FReadyPass's location within the cloud.

At least three provenance records will be written to the FReadyPass from the time it is downloaded to the time it is uploaded again. These include one location-update provenance record when the FReadyPass reaches the client, one modification provenance record after the FReadyPass has been modified and one more location-update provenance record when the FReadyPass reaches the server again. The integrity of these provenance records need to be maintained and cryptographic proof provided to prove the integrity of the provenance records. In the next section, the verification of the integrity of the provenance records is discussed.

9.7 Verify the integrity of provenance records

Figure 9.7.1 shows a proof of concept application verifying the integrity of a chain of provenance records. The file containing the valid provenance records is shown in Appendix B in Listing B.1.

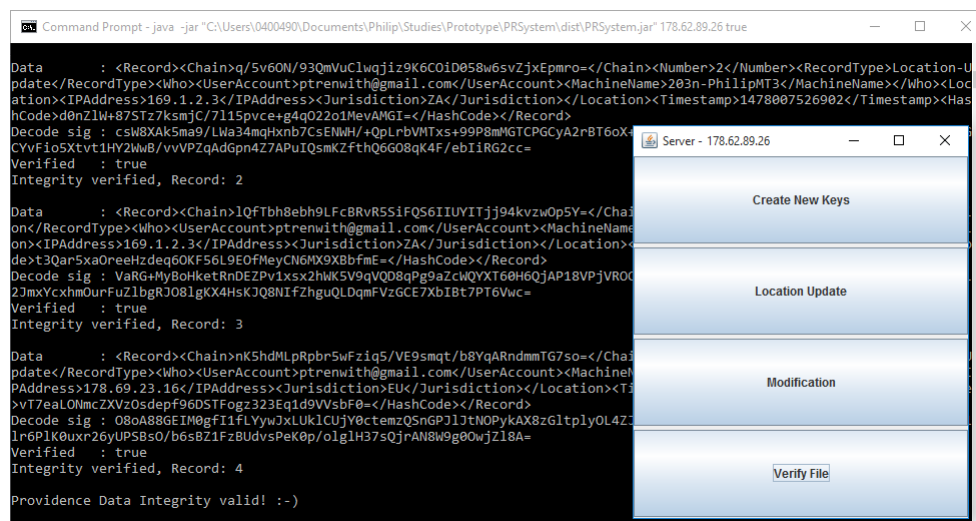


Figure 9.7.1: Shows a chain of provenance data to be valid

After verifying the data, the author modified a few of the provenance records to prove that the verification process notices modifications made to the chain of provenance data. Figure 9.7.2 shows the outcome. The file containing the invalid provenance data is shown in Appendix B in Listing B.2.

The example in Figure 9.7.2 shows how the system verifies each provenance record's integrity including the Index Record. The system also confirms that all the provenance records are present and none has been deleted. The system uses RSA signatures to verify the integrity of provenance data. In the next section, the author briefly discusses making provenance data available to users.

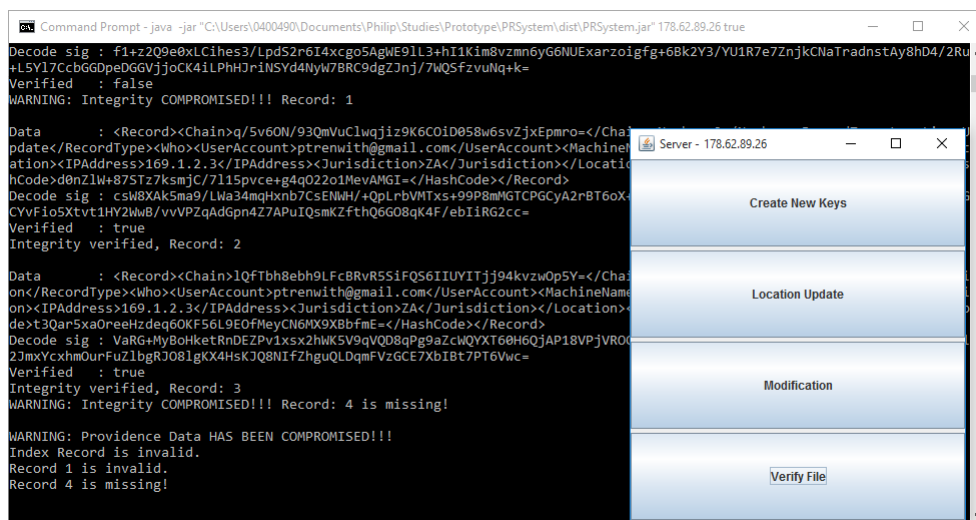
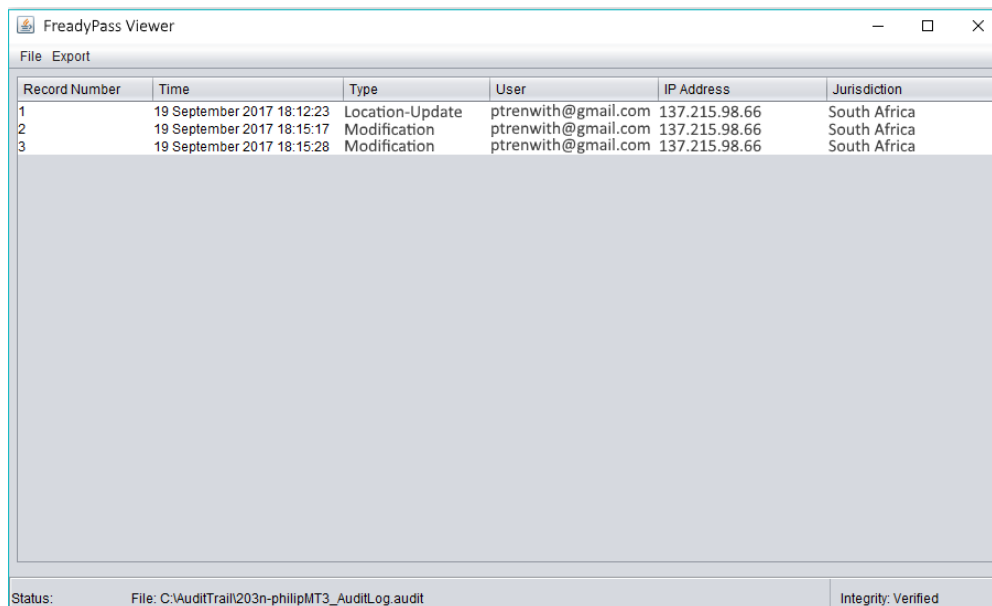


Figure 9.7.2: Shows a modified chain of provenance data to be invalid

9.8 FReadyPass provenance viewing application

Owners of a FReadyPass may want to see how their data has been accessed in and out of the cloud. In some cases, the provenance data may have to be made available to Digital Forensic Investigation teams. Whenever it is necessary for provenance data to be viewed the FReadyPass viewer application shown in Figure 9.8.1 provides that functionality. The FReadyPass header indicates the length of the provenance data in the FReadyPass. This allows the viewer application to read the provenance data directly from the FReadyPass. The FReadyPass viewer application therefore only decrypts the provenance data briefly to read the data into memory to be displayed by the application. Once the provenance data has been read, it is re-encrypted immediately thus, the FReadyPass remains secure. The provenance data displayed by the viewer application cannot be altered.

Figure 9.8.1 shows the FReadyPass viewer application designed to view provenance data.



The screenshot shows a window titled "FReadyPass Viewer" with a menu bar containing "File" and "Export". Below the menu bar is a table with the following data:

Record Number	Time	Type	User	IP Address	Jurisdiction
1	19 September 2017 18:12:23	Location-Update	ptrenwith@gmail.com	137.215.98.66	South Africa
2	19 September 2017 18:15:17	Modification	ptrenwith@gmail.com	137.215.98.66	South Africa
3	19 September 2017 18:15:28	Modification	ptrenwith@gmail.com	137.215.98.66	South Africa

Below the table is a large empty grey area. At the bottom of the window, there is a status bar with the text "Status: File: C:\AuditTrail\203n-philipMT3_AuditLog.audit" and "Integrity: Verified".

Figure 9.8.1: UI designed to view provenance data chain

9.9 Conclusion

In this chapter, the author discussed the setup of the prototype. The author further discussed the location-based access control mechanism and showed the process of determining a user's location before the user is granted access to download a FReadyPass. The author discussed the design of the client application and the process of uploading and downloading a FReadyPass and verifying the integrity of the provenance data by making use of the RSA cryptographic sign and verify functions. Using RSA keys to sign and verify the integrity of provenance data requires the RSA private keys to remain secure at all times. If a private key is compromised, a malicious agent may use the compromised private key to sign provenance records. Upon verification, these provenance records will pass the verification process because it was produced with a valid private key. Thus, it is of vital importance to ensure that private keys are not compromised. Thus, it is vital that the RSA key-pair is changed as soon as a suspicion exists that the private key may have been compromised.

If the server's private key is kept secure, it is possible to ensure that a FReadyPass can successfully be tracked through the cloud from its inception, to the current point in time, without any gaps in where it has been.

In part III the author proposed the design of a system to track data through the cloud by implementing a FReadyPass system. In chapter 9 the author presented and discussed a physical implementation of the proposed system.

In the next chapter, the author briefly discusses related work addressing some of the challenges identified in this dissertation. The author investigates the advantages and disadvantages of these systems and compares these systems to the proposed solution in this dissertation.

Chapter 10

Related Work

10.1 Introduction

The goal of this chapter is to identify similar research being undertaken as well as alternate solutions possibly addressing the same or similar issues. The author investigates the advantages and disadvantages of these solutions to evaluate how the proposed solution compares to other solutions.

10.2 Related work on the application of data provenance systems in the cloud

In this section, the author investigates related systems performing similar functions to the proposed system. The first system is the PASSv2 system proposed by Muniswamy-Reddy et al. [2009].

10.2.1 PASSv2

The PASSv2 system proposed by Muniswamy-Reddy et al. [2009] is designed to maintaining data provenance. This system is designed to build provenance-aware network storage (NFS) onto a Provenance-Aware Storage System (PASS). The author will not go into the technical details of

the PASSv2 system or any related work extending on it because most of the technical details of the PASSv2 system is concerning system calls at the operating system and this is beyond the scope of this work.

