AUTHOR’S DECLARATION

“I, Hassan Saad Al-Qahtani

declare that this thesis and the work presented in it are my own and has been generated by me as the result of my own original research.

A NOVEL APPROACH TO PROVIDING SECURE DATA STORAGE USING MULTI-CLOUD COMPUTING

I confirm that:

1. This work was done wholly or mainly while in candidature for a research degree at this University;

2. Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;

3. Where I have cited the published work of others, this is always clearly attributed;

4. Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;

5. I have acknowledged all main sources of help;

6. Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.”
ABSTRACT

The cloud computing is a new technology that has been presented in the market in recent years. However, it suffered several security breaches, and has some open issues, in terms of security. Due to that, the literature was investigated to highlight the cloud computing security issues, it was found that about 50% of cloud computing security issues are associated with data storage, applied cryptography, and applied authentication. On the other hand, Multi-cloud paradigms have been developed as efficient solutions in order to overcome some single cloud paradigm obstacles and limitations, and enhance the efficiency of ICT cloud-based solutions. Developing an approach that is stable and capable of delivering a very high level of security and availability cannot be achieved by relying on a higher layer of the delivered system (the software), the lower layer (the infrastructure) must be involved in order to achieve that level of service. This study aims to improve the security of the delivered cloud storage service via multiple-cloud computing and to develop an approach for providing a secure data storage system that could be installed, configured, and easily consumed through the appropriate multiple-cloud model. The developed approach is supposed to maintain the confidentiality, integrity, and authenticity of the protected data; besides that, it will support disaster recovery and auditing for the system. This study aims to reduce the complexity and required knowledge levels associated with consuming a multiple-cloud computing paradigm and enhance the flexibility. In order to validate and verify the developed approach, a prototype was developed and tested, the testing phase consists of three core experiments, the outcomes of these three experiments were analysed, presented, and discussed. From the collected feedback, we could conclude that the developed prototype performance is as expected and developed prototype has been validated and verified.
ACKNOWLEDGMENTS

I would like to express my deep and sincere gratitude to almighty God for giving me the energy, time, and strength to complete this degree. I would also like to thank my supervisory team, for there encouragement and great guidance and continuous support during my study. Their expertise and supervision have been of a great value to me.

Last but not least, I would like to thank my family for supporting me spiritually throughout the writing of this thesis and in my life in general.
PUBLICATIONS TO DATE


TABLE OF CONTENTS

AUTHOR’S DECLARATION ........................................................................................................ I

ABSTRACT ............................................................................................................................. II

ACKNOWLEDGMENTS .......................................................................................................... III

PUBLICATIONS TO DATE ................................................................................................. IV

TABLE OF CONTENTS ......................................................................................................... V

LIST OF TABLES ................................................................................................................ X

LIST OF FIGURES .............................................................................................................. XI

GLOSSARY ........................................................................................................................ X IV

CHAPTER I. INTRODUCTION .............................................................................................. 1

1.1 Research Rationale and Motivation............................................................................. 2

1.2 Aim and Objectives ..................................................................................................... 4

1.3 Research Questions ..................................................................................................... 5

1.4 Main Contributions to The Body of Knowledge ......................................................... 6

1.5 Research Methodology ............................................................................................... 7

1.5.1 Data Collection ....................................................................................................... 7

1.5.2 Primary Data .......................................................................................................... 10

1.5.3 Secondary Data ..................................................................................................... 10

1.6 Structure of the Thesis ............................................................................................... 11

1.7 Summary ..................................................................................................................... 13
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAPTER II. LITERATURE REVIEW</td>
<td></td>
</tr>
<tr>
<td>2.1 Cloud Computing</td>
<td>15</td>
</tr>
<tr>
<td>2.1 Multiple-Clouds Computing Paradigms</td>
<td>19</td>
</tr>
<tr>
<td>2.3.1 Classification Criteria</td>
<td>20</td>
</tr>
<tr>
<td>2.3.2 Multiple-clouds paradigms</td>
<td>22</td>
</tr>
<tr>
<td>2.1 Shifting to The Cloud</td>
<td>30</td>
</tr>
<tr>
<td>2.1 Cloud Computing Security Issues Systematic Literature Review</td>
<td>32</td>
</tr>
<tr>
<td>2.5.1 Review Strategy</td>
<td>34</td>
</tr>
<tr>
<td>2.5.2 Selection Procedure</td>
<td>35</td>
</tr>
<tr>
<td>2.5.3 Cloud Security Challenges and issues</td>
<td>37</td>
</tr>
<tr>
<td>2.5.4 Systematic Literature Review Findings</td>
<td>58</td>
</tr>
<tr>
<td>2.6 Multiple-Clouds Security Approaches</td>
<td>60</td>
</tr>
<tr>
<td>2.6.1 Multiple-Clouds as a Solution</td>
<td>60</td>
</tr>
<tr>
<td>2.6.2 State of the Art Multiple-Clouds Security Approaches</td>
<td>61</td>
</tr>
<tr>
<td>2.6.3 Critical Literature Review Findings</td>
<td>73</td>
</tr>
<tr>
<td>2.7 Summary</td>
<td>78</td>
</tr>
<tr>
<td>CHAPTER III. DEVELOPED APPROACH</td>
<td>80</td>
</tr>
<tr>
<td>3.1 Research Gap</td>
<td>81</td>
</tr>
<tr>
<td>3.2 Why This Approach</td>
<td>85</td>
</tr>
<tr>
<td>3.3 New Approach Requirements</td>
<td>91</td>
</tr>
<tr>
<td>3.4 Approach Concept</td>
<td>93</td>
</tr>
<tr>
<td>3.5 Approach Design</td>
<td>96</td>
</tr>
</tbody>
</table>
4.2 Prototype Requirements ................................................................. 132
4.3 Developed Android Application...................................................... 136
  4.3.1 The involved functions ............................................................... 136
  4.3.2 Prototype Architecture ............................................................. 138
  4.3.3 Prototype tasks and GUIs ........................................................... 140
4.4 Summary ..................................................................................... 155

CHAPTER V. RESULTS AND DISCUSSIONS ............................................. 156
5.1 Testing ....................................................................................... 157
5.2 Testing Environment ...................................................................... 157
5.3 Experiments ................................................................................ 159
  5.1.1 Tasks Order .............................................................................. 159
  5.1.2 Encryption ................................................................................ 163
  5.1.3 Performance ............................................................................. 165
  5.1.4 User Experience ....................................................................... 167
5.4 Discussion .................................................................................. 169
5.5 Challenges and Raised Issues .......................................................... 186
5.6 Summary ................................................................................... 188

CHAPTER VI. CONCLUSION AND FUTURE WORK ................................. 189
6.1 Overview .................................................................................... 189
6.2 Addressing Study Objectives .......................................................... 193
6.3 Research Implications .................................................................. 197
LIST OF TABLES

Table 2 - 1 Multiple-clouds agreement, collaboration, and migration metrics (Ferrer et al., 2012; Petcu et al., 2011; Kecskemeti et al., 2012) ................................................................. 21
Table 2 - 2 Multiple-cloud computing paradigms comparison .................................................. 29
Table 2 - 3 Publication assessment criteria .............................................................................. 36
Table 2 - 4 Multiple cloud security approaches comparison .................................................. 77

Table 3 - 1 Existing approaches' limitations ............................................................................. 84
Table 3 - 2 Cryptography parameters values ........................................................................... 115
Table 3 - 3 Threats' scores ...................................................................................................... 125

Table 4 - 1 Smartphone OS market share 2016 - 2018(IDC, 2018) ........................................ 133
Table 4 - 2 Vendor market share 2017 & 2018 (Gartner, 2018) ............................................. 133
Table 4 - 3 Development machine configuration ..................................................................... 135

Table 5 - 1 Task order scenarios ............................................................................................... 159
Table 5 - 2 Testing machines, and files specifications (Tasks order experiment) ............... 160
Table 5 - 3 Task order experiment outcomes ......................................................................... 162
Table 5 - 4 Testing machines, and files specifications (encryption) ........................................ 163
Table 5 - 5 Encryption experiment outcomes ......................................................................... 165
Table 5 - 6 Testing machine, and files specifications (Performance) ..................................... 166
Table 5 - 7 Performance experiment outcomes ...................................................................... 166
Table 5 - 8 Participants information (age, gender, year of experience, and qualifications) .. 168
Table 5 - 9 User experience outcomes .................................................................................... 168
Table 5 - 10 Existing approaches and developed approach comparison .............................. 185
LIST OF FIGURES

Figure 2 - 1 Cloud computing stack (CSA, 2017a) ................................................................. 16
Figure 2 - 2 Cloud computing services flexibility, cost, targeted consumers, and managed layers .................................................................................................................... 18
Figure 2 - 3 Multiple-cloud computing motivation (Toosi, Calheiros and Buyya, 2014) ........... 19
Figure 2 - 4 Multiple-cloud paradigms ....................................................................................... 22
Figure 2 - 5 Clouds-of-clouds (intra-cloud) (Chen, Nepal & Ren, 2011) ............................... 23
Figure 2 - 6 Bursted (hybrid) cloud (Ferrer et al., 2012).......................................................... 24
Figure 2 - 7 Centralized federated-cloud (Grozev & Buyya, 2014) ........................................... 25
Figure 2 - 8 Peer-to-Peer federated-cloud (Elmroth et al., 2011) ............................................. 26
Figure 2 - 9 Multi-cloud services (Grozev & Buyya, 2014) ....................................................... 28
Figure 2 - 10 Multi-cloud libraries (Grozev & Buyya, 2014) .................................................... 29
Figure 2 - 11 S.L.R guidelines steps ............................................................................................ 34
Figure 2 - 12 Number of selected publications for each phase .................................................. 35
Figure 2 - 13 Publications number based on the year and the database ................................... 37
Figure 2 - 14 Cloud computing security categories ................................................................. 38
Figure 2 - 15 Identified cloud computing security issues ........................................................... 40
Figure 2 - 16 Identified cloud computing security issues taxonomy ......................................... 59

Figure 3 - 1 New approach requirements .................................................................................... 92
Figure 3 - 2 The proposed framework workflow ....................................................................... 99
Figure 3 - 3 Distribution & recovery procedures ....................................................................... 100
Figure 3 - 4 Computer system architecture and API location (Smith, Nair, 2005) .................. 107
Figure 3 - 5 Proposed approach architecture design ................................................................. 108
Figure 3 - 6 Preference module ................................................................................................. 109
Figure 3 - 7 Data management module ........................................... 110
Figure 3 - 8 Data security module ...................................................... 111
Figure 3 - 9 The chaos algorithm (Lian, Sun and Wang, 2005) .............. 114
Figure 3 - 10 Encryption procedure .................................................. 116
Figure 3 - 11 Approach trust domains ............................................. 117
Figure 3 - 12 Approach DFD and trust boundaries ............................. 118
Figure 3 - 13 Approach assets and associated threats ........................ 119

Figure 4 - 1 Smartphone OS market share 2016 – 2018 (IDC, 2018) .......... 133
Figure 4 - 2 Developed prototype architecture design .......................... 139
Figure 4 - 3 Main page activity diagram .......................................... 141
Figure 4 - 4 Cloud service permissions, application ID, and application secret key .... 142
Figure 4 - 5 Dynamic passcode activity diagram ................................. 145
Figure 4 - 6 File selection activity diagram ....................................... 147
Figure 4 - 7 Upload and retrieve activity diagrams ............................... 152

Figure 5 - 1 Consumed time for each configuration (Encryption) – Machine 1 ........ 170
Figure 5 - 2 Consumed time for each configuration (Encryption) – Machine 2 ........ 170
Figure 5 - 3 Consumed time for each configuration (Decryption) – Machine 1 ........ 171
Figure 5 - 4 Consumed time for each configuration (Decryption) – Machine 2 ........ 172
Figure 5 - 5 Encryption time comparison ........................................... 173
Figure 5 - 6 Encryption CPU consumption comparison ........................ 173
Figure 5 - 7 Decryption time comparison .......................................... 175
Figure 5 - 8 Decryption CPU consumption comparison ......................... 175
Figure 5 - 9 Distribution time comparison ........................................... 176
Figure 5 - 10 Distribution CPU consumption comparison ........................ 177
Figure 5 - 11 Distribution RAM consumption comparison ........................ 177
Figure 5 - 12 Retrieval time comparison ................................................................. 178
Figure 5 - 13 Retrieval CPU consumption comparison ........................................... 179
Figure 5 - 14 Retrieval RAM consumption comparison .......................................... 179
Figure 5 - 15 Statistical Results for User Experience ................................................ 181
Figure 5 - 16 User Experience categorisation ......................................................... 181
# GLOSSARY

<table>
<thead>
<tr>
<th>Notation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3DES</td>
<td>Triple DES</td>
</tr>
<tr>
<td>AES</td>
<td>Advanced Encryption Standard</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>AVD</td>
<td>Android Virtual Device</td>
</tr>
<tr>
<td>BFT</td>
<td>Byzantine Fault Tolerance</td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit</td>
</tr>
<tr>
<td>CSA</td>
<td>Cloud Security Alliance</td>
</tr>
<tr>
<td>DDR3</td>
<td>Double Data Rate Type 3</td>
</tr>
<tr>
<td>DES</td>
<td>Data Encryption Standard</td>
</tr>
<tr>
<td>DHCP</td>
<td>Dynamic Host Configuration Protocol</td>
</tr>
<tr>
<td>DNS</td>
<td>Domain Name System</td>
</tr>
<tr>
<td>DoS</td>
<td>Denial of Service</td>
</tr>
<tr>
<td>ENISA</td>
<td>European Union Agency for Network and Information Security</td>
</tr>
<tr>
<td>FTP</td>
<td>File Transfer Protocol</td>
</tr>
<tr>
<td>GB</td>
<td>Gigabytes</td>
</tr>
<tr>
<td>GCAGR</td>
<td>Global Compound Annual Growth Rate</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HAIL</td>
<td>High Availability and Integrity Layer</td>
</tr>
<tr>
<td>HDD</td>
<td>Hard Disk Drive</td>
</tr>
<tr>
<td>HTML</td>
<td>Hypertext Markup Language</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>HTTPS</td>
<td>Hypertext Transfer Protocol Secure</td>
</tr>
<tr>
<td>IaaS</td>
<td>Infrastructure as a Service</td>
</tr>
<tr>
<td>IAM</td>
<td>Identity and Access Management</td>
</tr>
<tr>
<td>IC Storage</td>
<td>Intercloud Storage</td>
</tr>
<tr>
<td>ICT</td>
<td>Information and Communication Technology</td>
</tr>
<tr>
<td>Acronym</td>
<td>Full Form</td>
</tr>
<tr>
<td>---------</td>
<td>-----------</td>
</tr>
<tr>
<td>IDC</td>
<td>International Data Corporation</td>
</tr>
<tr>
<td>IDE</td>
<td>Integrated Development Environment</td>
</tr>
<tr>
<td>IEEE</td>
<td>Institute of Electrical and Electronics Engineers</td>
</tr>
<tr>
<td>IOS</td>
<td>International Organization for Standardization</td>
</tr>
<tr>
<td>iOS</td>
<td>iPhone Operation System</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>IP-ECC</td>
<td>Integrity Protected Error Correcting Code</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>ITIL</td>
<td>Information Technology Infrastructure Library</td>
</tr>
<tr>
<td>MIT</td>
<td>Massachusetts Institute of Technology</td>
</tr>
<tr>
<td>MPLS</td>
<td>Multi-Protocol Label Switching</td>
</tr>
<tr>
<td>NIST</td>
<td>National Institute of Standards and Technology</td>
</tr>
<tr>
<td>PaaS</td>
<td>Platform as a Service</td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computers</td>
</tr>
<tr>
<td>QOS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RACS</td>
<td>Redundant Array of Cloud Storage</td>
</tr>
<tr>
<td>RAID</td>
<td>Redundant Arrays of Inexpensive Disks</td>
</tr>
<tr>
<td>RAM</td>
<td>Random Access Memory</td>
</tr>
<tr>
<td>RPO</td>
<td>Recovery Point Objective</td>
</tr>
<tr>
<td>RSA</td>
<td>Rivest-Shamir-Adleman</td>
</tr>
<tr>
<td>RTO</td>
<td>Recovery Time Objective</td>
</tr>
<tr>
<td>SaaS</td>
<td>Software as a Service</td>
</tr>
<tr>
<td>SAN</td>
<td>Storage Area Network</td>
</tr>
<tr>
<td>SDK</td>
<td>Software Developing Kit</td>
</tr>
<tr>
<td>SDLC</td>
<td>Systems Development Life Cycle</td>
</tr>
<tr>
<td>SLA</td>
<td>Service Level Agreements</td>
</tr>
<tr>
<td>SOA</td>
<td>Service Oriented Architecture</td>
</tr>
<tr>
<td>SOAP</td>
<td>Simple Object Access Protocol</td>
</tr>
<tr>
<td>SQL</td>
<td>Structured Query Language</td>
</tr>
<tr>
<td>SSD</td>
<td>Solid State Drive</td>
</tr>
<tr>
<td><strong>Tahoe-LAFS</strong></td>
<td>Tahoe-Least-Authority File System</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------------------------</td>
</tr>
<tr>
<td><strong>TB</strong></td>
<td>Terabyte</td>
</tr>
<tr>
<td><strong>TCP</strong></td>
<td>Transmission Control Protocol</td>
</tr>
<tr>
<td><strong>TTP</strong></td>
<td>Trusted Third Party</td>
</tr>
<tr>
<td><strong>UDP</strong></td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td><strong>UML</strong></td>
<td>Unified Modelling Language</td>
</tr>
<tr>
<td><strong>VDC</strong></td>
<td>Virtual Device Contexts</td>
</tr>
<tr>
<td><strong>VLAN</strong></td>
<td>Virtual Local Area Network</td>
</tr>
<tr>
<td><strong>VM</strong></td>
<td>Virtual Machine</td>
</tr>
<tr>
<td><strong>VMM</strong></td>
<td>Virtual Machine Manager</td>
</tr>
<tr>
<td><strong>VPN</strong></td>
<td>Virtual Private Network</td>
</tr>
<tr>
<td><strong>VRF</strong></td>
<td>Virtual Routing and Forwarding</td>
</tr>
<tr>
<td><strong>WS</strong></td>
<td>Web Services</td>
</tr>
<tr>
<td><strong>XML</strong></td>
<td>Extensible Markup Language</td>
</tr>
</tbody>
</table>
Chapter I. INTRODUCTION

This chapter presents an introduction to the study, which provides an overview of the research. The chapter starts by explaining the motivation and rationale for the research project. After that, the aim and objectives of this study are clearly and accurately defined to show the direction of this study.

Next, the research questions are presented and the main contribution this study will make to the body of knowledge is highlighted, which will help to illustrate the scope and the boundaries for the study. After that, the selected approach, data collection, research design, and research instruments are presented. Finally, the organisation of the thesis chapters and the contents for each one are described.
According to Jansen and Grance (2011) and King and Raja (2012), using the cloud to store data is dangerous due to four main factors: 1) communication via the internet; 2) external auditing and monitoring; 3) the complexity of the cloud system; and 4) sharing the same resources with other customers. Recent security breach on the cloud storage services exposed sensitive information to the public; for instance, the security breach that occurred to iCloud service and led to publication of personal pictures, and the FedEx breach that exposed more than 100,000 customers’ scanned IDs, which were stored in Amazon (Kelion, 2014; Santillan, 2014; Chang, 2016; Bertrand, 2017; O’Sullivan, 2017; Scipioni, 2018; Whittaker, 2018). These incidents drew attention to: 1) the applied authentication techniques; 2) the efficiency and effectiveness of the applied encryption method, and 3) the security of cloud storage services (Samson, 2012; Kelion, 2014).

Gangula, Ansari and Gondhalekar (2013) argue that cloud storage suffers from the following issues: i) un-authorised access by third parties, ii) an absence of control mechanisms for users, iii) legal uncertainty, iv) lack of regulations and v) cross-border data. These cloud storage security breaches have led to a number of questions, including 1) how safe and secure the cloud is for storing data, 2) how the integrity and confidentiality of data is secured and 3) whether we can trust service providers.

These security breaches raised a number of issues, which are related to protected data in terms of data location, encryption, authentication, as well as integrity. For sensitive applications, storing the data in one place might not be a good choice regardless of whether the data is encrypted or not; distributing the protected file between several locations will enhance the security, and reduce the probability of unauthorised access, as the attacker needs to access all locations, in order to get the file pieces. In addition, the exchange of data must be encrypted before being sent, in order to protect it against any un-authorised read...
attempt, if captured while being transmitted. Moreover, the data integrity and confidentiality must be maintained and enhanced by reducing the involved parties (data owner and service provider); no other party should be able to access or read the data. In addition, the data owner, and the data owner only, must know the encryption key and data location. There are a number of approaches that have been developed to improve data security over cloud storage and specifically over multiple-cloud, which might help to overcome some of the previous points. However, these approaches are not appropriate for individuals in terms of the needed resources, capabilities, and operational costs. Such problems show the opportunity of developing an approach that utilises multiple-cloud computing, effective encryption algorithm, and the needed techniques in order to provide a solution capable of solving the defined issues. The developed approach should be able to preserve data security for individuals.
1.2 **Aim and Objectives**

This research aims to investigate existing approaches to improve the security of the delivered cloud storage service via multiple-cloud computing, and to propose an approach for providing a secure data storage system that could be installed, configured, and easily consumed by individuals through the appropriate multiple-cloud model. The proposed approach maintains the confidentiality, integrity, and authenticity of the protected data; besides that, it supports disaster recovery and auditing for the system. The developed approach aims to reduce the technical experience needed, the complexity, and to improve the flexibility of the delivered service. To achieve the study aims, a set of objectives have been formulated and divided into two stages. The 1\textsuperscript{st} stage is concerned with highlighting the research gaps, and developing the framework, while the 2\textsuperscript{nd} stage focuses on developing a prototype. The objectives for the 1\textsuperscript{st} stage have been formulated as follows:

- Conduct an extensive literature review that investigates existing cloud computing security issues and challenges, evaluates the existing multiple-cloud security approaches, and compares the existing multiple-cloud models to highlight their features, capabilities, requirements and architecture, to clearly identify the research gaps.

- Propose an approach that utilises the multiple-cloud computing model to preserve data confidentiality, integrity, and availability; which prevents third party involvement. In addition, avoid dependency between cloud service provider and distributing the data over a set of different geographical locations will prevent vendor-lock issue, and unauthorised access as well. Besides that, the approach must enhance the internal process automation, and minimise the interaction with the end-user in order to avoid issues that might happen due lack of experience or bad practice, and addresses the gaps identified through the literature review.
The 2\textsuperscript{nd} stage objectives that are concerned with developing the prototype are as follows:

- Develop an android application as a prototype for the developed approach (1\textsuperscript{st} stage), which will be used by individuals to protect the data through multiple-cloud storage.

- Implement extensive testing of the prototype that will be used to verify and validate the developed prototype, and produce results that will be used to evaluate and improve that approach.

1.3 \textbf{RESEARCH QUESTIONS}

In order to achieve the research aim and objectives, there are a set of research questions that have been formulated, and these questions are:

- What are the known approaches for securing data via multiple-cloud environments, and what are their capabilities?

- Is it possible to develop an approach that is capable of preserving data security through multi-cloud computing for an individual user?

- What is the most suitable multiple-cloud paradigm that might be utilised to preserve data security for individuals?

- How can we develop an inexpensive, quick, and simple approach for individuals to secure their data via multi-cloud libraries?
1.4 **Main Contributions to the Body of Knowledge**

The contributions of this study are listed according to the thesis structure as follows:

1. An updated study that illustrates the security issues, challenges and vulnerabilities associated with cloud computing, and insight into how these issues have been addressed by the proposed research.

2. A comprehensive and critical study of existing approaches for securing data in multiple-cloud environment, identifying their capabilities, but also areas for improvement that have been applied in the context of this research, and using this to design and develop an approach that bridges the existing gaps.

3. Identifying criteria, categorising, and comparing existing multiple-cloud models that can be used to evaluate these models to select the most appropriate model for the approach.

4. Identifying the need for developing an approach capable of protecting end-user data in a multiple-cloud environment that requires no agreement between cloud storage providers, which represents a contribution to the field of research.

5. Developing a new framework that utilises a multiple-cloud model that provides personal data protection by distributing the data segments over these clouds and storing one segment on the user machine to prevent any decryption/modification attempts from outside the user machine.

6. Developing an android mobile application that uses the developed approach, thus providing evidence of the real-world application of the model, and allowing new knowledge and insight to be gained.

7. Developing a road map that illustrates the study limitations and recommendations for future work, outlining future contributions.
1.5 Research Methodology

For this study, the mixed method approach was selected as the most appropriate approach that is capable of providing clear understanding, and developing deep, accurate and valid findings. This study consists of two phases based on the defined objectives: the first phase is focused on developing the approach, and the second phase focuses on the prototype development and related aspects. These phases cannot be completed by relying on only a quantitative or qualitative approach, and as such the research needs to deal with quantitative and qualitative data together. Integrating both quantitative and qualitative approaches provides the researcher with the ability to overcome and control their limitations and to benefit from their advantages, which will help to study the phenomenon in depth and to provide a clear understanding and accurate findings (Greene, Caracelli, & Graham, 1989; Greene & Caracelli, 1997; Tashakkori & Teddlie, 2010; Miles, Huberman & Saldana, 2013). After defining the selected research approach, the data collection for qualitative and quantitative data needs to be discussed alongside the tools used for analysis.

1.5.1 Data Collection

Through this research there are two types of data that are collected, which are primary and secondary data. A set of research instruments were utilised in order to collect these types of data. First of all, the research instruments used during this study are as listed below:

- Literature investigation

An exhaustive review and literature investigation are the core for building clear, current, reliable, and accurate studies. According to (Dawson, 2015), literature investigation is utilised for three main purposes: i) introducing the problem statements and providing a background
for the implemented research; ii) presenting the importance of the conducted study; and iii) evaluating and discussing the previous works and illustrating how this study is different.

Through this study, an exhaustive review and literature investigation are used in a systematic way to investigate and represent four aspects: i) the background of cloud computing technology that aims to provide a brief about the cloud computing field; ii) the existing cloud computing security issues and challenges that aims to define and highlight the security issues and challenges that must be solved, considered and overcome; iii) the existing multiple-cloud computing paradigms that aims to compare and distinguish between the existing paradigms; and iv) the existing multiple-cloud security approaches that aims to define the limitations, capabilities, obstacles, strengths, and weaknesses of the investigated approaches.

- Prototype

According to Kordon (2002) and Sommerville (2011), prototyping can be defined as an executable format of the developed approach / product, which aims to test and validate specific aspects, such as functionality, presentation, usability, and appearance. Implementing the prototype helps check both the performance and behaviour of the developed approach, whether it executes correctly or not, and if it meets customer requirements.

Through this study, the prototype is utilised to ensure that the application can be used without errors and unwanted behaviour. Moreover, the prototype is required for research, to test, validate, and verify the proposed approach. The testing and validation are carried out by the developer, end users, and experts, to provide different perspectives that help improve and enhance both the product and the process.
- Experiments

The implemented experiments aim to test and validate the performance and functionality of each feature and function through the developed prototype, which helps to discover unwanted behaviour, and improve performance in terms of functionality, consumed resources, and consumed time.

Conducting a design experiment helps to ensure that the participants implement the same tasks under the same conditions, and that will ensure that all participant test and use the same points. In practical terms the conducted experiments vary in design, participants, and purpose. All the experiments are explained in terms of design, participants, and tested aspects in Chapter V. Practically, the conducted experiments consists of four parts: i) the order of the internal task, ii) the encryption algorithms performance, iii) prototype performance, and iv) end-used experience.

- Questionnaire

According to Oates (2005), the questionnaire is a common tool used in IS research, as many researchers use it in order to collect the participants’ feedback. This type of feedback can be collected via various types of questions, such as open questions and multiple-choice questions. The questionnaire was distributed after implementing specific experiments, in order to measure specific aspects.

The questionnaire was implemented in the second phase of this study, as it aims to evaluate the developed prototype. The received responses have been filtered, analysed, evaluated, and discussed. The collected responses were used to measure the end-used feedback about the tested approach in terms of flexibility, usability, efficiency, learnability, and helpfulness.
1.5.2 Primary Data

The primary data collection took place during the second phase of this study. For this study the primary data was collected via the conducted experiments, and the distributed questionnaires. The conducted experiments help to examine, validate, and evaluate the developed prototype. These experiments vary in purpose, participants, implementation, and design as well. The developed prototype was tested by participants, and then a set of questions was used in order to collect their feedback, thoughts, and comments.

All the participants were protected from any kind of harassment, and their privacy, confidentiality and anonymity were maintained. Moreover, the data that the participants provided and collected through the research was limited to research purposes only and was not be passed to any third party. Both of the experiments and the questionnaire are discussed in more detail in Chapter V.

1.5.3 Secondary Data

The secondary data collection was achieved through literature investigation. The literature review consists of two studies. The systematic literature review focuses on cloud computing security issues, and a critical literature review extracts, analyses and evaluates the previous multiple-cloud computing security approaches. The systematic literature review was implemented due to the need to provide clear and comprehensive answers for particular questions in a rigorous and transparent way. The systematic review follows Kitchenham’s guideline (Kitchenham et al., 2009), which was chosen because of its ability to explain and detail all the activities that need to be implemented. Also, applying the quality assessment for the selected studies limits the review to high quality studies. The second study is a critical literature review, which evaluated the existing multiple-cloud security approaches in order to investigate these techniques and highlight their advantages, disadvantages, strengths, weaknesses, challenges and limitations. A critical literature review is one of the most popular
research methods and is capable of providing a critical evaluation/assessment of the current study for a specific topic. Moreover, the existing multiple-cloud computing paradigms have been covered during the literature investigation.

1.6 Structure of the Thesis

Chapter II: Literature Review

This chapter outlines the fundamental of cloud computing technology until the introduction of multi-cloud computing paradigms. Throughout this chapter, there are two Literature Review studies that have been implemented in order to explore the cloud computing security field in terms of issues and existing multiple-cloud security approaches; these consist of two independent reviews: a systematic literature review that focuses on cloud computing security issues, and a critical literature review that investigates, analyses and evaluates previous multiple-cloud computing security approaches.

Chapter III: Developed Approach

This chapter aims to bridge the research gap identified in the process of conducting research, using the designed research method to provide a clear, accurate, systematic, deep, and comprehensive explanation of the development of the approach proposed as a solution to overcome the defined limitations.

Chapter IV: Artefact Development

This chapter explains the prototype development process, and discusses the functionalities of propose approach that have been implemented via the prototype. In addition, the threats modelling will be discussed throughout this chapter.
Chapter V: Results and Discussions

This chapter presents the conducted prototype testing and explains the findings of the participants. The testing phase is explained, and the conducted experiments are reviewed and justified in terms of aim, scope, and measurements. It focuses on the experimental design and testing phase outcomes.

Chapter VI: Conclusion and Future Work

This chapter summarises the conducted study and illustrates possibilities for future work. The defined research questions are discussed, and the achievements matched to the study objectives, in order to measure the level of accomplishment. Finally, the study limitations and recommendation are listed to provide direction for future research
1.7 **Summary**

This chapter outlines the core points of the research. The research motivation and rationale are reviewed, the research aims and objectives are clearly defined and the research questions are stated. The contribution to existing knowledge is also stated and, finally, the organisation of the thesis is explained. The next chapter provides the background and fundamental information regarding the cloud computing environment, and its security.
Chapter II. LITERATURE REVIEW

This chapter aims to provide the fundamental of cloud computing technology from its roots until the introduction of multi-cloud computing paradigms. Throughout this chapter, cloud computing architecture and the deployment paradigms will be illustrated. Therefore, the core features, benefits and obstacles of this technology will be discussed. This chapter helps to provide a clear and comprehensive understanding of cloud computing technology, which is necessary to understand the nature of this technology and its way of working; this, in turn, aids understanding of the benefits, obstacles and limitations of cloud computing and the associated technologies. In addition, multiple-cloud computing will be discussed and explained in order to highlight the differences, capabilities, and limitation of each paradigm.

Following this, there are two independent studies: a systematic literature review that focuses on cloud computing security issues, and a critical literature review that extracts, analyses and evaluates the previous multiple-cloud computing security approaches. The aims, objectives and the conducted protocol for each study are then explained and discussed. This investigation provides better understanding of the field’s current status, and gives the researcher the ability to investigate, analyse, assess and evaluate the existing problems, obstacles, advantages and disadvantages from different perspectives.
2.1 **Cloud Computing**

The cloud computing revolution is one of the technologies that emerged as a result of the continual development processes of other technologies such as web services and virtualisation (Buyya, Broberg & Goscinski, 2010). Gillett and Kapor presented the first paper to introduce the ‘cloud’ in 1997 at the Massachusetts Institute of Technology (MIT) (Gillett & Kapor, 1997; Rosenberg & Mateos, 2010). The reason behind the use of the term is that researchers in the IT field usually use the cloud figure in their diagrams to represent the Internet; the cloud means the vast potential of the Internet that can be accessed and used (Rosenberg & Mateos, 2010).

