A Middleware Approach for Secure Data Outsourcing in the Cloud

A Thesis
Submitted in Partial Fulfillment of the Requirements for the Degree Of Master of Science in Software Engineering At the College of Computer and Information Sciences At Prince Sultan University

By Mona A. Yahya
May 2016
A Middleware Approach for Secure Data Outsourcing in the Cloud

By Mona Ahmed Yahya

This thesis was defended and approved on 17th of May 2016
Supervisor: Dr. Thavavel Vaiyapuri

Members of the Exam Committee

<table>
<thead>
<tr>
<th>Name</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dr. Thavavel Vaiyapuri</td>
<td>Chair</td>
</tr>
<tr>
<td>Dr. Anis Koubaa</td>
<td>Member</td>
</tr>
<tr>
<td>Dr. Suad Al Ramouni</td>
<td>Member</td>
</tr>
</tbody>
</table>
Dedication

My Parents...

My Husband...

My Sisters and Brothers...
Acknowledgements

I would like to express my sincere gratitude to my supervisor Dr. Thavavel Vaiyapuri whose expertise, understanding and patience added significantly to my experience. I appreciate her knowledge and skills and her assistance in writing this thesis paper. I also would like to thank Prof. Dr. Ajantha Dahanayake for her continues support throughout my study.

My greatest and deepest thanks and appreciation goes to my parents for their endless prayers, support, love and believing in me. Without them, I would not be able to go this far and take a master degree. A special thanks is expressed to my sisters and brothers for their support and being there for me whenever I needed them.

A very special gratitude and acknowledgment to my husband, without his love, understanding and encouragement I would not have finished this thesis.

Finally, I would like to thank all my colleagues at the Computing Services Office specially Eman, Ashwaq and Rana who supported me and encouraged me with their best wishes.
Abstract

Cloud computing has become the new face of computing and promises to offer virtually unlimited, cheap, readily available, “utility type” computing resources. Many vendors have entered the market with different offerings ranging from infrastructure-as-a-service such as Amazon, to fully functional platform services such as Google App Engine. However, cloud users face two challenges in utilizing the cloud storage infrastructure: First, heterogeneity, deploying applications to a cloud and managing them needs to be done using vendor specific methods. As a result application developers are enforced to use different interfaces provided by different cloud vendors to store and fetch data with them due to lack of accepted standard. This seen as a major hurdle in adopting cloud technologies to the enterprise. Second, Data outsourced to cloud is vulnerable to attacks from the internet thieves and from malicious employees of cloud. As result, the data owners are very sceptical to place their data outside their own control sphere.

This Master thesis presents a design of a middleware that extends and improve the iDataGuard middleware combined with a Security middleware that adapts to the heterogeneity of interfaces of cloud and enforces security constraints on outsourced data. This significantly simplifies the effort for application development. The design of the middleware that comprises abstract layer and different techniques and specialized procedures to efficiently protect the data from the beginning to the end, i.e., from the owner to the cloud and then to the user. It commences the process of outsourcing with the classification of data on the basis of three cryptographic parameters presented by the user, i.e., Confidentiality (C), Availability (A) and Integrity (I).

The techniques followed to protect the data utilizes the SSL (Secure Socket Layer) 128-bit encryption and can also be raised to 256-bit encryption if needed to enforce confidentiality, MAC (Message Authentication Code) is used for integrity check of data, data is classified into
three sections in cloud for storage to enhance the data availability. The division of data into three sections renders supplementary protection and simple access to the data.

Finally, on the basis of the proposed design, a prototype is implemented. The main quality of the system is that it not only encompasses all cryptographic operations to ensure confidentiality, integrity and availability of the stored data but are more importantly performed on the client side, which gives the users more control on the security of their data, and thus the data are not dependent on the security solutions provided by the servers.
ملخص البحث

الحوسبة السحابية اصبحت وجهة جديدة من الحوسبة التي توفر موارد حوسبة رخيصة، متاحة بسهولة وافتراضية غير محدودة. قامت العديد من الشركات بالدخول في هذا السوق مع عروض مختلفة تتراوح بين البنية التحتية كخدمة مثل الأمازون إلى منصة الخدمات تعمل بكامل طاقتها مثل محرك تطبيقات جوجل. ومع ذلك، يواجه مستخدمو الحوسبة السحابية تحديات عند قيامهم بالاستفادة من البنية التحتية لوحدات التخزين السحابية. اولا: عدم التناسق، حيث أن نشر التطبيقات على السحاب وإدارتها يحتاج إلى استخدام أساليب مختلفة حسب الشركة التي تقدم هذه الخدمة. ولنتيجة لذلك، يقوم مطورو التطبيقات باستخدام واجهات مختلفة مقدمة من قبل عدد من بائعي خدمة الحوسبة لتوفير واحضار البيانات معهم بسلاسة. ينظر لهذا على أنه العقبة الرئيسية في تبني التقنيات السحابية من قبل المؤسسات.

ثاني: الاستعانة بمصادر خارجية لتخزين البيانات في السحاب معرض لهجمات من الجرائم الإلكترونية ومن باحثي المعلومات بالنتائج. يشكون أصحاب البيانات في وضع البيانات الخاص بهم خارج نطاق سيطرتهم.

رسالة الماجستير هذه تقدم تصميم لوساطة امنة تهدف إلى تجاوز تحديات وواجهات وفروع الامنية على البيانات من مصادر خارجية. هذا يربط إلى حد كبير الجهد في تطوير مباني التطبيقات. تصميم الوسيلة التي تضم طبقة مغزية وتقنيات مختلفة وأجراءات متخصصة لحماية البيانات بكفاءة من البداية إلى النهاية، أي من المالك إلى السحاب ومن ثم إلى المستخدم. تبدأ عملية الاستعانة بمصادر خارجية مع تصنيف البيانات على أساس ثلاث معايير للتشيفر مقدمة من قبل المستخدم، السرية (C)، التوفر (A) والنزاهة (I).

التقنيات المعنية لحماية البيانات تستخدم SSL (البروتوكول ذو الطبقة الأمنة) مع تشغيل 128 بت والتي يمكن رفعها إلى 256 بت. بعد استخدام MAC (رمز مصادقة الرسالة) لفحص سلامة البيانات. تصنف البيانات في السحابة إلى ثلاثة أقسام لتخزين وتعزيز توافر البيانات. تضم البيانات بهذا الشكل تعزيز الحماية ويسهل الوصول إلى البيانات.

أخيرا، يتم تطبيق نموذج على اساس التصميم المقترح. يشمل هذا النظام ليس فقط على عمليات التشغيل لضمان سرية، نزاهة وتوافر البيانات المخزنة لكن الاهم ان هذه العمليات تتفق على جانب العمل، وهذا يعني المستخدمين المزيد من السيطرة على آمن المعلومات الخاصة بهم، وبالتالي فإن البيانات لا تعتمد على الحلول الأمنية التي تقدمها الخوادم السحابية.
# Table of Contents

Abstract .............................................................................................................................. iii

ملخص البحث .................................................................................................................. v

Table of Contents ............................................................................................................ vi

List of Tables ................................................................................................................... ix

List of Figures .................................................................................................................. x

Abbreviation ................................................................................................................... xii

Chapter 1: Introduction ................................................................................................. 2
  1.1 Introduction ........................................................................................................... 2
  1.2 Research Motivation ......................................................................................... 6
  1.3 Research Objectives ......................................................................................... 8
  1.4 Research Questions ............................................................................................ 9
  1.5 Research Methodology .................................................................................... 9
  1.6 Literature Study .............................................................................................. 10
  1.7 Analysis ............................................................................................................ 11
  1.8 Prototype ......................................................................................................... 12
  1.9 Research Contribution .................................................................................. 12
  1.10 Ethical Issues .............................................................................................. 13
  1.11 Chapter Outline ............................................................................................ 13

Chapter 2: Background & Literature Review ............................................................... 16
  2.1 Cloud Computing: Definitions ..................................................................... 16
  2.2 Cloud Key Characteristics .......................................................................... 17
  2.3 Cloud Service Models .................................................................................. 18
  2.4 Cloud Deployment Models .......................................................................... 19
  2.5 Data Security in Cloud Computing .............................................................. 21
3.5 Process Model for the Proposed Approach ........................................60
  3.5.1 Data Storing Process .................................................................60
  3.5.2 Data Retrieval Process .............................................................61

3.6 Conclusion ......................................................................................64

Chapter 4: Implementation and Results .................................................66
  4.1 Introduction ....................................................................................66
  4.2 Model Deployment Architecture ......................................................66
  4.3 Structure of Client GUI .................................................................67
  4.4 Simulation Environment ...............................................................69
    4.4.1 CloudSim ..............................................................................70
  4.5 Security Analysis ..........................................................................71
    4.5.1 DoS Attack Analysis to Ensure Data Availability ....................72
    4.5.2 Man-in-the-middle Attack .......................................................74
    4.5.3 Traffic Analysis ......................................................................75
  4.6 Time Performance Analysis ..........................................................76
    4.6.1 CloudSim Setup Configuration ...............................................76
    4.6.2 Time Performance Analysis Setup .........................................77
    4.6.3 Middleware Processing Time for Proposed Security Service ....78
    4.6.4 Middleware Hosted on CloudSim Security Service ...............81

Chapter 5: Conclusion and Future Work ..............................................85

Glossary ...............................................................................................88

References ..........................................................................................90
List of Tables

Table 2.1: Summary of Middleware Approaches .................................................................48
Table 2.2: Comparison between iDataGuard, Security Approach and the Proposed Middleware ..........................................................................................................49
## List of Figures

<table>
<thead>
<tr>
<th>Figure 2.1:</th>
<th>NIST Model of Cloud Computing [6]</th>
<th>17</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 2.2:</td>
<td>Cloud Deployment Model [7]</td>
<td>20</td>
</tr>
<tr>
<td>Figure 2.3:</td>
<td>Encryption &amp; Decryption</td>
<td>26</td>
</tr>
<tr>
<td>Figure 2.4:</td>
<td>Symmetric Encryption/decryption [19]</td>
<td>27</td>
</tr>
<tr>
<td>Figure 2.5:</td>
<td>Asymmetric Encryption/decryption [19]</td>
<td>28</td>
</tr>
<tr>
<td>Figure 2.6:</td>
<td>Signing and verifying data [18]</td>
<td>29</td>
</tr>
<tr>
<td>Figure 2.7:</td>
<td>Traditional Encrypted Search System [22]</td>
<td>31</td>
</tr>
<tr>
<td>Figure 2.8:</td>
<td>iDataGuard Architecture [3]</td>
<td>40</td>
</tr>
<tr>
<td>Figure 2.9:</td>
<td>TMAHP2P Middleware Architecture [51]</td>
<td>42</td>
</tr>
<tr>
<td>Figure 2.10:</td>
<td>SGCS Middleware Architecture [53]</td>
<td>44</td>
</tr>
<tr>
<td>Figure 2.11:</td>
<td>SSMU Middleware Architecture [55]</td>
<td>45</td>
</tr>
<tr>
<td>Figure 3.1:</td>
<td>Proposed Middleware architecture for outsourcing data securely</td>
<td>52</td>
</tr>
<tr>
<td>Figure 3.2:</td>
<td>Data storing process</td>
<td>61</td>
</tr>
<tr>
<td>Figure 3.3:</td>
<td>MAC Module [4]</td>
<td>61</td>
</tr>
<tr>
<td>Figure 3.4:</td>
<td>Activity Diagram for Confidentiality Module &amp; Authentication Process</td>
<td>62</td>
</tr>
<tr>
<td>Figure 3.5:</td>
<td>Confidentiality Module &amp; Authentication Process</td>
<td>63</td>
</tr>
<tr>
<td>Figure 3.6:</td>
<td>Data Retrieval Process</td>
<td>63</td>
</tr>
<tr>
<td>Figure 4.1:</td>
<td>Model Deployment Architecture</td>
<td>67</td>
</tr>
<tr>
<td>Figure 4.2:</td>
<td>Outsource Data Window</td>
<td>68</td>
</tr>
<tr>
<td>Figure 4.3:</td>
<td>Data Retrieval Window</td>
<td>68</td>
</tr>
<tr>
<td>Figure 4.4:</td>
<td>Simulation tools for Cloud Computing</td>
<td>71</td>
</tr>
<tr>
<td>Figure 4.5:</td>
<td>CloudSim Configuration - Cloud Service Provider (A) and Customer (B)</td>
<td>76</td>
</tr>
</tbody>
</table>
Figure 4.6: Time Performance Analysis for Setup 1 and 2.................................77
Figure 4.7: System performance for run Setup shown figure 4.6 - 1.................78
Figure 4.8: Resource Usage for run Setup shown figure 4.6 - 2.......................79
Figure 4.9: File Upload/Download time for Setup shown in figure 4.6.2 ..........80
Figure 4.10: File Uploading Time against File Size........................................80
Figure 4.11: File Searching Time against File Size ...........................................81
Figure 4.12: Resource Utilization from Customer perspective .........................82
Figure 4.13: Resource Utilization from Cloud Provider perspective ...............83
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSA</td>
<td>Cloud Security Alliance</td>
</tr>
<tr>
<td>EC2</td>
<td>Elastic Compute Cloud</td>
</tr>
<tr>
<td>NIST</td>
<td>National Institute of Standards and Technology</td>
</tr>
<tr>
<td>SaaS</td>
<td>Software as a Service</td>
</tr>
<tr>
<td>PaaS</td>
<td>Platform as a Service</td>
</tr>
<tr>
<td>IaaS</td>
<td>Infrastructure as a Service</td>
</tr>
<tr>
<td>SLA</td>
<td>Service Level Agreement</td>
</tr>
<tr>
<td>ISO</td>
<td>International Standards Organization</td>
</tr>
<tr>
<td>DoS</td>
<td>Denial of Service</td>
</tr>
<tr>
<td>TCP</td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td>VM</td>
<td>Virtual Machine</td>
</tr>
<tr>
<td>U2R</td>
<td>User to Root Attack</td>
</tr>
<tr>
<td>RSA</td>
<td>Rivest-Shamir-Adleman</td>
</tr>
<tr>
<td>IDP</td>
<td>Internet Data storage Provider</td>
</tr>
<tr>
<td>ABE</td>
<td>Attribute Based Encryption</td>
</tr>
<tr>
<td>XACML</td>
<td>eXtensible Access Control Markup Language</td>
</tr>
<tr>
<td>MAC</td>
<td>Message Authentication Code</td>
</tr>
<tr>
<td>DT</td>
<td>Data Translator</td>
</tr>
<tr>
<td>OT</td>
<td>Operation Translation</td>
</tr>
</tbody>
</table>
Chapter 1 ~ Introduction ~
Chapter 1: Introduction