The main advantage of the PASSv2 system to others is the fact that this system takes different layers of abstraction into account when collecting provenance data. Muniswamy-Reddy et al. [2009] defines the different layers as: The system call layer, a workflow specification, and the application layer. The system call layer intercepts read/write requests to the file system and capture provenance indicating what is being accessed and by which process. However, to do so the PASSv2 system has to be integrated with the Linux Kernel, and with each upgrade of the kernel, the system has to be reintegrated.

Muniswamy-Reddy et al. [2009] states that although existing provenance systems capturing provenance at a single level has value, the provenance captured at each level cannot relate with provenance captured at other levels. Muniswamy-Reddy et al. [2009] states that each level of provenance is valuable, but it is more valuable if it can be combined. He states that “the most valuable provenance is that which is collected at the layer that provides user-meaningful names.” Thus, if capturing of provenance is concerned with processes accessing files, system call provenance is most valuable, but if the capturing of provenance is concerned with users accessing files, provenance at a higher level has more value.

The PASSv2 system captures provenance at different levels of granularity and has the capability of combining the provenance captured at different levels, which has great value. The levels of granularity are shown in Figure 10.2.1. PM refers to Physical Machine, and VM refers to Virtual Machine.

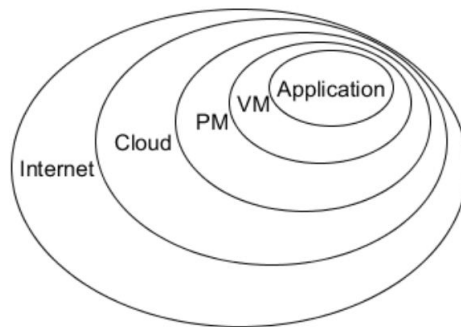


Figure 10.2.1: Granularity of provenance. Image from: Zhang et al. [2011]

Zhang et al. [2011] points out that the chain of provenance through different levels will break if data is moved between virtual machines unless provenance-aware network protocols are in place. Thus, for this system to function requires the entire CSP network and internal systems, including virtual machines, physical machines and internal networks to be configured and installed with the PASSv2 system to capture provenance. This setup is complicated and cannot be used as a “Plug-and-Play” solution for tracking data and capturing provenance. The author intends to develop such a solution that can be implemented by CSPs without disrupting their regular business in any way. Macko et al. [2011] extended on the work of Muniswamy-Reddy et al. [2009] by modifying the Xen hypervisor to collect provenance from running virtual machine kernels. This modification is discussed in the next section.

10.2.2 Xen Hypervisor modified for PASSv2

This approach places a hook in the Xen’s *syscall_enter* and *do_iret()* mechanisms. The approach intercepts virtual machine calls from the hypervisor and determines which of these calls are system calls. This system is similar to the PASSv2 system discussed in the previous section with the added advantage that it can start collecting provenance data immediately after the guest VM is powered on. Trenwith and Venter

[2013] propose the development of stand-alone provenance systems that does not require integration with a modified kernel. One such system is discussed in the next section.

10.2.3 Windows log files stored on a central log server

Trenwith and Venter [2013] developed a technique for proving the integrity of log files stored on a central log server. This is done by calculating a hash value at the time the logs are captured, this hash value is encrypted with a secret encryption key, and the corresponding ciphertext is stored in the log. The integrity of the log file can then be verified at any time by recalculating the hash value and then encrypting the value with a validation application that knows the secret encryption key. The application compares the current recalculated ciphertext against that saved with the log file. In the next section, the author briefly looks at a technique to capture provenance from computer network infrastructure.

10.2.4 Network Infrastructure used to determine location

Trenwith and Venter [2014] propose a system that captures provenance data from network infrastructure such as the TCP/IP protocol, corresponding with CSP logs to identify the physical location of objects in the cloud. In the work of Trenwith and Venter [2014] no mention is made of using the location information to determine the jurisdiction. The work proposed in this dissertation improves on the work of Trenwith and Venter [2014] by using the location information in the location-based access control mechanism, addressing the challenges associated with jurisdiction. Zhang et al. [2011] proposed a system called DataPROVE that takes many of the challenges mentioned so far into account and is discussed next.

10.2.5 DataPROVE, TrustCloud and Flogger

DataPROVE addresses issues identified with the Data Layer of the TrustCloud framework discussed in [Ko et al., 2011b]. TrustCloud attempts to increase data transparency and CSP accountability to increase the trust users have in CSPs. TrustCloud consists of five layers of granularity namely:

- System Layer: Monitoring among other, files and databases in the Cloud.
- Data Layer: Monitoring the creation and modification of data in the Cloud.
- Work-flow Layer: Monitoring the flow of data in the Cloud.
- Laws and Regulation layer: Monitoring the compliance and alignment of data flow in the Cloud.
- Policies Layer: Monitoring the compliance to internal policies.

A mechanism that captures System Layer provenance is Flogger[Ko et al., 2011a] Flogger monitors the transfer of files within a cloud as well as operations performed on files. The logs produced by Flogger include information regarding who accessed a file, what type of operation was performed, at what time and where, and the logs include virtual machine and physical machine operations.

Figure 10.2.2. shows the DataPROVE Network. Flogger is one of the mechanisms in this network. The others are:

- Change Tracker: Which monitors changes at the file system level. Change Tracker can capture among other which blocks of a file have been modified.

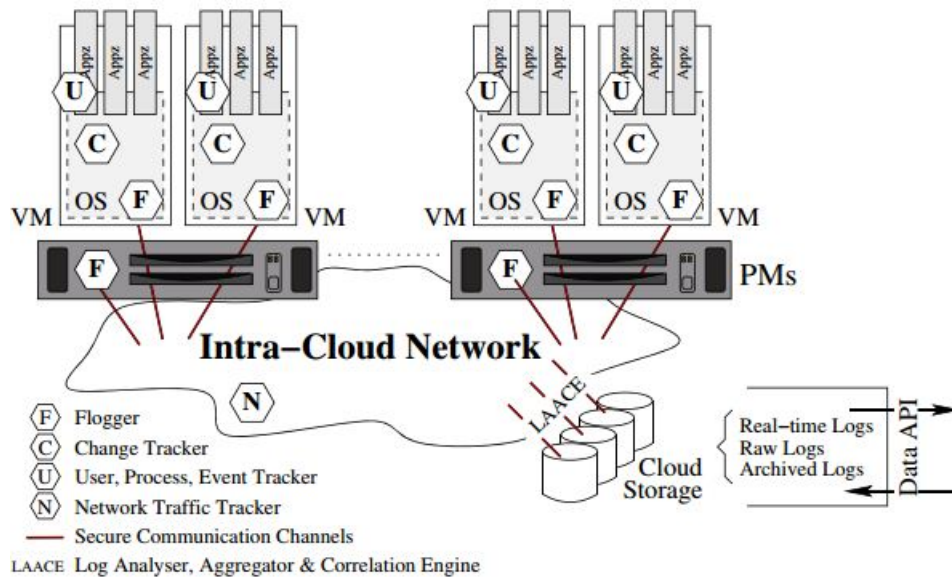


Figure 10.2.2: DataPROVE Network Framework. Image from: Zhang et al. [2011]

- User, Process and Event Tracker: Monitors user, and process activities and events. This mechanism is the foundation for capturing provenance across various layers of a machine.
- Network Traffic Tracker: Monitors the movement of packets within the network. This mechanism tracks where packets were sent from, which path it traveled and where it was received in the CSPs network. It can show the various network routes the packets of a file traveled during transfer.

A solution similar to the work proposed in this dissertation is the CloudDT system proposed by Tan et al. [2012]. CloudDT is designed to track data when it leaves the cloud and is discussed next.

10.2.6 CloudDT

CloudDT attempts to track data inside a controlled cloud environment and when the data leaves the cloud. The CloudDT system utilizes kernel-

level data event monitoring to track the data in the cloud. This poses a challenge for cloud service providers interested in utilizing this solution. The operating systems used by the CSP requires a modified kernel to track data in the cloud environment.

When a cloud user attempts to download data from the cloud, CloudDT enforces a checkout procedure whereby the system encapsulates the requested data files into a self-executing container containing a viewer application that is used to control access to the data files. The viewer application also attempts to transmit event logs back to the CSP containing information regarding what is being done with the data and who is accessing the data. Figure 10.2.3 and Figure 10.2.4 shows an overview of how the CloudDT viewer operates and Figure 10.2.4 shows an example of the event logs that is transmitted back to the server.

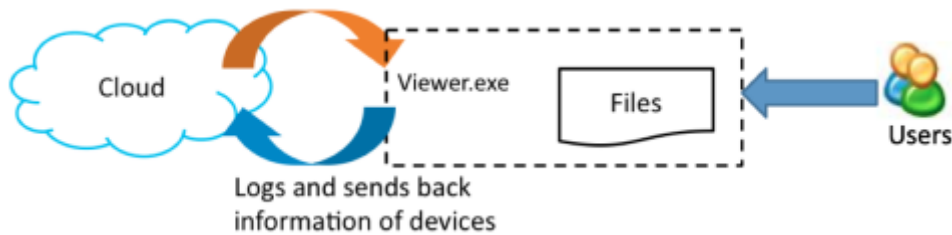


Figure 10.2.3: CloudDT Viewer from Tan et al. [2012]

```
Timestamp,Timestamp_precision,Filename,Actions,User_name,Process_Id,Host_name,Host_IP,Host,Mac
2011-03-23 14:00:26,1300860025,mounts,Read,alice,1427,Na@hp,210.21.12.37,D0:66:83:FE:00:76:00:DF
2011-03-23 14:00:34,1300860033,readme.txt,Read,alice,1428,Na@hp,210.21.12.37,D0:66:83:FE:00:76:00:DF
2011-03-23 14:00:41,1300860040,mounts,write,alice,1427,Na@hp,210.21.12.37,D0:66:83:FE:00:76:00:DF
2011-03-23 14:00:53,1300860053,readme.txt,write,alice,1428,Na@hp,210.21.12.37,D0:66:83:FE:00:76:00:DF
2011-03-23 14:01:01,1300860060,document.txt,Read,alice,3029,Na@hp,210.21.12.37,D0:66:83:FE:00:76:00:DF
2011-03-23 14:01:14,1300860073,document.txt,write,alice,3029,Na@hp,210.21.12.37,D0:66:83:FE:00:76:00:DF
2011-03-23 14:01:24,1300860083,document.txt,close,alice,3029,Na@hp,210.21.12.37,D0:66:83:FE:00:76:00:DF
```

Figure 10.2.4: CloudDT Event Logs from Tan et al. [2012]

The purpose of the CloudDT system is to track data both in and out of the cloud and provide the owner of the data with an audit trail

showing how the data is being used. Although this system is a good system and the design is well thought out, it does have a few limitations and disadvantages.