A lot of definitions have been used to define cloud computing, and these vary depending on the perspective of the person who gave them (Vaquero et al., 2008; Wang et al., 2008; Buyya et al., 2009; Fox et al., 2009). The issue is the characteristics and features of cloud technology; each definition presents the feature or characteristic that is the most important, or the core of the cloud, from one perspective. Cloud computing might also be defined as an IT solution that combines hardware, software and assistant tools and presents their resources and capabilities as a service in a pay-per-use and on-demand business model to improve the availability and the flexibility of the provided services, which could be configurable by the customer with very limited service provider interaction to achieve the required SLA.

The architecture of cloud computing might be represented as a stack, based on the abstraction level. In this abstraction, each level is used to provide various services that are distinguished from the others in several aspects such as service type, customer experience and service flexibility. however, the required customer experiences have to be very high in the IaaS level and the delivered service is also flexible. On the other hand, the SaaS level has the lowest requirements of customer experiences and the lowest service flexibility (Armbrust et al., 2010; Jansen & Grance, 2011; Josyula, Orr & Page, 2011).
Figure 2-1 shows the architecture and the services that could be provided at each level of abstraction. Each abstraction level is represented as XaaS, which means providing X as a service on demand and consuming the provided services (X) remotely through the Internet, regardless of the platform used. The architecture is as follows from the bottom to the top: IaaS, PaaS and SaaS. All these levels will be reviewed in the next section.

- **Infrastructure as a Service (IaaS)**

The lowest level of abstraction is the infrastructure level, as the raw computing resources or a virtualised image are presented to customers and users are able to interact with this provided infrastructure through the Internet (Jansen & Grance, 2011). In this level, the customer must have a high level of experience to deal with the provided infrastructure and a lot of work is required because the customer is responsible for configuring this infrastructure to effectively and efficiently meet the necessary capabilities (Mahmood & Hill, 2011). In addition, this level
gives customers a high level of control and flexibility because there is a spectrum of operating systems, programming languages and configurations to select from (Mather, Kumaraswamy & Latif, 2009).

• Platform as a Service (PaaS)

Here, the abstraction is increased and the customer does not deal with the infrastructure directly. The required level of experience remains an issue, but it is less than that required in the IaaS. The flexibility decreases as a result of dealing with an unconfigurable infrastructure (Rosenberg & Mateos, 2010). The difference between PaaS and IaaS is the number of interactions needed with the infrastructure, which reduces both the required work and the flexibility provided. A customer might need to develop the software on a narrow band of programming languages (Josyula, Orr & Page, 2011). Here, the service providers will provide the infrastructure and software framework.

• Software as a Service (SaaS)

At the highest level of abstraction, the customers expect to deal with a final piece of software (application), with a little interaction with other system components as possible (Mather, Kumaraswamy & Latif, 2009). In other words, the final capabilities will be provided to the customers; and the service provider expects to deal with customers who have no experience as a developer or technician (Josyula, Orr & Page, 2011). Practically, both PaaS and SaaS need servers, operating systems and supporting tools as well, to enable and facilitate the provided services. In addition, those who provide these types of services will have more responsibilities (configuration, optimisation, management, monitoring and maintenance). On the other hand, the provided services should have the core characteristics of clouding (on-demand, self-service, elasticity, etc) (Buyya, Broberg & Goscinski, 2010).
- **Security as a Service (SecaaS)**

According to Furfaro, Garro and Tundis (2014), SecaaS could be considered as a concept that is concerned with security factors in the delivered cloud services. In addition, CSA (2016) defines SecaaS as providing security services via cloud to the consumed cloud services or from the cloud to the end-user. SecaaS gives customers the ability to consume services from one cloud service provider and use another cloud service provider to maintain the security aspects of the service that are supposed to be delivered by the first one; this enhances the security level of the consumed services and gives choices to the customer; Figure 2-2 illustrates the cloud computing services’ (SaaS, PaaS, IaaS) flexibility, costs, targeted consumers and managed layers.

![Cloud computing services flexibility, cost, targeted consumers, and managed layers](image-url)
2.2 **MULTIPLE-CLOUDS COMPUTING PARADIGMS**

Shifting from a single cloud to multiple-clouds becomes necessary for many businesses and customers. Multiple-cloud paradigms have been developed as efficient solutions in order to overcome some single cloud paradigm obstacles and limitations, and enhance the efficiency of ICT cloud-based solutions (Toosi, Calheiros & Buyya, 2014); these motivations can be organised into six categories as shown in Figure 2-3. Multiple-clouds are a relatively new field and there are no standard definitions or classifications that state the features, advantages, disadvantages and requirements of the existing multiple-clouds models. In other words, a potential user of multiple-clouds is confronted with a plethora of terms that are used for referring to the different multiple-clouds models. Many of the used terms do not properly highlight the key features of a specific multiple-cloud model. The problem is not limited to the absence of clarity in the models’ definitions; a user will also find out that more than one term can be used to represent exactly the same model.

To address these concerns, we developed a set of classification criteria that might help to provide better understanding and clear differentiation between these paradigms, which has led to selecting the paradigm(s) that is/are most appropriate for the customers’ system requirements.

![Figure 2-3 Multiple-cloud computing motivation (Toosi, Calheiros and Buyya, 2014)](image-url)
2.2.1 Classification Criteria

There are a lot of metrics that might be used to develop a multiple-cloud classification, for instance, ownership, management, charge, control, migration, and so on. In this section, we focus on harmony and interoperability between the different instances of the multiple-clouds; this brings forth three main aspects:

- Agreement (Ferrer et al., 2012)
- Collaboration types (Ferrer et al., 2012)
- Migration methods (Petcu et al., 2011; Kecskemeti et al., 2012)

For the agreement and collaboration aspects, the metrics that might be used for agreement categorisation are: 1) prior-agreement between the cloud providers; or 2), no prior-agreement. In addition, the collaboration could also be loosely coupled, partially coupled and tightly coupled. The migration methods concern two key metrics, i.e. what will migrate (service or application), and how will it migrate (one time or real time). The agreement and collaboration, and migrations metrics are shown in Table 2-1.
<table>
<thead>
<tr>
<th>Agreement</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prior-agreement</td>
<td>The cloud service providers have a formal agreement between them that identifies all the collaboration aspects, responsibilities and privileges. The end-user will interact with these clouds as a single cloud.</td>
</tr>
<tr>
<td>No prior-agreement</td>
<td>The cloud service provider will not enter into an agreement and the third party (might be the end-user) will negotiate with the providers independently.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Collaboration</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loosely coupled</td>
<td>This level usually occurs between an independent cloud (private) and public cloud, and the best example for this is the hybrid cloud. At this level, the private cloud manager has complete control over the private cloud, but has very limited control over the other cloud; in some cases, there is no control over the remote cloud at all. This level is suitable when the resources have to be expanded (temporary) to deal with loads, and for small organisations that need to reduce the operational costs.</td>
</tr>
<tr>
<td>Partially coupled</td>
<td>This level occurs when two or more independent clouds have an official agreement to establish a predefined collaboration framework. Regardless of the collaboration level, this agreement should define all the collaborative resources, terms, conditions and control privileges. At this level, the control and monitoring of information will be exchanged between the clouds, and some clouds will have a defined level of control over the other clouds. The partially coupled level is suitable for organisations that need some sort of control over the remote resources.</td>
</tr>
<tr>
<td>Tightly coupled</td>
<td>This level usually applies to clouds that belong to the same organisation, which allows advanced control over the collaborated clouds, meaning that the cloud manager has the privilege of monitoring, auditing, managing, creating images and configuring. This level is optimum for systems that need a sort of centralised management and look for high-levels of throughput.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Migration</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service migration</td>
<td>The user will consume the remote cloud by migrating the service (SaaS, PaaS, IaaS) to it, which means that the remote cloud might be used as a layer (SaaS, PaaS, IaaS) to deliver the requested service.</td>
</tr>
<tr>
<td>Application migration</td>
<td>The application only will migrate to the remote cloud.</td>
</tr>
<tr>
<td>One time migration</td>
<td>The migration process will be done once by redeploying/rewriting the data/applications. Also, a sort of limited synchronisation process might be applied.</td>
</tr>
<tr>
<td>Real time migration</td>
<td>The type will apply a complete synchronisation for all the involved clouds.</td>
</tr>
</tbody>
</table>

Table 2 - 1 Multiple-clouds agreement, collaboration, and migration metrics (Ferrer et al., 2012; Petcu et al., 2011; Kecskemeti et al., 2012)
2.2.2 Multiple-clouds paradigms

The classification presented in this section was developed according to the criteria defined in the previous section. The terms that represent general concepts and the terms that have absence of specifications are not included in this classification, but will be included in Appendix A, which shows the terms and their main features, Figure 2-4 shows the multiple-clouds paradigms classification, and Table 2-2 illustrates a comparison between the paradigms based on the discussed criteria.

![Multiple-Cloud Paradigms](image)

**Figure 2 - 4 Multiple-cloud paradigms**

2.2.2.1 Intra cloud (clouds of clouds)

Intra-cloud computing happens when there are two or more different cloud services that belong to the same cloud service provider, which need to collaborate together (Chen, Nepal & Ren, 2011). Some resources (Cisco, 2014) define the intra-cloud as a cloud-networking method rather than a multiple-cloud model, as the clouds infrastructure or services cannot be totally independent. The intra-cloud is a sort of internal collaboration to deliver the required services. Figure 2-5 shows an example of intra cloud.
2.2.2.2 Hybrid cloud (bursted)

The hybrid-cloud has been defined by NIST as a deployment method other than public or private cloud (Hogan et al., 2011). The hybrid-cloud model is used when the private/community cloud cannot deal with loads or needs additional resources to fulfil the end user expectations. Therefore, the private/community cloud needs to expand the resources/capabilities to achieve the SLA, which can be done by utilising other public cloud resources/capabilities for a specific period then releasing these resources (Elmoth et al., 2011; Leavitt, 2013). The problem with the hybrid-cloud model is the absence of security, because being public will threaten the cloud directly and expose the consumed resources and data to danger (Elmoth et al., 2011). In this model, there is no need for prior-agreement, and the clouds will be loosely coupled. Also, it is suitable for one-time application migration. Generally, this type is cheap and does not need a high level of expertise to perform it, and could be done instantly. The core issue with this type is the security as the data will be exchanged between two different security domains, and that is a critical threat from an information security perspective. Figure 2-6 illustrates an example of this model.
2.2.2.3 Federated cloud

The federated-cloud is a form of interoperability cloud computing, when two or more independent cloud-service providers agree to share their infrastructure and collaborate together in order to deliver the required services within the agreed QoS level (Ferrer et al., 2012). Technically, the federated-cloud is preferred for governmental use for several reasons, such as the high cost of the required infrastructure and operation and ownership problems (Buyya, Ranjan & Calheiros, 2010). The core point that distinguishes this type from the others is the communications between the involved clouds, in this type of multiple clouds, the collaborated clouds will exchange status between each other, such as occupied resources, available capabilities, and loads. Technically, the clouds are tightly coupled with each other, and any issue with one will influence the system performance. The federated-cloud could be divided into two categories based on communication (interaction) methods, which are centralised and peer-to-peer methods (Moreno-Vozmediano, Montero & Llorente, 2012). In this model, there is a prior-agreement between the cloud service providers, and a cloud’s collaborative level and migration will vary between the centralised and peer-to-peer approaches.
2.2.2.3.1 Centralized federated-cloud.

In this type, the federated-cloud uses a central entity as a connection/control node between the clouds. The central entity is responsible for resource allocation, resource optimisation, load balancing, cloud interconnecting, cloud status, monitoring, and management (Grozev & Buyya, 2014). The users do not communicate with the central entity directly at all, and they communicate with the clouds directly as a single cloud. Within this model, the broker has more control over the federated-cloud because of the central entity; also, the operational costs are lower than those of the peer-to-peer model and that is caused by the method of communication. However, the centralised federated-cloud model provides only a single communication channel between each independent cloud service and the central entity, which means losing the availability of the federated-cloud if the central entity is down, or isolating the cloud service that loses its connection channel to the central entity. One of this paradigm’s drawbacks is the high level of dependability, which will impact on performance. Also, relying on a third party might not acceptable for all users. The design of this model is illustrated in Figure 2-7.

![Figure 2 - 7 Centralized federated-cloud (Grozev & Buyya, 2014)](image)

2.2.2.3.2 Peer-to-peer federated-cloud.

In the peer-to-peer model, each cloud service is connected to another directly, and all the management, load balancing, communication, charging and monitoring processes are
embedded in each individual cloud service (Villari et al., 2012). This model provides users with the highest availability and performance levels because of the redundancy and dynamicity in the communication channels between clouds (Li et al., 2011; Hassan et al., 2012). On the other hand, the operational costs are very high, which is due to:

- The number of communication channels that each cloud service uses.
- The need to keep the status of each cloud fully synchronised.
- The number of operational processes that are used to compute the load balance.
- The exchange of data used to calculate the optimum resources/capabilities.

As a result, this model might not be an efficient solution for a large number of cloud services, because as the cloud number increases, the operational costs will increase proportionally. In case of losing direct connectivity between two independent cloud services, there are additional channels that could be used and the user requests can be handled easily. Also, losing the availability of an independent cloud does not impact on the system’s availability because all the requests that were assigned to the absent cloud are directed to the other clouds, Figure 2-8 shows an example of peer-to-peer federated-cloud. Technically, the expensive operational cost for this paradigm might make it not suitable for some applications that do not have the necessary budget and expertise to deploy, operate, and manage such a type.

Figure 2-8 Peer-to-Peer federated-cloud (Elmroth et al., 2011)
2.2.2.4 Multi-cloud

In a multi-cloud computing model there is more than one independent cloud service that is used to execute the requested tasks through consuming the provided resources/capabilities; and here the customers will take the onus of resources/capabilities management, task scheduling, and load balancing, usually without any need for a prior-agreement between the involved cloud services (Ferrer et al., 2012). The multi-cloud can be applied through private clouds, regardless of the ownership identity (governmental/private) or any types. Similar to the federated-cloud, multi-cloud can be categorised into services and libraries. Technically, the end user is responsible to formulise how these cloud services will collaborate, and maintain the cloud services status. In this model, prior-agreement might be required, which will vary between the services (required) and libraries (not required) models. In addition, the collaboration level varies between loose and partial.

2.2.2.4.1 Multi-cloud services.

In the multi-cloud services model, the cloud service providers have to define and agree on the rules for the execution and delivery processes. After that, the developers will develop a service (mediator), which will be used to deliver the multi-cloud services (Elmroth et al., 2011; Grozev & Buyya, 2014). The developed service works more like a mediator between the users and the consumed cloud services. The problem with this model is that the absence of a multi-cloud services broker leads to losing the multi-cloud availability completely. In addition, lack of experience, or bad practice of management, may lead to exposing the involved cloud infrastructure, and the end user as well. Figure 2-9 illustrates multi-cloud services design.
2.2.2.4.2 **Multi-cloud libraries.**

In this model, the users develop their own service broker through a unified API (Kecskemeti et al., 2012). The developed broker is formed in libraries and it is embedded in the users’ access point (machine) (Petcu et al., 2011). This model offers an availability level higher than the previous model, which is caused by associating the multi-cloud library in the user’s machine. However, the users must be qualified to develop their multi-cloud libraries. Besides that, the required number of connections matches with the number of used cloud services, as there is a single connection between the user machine and cloud services, which will reduce the consumed resources and operational costs as well. Additionally, in case of failure there is no impact on the other cloud services, as all the utilised clouds services are completely isolated and independent; the design of this paradigm is illustrated in Figure 2-10. The only drawback of this paradigm is the need for a high level of expertise; on the other hand, this type will prevent any third party getting involved, which is very critical for some applications in terms of security.
Figure 2 - 10 Multi-cloud libraries (Grozev & Buyya, 2014)

<table>
<thead>
<tr>
<th></th>
<th>Agreement</th>
<th>Collaboration</th>
<th>Migration</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Prior-agreement</td>
<td>No Prior-agreement</td>
<td>Loosely coupled</td>
<td>Tightly coupled</td>
</tr>
<tr>
<td>Intra</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Hybrid</td>
<td></td>
<td>✔</td>
<td></td>
<td>✔</td>
</tr>
<tr>
<td>Centralised</td>
<td>✔</td>
<td></td>
<td></td>
<td>✔</td>
</tr>
<tr>
<td>Peer-to-peer</td>
<td>✔</td>
<td></td>
<td></td>
<td>✔</td>
</tr>
<tr>
<td>Services</td>
<td>✔</td>
<td></td>
<td></td>
<td>✔</td>
</tr>
<tr>
<td>Libraries</td>
<td>✔</td>
<td></td>
<td></td>
<td>✔</td>
</tr>
</tbody>
</table>

Table 2 - 2 Multiple-cloud computing paradigms comparison
2.3 SHIFTING TO THE CLOUD

Shifting to cloud computing services brings several advantages that support the shifting decision; on other hand, there are obstacles/concerns that have been raised by customers that might affect negatively their decision to consume cloud services; both the advantages and obstacles vary according to the targeted cloud service, service scale, budget and customer needs.

The benefits of consuming cloud computing services instead of using traditional ICT solutions is inspired by the key cloud computing characteristics, which have been explained and discussed earlier in this chapter (Furht & Escalante, 2010; Josyula, Orr & Page, 2011; Jansen & Grance, 2011; Mell & Grance, 2011). In addition, there are other benefits that have been addressed by several studies. For example, adopting the cloud will help to reduce operational costs and shortage of associated ICT budget; this occurs because the end-user rents the needed resources/services and is charged for the measured consumption, instead of buying/building the needed resources/services (Buyya et al., 2009). Also, maximising the utilisation of the consumed resources plays a role in reducing the costs (Armbrust et al., 2010).

Moreover, relying on a reliable and trustworthy service provider helps to outsource the IT management and simplify the management, operations, maintenance, upgrade and the needed expertise level, which helps to guarantee the QoS and maximise the efficiency of the provided resources/services. Also, the cloud helps to reduce power consumption, which is very important in terms of the environment and costs (Sosinsky, 2010; Alkhater, Wills & Walters, 2014). Further, the flexibility of the delivered services is a key advantage that the consumer will gain from the cloud; the flexibility allows scaling, customisation and control of the delivered services according to customers’ needs and budget (Marston et al., 2011; Buyya, Calheiros & Li, 2012). Furthermore, the failure possibility of the delivered applications (software) is minimised and the mobility is enhanced.
There are issues that have been addressed as obstacles that might prevent or reduce migrating to cloud computing services. Similar to the advantages, the obstacles vary based on the delivered service characteristics and the consumers. However, these issues differ from resource management, availability, privacy, security, recovery, ownership, dependency and governance, to political issues (Buyya et al., 2009; Dikaiakos et al., 2009; Armbrust et al., 2010; Sosinsky, 2010; Alkhater, Wills & Walters, 2014; Hsu, Ray & Li-Hsieh, 2014).

Armbrust et al. (2010) define ten obstacles and divide them into three groups based on their influence; for example, the cloud adoption obstacles include availability, data lock-in and confidentiality; further, business and policy obstacles are software licensing and reputation. Furthermore, data lock-in, government policy, government regulations, privacy and confidentiality leakages have been classified as business concerns. On other hand, Alkhater, Wills and Walters (2014) categorise the obstacles factors as follows: technological factors, organisational factors and environmental factors. Obviously, the technological factors concern the security aspects, as they define availability, reliability, security, privacy and trust as major concerns.
2.4 CLOUD COMPUTING SECURITY ISSUES SYSTEMATIC LITERATURE REVIEW

As we mentioned earlier, cloud computing is a huge market; it grows and expands in a continuous and rapid way. The reason for this continuous growth and increased demand on cloud computing services can be summarised as follows:

- The core features of cloud computing technology, which are pooling resources, elasticity, on-demand, pay-per-use and ubiquitous access (Josyula, Orr & Page, 2011).
- The diversity in the deployment methods (public, community, private, hybrid).
- The diversity in delivery methods (Infrastructure as a Service, Platform as a Service, Software as a Service) (Furht & Escalante, 2010).

Similar to other technologies, cloud computing has some open issues and limitations. According to Bouayad et al. (2012), security is one of the four main concerns and roadblocks for the majority of cloud customers that prevent an individual or organisation moving to the cloud.

Some studies are concerned about cloud computing from a security perspective. For instance, Armbrust et al. (2009) argue that cloud computing security drawbacks could be summarised into ten points, five of which are related to security: availability, data lock-in, confidentiality, auditability and malware/bugs inside the system. There are two core weaknesses in the work of Armbrust et al. (2009); the first is a lack of detail that might help to enhance understanding of the identified obstacles; for instance, the authors address auditability as an issue, but there is a critical absence of essential information that explains why audibility is an issue and no real cases are provided. The second defect is the date of the study; since it was undertaken in 2009, some issues have been solved, others have been considered as legal issues, and there are new issues that it could not possibly address.
Gonzalez et al. (2012) have released a quantitative analysis of current security concerns. This study develops a cloud computing security taxonomy that consists of privacy, architecture and compliance. They suggest that coming up with a format that is able to harmonise more than one solution will be more efficient and allow the enhancement of the delivered security level, as opposed to developing a single solution that is able to treat all the cloud computing security issues/challenges. It is obvious that this study focuses on the virtualisation aspects more than the others.

The survey by Modi et al. (2013) attempts to show a number of vulnerabilities, threats and attacks associated with cloud computing security. The study assigns the identified vulnerabilities, threats and attacks with proper solutions. However, the new cloud computing models, for instance, multi-clouds, mobile-cloud and ad-hoc cloud have not been included within this study.

There is, therefore, no systematic review study that is either up-to-date or comprehensive and accurate. Consequently, this study will look at the existing literature of cloud computing security issues. This will be implemented via searching, filtering, classifying and analysing the previous related studies, which will provide a detailed and comprehensive overview of what is already known about cloud computing technology in terms of security. Additionally, it will identify the potential challenges and threats of developing, providing, managing and consuming cloud-computing services.

The next section summarises the applied review strategy and explains the review process. Then, the review outcomes will be presented and after that, the study outcomes will be discussed.
2.4.1 Review Strategy

The Kitchenham systematic review guideline (Kitchenham, 2004; Keele, 2007; Kitchenham et al., 2009) has been used in order to define a clear strategy for implementing this systematic review. The guidelines steps are summarised in Figure 2-11.

- Defining the study’s questions and scope.
- Collecting the related papers.
- Extracting and analysing the papers’ contents.
- Defining the research’s strategy and the data sources.
- Applying quality assessment test.
- Answering the review’s questions.

As we mentioned before, this review has been developed in order to support and prove the study assumption; which is that ‘the single cloud storage can not be used as a secure and trusted method for protecting sensitive data’. For that, there are two research questions that have been identified.

- **What is known about cloud computing security?**

This question aims to provide a general overview of what is known about security aspects of cloud computing. Through the review process, there are a number of papers that focus on the security of cloud services from different perspectives, which are those of the owner, developer, and end user. The answer to this question will help to classify the published research based on the followed research methodologies for each study, and based on the defined aims for each study. These classifications will help to highlight the need for developing studies that propose approaches, testing, evaluating, investigating or explaining specific issues or topics.
What are the most critical security challenges and issues of cloud computing?

This question aims to address the objective of this review, which is highlighting the challenges and issues of cloud computing in terms of security and data protection. Additionally, this review will identify those aspects that are the most critical weaknesses. In order to do that, the addressed challenges and issues have been collected, extracted, analysed and documented in the results section.

2.4.2 Selection Procedure

As a part of the selection procedures, the selected studies had to go through four phases in order to select the most relevant studies and enhance the quality of the extracted data. Figure 2 - 11 illustrates these phases and the number of publications for each phase.

The first phase was based on the use of keywords; the selection was performed based on the publication’s title and whether or not it related to the cloud computing security field. The second phase focused on the publication’s abstract, which was based upon the papers that had been
selected at the first phase. In some cases, the abstracts are vague and cannot provide the paper concept clearly; in this instance, the sub-titles, figures, tables and main contents were scanned.

The third phase was reading the selected publications and identifying important information such as the study’s type, scope and aim. Finally, quality assessment was applied to identify the final selected publications. The applied quality assessment consists of nine factors, which can be seen in Table 2-3; these factors have been developed based on the S. L. R protocol and from the reviewed previous studies (Jamshidi, Ahmad & Pahl, 2013; Latif et al, 2014). After applying the quality assessment, all the publications that had a score less than five were discarded so as to enhance and guarantee the quality of the review; the score rating can be explained as follows: each selected publication was asked nine Yes/No questions, scoring one point for a ‘yes’ response and a zero for a ‘no’, giving a total out of nine. The detailed publication and assessment criteria can be seen in Appendix B.

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>Research Aim</td>
<td>Does the publication identify the research aim/s clearly?</td>
</tr>
<tr>
<td>Research Design</td>
<td>Does the publication explain the applied research methodologies?</td>
</tr>
<tr>
<td>Contribution</td>
<td>Does the publication outline the added value and the contribution?</td>
</tr>
<tr>
<td>Data Collection</td>
<td>Does the publication explain the applied data collection methodologies?</td>
</tr>
<tr>
<td>Sample</td>
<td>Does the publication provide information about the sample (type, size…etc.)?</td>
</tr>
<tr>
<td>Data Analysing</td>
<td>Does the publication provide a clear explanation regarding the data analysing process?</td>
</tr>
<tr>
<td>Research Outcomes</td>
<td>Does the publication state the research outcomes clearly and discuss these outcomes?</td>
</tr>
<tr>
<td>Cloud Computing</td>
<td>Is the publication related to cloud computing?</td>
</tr>
<tr>
<td>Cloud Computing Security</td>
<td>Is the publication related to cloud computing security?</td>
</tr>
</tbody>
</table>

Table 2-3 Publication assessment criteria
2.4.3 Cloud Security Challenges and issues

In order to provide a comprehensive and complete answer for RQ1 (2.5.1, p.p. 34), the following information has been extracted; this information was used to render a statistical overview regarding what is known about cloud computing security. Figure 2 - 13 represents the selected publications based on the searched databases. The IEEE has the biggest share of publications (41%) and Springer has the lowest number of selected publications, which represents only (9%); the rest of the results are in Appendix C.

![Publications Distribution (DataBase)](image)

Figure 2 - 13 Publications number based on the year and the database

In order to answer RQ2 (2.5.1, p.p. 35), selected publications have been analysed to identify cloud computing security challenges and threats. The extracted challenges and threats have been classified into six categories, which are shown in Figure 2 - 14. This classification is based on the source of the identified challenges/threats, which are as follows: software, virtualisation, Internet, trust and access.
However, there are some aspects that need to be clarified, which are the terms used and the applied classification. Throughout the review process there are a number of terms that are used to define and describe the identified issue/challenge, such as threat, risk and vulnerability.

The other aspect is the cloud computing security issues/challenges classifications. The cloud computing security issues classification varies between the reviewed publications. For instance, Zissis and Lekkas (2012) classify the identified issues into five classes as follows: data control, account control, multi-tenancy, malicious insiders and management console security. The addressed threats are limited to cloud computing only, and are not related to involved technologies such as web services, encryption, or communication protocols. On the other hand, Sengupta, Kaulgud and Sharma (2011) categorise the issues that they discuss into four categories: infrastructure, access, data and compliance, which rely on bottom-top layer concerns. The study by Grobauer, Walloschek and Stocker (2011) divides the associated security issues into general security issues and cloud security issues. Additionally, Gonzalez et al. (2012) propose a taxonomy that has three groups as follow:
1- Cloud architecture security: this group concerns the security of networks, virtualisation and interfaces.

2- Data privacy: this group focuses on both the security of data itself and the legal issues.

3- Compliance: this group concentrates on the responsibilities of each party and the applied policies.

Further, some organisations (CSA, NIST, ENISA, etc..) propose classifications. In the 2017 CSA report, the threats have been categorised based on a combination of the threat’s nature and the influenced domains (CSA, 2017b).

As we mentioned before, the purpose of this study is to highlight cloud computing security issues/challenges in order to prove that cloud computing services have a lack of security. For that, the proposed classification attempts to gather all the identified issues based on the origin of these issues. Figure 2 - 15 illustrates these issues and their categories. After that, these issues are discussed in depth to provide a comprehensive understanding of the issues raised and to identify the source of threats/vulnerability.
2.4.3.1 Software security issues

In traditional IT solutions, software security is a key issue. By moving into the cloud, the importance of consumed software security increases and that returns to the characteristics of cloud computing; for instance, the shared resources and capabilities and multi-tenancy (Grobauer, Walloschek & Stocker, 2011). In other words, the number of users, resources and services that could be affected by any unsecured software will be greater compared with traditional IT solutions. In this section, we will focus on the security issues in the developed software as a platform and interface. The majority of software security issues in the cloud
result from the absence of standards, good practices and metrics that could be applied through the development process and help to decrease the security and reliability (Hashizume et al., 2013; El Moctar & Konaté, 2017). In addition, the wide spectrum of the available programming languages and methods make producing standards and restricting roles too difficult. Based on the consumed cloud services, the software security issues could be divided into two sections: 1) the platform used to develop the delivered applications is limited to IaaS and PaaS, whereby the developer has a high level of flexibility in terms of having access to the consumed infrastructure; and 2), the front-end used to consume and access the delivered service, which could occur in SaaS, PaaS and IaaS. The majority of the identified issues are related to the developer and owner perspectives rather than the consumer.

1. **Platform**

The security issues related to the platform are limited to PaaS. As we mentioned before, the impact will not be limited to the customers who developed their software, but could impact on the other tenants and the shared resources and capabilities. Here the cloud service provider must ensure that the delivered platform is robust and able to isolate the executed code. Also, the PaaS providers have to ensure that its granted privileges will not affect the shared infrastructure (Subashini & Kavitha, 2011). The development platform could raise issues via two routes: firstly, when the delivered platform itself does not have the required level of isolation; secondly, when the developed software does not have a robust security structure and standards.

Rodero-Merino et al. (2012) discuss and analyse two development platforms (Java and .NET), stating that, ‘none of these platforms offer a fully secured hosting environment’ because of the unlimited impact on the delivered and shared resources. Using Java as a development platform will offer runtime checking for the executed cryptography and secure data transactions. Additionally, all the untrusted code will be isolated before execution. However,
Java has a problem with machine isolation, and there are two main proposed solutions. The first one isolates the executed application, development platform and operating system and this solution works perfectly, but it will not be efficient and effective in the case of multi-tenancy because of the very high cost of this type of isolation (Czajkowski & Daynás, 2012; Jia, Zhu & Liu, 2013). The second solution only isolates the executed application (code), which might be accepted in some cases with trusted customers (Majoul et al., 2010; Czajkowski & Daynás, 2012).

In addition, resource accounting is another problem that Java and .NET share. This problem means that when the executed code receives permission to consume the resources, neither Java nor .NET are capable of accounting for the consumed resources (Czajkowski & Daynás, 2012). However, terminating the thread in an unsafe way is an issue for both platforms. Unsafe termination will cause a serious problem, for instance, maliciousness in the execution process can consume the provided resources forever and prevent the other processes from operating (Czajkowski & Daynás, 2012). Also, there is no ability to execute a safe shutdown while there is an unterminated thread.

There are number of proposed solutions; however, the problem with these solutions is that they concentrate on the isolation issue. Plus, although these solutions work as expected, they ignore the other issues, which increases the complexity of the problem and the current system (Rodero-Merino et al., 2012).

2. User front-end

The user front-end refers to the interface that will be used by the customer to interact with the cloud, and this interface can interact with the cloud through the Internet. Connecting an element to the Internet will cause it to be exposed to a wide variety of attacks. In the case of SaaS, the user interface is threatened by different attacks, for instance, unauthorised access and masked code injection to breakdown the applied isolation (Grobauer, Walloschek &
Stocker, 2011; Tianfield, 2012; Vesyropoulos, Georgiadis & Pimenidis, 2014; Dixit, 2017). In the case of PaaS and IaaS, the interface could be used to modify the configuration of the delivered infrastructure and take control of the virtual machine monitor instead (Czajkowski & Daynàs, 2012; Jia, Zhu & Liu, 2013; Derfouf, Mimouni & Eleuldj, 2015; Hussain et. al, 2017). In addition, the majority of the software involved in front-end development is open source software; this means that software could be compromised easily and the possibility of being attacked through this breach is very high if the used open source software or Application Programming Interface (API) were from untrusted sources. Moreover, the lack of experience and the absence of good practice through using such software can led to expose the consumed infrastructure and impact negatively on the delivered service and the consumed infrastructure as well (Derfouf, Mimouni & Eleuldj, 2015; Khan & Al-Yasiri, 2016; Shanmugasundaram, Aswini & Suganya, 2017).
2.4.3.2 Virtualisation security issues

Virtualisation is a core technology element in cloud computing and it helps to increase the investment in this new market by increasing the number of virtualised machines, which leads to an increase in the number of tenants (Takabi, Joshi & Gail-Joon, 2010). However, there are some difficulties and issues related to virtualisation. Some of these issues are directly related to security aspects and others might be used to cause threats (Takabi, Joshi & Gail-Joon, 2010). Generally, virtualisation security issues have been analysed, investigated and studied by a lot of researchers as independent technology (Reuben, 2007; Pearce, Zeadally & Hunt, 2013; Rehman et al., 2014). According to (Jararweh et. al, 2016), applying the traditional security approach over virtualised environment is not a practical nor efficient solution.

