INTERNATIONAL JOURNAL OF ADVANCE RESEARCH, IDEAS AND INNOVATIONS IN TECHNOLOGY

ISSN: 2454-132X
Impact factor: 4.295
(Volume 5, Issue 1)
Available online at: www.ijariit.com

Data Protection in the Cloud

By: Kumar Kishan Chandra,
Supriya Raj, Dr. Anand Kumar Pandey

Contents

Chapter 1

Introduction

1.1 Motivation: Alice and her Suitors

The use of encryption schemes is often described through an analogy depicting the transmission on a plain-text message M from one entity, Alice to another entity, Bob. Here Alice wishes to ensure that only Bob will be able to read M. This analogy has persisted due to its ability to describe a prevalent communication style, that of unicast communication. However, this simple analogy does not necessarily represent the entirety of communication styles that are actively used, it does not take into account multicast communication: What if Alice's wish were to send her message not to Bob but to Bobs plural?

Traditional symmetric and asymmetric encryption schemes can be leveraged to provide Alice with a secure means through which she can send her message. However, with symmetric schemes each recipient will be in a position to decrypt all cipher-texts that have been encrypted with the same key: Access is too coarse-grained. With asymmetric schemes the encrypting entity needs to explicitly state for whom decryption is permissible: Access is too ne-grained. To reference the di erent styles of communication, symmetric schemes represent broadcast communication and asymmetric schemes unicast communication. A multicast encryption scheme is required that allows for a more expressive ne-grained means through which Alice can specify access over her data.

A promising solution to `multicast' encryption is that of Predicate Based Encryption (PBE). PBE is a novel family of asymmetric encryption schemes in which decryption of a cipher-text is dependent upon a set of attributes satisfying a predicate. A generalisation of both Attribute Based Encryption (ABE) [SW05] and Identity Based Encryption (IBE) [Sha85], PBE allows for selective ne-grained access control to be speci ed over encrypted data. Generally speaking, attributes are used to construct users' decryption keys and to encrypt plain-text messages. Decryption occurs when a match occurs between the attributes held by the entity (in their decryption key) and the attributes used to construct a cipher-text. This matching occurs through the use of predicates, that describe: a) the required attributes needed to decrypt; and b) the relationship between the attributes.

1.2 Cloud Computing

Cloud Computing is the name given to the recent trend in computing service provision. This trend has seen the technological and cultural shift of computing service provision from being provided locally to being provided remotely and en masse, by third-party service providers [Hay08]. Functionality such as storage, processing and other functionality is now o ered on demand, as a service and both freely and at cost [AF+09]. Data, that was once housed under a consumers own administrative and security domain, has now been extracted and placed under the domain of the Cloud Service Provider (CSP) [Pea09]. The consumer has e ectively lost control over how their data is being stored, shared and used, and also over the security used to protect their data. Moreover, it can be the case that a surreptitious employee of the service provider will have access to your data for legitimate purposes but will abuse this power for their own means [Won10]. Users are no longer in full control over the security of their data and the protection offered by the service provider is not absolute. There is a need for users to have more control over the protection of their data within the cloud: Users need to become empowered.

1.3 Research Project and Objectives

Given service users' need to regain control over their data together with PBE's ability to o er selective ne-grained access control over encrypted data. This thesis seeks to investigate:

How Predicate Based Encryption schemes can be lever-aged within the Cloud to protect data.

The investigation was divided broadly into three stages.

Data Security and the Cloud. The initial stage sought to provide a clear de nition for Cloud Computing and the security issues therein, looking to identify precisely where and when threats can occur to data and how these threats ought to be mitigated.

Predicate Based Encryption. The next stage focused solely upon PBE schemes discussing how they work and what they allow for. This provided a foundation upon which their deployment as part of a crypto-system could be explored and to de ne the types of problem that PBE schemes can be used to solve.

Leveraging PBE. The nal stage of the investigation built upon, and combined the results, of the previous stages. Here the investigation looked to determine the problems that PBE schemes can be used to solve within the Cloud, and the quality of solution provided.

1.4 Research Outcomes

From the investigation, it was determined that PBE can be used to protect data within the cloud. The main results for each stage of the investigation are outlined below.

Data Security and the Cloud. From the initial stage two threat-models were produced: one user, and the other CSP orientated. These models described the threats upon data in terms of the data lifecycle. Furthermore, it was determined that a privacy model centred around Kafka's The Trial together with the idea that the CSP provider could be trusted facilitates a better understanding of the problems present within the Cloud and how such problems can be solved.

Predicate Based Encryption. Characteristics that can be used to categorise PBE schemes were identi ed. Of which predicate placement had the greatest affect upon the access control a orded by the scheme. A generic model for deploying PBE schemes as part of a crypto-system was developed. From this model three modes of operation that characterises the deployment of a PBE scheme were identified.

Leveraging Predicate Based Encryption. Three scenarios are described that illustrate ways in which service users could specify precisely with whom they wish to share their data, for what purpose, and for how long. Furthermore, two additional scenarios are presented that would allow a service provider to facilitate keyword search over encrypted data using expressive queries supporting conjunction and disjunction of terms.

1.5 Organisation

Part I: Data Security and the Cloud Within the rst part of the thesis a de nition towards Cloud Computing is given (Chapter 2) together with an overview of the technical and legal security issues found therein|Chapter 3. Chapter 4 introduces two threat models that establish what threats can occur to data in the Cloud and where. Several security requirements that govern data in the Cloud are discussed in Chapter 5. Finally this part concludes with Chapter 6 that provides a discussion over the trappings of what makes a good solution.

Part II: Predicate Based Encryption The second part of this thesis introduces PBE, the construction of PBE schemes, and PBE schemes use as part of a cryptographic system. As with any modern encryption scheme the mathematics involved can be complex and intimidating for those not already versed in the eld. Chapter 7 provides a high-level overview towards PBE schemes and their operation. For those more versed in mathemat-ics, Chapter 8 looks at the underlying mathematics surrounding the construction of PBE schemes that use general predicates. Chapter 9 discusses the use of PBE as a part of a cryptographic system. Finally, Chapter 10 provides an evaluation over PBE.

Part III: Leveraging PBE in the Cloud The third part of this thesis discusses how PBE could be leveraged within the Cloud. Chapter 11 introduces ve scenarios and Chapter 12 evaluates them.

Part IV: Conclusion, Re ection, and Further Work Chapter 13, provides a summary over the research ndings and Chapter 14 provides information pertaining to future directions that this topic could be taken in and over related areas of interest.

Part I

Data Security within the Cloud

Chapter 2

Defining Cloud Computing

finition for Cloud Computing is given together with an overview concerning computing as a service and the bene ts from using the Cloud.

2.1 Overview

Cloud Computing is the name given to a recent trend in computing service provision. This trend has seen the technological and cultural shift of computing service provision from being provided locally to being provided remotely and en masse, by third-party service providers [Hay08]. These third-parties o er consumers an a ordable and exible computing service that consumers would otherwise not have been accessible, let alone a ord [MKL09, Chapter 1]. This new means of service provision has evolved from and is the culmination of research stemming from (among others) distributed and networked systems, utility computing, the web and software services research [Vou08; AF+09]. This paradigm shift has led to computing being seen as another household utility, aka \ fth utility", and has prompted many a business and individual to migrate parts of their IT infrastructure to the cloud and for this data to become managed and hosted by Cloud Service Providers (CSPs). However, Cloud Computing is the cause celebr among tech pundits and has led to the term `Cloud Computing' as an umbrella term being applied to di ering situations and their solutions. As such a broad range of de nitions for Cloud Computing exists, each of which di er depending on the originating authors' leaning. This chapter seeks to provide a coherent and general introduction to Cloud Computing.

2.2 Computing as a Service

One of the main tenets of Cloud Computing is the `as-a-Service' paradigm in which `some' service is o ered by a Service Provider (also known as a Cloud Service Provider) to a User (consumer) for use. This service can also be categorised according to the application domain of its deployment. Examples of application domains that o er services are: Financial e.g. Mint.com, Managerial e.g. EverNote and Analytical e.g. Google Analytics. The agreed terms of use, indicating the actions that must be taken by both the provider and consumer, are described in a contract that is agreed upon before service provision. Failure to honour this agreement can lead to denial of service for the consumer or legal liability for the service provider. This contract is often described as a Terms of Service or Service Level Agreement. Moreover, as part of this agreement the service provider will provide a Privacy Policy which outlines how the users data will be stored, managed, used and protected.

2.2.1 Service Levels

The services offered are often categorised using the SPI Service Model. This model represents the di erent layers/levels of service that can be o ered to users by service providers over the di erent application domains and types of cloud available [MKL09, Chapter 2]. Clouds can be used to provided as-a-Service: software to use, a platform to develop on, or an infrastructure to utilise. Figure 2.1 summarises the SPI Service Model.

Software as a Service The rst and highest layer is known as: Software as a Service (SaaS). It represents the applications that are deployed/enabled over a cloud by CSPs. These are mature

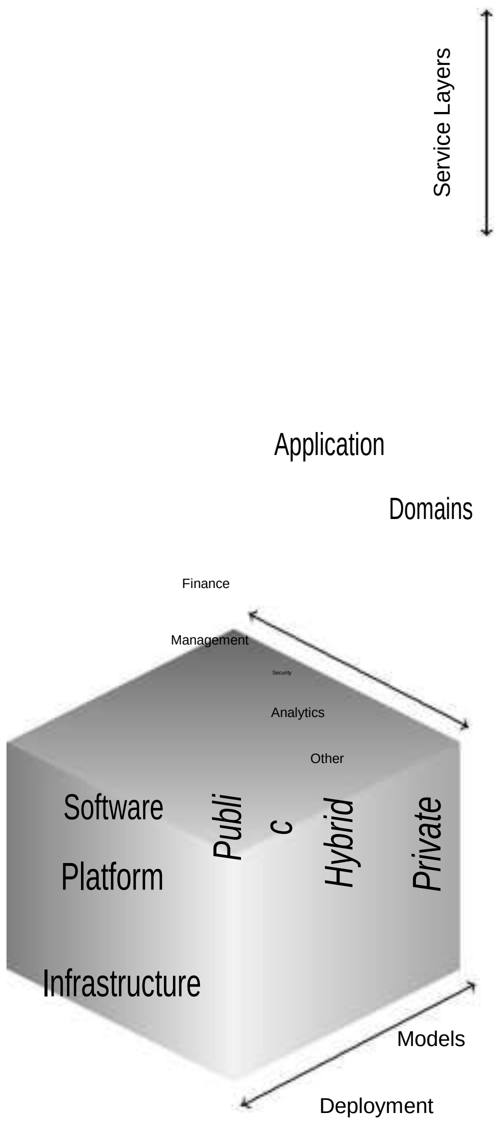


Figure 2.1: Summary of the SPI Service Model

Applications that often offer an API to allow for greater application extensibility. For instance, Google Docs can be seen as the archetypal SaaS application, it has been deployed solely within the Cloud and o ers several APIs to promote use of the application.

Platform as a Service The next layer is known as: Platform as a Service (PaaS). This represents a development platform that developers can utilise to write, deploy and manage applications that run on the cloud. This can include aspects such as development, administration and management tools, run-time and data management engines, and security and user management services. For instance, Force.com and Amazon Web Services [AWS] o ers a suite of services that allows developers to construct an application that is deployed using web-based tooling.

Infrastructure as a Service The nal and lowest layer is known as: Infrastructure as a Ser-vice (IaaS). CSP o er developers, a highly scaled and elastic computing infrastructure that are used to run applications. This infrastructure can be comprised of virtualised servers, storage, databases and other items. Two well known examples are the Amazon Elastic Compute Cloud, a commercial platform o ered as part of Amazon.com's Web Service platform and Eucalyptus, an open source platform that o ers the same functionality [AWS; NW+09].

2.2.2 Entities Involved

Cloud actors/entities can be divided into two main categories: A) CSP or Service Provider those who provide a service; and B) Cloud Service User (Users) those who use a service. Within Cloud Computing the di erences between the role played by a service provider and a user can be blurred. The service provider could also be the user of another service e.g. when infrastructure is the service. The exact de nition whether an entity is a provider or user is dependent on the context of the interaction and the service being o ered. Some service providers will o er services at all three service levels, or o er just one particular level of service and have their own internal IaaS infrastructure.

A possible refinement could be that CSP providers are either: a) Infrastructure Service Providers|those that o er IaaS and own and run the data centers that physically house the servers and software; or b) Service Providers|those that o er PaaS or SaaS services. And thatCloud Service Users are either: A) Platform Users| are users who buy into a service providers platform e.g. Facebook; and B) Consumers|are service users who use either SaaS or IaaS services.

2.3 Defining the Cloud

The term `cloud' has been used traditionally as a metaphor for networks and helps abstract over their inherent complexity. This term, however, has evolved to encompass the transparency between the technological infrastructure of the CSP and the consumers point of view. A cloud can be one of the following types:

Public Constituting publicly accessible services that are accessed over the Internet and are often described using the term \The Cloud".

Private These are private services deployed on private networks. Such clouds may also be managed by third parties.

Hybrid A combination of services o ered both privately and publicly. For example core-services may be o ered on a private cloud; other services originate from public clouds.

Vaquero, Rodero-Merino et al. provides a de nition for Clouds based upon the commonalities found among existing de nitions [VRM+09]. They de ne Clouds to be:

\. . . a large pool of easily usable and accessible virtualized resources (such as hard-ware, development platforms and/or services). These resources can be dynamically recon gured to adjust to a variable load (scale), allowing also for an optimum re-source utilization. This pool of resources is typically exploited by a pay-per-use model in which guarantees are o ered by the Infrastructure Provider by means of customized SLAs."

The provision of virtualised resources, as a service, allows for an elastic and on demand com-puting environment that can be expanded and shrunk as needed, depending on the needs of the consumer. Mather, Kumaraswamy and Latif [MKL09, Chapter 2] provides a slightly alternate de nition. This de nition is based on ve attributes that can be used to describe a cloud-based system. They are:

Multi-tenancy The sharing of resources being o ered by the service providers to the consumers at the network, host and application level.

Massive Scalability The ability to scale the storage, bandwidth and systems available to pro-portions unattainable if performed by the organisation itself.

Elasticity Consumers can rapidly and dynamically increase and decrease the resources required as needed.

Pay-as-you-go Consumers only pay for the resources they consume such as processing cycles and disk space.

Self-Provisioning of Resources Consumers have the ability for the self-provisioning of re-sources.

This is a better de nition, it does not pertain to a particular technology. However, one attribute Pay-as-you-go is in itself too constrictive as it pertains to a particular style of payment and prohibits subscription based models. A better attribute should be:

Flexible Payment Payment for resource usage is exible and consumers can be o ered di er-ent payment models dependent on their intended use of the resources.