- The author of the system states that it is possible that the self-executing container will become compromised outside the controlled environment of the cloud thus resulting in the data being lost.
- The author assumes that event logs regarding access to the data will be transmitted back to the server, to compile complete audit trails.
- The author assume that the checkout process will be invoked correctly to encapsulate the data into a self-executing container. If the checkout process is compromised, it is possible for data to leave the controlled environment of the CSP without the ability to track the data, thus resulting in data loss.
- The author states the requirement to have measures in place to ensure the consistency and integrity of the event logs outside the controlled environment. However, the CloudDT system only encrypts the data when it is being transmitted from the viewer to the server. It does not provide any integrity verification mechanism. Although, the encryption is done using the viewer application, therefore if the viewer application becomes compromised the encryption would also be compromised. Thus, there will be no way of validating the integrity of the event logs.
- The event logs is combined with the logs captured at the server at the CSP to produce the audit trail. There is no mechanism in place to prove the completeness of the audit trail.
- The self-executing container is encrypted to protect and control access to the data files. The decryption key is pre-programmed into the viewer to decrypt the data files. Thus, if the viewer is

decompiled the encryption key is compromised, and the data files will be lost.

10.2.7 Blockchain Technology

Blockchain technology is different from the other solutions discussed in this chapter because blockchain technology was not developed with the primary intention of capturing provenance data. Blockchain is essentially a distributed database[Crosby et al., 2016b]. All records in the database are shared among all entities that form part of the distributed network, making the database decentralized. Therefore, once a record has been written to this database, it can never be deleted. The distributed database is also referred to as a public ledger, and the records it contains can be any transaction or digital event. The entities that form part of the distributed network are referred to as miners or workers. Apart from storing the data in the blockchain, the minors also verify the integrity of the data in the blockchain. The blockchain is verifiable in much the same manner as the provenance data in this dissertation. Digital events stored in the blockchain are grouped in a block, and the block is hashed using a cryptographic hash function. The computed hash of block i is placed in the next block produced in the blockchain, block $i + 1$ as shown in Figure 10.2.5[Nofer et al., 2017].

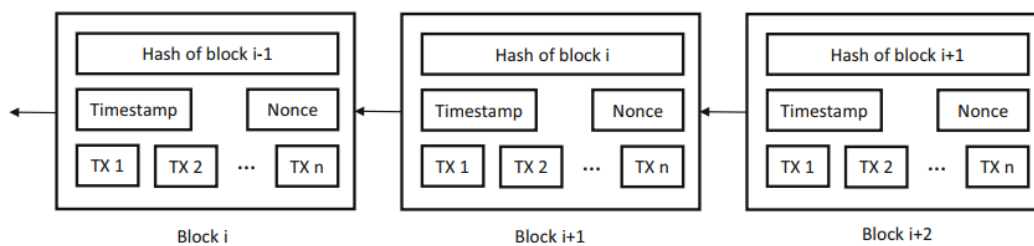


Figure 10.2.5: Blockchain example adopted from Nofer et al. [2017]

Each block in the blockchain contains the timestamp it was produced,

a nonch that is a random number used to verify the hash, the hash of the previous block and a number of transactions or digital events represented as Tx 1 to Tx n. The workers generate the blocks and receive a monetary reward in return for their effort. Before a worker can append a block to the blockchain, it must calculate the answer to a complicated question. The complexity of this question is determined based on the computational power of the network and the desired time required to produce a block. This is defined as the difficulty. Overall, the difficulty is set so that a block takes 10-minutes to produce. The underlying cryptographic theory behind blockchain does not allow a block to be modified and only the longest chain in the network is accepted. The security is therefore based on the premise that it is difficult to produce a longer chain than the longest chain in the network because of the time required to produce a block. However, this theory is flawed as discussed by Bastiaan [2015]. If a pool of workers holds approximately 51% of the processing power of the network, it is possible for that pool to generate the longest chain in the network and thus it becomes possible for that pool to add false data to the chain.

One of the biggest revolutionary concepts of blockchain technology is the elimination of a trusted third-party. Consider as an example, Alice wants to buy a property from Bob. Alice has to deposit money into Bob's bank account to buy the property after which Bob has to transfer the deed to the property to Alice. Alice trusts that Bob's bank, the third party, is trustworthy and will not steal the money. Alice also has to trust that Bob will transfer the deed once the money has been transferred. Blockchain allows Alice and bob to interact with one another without requiring a third party. This use case is handled by blockchain technology through smart contracts[Crosby et al., 2016b]. A smart contract is a computer program that is added to the blockchain that will automatically execute once a certain condition is met[Omohundro, 2014]. This condition is the transfer of money from Alice to Bob, which can be done using a cryptocurrency[Pre, 2016], after which the execution of the smart con-

tract transfers the deed to Alice.

[Liang et al., 2017] propose a trusted cloud data provenance architecture using blockchain technology. This architecture is referred to as ProvChain. ProvChain consists of a provenance data collection phase, a provenance data storage phase, and a provenance data validation phase. Provenance data is hashed into a Merkle tree. The nodes of the Merkle tree is attached to the blockchain as digital events.

ProvChain achieves four objectives.

- Collects provenance data in real-time by monitoring user operations.
- Provenance data is tamper-proof because the provenance records are published to a blockchain network.
- Data provenance records is associated with a hashed user ID instead of the user's ID itself. Thus, the user's privacy is preserved. Only the service provider can link each provenance record with the owner of the provenance record.
- Provenance data records are published to a blockchain network. The receipts received from the blockchain network is used to validate the provenance data.

The ProvChain architecture captures user operations by using hooks and listeners to detect user operations on files. Figure 10.2.6 shows a block of Provenance records showing how the provenance records look.

RecordID	DateTime	Username	Filename	AffectedUser	Action	TxHash	BlockHash	Validation
001	2016-9-7 10:00:00	A	X	None	Create at CloudA/A	m-bits	n-bits	True
002	2016-9-7 11:00:00	A	X	None	Copy to CloudB/A/backup	m-bits	n-bits	True
003	2016-9-7 12:00:00	A	X	None	Read at CloudA/A	m-bits	n-bits	True
004	2016-9-7 13:00:00	A	X	B	Share to User B	m-bits	n-bits	True
005	2016-9-7 14:00:00	A	Y	None	Write to CloudA/Y	m-bits	n-bits	True
006	2016-9-9 11:00:00	B	Z	A	Copy from CloudA/A to CloudB/B	m-bits	n-bits	True
007	2016-9-9 12:00:00	B	Z	A	Read at CloudB/B	m-bits	n-bits	True
008	2016-9-9 13:00:00	B	Z	A	Write at CloudB/B	m-bits	n-bits	True
009	2016-9-9 15:00:00	A	X	B	Share X to public	m-bits	n-bits	True
010	2016-9-9 14:00:00	B	Z	None	Delete from CloudB/B	m-bits	n-bits	True

Figure 10.2.6: File Provenance Record in Blockchain. Image from Liang et al. [2017]

The file operations are captured and stored showing the date, time, the user and file affected, and a description of what was done to the file. The TxHash is the hash of the transaction in the block, and the Block-Hash is the hash of the block.

In the next section, the differences between the FReadyPass system and some of the other solutions are discussed.

10.3 Proposed solution compared to other solutions

In section 10.2 of this dissertation, the author looked at the research being done in academia related specifically to tracking data in the cloud and the provision of provenance data in cloud computing environments. Many researchers focus on tracking data and providing data provenance with regard to data within a controlled environment, under the control of the cloud service provider and within the CSP's network.

Most of the available data tracking systems currently available in the cloud have many limitations. First, the existing solutions require operating systems with altered kernels to capture provenance data. Although this approach can be seen as potentially a more secure implementation with more control over the information being monitored, the limitation of such an implementation is that it cannot be used as a plug and play

solution in systems outside the controlled environment of the cloud service provider.

The DataPROVE system developed by Zhang et al. [2011] is one such system that has the capability to track user data in the cloud. However, the tracking solution implemented in the DataPROVE system does not track the data with regard to its physical location in the world, it tracks the data regarding its movement within the cloud service provider network and the physical and virtual hardware utilized when accessing and transferring user data.

Although most of the existing solutions capture provenance data at various levels from the virtual machines down to the physical hardware and network switches, there exists a greater concern regarding how sensitive data is being accessed and used outside the control of the cloud service provider. With the exception of the CloudDT system proposed by Tan et al. [2012], tracking of data when it leaves the control of the cloud service provider has not been successfully achieved.

The two closest systems to the proposed solution are the CloudDT system and the ProvChain system. The differences are discussed next.

10.3.1 FReadyPass system vs. CloudDT

Currently, no system exists that limits the flow of information across jurisdictional boundaries, even though Tan et al. [2012] mentioned that location information might be used as part of the authentication process, it was not implemented as part of their work. The author improves on the CloudDT system proposed by Tan et al. [2012] in the following ways:

- The CloudDT system attempts to transmit event logs back to the server to compile a complete audit log. If a malicious attacker can disrupt the transmission, it is possible that audit logs may

not be compiled or the logs may not be complete. The CloudDT system does not contain a mechanism to verify the integrity of audit logs. The FReadyPass improves on the CloudDT system in two ways. First, all the provenance data comprising the entire audit trail is kept with the data. Thus, the process of compiling the audit trail cannot be interrupted. Secondly, the FReadyPass provides cryptographic proof of the integrity of the data provenance as well as the integrity of the user data.