1. Availability

Availability refers to the responsibility of the cloud service provider to deliver the virtualised resources when required. The lack/absence of virtual resources' availability could be caused by DoS attacks, which aim to affect the availability in the physical infrastructure, which will impact on all the tenants that share the same resources (Takabi, Joshi & Gail-Joon, 2010; Basu et. al, 2018). Also, these attacks could be aimed at the virtualised resources, in which case the impact will be limited to the attacked image and have very limited impact on the physical infrastructure (Tsai et al., 2012; Zhang et. al, 2018). In addition, losing availability could be a consequence of implementing an improper load balancing mechanism.

2. Virtualisation management

In the virtualised machines there is low-level software called a virtual machine monitor, which manages and controls the virtual images and links the virtualised images with the physical infrastructure (Hashizume et al., 2013). One of the problems that VMMs (hypervisor) might
face is related to encryption, which arises if the VMM has to deal with large encrypted images (Vaquero, Rodero-Merino & Morán, 2011). In addition, all the images, especially the offline ones, should be kept and be perfectly protected against two main threats. The first threat is stealing these images and using them to understand the architecture and the underlying framework for any malicious purpose (Jansen & Grance, 2011). The second threat is injecting the image with malicious damage (Hashizume et al., 2013; Jia, Zhu & Liu, 2013). In addition, there is an issue called sprawl, which happens when the delivered resources and capabilities are occupied and consumed by images that are in the idle mode, as these resources cannot be controlled and managed efficiently. The sprawl will prevent the other executed images using all the resources and capabilities, which are occupied by the idle images (Luo et al., 2011; Ali, Khan & Vasilakos, 2015). This is a serious issue, but the worst scenario could happen when the idle image is injected maliciously to attack the infrastructure or occupy extra resources/capabilities.

In the virtualised network, the traditional solutions for managing and controlling the traffic cannot provide the same efficiency as if these solutions were applied through the physical layer (Vaquero, Rodero-Merino & Morán, 2011). For instance, Grobauer, Walloschek and Stocker (2011) point out that some control mechanisms such as IP zoning cannot be applied through a virtualised infrastructure. Wang and Ng (2010) find that virtualisation will cause varied packets delay and instability into Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). As a result, limitations on the administrative aspects will arise and an attacker could easily use these limitations to reach the cloud infrastructure. Furthermore, the virtualised network will suffer from packet sniffing and spoofing, and some other security issues related to establishing the virtualised channels in a dynamic way.

For the VM aspect, the studies of Vaquero, Rodero-Merino and Morán (2011) and Perez-Botero, Szefer and Lee (2013) state that monitoring each VM will consume the cloud’s resources and capabilities because of the huge number of computational processes involved.
Besides that, there is a form of attack called escape (Egele et al., 2012), which happens when the VM uses the delegated privileges to attack the VMM (hypervisor); in this case, as soon as the attacker (i.e., the VM) gains access to the VMM then all information that is related to the consumed resources, shared resources and monitoring will be available for exploitation by the attacker. Also, there are other escape attack mechanisms that vary in complexity and purpose. In the black market, for example, there are packs called crime packs and exploit packs; these packs include vulnerabilities that are not known in the security communities. These vulnerabilities will be used to attack the VMMs and the sold vulnerabilities are known as ‘zero day vulnerabilities’ (Modi et al., 2013). Subashini and Kavitha’s (2011) and Ali, Khan & Vasilakos (2015) studies identifies three VMM monitoring issues, which are interposition, isolation and inspection. Moreover, the failure of image isolation, and unwanted/unexpected consumption of cloud resources will lead to loss of control, infrastructure unavailability, and cloning issues (Tep et. al, 2015).

3. Mobility

Mobility here refers to the ability to deploy the VMs through the physical machine, and there are issues associated with this (Tsai et al., 2012; Halabi & Bellaiche, 2017). Deploying the images without examination and evaluation of the integrity and health of the deployed image will cause problems (Pék, Buttyán & Bencsáth, 2013; Zhang et. al, 2018). For instance, the deployed image may contain malicious exploits to take control or cause the service’s denial. In addition, copying and moving the images (VMs) from one physical machine to another will result in VM cloning, which is caused by cloud computing elasticity (Grobauer, Walloschek & Stocker, 2011; El Moctar & Konaté, 2017). Further, VM cloning will produce copies of the VM that will have the owner’s data. That means the incremental degradation of the available and monitored images, which consumes a lot of computational resources and capabilities (Duncan, Creese & Goldsmith, 2012; Tep et. al, 2015). As a result, the VMM must give less attention to some images in order to save cloud resources and capabilities, and then the
ignored images can be easily attacked. Moreover, insecure migration was defined as a critical threat (Khan & Al-Yasiri, 2016).

4. Malware

The new malware targets are expected to be cloud service providers, developers and consumers. Across the literature there are many examples of methods or breaches that could be the potential backdoor to infection by malware. For instance, Basak et al. (2010) and Ali, Khan & Vasilakos (2015) state that one of the infection methods is the rollback function. The rollback can be a main cause of infection by malware because executing the rollback function will lead to out-dating the used firewalls and anti-virus software, which will let the malware run and reach its targets. Also, the rollback function can mop up the new patches and this allows an attacker to go through using the un-patched vulnerability.

In addition, the malware in the virtualised environment has the ability to evade detection (Dahbur, Mohammad & Tarakji, 2011; Li, Loh & Tan, 2011; Ali, Khan & Vasilakos, 2015). Malware that is detected by Symantec products (i.e., Trojan.Maljava) was the first malware that copied itself through the VM images and spread through the VMs. Malware injection is an issue at the virtual machine level. Egele et al. (2012) and Li et. al (2015) studied such malware injection attacks, noting that these attacks start by injecting a VM or service into the cloud and executing it. Then, the requests that are opened by the authenticated user are intercepted by the attacker and replaced with the attacker code. In addition, Zhang et al. (2012), Okamura and Oyama (2010) and Ali, Khan & Vasilakos (2015) have identified two vulnerabilities, which are the side-channel and covert-channel attacks. The side-channel attack is based on observing the data flowing in a passive way without any interference, and the covert-channel attack injects bits to get specific information. Both of these attacks are too complex and cannot be performed simply. According to Green (2013), and El Moctar & Konaté (2017), these attacks are dismissed because of the high level of experience, skills and knowledge required.
2.4.3.3 Internet security issues

Cloud computing uses the Internet as a medium to both deliver and manage the required service. In particular, the Internet could be used in the traditional form (global network) or in private, where the access to the connection will be restricted to specific users (employees) (Yu et al., 2012). Likewise, for other involved technologies, the Internet can bring a massive and increased number of security issues; nevertheless, cloud computing services have been used as a suitable transmission medium for various reasons including cost, availability, standards, support and service type diversity.

1. Protocols and standards

The communication processes need protocols to identify the communication roles and enhance the harmony between the connected nodes. There are a large number of protocols and standards that are involved in the communication and these will vary based on their locations (in which layer) and objectives. Technically, the Internet Protocol (IP), Dynamic Host Configuration Protocol (DHCP) and Domain Name System (DNS) are the core protocols for the TCP/IP network model. All these protocols have critical vulnerabilities that could be used to breach the protection and take down the delivered cloud services. For example, any one of these attacks such as IP spoofing, DHCP snooping and DNS cache poisoning could be used to facilitate the network-based cross-tenant attacks (Chen & Zhao, 2012).

By moving to the higher layers, the HyperText Transport Protocol (HTTP) is the direct protocol that the user interacts with to manage, configure and receive the delivered cloud service. The HTTP sessions could be attacked by various types of attacks and objectives (Grobauer, Walloschek & Stocker, 2011; Ali, Khan & Vasilakos, 2015). For this reason, the HyperText Transport Protocol Secure (HTTPS) was introduced, which was supposed to improve the security level. Nonetheless, bad practices in implementation, and combining HTTP and HTTPS in the same session, has made HTTPS an inefficient solution because using HTTP
will provide attackers with breaches to reach the session contents (Prandini et al., 2010; Ali, Khan & Vasilakos, 2015).

By moving to PaaS, the HTTP or HTTPS could be used after deploying the developed software; however, the use of the Simple Object Access Protocol (SOAP) is recommended for using the platform, which is the optimum one to deal with complex service and data structures (Belqasmi et al., 2012). On the other hand, the File Transfer Protocol (FTP) and Virtual Private Network (VPN) are optimum for IaaS users because of the provided security and the ability to deal with a low-level of infrastructure (Chen & Zhao, 2012).

2. Web services

Web services have suffered from various security issues before the emergence of cloud computing services. Some of these issues have been prevented or reduced, while others are still real security threats (Subashini & Kavitha, 2011). One issue that is not related to cloud computing is the wrapping attack, which is relative to Extensible Markup Language (XML) (Kulkarni et al., 2012; Yan & Yu, 2015; Ramachandra, Iftikhar & Khan, 2017). In this attack, the attacker will re-write a captured massage and inject it into the connection channel to have access to the resources; then, the executing attacker’s requests can be performed through the original valid signature. Also, there are some studies that discuss the SOAPAction spoofing attack, which is simply a fake HTTP header to execute other operations rather than those specified in the original request (Anisetti et al., 2013; Jana et. al, 2017). According to Subashini and Kavitha (2011), using the SOA approach to deliver cloud computing services will increase the probability of the exchanged data being attacked (Subashini & Kavitha, 2011).

3. Web technologies

Allowing users to connect to public websites while they are connected to the cloud will increase the potential for being attacked or infected, as the user could be utilised as a bridge to reach the targeted infrastructure. Having an attack or infection while connected to the cloud will
facilitate spoofing, sneaking and reaching the cloud easily by the attacker. As soon as they are on the victim’s (user) machine, the attacker can exploit the machine and the data that transits through it (Medeiros, Neves & Correia, 2014; Shameli-Sendi et al, 2015; Somani et al, 2017). For example, the attacker can inject malware to collect specific information. In addition, the attacker can use the accessed website itself for capturing the required data (packets), which is known as a malicious website (Bin Mat Nor, Jalil & Manan, 2012). This type of attack is continuously increasing because of the rapid and continuous growth in the number of connected nodes (Shameli-Sendi et al, 2015; Osanaiye, Choo & Dlodlo, 2016; Jana et al, 2017).

4. Connectivity

According to (Singh et al, 2016), having a secure, and stable connection is an essential aspect in the cloud computing environment or similar technologies; such as, Internet of Things (IoT). The connectivity issues in the Internet have varied forms. These issues can be categorised as flood attacks, DoS attacks and bandwidth under-provision. In fact, each type contains a large number of attacks, which are different in target, mechanism and execution, but that are agreed on the same objective, which is stopping the delivered services. A flood attack aims to consume the allocated bandwidth by sending a massive number of requests to the server, which will cause overloading and downing of the delivered service (Karnwal, Sivakumar & Aghila, 2012; Gayatri et al, 2018). A DoS attack aims to stop the availability in various ways, for instance, by disrupting the physical infrastructure, disrupting the configuration or consuming the resources (Liu, 2010; Chonka et al., 2011; Kulkarni et al., 2012; Jana et al, 2017). The bandwidth under-provision issues arise when the hosted bandwidth is larger than the system capabilities (Cisco, 2015).
2.4.3.4 Trust security issues

Trust in the cloud computing environment is an essential issue; cloud customers must trust the cloud service provider and the parties that they have to deal with. In fact, trust varies based on the cloud service provider’s capabilities, methods, experiences and previous customer reviews (Abbadi & Martin, 2011; Monfared & Jaatun, 2011). Practically, trust security issues are driven by two main elements, which are publicity (Internet) and outsource services. The following discussion outlines the security issues that are related to trust.

1. Auditability

Auditability aims to monitor, record and analyse the used cloud situation, which will help to evaluate the provided security level (Marston et al., 2011; Gonzalez et al., 2012; Aceto et al., 2013). Usually, the cloud service provider prohibits the customers from having information regarding security matters. In this situation, a third party can be delegated to audit the delivered service, and that will raise the same issue again as some cloud service providers and customers might not be comfortable with TTP choice for various reasons; for instance, the sensitivity of the consumed service and the exchange of data, or the absence of TTP trust (Catteddu, 2010; Subha & Jayashri, 2017).

2. Human factors

Human factors are the core issue from a trust perspective. A person can destroy the cloud service provider’s reputation by an error and expose the whole cloud system. Also, social engineering has increased recently; threats from this type of attack are varied and dangerous, and a social engineer could attack the customer and the cloud administrator (Thompson, 2013; Dunca, Bratterud & Happe, 2016). All the people who interact with the cloud in a direct or indirect way should be concerned about their password, avoid password sharing, have a strong password and secure any personal information (DoB, address, ID, etc.) that might be used by someone to extract the password and gain illegal access (Marston et al., 2011;
In fact, the experience of the user himself will impact on the efficiency of social engineering attacks, whereby users with a good level of experience will be able to ignore the phishing and social engineering attacks (Cisco, 2015). On the other hand, users with little experience could interact with a phishing attack and use bad practices that might expose the cloud.
2.4.3.5 Computation and storage security issues

According to Wang et al. (2013) storage security is a key factor that impacts on the delivered service’s Quality of Service (QoS). In fact, the concept of outsourcing data storage can raise a lot of concerns from the customer’s perspective; customers would like to know where and how their data will be stored and who will have access to it. Kaaniche & Laurent (2017) argue that by outsourcing data the data owner has to lose control of the data, and hand over the control to the cloud service provider, which will be a critical issue if the cloud service provider is not capable of reaching the expected level of trust, or not capable of protecting the data in an effective way. The TTP has been introduced as a solution to overcome these issues; however, all the previous concerns can also be associated with the third party. Storage and computation security has several aspects, which will be discussed in the following section.

1. Data storage

Data storage has a lot of security issues, one of which is the geographical distribution of the data centres. In order to enhance redundancy and availability, the cloud service providers have to distribute and federate their infrastructure, which will improve redundancy and availability (Subashini & Kavitha, 2011). On the other hand, this distribution might cause security concerns for some customers (Zhou et al., 2010; El Moctar & Konaté, 2017). These customers view the status differently; applying multi-location data storage techniques means an increment in the number of online data stores (the same data in each location), which means increasing the risk of being attacked. According to Wang et al. (2010), there are two reasons why it is not possible for the data owner to check the data integrity, confidentiality and safety. Firstly, there is the physical separation between the data and the owner, who does not know the exact data location; secondly, the cloud service provider should prevent the owner from having access because this threatens all the data that is shared by the same resources. Besides, the data owner might not have the experience to perform such an operation, so a
trusted third party (TTP) could be the optimal solution for the cloud customer and cloud service provider (Xiao & Xiao, 2013; Derfouf, Mimouni & Eleuldj, 2015). According to (El Bouchti, Bahsani & Nahhal, 2016), the data availability will be influenced by the data location, and the geographical distribution of the outsource data will be a critical issue in terms of the geographical location of the infrastructure and the followed regulations and laws in that place; also, portability of the data is a serious problem, especially if there are no restricted rules in terms of the applied cryptographic systems, and the privileges to access such data.

2. Cryptography

In order to guarantee safety, integrity and confidentiality of the protected data, the implemented cryptographic system must be efficient and effective. In fact, the applied cryptography mechanism must be suited to the nature of the encrypted data, strong, updated and implemented in the right way in order to achieve the expected result. Mary & Amalarethinam (2017) and Jin et. al (2018) identify some of the threats that are associated with cryptography, such as insecure cryptography, poor key management and a faulty algorithm. Practically, these threats will not only leave the data without protection, but they can also cause serious consequences; for example, sub-optimal implementation will facilitate malicious attacks, which are known as brute-force attacks. According to Sood (2012) and Ora and Pal (2015), brute-force attacks are rapidly increasing because they are the easiest to exploit.

3. Malware

Injection by malware is a critical issue, but the worst problem is that modern malware can deactivate firewalls, anti-virus and other detection software (Chang & Ramachandran, 2016; Shanmugasundaram, Aswini & Suganya, 2017). Therefore, malware can take control of the infected machine. In some cloud services, there are some data that will be synchronised through various nodes. If the malware takes control of one of these nodes, then it can easily
be spread through the cloud by inserting itself into the synchronised data (Bhandari, Gupta & Das, 2016; Dixit, 2017; El Moctar & Konaté, 2017).

4. Sanitisation

Sanitisation refers to removing specific data from the cloud resources in any way. The sanitisation process is required when the data owner is no longer a cloud service customer. It also helps to reduce the operational costs compared to destroying the used storage and replacing it with a new one (physically). For that, the cloud service providers mention data destruction, which is the last phase of the data lifecycle in the service-level agreement (SLA) (Boampong & Wahsheh, 2012). The shared resources feature is raised here because wiping data from a shared physical unit could wipe the other users’ data as well (Grosse & Upadhyay, 2013; Pearson & Yee, 2013). According to Pearson and Yee (2013), reusing previous customers’ resources, which is known as cloud recycling, will expose the new customers’ data because the re-used resources might still contain some information from the previous user.


2.4.3.6 Access security issues

In traditional IT solutions, having access to the servers’ room or datacentre will be limited and follow restricted rules. Regardless of whether the access method is through the Internet (remote access) or physical (location access), the rules should be applied to enhance security and identify who can have access, why they have access, when they have access, how long they will stay and how they will access it. In the cloud-computing environment, the answers to these questions will be associated with the responsibilities of the users and protecting access in terms of authentication and authorisation. According to (Srinivasan et.al, 2018), the user information, authentication, and granted privileges must be maintained and protected in each different layer, as these aspects vary in each layer in terms of format, and specifications as well.

1. Physical access

Physical access security issues start from the beginning, when the cloud service provider establishes the idea of offering this kind of service. The location of the cloud infrastructure must be secure and safe and help to restrict and monitor the physical access (Zafar et.al, 2017). Indeed, the cloud service and underlying infrastructure can be easily attacked and destroyed if there is no restricted access, and that could be done by leaking information to unauthorised persons, malicious attacks and exposing the cloud service (Grobauer, Walloschek & Stocker, 2011; Gayatri et.al, 2018). According to Yu et al. (2012) and Alassafi et.al (2017), malicious insider attacks have the ability to de-activate all security detection and protection features and make the cloud vulnerable to any attack.

2. Authentication

The user identity should be authenticated before having access and in the cloud-computing environment the authentication process is completed remotely (Sengupta, Kaulgud & Sharma, 2011). Authentication security issues vary between the authentication methods, authentication
attacks and ID management. Usually, the cloud service providers combine the user ID and password to authenticate the user's identity, but they vary in the password techniques. The most common technique is a simple text password, which cannot provide the required level of authentication (Hart, 2009; Takabi, Joshi & Gail-Joon, 2010). There are also graphical password techniques, which are not preferred for the cloud-computing model because of the time and the resources that are consumed via this technique (Dinesha & Agrawal, 2012). In fact, a TTP can be the optimal solution for the large cloud, but not for the small cloud (Dinesha & Agrawal, 2012). Besides, the user ID itself can be attacked simply by entering a valid password, which causes suspicious behaviour and blocks the ID (Grobauer, Walloschek & Stocker, 2011; Halabi & Bellaiche, 2017; Zafar et.al, 2017). According to Alassafi et.al (2017), account hijacking and fake identity are continuous and developed issues that must be solved, as gaining access in an unauthorised way will lead to consequences that expose the consumed infrastructure (Kaura & Lal, 2017).
2.4.4 Systematic Literature Review Findings

In conclusion, the conducted systematic literature review finds that about 80% of the investigated studies do not propose any frameworks or techniques as a solution for the addressed security issues/challenges. In addition, the research methodologies used are vague and unclear in about 24% of the studies. However, the study illustrates that about 40% of the covered studies are reviewing the challenges and issues. In other words, there is a critical need to develop comprehensive and critical studies with a clear methodology that contribute to knowledge and provide a clear understanding of cloud computing security challenges. Further, there is a need to develop more studies that propose solutions/approaches for the identified security issues.

By moving to the identified security issues, the majority of the identified security issues are associated with access and the computational aspects; these two categories have about 50% of the security issues, which have been identified through the conducted study. Consequently, cloud computing access control security issues and cloud computing computation and storage security issues must gain more attention from researchers and developers in both academic and industrial fields. The authentication, applied cryptographic system and storage data issues have the largest share of the identified security issues. As a result, the developed approach will consider these aspects and will aim to overcome these issues as much as we can. Figure 2 - 16 illustrates the study’s outcomes in a taxonomical format and represents the main classifications, the associated categories and the identified issues/challenges. It also shows the share of each category and class of the research outcomes, the detailed identified issues can be seen in Appendix D.
Figure 2 - 16 Identified cloud computing security issues taxonomy

Cloud Computing Security Taxonomy

Software (8%)
- Platform (3.5%)
  - Isolation
  - Accounting
- Frontend (4.5%)
  - Open Source
  - Unauthorised access
  - Direct internet connection
  - Masked-code injection
- Virtualisation (14%)
  - Availability (1%)
  - Management (7%)
    - Large encrypted image
    - Image stealing
    - Virtual Machine monitor
    - Image access
    - Virtual network
    - Escpe attack & Zero-day
  - Mobility (3%)
    - VM cloning
    - VM mobility
  - Malware (3%)
    - Malware injection
    - Rollback function
    - Side-channel and covert-channel attacks
- Internet (14%)
  - Protocols (3%)
  - Web Services (6%)
  - Web Technologies (1%)
  - Connectivity (4%)
    - Network-based cross tenant attacks
    - HTTP and HTTPS
    - XML wrapping attack
    - Unauthorised session management
    - Flood attacks
    - DoS attacks
    - Bandwidth under-provision
- Trust (13%)
  - Auditability (8%)
    - Monitoring
    - Third party
  - Human Factor (5%)
    - Social engineering
    - Password attack
- Access (24%)
  - Physical Access (13%)
  - Authentication (11%)
- Computation and Storage (27%)
  - Data Storage (13%)
  - Cryptography (7%)
  - Malware (5%)
  - Sanitisation (2%)

59
2.5 **Multiple-Clouds Security Approaches**

2.5.1 **Multiple-Clouds as a Solution**

Multiple cloud models have emerged as a promising solution that could be used to overcome the single cloud model’s limitations and obstacles. The developed multiple cloud computing paradigm will help to promote independence, enhance security, increase redundancy, optimise operational costs and improve the quality of delivered services. Cachin, Haas and Vukolic (2010) identify the purpose of developing the multiple cloud model as improving the QoS that used to be delivered via single cloud by distributing dependency, trust, privacy, accessibility and security among multiple clouds. Technically, the existing approaches, which aim to enhance the delivered services in term of availability, security and privacy, will be implemented with no affect/change in terms of the provider frontends. In other words, these approaches will be developed as extensions that are used to enable specific features.

Practically, the multiple cloud model is inspired by the information dispersal concept, which is not new, but is very efficient and very popular. This concept aims to enhance the system’s confidentiality, availability and reliability by distributing the information through more than one location. For example, if we aim to improve the availability of file \( F \), then we need to decompose this file into \( n \) pieces/chunks, whereby \( F = n_1 + n_2 + n_3 + \ldots + n_i \), and \( 1 < i < n \). The most well known solution that has been developed based on this concept is Shamir’s Secret Sharing algorithm (1979). Shamir’s algorithm splits the file \( F \) into \( n \) segments, which will be distributed through \( j \) locations; to recover the distributed data the user must possess/know the pre-defined threshold-mechanism. For instance, to complete the file recovery procedure, the mechanism needs \( f \) segments out of \( n \) otherwise it is not possible.

The rest of this study will explain and discuss the existing multiple-cloud approaches that have been developed to improve the delivered services in terms of security, privacy and availability.
After that, the discussed approaches will be compared and evaluated in order to identify the limitations and highlight the gaps.

2.5.2 State of the Art Multiple-Clouds Security Approaches

Achieving the resiliency and enhancing the QoS of delivered services has become a critical concern for cloud service providers and customers. Due to these developments, and in order to solve other issues such as redundancy and privacy, the multiple-cloud model has emerged. The main principle underpinning this approach is to have more than one cloud infrastructure, which are geographically distributed and interoperable, but which will be managed, configured and controlled by a single centric management. In the cloud computing field, there are a number of approaches that have been developed, inspired by the information dispersal concept, which aims to enhance/guarantee the security, integrity, privacy and availability of the delivered services. In this study, a number of these approaches will be investigated, analysed, assessed, evaluated, explained and discussed. The scope of this critical literature review is restricted to the approaches that have been developed as promising solutions to overcome security issues in the multiple-cloud computing environment. These approaches have been selected after implementing an extensive research process through various databases such as the ACM digital library, ScienceDirect and the IEEE/IET electronic library. The purpose of implementing this critical review is to investigate and evaluate these approaches in order to provide comprehensive and clear understanding of the studied approaches; this will help to identify the research gap and the current field limitations, which will be considered while developing the proposed (new) approach. This study will examine the existing approaches from different aspects. These aspects might vary based on availability; however, the aspects of discussion focus on the approach of main concept, design, maintained security features, multiple-clouds module, targeted customers, and the form of the final product (algorithm, proxy, internal layer or application).
These aspects will help to provide clear understanding of the mechanism of the studied approaches; this will facilitate assessing, evaluating and highlighting the limitations, and will help to overcome these issues and fill the gap via the proposed approach. Next, the existing approaches/solutions for securing the data through multiple-cloud paradigms will be discussed.

2.5.2.1 Byzantine Fault Tolerance (BFT)

The BFT has been recognised as a protocol to be used over distributed systems. Technically, the BFT aims to manage the communication process between the end user and the replicated data/system. Basically, the BFT will guarantee data availability and confidentiality by replicating the data through different clouds (Rabin, 1989). The original BFT algorithm could guarantee availability by having replicas, where the number of faults have occurred. For instance, if there are two faults, the BFT needs seven replicas \((3 \times 2 + 1)\) to guarantee the system’s consistency.

Recently, there have been a number of research projects established aiming to develop solutions to guarantee the consistency of the delivered cloud computing services (Garraghan, Townend & Xu, 2011; Correia et al., 2012). Before reviewing the proposed solutions, we need to define the BFT. Additionally, we have to highlight the expected types of failures that the BFT is able to deal with. For the cloud computing paradigm, there are various faults that could be associated with infrastructure, platform and software layers, and which could be categorised as computation and data failures. Computation failures refer to any kind of failure that might occur to the consumed infrastructure, such as VM unavailability and storage access prevention. Schroeder and Gibson (2010) state that the majority of cloud infrastructure failures occur in memory, hard desk drives and processing units. The data failures refer to any kind of failure that might be caused by corrupted or missing data. This type might be caused by
improper network protocols, which might cause a delay or packet dropping, or overload requests.

In the cloud computing environment, the BFT aims to guarantee the availability of cloud services and data integrity. Technically, it depends on two core parameters, which could be explained as follows:

- **Recovery Point Objective (RPO)**, which refers to the data that is supposed to be lost through the failure (Panneerselvam et al., 2012; Ganesh, Sandhya & Shankar, 2014).
- **Recovery Time Objective (RTO)**, which refers to the time consumed until recovering the cloud from the failure (Panneerselvam et al., 2012; Ganesh, Sandhya & Shankar, 2014).

As the values of RPO and RTO decrease, this refers to the strength and the efficiency of the delivered cloud service. There are a number of BFT approaches that have been developed for cloud computing environments. Some of these approaches are not efficient, make no contribution, or lack critical information. For instance, Padiha and Pedone (2011) propose a storage system (Belisarius) that aims to maintain the confidentiality of stored data. The researchers argue that the provided confidentiality could be maintained by having available storage services, which will definitely guarantee confidentiality, but which is not an efficient solution. The Belisarius requires extra resources, which means extra operational cost in terms of time, communication, management and computation; for example, for a single byzantine cloud, the proposed system needs four available clouds, and six available clouds for two byzantine clouds, in order to maintain confidentiality. In addition, the system requires very high levels of cloud availability. Also, the approach does not refer to the multiple-cloud model.

The approach proposed by Correia et al. (2012) gives us another example of an approach that makes no contribution and lacks critical information. They argue that applying BFT is ‘often inefficient if not executed in a local network’ (Correia et al., 2012) and they propose their
approach to overcome this issue. The proposed approach is supposed to maintain integrity by exchanging the hashes only, instead of the exact data. The first flaw is the absence of the used multiple-cloud model identification, and the other flaw is ignoring critical issues such as data availability and privacy.

On the other hand, there is a project (TCloud) that has been supported by the European Union (Verissimo, Bessani & Pasin, 2012; Bessani et al., 2013b). This project focuses on critical application in terms of availability and security. The TCloud is presented to the end user as a platform that is capable of improving the quality of the delivered service in terms of security and availability. We argue that depending on the application layer to guarantee availability and security of delivered services is not efficient and that proposed solutions also need to cover the platform and infrastructure. The TCloud’s platform relies on several core mechanisms in order to improve the delivered service. For instance, the BTF and proactive recovery mechanism have been used over the higher layer, and multiple-clouds provide a robust infrastructure that is able to increase the redundancy of involved replicas. Additionally, the TCloud gives the end user the required flexibility to configure the TCloud for meeting their requirements and it supports the infrastructure’s heterogeneity. This approach is suitable for enterprises and is supposed to be implemented over intra or federated clouds; it does, however, consume a lot of provided resources because of the integrated services.

Das and Khilar (2013) develop another approach that maintains the cloud availability through maintaining virtualisation. The approach divides the virtualisation responsibilities between the cloud service provider and the customer. The provider will take the onus of the provided infrastructure and the customer is responsible for the software (virtualisation). The hardware failure cannot be recoverable, as failure is restricted to that machine. Additionally, this approach will save the delivered resources by wiping the memory of a recovered machine in order to render it not occupied. However, this feature could lead to a critical issue for services that are expected to have a live version of data.
For real-time cloud services, Malik and Huet (2011) developed an approach that is supposed to guarantee the reliability of the consumed infrastructure. The approach will maintain reliability for each virtual machine after each computational cycle, and that caused by continuous change of resources over time. However, it consumes the provided resources because of the need to collect, analyse and compute the status of resources. The approach can improve reliability by implementing the required tasks over the most reliable resources. It uses proactive mechanisms in order to prevent unreliable resources from performing any task. The reliability level is determined by analysing the targeted resource performance.

It is worth mentioning that all the developed approaches have been builtdesigned to be consumed over federated/intra clouds, whereby the system management will not be assigned to the end user but will be assigned to a service provider or TTP. In addition, the high-level of guaranteed availability is too expensive in terms of required resources, exchange data and management. All this makes the developed BFT approaches suitable solutions for enterprises.

### 2.5.2.2 High Availability and Integrity Layer (HAIL)

Bowers, Juels and Oprea (2009a) propose an approach that they called High Availability and Integrity Layer (HAIL). The proposed approach has been developed as an updated version of Redundant Arrays of Inexpensive Disks (RAID). Technically, RAID has been developed in order to provide redundancy through hard drives, while HAIL aims to provide it across the cloud storage. The HAIL approach has been introduced as ‘a distributed cryptographic system that allows a set of servers to prove to a client that a stored file is intact and retrievable’ (Bowers, Juels & Oprea, 2009a). HAIL has been designed to maintain the availability and integrity of stored data through increasing the data redundancy over independent clouds. Basically, HAIL was inspired by a tool called Proof of Retrievability (POR) (Juels & Kaliski JR, 2007; Bowers, Juels & Oprea, 2009b), which has been developed in order to prove to the client that the stored data is recoverable. This procedure is done by relying on a single TTP
(verifier). Before moving to the HAIL’s design, we need to list the main features that HAIL is capable of:

- A high level of integrity protection for the static stored data.
- Saving the consumed bandwidth for internal communication, compared to POR, by reducing the overhead size of the exchanged packets.
- Proving the validity of stored data to the client.
- Preventing corrupt clouds or data from infecting other system components.

Firstly, the targeted file is distributed over several independent clouds with a redundancy. In order to validate the distributed file integrity, the data owner can check the integrity of stored data. If corrupt data has been detected, then the system replaces it with an uncorrupted one, which guarantees the health of recovered data, whilst having a number of replicas enhances the data availability. According to Bowers, Juels & Oprea (2009a), the architecture of HAIL could be divided into three core layers:

- Dispersal code:

  The dispersal code is responsible for data segmentation and distribution across the different clouds. Additionally, the developers provide an Integrity Protected Error Correcting Code (IP-ECC), which works as an error correcting code and as a corruption-resilient message authentication code.

- Server code:

  The stored data segments are encoded by an error-correcting code, which provides the required protection against the corruption that occurs when the data fail to pass the integrity check.
- Aggregation code:

The aggregation code is responsible for recovering the original file by compressing the distributed data. In addition, it compresses the message authentication codes for each data segment into one message authentication code; plus, it passes the integrity variation if, and only if, all the compressed message authentication codes meet the integrity check. The integrity check values will be implemented across the clouds and no values are stored in the end user machine.

Practically, the operational costs of this approach are based on two main factors: 1) the file size, which impacts on the consumed bandwidth, storage, computation and time; and 2) the number of involved clouds. Precisely, the dispersal code costs depend on the number of involved clouds and the costs of the server code relies on the size of the targeted file.