For instance in Amazon EC2 pricing for image use is dependent on three factors: 1. Image Type| On Demand, Reserved, Spot 2. OS Used|UNIX/LINUX or Windows; and 3. Data Center Loca-tion|West Coast America, East Coast America or Ireland. Consumers are billed di erently per hour based upon these factors and other services used [AWS-Pricing]. On the other-hand Google Apps Premium Edition will cost companies $50 per user per year for their IT infrastructure solution [GooApps] and users will have a xed set of resources. The use of such models often ensures for a lower operational cost than when compared with an in-house solution.

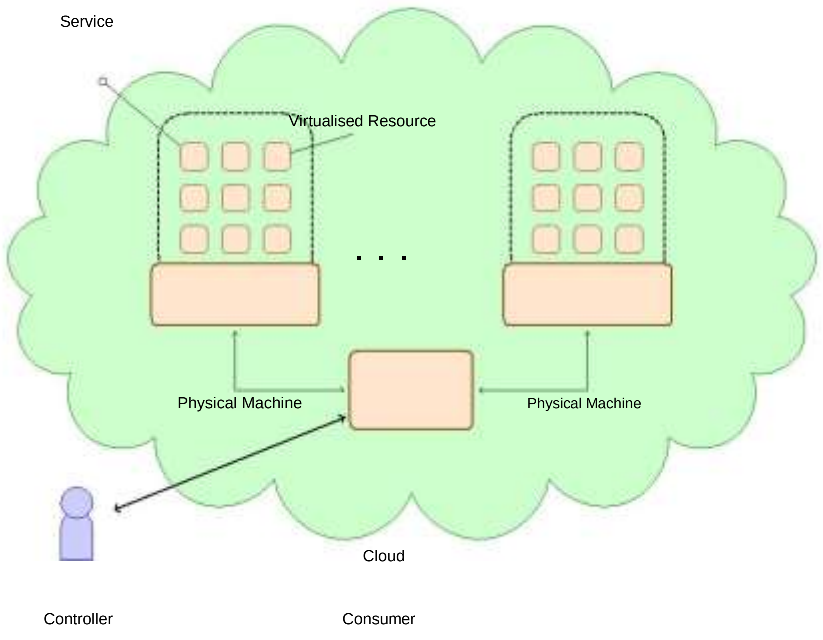


Figure 2.2: The Data Center Model for Cloud Computing.

When discussing the physical constructions of Clouds the Data Center Model is a popular choice. This model stipulates clusters of machines running specialist, dedicated hardware and software to provide the infrastructure for the large scale provision of virtualised resources [AWS; NW+09]. Though a consumer can access the virtualised resources directly, when managing their resources, consumers interact with a Cloud Controller that mediates the interaction between the consumer, and: a) the physical machines; and b) the virtualised resources owned by the consumer. Figure 2.2 illustrates a rather simpli ed view of this data center model. A more interesting realisation of clouds is the Adhoc model in which the existing (idle) computing power within an enterprise is tapped to provide the required infrastructure. The more interested reader is asked to read Kirby, Dearle et al. [KD+09] for more information concerning the adhoc cloud model.

2.4 Benefits of Cloud Computing

Many of the bene ts to be had when using Cloud Computing are the lower costs associated. At the infrastructure level, virtual images can be scaled and contracted with complete disregard for any associated hardware costs such as equipment procurement, storage, maintenance and use. This is all taken care of by the service provider and will be factored into the payment for the service: capital expenditure has been converted into operational expenditure. Resources within the cloud can be treated as a commodity, an `unlimited' medium. At both the platform and software level similar bene ts are seen. Aspects such as software installation, deployment and maintenance is virtually non-existent. This is taken care of by the provider within their own infrastructure. The service user only pays technical support.

Service providers at the SaaS level, often tout features that allow users to collaborate and interact with each other, in real-time, within the scope of the service being o ered. For example, Google Docs allows users to edit documents simultaneously and for users to see each others edits in real time. Moreover, the provision of platform and software `as a service' allows cloud service users the ability to aggregate services together either for their own use or to promote as another service i.e. Mashups. The aggregation could imply the combination of functionality from several services, or the change/combination of output from the services involved.

Remark. Service aggregation is a good example outlining how a service user can become a service provider.

chapter 3

Cloud Security Issues

Security issues to be found within the cloud from both a technical and socio-technical perspective are discussed.

3.1 Overview

Security issues come under many guises both technical and socio-technical in origin. To cover all the security issues possible within the cloud, and in-depth, would be herculean|a task not suited even for Heracles himself. Existing e orts look to provide a taxonomy over the issues seen. The Cloud Security Alliance[1] is a non-pro t organisation that seeks to promote the best practises for providing security assurance within the cloud computing landscape. In Hubbard, Sutton et al. [HS+10] the Cloud Security Alliance identify seven threats to cloud computing that can be interpreted as a classi cation of security issues found within the cloud. They are:

1. Abuse and Nefarious Use of Cloud Computing
2. Insecure Application Programming Interfaces
3. Malicious Insiders
4. Shared Technology Vulnerabilities
5. Data Loss/Leakage
6. Account, Service and Tra c Hijacking
7. Unknown Risk Pro le

For an alternate classi cation of threats to Cloud Computing, one can consult [ENISA-CC-RA09]. This chapter provides a general view of the security issues within the scope of the threats presented by the Cloud Security Alliance. This chapter concludes with an overview of the legislative aspects governing security within the cloud.

3.2 Abuse and Nefarious Use of Cloud Computing

Legitimate CSPs can be abused for nefarious purposes, supporting criminal or other untoward activities towards consumers. For instance, services can be used to host malicious code or used to facilitate communication between remote entities i.e. botnets. The emphasis is that legitimate services are used with malicious intent in mind. Other issues seen include the provision of purposefully insecure services used for data capture.

Service providers may entice potential users with o ers too good to be true. For example the promise of unlimited resources or a `30-day Free Trial'. During the registration process the consumer will be asked to provide more information than what would normally be required under the pretence of providing service personalisation e.g. location or age based advertisement. Commonly asked information includes the consumers name, email/postal address, D.O.B or even credit card details. Users are essentially being goaded to part with more information than required as a prerequisite for service use. Malicious entities can then use this information for nefarious purposes, most notably identity theft. Even if the entities are not malicious the disclosure of information to the CSP can also be said to be an abuse of service by the CSP themselves. CSPs may collect this information, or any other information provided at later stages, and market this information to third parties for data mining purposes. This is akin to store cards such as Tesco Clubcard in which the perceived benefits i.e. monetary savings, come at the cost of handing over personal information such as shopping habits.

Another security issue found is the provision of legitimate maliciously-oriented services. This service provision also known as (In) Security-as-a-Service, o ers users the same security guaran-tees as existing services yet their use is malicious. For example http://www.wpacracker.com/ is a cloud-based cracking service that can be used to `check' the security of WPA-PSK protected wireless networks. By using a cloud based service it is claimed that a process that would take around ve plus days now takes on average twenty minutes.

3.3 Insecure Interfaces and Application Programming Interfaces

Data placed in the Cloud will be accessed through Application Programming Interfaces (APIs) and other interfaces. Malfunctions and errors in the interface software, and also the software used to run the Cloud, can lead to the unwanted exposure of user’s data and impugn upon the data's integrity. For example a (fixed) aw in Apache, a popular HTTP server, allowed an attacker to gain complete control over the web server [Ho10]. Data exposure can also occur when a software malfunction affects the access policies governing user’s data. This has been seen in several Cloud based services in which a software malfunction resulted in which a user’s privacy settings were overwritten and the user data exposed to non-authorised entities [For10; Vas09]. Threats can also exist as a result of poorly designed or implemented security measures. If these measures can be bypassed, or are non-existent, the software can be easily abused by malicious entities. Regardless of the threat origin, APIs and other interfaces need to be made secure against accidental and malicious attempts to circumvent the APIs and their security measures.

3.4 Malicious Insiders

Although a CSP can be seen as being honest their employees may not be. A malicious insider is an employee of the CSP who abuses their position for information gain or for other nefarious purposes e.g. disgruntled employee. Regardless, of the employees’ motivation the worrying aspect is that a surreptitious employee will have access to consumers’ data for legitimate purposes but will abuse this power for their own means [Won10].

Another more subtle form of the malicious insider problem is through PaaS based services. If the service provider offers a platform that allows developers the ability to interact with users data i.e. Facebook Applications, users may unknowingly allow these developers access to all their data. Use of this platform may be unchecked. For example, it is well known on the Facebook Platform that once a user adds an application the application will have the ability to access all the users’ information, if allowed to do so, regardless of the applications function. With the popularity of these applications it raises the question of: How safe are Cloud based applications? Even if the application developers are not malicious this does not mean that the application cannot be hacked [Tho09; ACLU-fb-app; Per09].

3.5 Shared Technology Issues

A more interesting form of confidentiality issue relates to the construction of a cloud and the services themselves.

3.5.1 Virtualisation Issues

The underlying virtualisation architecture allows IaaS service providers the ability to host several machine images on a single server. Ristenpart, Tromer et al. [RT+09] discuss practical attacks on such services, concentrating on Amazon EC2. First, the authors showed that they could map the internal structure of the cloud, allowing them to determine if two virtual machines were co-resident with each other i.e. were running on the same physical machine. Secondly, they

were able to, purposefully, add a virtual machine to the cloud so that it was co-resident with another machine. Finally, the authors were able to show that once a machine was co-resident, they would be able to launch several attacks that would allow them to learn information regarding CPU cache use, network tra c rates and keystroke timings.

3.5.2 Service Aggregation

Aggregated services o er services based upon the functionality o ered by existing services. Often aggregated services o er the combined functionality of existing services allowing for rapid service construction. However, service aggregation presents consumers with several interesting problems [Pea09]. Data is now being shared across multiple service providers whose privacy policies will also subject to change. Under whose privacy policy is the data governed by, how to combine the two policies? Furthermore, service aggregation can occur in an ad-hoc and rapid manner implying that less stringent controls could have been applied to the protection of data, increasing the likelihood of a problem.

3.6 Data Loss or Leakage

Although insecure APIs can lead to data loss or the unwanted exposure of information, consumers can also loose their information through other means.

3.6.1 Availability Issues

Availability issues are when users data is made inaccessible to the consumer. The data has been made unavailable. Such a lack of availability can be a result of access privilege revocation, data deletion or restricting physical access to the data itself. Availability issues can be attributed to an attacker using fooding based attacks [JGL08]. For example, Denial of Services attacks, attempt to food the service with requests in an attempt to overwhelm the service and cease all of the services intended operations.

A monetary effect can also be seen with availability issues. Monetary issues affect not only the consumer but also the CSP. Recall from Section 2.3, that one of the benefits of Cloud Computing is that of Flexible Payment, consumers can be charged on a pay-as-you-go tariff. Flooding attacks will have an effect upon resource utilisation and will result in increases to power consumption, network usage and hardware maintenance. Ultimately this will also increase the amount of money the consumer will be charged for resource usage. Moreover, these monetary increases will also increase the operational expenditure of the service provider [JS09].

Fault tolerance protocols are used to combat the issue of node failure within the Cloud [Cri91]. Fault tolerance protocols replicate data across machines, and data centers, ensuring that if part of the cloud does fail a version of the data will still be available to the user. Part of the cloud will be redundant though for good measure. Data replication also requires that the data state of the data be synchronised across multiple nodes. However, poorly designed protocols can also lead to availability issues. Birman, Chockler and Renesse [BCR09] discuss that synchronisation protocols with tightly coupled nodes have more adverse a ects such as higher network usage and deadlock when compared to loosely coupled asynchronous nodes. Furthermore, the existence of multiple copies of data can also introduce con dentiality problems. The increased number of data instances also increases the likelihood that an attacker will be able to access the data.

3.6.2 Data Leakage

Another form of data leakage stems from the disclosure of information that, though hidden, is deduced from freely available information. For example: say that Bob is a member of a rather masculine society i.e. rugby club. Say that Bob is also homosexual and is actively involved with an LGBT society. Bob may wish to keep his sexuality a secret from his friends in the rugby club due to fears of being emasculated. In this situation Bob will wish to keep his on-line activities involving these two facets of his private life separate. However, this does not mean that never the twain shall never meet. In Chew, Balfanz and Laurie [CBL08] the authors discuss ways in which the privacy of social networking sites can be undermined. The authors describe how such personal information e.g. Bob's sexuality, can be disclosed from unwelcome linkage and also through social graph merger.

When users interact with a service they can leave a public trail, be it from status/update messages or through new postings. Unwelcome linkage occurs when new information is discerned about an individual through analysis of the individuals public trail i.e. links. This unwelcome linkage could be accidental or the result of the individual not covering their tracks. That is Bob may keep a private, anonymous blog detailing his frustration with being homosexual in the rugby club and use a photo from an online photo account that is linked to his real identity. This is accidental linkage. Similarly, a friend of Bob's from the LGBT society could place a link to Bob's blog within their own blog and refer to Bob's real identity. This is defined as trackback linkage from Chew, Balfanz and Laurie [CBL08].

Social graph merging is similar to unwelcome linkage however the links formed occur through the aggregation of social graphs. A social graph is a graph describing a person's social information such as friends, groups and interests.

Through combination of a persons social graphs from separate social networking sites, or the social graphs of people from the same social network site, new information can be deduced. Continuing with the example of Bob used earlier, it is feasible that through the analysis of Bob's social graph, and the social graphs of his friends, Bob's sexual orientation can be deduced. This attack to determine the sexual orientation of a person and the ability to `out' them was shown to be feasible in Jernigan and Mistree [JM09]. The authors determined that the more homosexual friends that an individual has the higher the probability that the person was also homosexual. Furthermore, both Narayanan and Shmatikov [NS09] and Wondracek, Holz et al. [WH+10] demonstrated practical attacks that analyse a persons social graph and allow the persons identity to be revealed i.e. de-anonymised.

3.7 Account or Service Hijacking

When communicating with the CSP malicious entities may seek to affect the integrity and authenticity of the user's communication with the CSP and vice versa. There are several ways in which the integrity and authenticity of a users session can be impugned [JS+09, Section 3.1{ 2]. The rise of `Web 2.0', has seen the web browser becoming increasingly used as a means to access remote services. Browser-based interfaces and authentication are used by consumers to establish a session with their service provider. A malicious entity can attempt to capture or hijack this session or steal the users credentials to access or in uence the users data, from within the browser. Most browsers operate on a Same Origin Policy where client scripts are allowed to access resources if they share the same origin. However, attacks such as Cross Site Scripting [XSS02] and Cross Site Request Forgery [XSRF10], as well as DNS Poisoning [Lem08] can be used to undermine this security feature. Other issues found include the manipulation of the data being sent within the sessions. For example the XML Signature Element Wrapping attack [MA05] targets protocols that make use of WS-Security. The attacker will capture an XML-Signed SOAP message and modify it such that the original body is placed within the SOAP header and in its place is the malicious payload. When checked by the recipient the original signature will still hold.

The effects of breaking session integrity are two-fold, for one the attacker will be able to steal the identity of their victim, and secondly impugn the reputation of the victim through falsi ed data. Such man-in-the-middle attacks will have lasting repercussions such as violation of the services terms of use, or criminal. For instance, the hacking of many a celebrities twitter account [Arr09].