- The CloudDT system requires a checkout process to encapsulate the data into a self-executing container. The advantage of the FReadyPass system over the CloudDT system is that once the data is compiled into the FReadyPass, it is encrypted to protect the data. Furthermore, it is only decrypted by trusted applications thus, the risk of data loss is much less.
- The CloudDT system encrypts data contained in the self-executing container. However, the encryption key is pre-programmed into the viewer application. It may be possible for a malicious attacker to decompile the viewer application to extract the encryption key which would result in the data files being compromised, possibly resulting in data loss. The FReadyPass system is more secure, exchanging the encryption key for accessing a FReadyPass using the RSA key exchange algorithm over a secure communication channel.
- There is no mechanism in place to prove the completeness of audit trails in the CloudDT system. However, the FReadyPass system chains provenance records together and digitally sign the provenance records to ensure that modifications to the provenance records do not go unnoticed. Furthermore, chaining the provenance records together proves the completeness of the provenance data.

10.3.2 FReadyPass system vs. ProvChain

ProvChain captures provenance data regarding file operations. ProvChain does not limit the flow of data across jurisdictional boundaries, nor does it determine access to a file based on a user's physical location as FReadyPass does. The integrity of ProvChain's provenance is based solely on the distributed nature of the blockchain technology. Consider a cookie jar locked away in a cupboard in a basement. If someone can break into the cupboard, they could steal all the cookies without being caught. However, if the cookie jar is placed on a clearly lit stage with a thousand people watching the jar at all time, it becomes impossible to steal from the jar without being caught, the security of blockchain is based on this principle.

The author improves on the ProvChain system proposed by Liang et al. [2017] in the following ways:

- The ProvChain system is based on blockchain technology. The blockchain technology is deemed secure because the chain is visible to the entire network and because of the difficulty factor. A malicious agent cannot produce a false chain that is longer than the longest chain in less time than the honest nodes produce a valid chain. This principle does not hold if a chain only contains provenance records of a single file. It would not be difficult to produce a false audit trail. Therefore, ProvChain keeps the provenance records of all files in a single chain. The FReadyPass system improves the security. The FReadyPass system keeps provenance data of a single file in a single chain, and the chain remains secure.
- ProvChain requires a distributed network of nodes to produce the blockchain as well as verify the blockchain's integrity. ProvChain does not work unless there exists such a distributed network of nodes. The nodes in the network produce the chain, and for the

work, they perform they receive a financial award. FReadyPass is better in this regard because the CSP does not have to pay workers to produce and verify the audit trail.

10.4 Conclusion

In this chapter, the author investigated related work being conducted in the field of data provenance and cloud computing environments. The author compared the proposed solution to other systems, paying special attention to how the proposed solution improves on other existing systems.

In the next chapter, the author discusses the work that was done in this dissertation. This is a critical evaluation of the author's own work providing his own opinion of the work and the value it adds to the community.

Chapter 11

Critical Evaluation

11.1 Introduction

This chapter is a critical evaluation of the work that was done, and the author discusses some of the shortcomings of the proposed solution. The author compares the proposed solution to the ideal solution and discusses why it is not possible or impractical to achieve the ideal solution. The work is compared to other solutions, highlighting some of the disadvantages of other solutions and further highlighting how the proposed solution improves on these disadvantages. The author discusses the work that was done regarding the research question that was proposed. The author evaluates the value that this work adds to the domain. The value of the author's contribution to the scientific community is briefly discussed.

11.2 Detailed review of the proposed model and prototype

The author proposed a model referred to as FReadyPass that attempts to achieve two primary goals. First, the model aims to monitor and

control the access of information across jurisdictional boundaries. The model provides the ability to both the users and the cloud service provider hosting the information to decide from which jurisdictional areas the data may be accessed. Providing a much-needed access control mechanism to the cloud that satisfies legislation put in place by numerous countries and governing bodies regarding the control and access to sensitive information from different locations. The model further address security concerns regarding the use of IP addresses to determine Geolocation. The model determines the authenticity of an IP address by initiating a 3-way handshake between the server and the client to ensure that the client is who it claims to be. In doing so, the model also verifies the determined location thus ensuring that the location information stored with the provenance data is accurate.

Providing CSPs with the ability to control the flow of information across jurisdictional boundaries is not an easy objective to achieve since the cloud cannot be laid out in physical space without the presence of sufficient anchor events. An anchor event physically binds digital space to physical space as defined by [Cohen, 2011, p.44-51]. Using the IP address lookup databases managed by IANA provides the author with an information source capable of anchoring digital space to physical space. Performing a reverse IP address lookup on a connection and verifying the authenticity of the IP address through a 3-way handshake, provides the user with information showing where the data is being accessed from.

The second goal of the proposed model is to track the data as it is transferred in and out of the cloud including access and modification to the data. The proposed model does not only track the data to determine the location from where the data is being viewed and accessed. The solution also compiles detailed provenance logs that can be used in any investigation to determine who had access to the data, where it was accessed from, what was being done to the data and what the state of the data was after the access or modification request was completed.

In order to compile detailed provenance logs, require that access to the information be monitored. Furthermore, a mechanism is required that is capable of verifying the integrity of the provenance data, proving that the provenance data has not been compromised. To achieve this, the author proposed the design of a FReadyPass. The FReadyPass encapsulates a data file referred to as the payload, together with the history of that digital object referred to as its provenance data. The FReadyPass, which is encrypted, can only be opened by the client application and the server. Thus, access to the FReadyPass is controlled.

Ideally, it will work well if it was possible to embed code able to track data in and out of the cloud inside the payload, instead of having to encapsulate the data inside a FReadyPass. Unfortunately, that is not possible. The FReadyPass serves two main purposes. Number one, the FReadyPass is its own file type, requiring a specially designed application to open the file. This provides a layer of security. The specially designed application required to open the FReadyPass file allows the system to determine the location of the FReadyPass before it is opened, thus, allowing the payload to be tracked and the location forms part of the access control requirements. The second purpose that the FReadyPass serves is the ability to encapsulate the data provenance with the payload.

The design of the FReadyPass together with the proposed FReadyPass system accomplishes the objectives that the author set out to achieve. The primary disadvantage of the system is the fact that the payload may potentially be stolen once the FReadyPass has been decrypted and opened. If it was possible to embed the tracking information or application code inside the payload the danger of the payload being stolen can be avoided because if the payload is copied the embedded code is copied with it. Unfortunately, it is not possible to embed application code into file types that were not designed to allow embedded application code. Furthermore, embedding executable code into a file type will more than

likely trigger anti-virus systems to block the file from opening or the tracking code from executing.

Because the cloud cannot be laid out in physical space, one cannot control the flow of information at a physical border. Consider when a person travels from one country to another via a train, when the train reaches the border it is stopped and the border officials check the passport's and visa's of the passengers and employees on the train to determine if they are allowed to enter the country. However, when a person travels from one country to another via air, the person's passport and visa have to be checked and verified before the aircraft departs, even though both the passport and visa is rechecked when the aircraft reaches its destination. If a traveler is not authorized to enter the country, he/she will have to return to the country they came from.

FReadyPasses being downloaded to a client more closely resemble international air travel. When the FReadyPass is requested from the server the location-based access control mechanism determines the validity of the request, similar to a passenger boarding a plane to travel internationally. Once the FReadyPass reaches its destination and is decrypted, the provenance data is updated to indicate the FReadyPass's arrival at the destination, similar to a person's passport being stamped when he/she enters a country.

Although this process does not block the flow of information at a jurisdictional boundary, it does limit where that information may be accessed from. Since the FReadyPass is encrypted, the CSP has full control over where it can be opened from. This model provides a decent foundation to control the access of information across jurisdictional boundaries.

One addition to this model that will make it more secure is the ability to detect remote connections to a device attempting to open a FReadyPass. Consider a user at computer A has a FReadyPass open for access.

However, another user at computer B has a Windows Remote Control connection, connected to computer A. With this remote connection open, it is possible for the user at computer B to see what the user at computer A is doing. Therefore, the unauthorized person can access and in fact, copy, sensitive information, while it is open on computer A. One possible technique to detect remote connections is to monitor the TCP/UDP protocol for data packets being sent over the network. This model will be more secure if the ability to detect remote connection can be provided.

Additionally, there are a couple of dangers that have not yet been addressed, while the FReadyPass is encrypted, it doesn't matter that the user can copy the data because the user requires the client application to decrypt and open the FReadyPass. The client application will authenticate the user and his/her location when an attempt is made to open the FReadyPass. Thus if the FReadyPass is copied onto a secondary storage device, while the FReadyPass is encrypted it is still under the control of the CSP. However, once the FReadyPass is decrypted the user may be able to copy the payload somewhere else, and the data may potentially be taken outside the control of the jurisdiction.

To account for this possibility, the author proposed and implemented a secure folder. The FReadyPass is decrypted into, and the payload extracted into this secure folder. The client application monitors the Operating System clipboard and does not allow the data in this folder to be copied or cut, and the client application also overrides the Operating System "Send to" command thus, preventing the payload from being sent to a secondary storage device. At first glance, this approach seems secure. Unfortunately, it only makes it more difficult to copy the payload, and it does not make it impossible. The user can do any one of the following things to bypass the secure folder and gain access to the payload. The options are listed in order of the complexity or time required to do them. Listed as least effort to most effort needed by the user to copy the data.

- A decrypted payload file may be opened with the relevant program and the “Save As” option used to save a copy to another location.
- A digital printer driver may be used to print a digital copy of the payload to another location, resulting in a second digital object.
- The user may kill the client application process, this will not delete the decrypted payload file in the secure folder, but it will kill the process monitoring the secure folder, thus making it possible to copy the payload file by conventional manners.
- Consider the case where the application monitoring the secure folder is installed as an Operating System service that cannot be killed by the user. The user can still disrupt power to the device and by pulling out the power cable or removing the device’s battery. This will result in an immediate shutdown, preventing any monitoring software to perform a graceful shutdown whereby decrypted data is encrypted again. After that, the user may remove the hard-drive from the computer and install the hard drive into another machine, making it possible to copy the payload files from the location on the disc. It is possible to prevent the user from copying the payload from the hard drive if the secure folder is encrypted by something like TrueCrypt [<http://truecrypt.sourceforge.net/>, 2016]. Although the development team responsible for the TrueCrypt project no longer supports it. They are advising users to migrate to BitLocker, referring to potential unfixed security issues as the reason to migrate.
- A screen recorder may be used to record the screen, and the data may later be transcribed from the captured frames.
- A malicious attacker may perform a memory dump while a payload file is open, which potentially may result in the whole, or part of the payload file being dumped to disc, allowing a skilled attacker to retrieve the payload file or part thereof from the memory dump.