One of HAIL’s core disadvantages is its inability to provide a live data version, which means it is limited only to static files (Wang et al., 2012; Bessani et al., 2013a). Further, it requires a code execution across the provided clouds to prevent any kind of latency that might be caused by executing the code across other platforms (Bessani et al., 2013a). Also, the segments are distributed across the clouds in the same order for each file, which could be considered a vulnerability, because the lack of randomisation will facilitate tracing the data segments. The client will not check the integrity for the whole segments, but the integrity validation process will be implemented over specific segments, which could cause a creeping-corruption attack over time. Also, HAIL does not support infrastructure heterogeneity, and it is supposed to be run over federated or intra cloud paradigms. Actually, HAIL is consumed as an integrated layer through the provided infrastructure and it is not provided to the end user as an application or as a service to consume. In addition, HAIL is limited to enhancing and guaranteeing the integrity and availability of customer data; it is unable to guarantee confidentiality of customer
data, which is a critical issue in terms of data security (Abu-Libdeh, Princehouse & Weatherspoon, 2010; Bessani et al., 2013a; Wang et al., 2013).

2.5.2.3 Redundant Array of Cloud Storage (RACS)

Abu-Libdeh, Princehouse and Weatherspoon (2010) developed the Redundant Array of Cloud Storage (RACS) as a proxy for cloud storage solutions; it aims to enhance the availability of stored data and avoid vendor lock-in and its connected issues, which is achieved through dispersing the targeted file across a number of cloud storage providers. The RACS system utilises a RAID5-like technique to provide a data replication system via multiple-clouds and to ensure the efficacy and availability in an economical way. Thomasian and Menon (1997) define RAID5 as ‘a set of disk drives and a controller, which can automatically recover data when one drive in the set fails’. Theoretically, stripping and distributing data through multiple-cloud providers will increase the operational costs, although applying the proper stripping mechanism might mitigate these additional costs. Applying this mechanism will also improve the agility and mobility, and prevent data outage incidents.

However, relying on a single proxy paradigm will cause a bottleneck issue that will occur because all exchange data need to pass through that a proxy to both encode and decode. For that, Abu-Libdeh et al. (2010) argue that running more than one RACS proxy, which must be connected together, will overcome this issue and enhance the throughput.

On the other hand, the RACS system has other limitations; for instance, it only focuses on enhancing data availability and avoiding vendor lock-in issues and does not give enough attention to other security aspects such as data consistency, confidentiality and integrity (Cachin, Hass & Vukolic, 2010). The RACS system might suffer from high levels of latencies because the conducted write operation will not be terminated until all of the active write processes for all the involved cloud storage is completed (Bermbach et al., 2011).
2.5.2.4 InterCloud Storage (IC Storage)

InterCloud Storage (IC Storage) is similar to the RACS approach and could be considered as an improvement of that solution (Cachin, Haas & Vukolic, 2010). IC Storage was developed due to the need to guarantee the dependability of the used clouds and to overcome asynchrony issues by utilising the toleration of clients’ protocol failures developed by Cachin, Haas and Vukolic (2010) to be used in terms of distributed storage.

Technically, the Intercloud approach expands the cloud service provider layer from a single paradigm to a multiple-cloud paradigm; consequently, and in order to provide the requested elasticity, the used proxies are also centric and client-side. These client layers are capable of being turned off or on in an individual way in order to deliver a service that meets the end-users’ requirements in a dependent way. The first layer is the confidentiality, which is maintained by applying a symmetric key encryption to the targeted data. The integrity layer preserves the data integrity through protecting against un-authorised alteration for single and multiple clients. Un-authorised data modification protection is maintained via the hash tree for the single client, and through applying the public key infrastructure for the multiple clients; for that, developers use fork-linearizability (Mazieres & Shasha, 2002) to guarantee detecting the data modification via the multiple clients. The last layer is the reliability and consistency layer, which has a set of fault tolerance protocols to meet the clients’ requirements. These protocols are used to distribute the data to the involved cloud service providers.

Similar to the majority of the developed multi-cloud storage approaches experience of limitations, these systems, including IC Storage, are only capable of tolerating the cloud outages and are not able to tolerate malicious or arbitrary behaviour in the multi-cloud system (Dobre, Viotti & Vukolić, 2014). IC Storage is better than the RACS and HAIL approaches in terms of dependability; this is achieved through two core points: 1) utilising asynchronous
fault-tolerant client-driven storage protocols; and 2) maintaining the stored data confidentiality, integrity and consistency (Chockler et al., 2009; Vukolić, 2010).

2.5.2.5 Tahoe- The Least-Authority File system (Tahoe-LAFS)

Tahoe-Least-Authority File System (Tahoe-LAFS) has been developed by Wilcox-O'Hearn & Warner (2008) as a solution for securing distributed storage systems in order to increase the availability of stored data. In addition, it permits users to validate the integrity of the stored data and access control. Tahoe-LAFS works as a gateway to distribute the data through the available cloud storage. Technically, this approach distributes the targeted data through ten cloud storages, and it is capable of retrieving the data via any three data chunks alone; this, according to Itkis et al. (2016), is more reliable than RAID inspired solutions. According to Seiger, Groß and Schill (2011), asymmetric and symmetric cryptographic solutions will improve the delivered data security by utilising the information dispersal concept.

The operations of this approach could be simplified into three core steps: 1) encrypt the data; 2) erase the data; and 3) disperse the data through the multiple-clouds. Furthermore, Tahoe-LAFS is an independent approach that has the ability to preserve data security and privacy even when the cloud services are unavailable. Also, the approach could maintain the access control aspects based on the granted capabilities (Read, Read and Write). In addition, combining the data version with the segment metadata will guarantee the freshness of the targeted segment (Bedi, Singh & Gupta, 2016). This approach, however, might need to implement some computations among the consumed clouds; although code execution might introduce other issues such as delay and data flow interruption. According to Popa et al. (2011), the Tahoe-LAFS approach is not able to either prove any occurred data breach or detect that breach.
2.5.2.6 CloudProof

Popa et al. (2011) propose an approach for cloud storage called CloudProof. This cryptographic approach aims to guarantee the security of stored data by focusing on the auditability aspects. Sood (2012) claims that relying on some cryptographic tools will facilitate detecting any unexpected attitude and prove it. In addition, CloudProof maintains the confidentiality, consistency and data freshness, and it allows all the involved parties to confirm any detected breach (Kamara, Papamanthou & Roeder, 2011). On the other hand, this approach requires some code execution, which is implemented and outsourced through the consumed cloud storage. Offloading a lot of tasks to the used clouds means increasing the operational costs in terms of required resources and consumed communications.

2.5.2.7 IRIS

IRIS is a cloud-oriented authenticated file system approach developed by Stefanov et al. (2012); it aims to enhance the authentication level, and it achieves this by providing efficient and genuine integrity verification to customers. The integrity of the stored data protects the data against corruption and unwanted modification. Furthermore, the consumed cloud services are audited by the Iris approach. Customers conduct this auditing in order to check the availability and correctness of the consumed services. Technically, IRIS relies on PORs, which have been developed in order to prove to the client that the stored data is recoverable (Ardagna et al., 2015).

Further, IRIS relies on two levels of authentication paradigm; the upper level is a Merkle tree (Carminati, 2009), which checks the file-block version integrity; for the lower level, two aspects are maintained: the data freshness is guaranteed via the file-block version, and the data integrity by the message authentication code. IRIS supports multi-writers but it has deficiencies in terms of access management and granted privileges. Also, it is costly in terms
of consumed communications. However, Wang et al. (2013) argue that IRIS is not capable of supporting large numbers of customers, which could be considered as an obstacle for enterprises, which are its targeted customers.

2.5.2.8 DepSky

DepSky is a system that has been developed across distributed cloud storage, which aims to assure the availability, confidentiality and integrity of the stored data (Rocha, Abreu & Correia, 2011; Rocha & Correia, 2011; Bessani et al., 2013a; Bessani et al., 2013b). This approach is limited only to the cloud storage services and there are two versions: DepSky-Availability and DepSky-Confidentiality and Availability. However, the DepSKY approach demonstrates some popularity because of the provided redundancy, and the capability for supporting multi-readers/multi-writers features for the stored data.

Practically, the stored data is spread over different cloud storage from different service providers, which will help to prevent vendor lock-in issues, and guarantee data integrity and availability; however, while the operational costs for DepSky are higher than relying on a single cloud, this is acceptable in terms of the delivered performance (Verissimo, Bessani & Pasin, 2012).

Rocha & Correia (2011) argue that DepSky guarantees confidentiality by utilising proper encryption schema, which are limited to the stored data that are owned by users and do not protect any data from the service provider side. On the other hand, DepSky shows a degree of inefficiency for some applications because it needs a large number of communications between all the involved data writers, which could be very expensive for some applications.
2.5.3 Critical Literature Review Findings

The findings from the previous critical literature review will be presented in several core points, that represent the limitations and obstacles that have been defined through this study. These points will help to define the difficulties, issues, and obstacles that are supposed to be investigated, solved, and overcome by researchers and developers. Table 2 - 4 illustrates a comparison between the investigated approaches, and shows how the proposed approach will distinguish from these approaches; the core points are listed as follows:

- **The approach environment:**
  The existing approaches are expected to be implemented in a cloud computing environment. However, some of these approaches have been developed to be consumed over distributed computing environment, such as the BTF and the Tahoe-LAFS approaches. The variation in the environments and their capabilities, features, and characteristics will impact on the effectiveness and approach performance in general. For example, Tahoe-LAFS needs 10 cloud storages in order to preserve availability, which is expensive in terms of system requirements, and the consumed resources. Moreover, off-site code execution is acceptable over distributed computing applications, but it is not saved in cloud computing applications, and this issue has been discussed and explained earlier.

- **The complexity:**
  The majority of the investigated approaches are too complex in terms of design and functions. This complexity led to several issues that vary between performance and requirements issues. For example, the RACS approach suffers from a bottleneck issue which emerges due to the need to exchange the encode and decode data through the same proxy, in order to check and validate the availability of the stored data. On the other hand, BFT and IRIS need a lot of resources and well experienced staff in order to configure and operate these approaches, and this causes complexity.
- **The operational costs:**

  The operational costs of all the investigated approaches are high, caused by several points, such as the used multiple-cloud computing paradigm, and the approach needs. These approaches might be useful for enterprises that are capable of deploying and operating such approaches in terms of trained staff, infrastructure, and resources. The Dep-Sky approach has been developed to preserve the data confidentiality and availability, one of the drawbacks of this approach is the need for a large number of communications, which means increments in the consumed resources which leads to increasing the operational costs by default. Another example is the IRIS approach, which is concerned with authentication through the system. IRIS is a costly approach in terms of the consumed communications.

- **The provided flexibility:**

  All the investigated approaches are not flexible in terms of the service selection, portability, and preferences configuration, apart from the Dep-Sky approach which allows choice of the required service (availability, confidentiality, or both). For example, the IC storage approach will maintain the data confidentiality, integrity, and reliability, but for some applications that do not need to maintain all these three aspects for any purpose, they are not able to do that. The absence of flexibility will lead to increasing the operational costs and executing unwanted functions, and that will impact directly on the performance. In addition, absence of flexibility might cause conflict with other utilised approaches.

- **The utilised multiple-cloud computing paradigms:**

  All the studied security approaches have been developed to be consumed over multiple-cloud computing paradigms that are not suitable to be consumed by individuals. In addition, some of these approaches have some issues and obstacles that have been discussed earlier. The used paradigms vary between federated and intra cloud computing paradigms, which are suitable for enterprises, which means the
developed approach cannot be consumed by individuals for several reasons. Both of these multiple-clouds models are suitable for large and medium enterprises, which is caused by the characteristics of these models (connection method, prior-agreement and the coupled level). For example, the used multiple-cloud computing paradigms are too expensive for individuals in terms of the required resources and capabilities. Moreover, the needs of the utilised infrastructure in terms of prior agreement, the collaboration level, and management location are not set up for providing such services to individuals.

- **Expertness & End user:**

  As we mentioned earlier, all the investigated approaches are complex and utilised paradigms are suitable for enterprises. All these approaches need a high level of knowledge and well-trained ICT staff and users as well, in order to configure, deploy, operate, maintain, and use; this kind of requirement will increase the operational costs. In addition, all these approaches need the individuals to be expert in order to consume them, if we discard the other obstacles. Moreover, failing to meet the need level of knowledge for developers and end users will introduce other issues that are relevant to performance and security, which will expose the system. None of the studied approaches can be operated or consumed by individuals or even by small enterprises, as the characteristics of the used cloud model, and the mechanism of each approach, need specific requirements in terms of resources and capabilities that cannot be accommodated by individuals.

- **Preserved security aspects:**

  All the developed approaches are not able to deliver the optimal solution that is able to solve/consider the majority of the identified security aspects; this means the chosen approach needs to be combined with other approaches to deliver the required security level. In other words, the total operational costs will be too expensive due to the need to configure, manage, maintain and consume other approaches in addition to the main
one. Moreover, combining approaches might impact on the performance and produce other issues that might expose the stored data. Also, these approaches might not be compatible between each other and that could led to duplicating data, exchanging non-encrypted data, unexpected termination, or occupying system resources.

**Approach Format:**

The developed approaches vary between interconnection protocol, internal layer or general algorithm. All the existing approaches increase the complexity of the consumed system and require a very high level of expertise to be configured and implemented. There is no approach that could be provided to customers and could be consumed as a service that is capable of enhancing the level of delivered security, integrity, authenticity and redundancy with the minimum required level of experience, regardless of whether or not the customer is an enterprise or a single user. Moreover, the existing approaches do not support portability, which is a critical issue in case of disaster, or the need for changing the machine for any purpose. Also, these formats are not friendly for the end user, and lack of experience will impact on the performance and expose the data.

To sum up, there is a need for an approach that could be installed, configured, and consumed by the individual in a simple way, without any concern about the multiple-clouds system architecture, agreements level, or cloud services dependencies. Besides that, the approach must be delivered in executable format, be portable and provide a high level of flexibility to the users, and must be consumed as a comprehensive approach that is capable of preserving several security aspects: for instance, privacy, confidentiality, availability, and integrity. In addition, the dependencies between the consumed cloud services must be minimised in order to prevent any impact on the system performance and availability, and that could be achieved by relying on a multi-clouds services paradigm. Also, the end-user must be able to consume the delivered approach over the preferred environment.
<table>
<thead>
<tr>
<th>Approach</th>
<th>Availability</th>
<th>Authentication</th>
<th>Integrity</th>
<th>Confidentiality</th>
<th>Cost</th>
<th>Computing Environment</th>
<th>Multiple-cloud paradigm</th>
<th>Configuration</th>
<th>Portability</th>
<th>Auditing</th>
<th>Used by</th>
<th>Specialty</th>
</tr>
</thead>
<tbody>
<tr>
<td>BFT</td>
<td>✓</td>
<td>✓</td>
<td>High</td>
<td>Distributed / cloud</td>
<td>Federated / Intra</td>
<td>Difficult</td>
<td>x</td>
<td>x</td>
<td>Enterprise</td>
<td>Needed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HAIL</td>
<td>✓</td>
<td>✓</td>
<td>Medium</td>
<td>Cloud</td>
<td>Federated</td>
<td>Medium</td>
<td>x</td>
<td>✓</td>
<td>Enterprise</td>
<td>Needed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RACS</td>
<td>✓</td>
<td></td>
<td>High</td>
<td>Cloud</td>
<td>Federated</td>
<td>Difficult</td>
<td>x</td>
<td>x</td>
<td>Enterprise</td>
<td>Needed</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IC Storage</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>High</td>
<td>Cloud</td>
<td>Intra</td>
<td>Difficult</td>
<td>x</td>
<td>x</td>
<td>Enterprise</td>
<td>Needed</td>
<td></td>
</tr>
<tr>
<td>Tahoe-LAFS</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>High</td>
<td>Distributed / Cloud</td>
<td>Federated</td>
<td>Difficult</td>
<td>x</td>
<td>x</td>
<td>Enterprise</td>
<td>Needed</td>
<td></td>
</tr>
<tr>
<td>CloudProof</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>High</td>
<td>Cloud</td>
<td>Federated</td>
<td>Difficult</td>
<td>x</td>
<td>✓</td>
<td>Enterprise</td>
<td>Needed</td>
<td></td>
</tr>
<tr>
<td>IRIS</td>
<td>✓</td>
<td></td>
<td></td>
<td>High</td>
<td>Cloud</td>
<td>Federated</td>
<td>Difficult</td>
<td>x</td>
<td>✓</td>
<td>Enterprise</td>
<td>Needed</td>
<td></td>
</tr>
<tr>
<td>Dep-Sky</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>High</td>
<td>Cloud</td>
<td>Federated</td>
<td>Medium</td>
<td>x</td>
<td>x</td>
<td>Enterprise</td>
<td>Needed</td>
<td></td>
</tr>
<tr>
<td>Our approach</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>Low</td>
<td>Cloud</td>
<td>multi</td>
<td>Easy</td>
<td>✓</td>
<td>✓</td>
<td>Individual</td>
<td>Not needed</td>
<td></td>
</tr>
</tbody>
</table>

Table 2 - 4 Multiple cloud security approaches comparison
2.6 **Summary**

The chapter was divided into four main parts. The first part reviewed the cloud computing, and presented the fundamental information in terms of definition, capabilities, features, deploy, and delivery model.

One of this study contributions was Identifying criteria, categorising, and comparing existing multiple-cloud models that can be used to evaluate these models to select the most appropriate model for the approach, and that was the third part of this chapter. The multiple-clouds computing was discussed in detail, including the motivations for developing the paradigm. In addition, the existing multiple-cloud computing paradigms were categorised, and the characteristics, benefits, and limitations were illustrated. The studied multiple-cloud paradigms were compared in terms of the operational costs, availability, and needed agreements and infrastructure.

The next section presented, analysed, and discussed the literature review phase; two literature reviews have been presented. The first literature review was a systematic literature review that aimed to provide an updated study that illustrates the security issues, challenges and vulnerabilities associated with cloud computing, and insight into how these issues have been addressed by the proposed research. This systematic literature review was defined as contribution to the body of knowledge, and conducted to provide a clear understanding about the nature and the source of the defined security issues. The conducted investigation highlighted the points that are supposed to gain more attention from the researchers; it also provided a clear understanding of how these issues/threats could happen and how they interact with other elements in the system environment. In addition, the implemented review showed the points that need to be considered and taken into account through the solution development process.
The second literature review was a critical literature review that investigated the existing multiple-clouds security approaches that have been developed as solutions and are expected to enhance the security of the delivered services and consumed infrastructure. This study represented one of the defined contributions for this study, which aimed to provide a comprehensive and critical study of existing approaches for securing data in multiple-cloud environment, identifying their capabilities, but also areas for improvement that have been applied in the context of this research, and using this to design and develop an approach that bridges the existing gaps. The next chapter discusses the developed approach in terms of research gap, novelty, approach concept, and design.
Chapter III. DEVELOPED APPROACH

This chapter aims to address the gap identified from the review of existing research and use the adopted research method to provide an accurate and comprehensive explanation of the development of the proposed solution.

The research gap and the aims, objectives and scope of the proposed method are defined. The need for our approach, what differentiates it from previous approaches, and the targeted and expected end users are discussed. The proposed approach is presented and discussed in terms of its originality, expectations, capabilities and contribution. The core concepts of the proposed approach are introduced to provide a better understanding and explanation of, and justification for, the employed techniques, mechanisms, and methods, which help develop an approach that achieves the defined objectives and meets expectations. The proposed approach is explained in terms of its design, which involves discussing the structure, components and procedures. The preserved security features of the developed approach are discussed, and how these features and aspects can be preserved and maintained is explained. The applied encryption algorithm is illustrated. Other sections of this chapter are concerned with the prototyping of the developed approach. The core points covered can be summarised in three areas of focus: the importance of the prototype model followed by the prototype requirements and implementation environment, and the developed Android application.
3.1 **Research Gap**

Defining the research gap, and identifying the need for developing an approach capable of protecting end-user data in a multiple-cloud environment was stated as one of this study contributions. The literature review produced a list of critical outcomes that require attention to provide efficient and practical solutions to prevent, avoid, or overcome the defined issues. The S.L.R finds that majority of cloud computing security issues are associated with data storage, applied cryptography, and applied authentication. Several studies identified outsourcing data as one of the core concerns that prevents both the enterprise and the individual from migrating to cloud computing solutions (Hashizume et al., 2013; Srinivasan, 2013; Ali, Khan & Vasilakos, 2015). The end user expects that the stored data must be handled in ways that guarantee its safety and integrity. In most cases the integrity, confidentiality, and data location cannot be validated via the end user (data owner), which is caused by not having physical access to the area where the data is stored, and by denying the validation by the service providers for several reasons, such as the privacy of other data and the shared infrastructure (Wang et al., 2010; Xiao & Xiao, 2013).

Having access to the stored data, and to any kind of log, record, or data set that might be used to get access to the stored data, is critical for the data owner. Maintaining those aspects is the responsibility of the service provider and the data owner. The service provider must prevent any unauthorised attempt that might expose the stored data, which could be achieved by employing an efficient identity mechanism and verifying granted privileges. To make the capture and exchange of data useless to an attacker, sniffing the network must be considered by using the proper encryption, masking, and other mechanisms that provide efficient identity (ID) management. These aspects require more attention from researchers and developers to find, develop or propose solutions. However, other issues should not be ignored or omitted, including cloud computing security, and inherited technology issues, such as the network and
internet. The proposed approach guarantees that stored data cannot be read or modified by unauthorised parties; even if attackers get the data, they will not be able to retrieve it, as they need to have the last segment, which will be stored in the user’s machine, and the encryption key, in order to retrieve the entire data.

Previous studies and evaluations of existing multiple-cloud security approaches raise several issues. First, they were based on multiple-cloud computing paradigms suitable for large and medium enterprises and not suitable for individuals or small enterprises; the multiple-cloud computing paradigms have been discussed earlier. The operational costs of deployment, development, configuration, training, management, and maintenance for these approaches do not match the needs or capabilities of small enterprises and individual users. The approaches are complex and utilising some of them as an individual might introduce other issues, especially those that need code execution through the consumed cloud infrastructure. Such an approach could threaten and expose the whole infrastructure if there are no good practice protocols or expertise. Targeting individuals and the capability of maintaining security aspects are absent from such approaches.

All the existing approaches expect the end user to be expert in cloud computing and security, which is critical. For those users who looking for protection in terms of integrity, confidentiality, and safety, they cannot rely on a single approach to maintain these aspects. For instance, the IRIS approach was developed to improve authentication, and relies on authentication to prove the recoverability and availability of the stored data, but the customer must employ other mechanisms to maintain confidentiality, which increases the operational costs and that is not suitable for all customers. Finally, most of these approaches are not portable or mobile, which may not be suitable for customers who want to use different platforms. The critical investigation and analysing of previous approaches is a significant step in the new approach development process, as it identifies and highlights several issues that should be considered in order to avoid them in the new approach; these limitations are summarised in Table 3 - 1.
<table>
<thead>
<tr>
<th>parameter</th>
<th>Existing approaches' limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code execution</td>
<td>Executing a code over the consumed cloud service is not recommended by some applications, as it is possible to inject malware attacks. In addition, lack of knowledge and following bad practice might lead to unwanted execution and termination as well.</td>
</tr>
<tr>
<td>Usability</td>
<td>All the investigated approaches are not easy to deploy, configure, and operate. These approaches must be installed, configured, and maintained by experts only, to preserve security aspects and constancy as well. Also, the end user needs training to be capable of consuming these services safely.</td>
</tr>
<tr>
<td>Multiple-cloud paradigms</td>
<td>The existing approaches rely on multiple-cloud paradigm that need prior-agreement between the collaborated cloud services, which might raise some issues in terms of management, synchronisation, availability, isolation, cost, and other party involvement.</td>
</tr>
<tr>
<td>Portability</td>
<td>One of the major drawbacks of existing approaches is the lack of portability. The majority of existing approaches will not be delivered as executable files, which is an issue that might lead to threat and attack as well.</td>
</tr>
<tr>
<td>Environment</td>
<td>There is no diversity in the supported platform and most of the existing approaches are not portable and need machines with specific capabilities to be implemented. The absence of supporting smart devices must be considered, as the market forecasts show rapid and continuous increment in consuming cloud services through these platforms.</td>
</tr>
<tr>
<td>Performance</td>
<td>The high operational cost of existing approaches and the absence of a comprehensive approach that is capable of preserving a set of security aspects in one solution could be considered a problem. This problem will enforce the user to utilise other approaches to maintain security, and that will produce a set of critical issues in terms of compatibility, flexibility, third party involvement, and additional operational costs.</td>
</tr>
<tr>
<td>Cost</td>
<td>The high operational costs of existing approaches come from the used multiple-cloud computing paradigm, and that influences the ownership costs and the needed connections between the collaborated cloud services. In addition, the need for training, expert users, and the huge amount of calculations that are implemented while the approach is running consume resources in terms of time, power, CPU, and storage.</td>
</tr>
</tbody>
</table>

Table 3 - 1 Existing approaches' limitations

The core contribution for this study was developing an approach that can maintain data integrity, confidentiality, and safety through proper multiple-cloud computing paradigms that can be consumed by individuals, reduce operational costs and minimise the requested experience level. The proposed approach is portable to provide the end user with the ability to consume the delivered service via the most suitable platform. Our approach is for cloud computing storage services, and guarantees the safety of the shared infrastructure by preventing code implementation over the cloud, and employing authentication mechanisms that have been developed by cloud storage service providers. In addition, the proposed approach allows the end user the ability to configure the requested level of security, based on their needs. The approach reduces operational costs by minimising the consumed resources, such as the bandwidth, and assures the stored data protection in case of provider security breaches, or while data is being exchanged.
3.2 Why This Approach

Before establishing an explanation and discussion of the developed approach, there are several points that need to be discussed, in order to clarify various aspects that help to highlight the approach’s importance, the difference from previous approaches and what is new in this approach. These aspects are listed as follows:

- **Why we need this approach?**

From the conducted exhaustive literature investigation, the previous approaches that have been studied and evaluated, are expensive in terms of operational costs and the needed infrastructure. The high operational cost comes from two core points: the used multiple-cloud paradigm, and the approach requirements. All the existing approaches utilised multiple-cloud paradigms that need prior agreement between the consumed clouds, and high level of dependency between these clouds. In addition, such paradigms need to be fully synchronised, and have a real time status update, which means the need for stable, fast, and appropriate connection to work is expected. Moreover, these paradigms are too complex in terms of design, configuration, procedures, and functions. This complicity requires experts to install, deploy, operate, and maintain, and have enough resources to complete the complex tasks and functions. Generally, decreasing the operation cost is needed and becomes a critical point for individuals and small enterprises. Our approach will help to reduce these costs and present an affordable solution that can be consumed by individuals. Practically, this will be achieved by using a multi-cloud libraries paradigm, and avoiding complicity in design, procedure, and functions.

As we discussed earlier, the existing approaches are too complex. The high level of complicity means these approaches must be operated and maintained by experts. This need of a high level of experience might lead to exposing the system, if those experts are not qualified
enough, or if they fail to follow good practice while configuring and interacting with the system. Our approach attempts to avoid such mistakes by simplifying the procedures and functions, and making it simple for a non-expert user, which will facilitate the consumption of the approach by a wide spectrum of users, and avoid bad practice mistakes by computing and performing all the procedures and functions internally in an automatic way.

The investigated approaches are supposed to be consumed over multiple-cloud computing paradigms that need an agreement between these clouds in advance, and that means the customers cannot change their cloud service provider, and that is a critical issue in terms of security, availability, and QoS. In order to overcome this issue, the developed approach supports cloud service provider diversity, and the end user will be able to include and exclude the cloud service provider based on end user preferences.

The majority of previous approaches are limited to specific platforms, and some of these approaches are like internal protocols that need specific platforms to be configured and implemented. This point might increase the operational costs in terms of consumed resources and the qualification of the human factor. Moreover, flexibility in terms of platform is absent and the end user must perform the wanted tasks through a specific platform. Through the proposed approach the end user will be able to operate and consume the system through their preferred platform, as our approach supports platform diversity, and allows the end user to consume it over the appropriate platform: for instance, Windows, Mac OS, Android, and IOS.

All the existing approaches are not portable, and that might be a drawback in terms of usability and security. For some applications, supporting portability is important, and especially for field tasks, when the end users need to protect the data as soon as they receive it. The developed approach will try to overcome this issue by supporting portability, and increasing the flexibility of the delivered service. In addition, the supporting of portability will facilitate the system’s sustainability, by allowing the end user to change the consumed machine and start performing
the needed task over a new machine, if the old machine is not safe or unavailable for any reason. Now, after discussing why our approach is important, we need to highlight the new features and what will distinguish this approach from the others.

**Why is it different?**

According to the conducted literature investigation, the existing approaches rely on multiple-clouds paradigms that need a mediator (broker) to update the cloud service status. For some paradigms, losing the broker availability will influence the system performance and availability. Moreover, for some applications dealing with sensitive information relying on TTP is not appropriate. The proposed approach prevents TTP and mediator issues by using a multi-cloud libraries paradigm. As we discussed earlier, no approach has been developed to be consumed by individuals. Besides that, the market forecasts show a continuous increment in cloud computing markets and increment in consuming the cloud computing service by individuals especially over smart devices. This is why our approach targets individuals.

By moving the used multi-cloud paradigm, no approach has attempted to take advantage of the multi-cloud libraries paradigm. As our approach targets individuals, and tries to reduce the operational costs, the multi-cloud libraries paradigm has been chosen as the most appropriate paradigm capable of delivering the wanted performance and meeting the designed approach requirements. The studied approaches attempt to maintain specific security aspects, and the end user must combine more than one approach to enhance the security. However, the compatibility between such approaches could be an issue in terms of complicity and security. The proposed approach is to preserve several security aspects (these will be discussed later) in one approach, which will help to avoid the compatibility issue, enhance delivery security, and customise the preserved security aspects based on the end user requirements.

As we discussed before, the complexity might lead to exposing the system. As a result, we aim to avoid unwanted complicity and deliver the approach in an executable format, that
minimises the interaction between end user and system, and avoids bad practice problems. Finally, portability is one of the features that will be supported. The developed approach will enhance the end user flexibility in terms of portability, and the diversity of the supported platforms as well. Technically, this feature will help to avoid locking-in the user to restricted machines or platforms, and consider low budget consumer capabilities.

In addition, the developed approach prevents outsource code execution under any substation and that to prevent any issue that might happened during the execution; such as, injection attack or bad practice. All the encryption, decryption, verification process will be implemented on-site (end-user machine) and that helps to prevent the earlier issues, and save the provided resources and capabilities in terms of bandwidth and power.

The developed approach prevents the dependency between the consumed cloud services, the used cloud service will be 100% isolated from each other, and there is no data will be exchange between these cloud service. That type of isolation avoids exposing the protected data in case of data leak happened. Moreover, preventing TTP involvement means that there is no party authorised to access the distributed data, as the data owner owns the data, and the key to decrypt the protected data. Besides that, the proposed approach was developed to be consumed over any platform without any implication on the consumed infrastructure, which improve the end-user choices in terms of the used platform, and the consumed cloud services.

- **Why is it better?**

The developed approach is better than existing approaches in various aspects, these aspects vary between the performance perspectives, security perspectives, and requirements perspectives. From requirements perspectives, the developed approach utilises the multi-cloud libraries paradigm, which needs no prior agreement between the consumed cloud storage, and can be installed and consumed without any change to the consumed infrastructure. In addition, the level of simplicity, the utilised multi-cloud libraries paradigm and
approach format (executable file) help to avoid the need for a high level of experience in order to install, configure, consume, and maintain the delivered approach. In addition, the end-user can move from cloud service provider to another one that more suitable for the end-user needs with no implication over the used infrastructure. Also, the approach does not require any kind of experience in order to consumed, and this point was addressed in order to reduce the operational cost, and avoid issues that might happened due to bad practice.

From security perspectives, the developed approach prevents any third-party involvements and the data owner, and data owner only, is able to access and read the protected data. The utilised cloud storage services are isolated from each other and no data will be exchanged between these providers. In addition, the developed approach prevents the need to combine more than one approach in order to enhance security, which helps to reduce the operational cost, decrease the needed level of knowledge, and prevent any threats that might occur due to conflict between approaches. Moreover, no code will be executed over the data (outsourced), and all the exchange data will be encrypted before leaving the end user machine. The developed approach guarantees the safety of the shared infrastructure by preventing code implementation over the cloud, and employing authentication mechanisms that have been developed by cloud storage service providers. In addition, the proposed approach allows the end user the ability to configure the requested level of security, based on their needs. The approach reduces operational costs by minimising the consumed resources, such as the bandwidth, and assures the stored data protection in case of provider security breaches, or while data is being exchanged.

In addition, the developed approach supports portability, which is one of the critical limitations that was identified earlier. The developed approach allows the end user to run and consume the provided service through the preferred platform, and is presented to the end user in an executable format that needs no high level of knowledge to operate at all. In addition, the approach allows the database of protected data to be moved from one machine to another in
order to enhance the portability and flexibility level. Moreover, the developed approach provides a high level of flexibility to the end user in terms of selecting the cloud storage providers, the preserved security features, and the used encryption algorithms.

Practically, the developed approach helps to save the resources to be consumed and occupied with tasks. Also, the multi-cloud libraries paradigm helps to reduce operational costs, and enhance the service provider diversity and availability. The details of these points are discussed in the following section.