Remark. The man-in-the-middle attack is also an a ront to the con dentiality of data if the attacker is able to read the data as well as modify it.

3.8 Unknown Risk Pro le

Risk Management is a business process that users can use to identify and mitigate threats. It allows users to determine their current stance towards the security of their data. Auditing information such as software version, code updates, current security practises, intrusion attempts et cetera, are used as a basis for determining this stance. CSPs may not be so forthcoming with this information. Consumers, when adopting a service, must also accept the Terms and Conditions (including privacy policy) of the service, together with any Service Level Agreements made. Consumers and providers need to comply with existing laws and regulations. However the degree to which service providers adhere to current security practises and legislation, or implement them may not be clear. This leaves the consumers with an unknown risk pro le. Users are unable to determine the risk to their data as they do not have su cient information to do so.

3.9 Jurisprudence Oriented Issues

Jaeger, Lin and Grimes [JLG08] discusses the widening gap between technology and local, na-tional and international legislation. Technological innovation occurs at a much more rapid pace than the pace at which legislation can change. This rapid pace has left a cloud of ambiguity concerning how users' data can be treated legally within the cloud. Given the nature for `ever-changing' service level agreements, the terms and conditions that one originally agreed to may not be the current versions. Furthermore, the laws and regulations themselves may not be suit-able, are open for interpretation and also bounded by jurisdiction. These issues are discussed below.

3.9.1 Service Level Agreements

Users implicitly trust the service provider not to violate terms and conditions, and be in a position to securely store their information. The terms and conditions set by the service provider may not actual conform with the established policies set by the consumers organisation. Non-compliance with such policies could lead to a loss of reputation and credibility.

Service providers will also adapt their terms and conditions over time and often not inform users of these changes in an explicit manner. This presents the user with a dilemma: Either they accept terms that are not absolute or a privacy policy that they disagree with, or they do not use the service. Users may be unaware that the terms and conditions have changed and will use a service governed by policies and agreements which were not the ones originally agreed upon. Even more so, the user may be subject to peer pressure in which the service is also used by the users' peer group and used actively for collaboration. Leaving the service may have an adverse a ect upon the users interaction with their peers. Further problems arise when the users themselves fail to adhere to the conditions set out in the agreement e.g. failure to pay or the uploading of inappropriate material. What happens to the users data when this occurs? Will the data be retained and can it be extracted from the service provider? Furthermore a CSP will also change, over time, the terms and conditions attached to the service to further the CSPs own business interests [Bar10]. Users are not in full control over the security of their data and that the protection o ered by the service provider is not absolute.

3.9.2 Expectations of Privacy

The Fourth Amendment of the United States Constitution states:

\The right of the people to be secure in their persons, houses, papers, and e ects, against unreasonable searches and seizures, shall not be violated. . . "

Similarly, Article 8 of the European Convention of Human Rights states:

\Everyone has the right to respect for his private and family life, his home and his correspondence"

This has resulted in a `reasonable expectation of privacy' existing, and being guaranteed, over ones home and its contents. Couillard [Cou09] discusses this problem under the auspices of United States Common Law. In the United States the contents of ones briefcase and also school satchel is governed under this expectation of privacy. With Cloud Computing society has naturally extended this reasonable expectation of privacy to cloud services that o er similar functionality e.g. Dropbox. Such services are often described using the terminology Virtual Containers. This raises the legal problem of can data stored in the cloud afford the same level of protection as data stored within ones home. The Katz Test, used to determine ones reasonable expectation of privacy requires that: a) there was a subjective expectation of privacy over the object; and b) the expectation was reasonable. Implying that there is some form of concealment i.e. opacity, to the object in question. Within the Cloud this would imply that simple password protection or data encryption would be su cient. Couillard comments that these measures have only been upheld within case law and not in written law. Furthermore, under US Law this expectation of privacy is diminished if the data is handed over to a third-party. The third-party doctrine is that transactional data, data that contains information regarding the transaction itself, can no longer be a orded the same expectation of privacy as the contents of the transaction. The third-party needs transactional data to operate. Couillard contends that a URL can be seen as transactional data and that unlisted links i.e. dynamic ones, do not fall under the fourth amendment. This implies that the contents of a URL that may include authentication tokens and other identifying information will also not be protected by the fourth amendment.

3.9.3 National Legislation

Though there is some debate concerning expectations of privacy, existing legislation exists that CSPs should comply with. Such legislation governs the country that the CSP is operating within. Failure to comply with local legislation, or if the information of service users were to be lost, stolen or exposed then such acts could lead have potential legal rami cations for the service provider. Which in turn could result in loss of credibility, loss of reputation and possible reduction in user base [Pea09].

3.9.4 Supra-National Legislation

Service provision is not in principle bound at a national level. Users from one country commonly access services located in another. This presents service providers and consumers with problems of under whose laws must the service abide by, and to what degree. Are the consumers allowed to export their data to a foreign country? And what provisions are there for the safety of the data? Amazon.com, for example, o er their Simple Storage Service from US (Northern California), European (Ireland) and Asian (Singapore) based data centers to be able to o er services within the different regulatory frameworks of those regions. Furthermore the European Commission's Directive on Data Protection prohibits the transfer of personal data to non-EU nations.

The Safe Harbour Framework is a supra-national framework for US based companies to comply with the data protection laws of Switzerland and the European Union [SafeHarbour]. Thus providing a `safe harbour' for Swiss and EU citizens data residing within the United States. For a US company to comply it must adhere to seven principles, derived from the Directive on Data Protection, of: Notice, Choice, Onward Transfer, Access, Security, Data Integrity and Enforcement. However, there has been signi cant criticism of this framework concerning the compliance and its enforcement. It was noted in an external study [Con08] that false claims were made by US companies concerning membership and certi cation. For more information concerning the criticisms made towards the Safe Harbour Framework see Connolly [Con08]; [SEC-2002-196]; [SEC-2004-1323].

3.10 Summary

While there are many security issues to be found within the Cloud, one of the common, and underlying, issues is related to the privacy of data stored by the consumer. Such data is sus-ceptible to unwanted exposure, the manner and means of which are dependent upon level of service offered and the protection given. Data stored within the Cloud is more vulnerable and open to attacks than before; its protection is paramount and users need to regain control over the protection of their data from Cloud Service Provider.

Chapter 4

Cloud Threat Models

The origin of threats towards data within the cloud are described together with two threat models based upon said lifecycle. The rst model represents a user-centric view, the other a Cloud Service Provider point of view.

4.1 Overview

By considering the Cloud as a remote storage system i.e. NAS, one can take existing threat models (c.f Hasan, Myagmar et al. [HM+05]) that look towards the area of remote storage and adapt them, as necessary, for the Cloud. Hasan, Myagmar et al. [HM+05] provides a discussion of two threat models for remote storage systems. The rst, classi es threats based upon their e ect upon the four `classical' security requirements of con dentiality, integrity, authorisation and availability. The second classi es threats according to how the threats a ect data during its lifecycle.

The CIAA Threat Model is based upon the four classical security requirements of: Con den-tiality, Integrity, Availability and Authentication|cf. Chapter 5. Proposed threats are classi ed according to their relation to each requirement. This model is rather coarse-grained with re-spect to the classi cation of threats. Threats are categorised upon the e ect they have on each requirement and not towards the context of their implementation. A more granular model is required that considers where the threats originate and when the threat is likely to occur during service operation. A threat model based upon the data lifecycle allows for such a model to be constructed.

Remote storage systems can be viewed as a single system by consumers. The presentation of a uni ed interface through which the consumers interact can also be used to hide the complexity of the underlying infrastructure. This is an important concept and is also shared by Cloud Computing. Cloud Users can also view the Cloud as a `single' system, however, this viewpoint prohibits for a complete threat model to be established for data in the Cloud. It only provides the users with a view of the ow of their data: into the cloud and back from the cloud.

Using these concepts two di erent yet related threat models for the cloud can be constructed. Section 4.2 provides an overview of the threats in terms of origin, goal, and means. The data life-cycle is enumerated in Section 4.3. Finally, Section 4.4 presents two threat models derived from an overall perspective and from a service user perspective.

Note. The purpose of a threat model is to classify the di erent threats and vulnerabilities into groups so that they can be resolved accordingly. As Chapter 3 has already discussed possible attacks to data within the cloud, if one were to reiterate these attacks there there will be a duplication of attack descriptions. For brevity the possible threats are not described in detail, furthermore, the attacks listed are not exhaustive.

4.2 Threat Overview

Before the data life cycle threat models are introduced, some time will be spent detailing the origin and nature of the threats.

4.2.1 Origin

The origin of threats to data can be divided into the following categories:

Outsiders are entities that exist outside of the system and attempt to subvert/cir-cumvent the security infrastructure of the service or masquerade as a legitimate service to ensnare users. Their motivation will stem from simple curiosity or from malice.

Insiders more serious threats originate from current or past employees of the CSP. Employees will have intricate knowledge of the actual infrastructure including that of security and as part of their remit may have had direct access to the data itself or through other means. Similar to insiders their motive may be out of curiosity or of malice.

Natural although both insiders and outsiders can induce `errors' within the infrastructure, other errors can occur naturally from the software itself or from hardware failure. For example, when Google pushed a software update to Google Docs [Vas09]. The software malfunction changed the sharing settings of several users documents to include those with whom the affected users had share documents with before.

4.2.2 Goals

Attackers will also have a goal that drives them. This normally implies targeting a particular asset. These are resources that are either tangible or abstract respectively data and data consist-ency. Hasan, Myagmar et al. [HM+05] provides an incomplete list of what these assets maybe. One can add to this list more cloud speci c assets. An incomplete list of possibly targeted assets includes:

Communication channel	Data consistency
Storage media	Virtual image availability
Data management software	Virtual image secrecy
Data availability	Virtual image integrity
Data secrecy	Virtual Image consistency
Data integrity	Service availability

4.2.3 Means

Attackers will usually accomplish their goals by exploiting some technical exploit to gain access to the assets. This can include service hijacking, service impersonation or through poorly secured APIs. If the attacker is a malicious insider they may not need to exploit technical insecurities and will have direct access to the data or will gain access through privilege escalation.

4.3 The Lifecycle of Data

The typical lifecycle of data can be describe as the following stages:

Stage 1: Data Creation/Transmission this is the initial stage, data is created by the user and then pushed to the Cloud for consumption.

Stage 2: Data Reception data is received in the Cloud before being written to storage and logs taken of activity.

Stage 3: Output Preparation data is prepared to be returned to the consumer, this involves any transformations that needs to be performed on the data prior to its return i.e. serialisation.

Stage 4: Data Retrieval data is received by the consumer from the cloud and has now within the domain of the user.

Stage 5: Data Backup the CSP will replicate data for archival purposes. This may involve the transferral of a copy of the data to an external store.

Stage 6: Data Deletion data is permanently deleted from the cloud.

4.4 Data Lifecycle Threat Models

While the CIAA Model can be used to model threats to data, it lacks context. The Data Lifecycle Threat Model seeks to classify the threats, rst by their placement within the data lifecycle and then using the CIAA model. It is through this classi cation that one is able to provide some context to the threats and able to produce a more ne grained classi cation. Hasan, Myagmar et al. [HM+05] introduced a data lifecycle model to describe the threats associated with remote storage. This threat model can be used as a basis upon which a model for cloud resident data can be introduced.

4.4.1 Overall Model

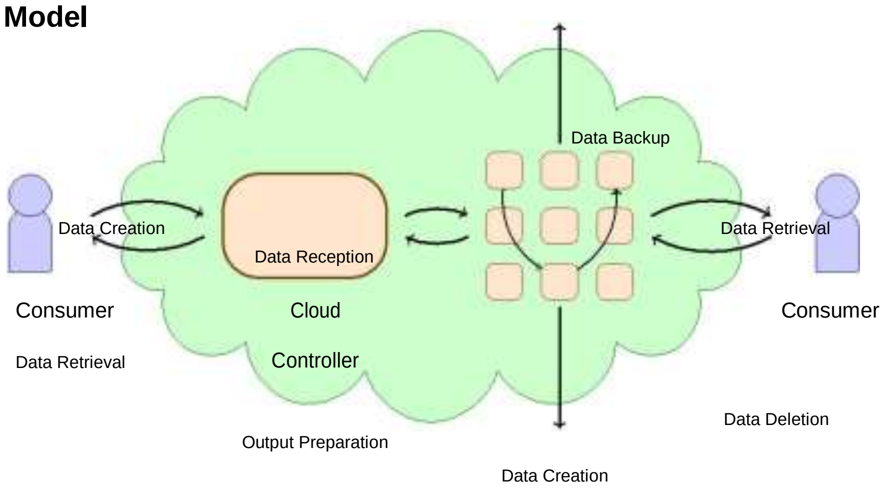


Figure 4.1: The Data Lifecycle Threat Model for the Cloud.

Recall from earlier that in the remote storage context one can assume that the data will be centrally stored within a single store. Within the cloud this cannot be guaranteed. Consumers interacting with both the Cloud Controller and also with the virtualised resources will also be presented with the appearance of a single entity. The data (including the virtual machine) may also be replicated on various virtualised resources and also among di erent physical machines. By combining this data center model with that of the data lifecycle model for remote storage systems a threat model for the cloud can be produced. Figure 4.1 summarises this new model. The di erence between this model and the existing one is that the single data store has been replace by virtual resources. This increases the number of places within the model where these stages can exist. The data creation and retrieval will always occur with the consumer. However, the data reception and output preparation stages can now occur either in the Cloud Controller, if interacting with the controller, or with one of the many virtual resources that will exist. Moreover, as a result one must add an additional stage, representing the internal migration of data within the cloud, to the data lifecycle:

Stage 7: Data Migration data is migrated from a resource inside the cloud to another for availability or scalability purposes.

The threats occurring at this new stage can be viewed as an amalgamation of the rst and fourth stages of the data lifecycle. That is the sending agent will transmit data and the receiving agent will receive the data. Furthermore, data migration can also be seen as a type of data backup and the threats to those stages can be included as well.

4.4.2 User Centric Model

The overall threat model represents an omniscient narrator's view of the cloud; they see and are privy to all aspects. However, one must also consider the user's point of view. In Chapter 3 it was established that the cloud, to a user, represents an unknown risk pro le. They are not aware of all aspects of the internal workings of the cloud. Thus, one can also present a user centric

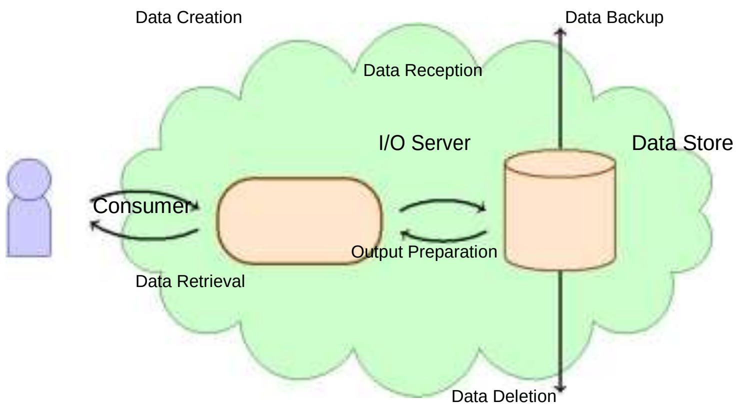


Figure 4.2: The Data Lifecycle Threat Model for Service Users.