Once the FReadyPass has been decrypted, ultimately ensuring that the data is never copied is an impossible task, if any part of the process to access the data is outside the control of the implemented system. The only way to ultimately ensure that FReadyPass payload data is never copied once it is decrypted is to build a client-side application that can open the payload file. Doing so ensures that the flow of data is never handed off to a third-party application. If the client application has the ability to open and modify Microsoft Word Documents, PDF files, Images and any other type of file, the application never relinquish control of the payload to a third party application. In order to truly maintain control of the payload requires another step to be taken as well. The encrypted FReadyPass should not be decrypted to a physical hard drive, allowing the data to be accessed by any application outside the FReadyPass system. The payload file should be decrypted in memory and accessed from memory alone.

Having to develop an application that has the capability of viewing and modifying any file type is by no means ideal. Unfortunately, if the decrypted FReadyPass's payload is allowed to be accessed by an application or process outside the control of the FReadyPass system, there is a risk that it may be copied. Therefore, there exists a trust relationship between the CSP, the owner of the data and those authorized to view and modify the data.

11.3 Value added to community

The FReadyPass system adds value to the cloud computing domain as well as the digital forensic domain by allowing data to be tracked not just inside the controlled environment of the cloud service provider but also outside that environment when the data is downloaded and accessed from outside the cloud.

During a digital forensic investigation, if the history of a digital object comes into question, it is necessary to prove the integrity of the history in order to discard any questions regarding the integrity of the data. Thus, the ability of the FReadyPass system to cryptographically prove the integrity of data adds tremendous value to the industry.

The FReadyPass system achieves its objectives since the access to a FReadyPass is entirely controlled and access restricted to a specific jurisdiction.

11.4 Conclusion

In this chapter, the author discussed some of the shortcomings of the proposed solution. The author compared the proposed solution to the ideal solution as well as other available solutions. The author discussed some of the disadvantages of other solutions and highlighted how the proposed solution improves on the other solutions. The author discussed the work that was done regarding the research question that was proposed and evaluated the value that the proposed solution adds to the domain.

In the next chapter, the dissertation is concluded. The author briefly summarizes the dissertation, reviews the research question and discusses the results that were obtained in this dissertation.

Part V

Summary

Chapter 12

Conclusion

12.1 Introduction

This chapter concludes the dissertation. The author provides his own views on the results obtained, indicating to what extent the research question has been addressed. This chapter concludes with a suggestion on future work to be done.

Applying the proposed model, the history of a digital object is captured and encapsulated with the object in the form of a forensic ready digital passport known as FReadyPass. The encapsulation takes place once the object comes under the control of the CSP. When the FReadyPass is requested for download an access control mechanism determines if the request is valid and should be allowed. In the event that the request is valid, the FReadyPass is downloaded, and the user may modify the payload encapsulated within the FReadyPass. Once the user is done with the FReadyPass new provenance records are captured to indicate the modification and location where it took place, and the FReadyPass is uploaded back to the CSP. This model provides the ability to track data under the control of the CSP by making use of a forensic ready digital passport.

12.2 Review the research question

The research question asked in this dissertation is stated as follows: “How can the implementation of a digital passport and visa system address the cross-jurisdictional issues faced by digital forensic investigation teams performing investigations in the cloud?”

This dissertation aims to investigate how a digital object can be tracked through the cloud and the object’s history stored in a forensic ready manner. The author proposed the design of a model to accomplish this goal while providing an access control mechanism that relies on location information.

The research question is answered by relying on a reverse IP address lookup against the IANA registrar databases. The author relies on a successful 3-way handshake established from the server to client device to rule out IP Address Spoofing. This ensures that a FReadyPass may only be accessed from within jurisdictional areas that have been pre-approved to access user data. The access control mechanism, together with the central database that provides logs of the location of FReadyPasses ensures that a FReadyPass can only be downloaded to pre-approved locations. Thus, when it is necessary for a FReadyPass’s payload to be examined, the CSP or investigating team can quickly determine the FReadyPass’s location through the use of the central database. Furthermore, because of location-based access control, the FReadyPass will be in a jurisdiction accessible to the investigating team and the CSP for easy retrieval. The author believes that the proposed FReadyPass system adequately answers the research question asked in this work.

12.3 Research Contribution

This solution proposed in this work benefits digital forensic investigations by giving investigators an easy way of determining the location of data that may be required in an investigation. This model provides the potential to stop jurisdictional disputes with regard to digital forensic investigations.

The FReadyPass system furthermore produces a complete history of where a data object has been, who had access to the data object, what was done to the data and when it was modified. Finally, the system provides cryptographic proof of the integrity of the data files as well as the provenance data, which has been listed as a requirement in digital forensic investigation.

The FReadyPass system is superior to alternative solutions in many ways. The system does not require altered operating system kernels to capture provenance data. The system tracks data in and out of the cloud, which is superior to the CloudDT system, and the system does not require additional workers to verify provenance data and has to be paid in return like the ProvChain system does.

12.4 Future Work

Some future work to be done includes a technique to successfully detect active remote control connections on client devices. When it is the CSP's intention to limit access to information based on the physical location of a user, a device that is being controlled remotely may provide access to information to users outside of the allowed jurisdictions, even if the user is only able to read that information. Furthermore, future work should focus on limiting the loss of control over data encapsulated in the FReadyPass when it is accessed at the client. The model would be more secure and add even more value to the domain if control over data never

has to be relinquished to third-party software, as is the case at present when data files are being accessed from within the secure folder. If it were possible to maintain complete control over the data and access to the data, the FReadyPass system would more closely resemble a digital rights management system.

12.5 Conclusion

Although the FReadyPass system does have a few shortcomings that remain to be addressed, it provides a good foundation for future research to be conducted. Not only does it provide the capability to both cloud users and the cloud service providers to determine the physical location from where data is accessed. It also provides a mechanism to capture data provenance describing where, when and by whom the user's data has been accessed. This mechanism provides the ability to cryptographically prove the integrity of the describing data provenance whenever the integrity of the data has to be verified. The proposed solution further provides an alternative to cloud users, enabling them to move past the point of simply trusting the cloud service providers with their data. The users now have a mechanism available to them to verify what is being done with their data, allowing for a "Trust but verify" relationship to be established between cloud users and cloud service providers.

Part VI

Bibliography and Appendices

Bibliography

- F. Adelstein. Live forensics: Diagnosing your system without killing it first. *Communications of the ACM*, 49:63–66, February 2006.
- A. Al Hasib and A.A.M.M. Haque. A comparative study of the performance and security issues of aes and rsa cryptography. *Convergence and Hybrid Information Technology, 2008. ICCIT'08. Third International Conference on*, 2:505–510, 2008.
- Phyllis Altheide. Spatial data transfer standard (sdts). In *Encyclopedia of GIS*, pages 1087–1095. Springer, 2008.
- Michael F Angelo, David F Heinrich, Hung Q Le, and Richard O Waldorf. Computer access via a single-use password, April 9 2002. US Patent 6,370,649.
- Claudio A Ardagna, Marco Cremonini, Ernesto Damiani, Sabrina De Capitani di Vimercati, and Pierangela Samarati. Supporting location-based conditions in access control policies. In *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*, pages 212–222. ACM, 2006.
- ARIN. American registry for internet numbers resource policy manual, May 2015. URL <https://www.arin.net/policy/nrpm.html#two4>.
- R. K. Baggili, I. Carthy, and J. T. Kechadi. Survey on cloud forensics and

- critical criteria for cloud forensic capability: A preliminary analysis. In *University College Dublin, Zayed University*, 2011.
- J. J. Barbara. Cloud computing: Another digital forensic challenge. *Digital Forensic Investigator News*, October 2009.
- Martijn Bastiaan. Preventing the 51%-attack: a stochastic analysis of two phase proof of work in bitcoin. In *Available at <http://referaat.cs.utwente.nl/conference/22/paper/7473/preventingthe-51-attack-a-stochasticanalysis-of-two-phase-proof-of-work-in-bitcoin.pdf>*, 2015.
- D. Birk and C. Wegener. Technical issues of forensic investigations in cloud computing environments. In *Ruhr-University Bochum, Horst Goertz Institute for IT Security Bochum, Germany*, January 2011a.
- Dominik Birk and Christoph Wegener. Technical issues of forensic investigations in cloud computing environments. In *Systematic Approaches to Digital Forensic Engineering (SADFE), 2011 IEEE Sixth International Workshop on*, pages 1–10. IEEE, 2011b.
- Matt Bishop and LT Heberlein. Attack class: Address spoofing. In *Proceedings of the Nineteenth National Information Systems Security Conference*, pages 371–377, 1996.
- Robert Braden. Rfc-1122: Requirements for internet hosts. *Request for Comments*, pages 356–363, 1989.
- John David West Brothers and Jeffrey G Smith. Proxy server for tcp/ip network address portability, November 23 2004. US Patent 6,822,955.
- B. Burr. Nist hash function standards status and plans. In *NIST Special Publications 800-145*, December 2005.
- Laura Chappell. Inside the tcp handshake. *NetWare Connection*, 2000.
- Stuart Cheshire and Marc Krochmal. Nat port mapping protocol (nat-mpm). Technical report, 2013.