- **Who will use it?**

The last point of this section is the targeted end user. The developed approach aims to be consumed by a non-expert user who needs to protect sensitive information. The approach does not require a high level of experience, as the majority of procedures and functions are done internally without human control. In addition, it targets small enterprises and individuals who looking for a specific level of protection, but are not capable of covering the operational costs for such an approach. This approach is suitable for users who would not prefer to combine more than one approach to maintain the delivered security aspects. Also, the developed approach can be an appropriate solution for customers who run out of memory and need to protect their data, or if they have limited resources. Now, after explaining and answering the previous questions, the developed approach will be explained and discussed in terms of concept, mechanism, and hierarchy.
3.3 NEW APPROACH REQUIREMENTS

By moving to the developed approach requirements, there are points that should be considered while developing the approach. First of all, the limitations of existing approaches and research gaps have been discussed earlier in Chapter II, and this chapter as well. From the investigation, the data fragmentation and distribution proves its effectiveness in several computing environments. It has been utilised through the previous approaches, and it performs as expected. However, combining this mechanism with a multiple-cloud computing paradigm, such as federated cloud, and intra-cloud, shows a set of limitations, and the identified limitations have been presented above in Table 3 - 1.

The requirements for the developed approach can be grouped into four groups: i) user requirements, ii) security requirements, iii) platform requirements, and iv) performance requirements; Figure 3 - 1 illustrates these requirements. The proposed approach should avoid third party involvement, avoid outsource code implementation, and encrypt all the exchange data. In addition, the approach must aim to reduce the operational cost, enhance flexibility, provide portability, and support platform diversity. Moreover, the approach must be compatible with the existing cloud infrastructure and environment as well, and capable of being consumed as an executable file with no training need. voiding third party involvement and reducing the number of connections is important in terms of security and operational costs. By focusing on the previous points, we developed an approach that is capable of preserving the data security in terms of confidentiality, integrity, availability, and authenticity over multiple-cloud computing in an effective way in terms of flexibility, performance, and usability.
Figure 3 - 1 New approach requirements
3.4 Approach Concept

The development of a cloud computing storage system that is stable and capable of delivering a high level of security and availability cannot be achieved by relying on a higher layer of the delivered system; the lower layer must be involved to achieve that level of service. The proposed approach preserves the security, integrity and confidentiality of the stored data. We believe that the end user must have the ability to define the settings in terms of the required security and privacy, instead of relying on the service provider; providing this kind of flexibility to the end user will help them to manage and save the consumed resources. In addition, the data encryption process must be implemented on the end user machine instead of on the cloud platform which will guarantee the privacy of the stored data. Additionally, exchanging data once it has been encrypted is much safer, and the end user will guarantee that the providers cannot access or alter the data, if they do not have the key.

As we discussed earlier that the existing approaches utilise the dispersal concept, and other concepts that were not developed for the cloud computing environment. For instance, the fault tolerance and data replication. This study adopted an approach of utilising the information dispersal concept and combining it with the multiple-cloud computing paradigm will help maintain core security aspects. There are several security approaches that have been developed based on that concept (AbuLibdeh, Princehouse & Weatherspoon, 2010; Rocha & Correia, 2011; Rocha, Abreu & Correia, 2011; Bessani et al., 2013a; Bessani et al., 2013b).

Technically, the existing approaches utilise the RAID technique in order to maintain the preserved security aspect. For instance, HAIL (Bowers, Juels and Oprea, 2009a) presented as cloud version of RAID and consumed as an integrated layer through the provided infrastructure. In addition, RACS (AbuLibdeh, Princehouse & Weatherspoon, 2010) utilises a RAID5-like technique to provide a data replication system via multiple-clouds, and focuses on enhancing data availability and avoiding vendor lock-in issue. One of the addressed issues
with HAIL was the absence of supporting heterogeneity, which means consuming matched infrastructure in terms of all layers. In addition, that approach requires off-site code execution across the consumed infrastructure, and that might influence the system throughout I terms of latency. The consumed infrastructures are tightly coupled, and exchange their status in order to enhance the availability.

As improvement, RACS was developed by (AbuLibdeh, Princehouse & Weatherspoon, 2010) in order to overcome HAIL issue, and utilise RAID5. This approach overcome the heterogeneity problem, and prevent vendor lock-in issue. However, this approach might suffer from high level of latencies, and that cased by the need to pass all the exchange data through a single proxy. Besides that, relying on third party that takes control of management is not preferred solution for many reasons; for instance, availability, and loads control.

The developed approach utilises the information dispersal concept and implements it through multiple-cloud computing environment (Multi-cloud libraries paradigm) in order to deliver the requested level of security. The approach splits the selected file into a number of chunks, that vary on number and sizes based on the end-user preferences. We believe that keeping one segment of the distributed data in the end user’s machine will prevent any attempt to recover the distributed data, because an attacker must have all the segments to recover that data, together with the key. Also, by keeping the last segment in the end user’s machine, the attacker will not be able to identify the first and last segments of the targeted data. Unlike the existing approaches, part of the protected file will be stored on-site. Moreover, only the selected file will be distributed and duplicated over the consumed Multi-cloud, instead of the whole system. One of the developed approach advantages is enhancing the delivered level of flexibility by allowing the end-user to configure the approach based on the preferences, and avoid unwanted high level of protection.

The developed approach provides a set of encryption algorithms, and the end-used can select the preferred one. Practically, the protected data only will be encrypted individually and that
encryption will vary between files in terms of encryption algorithms, and encryption keys. In terms of operational cost, avoiding whole cloud replication will decrease the operational costs, and facilitates data migration between the consumed infrastructure when require.

Generally, the proposed approach relies on dividing, encrypting and storing (distributing) data in several protected, and totally independent multi-cloud storage, and then storing one segment only on the end user machine. The approach provides a set of encryption methods to enhance flexibility and allow end users to customise the consumed services based on their requirements and preferences. Storing one segment locally on the user machine prevents reconstructing the whole chunk of data. Also, the proposed approach will help to avoid unauthenticated access by distributing the data into a multi-cloud environment, and imposing two authentication levels.

One of the approach advantages is the high level of flexibility; which means that end-user can configure the approach in order to maintain the require aspects. For instance, if the end user concerns about the integrity and would like to give that aspect a high priority over the others, the approach allows to the end-user to select the encryption algorithm that meets the required needs in terms of strength, speed, and cost. Also, if the end-user would like to maintain more than one aspect, and concerns about the data protection, that means increment in the operational cost in terms of time, bandwidth, CPU, RAM, and power. Improving the delivered security level means increment in the operational cost. For example, consuming the approach over environment that has unstable, or unsecure internet connection means that the possibility for data phishing, and corruption is very high, the end-user can overcome this issue by increasing the redundancy of the distributed data over more cloud service providers in order to guarantee the safety and integrity of the protected data, in this case increasing the redundancy means increasing the number of the involved cloud services which means increasing the operational cost. Practically, the operational cost depends on a set of factors, the used platform, the maintain aspects, and the consumed cloud services.
3.5 **Approach Design**

The design approach focuses on the core practices utilised in the proposed approach. This section explains and discusses the proposed approach in terms of workflow, Unified Modelling Language (UML) behaviour diagrams, data fragmentation, data compression, multi-cloud paradigm, and the utilisation of Application Programme Interfaces (API). This section is concerned with the proposed approach procedures and practices.

### 3.5.1 Proposed approach

The proposed approach utilises several concepts to present a model that can deliver the requested security level to consumers (individuals and enterprises). This will be achieved by maintaining data availability, integrity, confidentiality, and authenticity while minimising the requested level of experiences.

The workflow has been divided into two main processes: distributing and retrieving data, and the workflow of the proposed approach can be seen in Figure 5-3. The sequence of approach tasks has been selected after testing different scenarios of tasks order; that testing will be discussed later in Chapter V. The used tasks sequence is the sequence most capable of performing the job in a safe and fast way, saving the consumed resources in terms of power, CPU, connections, and memory, and preventing unwanted behaviour, such as resource occupying and unexpected termination. First, the user needs to sign in to the preferred cloud storage service and configure the preferred security settings for data compression, cryptography system, data replication, and the requested data segments. The distribution request is established by selecting the targeted file, which is selected by the platform file browser, and then used to extract a set of essential information used by the approach: for instance, file name, file path, file size, and segment size. The customer can then start the distribution procedure based on the default settings set up at the beginning, or alter these
settings for the file. Then, the system completes the rest of the steps in the background without any interaction between the customer and the system, and the end-user will be notified when the procedure is successfully completed. The system conducts several internal processes. After establishing the distribution procedure, the selected file will be encrypted with the preferred encryption algorithm.

The encrypted file is then split into \( n \) segments, the size of each segment is calculated based on the file size and the requested number of segments. Next, the file segments will be renamed, depending on the end-user preferences. The system can also apply data compression techniques to reduce size and save consumed bandwidth, storage, and other resources. The end-user must take account of the used cloud storage as it might not allow the uploading of some compression formats for security purposes.

After that, several files are produced, the files information, the original file, and encryption algorithms are inserted into the system database. If the end-user wants to increase the data redundancy, then each data segment can be uploaded to more than one cloud storage area, to achieve the requested level of redundancy. The system database will record and keep all the maintained file information, for example, the original file name, path, size, the used cloud storage, distributed file segments path.

After distributing the segments, the file record in the system database will show the protected segments, the associated cloud storage, and all the information required for retrieving the file. The last file segment will not be uploaded to any cloud storage, it will be stored in the end-user machine; this step, helps prevent an attacker from retrieving the entire data. If they successfully catch all the distributed segments, they need to have them all to retrieve the file successfully, in case the protected file has not been encrypted.

Finally, the system database is updated to represent the valid information in the files, and all the original files, file segments (except the last one), original file paths, and segments paths
will be deleted from the end-user machine. This protects the data if an attacker gets access to the end-user machine, as these files are no longer needed. At this point, the end-user should have one segment stored in the machine and (n-1) segments stored in the cloud, which could be increased depending on end-user preferences. The data records show all the related information, such as applied cryptography and segment locations.

The retrieve procedure will be established when the end-user selects the file record from the system database. Triggering the retrieve procedure sends a download request to the cloud and the data is downloaded. After a successful download, the segments will be sorted, decompressed and combined. The information for completing these processes is stored in the records of the retrieved files. After that, the system interacts with the used cloud storage to wipe the retrieved segments, and the segments distributed to enhance the redundancy will be wiped if the files are successfully retrieved.

The decryption process is implemented to provide the end-user with a readable version of the protected file. Finally, the record of the retrieved file is deleted from the system database, which includes all related information and files, such as the stored key, and the download data segments. Wiping data that is no longer needed from the cloud and the end-user machine will save consumed resources and enhance the confidentiality and integrity of the protected files; keeping such files could be considered vulnerable.

After discussing the two procedures, the core concepts defined earlier in this chapter are reviewed and how these concepts are utilised in the proposed approach and their importance is explained. The proposed framework’s workflow and the distribution and retrieve procedures are illustrated in Figure 3 – 2 & 3 - 3. Figure 3 - 2 illustrates the approach workflow, and the task order; while Figure 3 - 3 shows the key operational tasks associated with the distribution and recovery procedures. For the distribution procedure, the distributed segments will be
recorded with the associated cloud which facilitates the recovery procedure. The segments will be wiped from the cloud after recovering the image and the logs will be updated.

Figure 3 - 2 The proposed framework workflow
Practically, the protected file will go through three phases, these phases grouped based on the types of the tasks that will be conducted over the file; the phases are as follows:

- **File preparing**
  
  This phase is responsible for preparing the targeted file to be protected based on the end user preferences. Practically, this phase will implement a set of functions that will provide the approach with the essential information for establishing the protection process; these functions are:

  - Select file: it allows the end user to browse the system directory and select the data that must be protected.

---

**Figure 3 - 3 Distribution & recovery procedures**

<table>
<thead>
<tr>
<th>Distribution Procedure</th>
<th>Retrieve Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>User requirements (configuration)</td>
<td>Retrieve Request</td>
</tr>
<tr>
<td>File Selection</td>
<td>Check Logs</td>
</tr>
<tr>
<td>File Encryption</td>
<td>Identify Cloud Storage</td>
</tr>
<tr>
<td>File Fragmentation</td>
<td>Request Files</td>
</tr>
<tr>
<td>File Renaming</td>
<td>Download Segments</td>
</tr>
<tr>
<td>File Compression</td>
<td>Segments Decompression</td>
</tr>
<tr>
<td>Distribution (Upload)</td>
<td>Combine Segments</td>
</tr>
<tr>
<td>Logs</td>
<td>File Decryption</td>
</tr>
<tr>
<td>Logs Update</td>
<td>Wipe Data</td>
</tr>
</tbody>
</table>
- Extract information: this function will deal with the selected file, and extract some information that will be used to complete the distribution and the retrieve requests successfully. This information is the file name, file size and file path.

- Segment size and numbers: through this part the segment size and number will be calculated and computed according to the end user preferences.

- File processing

  This phase will handle the file and implement a set of processes to protect it according to the security requirements that are defined by the user. This phase will produce a set of compressed and encrypted files, which will be distributed in the followed phase. This phase consists of a set of functions as follows:

  - Encryption: this function will establish the protection process by encrypting the selected file according to the user preferences. The user will be able to select the cryptography algorithm based on needs in terms of cost and security level.

  - Fragmentation: splitting the encrypted file into a specific number of segments that has been computed and determined based on the user preferences and file size as well.

  - Compressing: the compression function will help to reduce the file size and save the consumed resources in terms of bandwidth, storage, and transmission time.

- File distribution and retrieving

  This phase is responsible for connections and determining the suitable cloud storage; through this phase, the communications and the requests between end user machine and involved cloud storage will be managed.

  - Cloud storage selection: determining the cloud storage that will be consumed is based on the user needs. The factors that influence the selection process
are cloud storage availability, distributed segments number, and preserved security features.

- Establish connection: this is responsible for validating and verifying the identity of user and establishing a secure connection between the consumed cloud storage and end user machine. Relying on the developed API will help to prevent any mistake or failure that might occur due to bad practice or lack of experience.

- Distribute and retrieve: via this function the upload and retrieve requests will be triggered. Practically, both requests will be established after initiating the secure connection.

- Logs: the logs are records containing information that represents the protected files, which helps to facilitate managing the files, and allows the user to manage these files in an effective way. Moreover, it could be utilised for auditing and monitoring purposes.

At the end of each request, the file deleting function will be conducted when each request is completed successfully. This function will delete unwanted data and that will help to save the resources in the consumed machine, and in cloud storage as well. In addition, it will help to prevent any conflict that might be caused due to data replication and multi versions.
3.5.2 Data Partitioning and Replication

Data partitioning and replication mechanism are utilised to improve performance and availability of the stored data (Wei et al., 2010). Data partitioning (stripping) can be described as an algorithm that divides the targeted file into several chunks (units) which are distributed through several nodes or saved in one place, while data replication is a procedure to increase data availability by creating more than one copy of the data (Ghemawat, Gobioff & Leung, 2003; Wei et al., 2010).

By applying data partitioning and replication, losing the availability of a specific data segment will not affect the retrieval process, as at least one replica from the same segment is available. The data replication concept has been utilised in many distributed systems (for instance Bonvin, Papaioannou & Aberer, 2010; Wei et al., 2010; Long, Zhao & Chen, 2014), and aims to enhance the availability of the replicated data, rapid retrieve/access, and save response time (Loukopoulos, Ahmad & Papadias, 2002; Wei et al., 2010; Sun et al., 2012).

Applying data partitioning and replication is not effective in all applications, which means that operational costs are too high in some applications. Performing data replication increases the system consumption of bandwidth, computation, and replica management. Due to this, Sun et al. (2012) argue that three key points must be clearly identified before performing data partitioning and replication, these are as follows: 1) the number of needed replicas, 2) the time for performing replication, and 3) the replicated data itself. Carrying out data portioning and replication at various times in the cloud systems might not enhance system effectiveness or data availability, and could lead to increased consumption of resources and operational costs (Sun et al., 2012).

In the proposed approach, data replication is performed based on end user performance, as we are trying to reduce the operational costs and system requirements. The limitations of the
end-user machine capabilities are considered, in terms of power, storage, internet connection, and the processor. The number of produced replicas is calculated based on segment size, the number of involved cloud storage areas, and the number of replicas required. Moreover, data replication is utilised to maintain data confidentiality.

### 3.5.3 Data Compression

Data compression is defined by Salomon (2004) as a ‘process of converting an input data stream (the source stream or the original raw data) into another data stream (the output, the bit stream, or the compressed stream) that has a smaller size’.

Information that contains audio, video, or images requires larger storage capacity and a communication method is needed for computing such files. According to Steinmetz and Nahrstedt (2002), exchanging such files in their original format through the network consumes the provided bandwidth, increases the costs and reduces the efficiency of the network, so the data must be compressed before being transferred to save bandwidth and improve feasibility. In addition, compression helps enhance the system and saves data transfer time.

Furthermore, it allows quality control of the files via the retrieval step (decompress), which helps to optimise the consumption of system resources and prevents unnecessary data consuming the resources. The proposed approach allows the user to select the compression option based on his/her preferences, which is an added feature that improves the approach performance and reduces operational costs. The capabilities of the targeted customer and the platforms used to consume the developed approach must be considered. Compressing the data reduces the size of consumed storage on the user’s machine, in the involved cloud, and the data exchange between the machine and the cloud.
3.5.4 Multiple-clouds Paradigm and Cloud Storage Service Provider

As mentioned in chapter II, this approach utilises multiple-cloud computing which increases the benefits of the consumed ICT solutions, as discussed in the previous chapter. One of the benefits is preventing the vendor lock-in issue and another is the enhancement of the redundancy of the stored data.

In the proposed approach, the multi-cloud libraries paradigm is selected as the most suitable paradigm for delivering the requested service as expected. The multi-cloud libraries paradigm allows consumption of cloud storage services that are independent and isolated from each other. There is no communication between the consumed cloud storages which enhances the confidentiality of the delivered service. In the multi-cloud libraries paradigm, the cloud storage mediators are allocated in the end-user machine, which means that no third party is involved while the end-user communicates with the cloud service provider, and all the exchanged data will be limited to the data owner and the cloud service provider. Losing the availability of a single cloud service does not affect the other cloud services.

Relying on the multi-cloud libraries paradigm helps minimise the impact of losing the availability of cloud storage service on the other cloud services, and reduces operational costs, as it reduces the communication required by avoiding a third party. The absence of a requirement to exchange keys between the utilised cloud service storage areas, and limit key exchange to the cloud service and end-users only, will enhance the confidentiality and integrity of the exchanged data. As there is no need to have pre-agreements between the used cloud storage providers, the end-users have more flexibility to select a service provider capable of meeting their different requirements, such as free storage capacity, varied upload/download limits, and extra costs for additional capacity.
3.5.5 API utilisation

The Application Programming Interface (API) consists of a group of tools that contains predefined protocols, procedures, and routines for providing smooth interactions between software units (Petcu, Craciun & Rak, 2011). API is utilised by developers to facilitate the development process, save time, save the consumed resources, and meet the applied policies. Figure 5-5 shows where the API is allocated. In the context of cloud computing, the API is utilised for interacting with the service provider through the internet to request or send information that might vary between authentication, send, or request data (Petcu, Craciun & Rak, 2011). According to Bhadauria & Sanyal (2012), the end-user has no control over the code or data that might be implemented by other customers, who share and consume the same infrastructure.

Most cloud service providers rely on a passkey to check identity and verify the assigned authorisation level, which is considered a critical security issue if the developer takes responsibility for developing the authentication procedure themselves. The absence of experience, or the lack of good programming and security practices by the developer, could lead to exposing the consumed and shared infrastructure (Bhadauria et al., 2013). However, this is not limited to the authentication process, as it could expose the exchanged data too.

By providing a list of APIs to the customers, the cloud service provider guarantees that the exchanged data fulfils the defined requirements and policies. In addition, it enhances the availability of the delivered service, as it assures that developers must obey the requested policies. This utilisation also saves the time consumed, receives the needed support from the service providers, and provides several libraries with complex or regular functions, for instance, upload, create, or edit.
The API of the involved cloud storage services will be used in this approach. The utilisation of these APIs will help to save the consumed time and other resources. In addition, it enhances the security level, as bad practice in programming is prevented, and the authentication and data exchange are implemented as expected.

![Computer system architecture and API location](image)

**Figure 3 - 4 Computer system architecture and API location (Smith, Nair, 2005)**

### 3.5.6 Dynamic passcode keypad

The developed approach uses a keypad to enter a dynamic passcode which is entered to move between activities. It aims to enhance the security level by dynamically changing the order of the numbers each time they are used. In other words, the numbers will be displayed in a random order, not logical order.

Relying on this passcode mechanism will prevent unauthorised access from the user machine. A dynamic keypad prevents obtrusional and snooping attempts to find the passcode, as the number changes each time the keypad is displayed. As individuals are the targeted customers, and with the huge increase in consumed cloud services via smart devices, applying that technique also prevents fingerprint tracking.
3.6 **System Architecture**

After discussing the core concepts, technologies and techniques, we discuss the overall architecture of the proposed approach. This section illustrates the modules of the developed application and their functions/tasks, and the architecture of the proposed approach and internal units are illustrated in Figure 3 - 5. The approach consists of four modules allocated between the end user platform used to implement the developed approach and the internet connection, which links the developed approach through the APIs of the selected cloud storage with the cloud storage services. As mentioned, the proposed approach might be consumed through different platforms including Windows, Android, or an iPhone Operation System (iOS).

![Figure 3 - 5 Proposed approach architecture design](image)
3.6.1 Preference Module

This module is responsible for customer preferences of the consumed service regarding the characteristics of the cloud storage service provider, the targeted data segments, and the required security level: Figure 3 - 6 shows the preference module. The cloud storage provider selection procedure will be implemented based on the end-user requirements, which include data size, the targeted number of segments, and the available storage in the targeted cloud storage. The maximum cloud storage file size could affect customer selection, especially if they want to protect large files, so it is worth mentioning that the file size limits vary between service providers and the consumed platforms. For instance, there is no file size limit for Dropbox cloud storage when the user uploads files via the desktop or smartphone applications, whereas if the user uploads files via the website the limit is 10 Gigabytes (GB) (Dropbox, 2016). In addition, this module allows the end users to configure behaviour to meet the expected level of protection; they are able to select the preferred cryptography system and the number of segments the targeted file will be divided into. The involved cloud storage services are represented to the end user in terms of authorisation, authentication, and availability. This module utilises the APIs developed by the cloud storage service providers to establish communications between the end user platform and the cloud services. Relying on the APIs and granting authorisation to the approach helps protect the authorisation and verification producers before they start exchanging data. A supporting back-up option helps retrieve the protected data when required.
3.6.2 Data Management Module

The data management module is responsible for data fragmentation, data recovery, renaming data fragments, and wiping data when requested. Data fragmentation has a core role in this approach: Figure 3 - 7 illustrates data management module. This module splits the targeted file into several segments based on end user preferences defined in the preference module. It combines the retrieved segments with the device segments to present the protected file. The wiping data unit ensures that records, files and segments selected for deletion from the system are deleted from the cloud storage, the end user machine and the data logs. The data segments will be wiped with the data logs when the targeted data has been successfully retrieved. Renaming the data segments helps hide the identity of the exchanged data to protect it from phishing attacks, and facilitates the archiving of the secured data. The data management module initialises the selected file for the next module, based on the configuration from the previous module, which formats the behaviour of other modules. It finalises the retrieved data and presents it in a readable format for the end user.

Figure 3 - 7 Data management module
3.6.3 Data Security Module

The data security module is responsible for cryptography and compression which are two key procedures; compression and cryptography are as shown in Figure 3 - 8. The cryptography procedure consists of encryption and decryption. This module performs the cryptography selected by the end user in the preference module. The developed approach provides a group of cryptography techniques, from which the end user selects one that meets their required level of data protection. The implementation of the cryptographic tasks (encryption and decryption), keys generation, keys management, and keys storing are completed in this module. Encryption is conducted before compression of the segments. Compressing the segments reduces the consumed storage in the end user machine and the involved cloud storage services, which helps reduce the consumed bandwidth. When the data is successfully retrieved, the data security module cooperates with the data management module and deletes the stored keys used for the targeted files, and does the same when the end user deletes a specific log. The data security module receives several segments from the data management module and passes them to the data distribution module as encrypted and compressed segments, according to end user requirements.

Figure 3 - 8 Data security module
3.6.4 Data Distribution Module

This module is responsible for the distribution procedure, retrieve requests and data logs. It utilises the cloud storage service APIs to provide an abstraction layer to provide stable and reliable connections. The data distribution procedure will be executed based on two main factors: the user security requirements and the status of the involved cloud storage providers. The security requirements in this module are defined as the number of segments and the requirement for segment redundancy. The distribution procedure is influenced by the status of the selected cloud storage providers. There are several issues that could affect uploads and downloads from the selected cloud storage services and are as follows:

- Valid authorisation, which means granting the approach access to the cloud service, and verifying the user account.
- The availability of the cloud service provider.
- The available storage in the cloud.
- The size of the exchanged segment.

The data logs unit facilitates the documentation of the protected files information; this information represents the segment’s names, segment’s cloud storage path, segment’s machine path, and cryptography information. The data logs unit passes the information to the data distribution and data request units, which tells them which cloud storage contains the targeted segments and their paths. In addition, enhancing the redundancy of the protected file is implemented in this unit, as each segment will be uploaded to more than one cloud storage based on end user preferences. The data logs unit is critical to the system back-up, as it is the only unit that has all the information about the distributed segments.
3.7 CRYPTOGRAPHY

Applying the proper cryptographic system is important to maintain the integrity, confidentiality, and safety of the protected files. The strength of the applied cryptographic system does not mean the utilised system is efficient, as that depends on other factors such as the consumed time, consumed resources and data types. The protection of image files is different to other types, such as text files. Using a traditional encryption algorithm, for instance, Data Encryption Standard (DES) (Coppersmith, 1994), Triple DES (3DES) (Nadeem & Javed, 2005), Advanced Encryption Standard (AES) (Daemen & Rijmen, 2001) and Rivest-Shamir-Adleman (RSA) (Rivest, Shamir & Adleman, 1978) as a cryptographic system is not an appropriate choice for various reasons and that is because of the algorithm mechanism requirements and the image file characteristics, such as correlation and redundancy, which means that encrypting images using these algorithms is expensive in terms of the consumed time, power and machine resources (Gao et al., 2006; Pareek, Patidar & Sud, 2006; Liu, Sun & Xu, 2009; Li, 2013; Hraoui et al., 2013; Mekhaznia & Zidani, 2016; Enayatifar et al., 2017; Çavuşoğlu et al., 2017; Ye & Huang, 2017; Parvaz & Zarebnia, 2018). However, through our approach the AES and chaotic map will be investigated and tested in order to compare these algorithms in terms of compatibility, performance, and consumed resources.

The popularity of mobile devices, including smart phones, tablets and laptops, continues to increase. As a result, the need to develop an encryption algorithm capable of delivering the requested level of protection with minimum operational costs, in terms of time and power, is critical. The characteristics of the image files must be considered, and chaos theory has received a lot of attention due to its features and characteristics. Chaos-based encryption algorithms are produced using a mathematical procedure, which could facilitate reproduction when required (Avasare & Kelkar, 2015; Mekhaznia & Zidani, 2016). According to Gao et al. (2006), sensitivity to and dependency on the conditions of the parameters are the main
reasons that chaos-based encryption algorithms are presented as a promising solution. There are many systems developed based on the chaotic concepts that aim to provide a strong and fast cryptography: for instance, Gao et al. (2006), Gao & Chen (2008), Wang et al. (2010) and Wang, Song & Liu (2016).

The chaotic map describes the behaviour of a specific dynamic system produced under specific conditions (Dachsel & Schwarz, 2001; Sathishkumar & Sriraam, 2011); the behaviour can be generated utilising different chaotic maps such as, tent, quadratic, and logistic maps (Parker & Chua, 1987; Dachsel & Schwarz, 2001; Sathishkumar & Sriraam, 2011). Figure 3 - 9 illustrates the typical chaos algorithm.

![Figure 3 - 9 The chaos algorithm (Lian, Sun and Wang, 2005)](image)

In the developed approach the chaotic map is utilised to enhance the security of encrypted images in terms of their integrity and confidentiality, and prevents un-authorised access. Relying on a chaotic-based image, a cryptographic system improves the provided protection, although, it increases the complexity of the cryptography unit in the system. The encryption algorithm analyses the protected image in three colour levels (red, green, blue) and then executes the XOR procedure with the key generated from the following equation developed by San-Um and Ketthong (2014), which represents the absolute nonlinearity-based value:

$$x_{n+1} = |\alpha \cdot x_n - 1|$$

The $x_n$ in the equation represents the chaotic map state variable which varies between one and zero. The $\alpha$ represents the chaotic map control parameter which is between one and two. The chaotic map is influenced by two factors: the values of control and the initial condition. To
establish the encryption process, the cryptography system needs a password and image. The provided image is the image that needs protection, and the password $J$ must be 12 characters, and include letters and numbers.

First, the password is grouped into three consisting of four characters for each $(A, B, C)$. These characters are converted into binary, which means that $(A, B, C)$ will consist of 32 bits each. After that, the three binary codes $(A, B, C)$, are employed to calculate three independent values $(D, E, F)$, that are used to generate the initial condition values $(I_R, I_G, I_B)$, which determines the control parameter. Table 3-2 shows the equations for calculating these values.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D$</td>
<td>$(A_0 \times 2^{31} + A_1 \times 2^{30} + A_2 \times 2^{29} + A_3 \times 2^{28} + \ldots + A_{31} \times 2^0) / 2^{32}$</td>
</tr>
<tr>
<td>$E$</td>
<td>$(B_0 \times 2^{31} + B_1 \times 2^{30} + B_2 \times 2^{29} + B_3 \times 2^{28} + \ldots + B_{31} \times 2^0) / 2^{32}$</td>
</tr>
<tr>
<td>$F$</td>
<td>$(C_0 \times 2^{31} + C_1 \times 2^{30} + C_2 \times 2^{29} + C_3 \times 2^{28} + \ldots + C_{31} \times 2^0) / 2^{32}$</td>
</tr>
<tr>
<td>$I_R$</td>
<td>$(D \times E) \mod 1$</td>
</tr>
<tr>
<td>$I_G$</td>
<td>$(D \times F) \mod 1$</td>
</tr>
<tr>
<td>$I_B$</td>
<td>$(E \times F) \mod 1$</td>
</tr>
<tr>
<td>$a$</td>
<td>$((D \times E \times F) \mod 1) \times 0.1 + 1.9$</td>
</tr>
</tbody>
</table>

Calculating the control parameter $a$, produces three series of values (red, green, blue) that represent the key image used for encryption. The three series are utilised to represent the key image pixel values (R,G,B). Before producing the key image, one series (blue) is used to mix up the original image that needs protection. By this phase, there are two images, the key image and the scrambled image. Each pixel in these two images is represented in a format (R,G,B), and the value associated as follows $(I_R, I_G, I_B)$.

Next, both images are isolated into three panels for each: these panels represent the red, green, and blue values for each pixel. After that, three XOR operations are executed between each two panels, for example, the key image red panel will be associated with the scrambled image red panel, which produces three panels. These panels will be combined to produce the
encrypted image, which can be safely exchanged between nodes. The decryption procedure starts by isolating the encrypted image into three panels (red, green, and blue), then the XOR operations are implemented and three panels are produced that are combined to retrieve the scrambled image. The scrambled image will be restored to its original status by omitting the added noise. Figure 3 - 10 illustrates the encryption procedure.

Figure 3 - 10 Encryption procedure
3.8 Threats Modelling

After discussing and explaining the proposed approach concept, work, and design, this approach needs to be studied from a threat modelling perspective. Practically, the threat modelling takes place after implementing the developed software functions, and this step will help to reduce the development costs and enhance security as well. According to Swiderski & Snyder (2004) and Wang, Wong & Xu (2007), threat modelling can be defined as reviewing the system design in a systematic way in order to discover, evaluate, and revise the studied system against the potential threats that might influence the security aspects in a direct or indirect way. Wang, Wong & Xu (2007) argue that it is not possible to state that the developed approach is perfect and secure, however, we can show that the developed approach is invulnerable to specific threats.