Threat model that uses as a basis the clients' viewpoint. In this model the client views the cloud as a single entity. The more astute reader will notice that this is reminiscent of the original data lifecycle model. Data goes in the cloud, data comes out the cloud, data gets archived, data is deleted. The user is not aware per se that the data could migrate between nodes within the cloud. Figure 4.2 illustrates this user centric model.

Data Security Requirements

Grid Computing is used as a basis for de ning a set of security requirements for data in the cloud. The responsibility over providing assurances towards these requirements is also discussed.

5.1 Overview

Chapter 4 provided two threat models that one can use to look at the security of data in the Cloud. The threat models presented classi ed threats according to their relevance within the data lifecycle. This chapter looks to establish a set of security requirements for data held within the Cloud.

Grid Computing is a related paradigm to cloud computing, both seek to make available large quantities of computing resources and use them for the processing of large amounts of data [Fos01]. The security issues discussed in Chapter 3 can be used together with cloud computing's similarity with grid computing to enumerate a list of data security requirements based upon existing grid computing requirements [BM03; NR05]. Speci cally, Broadfoot and Martin [BM03] provides a detailed set of security requirements for the grid. These are used as the basis for a possible set of, though not exhaustive, security requirements that a solution for securing data in the cloud must look towards. These security requirements will aide when developing or analysing any solution, and also when describing the security o ered.

Moreover, it was highlighted that when describing the possible threats to data in the cloud two viewpoints must be taken into account: a) the users; and b) the CSP. This arose as data is either in the cloud or not in the cloud. Thus the security requirements will be the responsibility of: a) the user; b) the CSP; or c) of both the user and the CSP. This shall be addressed for each security requirement presented.

5.2 Confidentiality

The data that is to be entrusted to the cloud may be of a sensitive nature and will thus be subject to several con dentiality measures. Although con dentiality of data is primarily seen as being solved via encryption, as discussed in Broadfoot and Martin [BM03], other aspects need to be considered.

User and Resource Privacy within the Cloud, con dentiality of data also extends to how the data is being processed/used and also the users actions. The means by which CSP can store or process the data is bound by law and these laws must be adhered to. Such data includes and is not limited to: auditing records indicating access attempts and changes (and their results) to the data; properties of the data including size, access policies and origin; and even the existence of the data itself.

Deducible Data Recall from the previous Chapter in Section 3.6.2 that hidden information pertaining to an individual can be deduced from existing information i.e. Bob's sexuality. An attacker should not be able to use existing information or information relating to the con dential data i.e. meta-data to deduce any other information. Such attacks should be made as di cult as possible. For those who have multiple personae on the web relating to the di erent facets of their private life. The ability of an attacker to link the two should be hard.

The requirement of con dentiality is an aspect that both the user and CSP need to be made aware of. Speci cally, the con dentiality of the plain-text data itself should be the responsibility of the user before it goes into the cloud. Guarantees towards user and resource privacy, and deducible data is best made by the CSP. They are in a better position to provide such guarantees.

5.3 Remote Access

The Cloud, by nature, is inherently a `public place'. Services are exposed over HTTP, a public medium. Access to these services need to be controlled and access kept to authorised personnel. Moreover as the data is held remotely, trust needs to be established with the service and with the security provided by the service over the data itself. Access to the data needs to be regulated. CSPs must ensure that entities trying to access the data are not only who they say they are (authentication) but also that they have the right to do so (authorisation) This is made more di cult as CSP will be interacting with multiple users from multiple companies (domains) each of whom will require di erent management and access policies; and all done remotely.

Authentication CSPs must ensure that those trying to access the service are who they say they are. Unauthenticated users and impostors should not be able to access the data. The identity of the entities must be assured. This will imply some form of identity management.

Authorisation once the identity of an entity has been established access to the data held by the CSP needs to be regulated and controlled. Authenticated entities should not be able to access data that they are not authorised to access. For example, two users from di erent companies should not be able to access each others remote data held by the CSP unless the access has been explicitly allowed.

Location Users may be accessing the service/resource from di erent locations. Authentication of the user should always be performed and should not be linked to the device from which the entity accesses the service.

Revocation An important requirement is that of revocation. The revocation of access to individual data and to the service itself must be permissible.

With remote access, guarantees towards location privacy, authentication of identity and author-isation to the data that is resident within the cloud should be made by the CSP. Revocation, and thus assignation, of access to the data should be made by the user themselves.

5.4 Non-Repudiation

Both the CSP and user should not be able to deny the origin or refute the integrity of data. Moreover, a veri able record of the data's lifecycle should exist. The lifecycle of the data and the operations performed on the data should be attestable if a CSP attempts to defraud a user and vice versa. This is especially important if exible payment models are used. A dishonest CSP could claim that more resources were used by the user and thus be in a position to bill the consumer more than what was actually the case. This implies that records need to be kept concerning usage that can be veri ed by both users and consumers. Guarantees towards non-repudiation should be made by both the user and the CSP.

5.5 Integrity and Consistency

The mobility of data within the cloud only increases the threats that can a ect the integrity of data. Data is being transported to-and-from the user and service providers, and also internally within the cloud. The integrity of the data must be guaranteed when it has been placed within the Cloud. Consistency problems can arise from omission and commission failures. Omission failures occur when an entity fails to act upon input. Commission failures are those that occur when an entity though responds to input the output is not what was expected. It is possible for hardware to fail or connections to be lost at which time the data may be in an inconsistent state or unrecoverable. If data is replicated for some reason e.g. to combat availability, scalability or archival purposes, the consistency of the replicated data must be ensured. The consistency of replicated data can be affected by omission and commission failures. Also, consistency problems can arise during multi-author collaboration when access to the data is performed concurrently.

With regards to integrity and consistency this is a joint responsibility, though it can be divided cleanly between the two entities. The user can make guarantees towards the integrity of the data before it goes into the cloud i.e. pre-cloud insertion integrity, and the CSP needs to ensure the integrity and consistency of the data once it is in the cloud i.e. post-insertion.

5.6 Availability and Fault Tolerance

Another problem with entrusting data to a service provider is ensuring the availability of the data once it has been placed within the cloud i.e. resource availability. This is essentially a guarantee that can only be made by the CSP themselves. Users only have the assurances made by their service provider that data will be made available. If the data were to be made unavailable for some reason, users will not be able to access their data and become inconvenienced. The inconvenience caused could also lead to pro t loss for both the user and the CSP. Moreover, internally the nodes within the Cloud must also be resilient to node failure and the data held on the nodes must still be available. Internally the Cloud must be fault tolerant.

Note. Availability also overlaps with remote access requirements. Access rights to the data can also be seen as a form of availability, however this was already discussed in terms of authorisation in Section 5.3.

5.7 Summary

Below a summation of the requirements according to their division is given.

User	Joint	CSP
Data Con dentiality;	Non-repudiation	User and Resource Privacy;
Access Right Revocation;		Deducible Data;
Access Right Assignment;		User Authentication;
Pre-Cloud Insertion Integrity		Location Privacy; Post-Cloud Insertion Integ- rity and Consistency; Resource Availability; Fault Tolerance

Chapter 6

The Trappings of a Solution

A discussion towards the security of, and how best to protect data in the Cloud is provided. The notion of data privacy is addressed. The degree of trust a orded to a Cloud Service Provider is analysed. Finally, responsibilities over guarantees towards security requirements are discussed.

6.1 Overview

Over the preceding chapters the notion of data security within the cloud was discussed. One of the prevailing factors has been the unwanted exposure of data be it a result of a software malfunction or malicious CSP. When entrusting data to the cloud the data creators i.e. service users and CSPs, need assurances over access to their data. In essence data creators need to regain control over this access, data creators need to become empowered. This chapter discusses what potential solutions should aim towards when looking to empower those whom create data. More precisely, the discussion begins with a look at what this empowerment should entail (Section 6.2) before discussing the trust that can be a orded towards a CSP|Section 6.3. The responsibilities concerning the protection of data is discussed within Section 6.4. Concluding this chapter is Section 6.5 in which the salient points over what makes a solution are presented.

6.2 Defining Empowerment

Within computer security the protection of one's data can be seen as being synonymous with the protection of one's privacy. The `right to privacy' is a fundamental right and is enshrined in many a countries constitution. A user empowered over the privacy of their data can be seen as being empowered over the protection of their data. A thorough grasp of this notion of data privacy is fundamental when building a solution to empower users. As such a better solution can thus be designed and what it means for a user to become empowered can be determined.

However one of the major problems with this interpretation of data privacy, is its de nition. Data Privacy is a rather vague and often misunderstood term. For example, take three existing solutions: None of Your Business (NOYB) [GTF08]; Privacy Manager [MP09]; and Content Cloaking (CoClo) [DVZ10]. Each of these three solutions each have a di erent take on how to protect the privacy of data.

Encryption. The CoClo solution dictated the encryption of data prior to its insertion into the cloud. Data was hidden completely from unauthorised users and the CSP.

Obfuscation. With Privacy Manager data was obfuscated. While `obfuscation' does not necessarily imply the encryption of data, obfuscated data can still nonetheless be operated upon by a CSP with the CSP not learning anything about the underlying data. Examples of obfuscation techniques can be found in Kantarcioglu [Kan08].

Contextual Integrity. The NOYB solution sought to destroy the link between the data and its creator, as well as hide the data itself.

The remainder of this section will discuss each of these takes on data privacy and will discuss how a model based upon Kafka's The Trial best describes how users can become empowered within the Cloud.

6.2.1 Obfuscating Data: 1984 was not a good year

The encryption and obfuscation of data to protect privacy is by far the most common response when seeking to protect the privacy of data. All the aforementioned solutions use this technique to some degree. Obfuscation techniques are a dogmatic response to the fears envisaged by the populistic metaphor of the Orwellian `nightmare'. Originating from George Orwell's famous novel 1984, this metaphor's core theme is the harm that arises from the constant surveillance of individuals. Through the constant surveillance of an individual `Big Brother' i.e. the government, is able to use this information for both useful and nefarious purposes. In the novel the harm depicted relates to how the information was used to curb and control social activities. Naturally, this Orwellian `nightmare' has led to the view that the data collector, in this setting the CSP, is inherently malicious, seeks to cause harm and should not at all be trusted. As mentioned previously the initial response to alley these fears is to prevent the collection of data through obfuscation techniques including that of data encryption: The CSP is not trusted at all. This was noted in Chapter 3 with the unknown risk pro le presented by the CSP which in turn resulted in the user-centric threat model presented in Chapter 4. In this threat model consumers cannot make any guarantees, themselves, towards the data once it is in the cloud.

However, when using obfuscation based techniques if the CSP is not aware of the obfuscation it will not necessarily be able to correctly process the information: some functionality of the service will be lost [MP09]. This loss of functionality may also have a detrimental e ect upon the service's user experience. In D'Angelo, Vitali and Zacchirolo [DVZ10] the authors note that by using client side encryption all server side functionality such as document search will be lost. This `loss of functionality' represents a lack of trust between the service user and CSP. A di erent approach to de ning privacy and what the protection of data entails is required. Such a solution needs to bridge the gap over the control of the privacy of the data and the functionality o ered by the CSP.

6.2.2 Contextual Privacy

The NOYB solution uses a privacy model based upon the contextual integrity of data: Contextual Privacy. Originating from Nissenbaum [Nis04], the privacy of data is based upon its context. Data Privacy holds if the data cannot be linked back to the originator of the data: the data will be viewed out of context. `Big Brother' will be able to collect data and use the data but will not be able to link the data back to the data originator. Ostensibly, this appears to be madness, one's data will still be public. However, the success of the contextual privacy is dependent on what the `context' of the data is and also what the data itself is. With NOYB the original context of the users data is their Facebook pro le [GTF08]. This information is then broken down into atoms representing data items that are common to all i.e. christian name, age and gender, before being randomly distributed among other pro les. While, the person's information will still reside on Facebook it will be viewed out of context of the persons own pro le. Facebook, will thus be able to use the information but will no longer be able to link the data back to the original users. Furthermore, the non-common data items such as email addresses, telephone numbers and addresses are rst broken down into atoms representing individual characters. While this is an interesting privacy model, users' data is nonetheless publicly available.

6.2.3 A Kafkaesque Approach

Solove [Sol07] argues that a better understanding of privacy can come from not seeing privacy as a single concept but rather as a pluralistic one. The privacy of data will take on various forms depending upon the context under which the data is being examined. This resembles the views expressed in the previous section in which the privacy of data holds if the data stays `out-of-context' i.e. cannot be linked back to the data originator. The underlying metaphor used by Solove [Sol07] stems from Franz Kafka's The Trial; the approach is Kafkaesque. The Trial depicts a bureaucracy that uses people's information to make decisions about them yet prevents the peoples ability to participate in how their information is used. Much like the problems associated with users placing their data within the cloud. Here the collection of data is not the main issue, rather it is the unauthorised and unchecked processing i.e. storage, use and analysis, of the collected data.

Taxonomy of Privacy

By conceptualising privacy as what happens to data once it has been collected Solove [Sol07] demonstrates that a taxonomy of privacy violations can be constructed. The taxonomy presen-ted by Solove presents four general categories of privacy problems. The four general categories are: a) Information Collection|representing the collection of information itself; b) Information Processing|representing the use of information by the data collector; c) Information Dissemin-ation|representing the unauthorised release of information; and d) Invasion|representing the use of information to intrude upon an individuals life.

Under the Orwellian model there is only `one' category of privacy violation, that of informa-tion collection. One can also view the taxonomy by Solove as relating to the potential lifecycle of data in terms of privacy violations. First information is collected, it is then processed, dissem-inated and then used purposefully to violate ones privacy. With the typical response to privacy woes being preventing the collection of data i.e. response to the Orwellian model, this response e ectively curtails all privacy violations. However, the prevention of the collection of data is a greedy solution, it seeks to solve all problems in one fell swoop. This is not the most e ective solution, it implies loss of service functionality when using obfuscated data|cf. Section 6.2.1. With this taxonomy a new and more informed response can be constructed.

The Kafkaesque Response

The use of this taxonomy allows for the shift of the discussion concerning privacy woes away from viewing the CSP as inherently malicious i.e. it wants to collect ones data for nefarious purposes. The focus is now on the data collector as an entity that (mis)uses one's data regardless of the user's own wish. This is the comment made in The Trial. The collector need not be an overtly malevolent or benevolent entity, their character is not the main issue. The issue is the loss of control that the user has over the use of their data; they have been left powerless.