1.1 Introduction

Information technology term is not recent and it is extremely grown fast in the last years. There is no doubt about the big progress of the internet, which is the main factor in IT world, especially with regard to speed in data transfer, both in terms of wired and wireless communication. People run their business, do their researches, complete their studies, etc. by using the facilities available over the Internet. In general, the outsourcing of facility management is becoming more and more common.

Since the need for online services is increasing, the extent of services available through the internet, such as online software, platform, storage, etc., is also growing. This leads to formation of a structured provision of services, called cloud computing, which actually provides a huge amount of computing resources as services through the Internet. One of the important services in the cloud is the availability of online storage, called cloud storage.

Cloud computing is a result of gradually development of providing services by forming clusters and grids of computers. The main concern is to provide a large amount of services in a virtualised manner in order to reduce the extension of server, inefficiencies and high costs. So in cloud computing the servers that are used to provide services, among others cloud storage, are fully virtualised. This virtualisation mechanism makes it possible for cloud storage users to get the specific amount of storage that they need, and thus they are only required to pay for the used storage.

Since this huge amount of services is available online, the use of distributed systems is growing, and thus this new technology, namely cloud computing, is becoming more and
more popular. People are moving towards using cloud storages in order to make use of the advantages, such as flexibility in accessing data from anywhere. People do not need to carry a physical storage device, or use the same computer to store and retrieve their data. By using cloud storage services, people can also share their data with each other, and perform their cooperative tasks together without the need of meeting each other so often. Since the speed of data transfer over the internet is increasing, there is no problem in storing and sharing large data in the cloud.

Cloud storage systems vary a lot in terms of functionality and size. Some of the cloud storage systems have a narrow area to focus on, like only storing pictures or e-mail messages. There are others that provide storage for all types of data. According to the amount of services they provide, they range from being a group of small operations to containing very large amount of services, such that the physical machinery can take up a big warehouse. The facility that houses a cloud storage system is called a data centre. If we just have one data server, and connect it to the internet, it is actually enough to provide a cloud storage system, though it is the most basic level. The common cloud storage systems in the market are based on the same principle, but there are hundreds of data servers that lie at the back end. The computers usually need to be maintained or repaired, so it is important to have copies of the same data on multiple machines. Without this mechanism a cloud storage system cannot ensure data availability to the clients. Most systems store copies of the data to the different servers that are supplied with different power resources. In this way the data would still be available when power failure occurs on one server.

When discussing about all these improvements, we have to remember that there is a very important issue in IT world that must be taken care of, i.e. ensuring security. Users use the cloud storage facility to store and share their data, and especially when these data are secret, the need of security is mandatory. It means that the confidentiality and integrity of data are needed to be ensured. Moreover the stored data must always be available for retrieval, i.e.
the system has to provide availability of data. In short, having security in cloud storage is actually ensuring confidentiality, integrity and availability of stored data.

Many cloud storage providers claim that they provide a very solid security to their users, but we should know that every broken security system was thought once to be unbreakable. As some examples we can mention Google’s Gmail collapse in Europe in February 2009\(^1\), a phishing attack on Salesforce.com in November 2007\(^2\) and a serious security glitch on Dropbox in June 2011\(^3\). If we look a bit deeper in the structure of cloud computing systems, we may feel even more insecure, because they make use of multi-tenancy. Many cloud computing providers work with third parties, so users lose even more trust, especially when they do not know these third parties well. In such a situation users may not dare use the cloud storage system to store their private data. Apart from this, until now there has not been made any standardisation for the security in the cloud. Any software update could lead to a security breach if care is not taken. The mentioned Dropbox security failure was actually caused by a software update. However there are some “local” security standards within every cloud computing system, and some of the providers claim that for every software update, they review the security requirements for every user in the system. Another remarkable issue is the local government laws, and as a result data can be secure in one country, but not secure in the same level in another country. Because of the nature of cloud computing systems as being virtualised systems, users, in most cases, do not know in which country their data is stored\(^4\).

If we look closely at the above mentioned security issues, we would know that there is one central cause for the security problem: Users have no other choices than trusting the

---

servers, because all the security operations are applied on the server side. In fact it is the cloud storage that has the responsibility to provide data security.

Further, in a related trend, there are many applications that are developed being developed that leverage the storage infrastructures provided by the cloud services. Such applications are typically sold to organizations or individual users. Jungle disk\(^5\) is an example of such an application which builds a network drive over the storage offered by Amazon S3. Herein lies the problem. Typically, organizations have service contracts with cloud services of their choice for their storage needs. Every cloud service provides its own interface for storing and fetching data within its infrastructure. There are no standards on interfaces for cloud services and therefore every service designs its own interface. For instance, in the Amazon S3 service, data is stored and accessed using the REST based protocol, and in Open X-Drive storage service the interface is based on the JSON-RPC protocol. The application developer now has the unenviable task of adapting his/her application to different cloud services, or otherwise risk significant reduction in clientele. This greatly complicates application development.

Our goal is to introduce an extended iDataGuard middleware – which is a middleware that offers a secure network drive interface to untrusted Internet data storage - that can run at client machine or in a trusted proxy. The middleware will be responsible for providing security solution for stored data in the cloud as well for providing uniform interface for application development. In this way users do not need to think about servers anymore, though the cloud storage facility needs to guarantee availability of the stored data. Also such middleware can handle the burden of heterogeneity of interfaces and can greatly simplify application development.

\(^5\) http://www.jungledisk.com/
The next parts describes the motivation for this research. Accordingly, it details the objectives that guided to formulate the research questions subsequently it defines the research questions used to help solve the research problem. Later it explains the way in which research is conducted may be conceived of in terms of the research methods utilized (and perhaps developed) in the pursuit of a goal - the research objective(s) - and the quest for the solution of a problem - the research question.

1.2 Research Motivation

Privacy and confidentiality are continuous hot topics. As of 2013, media is flooded with a stream of news related to privacy and confidentiality issues and confrontations. Just to enumerate some examples, one of the most relevant topics is the case of Edward Snowden, an ex-agent of the United States National Security Agency (NSA) who leaked information and details of several top-secret U.S. surveillance and espionage programs to the press. He defends that he did this to inform the public about what is done in their name and for the purpose of transparency. Since June 2013, he is wanted for the charges of “unauthorized communication of national defence information,” and “willful communication of classified intelligence to an unauthorized person”. He is both called a hero and a traitor by sectors of population. Some related news can be found in (6 and 7).

Another relevant news in which Google, as a defence against an action lawsuit for the lack of privacy from Gmail users, declared that whoever uses a service like Gmail should not expect privacy for an information that voluntarily is delivered to third-parties8. Google explained that Gmail Terms of Service (ToS) and Privacy presents its automated mail

---

6 http://www.nytimes.com/2013/06/22/us/snowden-espionage-act.html?_r=0
8 http://www.theguardian.com/technology/2013/aug/14/google-gmail-users-privacy-email-lawsuit
process analysis and that the user, when he accepts the use of the service, he is also obligated to accept these terms. A related news, in the same line, about Facebook is called “Why Facebook Home bothers me: It destroys any notion of privacy”⁹.

As it can be noted, there are a large amount of issues and concerns around privacy and confidentiality of data, and they are a continuous subject of controversy in our society which raises questions difficult to answer due to multiple moral implications [1]. Google Apps, Facebook, and other software solutions delivered through Internet are considered Cloud solutions and are not exempt of these concerns. Some examples of these questions that may arise from Cloud customers: “How secure is my data? Can I trust my cloud provider? Which are the risks and mitigations for any existing issue on my data in the cloud?”. Privacy is a high concern in the security requirement of Cloud Computing. As we will see throughout this document, even though a lot of effort has been put from governments, standard organizations and cloud industry into infusing trust and attract more customers to use Cloud services, Cloud technologies and models have not yet reached their full potential and have not yet acquired a degree to satisfy all potential circumstances of usage.

For these reasons, this Master Thesis aims at providing a state-of-the-art about Cloud Computing security, focusing on privacy and confidentiality matters which are the most significant ones. It will serve as a good starting point for readers interested in getting general knowledge about Cloud Computing and researchers interested in contributing on Cloud Computing privacy and confidentiality matters, allowing them to acknowledge the current status regarding privacy of this trendy paradigm. In this document we identify the issues and challenges of privacy and confidentiality.

⁹ https://gigaom.com/2013/04/04/why-facebook-home-bothers-me-it-destroys-any-notion-of-privacy/
1.3 Research Objectives

Cloud computing users work with data and applications that are often located off-premise. However, many organizations are uncomfortable with the idea of having their data and applications on systems they do not control. There is a lack of knowledge on how cloud computing impacts the confidentiality of data stored, processed and transmitted in cloud computing environments.

Furthermore, cloud users face challenge in utilizing the storage and service infrastructure of the cloud providers due to the heterogeneity: Different cloud service providers offer varied interfaces to application developers in order to store and fetch the data due to lack of accepted standards.

The goal of this thesis is to create a middleware approach that adapts to heterogeneity of interfaces of cloud providers and enforces security constraints on outsourced data by making stepwise recommendations on:

- How data can be classified on confidentiality, integrity and availability?
- How data classifications relate to the security controls needed to preserve the confidentiality of data?
- How process of security control is negatively influenced in cloud computing environments?
- How to cope with negative influences of cloud computing on the protection of data security?
- How to provide security service combating the heterogeneity of interfaces of cloud providers?
1.4 Research Questions

In order to achieve the research objectives stated earlier, the necessary knowledge will need to be obtained and combined. The following research questions will guide this research:

- What are the security requirements for cloud architectures present today?
- What are the essential components required to address these security requirements?
- Do all these components already exist?
- How these components are currently designed and deployed?
- What changes could be made to these components to adapt it to meet the security services?
- How can security service be rendered to combat the cloud heterogeneity interface?

These research questions will help find an answer to the most vital question of this thesis:

“How to design and employ a middleware for outsourcing data securely over Cloud combating the heterogeneity of cloud interface?”

1.5 Research Methodology

The methodology of this thesis is based on a substantial literature study and analysis of already existing security structures and components. The method used is the constructive research method where the key idea is the construction. In [2] Dodig Crnkovic writes that
the construction is “based on the existing knowledge used in novel ways, with possibly
adding a few missing links”. Dodig Crnkovic continues:

“The construction proceeds through design thinking that makes projection into the
future envisaged solution (theory, artifact) and fills conceptual and other knowledge
gaps by purposefully tailored building blocks to support the whole construction”.

The solution in constructive research entails constructing a practical, theoretical or both
practical and theoretical object that “solves a domain specific problem in order to create
knowledge about how the problem can be solved”. As for the construction, it is not mainly
discovered as it is designed and developed.

The research of this thesis is divided in three main phases: literature study, analysis and
prototype.

1.6 Literature Study

The first phase in the research is the literature study as an empirical investigation of the
research problem. Dodig Crnkovic in [2] writes that “only when sufficient understanding
of the research problem and the domain is obtained, one can start addressing a Software
Engineering problem by Constructive Research method”.

In this study, substantial focus is placed on the literature study. The literature used is both
in physical form as loans from libraries and in digital form as e-books. The literature is the
basis for the theoretical chapters regarding theory and theoretical outline of the service. It
is also used to answer the following research questions.

- What are the security requirements for cloud architectures present today?
What are the essential components required to address these security requirements?
• Do all these components already exist?
• How these components are currently designed and deployed?