Figure 3 - 11 Approach trust domains
Through the developed approach the trust domains can be divided into two general trust domains as illustrated in Figure 3 - 11. The conducted threat modelling step covers the end user machine that will be used to implement and consume the developed approach, the consumed cloud storages services that will be used to store the distributed data, and finally the internet connections between these nodes. It is important to state that each cloud storage provider has individual and independent trust domains. The identified threats and mitigations for each module will be covered based on STRIDE threats analysis (Shostack, 2014), Figure 3 - 12 details the Data Flow Diagram (DFD) in terms of process, data store, data flow, and trust boundaries, while Figure 3 - 13 illustrates the assets and the associated threats. Table 3 – 3 shows the score for each threat.
<table>
<thead>
<tr>
<th>Asset</th>
<th>Threats</th>
</tr>
</thead>
</table>
| Front End                  | • Spoofing
                                • Information disclosure
                                • Denial of Service |
| Database                   | • Tampering
                                • Repudiation
                                • Information disclosure
                                • Denial of Service |
| Network Communication      | • Spoofing
                                • Tampering
                                • Denial of Service |
| Cloud Storage Services     | • Spoofing
                                • Tampering
                                • Repudiation
                                • Information disclosure
                                • Denial of Service |

**Figure 3 - 13 Approach assets and associated threats**

- **Front End**

  The front end represents the firmware that controls and configures the behaviour and performance of the consumed approach. This front end is located onsite and interact with the consumed cloud storage services, and end-user platform. This front end will help to implement and consume the developed approach in a safe, proper, flexible, and effective way in order to protect the selected file. In terms of trust boundaries, this asset interacts with four different trust domains, which are the end-user machine, the developed approach, the cloud storage services, and the cloud APIs. This asset will be able to access end-user machine files, approach database, approach event logs, encryption algorithms, and the rest of system assets. In addition, managing the authentication, and establishing the secure connections between the end-user machine and consumed cloud storage services will be done through that asset. For that, the front end has a high level of potential threats, and the developer must prevent,
avoid, or reduce the possibility of these threats. There are three potential threats associated with this asset, these threats as follows:

- Information Disclosure: in order to prevent exposing the data, that might occur due to implementation failure, or end-user bad practice. The developed front end utilises debugging and error messages over all activities. All the data that is located inside the approach trust boundary will be encrypted. In addition, detecting errors, or unsafe termination will trigger an action that prevents performing any followed activity. Moreover, unsafe termination might expose the data and prevent completing the following steps as expected. In addition, all the data that will be outsourced to the approach trust boundaries will be encrypted with the proper encryption according to the end user needs.

- Denial of Service: during our approach, the utilised cloud computing paradigm has been designed to avoid relying on a third party between the user and the consumed cloud storage services. The utilised cloud computing paradigm (multi-cloud libraries) will communicate with the consumed cloud storage in a direct way, and avoid any external broker (mediator). In addition, losing the availability of the end-user machine is a potential threat. For that, the end-user can configure the approach to perform a regular backup based on the end-user preferences.

- Elevation of privileges: one of the critical threats is unauthorised access. For that, we need to prevent any authorised access attempt, and verify the level of privileges, and that was achieved by verifying the identity when the end-user moves from one activity to another. This verification process can be done by utilising several techniques: for instance, image password, finger print, or face recognition.

**Approach Database**

The database of our approach stores three set of data as follows: i) protected data information, ii) event logs, and iii) consumed cloud storage information. The protected
data information contains the files names, files locations, sizes, the used encryption algorithm, and compression. The consumed cloud storage services information will be located in different trust domain. An important point that must be highlighted is that the consumed cloud storage services sensitive information, such as the user name and password, will not be stored outside that domain or transmitted outside the approach, and will be excluded from any backup process in order to prevent any issue that might lead to expose this sensitive information. These sets of information will help to manage the protected files and be utilised for auditing and monitoring as well. There are four potential threats associated with this asset, these threats are:

- Tampering: tampering the stored data, the system time, and critical event must be considered as a high potential threat. Generally, all the stored data will be encrypted, and there is no data will be processed in plain format. The approach database will be limited for insert and read requests, and the same thing for the logs that document each conducted attempt, to protect data. The approach will delete the records from the main database when the file retrieved successfully. However, the delete request will be limited to the main database, and the secondary database will keep the records as it is. In addition, the additional activities logs can be used to verify and validate the data, which does not allow delete or edit records. Also, combining and comparing this information with cloud storage metadata can be used when needed.

- Repudiation: in the end-user machine, all the conducted activity will be monitored and logged with details and time stamped as well; regardless if this activity completed as expected in a successful way or terminated in unsafe way. On the other hand, the consumed cloud storage services are capable of proving who, when, where, and what relating to the performed requests. This information provides a clear evidence about what occurred, and prevent any repudiation attempt. Combing the event logs with the
cloud storage history will provide a robust evidence for all the conducted activities over the end-user machine, and the consumed cloud storages.

- **Information Disclosure**: the protected data information, event logs, and consumed cloud storage information is located in the approach trust boundary. All the stored data will be encrypted, and there is no information will leave the trust boundary, and the exchange data will not contain any sensitive or personal information that might be used to gain access to the consumed cloud storage. Moreover, all the transmitted data will be encrypted with the proper encryption algorithm before leaving the trust domain.

- **Denial of Service**: one of the reasons that lead to avoid relying on broker in order to reduce the possibility to lose the availability of the delivered service. The developed approach is a single user approach; however, if the attacker gains access to the approach database and performs a lot of requests over the database, in order to consume the provided resources. In that case, the end-user can move one or more copies of the database to another platform, and start using the delivered service. This operation can be done manually or automatically, based on the end user preferences; for instance, this backup task can be done after each distribute or retrieve request, or at a specific time (day, week, or month).

- **Network Communications**

  The developed approach prevents any third-party involvement, and all the conducted connections will be limited to the participant cloud services and end user machine. The communications will be used to distribute and retrieve the protected file. Practically, both the end user and cloud service provider share the responsibility to establish a secure and stable connection. While the cloud service provider is responsible for detecting and alerting the end user about any suspicious behaviour, the end user is responsible for protecting the authentication and authorisation information. The potential threats as follows:
- **Spoofing:** all the network communications through the developed approach will be limited to two points (end-user machine, and cloud storage service). And the impersonation can be avoided by utilising two levels of authentication. In addition, all the communication between the end user machine and the consumed cloud storage services will be via secure connections and the cloud service provider requirements will be followed.

- **Tampering:** in order to prevent tamper packages, all the exchange data will be encrypted and transmitted over secure connection. However, if the attacker captures all the transmitted packets successfully, the attacker must know the decryption key and gain access to the end user machine to find the last segment of data. In addition, applying some test to verify the integrity and safety of the exchanged data can be done.

- **Denial of Service:** one of the potential threats is attacking the connection availability and consuming the provided resources might influence the connectivity and availability as well. However, utilising an external in order to monitor any suspicious activity, or attempt to exhaust the provided resource, could help to maintain the connection availability. Practically, the developed approach keeps copies of the transmitted data onsite until receiving a message that confirms (upload, or download) successfully, which prevents data corruption if the availability was attacked.

**Cloud Storage Services**
The developed approach uses a set of cloud storage services in order to store the distributed file. The utilised multiple-cloud paradigms are totally isolated and independent. Each cloud storage service has its own policy, security, and trust boundaries. These services will not communicate with each other whether in a direct or indirect way; each cloud storage will exchange data with the end user machine only,
and no third party will participate through our approach. The potential threats for this asset as follow:

- **Spoofing**: The two levels of authentication will be utilised to prevent unauthorised access to the consumed cloud services. Practically, the cloud services APIs that have been developed by the cloud service providers will be used to establish a secure connection and verify the user identity. In addition, intrusion and suspicious behaviour detection techniques will be used by the cloud service provider.

- **Tampering**: all the exchange data between the end user machine and the cloud storage services will be encrypted before being transmitted. The data will not be dispatched to the approach trust boundaries before being encrypted. In order to enhance the integrity and the confidentiality of the data, the encryption keys will be owned by the data owner only.

- **Repudiation**: Third party participation has been avoided through our approach, and all the conducted process will be audited, logged, and updated in detail in the end user machine and the consumed cloud storage service. In addition, the data owner only is capable to read the distributed data, and the protected data can be checked to verify the integrity, and safety if there is any suspicious behaviour.

- **Information Disclosure**: the protected data will not be stored in one location and will be saved in independent and geographically separate locations. Each cloud service has its own trust boundaries, requirements, and policies. Selecting a trustworthy cloud service is the responsibility of the end user, at the same time preserving the protection of the saved data is the cloud service provider’s responsibility, as the end user has no control over the data.

- **Denial of Service**: increasing the redundancy, in order to enhance the availability of the distributed data, will help to overcome losing the availability of the cloud
service. In addition, the approach will avoid vendor lock-in issues by preventing dependency on a single cloud service. Moreover, the end-user is responsible to select the used cloud storage services, and selecting cloud service provider with high level of trust, and availability will be better than relying on a provider with lower level.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Front-end</th>
<th>database</th>
<th>Network communications</th>
<th>Cloud storage services</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spoofing</td>
<td>N/A</td>
<td>N/A</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Tampering</td>
<td>N/A</td>
<td>High</td>
<td>Medium</td>
<td>High</td>
</tr>
<tr>
<td>Repudiation</td>
<td>N/A</td>
<td>High</td>
<td>N/A</td>
<td>High</td>
</tr>
<tr>
<td>Information disclosure</td>
<td>Low</td>
<td>low</td>
<td>N/A</td>
<td>Medium</td>
</tr>
<tr>
<td>Denial of service</td>
<td>Low</td>
<td>Medium</td>
<td>Low</td>
<td>Medium</td>
</tr>
<tr>
<td>Elevation of privileges</td>
<td>High</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

*Table 3 - 3 Threats' scores*
3.9 Maintained Security Features

The developed approach has several security features to overcome the limitations defined in the research, which are as follows: confidentiality, integrity, authentication, retrievability, auditing, and disaster recovery.

3.9.1 Confidentiality

Confidentiality refers to the ability to preserve the privacy of the data and prevent unauthorised parties from reaching or reading it. Confidentiality of the stored data is preserved in the proposed approach using several techniques. Data confidentiality is achieved by relying on data fragmentation, a proper cryptographic system, two authentication levels and multi-cloud storage. Splitting the targeted file into chunks and distributing them to different places, which require individual authentication procedures to grant access, helps defend the protected file, as the attacker needs to have all the file chunks and the end user cloud service account information to reach them. Employing an efficient cryptographic system enhances the provided confidentiality level and encrypting the data increases the delivered level of data confidentiality.

3.9.2 Integrity

Data integrity means that the data is safe and has not been modified by an unauthorised party. The integrity of the stored data can be enhanced by splitting it into chunks, encrypting it and dispersing it in different multi-cloud storage areas. By treating the data in this way, it is difficult for the attacker to catch all the data chunks to recover a file. If an attacker retrieves the distributed data chunks, he still needs the chunk stored on the end user machine, which would require decryption. Using techniques to verify the integrity of a stored file such as MD5 guarantees their integrity, and proves that the stored file has not been altered.
3.9.3 Authentication

To maintain authentication, two steps are applied in the proposed approach. Relying on the cloud service APIs at the sign-in process minimises the possibility of bad practice that could occur through connection establishment with cloud storage service providers and assures that the conducted account verification procedure is implemented, according to the requested standards. The cloud storage applies two levels of authentication, to verify the sign-in request and grant the required permissions. When the end-user wants to protect a file, or see the protected files list, they are asked to verify themselves by entering a pass code. The pass code must be entered through a keypad on the end-user device and the order of the key values is different each time. This technique prevents the pass code being discovered by observation of the screen finger print, or by observing the end-user’s finger movements.

3.9.4 Auditing

As the proposed approach has been developed for the individual, data auditing aims to assess the protected files data, which is divided into four categories: original file data, split file data, protected file data, and cryptographic data. The original file data consists of the protected file source path, name and file size. The split file data shows the path of each data chunk and chunk name. The protected file data contains the consumed cloud storage name, the chunk path for the cloud storage, the chunk name, and the path and name of the chunk stored on the device and the cryptographic data and MD5. This information helps to audit the protected files, assures that the implemented process has been completed correctly and that all the data is valid. If the end-user wants additional information regarding the distributed data, it is available from the cloud storage service providers.
3.9.5 Disaster recovery

Disaster recovery is maintained by a pre-scheduled backup, or as requested. The backup procedure targets the protected data information including the data logs, which provide a detailed map for the protected data and the device's segments. The backup procedure allows the backed-up data to be stored on the device or sent to the selected email address. This data can be recovered using different devices, which produce detailed data logs and the device's segments for the protected data. In addition, the backup can be installed on more than one device and provide more than one choice of interaction with the protected files, but the protected data version is not synchronised, which means that any alterations are limited to the execution machine. For example, if the end-user adds a new record to the data logs, that record will not exist on the other devices, and the same if information is deleted. Preventing the synchronisation feature lowers the risk of exposure, as exchanging such types of data containing sensitive cryptographic data is a critical threat.
3.10 Summary

This chapter has discussed the proposed approach in detail. First of all, there were two contributions of this study have been achieved. The first one was identifying the need for developing an approach capable of protecting end-user data in a multiple-cloud environment. After that, a set of questions have been answered in order to show the approach importance, how is it different, and who will use it. Also, the requirements and the need for a new approach have been discussed. The second achieved contribution was developing a new framework that utilises a multiple-cloud model that provides personal data protection by distributing the data segments over these clouds and storing one segment on the user machine to prevent any decryption/modification attempts from outside the user machine. The proposed approach was presented and discussed in terms of its originality, expectations, capabilities and contribution. After that, the maintained security aspects of this approach were discussed, and a detailed discussion about preventing and reducing the impact of a possible failure have been provided. The next chapter discusses the developed prototype in detail in terms of requirements, functions, and architecture.
Chapter IV. ARTEFACT DEVELOPMENT

This chapter aims to developing an android mobile application that uses the developed approach, thus providing evidence of the real-world application of the model, and allowing new knowledge and insight to be gained. The chapter starts with introducing the prototyping phase and review the need of it. After that, the artefact development is discussed in terms of requirements, and the development environment is detailed as well. These two sections help to provide a clear understanding of this phase’s purpose and show its importance in the research context. Moreover, justifying all the decisions made through these two sections enhances the clarity.

Next, the developed android application is discussed in detail. The discussion covers several points: for instance, the design, hierarchy, and functions. The activity diagrams, screen shots and sources code are discussed, and explained in detail in order to show how this approach will perform.
4.1 Prototyping

In this study, the prototyping method aims to develop and implement the core features of the developed approach, to verify, validate, and test the proposed approach. The prototype consists of three phases: i) design, ii) implementation, and iii) testing. The design phase was discussed earlier and this chapter focuses on the implementation phase, while testing will be covered in the next chapter. Implementing the prototype helps check both the performance and behaviour of the developed approach, whether it executes correctly or not, and if it meets customer requirements. The prototype is utilised to ensure that the application can be used without errors and unwanted behaviour. Moreover, the prototype is required for research, to test and validate the proposed approach.

The android application prototype has been selected for several reasons. Employing android helps utilise the cloud storage service provider APIs, which guarantees and verifies user data and information, and prevents any issues that arise from bad practice during the authentication step. Finally, the android application prototype reduces the consumed resources of time and machine specification.

In the rest of this chapter, the prototype requirements are discussed, including development platform, development software, machine specifications, and the involved APIs. In addition, the developed android application is explained in terms of the developed functions, prototype architecture, and the developed Graphical User Interface (GUI).
4.2 **Prototype Requirements**

In this section, the requirements of the developed prototype are illustrated and discussed. The prototype has been developed for an android platform instead of for Personal Computers (PC) or Mac (Apple) machines for several reasons. First, the mobile cloud market is expected to increase from $12.07 billion in 2017 to approximately $74.25 billion by 2023 so it makes financial sense to target the smart device platform (Swamy, 2017). Maintaining the security aspects in the developed approach helps improve and enhance the consumed cloud services in terms of integrity, confidentiality and protects the customer, as this field has already gained the attention of attackers. In addition, developing the prototype for testing on a smart device guarantees performance efficiency, proves the efficiency via a machine with limited resources in terms of its processor, memory, power and connection, and so guarantees its efficiency on higher performance machines. For the selected operating system, choosing android is based on market statistics. Gartner (2016 & 2018) shows that Apple’s share of the market was 14.6% in 2015, which dropped to 11.9% in 2019. The International Data Corporation (IDC) report of 2018 shows that the market share of iOS devices dropped from 18.2% to 13.2% between quarter 4 of 2016 and quarter 3 of 2018, whereas for the same periods, the share of android devices was 81.4% and 86.8% (IDC, 2018). Table 4 -1 & 4 - 2 and Figure 4 - 1 show this in more detail. In addition, developing the prototype as an android application enhances portability, mobility, and compatibility, as there are huge numbers of devices with different specifications, so the prototype can be tested and checked using different capabilities, which helps expand the targeted customer spectrum. In term of testing and experiments, there are a number of tools that can be utilised to facilitate the conducted experiments, and provide accurate and detailed information regarding the performance and behaviour of the consumed device. Also, using an android platform provides flexibility in the testing and validation phase, and helps implement and test the targeted functions and mechanisms.
### Table 4 - 1 Smartphone OS market share 2016 - 2018 (IDC, 2018)

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Android</strong></td>
<td>81.4%</td>
<td>85.0%</td>
<td>88.0%</td>
<td>87.6%</td>
<td>80.3%</td>
<td>84.3%</td>
<td>87.8%</td>
<td>86.8%</td>
</tr>
<tr>
<td><strong>iOS</strong></td>
<td>18.2%</td>
<td>14.7%</td>
<td>11.8%</td>
<td>12.4%</td>
<td>19.6%</td>
<td>15.7%</td>
<td>12.1%</td>
<td>13.2%</td>
</tr>
<tr>
<td><strong>Others</strong></td>
<td>0.4%</td>
<td>0.2%</td>
<td>0.2%</td>
<td>0.1%</td>
<td>0.1%</td>
<td>0.0%</td>
<td>0.1%</td>
<td>0.0%</td>
</tr>
</tbody>
</table>

### Figure 4 - 1 Smartphone OS market share 2016 – 2018 (IDC, 2018)

### Table 4 - 2 Vendor market share 2017 & 2018 (Gartner, 2018)

<table>
<thead>
<tr>
<th>Vendor</th>
<th>2018-2Q Units</th>
<th>2018-2Q Market Share</th>
<th>2017-2Q Units</th>
<th>2017-2Q Market Share</th>
</tr>
</thead>
<tbody>
<tr>
<td>Samsung</td>
<td>72,336.4</td>
<td>19.3 %</td>
<td>82,855.3</td>
<td>22.6 %</td>
</tr>
<tr>
<td>Huawei</td>
<td>49,846.5</td>
<td>13.3 %</td>
<td>35,964.3</td>
<td>9.8 %</td>
</tr>
<tr>
<td>Apple</td>
<td>44,715.1</td>
<td>11.9 %</td>
<td>44,314.8</td>
<td>12.1 %</td>
</tr>
<tr>
<td>Xiaomi</td>
<td>32,825.5</td>
<td>8.8 %</td>
<td>21,178.5</td>
<td>5.8 %</td>
</tr>
<tr>
<td>OPPO</td>
<td>28,511.1</td>
<td>7.6 %</td>
<td>26,092.5</td>
<td>7.1 %</td>
</tr>
<tr>
<td>Others</td>
<td>146,096.1</td>
<td>39.0 %</td>
<td>156,190.8</td>
<td>42.6 %</td>
</tr>
</tbody>
</table>
As an android application has been selected to implement the developed prototype, we established the development process using Java. Initially, Eclipse was selected as the most appropriate development platform but, unfortunately, after developing most of the prototype targeted functions, the support for android Eclipse tools was ended by Google (Android, 2016). In 2016, Google announced that the android studio is the official android Integrated Development Environment (IDE), so we are using it (Android, 2015). They also announced that developers can migrate their projects from Eclipse to Android Studio, but we could not make our project utilise the number of libraries and APIs, which caused a lot of critical errors and difficulties with dependencies, so we decided to start the development from the beginning, again using Android Studio to prevent and avoid any behaviour that might threaten the data or cause issues with the cloud storage service providers.

Several types of software were employed at the implementation and testing stages. The programming process was conducted using Android Studio, as it is the official android IDE, and is also associated with the Android Software Development Kit (SDK). The SDK provides a simulator, some essential libraries and sample projects with their codes. Other libraries were utilised in the developed prototype: for instance, the encryption libraries and the System Query Language (SQL) library. For debugging and testing, Android Studio allows the developer to select the preferred platform – a physical or virtual device – to implement the code. Most implementation before finalisation of the developed application was done by the simulator. The built-in simulator Android Virtual Device (AVD) was used throughout the development process, alongside the Genymotion which is faster than the AVD due to OpenGL hardware acceleration and x86 architecture. The specifications of the machines used in the implementation and testing phases are shown in Table 4 - 3.
<table>
<thead>
<tr>
<th><strong>CPU</strong></th>
<th>Intel Core i7-3770</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RAM</strong></td>
<td>16 GB</td>
</tr>
<tr>
<td><strong>Operation System</strong></td>
<td>Windows 7 Ultimate (64-bit)</td>
</tr>
</tbody>
</table>
| **Software** | Genymotion Version 2.8.1  
Oracle VM Version 5.0.28 r111378  
Android Studio  
Android SDK |
| **Screen resolution** | Screen resolution greater than 1024 x 768 |
| **Storage** | 297 GB SSD  
930 GB HDD |
| **Graphics** | Intel® HD Graphics 4000 |

*Table 4 - Development machine configuration*
4.3 **DEVELOPED ANDROID APPLICATION**

Through this section the developed prototype is discussed and explained in terms of implemented functions, architecture, and GUIs. The discussion starts with introducing the developed functions, and the core capabilities that are delivered through this android application. Next, the architecture is explained in order to show how the internal modules will work and interact to complete the requested job. Finally, the developed GUIs and the important codes are explained and discussed, in order to clarify the tasks executions in terms of source code and implementation environment.

4.3.1 **The involved functions**

Our prototype has been developed to test and validate the proposed approach, in terms of its effectiveness, reliability, performance, and the end-user experience. The developed application implements functions proposed in the main approach. The implemented functions are most of those mentioned and discussed in previous Chapter. The implemented functions via the developed android application are listed as follows: cloud service storages sign in/out, file selection, file splitting and joining, data compressing and decompressing, file encrypting and decrypting, distribute and retrieve data, and system logs.

All the implemented functions and associated tasks are discussed next. However, it is important to highlight that during the application development process, several open source libraries and APIs were utilised: for instance, the SQL library and some of the SDK. In addition, the application was limited to support the security of image files only.

The developed application allows end-users to connect with the cloud and interact with the system database. During the development process, the requested experience level was considered, as the approach targets individuals and small enterprises, and users who are not
specialists in the field of information security or who have little or no experience; the approach minimises the interactions between the end-users and the system to prevent mistakes occurring due to lack of experience. At the same time, it provides the required flexibility to customise the delivery service based on end-user requirements. Next, the architecture, and the activities of the developed prototype are discussed. To sum up, the developed artefact supports these points:

a) Preserving the protected data security;

b) Reducing the operational costs by utilising the appropriate multiple cloud computing paradigm;

c) Maintaining the integrity, authenticity, and confidentiality through one approach;

d) Minimising the needed level of knowledge to install, configure, and consume such an approach; and

e) Delivering the approach as an executable file to the end user.

f) Supporting the portability, and enhancing the delivered flexibility level.
4.3.2 Prototype Architecture

The developed android application has the same architecture as the proposed approach, but some units have been removed due to exclusion at the prototype phase. In addition, in this application, the cloud storage providers are limited to three; OneDrive, Dropbox, and Box. These service providers have been selected for the following reasons: 1) popularity, 2) API availability, 3) SDK availability, 4) Java support, 5) authentication steps, 6) offered capacity and 7) the supported platforms.

These cloud storage providers give the developers the ability to register their developed applications on provider dashboards so they can verify the applications that try to interact with the delivered service and grant permissions. For example, when a developer registers the developed application that communicates with the cloud storage provider, the developer can set up general permissions for this application, such as sharing and accessibility. In addition, the cloud storage providers will protect the delivered infrastructure and provide the registered application with a unique client ID used by them to check credentials for using OAuth 2.0 as the authentication type.

The architecture of the developed android application is illustrated in Figures 4 - 2. In the following section, the implemented tasks are explained and associated with the application screen shots for the task.
Figure 4 - 2 Developed prototype architecture design
4.3.3 Prototype tasks and GUIs

This section represents the final product produced as a prototype for the proposed approach. The demonstrated prototype is used to transfer the proposed approach from theory to practice. The implementation of the prototype and the functionality of the developed tasks are described as follows:

- **Main page**

When the application is established, the end-user will be able to choose from three options: browsing the protected file list, selecting the file to protect, and the cloud storage dashboard. To enhance security and prevent unauthorised access, each time the end-user moves between the available tasks and activities, the system asks them to enter a passcode to gain access.

The end-user can move between the provided tasks, and there are buttons to represent the protected file list, select file to protect, and cloud storage dashboard in each activity layout. The core application fragmentations are listed as follows: passcode, protected file list, select file to protect, and cloud storage dashboard. Figure 4 - 4 shows the activity diagram for the application main page.
- **Cloud storage connection**

This activity is responsible for authentication and interaction with the selected cloud storage providers (OneDrive, Dropbox, Box). In this activity, the cloud storage APIs and SDKs produced for the developers are utilised to guarantee safe and reliable established communications and the granted privileges. There are three main independent activities for signing in and establishing a connection with the cloud storage, and one activity for each cloud storage area.

Before utilising the available APIs, the developed application must be authorised to communicate with the content of the cloud storage. For that, the cloud storage provider needs to register the application that communicates with the cloud service, and configure the preferred platform for the development. The cloud storage providers can verify the identity of the application and check the permissions via
the application ID, and the application secret generated by the service providers that needs to be included in the implemented code; Figure 4 - 5 shows an example for granted ID, permissions, and secret key.

![Figure 4 - 4 Cloud service permissions, application ID, and application secret key](image)

The applied authentication for the three cloud storage providers is the standard OAuth 2.0, which is used to check the credentials of the application key and secret. Relying on the APIs helps enhance the security of the authentication procedure, and prevent any mistakes or issues that might occur if the developer takes responsibility for developing the authentication activity by himself. In addition, utilising the APIs and SDKs saves time and guarantees following the best way to interact with the cloud. End-users can enhance the security of the application by employing two levels of authentication instead of one. The cloud storage activities provide the required bases for exchanging files with the storage areas and sign-in/out capabilities.
Passcode

The passcode activity is used to verify how a person interacts with the application and is activated (automatically), when the user moves between the main screens, which are the cloud storage, file selection, and protected file list main pages. To access previous activities, end-users are asked to enter a passcode. However, the order of the numbers on the virtual keypad are changed every time they are used to prevent attempts at catching the passcode, which is done by observing end-user finger movements or finger prints. This activity prevents unauthorised access, as the user cannot move between the activities until he verifies himself, otherwise he will be blocked at the passcode activity stage. Figure 4 - 5 illustrates the dynamic passcode UML activity diagram.
for loops for randomising the keypad values and prevent value repetition

```java
for (i = 0; i < 10; i++) {
    // generate random number
    rand_nu = generator.nextInt(10);
    for (j = 0; j <= i; j++) {
        // exam if the generated value was generate
        if (rand_nu == Array_Key[j]) {
            // regenerate a new random number if the value already generated before
            rand_nu = generator.nextInt(10);
            j = 0;
        }
    }
    // assign the value to the array if the value was not assigned before
    Array_Key[i] = rand_nu;
}
```

// validate the entered passcode
```
show_pass.setOnClickListener(new View.OnClickListener() {
    public void onClick(View vshow) {
        // if statement to determine if the passed value match the defined passcode or not
        if (Pass.getText().toString().equals("xxxxx")) {
            // move to the targeted activity if the passcode was correct
            Intent ii = new Intent(vshow.getContext(), TodoListActivity.class);
            startActivity(ii);
        }
        // display a message if the passcode was not correct, and stay in the same activity
        else {
            Toast.makeText(Pass_Code.this, "WRONG PASSWORD ...", Toast.LENGTH_SHORT).show();
        }
    }
});
```
**File explorer**

The file explorer activity is used to select the file that needs protection. As our scope is limited to image files only, the file explorer activity will only allow the end-user to select an image file. After selecting the image, the application will extract and present the image path, size, segment size and MD5. The activity acts as a handler to produce the selected image, extract and pass the required information to the following activity. The responsibility is limited to preparing the selected image for the next stage, which could be encrypting or splitting, based on end-user preferences. In addition, some extracted information is passed and stored in the system database, such as the file path and the MD5. The file explorer selection activity and the involved actions are shown in Figure 4 - 6.
OnClickListener loadButtonListner = new OnClickListener() {
    public void onClick(View v) {
        // start browsing the device file when LOAD button clicked, and make the
        // root is "/sdcard"
        Intent intent = new Intent(v.getContext(),
                FileDialog.class);
        intent.putExtra(FileDialog.START_PATH, "/sdcard");
        startActivityForResult(intent, SELECT_FILE);
    }
};

// calculate and display the file and segments sizes
// calculating the file size in kilobytes
    float fileSizeInKilos = fileSizeInBytes / 1024;
// calculating the file size in megabytes
    float fileSizeInMegs = fileSizeInKilos / 1024;
    int dotposition;
// declaring a float to save the segment size
    float ll;
// displaying the file size in megabytes
    TextView t = (TextView) findViewById(R.id.fileSize);
    t.setText(fileSizeInMegs + " Megabytes");
    EditText ff = (EditText) findViewById(R.id.splitsize);

    // calculating the segments size
    ll = fileSizeInMegs / 3;
    String hh = Float.toString(ll);
// defining the "." Position
    dotposition = hh.lastIndexOf(".");
    String cutval;
    cutval = hh.substring(dotposition, 5);
    ff.setText(cutval);
File encryption/decryption

The encryption and decryption activities are carried out internally without any interaction with the end-user. After selecting the requested file and extracting the required information, the file is encrypted. The encryption activity is implemented before splitting the file. When the activity is successfully completed, the involved information is passed to the system database, where it is saved along with the keys and algorithm used. The decryption activity begins when the cloud storage segments are combined with the device segments. The retrieved file is decrypted to produce a readable file. When decryption is accomplished, the encrypted file is deleted, along with the file record from the protected file list. As these two activities are implemented internally, the application displays a message to notify the end-user that the file has been encrypted/decrypted successfully.
// encrypting the file by passing the secret key and the file
private byte[] encrypt(byte[] secretk, byte[] efile) {
    SecretKeySpec skeySpec = new SecretKeySpec(secretk, "AES");
    Cipher cipher;
    // declaring an array to hold the encrypted file
    byte[] encrypted = null;
    try {
        // selecting the wanted encryption algorithm AES
        cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        // switch to the encryption mode
        cipher.init(Cipher.ENCRYPT_MODE, skeySpec, new IvParameterSpec(iv));
        // encrypting the file and hold the outcomes
        encrypted = cipher.doFinal(efile);
        // try to catch error
    } catch (Exception e) {
        e.printStackTrace();
    }
    return encrypted;
}

// decrypting the file by passing the secret key and the file stream
private byte[] decrypt(byte[] secretk, FileInputStream input) {
    SecretKeySpec skeySpec = new SecretKeySpec(secretk, "AES");
    Cipher cipher;
    // declaring and initiating variables to hold and save the decrypted data
    byte[] dData = null;
    CipherInputStream cinput = null;
    try {
        // selecting the decryption algorithm AES
        cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        cipher.init(Cipher.DECRYPT_MODE, skeySpec, new IvParameterSpec(iv));
        // declaring and saving the input stream for decrypting
        cinput = new CipherInputStream(input, cipher);
        // declaring and write the encrypted data to buffer (outputstream)
        ByteArrayOutputStream buffer = new ByteArrayOutputStream();
        byte[] data = new byte[8];
        while ((cinput.read(data)) != -1) {
            buffer.write(data);
        }
        buffer.flush();
        // saving the decrypted data from the buffer to dData
        dData = buffer.toByteArray();
        // trying to catch error
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        try {
            // closing the opened data stream
            cinput.close();
            input.close();
            // try to catch error
        } catch (IOException e) { e.printStackTrace(); }
        return decryptedData;
    }
}
- **File splitting / joining**

File splitting and joining activities are implemented internally. After selecting the requested image and extracting the required information, the file is split into four parts. As explained earlier, three segments are distributed to the cloud storage areas, and the fourth segment is stored on the end-user's device. When the activity is successfully completed, the segments information is passed to the system database for insertion in the data logs. Then, the end-user is asked about the requested task.