The objective is to now empower the users over the use of their data. This is a better t to the privacy woes presented by the Cloud than that of the default anti-collection response. The emphasis is letting the users take control over how their data can be used by the CSP. The empowerment stems from giving the users the choice of what happens to their data and preventing the CSP from making such choices on their behalf. However, this new approach implies that some trust needs to be placed with the CSP, to respect the users choice. In the next section this trust, between users and Cloud Service Providers, is discussed.

Remark. The important notion here is that of choice, the user should be able to choose what, if any, information should be collected and then used.

6.3 In CSP, we Trust?

Trust is an important notion. To trust an entity implies that one has con dence or faith in that entity. In the previous section the trust imparted to the data collector/CSP by the user varied and was dependent upon the view the user had of the CSP. For instance, the Orwellian approach stipulated no trust in the data collector, while the Kafkaesque approach stipulated that one could trust the CSP somewhat. From these di erences a `taxonomy of trust' can be constructed that describes how a user views and comprehends the actions of the CSP. They are:

No Trust|the Orwellian approach, one does not trust the CSP at all. Used by the CoClo solution [DVZ10].

Some Trust|the users trust the CSP to some degree. Used by the NOYB and Privacy Manager solutions [GTF08; MP09].

Complete Trust|the user completely trusts the CSP.

The remainder of this section will discuss each of these modes.

6.3.1 No Trust

With the Orwellian approach, one thought of the CSP as being inherently malicious: In CSP they did not trust. The typical response was to interact with the CSP using obfuscated data. As was discussed previously, this approach has one major drawback. When one starts to not trust the CSP with data and thus send obfuscated data, some functionality of the service will be lost if the CSP is unprepared, or such measures cannot t into operation of the service [MP09]. This is especially pertinent if one sends encrypted data. Having no trust in the CSP appears to lead to more problems than it does solutions. Moreover, from a legal point of view the transmission of obfuscated data may result in a breach of any service level agreement that the data creator has signed, or agreed upon, with their CSP.

6.3.2 Complete Trust

At the other end of the spectrum from having no trust in the CSP is to have complete trust. While this was not speci ed by any of the solutions mentioned thus far, its use has been seen. Grendel is a complete service-side solution touted by Hedlund [Hed10]. Grendel stores users data in an encrypted format and only decrypts the data when it is being used by an authorised entity. This approach implies that the user has complete faith in the CSP to provide security over their data. The user has e ectively acquiesced to the CSP providing all guarantees over the protection of their data. This acquiescence also extends to how the data is used. Users are left with no real empowerment over the control over the protection of their data. Therein lies a problem, all the security is now service-side. Any consumer must have complete trust in the CSP to keep their data secure. Though the cryptographic operations maybe secure the CSP still is responsible entirely for the security of the data. Moreover, key management and storage is also completely service-side, the client must have faith in the CSP concerning key management and storage. This leads one to acknowledge that there is a trade-o between trusting, and thus letting, the CSP protect and use data, and its security.

Note. Grendel's operation for securing and sharing data is the same as that of CoClo. Unlike CoClo, however, all the operations are performed service-side, thus relieving the client of the burden concerning the security of their data.

6.3.3 Some Trust

The third and nal mode of trust implies that the user has some trust in the operation of the CSP but is allowed to have reservations about said operation. Such trust can originate from the service level agreements provided by the CSP (see Section 3.9.1) in which a certain standard of protection has been promised. The reservations, will originate from the unknown risk pro le of the service|see Section 3.8. With this mode of trust, one needs techniques that allow for the user to trust the CSP to some degree over the protection of their data and remain empowered. Two such examples that have already been mentioned in this chapter were NOYB and Privacy Manager. They allowed the user to use the service and more importantly not lose functionality provided by said service. However, with these examples, particularly that of Privacy Manager, it was noted that for the solution to work the CSP needs to have su cient knowledge of the techniques used by the user to protect their data so that the CSP can work with the data and respect the users choice.

6.4 Security Responsibilities

When protecting data that is to be placed in the Cloud, one also has to think about who is responsible for the protection of data. That is for which security requirements are users and the CSP liable. Throughout the discussion presented in this chapter a recurrent theme is that of both the user and the CSP playing a role. For instance, Section 6.2 mentioned that the Kafkaesque approach implied both the user and the CSP needing to work together for privacy to hold. This was reiterated within Section 6.3 with the some trust view. From this, it can be argued that both the CSP and user need to take active responsibility over this protection. Such as view was stated within Chapter 5 and a division of responsibility was summarised in Section 5.7. The service user is responsible for the con dentiality and integrity of data prior to its insertion into the cloud, and for specifying access to the data post insertion. The CSP is responsible for the security guarantees that the users themselves cannot make such as: data availability, accessibility (as speci ed by the user), and the consistency and con dentiality of the data once it has been placed within the cloud.

6.5 Towards a Solution

When designing a solution that allows those that place their data in the cloud, one needs to take into account the following salient points.

User empowerment does not alone equal Data Obfuscation.

When ensuring the con dentiality over one's data the aim should be for the data creator to gain control over how the data is being used rather than solely preventing its initial collection. This will include the data creator being in a position to dictate to whom access to their data should be granted.

The user need not be responsible for ensuring all security guarantees.

By design the service user has entrusted their data to some service. It is obvious that some form of co-operation must occur between the service user and the CSP over the protection of this data. A service user cannot be responsible for all aspects governing the con dentiality of their data. A good solution must recognise this and allow the service user to have trust in the CSP's ability to o er data protection.

Some CSPs may be evil but some are more evil than others.

The maliciousness of a CSP will di er from CSP to CSP. The degree of trust a orded to each individual CSP will as a result also di er. Service users who can recognise this di erence should have the option of allowing trusted CSPs access to their data, and also be able to evoke such access.

Part II

Predicate Based Encryption

Chapter 7

Introduction to Predicate Based Encryption

A high-level overview of Predicate Based Encryption schemes is given looking at their: definition, characteristics, access control, and cryptographic key composition.

7.1 Overview

Predicate Based Encryption (PBE), represents a family of asymmetric encryption schemes that allows for selective ne-grained access control as part of the underlying cryptographic operation [KSW08]. The origins of PBE are in Identity Based Encryption (IBE) [Sha85]. In IBE schemes an entity's encryption key is derived from a simple string that represents the entity's own public identity e.g. an email address. For example, given an entity Albert his corresponding encryption key will be Enc(Albert) == albert@foobar.com. During encryption, the resulting cipher-text will e ectively be labelled with the string representing the encryption key, the entity's public identity. An entity's decryption key will be derived from the same string used for the encryption key e.g. Albert's decryption key will be derived from his e-mail address. On recipt of a cipher-text message the recipient will be able to decrypt the cipher-text if and only if the two identities, contained within the decryption key and cipher-text, are `equal'. PBE schemes o er a richer scheme in which an entity's `identity' can be constructed from a set of attributes and decryption is associated with access policies that o ers a more expressive means with which to describe the relation between the attributes.

Generally speaking, within PBE schemes entities and cipher-texts are each associated with a set of attributes. These attributes are used to describe some aspect of the entity, the data that is being encrypted, and the environment. An entity will be able to decrypt a cipher-text only if there is a match between the attributes of the cipher-text and the decrypting entity. Matching is achieved through predicates (access policies) that denote: a) the set(s) of authorised attributes that an entity must possess in order to decrypt and access the plain-text; and b) the relationship between the attributes. Taking a dating website as an example. To indicate that only females over $5^0 9^{00}$ with an ability to speak Dutch or British English will be able to access encrypted data a user can use the following policy:

gender:female ^ (speaks:dutch _ speaks:british-english) ^ height >= $5^0 9^{00}$

Various constructions of PBE schemes have been proposed that use general predicates (represented as Boolean formula) or specific predicates such as equality, hidden vector or inner product. The choice of predicate will have a direct a ect upon the scheme, its characteristics and the composition of the access policies. Moreover, the placement of the predicate (either with the cipher-text or the decryption key) has a great a ect upon the workings of a PBE scheme. The a ect of predicate placement and other characteristics are discussed in Section 7.3. This chapter provides a high-level overview of PBE schemes. Details concerning the construction of PBE schemes are to be found in Chapter 8. Their use as part of a cryptographic system is discussed within in Chapter 9.

7.2 Definition

Common to all PBE schemes are four operations allowing for encryption, decryption and key generation. The precise value for encryption and decryption keys is dependent upon both the construction of the scheme and placement of predicates. This is de ned later in Section 7.3.2.

A general PBE scheme is defined as follows:

Definition 1 (General Predicate Based Encryption (PBE) scheme). A general PBE scheme consists of the four operations:

Setup: initialises the crypto-scheme and generates a master secret key MSK, used to generate decryption keys, and a set of public parameters MPK.

$$(MSK; MPK) := Setup() \qquad (7.1)$$

KeyGen: generates a decryption key Dec(entity) based upon the master secret key and some entity supplied input.

$$Dec(entity) := KeyGen(MSK; input) \qquad (7.2)$$

Encrypt: encrypts a plain-text message M using the public parameters and supplied encryption key for an entity.

$$CT := Encrypt(M; MPK; Enc(entity)) \qquad (7.3)$$

Decrypt: decrypts a cipher-text if and only if the attributes held by the entity can satisfy the access policy.

$$M := Decrypt(CT; MPK; Dec(entity)) \qquad (7.4)$$

Note. Several constructions also provide a fth operation: Delegate in which an entity will delegate decryption to a secondary entity. This operation is not common to many a PBE con-struction and for brevity it shall be omitted. The more interested reader should consult Bethen-court, Sahai and Waters [BSW07] for an example of a scheme that supports delegation.

7.3 Characteristics

PBE schemes will each possess di erent characteristics. Theses characteristics are based upon:

a) The type of predicates used Section 7.3.1; b) the placement of the predicate|Section 7.3.2; and c) visibility of attributes and predicates|Section 7.3.3. These characteristics have been derived and collated from several existing sources governing PBE schemes. Primarily Katz, Sahai and Waters [KSW08], Goyal, Sahai et al. [GS+06], Waters [Wat10] and Bethencourt, Sahai and Waters [BSW07] were consulted. PBE schemes can also be categorised further according to di erences in how the scheme was constructed. The di erent ways in which PBE schemes can be realised is discussed in Chapter 8.

7.3.1 Types of Predicates

When looking at the di erent PBE schemes, three groupings (or family) of schemes emerge based upon the predicates being used, the scheme's aim and the schemes construction.

Identity Based Identity Based Encryption (IBE) schemes are used to encrypt messages using a single attribute that refers to the users identity i.e. email address or passport number Shamir [Sha85]; Boneh and Franklin [BF01]. In these schemes the access policy is also comprised of a single attribute. To add extensibility to IBE schemes these `single' attributes were extended using string concatenation to encode more information.

Attribute Based Attribute Based Encryption (ABE) schemes [GS+06] further generalises upon IBE schemes and uses general predicates styled as boolean formula covering opera-tions such as conjunction, disjunction and thresholds. The attributes themselves need not necessarily refer to an entity's identity, or even to an entity, and can refer to non-identity related aspects such as TCP/IP port numbers and addresses.

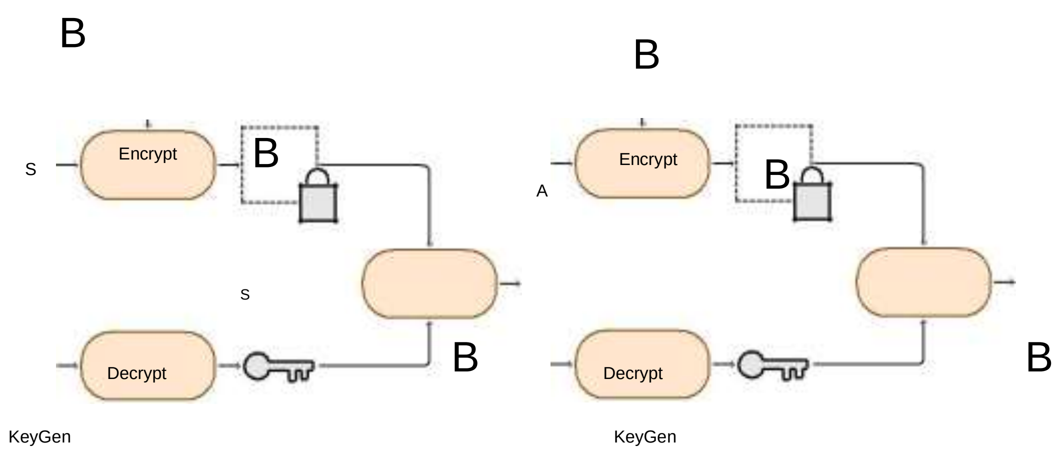
Specific Predicate Based The nal family of schemes are those that use speci c predicates during their construction. Although these schemes can be referred to by the predicate being used, the general term Predicate Based Encryption can also be used. Speci c predicates that have been used include that of inner product [KSW08] and hidden vector [BW07] predicates.

7.3.2 Predicate Placement

The placement of the predicate will have an effect upon the values that the cryptographic keys can take and ultimately the parameters used by the operations described at the beginning of this section. Ciphertext-Policy schemes are schemes in which the access policy is associated with the cipher-text. Key-Policy schemes are those in which the access policy is associated with the decryption key. The di erence between these two types of scheme, when used as part of a PBE crypto-system, is discussed further in Chapter 9.

Key-Policy In Key-Policy (KP) schemes an entity's decryption key is derived from an access policy A, and encryption keys are sets of attributes S. During encryption a set of attributes S is used to encrypt a message, and S is also encoded alongside the resulting cipher-text for use during decryption. A decryption key will be able to decrypt a cipher-text if the access policy used to generate the decryption key can be satis ed by the attributes encoded alongside the cipher-text. Figure 7.1a illustrates the overall operation of a Key-Policy scheme.

Remark. An alternate way to think of KP schemes is that the decryption key tells the de-crypting entity what `types' of cipher-text they can decrypt. The encryption key, or encryption process, will assign `types' to the cipher-text.



(a) Key-Policy Scheme (b) Ciphertext-Policy Scheme

Figure 7.1: Outline of Ciphertext-Policy and Key-Policy Schemes' Operation.

Ciphertext-Policy In Ciphertext-Policy (CP) schemes encryption keys are access policies and an entity's decryption key is derived from a set of attributes S. During encryption an access policy A is used to encrypt messages, and A is also encoded alongside the resulting cipher-text for use during decryption. Decryption of a cipher-text will occur only if the values contained within the decryption key i.e. within S, can satisfy the access policy embedded alongside the cipher-text. Figure 7.1b illustrates the overall operation of a Ciphertext-Policy scheme.

Remark. An alternate way to think of CP schemes is that the encryption key explicitly states the conditions under which decryption can be performed. While the decryption key is the condi-tion itself.