Furthermore, it is used to fully grasp the huge field of cloud storage infrastructure and Cloud security services.

1.7 Analysis

The analysis is performed in two phases. In the first phase the iDataGuard middleware developed by Ravi Chandra et al [3] and data security approach proposed by Sandeep [4], is studied and analysed. The method of analysis comprise of deploying, testing and evaluating its functionality through provided interfaces. Afterwards, the application code and security structure are examined to find possible and required improvements.

In this phase Design Research will be used as Constructive Research, which “involves the analysis of the use and performance of designed artifacts (constructs) in order to understand, explain and improve designed systems”. [2]

The second phase comprise of determining the suitability of designing a interoperable middleware that adopts heterogeneity interfaces of cloud providers and will ensure data security in cloud is discussed in Chapter 4. This is used as a base for designing the proposed middleware.
1.8 Prototype

The analysis eventually leads to a prototype. The prototype consists of a theoretical description of the architecture, requirements of the system and a simple prototype of the web interface to show the typical functionality the cloud service should provide.

The method of prototyping the web interface can be compared to Extreme Prototyping in Software prototyping where development is performed in three phases. The prototype will only reach the second phase in which the views are constructed and are fully functional without any underlying infrastructure to support the operations. The prototype is evaluated by a proof of concept method to verify that the core ideas are feasible. The core ideas to be tested are discussed later in Chapter 3.

1.9 Research Contribution

This research work attempts to contribute:

- An extended design of the iDataGuard middleware for data security in cloud with the following features:
  - To protect the data from the beginning to the end, i.e. from the owner to the cloud and then to the user by classifying the data on three cryptographic parameters, confidentiality, availability and integrity.
  - To combat heterogeneity of service interfaces of different cloud service provider and reduce the development burden for application developers.

- A web interface prototype illustrating the functionality a cloud service provider should render to ensure data security.
1.10 **Ethical Issues**

All the ethical issues which are to be taken in account while carrying a research study and writing the related work in the form of a report is considered. For example,

- Research participates will be briefed about the aims and objectives of the study and will be acknowledged for the valuable contribution.
- Falsification, fabrication and misinterpretation of data will be avoided.
- Works of other researchers and authors used in research will be referenced and cited.

1.11 **Chapter Outline**

This thesis is organized into four chapters, which will be discussed here one by one to describe our solution and its related topics.

Chapter 2 describes background information and definitions about the Cloud Computing paradigm, with the aim of providing basic knowledge to fully understand the concepts described in the following chapters. Next it elaborates the key characteristics, service models, deployment models, and security issues related to cloud computing. Then it introduces cryptography in which symmetric and asymmetric cryptography is described with special regard to the security and performance of related algorithms

It also explains the performed systematic Literature Review and Analysis on topics of cloud computing and security in order to find answers to the research questions. It brief about privacy and confidentiality applied to the paradigm of Cloud Computing, classifies the issues and challenges found in the literature and outlines the solutions found for these issues or research gaps in the different identified areas.
Chapter 3 presents the design of client side interoperable security middleware that adapts to the heterogeneity of interfaces of cloud providers and enforces security constraints on outsourced data. The middleware design brief the mapping from data encryption to appropriate cloud architectures, and show how the security problems can be anticipated.

Chapter 4 the performance of the proposed model is demonstrated in this chapter. In addition, the simulations tools and results obtained are displayed here

Chapter 5 exposes final remarks about the work done for this Master Thesis and describes future work that could be done related to what we have explained so far.
Chapter 2 ~ Background and Literature Review ~
Chapter 2: Background & Literature Review

In this chapter, in order to completely understand our systematic review about privacy and confidentiality in Cloud Computing, we will introduce background information to position the reader into the context. We will not provide a deep background on Cloud Computing as it is out of the scope of this Master Thesis, but the general concepts that will enable readers to firmly grasp the idea of what is cloud computing and its benefits, and understand the rest of this document. If you are interested in getting more knowledge on Cloud Computing, we firmly recommend reading [5].

As discussed in Chapter 1, It is a requirement to store data securely in the cloud, i.e., the cloud storage facility should ensure confidentiality, integrity and availability of the stored data. Secure storage in cloud computing may be achieved through cryptographic access control. Since the main mechanism in cryptographic access control is cryptography, we next introduce cryptographic techniques that are applicable in existing cloud storage solutions.

2.1 Cloud Computing: Definitions

Various definitions and interpretations of “clouds” and / or “cloud computing” exist. The National Institute of Standards and Technology (NIST) provide a working and official definition of cloud computing [6] [7] [8]:

Cloud Computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers,
storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Other works like [1] and [38] define Cloud Computing as platform or infrastructure in which dynamically scalable (elastic) resources are provided as a service through Internet, enabling users to process the data outside the boundaries of the company, providing economic benefits through virtualized and shared infrastructure without the need of expertise nor knowledge over the underlying technology.

In either definition, both describe a paradigm in which users can demand services through Internet (servers, applications, infrastructure, development platforms) whenever they need it, like a commodity. This saves the necessity of buying expensive 1000€ licenses, which represents cost-savings.

### 2.2 Cloud Key Characteristics

The cloud as defined by NIST is composed of five main characteristics, three service models and four deployment models as shown in figure 2.1. [6] [7] [8].

![NIST Model of Cloud Computing](image.png)

**Figure 2.1:** NIST Model of Cloud Computing [6]
The five main characteristics of cloud computing as defined by NIST are:

- **On-demand self-service**: This characteristic aims to provide clients with computing resources as needed without human interactions. All resources are provided through the cloud service provider website.

- **Broad network access**: Broad Network Access is used to provide access to different resources provided by the cloud service provider over the network using different client platform through a web browser.

- **Resource pooling**: The main purpose of this characteristic is to pool computing resources to serve multiple clients.

- **Rapid elasticity**: Rapid Elasticity is used to help customers to flexibly use and release resources as they need.

- **Measured service**: The use of computing resources are automatically monitored, controlled and optimized in cloud systems by the cloud service provider. [6] [7]

The Cloud Security Alliance (CSA) adds one more characteristics to the list:

- **Multitenancy**: The use of multitenancy helps in sharing the computing resources by allowing the access to the same resource or application by different cloud customers [7].

### 2.3 Cloud Service Models

- **Software as a Service (SaaS)**: This model provides the client the use of applications running on a cloud infrastructure. Those applications are accessible
from different client devices through web or a program. Some of the most famous SaaS applications are Google Docs, and Microsoft Online Services [6] [9] [10].

- **Platform as a Service (PaaS):** PaaS allows the customers to develop and deploy their applications in the cloud and have control over those applications without controlling or managing the cloud infrastructure. Some of the PaaS providers are Google App Engine, Microsoft Azure, Salesforce.com, Engine Yard and Collabnet [6] [9] [10] [11].

- **Infrastructure as a Service (IaaS):** IaaS provides the client with different computing resources; online processing, data storage space or network capability on a virtualized environment. The customers will be able to deploy their software that can include operating systems and applications on this infrastructure and will have control over those deployed software but not the infrastructure itself. The most known IaaS providers are Amazon EC2, Terremark, Rackspace and IBM Cloud [6] [9] [10].

## 2.4 Cloud Deployment Models

- **Private Cloud:** Private Cloud is exclusively used by a single organization and it can be managed by the organization, a third party or a combination of both. This type of cloud can be on or off premises [6] [12].

- **Community Cloud:** Community Cloud is exclusively used by a specific community from organizations that shares the same concerns. It can be owned by
one of the organizations in the community, a third party or a combination of both and it can exists on or off premise [6] [12].

- **Public Cloud**: Public Cloud is used by the public and it exists on premise of the cloud provider. The cloud provider is responsible for maintain and secure the data [6] [12].

- **Hybrid Cloud**: Hybrid Cloud is a combination of two or more cloud deployment model [6] [12].

![Cloud Deployment Model](image)

Figure 2.2: Cloud Deployment Model [7]

Figure 2.2 depicts the summary of the four primary cloud deployment models. It should be noted that there are initiatives for deployment models that not necessarily fall inside one of the below categorizations. For example, Amazon offers virtual private clouds that use public cloud infrastructure in a private manner connecting the public cloud resources to the organization’s internal network [7].
2.5 DATA SECURITY IN CLOUD COMPUTING

Protecting user’s data is a critical security concern for almost all organizations. Cloud users must identify the data that need to be secured and categorize them based on their effect on security and then define the security policy for data protection and the policy enforcement mechanisms before moving into the cloud. Various types of data involves various value of protection to ensure data confidentiality, integrity and availability [13].

2.5.1 Security Principles and Data Requirements

The major security service for data security should guarantee confidentiality, integrity and availability. The matter of data security turns out to be complicated in cloud computing due to the inherent of cloud characteristics [13]. The International Standards Organization (ISO) defined six cloud computing principles and data requirements: [13] [14]

- **Identification and Authentication**: It is very important to identify the users accessing the cloud by identifying their user ID then validate this ID with the password in cloud database system.

- **Authorization**: After identifying and validating the user, they should be given proper access permissions to different cloud resources. The user might be able to access one resource but not the other for that, authorization is used.

- **Confidentiality**: Confidentiality is a must when using a public cloud due to the access nature of this type of cloud. Also it is a major principle that should be used to maintain the control of organization’s data that are located in various distributed databases.
• **Integrity**: Keeping data integrity and make sure it is not altered by unauthorized user is very important. For that, the use of ACID properties which are Atomicity, Consistency, Isolation and Durability should be pushed across all cloud computing delivery models.

• **Non-repudiation**: There should be a type of security protocol such as digital signatures, timestamps and confirmation receipts services used when transmitting data across the cloud to keep from rejecting authenticated access.

• **Availability**: Availability is one of the most important requirement that should be kept in mind when choosing cloud deployment models (public, private or hybrid). Therefore, SLA should provide information on the availability of cloud services and resources between cloud provider and clients.

### 2.5.2 Cloud Environment Security and Privacy Concerns

There are several security issues related to cloud computing, some of those issues are: privileges and access to the data in the cloud which is a very important issue that must be taken into account. Another issue is the location of the data hosted in the cloud. You will not know where your data are hosted and if this place has different policies they follow regarding privacy issues; for this you have to make sure your data is save and confidential where no one can access it. The shared environment in the cloud where the data are hosted is another security issue. Clients should make sure that their data is encrypted and saved and only they have access to it. All those issues and more should be kept in mind when deciding to move data to the cloud [14].
The authors of “A New Technique of Data Integrity for Analysis of the Cloud Computing Security,” provided detailed analysis of the cloud security problem. They provided information about the cloud data storage architecture when clients have large amount of data and want to store their data in different clouds. They also discussed the differences between cloud models and analyze the issue of cloud data storage [15].

An analysis of cloud computing security issues is a paper that discusses cloud security problems from different cloud perspectives like architecture, characteristics, delivery model and stakeholders. It also points some main research challenges of applying cloud-aware security solutions [16].

In this paper, the authors pointed out some open research problems for cloud computing model. Those problems are cloud security, vendor Lock-in, multi-tenancy, secure data management, service portability and SLA management. Moreover, in their analysis they focused on the following data security issues:

- **Multi-tenancy**: involves sharing of resources, storage, services, and applications with different tenants. It is not easy to secure shared data for that there should be a level of isolation for tenant data.

- **Elasticity**: involves scaling up or down resources allocated to services by consumers based on the current demand. This makes the data exposed and leads to confidentiality issues.

- **Availability of Information SLA**: insuring that data are available whenever the organization needs it is not easy. There might be different problems occurring at the provider’s side that lead to data unavailability. For example, a breakdown or attack on the provider’s storage or resources. For that, there should be a Service
Level Agreement between consumers and cloud provider to define the maximum time for which data are not available to the consumer for use. Moreover, there should be a system that alerts the consumer about data being offline and the possible downtime.

- **Secure Information Management**: security management should include security requirements and policies specifications derived from tenant organizations that are reviewed and applied in tenant’s logical and physical environment.

- **Information Integrity and Privacy**: some of the main information privacy issues are lack of authentication and authorization and no management of encryption and decryption keys. The solution for this issue is to maintain a trust between providers and consumers to achieve confidentiality and integrity of information.

- **Cloud Secure Federation**: when multiple clouds integrated together to bring a bigger pool of resources their security requirements needs to be federated and enforced on physical and logical different cloud platforms. [16]

“State of the Art Survey on Cloud Computing Security Challenges, Approaches and Solutions” is another paper addressing the same issue of cloud security and privacy concerns. The author here provided six main security risks that should be taking care of when moving to cloud. Those risks are [17]:

- **Outsourcing**: data control might be lost and cloud providers might misuse customers’ data. For that, proper tools should be used to prevent such data lose or misuse.
• **Extensibility and Shared Responsibility**: there is a trade-off between extensibility and security responsibility for customers in different cloud delivery models.

• **Virtualization**: flexible access control system should be used to insure strong isolation techniques and secure sharing and communication between virtual machines.