- William Roberts Cheswick and Edward G Whitten. Firewall security method and apparatus, February 6 2001. US Patent H1,944.
- Mike Chirico. Ntp, utc, and working with time. 2017. URL http://souptonuts.sourceforge.net/readme_working_with_time.html.
- Claudio U. Ciborra. *Reframing the role of computers in organizations - The transaction costs approach*. Elsevier Science Publishers B.V. Amsterdam, Universita di Trento, Via Verdi 26, Trento Italy, 1987.
- F. Cohen. *Digital Forensic Evidence Examination - 3rd Edition*. Fred Cohen and Associates out of Livermore, 2011. ISBN 1-878109-46-4.
- Comodo. Code signing. 2017. URL <https://www.comodo.com/business-security/code-signing-certificates/code-signing.php>.
- POPI Compliance. What is popi all about, May 2015. URL <http://www.popi-compliance.co.za/>.
- Douglas Crawford, 2016-01-20. URL <https://www.bestvpn.com/vpns-beginners-need-know/>.
- Michael Crosby, Pradan Pattanayak, Sanjeev Verma, and Vignesh Kalyanaraman. Blockchain technology: Beyond bitcoin. *Applied Innovation*, 2:6–10, 2016a.
- Michael Crosby, Pradan Pattanayak, Sanjeev Verma, and Vignesh Kalyanaraman. Blockchain technology: Beyond bitcoin. *Applied Innovation*, 2:6–10, 2016b.
- Gary-Berg Cross, 2014. URL <https://www.rd-alliance.org/group/data-foundation-and-terminology-wg/post/community-discussion-definition-digital-object.html>.
- DB-IP. Ip geolocation database, 2016. URL <https://db-ip.com/db/>.

- W. Delpont, M. Kohn, and M. S. Olivier. Isolating a cloud instance for a digital forensic investigation. In *Proceedings of the 2011 Information Security South Africa (ISSA 2011) Conference*, Johannesburg, South-Africa, August 2011.
- Dorothy E Denning and Peter F MacDoran. Location-based authentication: Grounding cyberspace for better security. *Computer Fraud & Security*, 1996(2):12–16, 1996.
- DigitalOcean. How to configure ssh key-based authentication on a linux server, 2016. URL <https://www.digitalocean.com/community/tutorials/how-to-configure-ssh-key-based-authentication-on-a-linux-server>.
- Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. Technical report, DTIC Document, 2004.
- Kjeld Egevang and Paul Francis. The ip network address translator (nat). Technical report, 1994.
- Alan D Eldridge and Charles W Kaufman. Method and apparatus for password based authentication in a distributed system, July 25 2000a. US Patent 6,094,721.
- Alan D Eldridge and Charles W Kaufman. Removable media for password based authentication in a distributed system, May 9 2000b. US Patent 6,061,799.
- T. Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, Jul 1985. ISSN 0018-9448. doi: 10.1109/TIT.1985.1057074.
- Kevin R Fall and W Richard Stevens. *TCP/IP illustrated, volume 1: The protocols*. Addison-Wesley, 2011.
- David F Ferraiolo and D Richard Kuhn. Role-based access controls. *arXiv preprint arXiv:0903.2171*, 2009.

- David F Ferraiolo, Ravi Sandhu, Serban Gavrila, D Richard Kuhn, and Ramaswamy Chandramouli. Proposed nist standard for role-based access control. *ACM Transactions on Information and System Security (TISSEC)*, 4(3):224–274, 2001.
- Thomas Finley. Two’s complement. 2000. URL <https://www.cs.cornell.edu/~tomf/notes/cps104/twoscomp.html>.
- S.L. Garfinkel. Digital forensics research: The next 10 years. *Science Direct*, (7):S64 – S73, 2010.
- Brodkin J Gartner. Seven cloud-computing security risks. 2008.
- Paul C Giannelli. Chain of custody and the handling of real evidence. *Am. Crim. L. Rev.*, 20:527, 1982.
- Giuliano Giova. Improving chain of custody in forensic investigation of electronic digital systems. *International Journal of Computer Science and Network Security*, 11(1):1–9, 2011.
- F Gont and SM Bellovin. Rfc 6528-defending against sequence number attacks. Technical report, Technical report, Feb, 2012.
- Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM conference on Computer and communications security*, pages 89–98. Acm, 2006.
- RJ Hanisch, A Farris, EW Greisen, WD Pence, BM Schlesinger, PJ Teuben, RW Thompson, and A Warnock. Definition of the flexible image transport system (fits). *Astronomy and Astrophysics*, 2000.
- J Heiser. What you need to know about cloud computing security and compliance. *Gartner, Research, ID, (G00168345)*, 2009.
- John Horswell and Craig Fowler. Associative evidence—the locard exchange principle. *The Practice Of Crime Scene Investigation*, page 45, 2004.

- <http://checkip.dyndns.org>, 2016. URL <http://checkip.dyndns.org>.
- <https://sourceforge.net/projects/upnp-portmapper/>, 2016. URL <https://sourceforge.net/projects/upnp-portmapper/>.
- <http://truecrypt.sourceforge.net/>, 2016. URL <http://truecrypt.sourceforge.net/>.
- IANA. Internet assigned numbers authority, 2015. URL www.iana.org.
- ICANN. Whois. 2017. URL <https://whois.icann.org/en/technical-overview>.
- InfoSec_Institute. Differences between the privacy laws in the eu and the us, May 2015. URL <http://resources.infosecinstitute.com/differences-privacy-laws-in-eu-and-us/>.
- Paul T Jaeger, Jimmy Lin, Justin M Grimes, and Shannon N Simmons. Where is the cloud? geography, economics, environment, and jurisdiction in cloud computing. *First Monday*, 14(5), 2009.
- Tariq Jamil. The rijndael algorithm. *IEEE potentials*, 23(2):36–38, 2004.
- Peter Johnson. Mobile computing, 1996.
- Ari Juels and Burton S Kaliski Jr. Pors: Proofs of retrievability for large files. In *Proceedings of the 14th ACM conference on Computer and communications security*, pages 584–597. ACM, 2007.
- Rafael Accorsi Keyun Ruan. Challenges of cloud forensics: A survey of the missing capabilities. *ERCIM News*, 90:32–32, 2012.
- Ryan KL Ko, Peter Jagadpramana, and Bu Sung Lee. Flogger: A file-centric logger for monitoring file access and transfers within cloud computing environments. In *2011IEEE 10th International Conference on Trust, Security and Privacy in Computing and Communications*, pages 765–771. IEEE, 2011a.

- Ryan KL Ko, Peter Jagadpramana, Miranda Mowbray, Siani Pearson, Markus Kirchberg, Qianhui Liang, and Bu Sung Lee. Trustcloud: A framework for accountability and trust in cloud computing. In *2011 IEEE World Congress on Services*, pages 584–588. IEEE, 2011b.
- Paul J Leach, Michael Mealling, and Rich Salz. A universally unique identifier (uuid) urn namespace. 2005. RFC 4122.
- Xueping Liang, Sachin Shetty, Deepak Tosh, Charles Kamhoua, Kevin Kwiat, and Laurent Njilla. Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability. In *Proceedings of the 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, pages 468–477. IEEE Press, 2017.
- Yehuda Lindell. Anonymous authentication. *Journal of Privacy and Confidentiality*, 2(2):4, 2007.
- Rongxing Lu, Xiaodong Lin, Xiaohui Liang, and Xuemin Sherman Shen. Secure provenance: the essential of bread and butter of data forensics in cloud computing. In *Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security*, pages 282–292. ACM, 2010.
- Wai-pai Lu and Malur K Sundareshan. Secure communication in internet environments: A hierarchical key management scheme for end-to-end encryption. *Communications, IEEE Transactions on*, 37(10): 1014–1023, 1989.
- Peter Macko, Marc Chiarini, Margo Seltzer, and SEAS Harvard. Collecting provenance via the xen hypervisor. In *TaPP*, 2011.
- Alexandros Marinos and Gerard Briscoe. Community cloud computing. In *IEEE International Conference on Cloud Computing*, pages 472–484. Springer, 2009.

- P. Mell and T. Grance. The nist definition of cloud computing. In *Information Technology Laboratory - Computer Security Division*, September 2011.
- D. Mills, U. Delaware, J. Ed. Martin, J. Burbank, and K. Kasch. Network time protocol version 4: Protocol and algorithms specification. June 2010. URL <http://tools.ietf.org/html/rfc5905>.
- Paul Mockapetris and Kevin J Dunlap. *Development of the domain name system*, volume 18. ACM, 1988.
- Scott A Moss. Litigation discovery cannot be optimal but could be better: The economics of improving discovery timing in a digital age. *Duke Law Journal*, pages 889–954, 2009.
- David C Mowery and Timothy Simcoe. Is the internet a us invention?—an economic and technological history of computer networking. *Research Policy*, 31(8):1369–1387, 2002.
- Kiran-Kumar Muniswamy-Reddy and Margo Seltzer. Provenance as first class cloud data. *ACM SIGOPS Operating Systems Review*, 43(4):11–16, 2010.
- Kiran-Kumar Muniswamy-Reddy, Uri Braun, David A Holland, Peter Macko, Diana Maclean, Daniel Margo, Margo Seltzer, and Robin Smogor. zhang2011track. In *Proceedings of the 2009 USENIX Annual Technical Conference*, 2009.
- Jude Nelson, Muneeb Ali, Ryan Shea, and Michael J Freedman. Extending existing blockchains with virtualchain. In *Workshop on Distributed Cryptocurrencies and Consensus Ledgers (DCCL'16), (Chicago, IL)*, 2016.
- Michael Nofer, Peter Gomber, Oliver Hinz, and Dirk Schiereckh. Blockchain. *Business Information Systems Engineering*, 59:183–187, June 2017.

- Steve Omohundro. Cryptocurrencies, smart contracts, and artificial intelligence. *AI matters*, 1(2):19–21, 2014.
- Oracle. Server socket. 2016. URL <https://docs.oracle.com/javase/7/docs/api/java/net/ServerSocket.html>.
- Igor Ostrovsky. Why computers represent signed integers using two's complement. 2010. URL <http://igoro.com/archive/why-computers-represent-signed-integers-using-twos-complement/>.
- Oxford. Oxford dictionaries: Spooof, 2016. URL <http://www.oxforddictionaries.com/definition/english/spooof>.
- G. Palmer. A road map for digital forensic research. Technical Report DTR - T001-01 FINAL, First Digital Forensic Research Workshop (DFRWS), August 2001.
- José Luis Gómez Pardo. Private-key encryption. In *Introduction to Cryptography with Maple*, pages 131–179. Springer, 2013.
- Alan P Parkes. Elements of formal languages. In *A Concise Introduction to Languages and Machines*, pages 11–42. Springer, 2008.
- C. P. Pfleeger and S. L. Pfleeger. *Security in Computing Fourth Edition*. Prentice Hall, 2006. ISBN 0-13-239077-9.
- Jon Postel. Rfc 768: User datagram protocol. *User datagram protocol*, pages 1–3, 1980.
- Jon Postel. Rfc 791: Internet protocol. 1981.
- Jon Postel. Rfc 793: Transmission control protocol, september 1981. *Status: Standard*, 88, 2003.
- Wolfgang Pree. Blockchain: Technology and applications. 2016.
- Alan Presser, Lee Farrell, Devon Kemp, and W Lupton. Upnp device architecture 1.1. In *UPnP Forum*, volume 22, 2008.