7.3.3 Privacy

The visibility of the access policies and attributes will di er between schemes. By design all PBE schemes are Payload Hiding (PH). This ensures that the payload cannot be accessed by a mali-cious entity. A PBE scheme is Attribute Hiding (AH) if the scheme also hides knowledge of the encryption key i.e. attributes/access policies, used to encrypt the cipher-text. During decryption when an entity attempts to access the plain-text they will only learn if the decryption procedure is successful or not and will not learn anything else concerning the cipher-text and its attrib-utes/access policy. While all schemes by default are payload hiding, attribute hiding schemes are dependent upon the underlying predicates used to realise the scheme. Such functionality is achieved when using inner product [KSW08] and hidden vector [BW07] predicates.

7.4 Access Control

Through the use of attributes and predicates, a richer form of access control has been built into the cryptography when compared to more traditional asymmetric encryption schemes such as RSA and El Gamal [RSA78; ElG85]. In traditional asymmetric schemes cipher-texts can only be decrypted using a single key that is paired with a single encryption key. One can summarise the relationship between decryption keys and cipher-texts as being one-to-one: one cipher-text can be decrypted only by one decryption key. However, with PBE schemes this relationship is more exible. Decryption in a PBE scheme will occur if the predicate can be satis ed by a given set of attributes. This relationship can be described as being one-to-many: one cipher-text can be decrypted by many decryption keys.

By specifying access control in terms of attributes and predicates, the access control o ered by PBE is analogous to Attribute Based Access Control (ABAC) and involves the assignment of descriptive attributes to entities, resources and the environment [YT05]. Access to a resource is decided upon by access policies that described the sets of attributes required, often as a boolean formula, for access to occur. ABAC is considered to be more exible and scalable when compared to other existing access control techniques such as Role Based Access Control (RBAC) [SC+96] and Lattice Based Access Control (LBAC) [SS94]. Moreover, ABAC is able to combine the functionality of both RBAC and LBAC into a single access control model. The authorisation architecture for ABAC consists of the following four actors:

An Attribute Authority (AA) is responsible for the creation and management surrounding attributes used to describe the resources, entities and the environment respectively.

The Policy Authority (PA) is responsible for the creation and management of the access policies that govern access to resources.

The Policy Decision Point (PDP), is where the access policies governing access to a resource are evaluated against the attributes attributed to the requesting entity, resource and the environment.

A Policy Enforcement Point (PEP) is a xed point where the right of the entity requesting access to a resource is challenged. This di ers from the PDP in that here the challenge is issued, and at the PDP the challenge is performed.

PBE does not replicate the functionality seen in ABAC completely. Unlike ABAC the environ-ment, entities and resources are not assigned to attributes directly, they are attached to crypto-graphic keys. The use of which, will have an a ect upon the access control. Furthermore, the distribution of these four actors to entities will di er based upon the placement of access policies i.e. the type of PBE scheme. Moreover, there is no central point at which policy enforcement and decision will occur. The remainder of this discussion is deferred until Chapter 9.

7.5 Attributes

The set(s) of attributes used to label both entities and cipher-texts, and construct access policies, originates from a universe of attributes U. The size of the attribute universe is dependent upon the construction of PBE being used. The universe of attributes is either: a) Small|if the number of attributes is xed and the attribute values speci ed as part of the system instantiation; or b) Large|if the number of attributes is unlimited in size and values can be speci ed later.

While the precise semantics governing these attributes are dependent upon the setting of use much can nonetheless be discerned. Attributes are either textual or numerical in value, and can also be represented as key-value pairs in which the key can possess multiple di erent values. Within PBE attributes are used either as an identi er or as a descriptor of some (in)tangible aspect of either the entity or data that is to be encrypted. Attributes can be divided roughly into three categories corresponding to whom and what they reference:

Entity Attributes Attributes that de ne the identity and other related characteristics as-sociated with an entity. Such attributes may include an entity's name, pay grade, security clearance level and organisation a liation.

7.6. Access Policies

Resource Attributes Attributes that describe characteristics associated with the resource that is to be encrypted. Taking TCP/IP connections as an example; the source and destin-ation IP address and port numbers could be used as attributes within the cryptographic system.

Environment Attributes There will often be attributes that relate neither to a resource nor an entity e.g. the current date and current locale. Such attributes refer to the environment in which both entities and resources reside.

Remark. It will be the case that entities or cipher-texts will have several attributes in common, for example: o ce number, address, IP-address et cetera. It is this commonality that allows for the selective ne-grained access control to exist.

7.5.1 Attribute Uniqueness

Attributes must be uniquely named and their meaning clearly distinguishable from other at-tributes. Moreover, related types of attributes such as dates should be presented in a common format to ensure their canonicity. This could otherwise give rise to the malicious use of attrib-utes. For example, take the use of attributes to capture the source and destination IP addresses within a TCP connection. Naively, one would be tempted to simply allow for IP addresses to be captured and used in their standard form i.e. 127.0.0.1. However, this form conveys no inform-ation concerning whether or not this is the source or destination address within the TCP header. A more suitable and canonical form would be DEST.ADDRESS=127.0.0.1. This conveys more information concerning the meaning of the attribute and leaves rise to less confusion concerning as to if this were the destination or source address.

7.5.2 Compound Attributes

Attributes with PBE schemes are represented within a single set. When attempting to satisfy an access policy, all possible combinations of the attributes issued can be used. While this is useful for natural singleton attributes, compound attributes representing natural combinations of single attributes are more di cult to specify and use. Such attributes arise when re-enforcing the notion of attribute uniqueness. A typical example of where compound attributes come into existence are when attributes are used to refer to entities. Within an organisational setting individuals are often given some short term responsibility. For example, within a university PhD students are often tasked with teaching assistant duties. These duties will change yearly, if not each semester. This would result in attributes of the form: TA:<course code>:<year> being used to identify a teaching assistant for a course, for a particular year. This causes problems as this precludes the use of the attribute in an access policy that grants access to all teaching assistants i.e. TA. PBE schemes are not designed to decompose compound attributes.

Naively, one may be tempted to solve the problem of compound attributes by stating that when using only singleton attributes the `construction' of compound attributes can be achieved within access policies using conjunctions. For example, TA:<course code>:<year> would be-come: TA ^ <course code> ^ <year>. However, this can give rise to the malicious use of attributes to satisfy poorly designed access policies if the attributes were not uniquely named.

Although there have been attempts at the creation of PBE schemes that alleviate the prob-lems associated with compound attributes (see Bobba, Khurana and Prabhakaran [BKP10]) the problem of compound attributes still remain an important issue when using PBE schemes.

7.6 Access Policies

Predicates are used to de ne the required sets of attributes needed, often in terms of boolean for-mula, by the decrypting entity for decryption to occur. Within PBE the term Access Policy can be used interchangeably with the term predicate, and within Key-Policy schemes with decryption key. This section presents an overview of access policies, their de nition and limitations.

7.6.1 Expressiveness of Access Policies

The expressiveness of an access policy is dependent upon both: a) the predicates used by the scheme; and b) other mathematical constructs used within the scheme's construction. That is, the permissible operations available when constructing an access policy are not universal and will di er. The reasons for this di erence, aside from di erences in underlying predicates, is discussed further within Section 8.3. This di erence, in expressiveness, can be summarised in terms of the operations available and the number of arguments that these operations allow. The most common form of predicate seen supported, is that of a two-argument operator supporting conjunction and disjunction operations. Other forms seen include the use of multi-argument boolean operations supporting threshold, numerical comparisons and attribute negation. Section 7.7 presents several PBE schemes and notes the expressiveness of their access policies.

Though there are constraints over the exact operations permissible, these constraints reflect the internal use and representation of the predicates within the crypto-scheme itself. The in-ternal representation represents a normalised form that the predicates must take. Implying that more complex policies can be constructed as long as they can be transformed into the required normalised form.

7.6.2 Definition

The disparity between the expressiveness of access policies in PBE schemes increases the scope of the discussion greatly. For simplicity the scope of the discussion concerning access policies will be restricted to the use of general predicates. General predicates can be represented as a boolean function F. The de nition for an access policy is as follows:

Definition 2 (Access Policy). Adapted from [BL88, Section 2]: Given a universe of attributes U, an access policy A is a boolean function F, indexed by a set of attributes P = fP_1; : : : ; P_ng; P_i 2 U. The function F supports the following boolean operations:

Disjunction: _|1 out of n attributes.

Conjunction: ^|n out of n attributes.

Threshold: T_n(a_1; : : : ; a_m)|n out of the following m attributes must be present.

The operations de ne the set A of a P for which F is true whenever the variables indexed by a set in A are also set to true. That is, the function F returns true if and only if an entity is able to satisfy at least one a 2 A.

Remark. Conjunction and disjunction operations represent speci c instances of a threshold function when n = m and n = 1 respectively.

Note. With respect to multi-valued attributes standard set notation can be introduced as a short hand in certain cases. When the same attribute is referred to in a series of disjunction operations the following can be speci ed att 2 fa_1; : : : ; a_mg. Where `att' refers to the key and each a_i is the di erent values. Similarly when assigning an attribute set that allows for multiple values to be assigned the following can be speci ed: att = fa_1; : : : ; a_mg.

7.6.3 Additional Operations

Although De nition 2 provides a healthy set of operations that can be performed upon attributes using general predicates, other operations can also be used.

Numerical Comparison

Numerical operations allow for the existence of key value pairs to be specified, and used within access policies to construct range queries. In supporting PBE schemes the standard set of numerical comparison operations i.e. <; >; ; ; =, can be represented.

Unfortunately numerical comparisons are more troublesome to implement than regular at-tributes. Conceptually it seems trivial to use numerical comparisons in access policies. Their result is boolean; the number either satis es the comparison or it does not. Numerical attributes can be assigned to individual entities however in these access policies one needs to explicitly state ranges of numerical attributes i.e. all the attributes that are required. For example: an entity can be assigned a pay grade of four in a system with twelve levels, yet an access policy can state

7.7. Note on Existing Schemes

Paygrade < 4 which, when using the previously mentioned access policy de nition, would imply that the policy would be expanded as follows:

Paygrade = 3 _ Paygrade = 2 _ Paygrade = 1

In Bethencourt, Sahai and Waters [BSW07] the authors present a PBE scheme construction that supports numerical comparisons and provides a novel means by which to implement them. For all numerical attributes the authors re-represent them in base-2 form rather than in base-10. In this form bit-masking can be used to indicate preferred values, and positions of each individual binary digit within a binary string. Thus, an n-bit integer would be represented as n attributes. For example using 4-bit binary strings, the attribute a = 5 would instead be represented as: a:1***, a:*0**, a:**0*, and a:***1.

When used in conjunction with disjunction and conjunction operations, the numerical com-parison operation can be represented as an access policy in its own right. That is the operation can be represented as an access policy comprised of various bit masking values that when com-bined produce a value that satis es the comparison. For example, the comparison a < 11 using 4-bit integers can be represented as follows:

a:0*** _ (a:*0** ^ (a:**0* _ a:***0))

The solution presented in Bethencourt, Sahai and Waters [BSW07] is a natural extension to the use of general predicates and ts in naturally with the existing de nition. Numerical comparisons will be included when using general predicates within this thesis as a result.

Attribute Negation

Attribute negation can be seen as an important operation to perform upon an attribute. It is particularly useful when trying to exclude a certain member of a group where their presence may result in a con ict of interest. For example organising a surprise party for Bob who is leaving the Digital-Security research group. In this situation an ideal access policy for organisational messages would be:

Digital-Security ^ :ID_{bob}

Where: indicates attribute negation. Here Bob would not be able to access these secret messages and spoil the surprise. However, the addition of a negation operation is not trivial. Ostrovsky, Sahai and Waters [OSW07] looked towards, among other aspects, at implementing attribute negation. As such attribute negation will not be a supported operation within this thesis. More details concerning how attribute negation can be achieved can be found in Ostrovsky, Sahai and Waters [OSW07].

7.6.4 Examples

This section ends with an example to illustrate further the use and declaration of an access policy. The example is as follows:

(Role:ceo _ (T_2(Paygrade = 3; Group:audit; Role:secretary) ^ O ceNumber = 1234))

With reference to De nition 2: The policy itself is the function F acting upon the set of attributes P = fceo; Paygrade = 3; audit; secretary; O ceNo = 1234g. For the access policy to be satis ed the entity must either be a ceo, or work in O ce 1234 and is either: a) a secretary or part of the audit team with a Paygrade equal to three; or b) a member of both the audit team and is a secretary.

7.7 Note on Existing Schemes

Numerous constructions of PBE are known to exist. Table 7.1 presents a categorisation of a select number of these constructions according to the characteristics as discussed in Section 7.1 and the expressiveness of their access policies. This is to give the reader some indication of the di erent constructions available and their di erences at a high-level. Although the scheme presented in Bethencourt, Sahai and Waters [BSW07] is the only one that appears to support numerical operations, the use of numerical attributes and comparisons on said attributes can easily be extended to the other schemes during their implementation.

The papers listed also contain several variants of the main construction, found within each paper, that differ in relation to their implementation and assumptions made. The more mathematical oriented reader is asked to consult Chapter 8 for more information regarding the implementation of PBE schemes.

Table 7.1: Categorisation of several known PBE constructions

Construction	Family	Privacy	Access Policy Expressiveness		
			Placement	Operators	Multi-Arg.
[GS+06]	ABE	PH	KP	_; ^; Tn(); =	yes
[PT+06]	ABE	PH	KP	_; ^; Tn(); =	yes
[BSW07]	ABE	PH	CP	_; ^; Tn(); <	yes
				; >; ; ; =	
[Wat10, Appendix C]	ABE	PH	CP	_; ^; =	no
[KSW08, Appendix C]	PBE	PH, AH	KP	_; ^; Tn(); =	yes

Other existing constructions not included in the categorisation include: Sahai and Waters [SW05]; Cocks [Coc01]; Yao, Fazio et al. [YF+04]; Gentry [Gen06]; Liang, Cao et al. [LC+09]; Ibraimi, Petkovic et al. [IP+09]; Shen, Shi and Waters [SSW09]; Lewko, Okamoto et al. [LO+10] and Bobba, Khurana and Prabhakaran [BKP10]. From which, the following presents some interesting findings:

Shen, Shi and Waters [SSW09] Introduces a symmetric variant of PBE that would allow for predicate privacy in a remote setting. For example, a user can upload their les to a remote server and query the server for les that satisfy a certain access policy using a specially constructed token. The server performing the query would not learn anything regarding the construction of the access policy.

Bobba, Khurana and Prabhakaran [BKP10] Extends upon the construction found in Beth-encourt, Sahai and Waters [BSW07] by increasing the exibility of representing user at-tributes in the keys.

Chapter 8

Constructing PBE Schemes

The construction of Predicate Based Encryption schemes using general predicates is discussed. A CP-ABE from Waters [Wat10] is used to illustrate a working PBE scheme. A discussion over the security of PBE schemes is also provided.

8.1 Overview

Chapter 7 provided a high-level overview over how PBE schemes work. Several existing schemes were discussed in Section 7.7. The aim of this chapter is to discuss how these PBE schemes can be realised.