• **Multi-tenancy**: a secure multi-tenant environment should be obtained to inforce access policies, application deployment and data access and protection

• **Service Level Agreement**: a secure negotiation method should be available for the communication between service providers and service customers.

• **Heterogeneity**: integration challenges may occur when multiple cloud providers uses different approaches to provide security and privacy techniques.

### 2.6 Cryptography In Cloud Computing Environment

Cryptography as described in [18] is the most common technique for ensuring a secure communication between two parts in the presence of a third party. If we have two parties $A$ and $B$ sends messages to each other keeping in mind that this message is private and no one should read it or change its content. A secure communication channel should be used to provide message integrity and a transmission method $T$ is used to transfer this message between $A$ and $B$. A third party or an intruder $I$ tries to interfere with this message by accessing it or changing its content. Every time a message is on its way to the destination, it is in danger of being accessed by $I$, who can perform the following actions:
1. *Block* the message, so it never reaches its destination therefore the availability is violated.
2. *Intercept* the message, so it is not secret anymore in this way the confidentiality is destroyed.
3. *Change* the content of the message thus the integrity is violated.
4. *Fake* a message and impersonate the sender $A$ and send the message to $B$. This violates also the integrity of the message.

### 2.6.1 Encryption & Decryption

Data encryption and decryption which are also called encoding and decoding, or enciphering and deciphering are the techniques used in cryptography. Encryption, encode or encipher is a technique by which the original text which is known as the plaintext, is altered, such that the meaning of the text is hidden, that is the plaintext is converted into a meaningless string of text called Ciphertext. In order to convert the Ciphertext back to the plaintext, it has to be decrypted, decoded or deciphered. Figure 3.3 shows an overview of encryption/decryption procedure. [19]

![Encryption & Decryption](image)