File joining is established when all the segments are successfully downloaded. This activity combines the three downloaded segments and the segment stored on the end-user's device to produce one file. When file splitting has been completed, the original file is deleted, as are the three distributed segments, when they are uploaded to the associated clouds. The three downloaded segments and the fourth segment are deleted when they are combined and a combined file produced. As these two activities are implemented internally, there is no layout. The system will display a message notifying the end-user that the activity is successfully completed.
// declaring an array to hold the splitted bytes
byte[] kilobuffer = new byte[1024];
try {
// starting the splitting process while the input stream not empty
while (is.read(kilobuffer) >= 1) {
kiloByteCounter++;
// if statement to split the stream base on the defined size
if (kiloByteCounter > splitValInKiloBytes) {
kiloByteCounter = 1;
suffixCounter++;
// name the segment
newfilePath = filePath
+ getStringSuffixFromInt(suffixCounter, splitValInKiloBytes);
// writing the segment path to array before inserting to database
chunk_path_list[suffixCounter - 1] = newfilePath;
// creating a record and inserting the file and segments information into the database
insert_intent.putExtra("Original File Path", filePath);
insert_intent.putExtra("CHUNK 1", chunk_path_list[0]);
insert_intent.putExtra("CHUNK 2", chunk_path_list[1]);
insert_intent.putExtra("CHUNK 3", chunk_path_list[2]);
insert_intent.putExtra("PHONE PATH", chunk_path_list[3]);
startActivity(insert_intent);
} os.close();
// declaring array input stream , and write one segment to each variable
ByteArrayInputStream bais1 = new ByteArrayInputStream(ff1);
ByteArrayInputStream bais2 = new ByteArrayInputStream(ff2);
ByteArrayInputStream bais3 = new ByteArrayInputStream(ff3);
ByteArrayInputStream bais4 = new ByteArrayInputStream(ff4);
// combining the input streams into a single sequence in order to join the segments
SequenceInputStream sis1 = new SequenceInputStream(bais1, bais2);
SequenceInputStream sis2 = new SequenceInputStream(bais3, bais4);
SequenceInputStream sis = new SequenceInputStream(sis1, sis2);
// declaring and creating a new file to write and save the combined segments
newFileName = (Environment.getExternalStorageDirectory() + "/joind.jpg");
// reading the single sequence stream till the end, and write it to joining file
while ((ch = sis.read()) != -1) {
try {
os.write(ch);
} catch (IOException e) {
System.out.println("Error write files");
lasterror = 1;
break;}
Toast.makeText(getBaseContext(), "Successfully Joined Into File "+ ". . . DONE!", Toast.LENGTH_SHORT).show();
}
- **File Upload / retrieve**

These two activities are established from the protect files list, where the system displays information about the split files. The end-user can upload the segments, retrieve and delete the whole record. The sent requests, regardless of whether they are uploads or downloads, are sent in parallel, which saves time and reduces the impact of any delay caused by the cloud storage providers, Figure 4-7 shows upload and retrieve activity diagrams.

```java
// uploading the file
public void dropboxupload() {
    // declaring a file variable, and assign the upload file path
    File file = new File("UPLOADED FILE PATH");
    if (file != null) {
        // starting the uploading process via calling the client information
        new UploadTask(DropboxClient.getClient(ACCESS_TOKEN), file, getApplicationContext()).execute();
        Toast.makeText(DropBoxActivity.this, "UPLOAD ... DONE !!!", Toast.LENGTH_LONG).show();
    }
}

// file downloading
protected File doInBackground(Object[] params) {
    // declaring a file variable, and creating a new file to write the downloaded file into
    File dfile = new File("DOWNLOAD FILE PATH");
    try {
        // creating output stream to write the retrieved file
        ByteArrayOutputStream outstream = new ByteArrayOutputStream();
        OutputStream fis = new FileOutputStream(dfile);
        // selecting the file from the cloud storage, and download it
        dbxClient.files().download("download file path (cloud storage)").download(outstream);
        // writing to the defined file, and closing the opened stream
        outstream.writeTo(fis);
        fis.close();
    }
}
```
Figure 4 - 7 Upload and retrieve activity diagrams
- **Protected files list**

   The protected files list displays the file records in the application database. This activity is established by default when the selected files are split successfully. It could be considered as a dashboard for interacting with the protected files list in terms of upload, download, and delete.

   // creating the database
   
   public class DBHelper extends SQLiteOpenHelper {
   
   // declaring the database name
   static final String DB_NAME = "wqir.DB";
   // declaring the table name
   public static final String TABLE_NAME = "phdtest3";
   
   // declaring the table items
   // _ID is a primary key, which will be calculated automatically
   public static final String _ID = "_id";
   public static final String ORIGNAL_PATH = "file_location";
   public static final String CHUNK1_PATH = "chunk1";
   public static final String CHUNK2_PATH = "chunk2";
   public static final String CHUNK3_PATH = "chunk3";
   public static final String PHONE_PATH = "phone_path";
   public static final String CLOUD1_PATH = "cloud1";
   public static final String CLOUD2_PATH = "cloud2";
   public static final String CLOUD3_PATH = "cloud3";
   
   // declaring the database version
   static final int DB_VERSION = 1;
   
   // creating table SQL query
   private static final String CREATE_TABLE = "create table " + TABLE_NAME + " (" + _ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " + ORIGNAL_PATH + " TEXT, " + CHUNK1_PATH + " TEXT, " + CHUNK2_PATH + " TEXT, " + CHUNK3_PATH + " TEXT, " + PHONE_PATH + " TEXT, " + CLOUD1_PATH + " TEXT, " + CLOUD2_PATH + " TEXT, " + CLOUD3_PATH + " TEXT);";"
- **File deletion**

When the end-user wants to delete a protected file, it can be easily done by selecting the targeted file from the protected file list and clicking on the delete button. There are two types of record deletion, as follows: after splitting and before distributing segments to the cloud, and after distributing to the cloud. For the first type, the record will be deleted along with the existing segments (four segments in this case). The system might not be able to delete the segments, as it depends on the granted permissions and the applied application restrictions. If there are no restrictions, the segments will be deleted from the cloud and from the end-user device.

The deletion activity can be triggered automatically. The file record will be deleted automatically when the segments are retrieved and combined successfully. In addition, the original file will be deleted when the file is split and the segments information is inserted in the system database. The following Figures illustrate the deletion activities for OneDrive, Dropbox, and Box. As this activity is implemented internally.

```java
// getting the file segments form the displayed table (database)
chunk_1 = chunk1EditText.getText().toString();
chunk_2 = chunk2EditText.getText().toString();
chunk_3 = chunk3EditText.getText().toString();
phone = pathEditText.getText().toString();
// declaring files and assigning the segments path
File file1 = new File(chunk_1);
File file2 = new File(chunk_2);
File file3 = new File(chunk_3);
File file4 = new File(phone);
// deleting the segments
file1.delete();
file2.delete();
file3.delete();
file4.delete();
// deleting the record from the database , and show the delete
confirmation massage
dbController.delete(_id);
Toast.makeText(getBaseContext(), 
"RECORD DELETED", Toast.LENGTH_SHORT).show();
```
4.4 SUMMARY

This chapter has discussed the developed mobile application that uses the developed approach, in order to provide evidence of the real-world application of the model, and allowing new knowledge and insight to be gained. First of all, the prototyping aims and purposes were discussed. Then, the prototyping requirements in terms of the development environments and the targeted platform were discussed and justified. Finally, the last part focused on the developed android application aspects that covered the developed functions through that artefact, the architecture, tasks, codes, and screenshots for the GUIs. The developed artefact will be examined and tested in the next chapter in order to validate and verify the developed approach in various aspects, which will be detailed, discussed, and explained next.
Chapter V. RESULTS AND DISCUSSIONS

This chapter explains the findings of the conducted experiments that aimed to validate and verify the developed prototype. The chapter is divided into five parts. First, the testing phase is explained in order to understand the purpose of the experiments, then the testing environment is reviewed. The third part focuses on the implemented experiment aims, techniques, and design. The three core tests are discussed to clarify their purpose for mechanisms, testing level and measurements.

The fourth part focuses on the test outcomes and this discussion is divided into three parts, according to the implemented experiments. The data collected from these experiments is categorised, analysed, presented and discussed. The collected data is used to verify, validate and assess the developed prototype, which helps define its advantages, disadvantages and challenges. Finally, the faced challenges, and raised issues are discussed.
5.1 Testing

The Institute of Electrical and Electronics Engineers (IEEE) defines testing as ‘an activity in which a system or component is executed under specified conditions, the results observed or recorded, and an evaluation made of some aspect of the system or component’ (Radatz, Geraci & Katki, 1990). In the testing phase, various aspects of the developed software are investigated. First, the software must behave as expected and must successfully perform the aimed functions. The software is checked for mistakes and faults that might lead to failure. Other elements must be considered as well as the functional aspects, for instance, the presentation and the developed interfaces must be tested for usability, design and portability. There is no constant definition of testing and the proposed definitions vary, based on their context, purpose, and the tested elements (Luo, 2001).

According to Gomaa (2011) and Sommerville (2011), the main goal of software testing is to provide evidence to clients and developers that the developed software works as expected and can do what it is supposed to do. Testing highlights any unwanted or unexpected behaviour in the developed software, and each single feature, function, event and procedure must be tested and investigated independently and then together as a system. Such tests help to reduce the probability of system faults. According to Boehm (1981), software testing should answer two questions: ‘Are we building the right product?’ and ‘are we building the product right?’, which represents validation and verification of the tested software.

5.2 Testing Environment

In order to implement the testing of the developed prototype, the developer need to provide a safe and appropriate environment that is capable of executing the tested tasks safely. For that Android Studio was used, as it allows the developer to select the preferred platform – a physical or virtual device – to implement the code, and provide detailed statistics for the
executed machine, such as CPU and memory status. Most implementation before finalisation of the developed application is done by the simulator. The built-in simulator, Android Virtual Device (AVD), is used throughout the development process, alongside the Genymotion, which is faster than the AVD, due to OpenGL hardware acceleration and x86 architecture; these two options run over Oracle VM. In addition, the MATLAB has been utilised through the encryption development; the detailed information for the testing environment was discussed earlier in (4.2).

During the testing phase, there were a number of difficulties we faced. For instance, the need for continuous and accurate maintenance for the used APIs, in order to preserve the connectivity between the user machine and the consumed cloud storage services, as the cloud storage providers keep updating their policies and rules in order to maintain the preserved security level. In addition, the absence of JAVA APIs for some cloud storage providers reduces the available choices for the developer. Moreover, the need to shift the development platform from Eclipse to Android Studio caused disruption and delayed the development process.

In the following section, the conducted experiments will be discussed. Practically, there three core experiments were implemented. Each experiment has specific objectives, these objectives represent the purpose for implementing these experiments, and how they will be used in the context of this research.
5.3 Experiments

This section focuses on the implemented experiments, and aims to provide a clear explanation about each one in terms of purpose, design, implementation, and collected data. Our testing phase consists of three core tests, with a set of subtests for each one. The performed experiments are task order experiments, which aim to define the most appropriate order for the implemented tasks, the encryption experiments that compare the utilised encryption algorithms, and finally the user experience test implemented.

5.3.1 Tasks Order

This experiment aims to identify the most efficient order for the implemented tasks in terms of speed, performance and consumed resources. During this experimental phase 12 scenarios were investigated. In order to have accurate and reliable measurements, there were ten tests conducted for each single scenario, Table 5 - 1 illustrates these scenarios, whilst Table 5 - 2 shows the specifications of testing machines and files.

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Retrieval</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 File split</td>
<td>7 Segments decompression</td>
</tr>
<tr>
<td>Segments encryption</td>
<td>Segments decryption</td>
</tr>
<tr>
<td>Segments compression</td>
<td>Segments combine</td>
</tr>
<tr>
<td>3 File encryption</td>
<td>10 Segments decompression</td>
</tr>
<tr>
<td>5 File compression</td>
<td>12 Segments combine</td>
</tr>
<tr>
<td>2 File split</td>
<td>9 Segments decompression</td>
</tr>
<tr>
<td>Segments compression</td>
<td>Segments decompression</td>
</tr>
<tr>
<td>Segments encryption</td>
<td>Segments decryption</td>
</tr>
<tr>
<td>4 File split</td>
<td>11 Segments combine</td>
</tr>
<tr>
<td>Segments decompression</td>
<td>Segments decompression</td>
</tr>
<tr>
<td>Segments decryption</td>
<td>Segments combine</td>
</tr>
</tbody>
</table>

Table 5 - 1 Task order scenarios

159
<table>
<thead>
<tr>
<th>OS version</th>
<th>Marshmallow</th>
<th>Nougat</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Name</td>
<td>Samsung Galaxy S7</td>
<td>Google Pixel XL</td>
</tr>
<tr>
<td>RAM</td>
<td>4 GB</td>
<td>4 GB</td>
</tr>
<tr>
<td>Processor</td>
<td>Octa-core (4x2.3 GHz Mongoose &amp; 4x1.6 GHz Cortex-A53)</td>
<td>Quad-core (2x2.15 GHz Kryo &amp; 2x1.6 GHz Kryo)</td>
</tr>
<tr>
<td>Storage</td>
<td>64 GB</td>
<td>128 GB</td>
</tr>
<tr>
<td>Screen</td>
<td>5.1 inches</td>
<td>5.5 inches</td>
</tr>
</tbody>
</table>

Files Properties

<table>
<thead>
<tr>
<th></th>
<th>Dimensions</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1600*1600</td>
<td>173 KB</td>
</tr>
<tr>
<td>2</td>
<td>1980*1311</td>
<td>834 kb</td>
</tr>
<tr>
<td>3</td>
<td>2000*1131</td>
<td>1.24 MB</td>
</tr>
<tr>
<td>4</td>
<td>1200*899</td>
<td>215 KB</td>
</tr>
<tr>
<td>5</td>
<td>3591*1417</td>
<td>467 KB</td>
</tr>
</tbody>
</table>

Table 5 - 2 Testing machines, and files specifications (Tasks order experiment)

Practically, we investigated five cases over two different machines for each scenario. The properties and specifications of the used images and machines are detailed in Appendix F.

The observations for this experimental phase were divided into two parts: i) Distribution, and ii) Retrieve. For Distribution case 4 (File Encryption, File Split, and Segments compression) did the job faster than the other configurations, and the variation in files size did not have any significant implication on the speed. Practically, there was no significant difference in terms of speed for configuration (1-4). In terms of the consumed resources, configurations (1,2,5) consumed the machine resources more than the others and occupied the RAM longer than other configurations. By moving to the retrieve part, the configurations that occupied the machine resources more than the others were (8, 11); also, they did not complete the implemented job successfully for all tests. On the other hand, the (9, 10) configurations were the fastest, and capable of completing the jobs in a successful way. There was no significant
implication regarding segments size; also, the consumption of storage and processor were the lowest, the detailed collected data for both parts are in Table 5 - 3.

<table>
<thead>
<tr>
<th>Machine 1</th>
<th>Distribution</th>
<th>Retrieve</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time (Sec)</td>
<td>CPU</td>
</tr>
<tr>
<td></td>
<td>configuration 1</td>
<td>configuration 7</td>
</tr>
<tr>
<td>File 1</td>
<td>16.021</td>
<td>44%</td>
</tr>
<tr>
<td>File 2</td>
<td>19.039</td>
<td>46%</td>
</tr>
<tr>
<td>File 3</td>
<td>20.005</td>
<td>46%</td>
</tr>
<tr>
<td>File 4</td>
<td>24.032</td>
<td>51%</td>
</tr>
<tr>
<td>File 5</td>
<td>26.440</td>
<td>51%</td>
</tr>
<tr>
<td></td>
<td>configuration 2</td>
<td>configuration 8</td>
</tr>
<tr>
<td>File 1</td>
<td>18.497</td>
<td>42%</td>
</tr>
<tr>
<td>File 2</td>
<td>19.110</td>
<td>44%</td>
</tr>
<tr>
<td>File 3</td>
<td>22.346</td>
<td>44%</td>
</tr>
<tr>
<td>File 4</td>
<td>23.101</td>
<td>47%</td>
</tr>
<tr>
<td>File 5</td>
<td>23.982</td>
<td>48%</td>
</tr>
<tr>
<td></td>
<td>configuration 3</td>
<td>configuration 9</td>
</tr>
<tr>
<td>File 1</td>
<td>17.030</td>
<td>30%</td>
</tr>
<tr>
<td>File 2</td>
<td>18.002</td>
<td>37%</td>
</tr>
<tr>
<td>File 3</td>
<td>18.977</td>
<td>37%</td>
</tr>
<tr>
<td>File 4</td>
<td>20.340</td>
<td>42%</td>
</tr>
<tr>
<td>File 5</td>
<td>21.600</td>
<td>42%</td>
</tr>
<tr>
<td></td>
<td>configuration 4</td>
<td>configuration 10</td>
</tr>
<tr>
<td>File 1</td>
<td>13.46</td>
<td>19%</td>
</tr>
<tr>
<td>File 2</td>
<td>14.211</td>
<td>22%</td>
</tr>
<tr>
<td>File 3</td>
<td>15.672</td>
<td>22%</td>
</tr>
<tr>
<td>File 4</td>
<td>16.366</td>
<td>25%</td>
</tr>
<tr>
<td>File 5</td>
<td>17.683</td>
<td>25%</td>
</tr>
<tr>
<td></td>
<td>configuration 5</td>
<td>configuration 11</td>
</tr>
<tr>
<td>File 1</td>
<td>21.033</td>
<td>41%</td>
</tr>
<tr>
<td>File 2</td>
<td>26.004</td>
<td>41%</td>
</tr>
<tr>
<td>File 3</td>
<td>33.870</td>
<td>43%</td>
</tr>
<tr>
<td>File 4</td>
<td>38.102</td>
<td>46%</td>
</tr>
<tr>
<td>File 5</td>
<td>40.442</td>
<td>47%</td>
</tr>
<tr>
<td></td>
<td>configuration 6</td>
<td>configuration 12</td>
</tr>
<tr>
<td>File 1</td>
<td>28.311</td>
<td>35%</td>
</tr>
<tr>
<td>File 2</td>
<td>34.252</td>
<td>35%</td>
</tr>
<tr>
<td>File 3</td>
<td>38.621</td>
<td>36%</td>
</tr>
<tr>
<td>File 4</td>
<td>43.335</td>
<td>38%</td>
</tr>
<tr>
<td>File 5</td>
<td>49.728</td>
<td>38%</td>
</tr>
<tr>
<td></td>
<td>Distribution</td>
<td>Retrieve</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td>Time (Sec)</td>
<td>CPU</td>
</tr>
<tr>
<td>configuration 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 1</td>
<td>19.233</td>
<td>39%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 2</td>
<td>20.880</td>
<td>39%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 3</td>
<td>21.001</td>
<td>40%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 4</td>
<td>25.030</td>
<td>40%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 5</td>
<td>27.401</td>
<td>41%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>configuration 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 1</td>
<td>20.910</td>
<td>42%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 2</td>
<td>22.144</td>
<td>44%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 3</td>
<td>26.311</td>
<td>44%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 4</td>
<td>28.303</td>
<td>47%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 5</td>
<td>30.318</td>
<td>48%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>configuration 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 1</td>
<td>20.101</td>
<td>30%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 2</td>
<td>22.502</td>
<td>36%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 3</td>
<td>failure</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 4</td>
<td>failure</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 5</td>
<td>failure</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>configuration 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 1</td>
<td>15.221</td>
<td>20%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 2</td>
<td>16.741</td>
<td>20%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 3</td>
<td>17.020</td>
<td>21%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 4</td>
<td>18.561</td>
<td>22%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 5</td>
<td>19.455</td>
<td>22%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>configuration 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 1</td>
<td>36.200</td>
<td>40%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 2</td>
<td>42.193</td>
<td>40%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 3</td>
<td>48.363</td>
<td>42%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 4</td>
<td>55.223</td>
<td>43%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 5</td>
<td>failure</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>configuration 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 1</td>
<td>38.228</td>
<td>36%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 2</td>
<td>47.220</td>
<td>37%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 3</td>
<td>49.608</td>
<td>37%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 4</td>
<td>58.784</td>
<td>39%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 5</td>
<td>69.449</td>
<td>40%</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 5 - 3 Task order experiment outcomes
5.3.2 Encryption

The aim of these experiments was to investigate the performance and the compatibility of the used encryption systems over the defined smart devices. During these tests the encryption and decryption tasks were performed over different machine, the properties and specifications of the used images and machines are detailed in Table 5-4. There was no background process while the tasks were implemented, and that was aimed at avoiding occupying the machine resources by other tasks; also, all the tests were performed over the same physical machine.

<table>
<thead>
<tr>
<th>Virtual Device Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS version</td>
</tr>
<tr>
<td>Device Name</td>
</tr>
<tr>
<td>RAM</td>
</tr>
<tr>
<td>Processor</td>
</tr>
<tr>
<td>Storage</td>
</tr>
<tr>
<td>Screen</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Files Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

Table 5-4 Testing machines, and files specifications (encryption)
As mentioned earlier, these tests were not aiming to conduct security analysis, statistical analysis, or sensitivity analysis for these two encryptions algorithms, which had already proved their effectiveness and strength through several studies and investigations conducted by other researchers and developers. The tested file size increased gradually, the first three files were encrypted and decrypted successfully for both algorithms. However, the logistic map did not encrypt the rest of the files (4 - 8); while AES completed the encryption and decryption successfully, Table 5 - 5 shows the observations from these tests. The machine processor was occupied by about 86 % during logistic map implementation before unsafe termination. Besides that, the logistic map did not work as expected in terms of speed and consumed resources.

<table>
<thead>
<tr>
<th>Encryption</th>
<th>Time (Sec)</th>
<th>CPU</th>
<th>RAM</th>
<th>Encryption</th>
<th>Time (Sec)</th>
<th>CPU</th>
<th>RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>File 1</td>
<td>1.022</td>
<td>69%</td>
<td>18%</td>
<td>File 1</td>
<td>4.081</td>
<td>28%</td>
<td>22%</td>
</tr>
<tr>
<td>File 2</td>
<td>1.901</td>
<td>61%</td>
<td>18%</td>
<td>File 2</td>
<td>6.002</td>
<td>29%</td>
<td>26%</td>
</tr>
<tr>
<td>File 3</td>
<td>2.455</td>
<td>71%</td>
<td>20%</td>
<td>File 3</td>
<td>12.010</td>
<td>30%</td>
<td>27%</td>
</tr>
<tr>
<td>File 4</td>
<td>Failure</td>
<td></td>
<td></td>
<td>File 4</td>
<td>13.466</td>
<td>31%</td>
<td>28%</td>
</tr>
<tr>
<td>File 5</td>
<td>Failure</td>
<td></td>
<td></td>
<td>File 5</td>
<td>14.988</td>
<td>32%</td>
<td>31%</td>
</tr>
<tr>
<td>File 6</td>
<td>Failure</td>
<td></td>
<td></td>
<td>File 6</td>
<td>16.253</td>
<td>33%</td>
<td>34%</td>
</tr>
<tr>
<td>File 7</td>
<td>Failure</td>
<td></td>
<td></td>
<td>File 7</td>
<td>18.066</td>
<td>36%</td>
<td>36%</td>
</tr>
<tr>
<td>File 8</td>
<td>Failure</td>
<td></td>
<td></td>
<td>File 8</td>
<td>20.002</td>
<td>39%</td>
<td>38%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Encryption</th>
<th>Time (Sec)</th>
<th>CPU</th>
<th>RAM</th>
<th>Encryption</th>
<th>Time (Sec)</th>
<th>CPU</th>
<th>RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>File 1</td>
<td>3.300</td>
<td>71%</td>
<td>32%</td>
<td>File 1</td>
<td>9.361</td>
<td>30%</td>
<td>35%</td>
</tr>
<tr>
<td>File 2</td>
<td>3.988</td>
<td>73%</td>
<td>37%</td>
<td>File 2</td>
<td>11.685</td>
<td>31%</td>
<td>36%</td>
</tr>
<tr>
<td>File 3</td>
<td>4.501</td>
<td>73%</td>
<td>39%</td>
<td>File 3</td>
<td>14.911</td>
<td>33%</td>
<td>36%</td>
</tr>
<tr>
<td>File 4</td>
<td>Failure</td>
<td></td>
<td></td>
<td>File 4</td>
<td>18.633</td>
<td>35%</td>
<td>38%</td>
</tr>
<tr>
<td>File 5</td>
<td>Failure</td>
<td></td>
<td></td>
<td>File 5</td>
<td>23.662</td>
<td>36%</td>
<td>40%</td>
</tr>
<tr>
<td>File 6</td>
<td>Failure</td>
<td></td>
<td></td>
<td>File 6</td>
<td>28.108</td>
<td>36%</td>
<td>41%</td>
</tr>
<tr>
<td>File 7</td>
<td>Failure</td>
<td></td>
<td></td>
<td>File 7</td>
<td>34.225</td>
<td>37%</td>
<td>42%</td>
</tr>
<tr>
<td>File 8</td>
<td>Failure</td>
<td></td>
<td></td>
<td>File 8</td>
<td>38.004</td>
<td>37%</td>
<td>42%</td>
</tr>
</tbody>
</table>
5.3.3 Performance

The aim of these experiments was to investigate the performance the developed prototype over the defined smart device. During these tests the distribution and retrieval tasks were performed over different formats and sizes of images, the properties and specifications of the used images and machine are detailed in Table 5 - 6. There was no background process while the tasks were implemented, and that was aimed at avoiding occupying the machine resources by other tasks; also, all the tests were performed over the same physical machine.

<table>
<thead>
<tr>
<th>Decryption</th>
<th>Time (Sec)</th>
<th>CPU</th>
<th>RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>File 1</td>
<td>6.122</td>
<td>83%</td>
<td>33%</td>
</tr>
<tr>
<td>File 2</td>
<td>8.132</td>
<td>86%</td>
<td>36%</td>
</tr>
<tr>
<td>File 3</td>
<td>11.966</td>
<td>86%</td>
<td>39%</td>
</tr>
<tr>
<td>File 4</td>
<td>Failure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 5</td>
<td>Failure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 6</td>
<td>Failure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 7</td>
<td>Failure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 8</td>
<td>Failure</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Decryption</th>
<th>Time (Sec)</th>
<th>CPU</th>
<th>RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>File 1</td>
<td>9.094</td>
<td>27%</td>
<td>29%</td>
</tr>
<tr>
<td>File 2</td>
<td>11.399</td>
<td>29%</td>
<td>30%</td>
</tr>
<tr>
<td>File 3</td>
<td>18.300</td>
<td>30%</td>
<td>31%</td>
</tr>
<tr>
<td>File 4</td>
<td>20.847</td>
<td>31%</td>
<td>32%</td>
</tr>
<tr>
<td>File 5</td>
<td>26.411</td>
<td>31%</td>
<td>34%</td>
</tr>
<tr>
<td>File 6</td>
<td>33.647</td>
<td>32%</td>
<td>37%</td>
</tr>
<tr>
<td>File 7</td>
<td>37.302</td>
<td>33%</td>
<td>39%</td>
</tr>
<tr>
<td>File 8</td>
<td>40.305</td>
<td>36%</td>
<td>40%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Decryption</th>
<th>Time (Sec)</th>
<th>CPU</th>
<th>RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>File 1</td>
<td>9.787</td>
<td>80%</td>
<td>39%</td>
</tr>
<tr>
<td>File 2</td>
<td>12.332</td>
<td>81%</td>
<td>43%</td>
</tr>
<tr>
<td>File 3</td>
<td>15.009</td>
<td>81%</td>
<td>45%</td>
</tr>
<tr>
<td>File 4</td>
<td>Failure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 5</td>
<td>Failure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 6</td>
<td>Failure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 7</td>
<td>Failure</td>
<td></td>
<td></td>
</tr>
<tr>
<td>File 8</td>
<td>Failure</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Decryption</th>
<th>Time (Sec)</th>
<th>CPU</th>
<th>RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>File 1</td>
<td>20.036</td>
<td>33%</td>
<td>29%</td>
</tr>
<tr>
<td>File 2</td>
<td>27.322</td>
<td>36%</td>
<td>30%</td>
</tr>
<tr>
<td>File 3</td>
<td>34.966</td>
<td>36%</td>
<td>31%</td>
</tr>
<tr>
<td>File 4</td>
<td>38.741</td>
<td>40%</td>
<td>32%</td>
</tr>
<tr>
<td>File 5</td>
<td>40.449</td>
<td>41%</td>
<td>32%</td>
</tr>
<tr>
<td>File 6</td>
<td>46.020</td>
<td>43%</td>
<td>36%</td>
</tr>
<tr>
<td>File 7</td>
<td>50.090</td>
<td>44%</td>
<td>38%</td>
</tr>
<tr>
<td>File 8</td>
<td>58.771</td>
<td>44%</td>
<td>38%</td>
</tr>
</tbody>
</table>

Table 5 - 5 Encryption experiment outcomes

<table>
<thead>
<tr>
<th>Virtual Device Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS version</td>
</tr>
<tr>
<td>Device Name</td>
</tr>
<tr>
<td>RAM</td>
</tr>
<tr>
<td>Processor</td>
</tr>
<tr>
<td>Storage</td>
</tr>
<tr>
<td>Screen</td>
</tr>
</tbody>
</table>
As mentioned earlier, these tests aim to test and compare the performance of the developed prototype over a set of image files that vary in terms of formats and size as well. The tested file size increased gradually, all the tested files were distributed and retrieved successfully for all formats. However, the consumed time and the occupied CPU and RAM and that due to the file formats properties in terms of compression and depth, Table 5 - 7 shows the observations from these tests. The Tiff format files consumed 159.83 seconds to distribute the largest file, and that was the longest time. However, the occupied CPU and RAM percentage were too close to all conducted tests.

<table>
<thead>
<tr>
<th>File</th>
<th>PNG</th>
<th>JPG</th>
<th>TIFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>File 1</td>
<td>10.19 MB</td>
<td>12.83 MB</td>
<td>12.72 MB</td>
</tr>
<tr>
<td>File 2</td>
<td>12.73 MB</td>
<td>13.32 MB</td>
<td>13.53 MB</td>
</tr>
<tr>
<td>File 3</td>
<td>17.51 MB</td>
<td>17.3 MB</td>
<td>17.62 MB</td>
</tr>
<tr>
<td>File 4</td>
<td>18.87 MB</td>
<td>18.59 MB</td>
<td>18.76 MB</td>
</tr>
<tr>
<td>File 5</td>
<td>25.7 MB</td>
<td>26.11 MB</td>
<td>26.09 MB</td>
</tr>
<tr>
<td>File 6</td>
<td>35.2 MB</td>
<td>36.05 MB</td>
<td>35.97 MB</td>
</tr>
</tbody>
</table>

Table 5 - Testing machine, and files specifications (Performance)
5.3.4 User Experience

The second part of the conducted testing phase concerned the user experience. Using experience testing is a systematic approach that is utilised to ensure that the developed prototype meets the requirements from the end user perspective. During this test, the developed approach was tested by a group of users (50 participants), and they were asked to complete a questionnaire which measured specific criteria. More than 40% of the participants are aged between 20 and 30 years, and 7 are over 50 years old. 20% of the participants have Doctorates, and 12 have a Diploma (24%). More than 60% of the participants have less than 5 years’ experience, and two participants have worked in the field more than 15 years; Table 5 – 8 details all the participants information in terms of age, gender, year of experience, and qualifications. The measured criteria are: flexibility, usability, efficiency, learnability, and helpfulness (Bevan, 1995). The participants were asked to provide their evaluation regarding the tested prototype via a 1 to 5 scale, Table 5 - 9 shows the collected results.

<table>
<thead>
<tr>
<th>Item</th>
<th>Scale</th>
<th>Number</th>
<th>Percentage (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Age</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>20-30 years</td>
<td>20</td>
<td>40.0</td>
</tr>
<tr>
<td></td>
<td>30-40 years</td>
<td>15</td>
<td>30.0</td>
</tr>
<tr>
<td></td>
<td>40-50 years</td>
<td>8</td>
<td>16.0</td>
</tr>
<tr>
<td></td>
<td>&gt;50 years</td>
<td>7</td>
<td>14.0</td>
</tr>
<tr>
<td><strong>Gender</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Male</td>
<td>33</td>
<td>66.0</td>
</tr>
<tr>
<td></td>
<td>Female</td>
<td>17</td>
<td>34.0</td>
</tr>
<tr>
<td><strong>Qualification</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Doctorate Degree</td>
<td>10</td>
<td>20.0</td>
</tr>
<tr>
<td></td>
<td>Master Degree</td>
<td>15</td>
<td>30.0</td>
</tr>
<tr>
<td></td>
<td>Bachelor Degree</td>
<td>13</td>
<td>26.0</td>
</tr>
<tr>
<td></td>
<td>Diploma Certificate (Ordinary and Higher)</td>
<td>12</td>
<td>24.0</td>
</tr>
<tr>
<td><strong>Interest</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Education</td>
<td>8</td>
<td>16.0</td>
</tr>
<tr>
<td></td>
<td>Banking</td>
<td>11</td>
<td>22.0</td>
</tr>
<tr>
<td></td>
<td>Industry</td>
<td>9</td>
<td>18.0</td>
</tr>
<tr>
<td></td>
<td>IT</td>
<td>5</td>
<td>10.0</td>
</tr>
<tr>
<td></td>
<td>Military</td>
<td>4</td>
<td>8.0</td>
</tr>
<tr>
<td></td>
<td>Academic</td>
<td>9</td>
<td>18.0</td>
</tr>
<tr>
<td></td>
<td>Media</td>
<td>4</td>
<td>8.0</td>
</tr>
</tbody>
</table>
After analysing the collected data, the produced outcome provided a clear observation about the measured criteria. There was only 1 participant who was not satisfied with the prototype flexibility, and that represented 2%. On the other hand, 98% were satisfied with the delivered flexibility level. The end users were able to set up the preferred configuration, that helped prevent consuming the system resources and capabilities more than what was required. Moreover, that helped to avoid occupying unneeded resources, and prevented critical issues such as unsafe termination and availability. In terms of usability, the majority of participants felt that the tested prototype was user-friendly, and easy to use; 6% believed that prototype was not easy to use.