As with many a encryption scheme, PBE schemes encrypt messages using some secret value. PBE schemes di er in their operation by using Linear Secret Sharing Scheme (LSSS) to distribute this secret value among a set of attributes according to some access policy. Only those authorised to decrypt the cipher-text will be able to reconstruct the secret value. While it is typical to see PBE schemes being implemented using pairing based cryptography, the techniques used to transform the predicate/access policy into a LSSS will vary. Moreover where this policy transformation occurs is dependent upon whether the scheme is a Key-Policy or Ciphertext-Policy scheme. To illustrate and discuss all the di erences between the various methods of construction for di erent predicates only increases the scope and complexity of this discussion. As such the discussion within this chapter will only focus on general predicate PBE schemes. It is hoped by focusing on these schemes that the reader can gain a `feel' for how other PBE schemes operate.

Constructions of PBE schemes can be characterised in terms of: a) how predicates are trans-formed into LSSSs; b) how the resulting LSSS integrates with the pairing based cryptography. c) the underlying pairing based cryptography. Section 8.2 o ers a short introduction to pair-ing based cryptography and how the encryption and decryption operations can be realised. Section 8.3 provides an alternate and more formal de nition for access policies, and how they relate to LSSS and cryptographic operations of PBE schemes. For many general predicate PBE schemes the means by which the LSSS and pairing based cryptography are integrated will di er substantially. Section 8.5 details how this can be achieved using the large universe CP-ABE scheme from Waters [Wat10]. Finally, Section 8.6 discusses several security aspects related to the construction of PBE schemes.

8.2 Pairing Based Cryptography

The construction given in Section 7.7 use pairing-based cryptography, speci cally bilinear pairings, as a means to realise the encryption and decryption operations.

Definition 3 (Bilinear Pairings). From Bethencourt, Sahai and Waters [BSW07]: Let G_1 and G_2 be two multiplicative cyclic groups of prime order p. Let g be a generator of G_0 and e be a bilinear map, e : G_0 G_0 ! G_1. The bilinear map e has the following properties:

1. Bilinearity: for all u; v 2 G_0 and a; b 2 Z_p, we have $e(u^a; v^b) = e(u; v)^{ab}$.

2. Non-degeneracy: e(g; g) 6= 1.

We say that G_0 is a bilinear group if the group operation in G_0 and the bilinear map e: G_0 G_0 ! G_1 are both e ciently computable. Notice that the map e is symmetric since $e(g^a; g^b) = e(u; v)^{ab} = e(g^b; g^a)$.

Bilinear pairings are used to perform the `cryptographic step' used to encrypt a message [BF01]. The encryption and decryption operations can be defined, generally, as follows: When encrypting a message M the encrypting entity will perform some `encrypting' operation, say, on m using a well-known value v, i.e.

$$CT = M\ v:$$

This well known value is part of the recipients public key. During the `decrypting' step the recipient will apply their `decrypting' value, say $v^{\ 1}$, that reverses the original operation. The decrypting value is essentially the inverse of the encrypting value i.e.

$$M = CT\ v^{\ 1} = M\ v\ v^{\ 1}:$$

For example, take the well known RSA and El Gamal encryption schemes [RSA78; ElG85]. For El Gamal the transformation of a message m to a cipher-text and back is:

$$CT = M\ \ s: M = CT\ \ s^{\ 1} = M\ \ s\ s^{\ 1}:$$

Where s is the shared secret shared between sender and recipient. For RSA the transformation is as follows:

$$CT = M^e \text{ (mod n)}: M = CT^{\ d}\ (M^e)^d \text{ (mod n)}:$$

Where e is the encryption exponent and d is the decryption exponent that `cancel' each other out because of the modulus operation. Within pairing-based cryptography the operation performed upon a plain-text message M, represented as a group element, is:

$$CT = M\ e(g; g)^s: \quad M = \frac{CT}{e(g; g)^s} = \frac{M\ e(g; g)^s}{e(g; g)^s}$$

Where e(g; g) is a bilinear mapping and s is the secret exponent. More interested readers are asked to consult Dutta, Barua and Sarkar [DBS04]; Menezes [Men09] and Boneh [Bon07] for more information on pairing-based cryptography. The ne-grained access control associated with PBE comes from the use of LSSS schemes to distribute the secret exponent s among group elements representing attributes. The security of pairing based cryptography is discussed in Section 8.6.4.

8.3 General Predicates and LSSS

As it was established within Section 7.1, predicates are used to specify for which sets of attributes decryption is permissible. This use of a general predicate is analogous to that of an Access Structure, a known mathematical construct. Formally, access structures are defined as follows:

Definition 4 (Access Structure). Adapted from Beimel [Bei96, Section 3.2 De nition 3.5]: Let P = fP1; P2; P3; : : : ; Png be a set of parties, an access structure A is the set of all authorised subsets of P. Access structures are often monotone, that is given sets Y and Z, if Y 2 A and Y Z then Z 2 A for all Y; Z P

On their own access structures cannot be used directly within the cryptographic operations.

Access structures must first be transformed into a Secret Sharing Scheme (SSS) scheme.

Definition 5 (Secret Sharing Scheme (SSS)). A SSS is a scheme used to distribute a secret value s K, where K is a nite eld, among a prede ned set of participants P according to some access structure A. The result of this distribution is that given an set of participants B P, if:

B 2 A then s can be computed.

B 2= A then s cannot be computed.

For each scheme there will exists a share (piece) generation and secret reconstruction function.

While, SSSs can be used to distribute the secret, they imply that a xed number of participants are required before the secret can be reconstructed. Threshold secret sharing di ers from general secret sharing in that a minimum number of participants i.e. a quorum, is required before the secret can be reconstructed.

Definition 6 (Threshold Secret Sharing). Given a set S of possible secret values, a (k; n)-threshold scheme on S is a (randomised) method of dividing each s 2 S into an array of shares si s such that:

1. Given any set of k or more of the si, the secret value s can be reconstructed easily.
2. Given any set of fewer than k of the si, the secret value s is completely undetermined in an information theoretic sense.

Furthermore, a Threshold-SSS can also be linear, resulting in a LSSS.

Definition 7 (Linear Secret Sharing Scheme (LSSS)). Adapted from Beimel [Bei96]: A secret sharing scheme is linear (a linear generation of pieces) if:

1. The piece of each party is a vector over K.
2. During generation of the pieces, the dealer chooses independent random variables, denoted r1; : : : ; rl each one distributed uniformly over K. Each co-ordinate of the piece of every party is a linear combination of r1; : : : ; rl and the secret s.

Generally speaking: With pairing based cryptography each attribute is represented as a group element. By virtue of the bilinearity property it allows for two independent sets of operations to be performed upon a set of group elements representing each Pi 2 P. These operations hide the secret exponent among the group elements such that when the result of these operations are combined if the conditions are right the secret exponent to be recovered. These conditions are dictated by the LSSS.

The precise use of LSSSs to hide and recover the secret exponent is dependent not only upon the placement of the predicates within the PBE scheme but also upon the exact predicate used. For the remainder of this section, a general overview of how LSSSs are used as part of both CP and KP schemes. Section 8.5 provides a concrete example of how one can use LSSS precisely as part of a CP-ABE construction.

Note. Several techniques that allow access structures to be turned directly into LSSSs are de-tailed within Section 8.4.

8.3.1 Ciphertext-Policy

Recall from Section 7.3.2 that with CP schemes a message will be encrypted under an access policy A and can be decrypted from a key that is derived from a set of attributes S. The use of LSSS within CP schemes can be seen as LSSS in its standard form.

Encrypt The LSSS is used to generate piece vectors, from the secret exponent, for each group element that represents an attribute Pᵢ 2 P as defined in A. These vectors along with a description of the LSSS are stored along side the encrypted message.

Key Generation With the generation of decryption keys each user is assigned a set of attrib-utes represented as a set of group elements modified by some random secret value.

Decrypt During decryption the LSSS, described in the cipher-text, will only reconstruct the secret exponent if an authorised set of attributes can be found within the elements of the decryption key.

8.3.2 Key-Policy

With Key-Policy schemes, a message will be encrypted under a set of attributes S and can be decrypted using a key that is derived from an access policy A. The use of LSSS di ers from its standard use.

Encrypt After the message has been encrypted the secret exponent is hidden among a set of group elements that have been derived from each Sᵢ 2 S, the encryption key. These elements along with a description of S are stored along side the encrypted message.

Key Generation When generating the decryption key, the LSSS is used to distribute a sec-ondary secret value among each attribute Pᵢ 2 P from A. The resulting piece vectors and description of A are returned as the decryption key.

Decrypt With decryption the LSSS, taken from the decryption key, will only reconstruct the secret exponent if an authorised set of elements exists within the elements stored alongside the cipher-text.

8.4 Transforming an Access Policy into a LSSS

There are two families of technique that can be used to transform general predicates into a LSSS.

Access Tree: Constructs a LSSS, based upon Shamir's construction [Sha79], directly from a tree representation of the access structure.

Monotone Span Program (MSP): Constructs a LSSS derived from a MSP, often referred to as the LSSS-Matrix technique.

The choice of technique when constructing a PBE scheme is important and will a ect not only the supported operations that can be used when de ning access policies (cf. Section 7.7) but also the e ciency of the implementation. It is known that implementations of LSSSs based upon Shamir's construction [Sha79], are not the most elegant way in which such schemes can be built. Beimel [Bei96] demonstrated the equivalence between a LSSS and a MSP. The LSSS-Matrix family of techniques presents a more e cient, and also interesting, construct. However, a downside with this technique is that the number of permissible arguments that each operator can have is dependent upon the precise algorithm used.

This section discusses three algorithms that can be used to transform an access structure into a LSSS by way of a MSP. These three algorithms are: a) Lewko-Waters; b) Beimel; and c)Liu-Cao. Before these algorithms are discussed some background information on MSPs is given, before the de nition for a LSSS-matrix.

8.4.1 Preliminaries

Span Programs were rst introduced in Karchmer and Widgerson [KW93], the following descrip-tion of a span program is taken directly from Karchmer and Widgerson [KW93]:

A Span Program is a linear algebraic model that computes a function f on n variable, over any eld K. Given a xed vector space W over K, a non-zero vector w 2 W and let $X_i^"$ (with 1 n; " 2 f0; 1g) be 2n subspaces of W that correspond to the 2n literals of f. Any truth assignment to the variables makes exactly n literals `true'. We demand that the n associated subspaces span the xed vector w i f( ) = 1.

More formally the de nition for a Span Program is:

Definition 8 (Span Program). From Karchmer and Widgerson [KW93, Definition 2]: Fix a field K. A span^ M;) where M is a matrix over K and program over K is a labelled matrix M ("^: rows(M) ! fxi j i 2 [n]; " = 0; 1g. The size of M is the number of rows in M. of is when using span programs, for every input sequence 2 f 0; 1 gn, the sub-matrix M×M.

"

Definition 8 (Span Program). From Karchmer and Widgerson [KW93, Definition 2]: Fix a field K. A span defined by keeping the rows r that are (r) = x_i and $_i$ = ". The span of M is de ned by the subspace generated by the rows of M. It is denoted using span(M). Given the all one vector ~ ^ ~ 2 span (M). Span programs can compute a Boolean function F, if 1 in W , M accepts i1 the program accepts exactly the inputs where F ( ) = 1. Now the de nition for a MSP can be given:

Definition 9 (Monotone Span Program). Adapted from Karchmer and Widgerson [KW93,^ M Section 3]: A span program is called monotone, if the image of is only the positive); (M literals fx_1; : : : ; x_ng. Such span programs can also compute monotone functions.

By adapting the de nitions for both LSSSs and MSPs a de nition for an LSSS-Matrix can be constructed:

Definition 10 (Linear Secret Sharing Scheme (LSSS)-Matrix). Adapted from Waters [Wat10] A secret-sharing scheme over a set of parties P is called linear (over Z_p) if:

1. The shares for each party form a vector over Z_p.
2. There exists a matrix M called the share-generating matrix for. The matrix M has l rows and n columns. For all i = 1; : : : ; l, the i^{th} row of M we let the function define the party labelling row i as (i). We consider the

column vector v = (s; r2; : : : ; rn), where s 2 Zp is the secret to be shared, and r2; : : : ; rn 2 Zp are randomly chosen, then Mv is the vector of l shares of the secret s according to . The share (Mv)i belongs to party (i).

Remark. This is the de nition for an LSSS-Matrix that has been used in several PBE construc-tions [GS+06; BSW07; Wat10]. When using such access structures, it is essential to remember that it is a MSP.

Note. For notation purposes the following shall be used to represent a LSSS access structure when describing PBE constructions: (M;).

8.4.2 Lewko-Waters Algorithm

First hinted in Beimel [Bei96, Chapter 4 Section 4.2 Example 4.5] and ultimately described in Lewko and Waters [LW10, Appendix G] this algorithm, based upon symmetric branching programs, requires that the access policy consists of two-argument conjunction and disjunction nodes and that leaf nodes represent the attributes of the policy. General threshold gates and multi-argument operators are not supported. An example of a supported formula is: A ^ (D _ B ^ C)).

Remark. Although, the algorithm was rst hinted at in Beimel [Bei96] for the purposes of this document it shall be called the Lewko-Waters Algorithm.

The Lewko-Waters Algorithm directly constructs the LSSS-Matrix from the access policy. This algorithm iterates over the nodes within the tree and labels each node with a vector whose construction is dependent on the type of node and its parent vector. Once each node has been labelled the vectors of the leaf node are used to construct the LSSS-Matrix.

Note. In Liu and Cao [LC10, Section 1 Related Work] it is hinted that threshold gates con-tained within the access structure are rst expanded to DNF form before use within the algorithm. Threshold gates are not supported directly. However, this has not been made explicit in the papers concerning this construction.

8.4.3 Beimel Algorithm

This method as hinted in Beimel [Bei96, Chapter 4 Section 4.4] requires that the monotone boolean formula is rst transformed into a LSSS and then into a monotone span program. To do so one can use the Benaloh-Leichter technique [BL88] to transform the boolean formula into a LSSS and then use the transformation method described in Beimel [Bei96, Chapter 4 Section 4.4 Claim 4.9] to transform the LSSS piece vectors into an MSP. Like the Lewko-Waters construction the boolean formula, taken as input, are represented as an access tree (or boolean circuit), however, unlike the previous technique, the interior nodes can be multi-input AND, OR and THRESHOLD gates. An example of a supported formula is: (A ^C ^D) _(B ^C) _T2(D; E; F).

Using this algorithm the size of the LSSS-Matrix is not optimal, the number of rows in the matrix is equal to the number of leaf nodes. Also Liu and Cao [LC10] mention that with threshold operators the number of rows within the LSSS-Matrix will be unnecessarily large. It is clear that LSSS-Matrices constructed using this technique are not optimal.