Figure 2.3: Encryption & Decryption

In the example shown in figure 2.3, the plaintext $P$ is considered as a sequence of characters $P = \langle H,e,l,l,o,\ W,o,r,l,d,!\rangle$ and in the same way the Ciphertext $C =$

```
A system that encrypts and decrypts data is called a cryptosystem. If we represent the two processes in a cryptosystem formally, it would be $C = E(P)$ and $P = D(C)$, where $C$ is the Ciphertext, $P$ is the plaintext and $E$ and $D$ are encryption and decryption algorithms respectively. The cryptosystem is represented as $P = D(E(P))$, which means that the plaintext $P$ is the decryption of encrypted $P$.

### 2.6.2 Symmetric & Asymmetric Algorithms

In cryptosystems a key $K$ is usually used with an algorithm in order to encrypt or decrypt the data. If the same key is used for both encryption and decryption, then the process is called symmetric encryption, and the key is called a symmetric key. In this case the encryption and decryption algorithms are symmetric and they can be considered as reverse operations with regard to each other. The formal notations would be $C = E(K, P)$ and $P = D(K, C)$, and the cryptosystem is represented as $P = D(K, E(K, P))$.

If the key used for encryption is different from the one used for decryption, then the process is called asymmetric encryption. Here we use two keys, namely an encryption key (often called private key) $KE$ for encryption and a decryption key (often called public key) $KD$ for decryption. The formal notations in this case would be $C = E(KE, P)$ and $P = D(KD, C)$ and the cryptosystem is accordingly denoted as $P = D(KD, E(KE, P))$. Figure 2.4 and Figure 2.5 show overviews of the symmetric and asymmetric algorithms. [19] [20] [21]

![Symmetric Encryption/decryption](image-url)
2.6.3 Digital Signature

Digital signature is used to insure integrity of data, and it has the same principle as the handwritten signature. The difference is that if a digital signature is implemented properly, it is much more difficult to fabricate than the handwritten signature. To apply a digital signature to messages, asymmetric encryption is used. For instance, Alice wants to send a message to Bob. The message can be encrypted or not, but people usually encrypt the message to insure secrecy. Then she generates a pair of keys, i.e. a private key and a public key. She keeps the private key secret, and publishes the public key. She signs her message using the private key, and then she sends the signed message to Bob. When Bob receives the message, he tries to verify the signature by using the corresponding public key. If the signature is verified successfully, then he is sure that the message is untouched and the actual sender of the message is Alice. If the verification is failed, then Bob knows that either the message has been tampered with, or it is not sent by Alice at all. Figure 2.6 describes message singing and verifying. [18]
2.6.4 Hash Functions

Hash functions are also widely used, especially in digital signatures. A hash function produces a short and fixed length message digest, which is unique for each message. So it is a great advantage to use a short message digest for digital signature, instead of the whole message, especially if the message is too long.

The main requirements for the security of hash functions are that they must be one-way functions, and they must be collision resistant. A collision occurs when for two different inputs, the hash function gives the same output, for instance, $\text{hash}(m_1) = \text{hash}(m_2)$.

Cryptographic hash functions are classified into un-keyed hash functions and keyed hash functions. Un-keyed hash functions, also known as Modification Detection Codes (MDCs), use message as a single input whereas keyed hash functions, also known as Message Authentication Codes (MACs), can be viewed as hash functions which take two functionally distinct inputs, a message of arbitrary finite length and a fixed length secret.
This research work deals with MACs to ensure data integrity as they rely on data owner secret key. [18]

### 2.7 Searchable Encryption

Several researches have addressed the issue of ensuring confidentiality and privacy of cloud data without compromising the user functionality. Here, confidentiality refers to the secrecy of the stored data so that only the client can read the contents of the stored data. To solve the problem of confidentiality, data encryption schemes can come in handy to provide the users with some control over the secrecy of their stored data. This has been adopted by many recent researches which allow users to encrypt their data before outsourcing to the cloud. However, standard encryption schemes will reduce users’ searching ability over the stored data, since after encryption a user simply cannot use a plaintext keyword to perform a search anymore and cannot retrieve the contents in an efficient way. Therefore, to ease the data retrieval from a secure cloud, we need a scheme which enables user capability to search over encrypted contents.

To provide a secure and efficient retrieval of data, one needs to ensure that the user can perform a search over the encrypted data without revealing the contents and the searched keyword to the server. The cryptographic primitive that provides this feature is widely known as searchable encryption (SE).

#### 2.7.1 Searchable Data Encryption Architecture

The traditional encrypted search system over the cloud consists of three main applicants; provider, cloud and the user see figure 2.7 [22].
• **The Provider:** owns a group of documents and their indexes. The provider plans to outsource these documents to the cloud and allow users to contact the cloud for search services.

• **The Cloud:** is a business corporation that offers computation and storage resources.

• **The User:** is a person who presents keywords to search documents that have these keywords.

As you can see in figure 2.7 the process of searching encrypted data over the cloud goes into three main steps: documents and indexes uploading process, trapdoor generation process and document retrieval process [22].
• **Documents and Indexes Uploading Process:** At the beginning, the provider examines all words in the documents to be saved in the cloud and preserves these terms. After that, each term is encrypted and treated as one index keyword. The number of occurrences of each term in the document is added up and recorded into the matching entry of the document index. At the end, this index will be encrypted and outsourced to the cloud along with its encrypted document [22].

• **Trapdoor Generation Process:** First, the user should be authenticated with the provider. The provider will send a secret key to the user during the authentication process to decrypt the documents saved in the cloud. As soon as the user is authenticated, he would send the search keywords to the provider then the provider would processes trapdoors and replies back [22].

• **Document Retrieval Process:** The user sends the noised trapdoor to the cloud. Then the cloud eliminates the noises and searches the indexes to look for the documents requested by the user. Once the documents are located, the cloud levels them corresponding to the score of each document. After that, the top related documents are sent to the user. At the end, the user decrypt and recover the documents [22].

### 2.8 Middleware

The earliest approach of software development merely involves programming and writing code. The biggest drawback is that the majority part of the codes is unable to be reused. After this, the component-based software development method was proposed. It packages common programming features, and provides standardized and unified interfaces which can be invoked by other programs. With the component-based software development
approach, it is possible to achieve system modularization. To put this development approach into practical use, middleware is the key enabling technology. Middleware is a kind of computer software that connects software components and applications. [23]

“Middleware is an independent program of software or service. It does not only achieves interconnection, but also realizes interoperability between different components in the application. It consists of a set of enabling services that allow other program processes to interact with.”

Nowadays, middleware is widely used in a variety of large-scale projects. It provides services, supports application functions, separates concerns, and integrates components. Both the "polymorphism" of an application and the complexity of system software must rely on the various types of middleware, which play a role in the logistics and mediation.

The advantage of using middleware is obvious. The program interface of the middleware creates a relatively stable high-level application environment. Regardless of how the bottom computer hardware and system software changes, the application software requires almost no modification, as far as the middleware updated while the middleware external interface definition unchanged. This property can assist enterprises to reduce the significant investment based on the application software development and maintenance.

It is obvious from the above that the middleware solution is a good choice to achieve the goal in this case. The main task for the author in relation to this research is in designing and implementing a heterogeneous data source middleware which can provide different kinds of data source services. The middleware is supposed to assist other team members and, the application developers, in order to build an elegant framework which is loose coupling and high cohesion. It is supposed to make the system easier to develop and
maintain. In addition, this middleware can also be applied to other applications based on its generality and reusability.

2.9 REVIEW ON DATA CONFIDENTIALITY

Encryption, as explained earlier in this chapter, is the process of encoding messages or information in a way that only authorized parties can read this information, preventing unwanted parties and hackers read it. To protect the confidentiality of sensitive data stored in the cloud, encryption is the widely accepted technique in Cloud Computing architectures. In fact, encryption is the only recognized standard for data protection, like the NIST Federal Information Processing Standards (FIPS) [24]. Nevertheless, encryption is not all-purpose as it alone cannot provide complete solutions to all privacy issues in cloud computing, and it complicates query processing on the data.

Most of the evidence found in the literature agrees that the biggest problem regarding encrypted data in the cloud is data access efficiency while preserving confidentiality. In the literature, several papers which deal with encryption issues and proposals have been found. Although encryption seems a favorable solution to data confidentiality, an important aspect has not yet been discussed. Can we perform efficient operations on an encrypted data? A cloud acting as a simple storage platform is not a feasible economic model. It must enable us to perform operations on the data. At the very least, a cloud must act as a virtual database which can process queries and retrieve records from. Until now no major cloud provider has come up with a facility to search on encrypted data. For instance, on Amazon's SimpleDB, encrypted data cannot be used as a part of query filtering conditions [25]. The only way to run queries on encrypted data is to retrieve the entire dataset, decrypt it and then run queries. Clearly, it is not a practical solution for large databases. Even with technologies like Transparent Data Encryption used by Microsoft and Oracle, which
encrypts the physical files rather than the data itself, one still cannot run queries on encrypted data and must rely on decryption of the data before querying.

In the past decade, a lot of literature has emerged in the domain of searchable encryption. Such a scheme generates a search index, over the full-text or a keyword, and encrypts this index. An authorized user is given a token, using which files that contain a keyword can be retrieved. The token can only be generated using a key which is only available to an authorized user. Without the token, the index is not revealed. The output of such a retrieval process does not reveal the contents of the files. It only indicates that the files have a keyword in common.

First searchable encryption scheme was proposed in [26]. In the early years, only single-keyword search was supported [27], [28] and [29]. Multi-keyword search involving conjunctive and disjunctive queries have been proposed in [30], [31], [32] and [33]. Also, [34] and later improved in [35], introduced the concept of searching on a part of the encrypted data using attribute based encryption which also concealed the token. These work indicate the possibility of performing search on encrypted data, however, no real system even at a moderate scale has proved the practicality of encrypted search. The complexity of such operations is high, making them impractical for large scale operations such as on a cloud. Recently, fully homomorphic encryption was developed in [36]. It allows algebraic operations to be performed on plain data such that it is equivalent to performing the operation on encrypted data. It is described as the "holy grail" of cryptography. It would enable searching on encrypted data in a cloud system as if searching on unencrypted data without exposing the data or the search query to the cloud provider. The theoretical framework is in place but it is still far from being practical. It takes immense computational power. A single Google search would take would be one trillion times slower with fully homomorphic encryption and in the future is expected to be reduced to
100,000 times the computation required for unencrypted computing. In [37] and more recently in [38], it is evident that fully homomorphic encryption is not yet sufficiently efficient enough for any practical application. The idea of using a hybrid cloud has also emerged in the academia. The work in [39] proposes to use two clouds. A cloud which is trusted encrypts the data in the setup phase and performs security-critical operations. This data is sent to an untrusted cloud which handles high load of queries and communicates with the trusted cloud using a secure channel. Still the performance aspect is not discussed and we believe it is not as efficient as if encryption is still involved. Secondly, the reliance on assigning one cloud as the trusted party does not eliminate the possibility of a data breach within that cloud.

Using trusted hardware instead of a software service provider is introduced in [40]. A server-side trusted hardware is used to achieve full privacy control on the data. However, trusted hardware has performance bottleneck when working with large data due to heat dissipation and are constrained in computing and memory capacity. Besides, although their system achieves better performance than a complete encryption based system, it is far less efficient than querying on an unencrypted database. A trusted hardware is also not immune to attacks, such as, during bootstrapping [41].

Most recently secure ranked keyword search over encrypted cloud data has been propose in [42]. In this model, a secure searchable index is built from distinct keywords that are a part of the data. The index and the encrypted data are sent to the cloud and when a query in the form of a keyword is received, cloud searches the index and returns ranked results. Searching over untrustworthy servers and retrieving certain top results using a confidential index were discussed in [43]. While, [44] presents ranked search over order preserving cryptographic function. However, [43] is inefficient and [44] does not support dynamic

---

changes in score. The work in [42] uses a new order preserving symmetric encryption scheme.

The mentioned papers closely addresses the problem we present in this research work. This makes our approach more suitable to adopt searchable encryption technique as we address the problem of data confidentiality while outsourcing data over cloud.

2.10 REVIEW ON MESSAGE AUTHENTICATION CODES

Deswarte in [45] introduced the first solution to remote data integrity that uses the RSA-based functions to hash the entire data file for all authentication challenges. This technique it not effective on large data files that requires long time to compute and transfer their hash values. In [46] Caronni introduced a different protocol that uses a server to send MAC data as an answer to the message as a replacement for saving the hash of the entire data. A unique random key is sent by the verifier for the MAC to gain data integrity. In this way, a segment of data is saved on the server not the whole data. Another scheme that uses homomorphic provable tag which is calculated as a number equivalent to two times of the data chunks number and saved both the data file and its tags on the server was introduced by Ateniese. The user can later use the queried blocks and their matching tags to validate data integrity and in the same time the server produces a proof of integrity [47]. Wang in [48] examined the issue of certifying that data storage is available and secure in cloud computing for the dynamic data integrity authentication. They applied the homomorphic token and error correcting codes to get the combination of storage correctness protection and data error localization. These literatures motivated to adopt MAC in this research work to ensure data integrity. Nevertheless, it was also understood from these literatures that the strength of data integrity depends on the cryptographic strength of the underlying hash functions. This strived to review the literatures to determine the widely used hash functions.
Hash functions are commonly used in digital signatures. It produces a unique message digest with short and fixed length. Hash functions must be one-way functions and collision resistant to guaranty their security. Message Digest Algorithm (MD2) is the oldest hash function algorithm. MD2 was developed in 1989 by Ronald Rivest but it had several security issues. For that, Ronald developed MD4 a newer version of MD2 in 1990. MD4 also showed to be insecure and Ronald had to introduce MD5 to overcome those issues. MD5 come to be popular and commonly used, but in 1994-95 collisions were found in MD5. In 1993 NIST introduced a Secure Hash Algorithm (SHA-0). SHA-0 showed some critical faults which was replaced by SHA-1 a 160 bit hash function. SHA-1 also showed some security issues which made NIST develop a newer version called SHA-2 family algorithms. SHA-2 family didn’t report any security issues and is the most commonly used hash function algorithms [18] which is adopted in this research work.

2.11 Review on Middleware Approaches for Security

A middleware which was briefed in section 2.8 is a software layer residing on top of the operating system that connects different software components or applications. It provides interoperability and other services like the distribution of functionality, scalability, load balancing and fault tolerance [23]. Since middleware handles all the communications between a client and the target application, it can also mask all the underlying complexities including the security aspects. The security is enforced within the middleware by utilizing many security features such as authentication where identities are verified, credentials are handled, messages are protected from unauthorized modifications and disclosure, and access control policies are managed and audited for accountability. These security features are all integrated in a way that preserves the main functionalities of the middleware and fits into its structure.
Following are some examples of middleware approaches examined in recent researches done in this field. Those examples illustrates several areas and application domains which contains safe approach to wireless devices and applications, strong computing resources and sources of data and storage.

2.11.1 iDataGuard

iDataGuard was designed at the University of California and Brigham Young University [3]. This middleware offers a secure network drive interface to untrusted Internet data storage. It is also an interoperable security middleware that tackles heterogeneity together with security issues considering IDP. Apart from the success of IDPs, it is not fully controlled and it still have some limitations. One of those limitations is trust, since the data is saved in plain text and this makes it exposed to malicious attacks. The other limitation is the lack of standard protocol for the IDP services interface design. The iDataGuard was developed to mask the heterogeneity of the interfaces and handle the security requirements.