- Putty. Putty, 2016. URL <http://www.putty.org/>.
- Vincent Rainardi. Functional and nonfunctional requirements. *Building a Data Warehouse: With Examples in SQL Server*, pages 61–70, 2008.
- Indrakshi Ray and Mahendra Kumar. Towards a location-based mandatory access control model. *Computers & Security*, 25(1):36–44, 2006.
- D Reilly, Chris Wren, and Tom Berry. Cloud computing: Forensic challenges for law enforcement. In *Internet Technology and Secured Transactions (ICITST), 2010 International Conference for*, pages 1–7. IEEE, 2010.
- Yakov Rekhter, Bob Moskowitz, Daniel Karrenberg, Geert Jan de Groot, and Eliot Lear. Address allocation for private internets. Technical report, 1996.
- Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.
- R. Rowlingson. A ten step process for forensic readiness. *International Journal of Digital Evidence*, 2, 2004.
- Joni Salonen. Why we use two’s complement. 2013. URL <http://jonisalonen.com/2013/why-we-use-2s-complement/>.
- Ravi S Sandhu, Edward J Coyne, Hal L Feinstein, and Charles E Youman. Role-based access control models. *Computer*, 29(2):38–47, 1996a.
- Ravi S Sandhu, Edward J Coyne, Hal L Feinstein, and Charles E Youman. Role-based access control models. *Computer*, 29(2):38–47, 1996b.
- John C Schmitt and Dale R Setlak. Access control system including fingerprint sensor enrollment and associated methods, May 11 1999. US Patent 5,903,225.

- Yuliang Shi, Kun Zhang, and Qingzhong Li. A new data integrity verification mechanism for saas. In *Web Information Systems and Mining*, pages 236–243. Springer, 2010.
- Yogesh L Simmhan, Beth Plale, and Dennis Gannon. A survey of data provenance in e-science. *ACM Sigmod Record*, 34(3):31–36, 2005.
- William Allen Simpson. Ppp challenge handshake authentication protocol (chap). 1996. URL <https://www.ietf.org/rfc/rfc1994.txt>.
- Ronggong Song. Advanced smart card based password authentication protocol. *Computer Standards & Interfaces*, 32(5):321–325, 2010.
- Peter Stephenson. Modeling of post-incident root cause analysis. *International Journal of Digital Evidence*, 2(2):1–16, 2003.
- Qixiang Sun. The dmca anti-circumvention provisions and the region coding system: Are multi-zone dvd players illegal after the chamberlain and lexmark cases. *U. Ill. JL Tech. & Pol’y*, page 317, 2005.
- J. Tan. Forensic readiness. *Cambridge, MA:@ Stake*, 2001.
- Yu Shyang Tan, Ryan KL Ko, Peter Jagadpramana, Chun Hui Suen, Markus Kirchberg, Teck Hooi Lim, Bu Sung Lee, Anurag Singla, Ken Mermoud, Doron Keller, et al. Tracking of data leaving the cloud. In *2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications*, pages 137–144. IEEE, 2012.
- TechTarget.com. Network time protocol (ntp). 2017a. URL <http://searchnetworking.techtarget.com/definition/Network-Time-Protocol>.
- TechTarget.com. What is geo-fencing. 2017b. URL <http://whatis.techtarget.com/definition/geofencing>.
- TimeAndDate.com. The difference between gmt and utc. 2017. URL <https://www.timeanddate.com/time/gmt-utc-time.html>.

- Townsend. The full life-cycle of keys. 2017. URL <https://info.townsendsecurity.com/definitive-guide-to-encryption-key-management-fundamentals>.
- Philip M Trenwith and Hein S Venter. A digital forensic model for providing better data provenance in the cloud. In *Information Security for South Africa, 2013*, pages 1–6. IEEE, 2014.
- Philip M Trenwith and HS Venter. Digital forensic readiness in the cloud. In *Information Security for South Africa, 2013*, pages 1–5. IEEE, 2013.
- Verio. 10 benefits of cloud computing. 2012. URL <http://www.verio.com/resource-center/articles/cloud-computing-benefits/>.
- whatismyipaddress.com. What is an rir. 2017. URL <http://whatismyipaddress.com/rir>.
- Sue Wilkinson and D Haagman. Good practice guide for computer-based electronic evidence. *Association of Chief Police Officers*, 2010.
- Ming Xue and Changjun Zhu. The socket programming and software design for communication based on client/server. In *Circuits, Communications and Systems, 2009. PACCS'09. Pacific-Asia Conference on*, pages 775–777. IEEE, 2009.
- Chen Yang, Wenping Ma, Benxiong Huang, and Xinmei Wang. Password-based access control scheme with remote user authentication using smart cards. In *Advanced Information Networking and Applications Workshops, 2007, AINAW'07. 21st International Conference on*, volume 2, pages 448–452. IEEE, 2007.
- Olive Qing Zhang, Markus Kirchberg, Ryan KL Ko, and Bu Sung Lee. How to track your data: The case for cloud computing provenance. In *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*, pages 446–453. IEEE, 2011.

-
- P. Zimmermann. A proposed standard format for rsa cryptosystems. *Computer*, 19(9):21–34, Sept 1986. ISSN 0018-9162. doi: 10.1109/MC.1986.1663326.

Appendix A

Client Code

The images labeled A contain client code.

```

private String sSecureFolder =
    @"C:\Users\0400490\Documents\Philip\Studies\Prototype\Secure";
[DllImport("User32.dll")]
protected static extern int SetClipboardViewer(int hWndNewViewer);

[DllImport("User32.dll", CharSet = CharSet.Auto)]
public static extern bool ChangeClipboardChain(IntPtr hWndRemove,
    IntPtr hWndNewNext);

[DllImport("user32.dll", CharSet = CharSet.Auto)]
public static extern int SendMessage(IntPtr hwnd, int wMsg,
    IntPtr wParam,
    IntPtr lParam);

IntPtr nextClipboardViewer;

/**
 * Monitor the clipboard for data copied from the Secure directory
 */
protected override void WndProc(ref System.Windows.Forms.Message m)
{
    // defined in winuser.h
    const int WM_DRAWCLIPBOARD = 0x308;
    const int WM_CHANGECHAIN = 0x030D;

    switch (m.Msg)
    {
        // clipboard
        case WM_DRAWCLIPBOARD:
            DisplayClipboardData();
            SendMessage(nextClipboardViewer, m.Msg, m.WParam,
                m.LParam);

            break;

        case WM_CHANGECHAIN:
            if (m.WParam == nextClipboardViewer)
                nextClipboardViewer = m.LParam;
            else
                SendMessage(nextClipboardViewer, m.Msg, m.WParam,
                    m.LParam);

            break;
        // end-clipboard
        default:
            base.WndProc(ref m);
            break;
    }
}

```

Figure A.0.1: Override WndProc in order to monitor the Windows clipboard

```
private void DisplayClipboardData()
{
    try
    {
        IDataObject iData = new DataObject();
        iData = Clipboard.GetDataObject();

        if (iData.GetDataPresent(DataFormats.FileDrop))
        {
            StringCollection newDropList = new StringCollection();
            var dropList = Clipboard.GetFileDropList();
            String files = "";
            for (int i = 0; i < dropList.Count; i++)
            {
                if (isFolderSecure(dropList[i] + ""))
                {
                    files += "YOU ARE NOT ALLOWED TO COPY THIS: " +
                        dropList[i] + "\r\n";
                }
                else
                {
                    newDropList.Add(dropList[i]);
                }
            }
            Clipboard.Clear();
            if (newDropList.Count > 0)
            {
                Clipboard.SetFileDropList(newDropList);
            }
            log(files);
        }
    }
    catch (Exception e)
    {
        MessageBox.Show(e.ToString());
    }
}
```

Figure A.0.2: Reading and monitoring clipboard data

Appendix B

Provenance Data

The listings labeled B contain provenance data.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <Provenance>
3   <IndexRecord>
4     <Index>
5       <LastRecordNumber>4</LastRecordNumber>
6       <LastRecordHashCode>
7         +mxqVgkt31th+qOPThyR2dcDlaBpcfQkkaD1X1F1ipg=
8       </LastRecordHashCode>
9     </Index>
10    <DigitalSignature>
11      QZk9BIHWuJQtcd0phaLfA3Y0oAP2jeLieFhj+1xA4uPkr
12      wnNxXrhKshu/1dWk5njH0L8j51My1dOGecuH7xExT5038
13      VKlG5tKDg75r03OErR5LATHuHCdazcQWaqC5MspyChu
14      hnWbLogRS40w/joNK8t3OpLMPwzo09ILM+q4s==
15    </DigitalSignature>
16  </IndexRecord>
17  <ProvenanceRecord>
18    <ProvenanceData>
19      <Chain/>
20      <Number>1</Number>
21      <RecordType>Location-Update</RecordType>
22      <Identity>
23        <OSDomain></OSDomain>
24        <OSUsername>Administrator</OSUsername>
```