8.4.4 Liu-Cao Algorithm

In Liu and Cao [LC10], a recently released ePrint, the authors discuss and present an algorithm to construct optimal nay e cient LSSS-Matrices for use in CP-ABE, using an MSP based con-struction. They claim to be able to convert access trees in the same form as used with the aforementioned Lewko-Waters algorithm. For example given the following access structure:

= (A ^ B) _ (B ^ C) _ (A ^ C)

= (B ^ (A _ C)) _ (A ^ C)

= T2(A; B; C)

And using the middle transformation as input, the Lewko-Waters algorithm generates the fol-lowing LSSS-Matrix:

00 1 0 1 0A1

1 1 0 B

B C B C

matrix := B0 1 0 C := BCC

B C B C

B C BC

1 0 1 A

@ 0 0 1A @C A

Using the algorithm from Liu and Cao [LC10] they generate the following:

1 1 A

matrix := 0 1 21 := 0B 1

@ 1 3A @C A

It is clear that this matrix is smaller in size: Liu and Cao contend that when the boolean formula does not contain threshold gates, its size will be the same as that produced using the Lewko-Waters algorithm. If threshold gates are used then the Liu-Cao algorithm produces a smaller matrix. Given the recent publication status of the paper and that it was published as an ePrint leaves one with reservations concerning the papers claims i.e. it has yet to be peer-reviewed. However, once the authors' claims have been substantiated then the Liu-Cao algorithm certainly does look promising.

8.5 Waters Construction

Section 8.2 detailed how pairing based cryptography performs the cryptographic step. Section 8.3 discussed how pairing based cryptography can be combined with a LSSS to ensure that only entities in possession of an authorised set of attributes can recover the secret exponent used to encrypt the message. Converting a predicate into a LSSS was discussed in Section 8.4. This section illustrates how all this can be combined to provide a coherent PBE scheme.

The \large-universe" construction (see Waters [Wat10, Appendix C]) has been chosen, it does not require any restriction on the size of U due to the use of a random oracle. This construction uses the Decisional Parallel Bilinear Di e-Hellman Exponent (d-Parallel BDHE) as the main security assumption with respect to the pairing-based cryptography. The authors provide a Game based security proof for their construction, for brevity this scheme is given sans security proof. The full proof can be found in Waters [Wat10, Section 2.4.1, Section 3.1 and Appendix C.1]. The security of this scheme and other PBE is discussed more in Section 8.6.5.

8.5.1 Definition

Definition 11 (Waters Ciphertext-Policy (CP) Attribute Based Encryption (ABE) Scheme). Adapted from Katz, Sahai and Waters [KSW08, Section 3.1]: The Waters CP ABE scheme consists of four fundamental algorithms:

Setup: takes as input a security parameter and outputs a master public key MPK (public parameters) and a master secret key MSK:

$$(MSK; MPK) := Setup(1^n) \tag{8.1}$$

Encrypt: takes as input a master public key, an access structure A and a message M in some associated message space. It returns a cipher-text CT :

$$CT := Encrypt(MPK; A; M) \tag{8.2}$$

KeyGen: takes as input the master secret key and a set of attributes S that describe the key. It returns the decryption key Dec(S):

$$Dec(S) := KeyGen(MSK; S) \tag{8.3}$$

Decrypt: takes as input the public parameters MPK, a decryption key Dec(S) and a cipher-text C. It outputs either a message M or the distinguished symbol ? if the set of attributes S do not satisfy the access structure A.

$$Decrypt(MPK; Dec(S); C) = \begin{cases} M & \text{If correct private key} \\ ? & \text{If incorrect private key} \end{cases} \tag{8.4}$$

Remark. The decrypt operation can also be de ned without the public parameters and that their existence for decryption is implicit. In Bethencourt, Sahai and Waters [BSW07] the de-scribe their system with the public parameters and when describing the implementation the public parameters are absent from the functions parameters.

The remainder of this section will detail the implementations for each of the four algorithms speci ed in the de nition.

8.5.2 Setup

The Setup algorithm (listed in Algorithm 1) takes as input a security parameter and outputs a master public key MPK (public parameters) and a master secret key MSK.

Algorithm 1 Waters Setup Algorithm

```
Input: $1^n$ : Implicit Security Parameter
Output: MSK : Master Secret Key
```

```
Output: MPK : Master Public Key

1. Generate a prime p
2. Select G of prime order p with generator g
3. De ne a Hash function H : f0; 1g ! G
4. Choose random  ; a 2 $Z_p$
5. let MPK := (g; e(g; g) ; $g^a$; H)
6. let MSK := g
7. return (MPK; MSK)
```

8.5.3 KeyGen

The KeyGen (listed in Algorithm 2) takes as input the master secret key and a set of attributes S that describe the key. It returns the decryption key Dec(S).

```
Algorithm 2 Waters Key Generation Algorithm

   Input: MSK: Master Secret Key
   Input: S : Set of attributes
   Output: Dec(S) : Decryption Key associated with S
1. Choose random t 2 $Z_p$
2. let K := g $g^{at}$
3. let L := $g^t$
4. let K := (8x 2 S let $K_x$ := $H(x)^t$)
5. return (K; L; K)
```

8.5.4 Encrypt

The encryption algorithm works by rst using the access structure to distributed the secret s among the attributes and constructs a series of randomised variables to hold the result of this distribution. It then constructs the cipher-text and publishes that along with a description of the access structure. The algorithm is listed in Algorithm 3.

```
Algorithm 3 Waters Encryption Algorithm

Input: MPK : Master Public Key
Input: (M; ) : An LSSS Access Structure
Input: M : Plaintext
Output: CT : Ciphertext
```

```
1.  Choose random s 2 Zp
2.  // Distribute the secret exponent
3.  let M be a l  n matrix
4.  Construct random vector ~v = (s; y2; : : : ; yn) 2 Z^n_p
5.  for i := 1 to l // For each row in matrix do
6.          let  i := ~v  Mi
7.          Choose random r1; : : : ; rl 2 Zp
8.  end for
9.  // Construct the Cipher-text
10. let C := M e(g; g) ^s
11. let C^0 = g^s
12. for i := 1 to n do
13.         let Ci := g i H( (i)) ^ri
14.         let Di := g^ri
15. end for
16. return (C; C^0; (Ci; Di)) where 1   i   n and a description of (M; )
```

Remark. Recall: The function associates a row in M to an attribute. In this construction is an injective function and attributes are associated with at most one row in M.

8.5.5 Decrypt

Decryption of a cipher-text is relatively straight forward. First the algorithm checks to ascertain if the private key satis es the access structure. If satisfaction does not occur then the algorithm returns the distinguished symbol?. Otherwise, the algorithm will identify the attributes that satis ed the access structure as: I f1; 2; : : : ; lg where I = fi : (i) 2 Sg. The existence of I implies that there exists constants f! 2 Zpgi2I such that if f ig are valid shares of the secret

exponent s then i2I !i i = s. These constants can be obtained using the LSSS associated with the access structure. The algorithm then uses these constants to reconstruct the secret exponent s and will then perform a calculation using the value C from the cipher-text that when divided out will return M. The algorithm can be found in Algorithm 4.

8.6 Security Discussion

This section discuss several security issues relating to the construction of PBE schemes in gen-eral and also relating to the Waters scheme illustrated in Section 8.5. For readers who are unfamiliar with formal cryptographic security the literature can be quite daunting, confusing and downright incomprehensible at times. This is due to a plethora of di ering concepts and de nitions arising from the various literature. Interested readers should consult Stinson [Sti04] for an interesting polemic on cryptographic security. As such this section of the thesis when dealing with more advanced concepts such as provable security and computational security will, at times, be sparse and shall include general descriptions only and where appropriate links to relevant further material.

Algorithm 4 Waters Decryption Algorithm

Input: Dec(S) : Decryption Key

Input: CT : Cipher-text

Input: (M;) : Access structure associated with the Cipher-text, inclusion is implied.

Output: M : Plain-text

```
1.  // Check for satisfaction
2.  if Dec(S) does not satisfy (M; ) then
3.          return ?
4.  else
5.          let I = fi :  (i) 2 Sg where I  f1; 2; : : : ; lg
6.          let f! 2 Zpgi2I be a set of constants obtained from CT .
7.          // Perform Decryption
8.          let
            $:= \frac{e(C^0; K)}{(e(C_i; L)e(D_i; K_{(i)}))^{!_i}} = \frac{e(g; g)^s e(g; g)^{ast}}{e(g; g)^{ta_i !_i}} = e(g; g)^s$
            $i2I \qquad i2I$
9.          return $C = \frac{Me(g;g)^s}{e(g;g)^s} = M$
10. end if
```

8.6.1 Policy Expressiveness

The expressiveness allowed by an access policy is dependent on the transformation technique used to convert the predicate into a LSSS. The permissible node types and number of node inputs have been summarised within Table 8.1. From Table 8.1 it is clear that the Lewko-Waters technique di ers in that it only permits operators with two arguments and does not support threshold operations natively. The remaining techniques do not possess such limitations.

Technique	Interior Node Type			Multi-input Gates
	OR	AND	THRESHOLD	
Access Tree	yes	yes	yes	yes
LewkoWaters	yes	yes	not directly	no
Beimal	yes	yes	yes	yes
Liu-Cao	yes	yes	yes	yes

Table 8.1: Summary of Access Policies Realisation and Permissible Boolean Formula Composi-tion.

8.6.2 Entity Collusion

As the relationship between cipher-texts and entities, when decrypting, is one-to-many, one of the problems encountered when designing an PBE scheme is that of entity collusion. Entity collusion is an attack performed by two or more entities who individually do not have enough attributes to satisfy an access policy. The entities will then combine their attributes in an attempt to satisfy the policy. This is an important attack that constructions of PBE must be resistant to and many schemes have been designed to counter such an attack.

PBE schemes can be resistant to such attacks through the use of blinding. With blinding the secret exponent is blinding using a random value, say |the blinding value When decrypting a cipher-text the algorithm must recover the secret exponent as well as this blinding value i.e. e(g; g)^{S} . This value can be removed (blinded out) if the decryption key can satisfy the access policy. This random value and its distribution among the attributes is performed on a per entity basis. Malicious entities cannot combine their attributes and reconstruct the blinding value.

8.6.3 Complexity

The computational complexity of PBE schemes is dependent upon the exact construction of the scheme. Regardless, of exact construction some general observations over the complexity can be discerned. Below the complexity in terms of decryption key and cipher-text, and encryption and decryption times are discussed. For a more in-depth look at the complexity of several constructions of PBE the reader is asked to consult Emura, Miyaji et al. [EM+09].

Size of Key(s) and Cipher-text the size of the cryptographic keys and cipher-text within PBE schemes is inherently dependent on the number of attributes used during key construction. The size of the decryption keys and cipher-text will be based upon the bit-length of the group elements involved, if the construction is based upon pairings. The size of both MPK and MSK will remain constant during operation. For Ciphertext-Policy schemes the size of the cipher-text will be O(n + 1), where n is the size of (number of group elements used in) the access policy i.e. number of leaf nodes, and the one represents the element representing the encrypted message. Decryption keys will be O(j S j). For Key-Policy schemes the size of the cipher-text will be O(j S j +1) where the one represents the encrypted message. For both types of schemes the encryption keys can be represented a plain-old-strings.

Computational Complexity Related to the size of the cipher-text and is the computational complexity associated with the encryption and decryption steps. The following discussions apply to both Ciphertext-Policy and Key-Policy schemes.

Encryption the time required to perform encryption is dependent upon the number of exponentiations performed. For the Waters construction (see Section 8.5) the number of ex-ponentiations required is O(n), where n is the size of (number of group elements used in) the access policy i.e. its leaf nodes.

Decryption The time required is dependent upon: a) the number of pairing operations performed; and b) the minimum number of nodes required for the access policy to be satis ed. For the Waters construction the number of pairing operations required is O(P), where P is the minimum number of nodes needed to satisfy the access policy.

8.6.4 Computational Security

The security of pairing-based cryptographic schemes relies upon the assumption made concerning the di culty of computing some problem within some group model|see Dutta, Barua and Sarkar [DBS04, Section 2] for a partial list of these problems. For the Waters construction discussed, the authors speci cally mention that they use a more formal assumption than the construction given in Bethencourt, Sahai and Waters [BSW07].

Within the construction given in Bethencourt, Sahai and Waters [BSW07] the authors rely upon the security of the generic bilinear group model (cf. Boneh, Boyrn and Goh [BBG05]) and also upon the use of random oracles. Bethencourt, Sahai and Waters argue that their construction itself is secure and that if a vulnerability were to be found then it must exploit speci c mathematical properties of the generating elliptic curve or cryptographic hash function used. The assumption made in Waters [Wat10] is the authors own. The authors de ne and make use of the Decisional Parallel Bilinear Di e-Hellman Exponent (d-Parallel BDHE) in conjunction with a random oracle to support a large universe.

When comparing both constructions the Bethencourt Sahai Waters (BSW) construction has been based upon the generic group model, though this assumption aides in an easier and more expressive construction it is nonetheless not `ideal'. A more formal model should be given. The Waters construction provides an almost identical scheme concerning functionality and perform-ance yet under a more stronger and formal proof model.

8.6.5 Provable Security

Aside from the computational security (see Section 8.6.4), proof of security for PBE schemes can be given. These proofs seek to de ne what it means for a cryptographic scheme to be secure, and the actions that a malicious entity (or entities) must take in order to compromise the scheme. The exact model used to de ne the security of the scheme is dependent upon the type of predicates used, access policy placement, if the scheme is attribute as well as payload hiding, and the security aims of the authors. Moreover, the proofs themselves are also dependant on the construction of the scheme itself. Two common approaches to constructing these models and proofs are that of Game and Simulation based approaches. However, these two approaches do have their di erences [Den06, Section 2 c]. Below an overview of the game and simulation approaches are given together with their di erences.

Game-Based Approach Game-based approaches looks at the interaction between an attacker and an instance of the cryptographic scheme, this interaction is the `game' itself. This interaction is used to determine if the attacker is able to break the security of scheme. If it can be shown that the probability that the attacker is successful is small then security has been achieved. More interested readers are asked to consult individual constructions for di erences in security models and Bellare and Rogaway [BR08, Introduction] for more information on Game based approaches. However, as noted in [Den06], a game based approach only looks at a single instance of the game and does not look at the security of the scheme in a larger context. Simulation is used to address this issue.

Simulation-Based Approach With simulation based approaches the goal is to look at the out-put from the crypto-scheme and compare it to the output from an idealised instance [GMW86]. Both an adversary and a representation of the environment in which the crypto-scheme is placed. The idealised version is used under the proviso that the idealised version of the crypto-scheme is unconditionally secure. If the output from the idealised version is indistinguishable from that of the non-idealised version, the non-idealised crypto-scheme is said to be secure. Hence, if the probability that a di erence can be established between the outputs is small then the crypto-scheme is also secure. Although simulation based approaches presents a stronger security model the proofs are, inherently, more complex. For more information surrounding a simulation based security model, and example proofs, readers are asked to consult Katz, Sahai and Waters [KSW08] and Shen, Shi and Waters [SSW09].