iDataGuard is placed inside a secure network drive which is built above any IDPS; it operates on the user machine to deliver the data services to them. The middleware architecture (figure 2.8) contains: Data Translator (DT) which translate the outsourced file into an abstract data model. This abstract model models objects and includes the data models of a range of IDPs. Those objects are protected cryptographically by the Crypto module that uses a master password from the user. Getting the objects and saving them is done by specific modules of IDP service adapters. The application issues all system operations files directly to the iDataGuard. Those files are then translated by the Operation Translation (OPT) into abstract operation modules which are a group of generic operations suiting a range of IDPs. The index generator component is also available and it generated
a cryptographic index for the entire outsourced textual files. The capacity to adapt to heterogeneity is one of the main features of the iDataGuard. Other features are its security mode and the capacity to search data that are encrypted. Using cryptographic methods, the security model proposes user’s data confidentiality and integrity. These methods hide entire content, metadata and structure of a file system and don’t expose those information to the server.

![Figure 2.8: iDataGuard Architecture][3]

2.11.2 AMUWA

In order to adopt ubiquitous web access, an authentication middleware approach was designed at the University of Manchester, UK [49]. This middleware is utilized to protect ubiquitous data access across web services. It uses a collection of various types of authentication techniques corresponding to several places to address security problem. In this way, the middleware uses several levels of authentication corresponding to the place.
of the user at the time he requests for a service. This middleware attempts to address some problems concerning authentication in which the help of both heterogeneous authentication and authorization methods are needed. What plays a huge role in strengthen the authorization and authentication offered to access specific services is the access control policy. Furthermore, it is essential to allow a user working away to be authenticated without the use of strict technology and authentication procedures to support user roaming.

The AUMA middleware is based on FAME-PERMIS (Flexible Access Middleware Extensions to PERMIS) that is planned to improve multi-factor authentication in addition to authentication power that helps a large group of techniques that includes IP addresses, a pair of username and password, certificate-based soft tokens, Java cards and biometrics. Therefore, the user will have the choice to select any grouping of authentication techniques to get a great Level of Assurance (LOA). FAME consists of three key elements: the Authentication Token Manager (ATM), the Device Manager (DM) and the Network Manager (NM). The middleware attempts to track a standard that will describe the power and the level of the authentication technique using the NIST standard that describes four levels of authentication strength.

2.11.3 TMAHP2P

Carlos III University of Madrid developed a trust-based middleware that provides security to ad hoc Peer-to-Peer (P2P) applications [50]. This middleware is applied to secure digital content shared between universal devices. It is also used to deliver adaptable middleware that achieves an independent trust management and excludes the complexity of starting an extra trust-based connection with new devices. Furthermore, it minimized the need of a central server and removes the demand of a manual settings. As a result, whichever device can contribute in a P2P application. Figure 2.9 demonstrate the work of this middleware.
The TMAHP2P middleware is aimed to be open and extensible. The architecture of this middleware contains three layers: 1) Trust Layer, 2) Security Layer and 3) Application Layer. An access control based on XACML-complaint polices is used to connect the three layers and this will allow or reject access to the resources.

- **Trust Layer**: controls a trust manager that is used to operate four processes: specifications, establishment, monitoring and trust termination.

- **Security Layer**: consists of the authentication manager, API communication, cryptographic provider and credentials manager.
• **Application Layer:** consists of the application developed by the Wireless and Secure File Exchange Protocol (WSFEP). This application goes across two stages: discovery and exchange. Discovery stage is where devices tries to identify the environment surrounding them and determine the files available for exchange. Exchange stage is where the real files exchange take place. There are two modes for the file exchange it can be either efficient mode where files are transferred without being secured or secure mode where files can be either signed, encrypted or both.

### 2.11.4 SGSC

This middleware is known as Secure Group Communication Service that is used for mobile ad hoc networks and it was developed at Arizona State University [52]. The SGSC offers flexible secure group management (figure 2.10). The development and execution of distributed applications is supported as well. Because of the narrow abilities of the members and their mobility, mobile ad hoc networks enforces an extra requirements for secure group communication. The developers used context-sensitive middleware to develop the SGSC and it contains the following elements:

1) SGCS Daemon which is used to create and manage services.
2) Group Services which offers core functionality and is created on the devices themselves.
3) SGSC API which is used by the distributed applications to utilize secure group functionality.

The next steps are required to develop and execute distributed applications using SGCS:

- The use of RCSM middleware to develop a distributed application with secure group communication.
Categorize the members of every secure group required by the distributed application.

The SGCS will automatically manage the secure groups after running the distributed application which will support secure communication among the group member.

Figure 2.10: SGCS Middleware Architecture [53]

2.11.5 SMMU

SMMU supports ubiquitous computing environment using a security management middleware [54]. This approach authorizes administrators to describe the required policies and offers management services to monitor and control the joined devices. Offering trust management services and assisting real-time mobile application scenarios is the focus of this middleware architecture. There are several benefits regarding this middleware; it offers
authentication, checks permissions and monitors activities of the connected devices see figure 2.11.

![SSMU Middleware Architecture](image)

Figure 2.11: SSMU Middleware Architecture [55]

The architecture of this middleware contains the following elements:

- A policy manager that is applied to describe access control rules, privacy and accessibility of resources and saved information
- An object manager that is applied to monitor and control target devices.
- A context manager that gathers the contexts and combines the data chosen to be directed to the status monitoring manager.
- A status monitoring manager that gathers the status information to send it to the database manager.
- An authentication manager that operates X.509 credential for identification.
Those are only some examples of the middleware approaches. A summary of other approaches can be found in table 2.1.

<table>
<thead>
<tr>
<th>Middleware Approach</th>
<th>Target Environment</th>
<th>Problems Tackled</th>
<th>Security Tools</th>
<th>Characteristics</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSMAP [56]</td>
<td>Ubiquitous environments that uses security management middleware</td>
<td>Access control, securing data, heterogeneity and authentication</td>
<td>Reconfigurable on-demand elements. Compulsory access control using dynamic sensitivity levels</td>
<td>Virtual server offers on demand collected security elements</td>
<td>It is still not applied and tested in a real platform</td>
</tr>
<tr>
<td>MSHAC [57]</td>
<td>Secure access and control for home automation and works on TCP/IP, IEEE 802.11 or Bluetooth</td>
<td>Access control, securing data, low resources and authentication</td>
<td>Diverse authentication techniques with several security levels</td>
<td>Context and user profile that are used for authorization and offers more intelligent and secure location-based services</td>
<td>The offered architecture is applied and tested as a proof of concept</td>
</tr>
<tr>
<td>TDAMU [58]</td>
<td>Ubiquitous environment that uses role-based and distributed authentication</td>
<td>Access control, group communication, low resources, authentication, heterogeneity,</td>
<td>Tasks given to groups of mobile object to help managing security policies. It</td>
<td>Security offered over communication s of three layers in each mobile device.</td>
<td>This is a joined scheme that offers middleware security between</td>
</tr>
<tr>
<td>DMW [59]</td>
<td>This middleware imposes Digital Right Management in P2P networks using mobile and stationary devices</td>
<td>Access control, authentication, securing data and mobility</td>
<td>Planned for the application layer, hardware and operating system</td>
<td>Offers a theoretical framework that focus on DRM rule execution and provides services like obtaining content, access/copy control and right analysis</td>
<td>It is not evaluated experimentally and it is valuable to be used as a guideline for upcoming architectural designs</td>
</tr>
<tr>
<td>----------</td>
<td>-------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------</td>
<td>-------------------------------------------------</td>
<td>--------------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>S-MARKS [60]</td>
<td>Ubiquitous computing environment that used secure middleware</td>
<td>Access control, low resources, mobility, heterogeneity and authentication</td>
<td>Tackles device validation, handling harmful recommendatio ns, trust-based resources detection and avoiding</td>
<td>Determining devices according to response, challenge and recommendation of a new node. This trust model is reflexive and</td>
<td>A prototype is implemented on portable PC’s using C# and .net framework and it supports device validation and</td>
</tr>
</tbody>
</table>
2.12 Literature Review Conclusion

Although we performed a literature search on different, high ranked journals, we found little to zero literature linked to Middleware approaches to address the security issues closely related to our research on Cloud Computing. This proves that our research area is in its infancy and there are a lot of open issues to be answered.

During the literature review, some papers were found on the topic of encryption with the outcomes that searchable encryption are widely used to data confidentiality in cloud. Similarly, from the literature review it is clear that the researchers adopt MACs to ensure data integrity in Cloud. Although data security issues are addressed by several researchers it was found that few attempts were made towards delivering the security requirements to meet the heterogeneity that arise across different cloud service provider interfaces.

Thus, the literature review did not produce the information needed to answer the research questions specified in the Chapter 1. However, we did find three interesting concepts on how data security issues are addressed in cloud and how middleware is used in ubiquitous computing to provide database and management services. In the following chapter we explore how these concepts can be integrated in our research, by mapping them to dimensions in the cloud computing context.
Table 2.2 compares between the iDataGuard, Security Approach and the proposed middleware. It also provides information about the proposed middleware contribution to the iDataGuard middleware.

<table>
<thead>
<tr>
<th>Features</th>
<th>iDataGuard</th>
<th>Security Approach</th>
<th>Proposed Middleware</th>
<th>Proposed Middleware Contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Implement MAC module to iDataGuard to ensure integrity</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Add Classification, Index building and encryption to iDataGuard to maximize data confidentiality and availability</td>
</tr>
<tr>
<td>Heterogeneity</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Cryptography</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

Table 2.2: Comparison between iDataGuard, Security Approach and the Proposed Middleware
Chapter 3 ~ Proposed Middleware ~
Chapter 3: Proposed Middleware

3.1 INTRODUCTION

This chapter describes the functional requirements in-line to the research question defined in Chapter 1. Then, it describes how the core components of the proposed Middleware architecture addresses the heterogeneity of interfaces provided by different IDPs and also capable of outsourcing the data securely with respect to the major security parameters such as confidentiality, integrity and availability. Finally, it describes the process model to illustrate how the dynamic behaviour of the core components satisfies the functional requirements and subsequently how it answers the research question defined in Chapter 1.

3.2 FUNDAMENTAL REQUIREMENTS OF MIDDLEWARE APPROACH

As discussed in Chapter 1 and 2, there is pressing need to design a Middleware that is generic enough to encompass all the IDPs that are currently functioning on the Internet. The following requirements are expected from a Middleware

- It should be able to describe abstractly the data format and the functionality that IDPs confidentiality and integrity of the objects outsourced.
- It should place minimal requirements on the functionality of the IDPs.
- It should be able to preprocess data outsourced by the client applications with security concerns such as confidentiality, integrity and availability without violating the service agreements of the IDPs.
- It should be able to provide file system interfaces to application programmers to enable the client applications outsource a file system to the middleware, which in turn appropriately processes the data and stores it at the IDPs. This interface is
provided since application developers are already used to a file system like interface for data storage.

- Importantly, the client application should not be aware of the IDP where the data is stored. As far as it is concerned, the data is stored with the middleware. Middleware is responsible for storing and fetching the data with the IDPs. IDPs assume that they are interacting with a client application. The clients configure the middleware and state their IDP preferences for data storage.

### 3.3 Proposed Middleware Architecture

The proposed middleware shown in figure 3.1 is designed to enhance the security by enabling 2-step authentication, verification of digital signature and integrity check of data. This enhancement is achieved by applying confidentiality module (figure 3.4) along with the Integrity module (figure 3.3) to the iDataGuard architecture described in section 2.11.1. Moreover, Classification module is used to ensure data availability.

![Figure 3.1: Proposed Middleware architecture for outsourcing data securely](image)

Figure 3.1: Proposed Middleware architecture for outsourcing data securely
Chapter 3 – Proposed Middleware

3.4 NON-FUNCTIONAL REQUIREMENTS OF THE PROPOSED MIDDLEWARE

3.4.1 Abstract Service Model to Support Heterogeneity

The iDataGuard Middleware approach uses the recommended model to address the heterogeneity of different interfaces provided by different IDPs and simplifies the effort for application development in enforcing security on outsourced data. The model comprises the following three sub-models to heterogeneity:

- **Abstract Data Model:** IDPs store and fetch objects. Objects are atomic units of data. For instance, a pdf document, or an image can be an object. Every object stored in IDP has a unique id and a set of attributes such as name, content, metadata. The content part of the object represents object’s content. The metadata part of the object represents information about the object. The name part is a special type of metadata and its use will be described in the next section. All files outsourced by the client application are translated to an object before it is stored in IDP.

- **Abstract Operation Model:** All IDPs provide the following three basic operations on any objects that are outsourced:
  - **Store_object():** Given an object, this operation allows IDP to store the object including all its attributes.
  - **Fetch_object():** Given the id of an object, this operation allows IDP to retrieve the required object. Fetch_object operation is an overloaded operation, the middleware can use the other overloaded parameter to fetch the metadata if required.
  - **Delete_object():** Given the id of an object, this operation allows IDP to delete the object at the server.
- **Update_object**: Given the id of an object and new content of an object, this operation allows IDPs to replace the old content of an object with the new content.

- **File Conversion to Abstract Data Model**: The proposed Middleware treats every file and directory as an object. In this context, when client application outsources a file to IDP through the proposed Middleware, it creates an object O whose content (O.content) is the content of the file, metadata (O.metadata) is information about the file such as last modified date, file size, etc. and object name (O.name) is the name of the file.

  For a directory, the object’s content (O.content) is set to null. The object’s name (O.name) is the name of the directory. The metadata of the directory object will include size of the directory, directory contents, etc. Additionally, the directory object maintains a child_references attribute in the metadata. Child_references contains a list of pointers to the immediate children of the directory.

### 3.4.2 MAC Module to Support Integrity

This section details how the MAC model which is added to the iDataGuard middleware ensures to preserve integrity of the outsourced data. That is, it should detect any data tampering attempts at the IDP. To achieve this, the hash function discussed in Chapter 2 is employed. The MAC is computed using hash function for each object before storing on the server. When the object is retrieved from the server, its MAC is also returned. The client calculates MAC again and compares it to the original MAC. If they are equivalent, then no tampering has occurred.
The MAC of an object is calculated as follows []:

\[ MAC (O.id \parallel O.Name \parallel O.Content \parallel O.metadata \parallel Version) \]

The version number is used to determine the freshness of the object. Every time when the object is updated, the version number is increased and the MAC calculated again. One possibility is to use the last modified date of the object as the version number. Such a date could be stored in the object’s metadata. Whenever the object is accessed, the object is retrieved from the server and the MAC is calculated again. The proposed Middleware uses this model to confirm the version number and MAC to determine if any tampering as taken place.

### 3.4.3 Data Encryption Model to Support Confidentiality

This section describes how the proposed Middleware uses the data encryption model to ensure confidentiality of user’s outsourced data. The id of the object does not reveal any information about the object at the server side and hence it is left untouched. The object’s id is used to fetch it from the server. Content, name and metadata part of an object are encrypted using the object’s encryption key (OEK).

- **Key Generation**: the key required to encrypt/decrypt the object is generated on the fly depending on the metadata that is attached to the object. Using the metadata, this component generates an object encryption key (OEK) for every object that is outsourced to IDP.

The key derivation function (KDF) of the password based encryption specification PKCS #5 [8] is used to generate OEKs. The KDF function calculates keys from passwords in the following manner:
Key = KDF(Password, Salt, Iteration)

The Salt is a random string to prevent an attacker from simply recalculating keys for the most common passwords. The KDF function internally utilizes a hash function that computes the final key. To deter an attacker from launching a dictionary attack, the hash function is applied repeatedly on the output Iteration times. This ensures that for every attempt in a dictionary attack, the adversary has to spend a significant amount of time.

The OEK Ki for each file object is calculated as follows:

\[ Ki = KDF(FileName||MasterPassword, Salt, Iteration) \]

The password parameter in the KDF function is the concatenation of the filename and the master password. Salt is a large random string that is generated the first time the object is created and is stored in plaintext along with the encrypted object. The filename does not include the full path name to permit a file objects to be moved between directories without changing its OEK. The iteration count is set to 10000, the recommended number.

The primary reason that we generate a key for every individual data object is to prevent cryptanalysis attacks, whose effectiveness increases with the amount of ciphertext available that is encrypted with the same key. Another approach is to generate a random key for each object and encrypt the object with that key. The random key could then be encrypted with the key derived from the password. We chose to generate the key since the KDF function is inexpensive compared to retrieving a key along with each object from the server. This reduces network bandwidth requirements, especially for small objects when the cost of retrieving the key would dominate.
• **Master Password Management:** It is of paramount importance that a user does not reveal his/her master password to anyone. The loss of a master password could lead to disastrous effects, since now the adversary can have complete control over the user’s file system. Passwords are prone to be lost or stolen. Therefore, there is a requirement to build mechanisms that change the master password as follows:

**Changing the master password:** Changing the master password will have its effect on the generation of keys. Hence, changing the master password could potentially be a very expensive operation, since it will require all the file(s) structure to be decrypted with the old keys and encrypted again with the new keys. Fortunately, we can do this expensive operation lazily. This component keeps track of both the new master password and the old master password in a configuration file that is encrypted with a key that is generated by the new master password. This component will continue to decrypt the files that are fetched from the server with the keys generated from the old password. When an update is made to the file, then component will encrypt the updated file with the key that is generated with the new password. The component will then set a flag in file structure to indicate that in future, it needs to use the key generated with the new master password. After a while, all the files will be encrypted with keys from the new master password.

Lazily changing the master passwords is typically done to periodically update the master password, a recommended practice. In a situation where the master password is compromised, the user could request the change of keys immediately. This is an expensive operation, the user has no choice but to wait till all the files are fetched, decrypted with the old key and encrypted with the new key. If the user decides to change the master password more than once, the component will again force the change of all the keys.
3.4.4 Data Classification Model to Support Availability

An approach is introduced to store the data in different sections in the cloud based on three cryptographic parameters (Confidentiality, Availability and Integrity). This approach is added to the iDataGuard middleware to ensure data availability. The value of confidentiality (C) is based on the level of privacy needed for data processing, the value of integrity (I) is based on data accuracy, information reliability and protection from unauthorized data modifications and the value of availability (A) is based on the frequency of data accessing and instant availability when requested.

Algorithm used in Classification Stage

1) Input: Data, protection part, D[ ] array of n integer size where D[ ] array consists of C, I, A, SR of n integer size.
2) Output: Categorized data for corresponding section.
3) For x = 1 to n
   a. C[x] → Value of Confidentiality
   b. I[x] → Value of Integrity
   c. A[x] → Value of Availability
   d. Calculate SR[x] = (C[x] + (1/A[x]) * 10 + I[x]) /2  [4]
      /* SR[x] → Value of Sensitivity Rating */
      /* Security and confidentiality are directly proportional to integrity where availability is inversely proportional to security */
4) For y = 1 to 10
   For x = 1 to n
      If SR[x] = = 1 || 2 || 3 then /* || represents OR operation */
      S[x] → 3 /* “3” selected to D[x]th data */
      If SR[x] = = 4 || 5 || 6 then

In the algorithm mentioned above, the main job of an owner is to categorize the data based on $C$, $I$ and $A$. Then, the proposed formula is applied to calculate the value of $SR$. The value of $SR$ is used to allocate the data to one of the three sections in cloud; $S3$ for public section, $S2$ for private section or $S1$ for owner’s limited access section.

For example, let’s have the values of $C$, $A$ and $I$ be 8, 20, 10 respectively. The value of $SR$ will be $SR= (8 + (1/20)*10 + 10)/2 = 9$ for this the data will be in section 1. Another example is when the values of $C$, $A$ and $I$ are all 3 then $SR = (3 + (1/3)*10 + 3)/2 = 4$ so the data will be saved in section 2.

Data classification depends on the type of data to be stored. For some sectors, classification might not be applicable as all data are confidential and should be stored in the owner’s limited access section of the cloud storage.

### 3.4.5 Keyword Search over Encrypted Data

It is complicated to search over encrypted data stored in the cloud, for that, an index is built to facilitate the search over encrypted data while it is being retrieved. Figure 3.2 shows the usage of index builder, encryption and how data is stored in the cloud. An index offers a quick way to retrieve files. One way to build up an index is by using a keyword of interest. This index will be encrypted to provide more security. An index basically contains a list of keywords and each keyword contains a list of pointers that points to the document where...
the keyword appears. Keywords are considered as words of interest that a user may want
to search for later.

3.5 PROCESS MODEL FOR THE PROPOSED APPROACH

This section illustrates the process model for the proposed approach that has been
structured to provide complete security to the data throughout the entire process of cloud
computing, be it in cloud or in transit. Thus, multiple mechanisms and available techniques
are applied to shield the critical information from unauthorized parties. The proposed
approach consists of two process. First process deals with storing data securely into the
cloud. Second process deals with the retrieval of data from cloud and showing the
generation of requests for data access, double authentication, verification of digital
signature and integrity, thereby providing authorized user with data on passing all security
mechanisms.

3.5.1 Data Storing Process

The data storing stage deals with techniques to store and secure the data and transfer the
encrypted data to the cloud (figure 3.2). Thus this process is triggered based on client
application initiation to outsource data to IDP. Firstly, this process interacts with data
translator component to transform the data file to be outsourced to abstract data type. Next
the classification component is invoked to determine the sensitivity rate of the data based
on user requirement. Accordingly, the cloud section where the data to be outsourced is
determined. Then the confidentiality of the data is focused by calling the crypto module to
encrypt the data object. Subsequently, to facilitate the search over the outsourced data,
index is computed using Index builder component based on the keyword interest. Finally, the proposed middleware uploads the encrypted data along with the MAC to the IDP.

**Figure 3.2:** Data storing process

**Figure 3.3:** MAC Module [4]

### 3.5.2 Data Retrieval Process

Data retrieval must be supported with secure methods. At the beginning, data retrieval involves the user to be registered with the organization to have a username and password which will be stored in the cloud directory. A request with the user username and password is sent to the cloud whenever the user requires access to the data (see figure 3.4 the activity...
diagram for the same process and figure 3.5 which describes confidentiality module for accessing request and authentication process). After that, the cloud checks the request and categorize it according to the section it belongs to. If it belongs to (section 3) which is the public section, no authentication is required and the user is granted access to the data. If the requested data belongs to the private section (section 2) or limited access section (section 1) authentication is required and the cloud checks for the username and password provided by the user in the cloud directory. When the username matches a record in the directory, the cloud forwards the username to the owner for authentication.

To authenticate a user, the user should first sends his password to the owner and if it is correct, the user will be asked to provide the accurate answer to the security question. Then the user will be authenticated. Data owner will send the user identity and the digital signature to the cloud to keep a record of the user accessing the data. After that, the user sends a request for the data to the owner and the owner sends the digital signature, requested data keyword and a master key to decrypt the data. Figure 3.6 shows the data retrieval process.

![Activity Diagram for Confidentiality Module & Authentication Process](image-url)
Figure 3.5: Confidentiality Module & Authentication Process

Figure 3.6: Data Retrieval Process
3.6 **CONCLUSION**

The outsourced file system from the application is first translated into an abstract data model by the middleware. Data translator (DT) is in charge of that operation. The file system is mapped to a set of objects. The objects are then cryptographically secured by the Crypto module. The crypto module requires a secret provided by the client called the master-password to generate the cryptographic keys that are used to secure the objects. The objects are stored and fetched by the middleware by utilizing service adapters. Service adapters are IDP specific modules. For every service the middleware supports, a service adapters needs to be written.

Service adapters utilize the interface (API) by the IDP to store and fetch objects. The index generator component generates the cryptographic index for all the text based files that are outsourced. The application can issue all file system operations to iDataGuard. iDataGuard translates all file system operations into equivalent operations on the abstract operation model. The translation is done at the Operation translation (OPT) component. The operations are first executed at the object level and their effects propagated to the IDPs via the service adapters.
Chapter 4 ~ Implementation and Results ~
Chapter 4: Implementation and Results

4.1 Introduction

Here in this chapter, we evaluate the middleware architecture proposed in Chapter 3 with regard to its performance and security. The first part presents the structure of client GUI to access the services of proposed Middleware. In the second part of the chapter, we evaluate the security of the system by discussing about whether the system is immune against the various possible attacks on the security of the system. At the end of this chapter, we evaluate the performance of different functionalities rendered by the proposed Middleware, namely storing data, retrieving data, and encryption/decryption.

4.2 Model Deployment Architecture

Figure 4.1 described the deployment architecture of the proposed model. The model consists of three parts:

- **The Client Application**: which is discussed in the next section 4.3.

- **The Middleware**: the middleware is an asp.net web service which is a component resides on a web server and provides information and services to other network applications using standard web protocols like HTTP and Simple Object Access Protocol.

- **The Service Provider**: the used cloud provider is CloudSim simulation which is discussed later in section 4.5
4.3 Structure of Client GUI

This section presents the overview of the structure of the GUI by showing a screenshot of the prototype. The GUI contains two windows to access the two main services of the proposed Middleware namely, File Outsourcing and File Retrieval. The file outsourcing window as shown in figure 4.2 comprises three frames to collect the following data from data owner:

- Data Classification: this part of the frame prompts user to provide value for confidentiality, availability and integrity
- Data Encryption: this part of the frame prompts user to enter the master-password
- File Details: this part of the frame prompts user to browse the file for outsourcing
Figure 4.2: Outsource Data Window

Figure 4.3: Data Retrieval Window
The data retrieval window contains two frames as shown in figure 4.3 to collect the following data from data owner:

- User details: this part of the frame prompts user to provide his ID and digital signature
- Keyword: this part of the frame prompts user to enter the search keywords

### 4.4 Simulation Environment

Simulation is a science and technique to make a model of processes or a real system. It is designed to evaluate and test strategies that aims to study and understand the behavior of the system or evaluating different strategies [61]. In cloud environment, implementing and evaluating all operations is very expensive, hence we are not able to achieve all aspects of advantages or disadvantages in real world. A Study of distributed, virtualized, and elastic resources can be carried out in a controlled manner with simulation to gain insights of the Application performance.

Simulation tools are alternatives to a real infrastructure and are used to study the system before deploying it to the real environment [62] [63]. There are variety of simulator tools for modelling and simulation of large scale Cloud computing environments. Generally these simulation tools are classified in two groups, the first group is command line or non-graphical tools. This group of simulators are based on java language, open source and event oriented. Hence, Attributes are defined as they interact with each other via event. An important feature of these tools is that they are open source tools and more flexible than other tools. The second group is graphical tools that provides simulation and modeling operation without the need of programming skills. Figure 4.4 shows some of the popular simulators which are used mainly by researchers. The most used tools in the graphical
group are CloudAnalyst and CloudReports while CloudSim from the non-graphical group is considered as the most used tool. Next section describes CloudSim.

### 4.4.1 CloudSim

CloudSim is a Toolkit developed by the GRIDS Laboratory at University of Melbourne. It provides system and behavioral modelling of Cloud Computing Components [64] [65]. As Architectural views composed of multilayers - which the CloudSim simulation layer provides – of support to modelling and simulation of virtualized Cloud-based data center environments including dedicated management interfaces for VMs, memory, storage, and bandwidth. Also it allows to model host, service brokers, scheduling and allocation policies of a large scaled Cloud platform. Hence CloudSim allows simulation of scenarios modelling IaaS, PaaS, and SaaS services [64] [65]. Between all of command line simulator tools of cloud computing, CloudSim is the best toolkit. This toolkit was chosen in this work for the following features:

- It runs in a virtual machine on the Amazon Web Services (AWS) cloud. Therefore, it allows users to launch, terminate and monitor virtual machines in the AWS cloud.
- Different configurations can be launched, depending on the requirements, and available machines on the cloud.
- Each CloudSim configuration maps to a constellation, which are collections of multiple virtual machines running together.
Figure 4.4: Simulation tools for Cloud Computing

### 4.5 Security Analysis

As discussed in Chapter 2, the available security solutions for the well-known cloud storage systems are server centric. There are many providers, who offer cryptography in their cloud storage systems, but the encryption and decryption processes are performed on the server. Moreover they do not support any kind of trustworthy data integrity mechanisms. As we have mentioned in Chapter 3, middleware with required all the security services can aid the client application to outsource data securely without the concern for heterogeneity of interfaced provided by different IDP, and therefore it gives more security to user’s data and at the same time reduces the burden of application programmer from heterogeneity of cloud computing. Since the data is encrypted with the most powerful encryption algorithm, it is practically impossible to decrypt it without the symmetric key. If a key length of 128 bits
or longer is used, a brute force attack would not be successful with the current computer technology.

Another action that a malicious user can perform is to modify the data without knowing the content of it, but because the system ensures integrity of data, the authorized users would know that the data has been modified by an intruder whenever they retrieve the data. Here the proposed approach uses MAC to check whether the data is updated by everyone. Thus the authorized users would always detect an unauthorized modification when he downloads the data. By using a MAC integrity service supported by the proposed Middleware, an intruder would not be able to update the stored data without the use of the correct private key. Nevertheless, whenever the modified data is uploaded, the middleware uses the corresponding version number and public key to verify the signature, and if the data is not signed with the proper private key, the verification fails, and thus the data is not updated.

To sum up, we can say that the attacks on the actual data without the presence of corresponding keys would not be successful. If we assume that the keys are kept secret and are not accessible in any way, then the confidentiality and integrity of data are guaranteed.

However other kinds of attacks that can be performed on network systems are also applicable in cloud storage systems, even with the presence of cryptographic access control. In the following we mention some of the well-known attacks that threaten almost all network systems.

### 4.5.1 DoS Attack Analysis to Ensure Data Availability

Denial-of-service (DoS) attack has always been a threat to the distributed systems. The main principle in DoS attack is to send a lot of requests to a service provider occupying
most of its resources, such that it cannot respond to legitimate traffic. A more advanced type of this attack is called distributed DoS (DDoS) attack, which uses many systems to attack a single system. In this case there would be more than one victim, namely the target, and all the other systems that are used for this purpose. The attack is mostly targeted towards the enterprise application servers, but as the use of cloud computing systems increases, this attack also threatens the cloud services. Because of the elasticity of the cloud computing systems, the attack mostly affects the users. If a communication between a user and the service is a victim of the attack, and as a result the service is not available for the user, then the cloud can just provide more resources to make the service available. In this case the user has to pay for both the resources he has used and for the resources that was used by the attack.

As other services in the cloud can be victims of this attack, the cloud storage system would not be an exception. An attacker in this case can transfer a lot of files to a server node using DDoS, which would cause the node to become busy, and as a result data transfer by legitimate users would not be possible, or become very slow on that node. In the case of data retrieval, if the cloud storage system supports replicated data, then there would not be a problem in accessing data, but the choice of replication would have a high cost for a user, because he will use much more space than the size of his data. In practice users would like to avoid high costs, so they would only require the space needed to store their data. In such a situation their data would be unavailable when the server node is attacked. However those servers, who support some kind of backup for the stored files as default, makes it possible for the user to access his data anyway, but if the attack is widely spread in the cloud storage system, then it can significantly decrease the data availability. All in all, the effect on availability of data depends on the extent of a DDoS attack, and the power of services provided by the cloud to avoid the attack.