```
25         <UserAccount>ptrenwith@gmail.com</UserAccount>
26         <MachineName>SERVER</MachineName>
27     </Identity>
28     <Location>
29         <IPAddress>178.69.23.16</IPAddress>
30         <Jurisdiction>EU</Jurisdiction>
31     </Location>
32     <Timestamp>1478007526440</Timestamp>
33     <HashCode>
34         B4/yLzwXujheU0IVAijvhtl5DdyUMeVGII9EKpCQK7Y=
35     </HashCode>
36 </ProvenanceData>
37 <DigitalSignature>
38     fl+z2Q9e0xLCihes3/LpdS2r6I4xcgo5AgWE9IL3+hI1Kim8vzmn
39     6yG6NUExarzoigfg+6Bk2Y3/YU1R7e7ZnjkCNaTradnstAy8hD4
40     /2Ru+L5Y17CcbGGDpeDGGVjjoCK4iLPhHJriNSYd4NyW7BR
41     C9dgZJnj/7WQsfzvunq+k==
42 </DigitalSignature>
43 </ProvenanceRecord>
44 <ProvenanceRecord>
45     <ProvenanceData>
46         <Chain>
47             q/5v6ON/93QmVuClwqjiz9K6COiD058w6svZjxEpmro=
48         </Chain>
49         <Number>2</Number>
50         <RecordType>Location-Update</RecordType>
51     <Identity>
52         <OSDomain></OSDomain>
53         <OSUsername>Philip</OSUsername>
54         <UserAccount>ptrenwith@gmail.com</UserAccount>
55         <MachineName>203n-PhilipMT3</MachineName>
56     </Identity>
57     <Location>
58         <IPAddress>169.1.2.3</IPAddress>
59         <Jurisdiction>ZA</Jurisdiction>
60     </Location>
61     <Timestamp>1478007526902</Timestamp>
62     <HashCode>
63         d0nZlW+87STz7ksmjC/7l15pvce+g4qO22o1MevAMGI=
```

```
64     </HashCode>
65   </ProvenanceData>
66   <DigitalSignature>
67     csW8XAk5ma9/LWa34mqHxnb7CsENWH/+QpLrbVMTxs+99
68     P8mMGTCPCyA2rBT6oX+BZYgWRChWW542yJIDym5g5N
69     n0utW9r8p2UVpvjgLWGCYvFio5Xtvt1HY2WwB/vvVPZqAdG
70     pn4Z7APuIQsmKZfthQ6GO8qK4F/ebLiRG2cc==
71   </DigitalSignature>
72 </ProvenanceRecord>
73 <ProvenanceRecord>
74   <ProvenanceData>
75     <Chain>
76       IQfTbh8ebh9LFcBRvR5SiFQS6IIUYITjj94kvzwOp5Y=
77     </Chain>
78     <Number>3</Number>
79     <RecordType>Modification</RecordType>
80     <Identity>
81       <OSDomain></OSDomain>
82       <OSUsername>Philip</OSUsername>
83       <UserAccount>ptrenwith@gmail.com</UserAccount>
84       <MachineName>203n-PhilipMT3</MachineName>
85     </Identity>
86     <Location>
87       <IPAddress>169.1.2.3</IPAddress>
88       <Jurisdiction>ZA</Jurisdiction>
89     </Location>
90     <Timestamp>1478007526918</Timestamp>
91     <HashCode>
92       t3Qar5xaOreeHzdeq6OKF56L9EOfMeyCN6MX9XBbfmE=
93     </HashCode>
94   </ProvenanceData>
95   <DigitalSignature>
96     VaRG+MyBoHketRnDEZPv1xsx2hWK5V9qVQD8qPg9aZcWQY
97     XT60H6QjAP18VPjVROCN8wkOrY4LpKy+cBXaBXtNqsxUgr
98     vP3AguGU2U2jNl2JmxYcxhmOurFuZlbgRJO8lgKX4HsKJQ8Nif
99     ZhguQLDqmFVzGCE7XbIBt7PT6Vwc==
100  </DigitalSignature>
101 </ProvenanceRecord>
102 <ProvenanceRecord>
```

```

103     <ProvenanceData>
104         <Chain>
105             nK5hdMLpRpbr5wFziq5/VE9smqt/b8YqARndmmTG7so=
106         </Chain>
107         <Number>4</Number>
108         <RecordType>Location-Update</RecordType>
109         <Identity>
110             <OSDomain></OSDomain>
111             <OSUsername>Administrator</OSUsername>
112             <UserAccount>ptrenwith@gmail.com</UserAccount>
113             <MachineName>SERVER</MachineName>
114         </Identity>
115         <Location>
116             <IPAddress>178.69.23.16</IPAddress>
117             <Jurisdiction>EU</Jurisdiction>
118         </Location>
119         <Timestamp>1478007526930</Timestamp>
120         <HashCode>
121             vT7eaLONmcZXVzOsdepf96DSTFogz323Eq1d9VVsbF0=
122         </HashCode>
123     </ProvenanceData>
124     <DigitalSignature>
125         O8oA88GEIM0gf1fLYywJxLUklCUjY0ctemzQSnGPJlJtNOP
126         ykAX8zGltplyOL4ZJJ9ChU0J98+EdYn3Ytu9ukrMaozwGOF+
127         V0QS/yXxw+1lr6PIK0uxr26yUPSBsO/b6sBZ1FzBUdvsPeK0p
128         /olglH37sQjrAN8W9g0OwjZl8A==
129     </DigitalSignature>
130 </ProvenanceRecord>
131 </Provenance>

```

Listing B.1: Valid Provenance Chain

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <Provenance>
3     <IndexRecord>
4         <Index>
5             <LastRecordNumber>4</LastRecordNumber>
6             <LastRecordHashCode>
7                 +mxqVgkt31th+qOPThYR2dcDlaBpcfQkkaD1X1F1ipg=
8             </LastRecordHashCode>

```

```
9      </Index>
10     <DigitalSignature>
11         QZk9BIHWuJQtcdr0phaLfa3Y0oAP2jeLieFhj+1xA4uPkr
12         wnNxXrhKshu/1dWk5njH0L8j51My1dOGecuH7xExT5038
13         VKlG5tKDg75r03OErR5LATHuHCdazcQWaqC5MspyChu
14         hnWbLogRS40w/joNK8t3OpLMPwzo09ILM+q4s==
15     </DigitalSignature>
16 </IndexRecord>
17 <ProvenanceRecord>
18     <ProvenanceData>
19         <Chain/>
20         <Number>1</Number>
21         <RecordType>THIS WAS ALTERED</RecordType>
22         <Identity>
23             <OSDomain></OSDomain>
24             <OSUsername>Administrator</OSUsername>
25             <UserAccount>ptrenwith@gmail.com</UserAccount>
26             <MachineName>SERVER</MachineName>
27         </Identity>
28         <Location>
29             <IPAddress>178.69.23.16</IPAddress>
30             <Jurisdiction>EU</Jurisdiction>
31         </Location>
32         <Timestamp>1478007526440</Timestamp>
33         <HashCode>
34             B4/yLzwXujheU0IvAIjvhtl5DdyUMeVGII9EKpCQK7Y=
35         </HashCode>
36     </ProvenanceData>
37     <DigitalSignature>
38         f1+z2Q9e0xLCihes3/LpdS2r6I4xcgo5AgWE9lL3+hI1Kim8vzmn
39         6yG6NUExarzoigfg+6Bk2Y3/YU1R7e7ZnjkcNaTradnstAy8hD4
40         /2Ru+L5Y17CcbGGDpeDGGVjjoCK4iLPhHJriNSYd4NyW7BR
41         C9dgZJnj/7WQSfzvunq+k==
42     </DigitalSignature>
43 </ProvenanceRecord>
44 <ProvenanceRecord>
45     <ProvenanceData>
46         <Chain>
47             q/5v6ON/93QmVuClwqjiz9K6COiD058w6svZjxEpmro=
```

```
48     </Chain>
49     <Number>2</Number>
50     <RecordType>Location-Update</RecordType>
51     <Identity>
52         <OSDomain></OSDomain>
53         <OSUsername>Philip</OSUsername>
54         <UserAccount>ptrenwith@gmail.com</UserAccount>
55         <MachineName>203n-PhilipMT3</MachineName>
56     </Identity>
57     <Location>
58         <IPAddress>169.1.2.3</IPAddress>
59         <Jurisdiction>ZA</Jurisdiction>
60     </Location>
61     <Timestamp>1478007526902</Timestamp>
62     <HashCode>
63         d0nZlW+87STz7ksmjC/7115pvce+g4qO22o1MevAMGI=
64     </HashCode>
65 </ProvenanceData>
66 <DigitalSignature>
67     csW8XAk5ma9/LWa34mqHxnb7CsENWH/+QpLrbVMTxs+99
68     P8mMGTCPCGCyA2rBT6oX+BZYgWRChWW542yJIDym5g5N
69     n0utW9r8p2UVpvjgLWGCYvFio5Xtvt1HY2WwB/vvVPZqAdG
70     pn4Z7APuIQsmKZfthQ6GO8qK4F/ebliRG2cc==
71 </DigitalSignature>
72 </ProvenanceRecord>
73 <ProvenanceRecord>
74     <ProvenanceData>
75         <Chain>
76             lQfTbh8ebh9LfcBRvR5SiFQS6IIUYITjj94kvzwOp5Y=
77         </Chain>
78         <Number>3</Number>
79         <RecordType>Modification</RecordType>
80         <Identity>
81             <OSDomain></OSDomain>
82             <OSUsername>Philip</OSUsername>
83             <UserAccount>ptrenwith@gmail.com</UserAccount>
84             <MachineName>203n-PhilipMT3</MachineName>
85         </Identity>
86         <Location>
```

```
87         <IPAddress>169.1.2.3</IPAddress>
88         <Jurisdiction>ZA</Jurisdiction>
89     </Location>
90     <Timestamp>1478007526918</Timestamp>
91     <HashCode>
92         t3Qar5xaOreeHzdeq6OKF56L9EOfMeyCN6MX9XBbfmE=
93     </HashCode>
94 </ProvenanceData>
95 <DigitalSignature>
96     VaRG+MyBoHketRnDEZPv1xsx2hWK5V9qVQD8qPg9aZcWQY
97     XT60H6QjAP18VPjVROCN8wkOrY4LpKy+cBXaBXtNqsxUgr
98     vP3AguGU2U2jN12JmxYcxhmOurFuZlbgRJO8lgKX4HsKJQ8Nif
99     ZhguQLDqmFVzGCE7XbIBt7PT6Vwc==
100 </DigitalSignature>
101 </ProvenanceRecord>
102 </Provenance>
```

Listing B.2: Invalid Provenance Chain