In the case of storing data to the cloud, there will not be such a big problem for a user when a server node is attacked. As we know, the cloud computing systems are built in a way that
they provide services in a virtual manner. When a user buys a specific storage space in the cloud, he gets a virtual storage space, which means that his data are not necessarily stored to one server node. So whenever a server node is not available, which can be because of a DDoS attack, it will still be possible for a user to store his data, because the cloud storage facility would just provide another server node for the user, where he can store his data.

It is worth mentioning that users usually use applications as services provided by the cloud when storing and retrieving data, and if these services are targeted by DDoS attack, then the cloud computing system would provide more resources to make the services available. This leads to more cost for the user as we mentioned in the beginning of this section. So as a result, whenever DDoS attack occurs in the cloud, the user interactions in the attacked part would often be affected, which leads to typical problems like high cost, data unavailability, etc. This is handled in proposed approach based on the sensitivity rate which is computed using the user security requirement for data availability. The storage section of the cloud is decided based on the sensitivity rate of the user data.

4.5.2 Man-in-the-middle Attack

The previous section discussed about DDoS attack, which is mostly related to data availability in the cloud storage. It is not a threat against data confidentiality and integrity, and therefore the attack is not applicable on the cryptographic access control mechanism. Another type of attack, which is related to cryptographic systems, is man-in-the-middle attack. Next section discusses about whether or not this attack can be successful on our security services rendered by the proposed Middleware.

When storing and retrieving files are performed, there are two endpoints involved in the process, namely the client and the server. The client gives his public key to the server, which is used to verify the future updates to the file. Except the public key, no other keys
are provided, so the only possible malicious action that the server can do, is to allow
unauthorised modification of data. In such a situation the user would still know that the
data is modified when he retrieves it back from the server. The man-in-the-middle attack
is only applicable when two users want to exchange their symmetric and private keys with
each other in order to grant access permission to a file or a key ring. Here an intruder can
interfere in the communication in order to get the keys. The key exchange is performed
outside the whole system, so the server is not involved in it. Users can use a secure channel
that protects against man-in-the-middle attack to exchange keys, or a 100% secure way
would be to exchange the keys by meeting each other in person. The system does not
support symmetric and private key exchange, so it is the user’s responsibility to distribute
his keys in a secure way. Apart from that, man-in-the-middle attack is not applicable in the
system.

4.5.3 Traffic Analysis

Traffic analysis is an attack that is applicable in the systems, even with the presence of
cryptography, because one of the purposes of this attack is to watch how the sizes of files
are changed, and by that the attacker can estimate the number of updates that have been
performed on the files. By using traffic analysis one can also know which users have shared
their data with each other. This knowledge can be achieved by watching which users make
updates to a single file.

In this work, the middleware is concerned only with storage and retrieval of data. By only
looking at the stored file and downloading it, one cannot know from where the file is
uploaded, but by watching the traffic, an intruder may find the location of the client. It
enables the intruder to perform his attack on the client in order to access the private keys.
But in our approach the presence of middleware between client application and cloud
prevents the intruder to find the location of the client and makes the client to have
anonymity when interacting with cloud. Moreover users can install firewall on the client side, which also hinders unauthorized access.

### 4.6 Time Performance Analysis

#### 4.6.1 CloudSim Setup Configuration

Figure 4.5 shows the configuration parameter that was considered for our simulation study using CloudSim with respect to cloud providers and customers respectively.

Figure 4.5 (A) shows the configuration of the cloud service provider. The provider have two datacenters with ten hosts. Figure 4.5 (B) shows the customer configuration. The total number of customers is three and the number of hosts in each customer is seven.
4.6.2 Time Performance Analysis Setup

To analyze the time, two different configuration setups were used. The first setup was with single replication (figure 4.6 - 1) to verify the correctness of the system. Due to insufficient specification by the system for confidence level 95% in half width as shown in figure 4.7, a second setup was done with 50 replication (figure 4.6 - 2) to obtain a confidence results against the performance of the system as indicated in half with column in figure 4.9.

Figure 4.6: Time Performance Analysis for Setup 1 and 2
### Resource

#### Usage

<table>
<thead>
<tr>
<th>Instantaneous Utilization</th>
<th>Average</th>
<th>Half Width</th>
<th>Minimum Value</th>
<th>Maximum Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS 1</td>
<td>0.9688</td>
<td>(Insufficient)</td>
<td>0.00</td>
<td>1.0000</td>
</tr>
<tr>
<td>WS 2</td>
<td>0.9323</td>
<td>(Insufficient)</td>
<td>0.00</td>
<td>1.0000</td>
</tr>
<tr>
<td>WS 3</td>
<td>0.8595</td>
<td>(Insufficient)</td>
<td>0.00</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Number Busy</th>
<th>Average</th>
<th>Half Width</th>
<th>Minimum Value</th>
<th>Maximum Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS 1</td>
<td>0.9688</td>
<td>(Insufficient)</td>
<td>0.00</td>
<td>1.0000</td>
</tr>
<tr>
<td>WS 2</td>
<td>0.9323</td>
<td>(Insufficient)</td>
<td>0.00</td>
<td>1.0000</td>
</tr>
<tr>
<td>WS 3</td>
<td>0.8595</td>
<td>(Insufficient)</td>
<td>0.00</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Number Scheduled</th>
<th>Average</th>
<th>Half Width</th>
<th>Minimum Value</th>
<th>Maximum Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS 1</td>
<td>1.0000</td>
<td>(Insufficient)</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>WS 2</td>
<td>1.0000</td>
<td>(Insufficient)</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>WS 3</td>
<td>1.0000</td>
<td>(Insufficient)</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scheduled Utilization</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS 1</td>
<td>0.9688</td>
</tr>
<tr>
<td>WS 2</td>
<td>0.9323</td>
</tr>
<tr>
<td>WS 3</td>
<td>0.8595</td>
</tr>
</tbody>
</table>

Figure 4.7: System performance for run Setup shown figure 4.6 - 1

#### 4.6.3 Middleware Processing Time for Proposed Security Service

The proposed technique is implemented with different file size ranges from 100KB to 60MB using a simple file transfer between a customer and cloud provider in this case CloudSim. The experiments were conducted to analyze the performance between existing technique and the proposed technique.

- **File Upload Time**: It includes the time to encrypt the file as requested by the client Application. It is the time between the points when user requests the cloud system
to upload the file and the time when the tasks of data translation, data classification, encryption and generating key actually finishes and the encrypted file is actually stored in the cloud data storage. Figure 4.10 shows the time taken for file uploading for different file sizes.

- **File Search time**: It includes time to retrieve the q-gram and keyword entries based on the user query and then retrieved of file metadata. It is the time between the two points when the user makes a request to search with queries and actually receives the list of files with the requested keyword pattern. Figure 4.11 shows the time taken for file search process for different file sizes.

<table>
<thead>
<tr>
<th>Resource</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Instantaneous Utilization</strong></td>
<td>Average</td>
</tr>
<tr>
<td>WS 1</td>
<td>0.9568</td>
</tr>
<tr>
<td>WS 2</td>
<td>0.9352</td>
</tr>
<tr>
<td>WS 3</td>
<td>0.8313</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Number Busy</strong></th>
<th>Average</th>
<th>Half Width</th>
<th>Minimum Average</th>
<th>Maximum Average</th>
<th>Minimum Value</th>
<th>Maximum Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS 1</td>
<td>0.9568</td>
<td>0.01</td>
<td>0.8772</td>
<td>0.9968</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>WS 2</td>
<td>0.9352</td>
<td>0.01</td>
<td>0.8368</td>
<td>0.9942</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>WS 3</td>
<td>0.8313</td>
<td>0.01</td>
<td>0.7479</td>
<td>0.9214</td>
<td>0.00</td>
<td>1.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Number Scheduled</strong></th>
<th>Average</th>
<th>Half Width</th>
<th>Minimum Average</th>
<th>Maximum Average</th>
<th>Minimum Value</th>
<th>Maximum Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS 1</td>
<td>1.0000</td>
<td>0.00</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>WS 2</td>
<td>1.0000</td>
<td>0.00</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>WS 3</td>
<td>1.0000</td>
<td>0.00</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Scheduled Utilization</strong></th>
<th>Average</th>
<th>Half Width</th>
<th>Minimum Average</th>
<th>Maximum Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>WS 1</td>
<td>0.9568</td>
<td>0.01</td>
<td>0.8772</td>
<td>0.9968</td>
</tr>
<tr>
<td>WS 2</td>
<td>0.9352</td>
<td>0.01</td>
<td>0.8368</td>
<td>0.9942</td>
</tr>
<tr>
<td>WS 3</td>
<td>0.8313</td>
<td>0.01</td>
<td>0.7479</td>
<td>0.9214</td>
</tr>
</tbody>
</table>

Figure 4.8: Resource Usage for run Setup shown figure 4.6 - 2
### Table 4.9

<table>
<thead>
<tr>
<th>Time Type</th>
<th>Average</th>
<th>Half Width</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Minimum Value</th>
<th>Maximum Value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NVA Time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File download</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>File Upload</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Wait Time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File download</td>
<td>30.387</td>
<td>13.41</td>
<td>18.7786</td>
<td>83.2751</td>
<td>0.00</td>
<td>180.00</td>
</tr>
<tr>
<td>File Upload</td>
<td>51.3577</td>
<td>18.51</td>
<td>19.3241</td>
<td>87.9347</td>
<td>0.00</td>
<td>170.00</td>
</tr>
<tr>
<td>Transfer Time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File download</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>File Upload</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Other Time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File download</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>File Upload</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Total Time</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File download</td>
<td>45.7104</td>
<td>13.41</td>
<td>27.5379</td>
<td>92.0811</td>
<td>5.6488</td>
<td>197.00</td>
</tr>
<tr>
<td>File Upload</td>
<td>58.3951</td>
<td>18.48</td>
<td>26.3884</td>
<td>95.4884</td>
<td>6.0404</td>
<td>182.00</td>
</tr>
</tbody>
</table>

#### Figure 4.9
File Upload/Download time for Setup shown in figure 4.6.2

#### Figure 4.10
File Uploading Time against File Size
As it shows in figures 4.10 and 4.11:

1) The iDataGuard shows a nominal line as it only provide one security feature which is confidentiality
2) The Security Approach shows a high line as it provide complex security features
3) The proposed middleware shows a middle line as it provides the security features in the middleware itself not in the cloud.

4.6.4 Middleware Hosted on CloudSim Security Service

To evaluate the performance of proposed middleware, two experiment were conducted. First experiment client application consumes data storage services through the middleware hosted in CloudSim comparing with direct consuming the data storage service on the client side as shown in figure 4.12.
Figure 4.13 shows the overhead of consuming of the data storage service on the middleware (middleware series) versus directly consuming and combining the two services on the mobile client (client series). The y-axis is the number of executions of the data storage service (50 samples in total). The time interval between each sample request is 1 minutes, so they do not cause a heavy load on the middleware. The x-axis is the total processing time including network latency and parsing time. The average response time of the middleware security service is 753.48ms with a standard deviation of 99.5. The average response time of client side security service is 942.22ms with standard deviation of 97.7. Both of the two series have a lot of fluctuations which is mainly caused by network latency. The result shows executing the security service on the middleware is faster than executing it on the client, because the middleware has access to more bandwidth, network connections and processing power.

Figure 4.12: Resource Utilization from Customer perspective
Figure 4.13: Resource Utilization from Cloud Provider perspective
Chapter 5 ~ Conclusion and Future Work~
In this work as discussed in Chapter 1, a middleware solution for data confidentiality, integrity and availability in cloud storage systems is examined. The available solutions in the market are studied, whereupon the possibilities for a solution based on cryptography is analyzed, and as a result the cryptographic services through middleware mechanism is proposed. In this security solution, cryptography is used to ensure confidentiality and integrity of the stored data. The main quality in this solution is that the security operations are performed in the proposed Middleware, and thus the users do not need to trust the cloud servers. The only elements that make it possible to access the stored data are the corresponding keys, and thus file sharing between users can only happen by exchanging keys.

Chapter 2 discuss and compares the available security solutions for well-known cloud storage systems. This exploration shows that there is not any standard or a common agreement regarding the security solutions in the cloud. Various solutions are used by different providers, usually in a hybrid way. Most well-known cloud computing providers like Amazon, Google and Microsoft use security solutions at the server side. However there are some providers, such as Carbonite, Mozy, etc. that offers symmetric encryption at the client side, but what is common for most of the providers is that the presence of a trustworthy data integrity mechanism is missing. In contrast to the available security solutions, security services through middleware ensures confidentiality and integrity of data at the client side. On the basis of the analysis, a Middleware approach is designed to meet the expected cloud security requirements. Chapter 3 of this document describes the core components of the proposed middleware solution.
We have accomplished an evaluation of the proposed Middleware solution, which gave us an idea about whether or not the system is usable in real-world. The evaluation process can be divided in two main parts: security and performance evaluation of the whole system.

Regarding the security of cryptographic mechanism, we should mention that the only possible way to read or write the data is to have the corresponding keys, and the keys are only available on the authorized client’s machine. (It is of course assumed that the client’s machine is immune against unauthorized access.) On the other hand, the algorithms used for encryption and digital signature are fully tested and approved over many years, and as a result they are practically unbreakable with the current technology. We have to admit that the solution primarily guarantees data confidentiality and integrity.

There are many other threats against the security of distributed systems, and the cloud computing system is not an exception in this context. The well-known attacks are discussed in this document, which are DoS (Daniel-of-service) attack, man-in-the-middle attack and traffic analysis. Many of these kinds of attacks are mostly targeted towards the data availability and communication, but they do not threaten the cryptographic mechanism directly. However traffic analysis can be used to trace the client, and thus get access to the keys, but in the proposed solution the middleware enables clients to have anonymous interactions with the cloud storage and diminishes this threat.

The result of the test regarding the performance of cryptographic mechanism showed that actually this solution is on its way to be usable in practice. We compared the data storage and retrieval process through Middleware and through direct access to cloud service. However the implementation through Middleware was faster than direct access to cloud services leveraging the client computation from security requirement concerns. Thus the results indicated that the proposed Middleware approach can surely be used as a practical security solution for the cloud storage systems.
The proposed middleware solution provides a way to protect the data, check the integrity and authentication by following the best possible industry mechanisms. It introduces the division of data into different sections, Index builder, 128-bit SSL encryption, Message authenticate code and a double authentication of user by owner and other by cloud and verification of digital signature of the owner. It provides availability of data by surpassing many issues like data leakage, tampering of data and unauthorized access even from the cloud service provider. Proposed method achieves the availability, reliability and integrity of data traversing through owner to cloud and cloud to user. In addition to this, it utilizes an index based approach to allow keyword based search on encrypted data. Thereby making the proposed middleware approach practically feasible. More importantly, it provides data abstraction to reduce the burden of developing applications that leverage the storage infrastructures provided by internet based data storage providers. Thus the proposed solution is expected to be a useful tool for both application developers and clients.

However storing data to CloudSim is not fully consistent with what is expected, but it is a shortcoming in its implementation. Considering and applying a solution to this issue would be the future scope of this work.
Glossary

**REST Protocol** (REpresentational State Transfer) is a software architectural style and an approach to communications that is often used in the development of Web services.

**JSON-RPC Protocol** is a stateless, light-weight remote procedure call and it is a very simple protocol defining only a handful of data types and commands.

**MS Online Services** includes Exchange Online, SharePoint Online, Office Communications Online, Microsoft Forefront, and Microsoft Office Live Meeting

**Salesforce** is a global cloud computing company

**Engine Yard** is a privately held platform as a service company focused on Java, Ruby on Rails, PHP and Node.js deployment and management.

**CollabNet** is a developer of software development and application lifecycle management tools

**Terremark** is a subsidiary of Verizon Communications, is a provider of information technology services which include managed hosting, colocation, disaster recovery, data storage and cloud computing.

**Rackspace** is a managed cloud computing company helps design, build, and operate workloads across cloud servers or dedicated servers depends on the customer needs

**CapEx or Capital Expenditure** is expenditure creating future benefits. A capital expenditure is incurred when a business spends money either to buy fixed assets or to add to the value of an existing asset with a useful life that extends beyond the tax year.
**Opex or Operational Expenditure** refers to expenses incurred in the course of ordinary business, such as sales, general and administrative expenses.

**Elgamal Algorithm** is an algorithm invented and introduced by Taher Elgamal an Egyptian American cryptographer in 1985. This algorithm is based on the Diffie-Hellman key exchange which is based on one-way called discrete logarithm.
References