In addition, 96% of participants found the tested prototype was an efficient solution for data protection. There was 1 participant who found this approach inefficient. In terms of learnability, 47 out of 50 found the tested prototype easy to operate and learn, while only three participants faced some difficulties. Finally, only one participant found that the prototype was not capable of helping and facilitating data protection. On the other hand, 98% said the approach helped them make the data protection process easier.
5.4 Discussion

First of all, we start with the task order experiment; this experimental part aimed to define the optimal order for the tasks. For the distribution part, we found that configuration 4 was the most suitable configuration. This configuration completed the jobs successfully, saved time and the consumed resources as well. There was no significant implication for the file size increment. Also, there was no crash or failure recorded via the configuration. On the other hand, configurations 5 & 6 showed an obvious delay with large files and suffered from a number of uncompleted jobs, which were caused by the limitation of the consumed resources. For the retrieve job, the 9 & 10 configurations were the two configurations that completed the implemented jobs faster than the others, although the 9th configuration consumed and occupied more resources compared with configuration 10. For the first machine the 4th configuration was the best in terms of consumed time for all tested files, and it occupied about 20% of the provided CPU and RAM which saved these resources 15 % better than the other configurations. In addition, the 4th configuration completed the encryption faster than the other configurations, and consumed fewer resources as well; and that made it the optimal configuration for the encryption, due to its capability in performing the task successfully, and saving time and provided resources. Figure 5 - 1 & 5 - 2 illustrate the consumed time for the encryption over two different machines.
Figure 5.1 Consumed time for each configuration (Encryption) – Machine 1

Figure 5.2 Consumed time for each configuration (Encryption) – Machine 2
By moving to the decryption procedure, a number of configurations failed to complete the job successfully, as illustrated in Figures 5 - 3 & 5 - 4. The 10th configuration; which is the reversal order of the 4th configuration, was the fastest configuration, capable of completing the decryption jobs successfully for all encrypted files, and also saving operational costs in terms of the consumed CPU and RAM. Moreover, the 9th and 11th configurations had a number of failures and unsafe terminations, which happened while decrypting the 4th and 5th files over the 1st machine. To sum up, the most appropriate task order was encrypt, split, compress (for distribution), and decompress, join, decrypt (for retrieval); that was capable of performing the job in a successful way, and saving time, CPU, and RAM through the two utilised machines. In addition, the failure scenarios happened due to lack of resources and capabilities of the used environments, which were not capable of dealing with the increment in files and segment sizes.

![Figure 5 - 3 Consumed time for each configuration (Decryption) – Machine 1](image-url)
The second experimental part was aimed at investigating and analysing performance and compatibility of the utilised encryption algorithms to find out the most appropriate algorithm for the developed approach. We found that the logistic map algorithm failed to encrypt the selected 8 files successfully over both machines: Figure 5 - 5 illustrates the consumed time for each algorithm over the two machines through encryption. Obviously, the logistic map algorithm was not an appropriate algorithm to perform such a job over a limited resource platform. However, this algorithm was better than AES for Files 1, 2, and 3 in terms of the consumed time to complete the encryption; but this algorithm occupied many resources in terms of CPU and RAM. For instance, the logistic map occupied 96% and 71% while encrypting File 1 through machines 1 & 2 respectively, whilst AES used only 28% and 30% respectively; Figure 5 - 6 shows the consumed CPU for all files during files encryption.
Figure 5 - 5 Encryption time comparison

Figure 5 - 6 Encryption CPU consumption comparison
Similar to encryption, the logistic map failed to decrypt the encrypted files. There were three files that were decrypted successfully, and for these three files the logistic map was much better than AES in terms of consumed time; for instance, the logistic map needed only about 15 seconds to decrypt a file (1.24 MB) while AES consumed about 35 second to decrypt it over the same machine. Figure 5 - 7 illustrates the consumed time for each algorithm over the two machines through encryption. However, the logistic map occupied 86% and 81% while encrypting the same file through machines 1 & 2 respectively, whilst AES used only 30% and 36% respectively; Figure 5 - 8 shows the consumed CPU for all files during files encryption.

To sum up, the most appropriate for encryption and decryption was AES; that was capable of performing the job in a successful way, and save time, CPU and RAM through the two utilised machines. The logistic map algorithm might be more efficient over different platforms that have powerful resources and capabilities. Practically, the logistic map algorithm failed to complete the both experiments (encryptions & decryption) due to a set of reasons. For the successful experiments, the huge consumption of CPU and RAM are obvious, and that due to the internal process of such algorithm. The failure occurred due to shortage of the provided resources, as such algorithm generates key image, disorganised image, and encrypted image; all these images consume the CPU and occupy the provided RAM as well. The unsafe termination happened when the encryption and decryption reach the peak of the provided resources. Technically, the logistic map algorithm supposed to work perfectly over high end devices. One of the solutions to prevent this issue (unsafe termination) is implementing the internal process of the logistic map algorithm in sequential way instead of parallel way, which will help to prevent overload the CPU and RAM as well. However, that will increase the operational cost in terms of time and consumed resources, and that is acceptable solution for some cases. For the developed approach, one of our aims is reducing the operational cost, which will make it unaccepted solution. For that, this algorithm needs more investigation and development in terms of performance, and memory management over limited resources platforms.
Figure 5 - 7 Decryption time comparison

Figure 5 - 8 Decryption CPU consumption comparison
The third experimental part was aimed at investigating and comparing performance of the developed prototype over a set of images that vary in terms of size and format. We found that the approach worked as expected, and completed the both tasks (distribution and retrieval) successfully for all files; Figure 5 - 9 illustrates the consumed time for each file through distribution task. Obviously, PNG files were distributed faster than the other formats, and that due to the PNG properties. The increment of the consumed time during the distribution task was normal, and that due the files sizes increment. During the distribution task, there was no significant change of the occupied resources in terms of CPU and RAM. For instance, the PNG files consumed between 29% and 32% of CPU while TIFF files consumed between 31% and 34%. In terms of RAM, the occupied RAM vary between 33% and 44%; Figure 5 – 9 & 10 & 11 shows the consumed time, CPU, and RAM for all files during files distribution task.

![Distribution time comparison](image)

*Figure 5 - 9 Distribution time comparison*
**Figure 5 - 10 Distribution CPU consumption comparison**

**Figure 5 - 11 Distribution RAM consumption comparison**
Similar to distribution task, the revival task was completed successfully for all files. The retrieval tasks for PNG were completed faster than JPG, and TIFF files, and that due to compression feature; for instance, a PNG file (about 17.5 MB) needed about 100 seconds to be retrieved while TIFF consumed about 122 second to be retrieved over the same machine. Figure 5 - 12 illustrates the consumed time for each file through file retrieval task. However, there was no significant different in terms of CPU consumption for all the retrieved files, as the average was about 43%; for instance, File 1 (PNG) occupied 41%, whilst file 6 (TIFF) used only 44% and; Figure 5 - 13 shows the consumed CPU for all files during files retrieval task. In terms of RAM, the consumption for PNG files varied between 33% and 41%, and JPG files average consumption was about 43%; Figure 5 - 14 shows the consumed RAM for all files during files retrieval task. To sum up, the developed prototype was capable of performing the job in a successful way, and save time, CPU and RAM. However, the variation in the consumed time was normal, and expected as well, and that due to the properties of each file format.

![Graph of retrieved times for different file types](image-url)

**Figure 5 - 12 Retrieval time comparison**
Figure 5 - 13 Retrieval CPU consumption comparison

Figure 5 - 14 Retrieval RAM consumption comparison
For user experience, 98% of participants found the prototype flexible. The delivered flexibility level allows the end user to select the cloud storage service provider, select the security aspects that need to be preserved, and configure the approach performance based on the end user preferences. That level of flexibility helps to enhance the performance and consume the needed service, resources and capabilities only. Moreover, it will reduce the operational costs in terms of time and resources consumption. In addition, avoiding unsafe termination, and resource occupying will prevent behaviours that might influence the system in terms of security and availability. The negative feedback was not significant for all the criteria; the highest percentage was 4% for usability and learnability, while the lowest negative feedback was 2% for flexibility and helpfulness. The majority of participants were satisfied with the usability, which represents the clarity of functions, capabilities and design. This means the developed prototype is user-friendly, and the developed GUI is easy to navigate. On the other hand, the majority of participants found that the approach was efficient and that was because of a number of reasons; for instance, the approach is capable of preserving a set of security aspects, and allows the user to select the preserved security aspects based on the end user needs. Having 96% of the participants’ responses in the favour of efficiency proved that the developed prototype was capable of working as expected, and of completing the needed task in an easy and successful way. Providing a fast and strong cryptography system, simple and organised tasks, a high level of flexibility, and comprehensive solutions are the elements that enhance the approach efficiency. 94% of the participants said that the prototype was easy to adapt and the tasks can be implemented easily, which proves the ability to be used by individuals who are not IT expert or need advanced training. There were only three participants, who argued that they found this approach difficult to use, which was not significant. Finally, 98% of the participants argued that the approach worked as expected and the prototype helped to protect the targeted data based on the end user preferences and delivered the expected level of flexibility. Figure 5 - 15 illustrates the participants’ responses
in terms of criteria, while Figure 5 - 16 categorises the participant feedback into positive and negative.

![Figure 5 - 15 Statistical Results for User Experience](image)

![Figure 5 - 16 User Experience categorisation](image)
From the earlier discussion we can conclude that the aim of the conducted study has been achieved, and we produced a contribution to the body of knowledge that bridges the identified gaps. The developed approach addressed all the identified limitations, and helped to deliver a solution that is capable of overcoming these limitations to individuals. Next, the benchmarks for the conducted approach are presented, Table 5 - 10 compares the existing approaches against the proposed approached

- **Operational costs**

The approach helped to reduce the operational costs by utilising the appropriate multiple cloud computing paradigms, that can be used with no change in the existing infrastructure and thus avoid additional connections that might consume the provided resources in terms of bandwidth and hardware as well. In addition, the simplicity and the delivery format (executable file) of the developed approach helps to avoid an extra cost that might be needed due to installations, maintenance, and configuration.

- **Code execution**

The developed approach avoids any outsource code execution, as this was defined as a threat that might expose the data and the infrastructure as well. In addition, all the exchanged data is encrypted before leaving the end user machine, and no third party will have the decryption key; the data owner, and data owner only, owns the key.

- **Third party**

The developed approach prevents any third-party involvement, and there is no broker or mediator located between the end user and cloud service providers. This point will help to reduce the operational costs, avoid dependencies and enhance the availability of the delivered approach, as well as maintain the data integrity and confidentiality.
During our approach only the data owner and cloud storage service provider are able to access the stored data, and the data owner only can read and write the data.

- **Portability**

Supporting portability is one of the critical limitations that was identified earlier. Our approach supports portability and allows the end user to run and consume the provided service through the preferred platform. The developed approach is presented to the end user in an executable format that needs no high level of knowledge to operate. In addition, the approach allows movement of the database of protected data from one machine to another, in order to enhance the portability level and facilitate disaster recovery.

- **Flexibility**

The developed approach provides a high level of flexibility to the end user in terms of selecting the cloud storage providers, the preserved security features and the used encryption algorithms. Enhancing the flexibility helps to perform the job based on the end user need and capabilities. In addition, it helps to save the provided resources being consumed and occupied with tasks of little worth.

- **Multiple-cloud paradigm**

As discussed earlier, the existing approaches utilised a multiple-cloud paradigm that needs a prior agreement between the collaborated cloud storage services and needs a high level of experience to be operated, configured, and maintained. In addition, theses paradigms are not right for individuals, for various reasons discussed earlier. Our approach utilised a paradigm that isolates the consumed cloud storage service and avoids any connection between these cloud storages, and avoids third party involvement as well. The multi-cloud libraries paradigm helps to reduce the operational
costs, enhance the service provider diversity, availability, and improve security by preventing the need for a third party.

- Usability

The developed approach avoided complicity and lowered the needed level of knowledge in order to allow individuals to consume that approach. The absence of any need for training to consume that approach is a critical point that proves the ease and usability of this approach. The conducted user experience experiments and the collected feedback show that 94% of the participants said the prototype was easy to adapt and the tasks can be implemented easily, which proves its ability to be used by individuals, who are not IT expert or need advanced training; only three participants found this approach difficult to use, which was not significant.

- Preserved security features

The developed approach is designed to be capable of preserving several security aspects in one single approach, the technical details for each aspect were discussed earlier. In addition, the developed approach prevents the need to combine more than one approach, in order to enhance security, which helps to reduce the operational costs, decrease the needed level of knowledge, and prevent any threats that might occur due to conflict between approaches.

On the other hand, a number of findings, challenges and limitations were faced during the research journey, which must be addressed in order to provide clear focal points for new research, and direction for future work as well. These raised issues, and the challenges are discussed in the following section.
<table>
<thead>
<tr>
<th>Approach</th>
<th>Availability</th>
<th>Authentication</th>
<th>Integrity</th>
<th>Confidentiality</th>
<th>Cost</th>
<th>Computing Environment</th>
<th>Multiple-cloud paradigm</th>
<th>Configuration</th>
<th>Portability</th>
<th>Third party involvement</th>
<th>Platform diversity</th>
<th>Outsource code execution</th>
<th>Auditing</th>
<th>Used by</th>
<th>Specialty</th>
</tr>
</thead>
<tbody>
<tr>
<td>BFT</td>
<td>✔</td>
<td></td>
<td>✔</td>
<td>High</td>
<td>Medium</td>
<td>Distributed / cloud</td>
<td>Federated/Intra</td>
<td>Difficult</td>
<td>✔</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>Enterprise</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>HAIL</td>
<td>✔</td>
<td>✔</td>
<td></td>
<td>Medium</td>
<td>Cloud</td>
<td>Federated</td>
<td>Medium</td>
<td>×</td>
<td>✔</td>
<td>×</td>
<td>×</td>
<td>✔</td>
<td>Enterprise</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>RACS</td>
<td>✔</td>
<td></td>
<td>High</td>
<td>Cloud</td>
<td>Federated</td>
<td>Difficult</td>
<td></td>
<td>×</td>
<td>✔</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>Enterprise</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>IC Storage</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>High</td>
<td>Cloud</td>
<td>Intra</td>
<td>Difficult</td>
<td>×</td>
<td>✔</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>Enterprise</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Tahoe-LAFS</td>
<td>✔</td>
<td>✔</td>
<td>High</td>
<td>Distributed / Cloud</td>
<td>Federated</td>
<td>Difficult</td>
<td></td>
<td>×</td>
<td>✔</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>Enterprise</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>CloudProof</td>
<td>✔</td>
<td></td>
<td>High</td>
<td>Cloud</td>
<td>Federated</td>
<td>Difficult</td>
<td></td>
<td>×</td>
<td>✔</td>
<td>×</td>
<td>×</td>
<td>✔</td>
<td>Enterprise</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>IRIS</td>
<td>✔</td>
<td></td>
<td>High</td>
<td>Cloud</td>
<td>Federated</td>
<td>Difficult</td>
<td></td>
<td>×</td>
<td>✔</td>
<td>×</td>
<td>×</td>
<td>✔</td>
<td>Enterprise</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Dep-Sky</td>
<td>✔</td>
<td></td>
<td>High</td>
<td>Cloud</td>
<td>Federated</td>
<td>Medium</td>
<td></td>
<td>×</td>
<td>✔</td>
<td>×</td>
<td>×</td>
<td>✔</td>
<td>Enterprise</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>Our approach</td>
<td>✔</td>
<td>✔</td>
<td>✔</td>
<td>Low</td>
<td>Cloud</td>
<td>multi</td>
<td>Easy</td>
<td>✔</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>✔</td>
<td>Individual</td>
<td>×</td>
<td></td>
</tr>
</tbody>
</table>

Table 5 - 10 Existing approaches and developed approach comparison
5.5 Challenges and Raised Issues

From the collected feedback we could conclude that the developed prototype performance is as expected and the developed prototype has been validated and verified. In addition, the proposed approach, developed prototype, and testing feedback provide a useful contribution. However, there are some points that have been highlighted, which might be considered as limitations. One of the stated contributions was defining the study limitations and recommendations for future work, and outlining future contributions. The collected feedback will be used to improve the designed approach in terms of usability, performance, and functionality. These suggestions, drawbacks or failures will be considered during the development process, which will help to prevent and overcome these issues, and enhance the system performance. The raised issues, and challenges are listed below:

- The supported encryption algorithms need to be expanded, to enhance flexibility and meet the end-user requirements and capabilities.

- Expand the supported cloud storage service providers and allow users to add personal cloud storage.

- The operational costs of the consumed time and power must be reduced, which means that mechanisms for data fragmentation and data processing need to be improved. The approach needs to be optimised and enhanced throughout.

- Duplicating the system database and having dual insert commands for both databases could improve auditing. The auditing database only supports the insert and select commands, but is used as a solid record that illustrates the conducted protection process in detail.
- Optimising the protection preferences must be considered. Implementing overqualified preferences could lead to system crashes. In addition, selecting such preferences does not provide the expected feasibility.

- Adding more mechanisms for verifying the user as well as the dynamic keypad, such as utilising fingerprints, is helpful, especially for capable machines.

- To enhance the developed approach and the delivered service, after granting the required privileges to the data owner, the catch errors or unexpected incidents need to be recorded and provided to the developers.
5.6 Summary

Throughout this chapter the conducted tests and experiments have been discussed. First of all, the software testing phase has been discussed in terms of importance and environment. Additionally, the conducted experiments have been discussed in terms of purpose, design, participants, and measured aspects and then the collected responses were presented, analysed, and discussed. After discussing the collected data, we presented how our approach helped to fill the defined gaps and overcome the limitations from previous works. Finally, the last contribution of this study was achieved via defining the disadvantages, difficulties, concerns, raised issues, and suggestions that have been presented as recommendation for future work, outlining future contributions.
Chapter VI. Conclusion and Future Work

6.1 Overview

Cloud computing is a developed and growing market that can been considered as an optimum solution for several reasons (see Section II). As discussed, the cloud computing market size increases in a continuous way, and that increases the need for developing and research to enhance the provided services and overcome the existing obstacles, challenges, and limitations. As presented in (Section 2.4), one of the core obstacles that prevents shifting from traditional computing solutions to cloud computing solutions was information security. Moreover, recent data breaches that occurred for a number of cloud computing storage services and led to exposing private information, raised serious concerns about cloud security.

As discussed in the systematic literature review (Section 2.5), the majority of the defined security issues were associated with access and computational aspects; these two categories have about 50% of the security issues. As a result, cloud computing access control security issues and cloud computing computation and storage security issues must gain more attention from researchers and developers in both academic and industrial fields. The authentication, applied cryptographic system and storage data issues have the largest share of the identified security issues (Section 2.5.3).

As presented in (2.3), the multiple cloud computing concept was developed to solve a set of issues and overcome limitations. There was a need to develop a criteria that can help to categorise and distinguish between the existing multiple cloud computing paradigms, because of the existing vagueness in terms of architecture, features, capabilities, requirements, and limitations. The used criteria helped the paradigms to be understood, and allowed the most appropriate paradigm for the user to be selected (Section 2.3.2).
As presented in (Section 2.6.2), the critical literature review on existing multiple-cloud security approaches was implemented to highlight any limitations and gaps in the research. We found that the studied approaches varied between interconnection protocols, internal layers or general algorithms, which led to increasing the complexity of the consumed system, and required a high level of expertise for configuration and implementation. In addition, these approaches were developed to be implemented over multiple-cloud paradigms that require high levels of cooperation between the involved clouds. No such approach has been developed to be consumed over multi-clouds libraries paradigms, or to be consumed by individuals. Moreover, the end user must combine more than one approach to preserve data security, which is not an efficient solution, because of conflict between these approaches, complexity, and increment in operational costs.

To fill the identified gap and overcome the highlighted limitations, we developed an approach that could provide a secure cloud storage service for individuals via multiple-clouds. The proposed approach relies on firstly: dividing, encrypting and storing (distributing) data in several protected multi-cloud storages, and secondly: storing one segment only on the end user machine. Using this approach, the multi-cloud libraries paradigm has been selected as the most suitable for delivering the requested service as expected, because it allows the consumption of cloud storage services that are independent and isolated from each other (Section III).

As presented in (IV), the approach prototype was developed that was aimed at implementing, verifying, validating and testing the core features of the proposed approach. At this stage, there were several tests that had different purposes. These tests helped to discover the most suitable order for the approach tasks, investigate the utilised cryptosystem efficiency, measure the prototype performance, and collect end user experience feedback. The collected responses from these tests show that the developed prototype works as expected, and the prototype behaviour and performance match with the proposed approach. However, there are
a number of obstacles that will be considered to improve the developed approach performance, behaviour, functionality, and usability (Section V). Finally, the developed approach has a number of benefits; these benefits are as follows:

- Reduces operational costs: the developed approach helps to reduce the operational costs in terms of the consumed bandwidth, training, broker, and cloud storage. The approach interacts with the consumed cloud storage without any prior agreement, or third-party involvement. In addition, the automation level minimises the need for human interaction with the systems, and prevents threats that might arise, due to lack of good practice.

- Flexibility: the developed approach allows the end user to configure the preference of the wanted service, and fully control the security level based on capabilities and requirements. That level of flexibility helps to save the provided resources, and avoids occupying or performing unwanted tasks.

- Avoids dependency: the developed approach utilised the multiple cloud computing concept that was developed to overcome the vendor lock-on issue. The end user will not depend on a single service provider, and will be able to use the delivered service even if there is a lack of availability.

- Preserves security: the developed approach is capable of preserving several security aspects in one approach without any need to combine it with other approaches. In addition, the end user can select the feature that needs to be maintained via the approach.

- Avoids third party: the utilised multiple cloud computing paradigm for the developed approach is the multi-cloud paradigm (libraries). This paradigm avoids relying on a third party to manage and control the consumed services; we avoid third party involvement because it was stated as one of the obstacles in shifting to cloud
computing. In addition, losing availability of the third party is a serious issue that must be avoided.

- Portability: supporting portability, and allowing the end user to consume the delivered services over various environments, is one of the benefits that will be provided. The end user can move the data, and all the protected data information, from one platform to another and consume the service smoothly; this feature will help with disaster recovery, losing the previous platform availability, or moving to a new machine.

- Diversity: the developed approach is not limited to a specific infrastructure, operating system or cloud service provider. The approach can be installed, configured, and consumed via various platforms. In addition, it allows the end user to select the preferred cloud service providers that meet their requirements.

- Usability: the developed approach will be delivered and consumed as an executable file, which will help to enhance the usability level, and avoid any mistake that might happen due to complexity and lack of good practice.
6.2 **ADDRESSING STUDY OBJECTIVES**

The goal of the conducted research was to investigate the existing approaches to improve the security of the delivered cloud storage service via multiple-cloud computing, and to propose an approach for providing a secure data storage system that could be installed, configured, and easily consumed by individuals through an appropriate multiple-cloud model. The proposed approach maintains the confidentiality, integrity and authenticity of the protected data; besides that, it supports disaster recovery and auditing for the system. The developed approach aims to reduce the needed technical experience, the complexity, and improve the flexibility of the delivered service. In order to accomplish the research aim in a clear, accurate, reliable, valid, and successful way, there were a set of research objectives that have been defined and divided into two phases.

The objectives of the 1st phase were concerned with highlighting the research gaps and developing a framework that aimed at overcoming these gaps; these objectives were formulated as follows:

- Conduct an extensive literature review that investigates existing cloud computing security issues and challenges, evaluates the existing multiple-cloud security approaches, and compares the existing multiple-cloud models to highlight their features, capabilities, requirements and architecture, to clearly identify the research gaps:
  
  - A systematic literature review that defined and prioritised cloud computing security issues and challenges was implemented (Chapter II).
  
  - A critical literature review was conducted; that study investigated, analysed, and compared the existing security approach (Chapter II). This study defined
the limitations, capabilities, mechanisms, drawbacks, and challenges, which helped to define the gaps and limitations that must be filled and solved.

- A comparative study between the existing multiple cloud computing paradigms was developed (Chapter II). This study proposed a set of criteria that helped to categorise the paradigms and underlined their features, capabilities, requirements and architecture; this helped to utilise the most appropriate paradigm for the developed approach.

- Propose an approach that utilises the multiple-cloud computing model to preserve data confidentiality, integrity, authenticity and availability; which prevents third party involvement. In addition, avoid dependency between cloud service provider and distributing the data over a set of different geographical locations will prevent vendor-lock issue, and un-authorised access as well. Besides that, the approach must enhance the internal process automation, and minimise the interaction with the end-user in order to avoid issues that might happen due lack of experience or bad practice, and addresses the gaps identified through the literature review:

- An approach that utilised multiple cloud computing was developed (Chapter III). The developed approach was simple, flexible, easy to install, consume, and be maintained by individuals. In addition, the approach guaranteed the protected data integrity, confidentiality, authenticity, and availability.

The second set of objectives were for the 2nd phase that was concerned with developing the prototype; these objectives were formulated as follows:

- Develop an android application as a protype for the developed approach (1st Stage), which will be used by individuals to protect the data through multiple-cloud storage.
An android application was developed as a prototype of the proposed approach (Chapter IV). This prototype utilised a set of cloud storage service providers and cryptographic systems to maintain the delivered security features.

- Implement extensive testing of the prototype that will be used to verify and validate the developed prototype, and produce results that will be used to evaluate, and improve that approach.

- The order of the approach tasks was tested, investigated, and compared (Chapter V). These sets of tests aimed to define the most appropriate order for the tasks; during these tests the distribute and retrieve procedures were covered.

- The approach performance was tested over a set of defined scenarios (Chapter V); this test investigated the performance and behaviour of the prototype, and defined the drawbacks, mistakes, and limitations.

- The cryptosystem performance over an android environment was tested (Chapter V); the goal of this test was to discover, measure, and compare the performance and compatibility over that environment.

The end user experience test was implemented (Chapter V); the goal of this test was to ensure that the developed prototype meets the requirements from the end user perspective. The conducted test measured the flexibility, usability, efficiency, learnability, and helpfulness. By achieving the research goal, and completing the defined study objectives, we can conclude that there was a set of contributions were added to the body of the knowledge. These contributions can be listed as follows:
- An updated study that illustrates the security issues, challenges and vulnerabilities associated with cloud computing, and insight into how these issues have been addressed by the proposed research.

- A comprehensive and critical study of existing approaches for securing data in multiple-cloud environment, identifying their capabilities, but also areas for improvement that have been applied in the context of this research, and using this to design and develop an approach that bridges the existing gaps.

- Identifying criteria, categorising, and comparing existing multiple-cloud models that can be used to evaluate these models to select the most appropriate model for the approach.

- Identifying the need for developing an approach capable of protecting end-user data in a multiple-cloud environment that requires no agreement between cloud storage providers, which represents a contribution to the field of research.

- Developing a new framework that utilises a multiple-cloud model that provides personal data protection by distributing the data segments over these clouds and storing one segment on the user machine to prevent any decryption/modification attempts from outside the user machine.

- Developing an android mobile application that uses the developed approach, thus providing evidence of the real-world application of the model, and allowing new knowledge and insight to be gained.

- Developing a road map that illustrates the study limitations and recommendations for future work, outlining future contributions.
6.3 Research Implications

The continuous increment in the cloud computing market, and growth in consumption of the delivered services via various platforms, makes that field a target for various attacks. Practically, efforts from the vendors, researchers, and developers need to be increased to prevent and avoid these attacks and enhance the quality of the delivered services.

Through the conducted investigation, we found that multi-cloud computing paradigms can overcome and solve some of the defined security issues, and can be utilised as effective solutions that are capable of preserving specific security features. Multiple cloud emerged as a solution for a set of issues, such as availability and avoiding vendor lock. However, it was difficult and vague to distinguish between the various names and models that were defined through the literature. For that, we came up with criteria that aimed at facilitating understanding the capabilities, drawbacks, and requirements for each paradigm.

In the context of the academic field, an important point that must be highlighted is that the prototype was developed for testing purposes, and to validate the developed approach; besides that, the contribution of the conducted research was not limited to the prototype. The developed approach worked as expected and met the defined requirements. However, the chaotic encryption used suffered from failures and drawbacks, and that was due to compatibility with the used platform and the provided resources. This point was defined as a direction that needs more investigation and was suggested as future work due to time limitation and the scope of this research.
6.4 STUDY LIMITATIONS

As in other research, this study has limitations. Despite supervisor support and the continuous work to solve some of the problems, not all could be solved due to time restrictions. However, the defined limitations are discussed in this section.

First of all, the involvement of cloud storage providers was limited, as we were restricted to the storage services that have APIs and are capable of being implemented on an android platform. In addition, the programming language used limits the range of available cloud storage. Expanding the provided choices of cloud storages services will help to enhance the delivered level of flexibility, and allow end users to consume the preferred cloud storage service. In addition, cloud storage selection and load balance between the utilised cloud storages must be investigated, in order to develop a mechanism capable of selecting the most appropriate cloud storage.

Moreover, the provided cryptography needed to be investigated to ensure efficiency in terms of security, speed and performance. However, the algorithms must be tested and evaluated to verify their effectiveness in a smart device environment, in terms of consumed power, CPU and memory requirements, and this limitation can be considered as future work.

In terms of testing, the number of testing participants was sufficient and met statistical requirements. However, increasing their numbers would enhance the feedback and increase diversity, helping to explore and study feedback from various perspectives. Also, the developed prototype was limited to android which represents most of the smart device market, according to the statistics; however, other supported platforms must be tested.

Finally, developing a complete Beta version in the real world instead of a prototype is important, as this would help to validate and verify the approach in terms of compatibility and integration between internal components, and help to investigate the approach efficiency, and compare it with other approaches.
6.5 RECOMMENDATIONS FOR FUTURE RESEARCH

As we discussed earlier, the conducted research aimed to investigate the existing approaches to improve the security of the delivered cloud storage service via multiple-cloud computing, and to propose an approach for providing a secure data storage system that could be installed, configured, and easily consumed by individuals through the appropriate multiple-cloud model. The proposed approach maintains the confidentiality, integrity, and authenticity of the protected data; besides that, it supports disaster recovery and auditing for the system. The developed approach aims to reduce the needed technical experience, complexity, and improve the flexibility of the delivered service as well. However, further investigation, development, and testing must be conducted to present a comprehensive, robust, and effective solution. In the following section, we discuss a number of points for future work.

First of all, there is a need to develop a Beta version that supports all the developed features. This will help examine the effectiveness of the developed approach in the real world, and improve the performance. Also, investigating the approach performance, in terms of the consumed resources and time, can be considered as a direction. Moreover, Mobile Cloud Computing is a growing market that needs more research and development from academics and industry. For the final version, the approach must utilise some smart device capabilities to enhance the security of the delivered service: for instance, finger print and face recognition.

In addition, the cryptography system in IT solutions is a critical point that gains a lot of attacker attention. Focusing on the feasibility and the compatibility of the cryptography system for smart devices and implementation platforms are points that must be investigated and improved, as efficiency of the cryptography system over a specific platform does not mean compatibility for all platforms. Another direction for future research is investigating the performance of the cryptography system over a limited resource platform to ensure its efficiency.
In terms of compatibility, the developed approach must be tested over other environments, for instance, Windows and IOS. In order to enhance the performance, such testing will help to define, investigate, and improve the limitations and obstacles. In addition, developing a mechanism that aims to optimise the cloud storage selection during the distribution process is a good area for research.

Finally, the boundaries of the developed approach must be pushed; as the approach can be shifted from Storage as a service to PaaS, which means developing a development environment that allows the developers to build their own application, based on the required features and the preferred cloud storage providers.
REFERENCES


BERTRAND, N., 2017. GOP data firm that exposed millions of Americans’ personal information is facing its first class-action lawsuit, Available at: https://goo.gl/mpmCpr [Accessed January 2019].


- CHANG, L., 2016. The latest data breach involves the voting records of 93.4 million Mexican citizens, Available at: https://goo.gl/xmojTF [Accessed 31 December 2017].


- CSA, 2016. Defined Categories of Security as a Service (Preview), Available at: https://goo.gl/XQnMC9 [Accessed 31 December 2017].


- MELL, P. and GRANCE, T., 2011. The NIST definition of cloud computing.


- O’SULLIVAN, D., 2017. The RNC Files: Inside the Largest US Voter Data Leak, Available at: https://goo.gl/7ZGoQi [Accessed January 2019].


- SCIPIONI, J., 2018. FedEx data breach: 119,000 passports or photo IDs found on unsecured server, Available at: https://fxn.ws/2sypWaK [Accessed January 2019].


- WHITTAKER, Z., 2018. Unsecured server exposed thousands of FedEx customer records, Available at: [https://zd.net/2DeCslU](https://zd.net/2DeCslU) [Accessed January 2019].


APPENDICES
### APPENDIX A

<table>
<thead>
<tr>
<th>Term</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sky Computing</strong></td>
<td>A set of geographically distributed resources/capabilities that is delivered to customers dynamically (Keahey et al., 2009).</td>
</tr>
<tr>
<td><strong>Vertical Federation</strong></td>
<td>A set of federated clouds, whereby each cloud is used to deliver one layer (IaaS, PaaS, SaaS). The collaboration occurs between the adjacent levels only (IaaS to PaaS, PaaS to SaaS) (Villegas et al., 2012).</td>
</tr>
<tr>
<td><strong>Horizontal Federation</strong></td>
<td>A set of federated clouds in which the clouds collaborate to enhance the delivered services. The collaboration occurs between the match layers only (IaaS to IaaS, PaaS to PaaS, SaaS to SaaS) (Celesti et al., 2012).</td>
</tr>
<tr>
<td><strong>Hybrid Cloud</strong></td>
<td>A temporary combination between a private cloud and one or more public clouds, which is established when the private cloud status meets predefined terms and conditions. Usually, the private cloud does not have control over the remote clouds (loosely coupled) (Hogan et al., 2012; Leavitt, 2013).</td>
</tr>
<tr>
<td><strong>Inter-cloud</strong></td>
<td>A set of collaborative clouds that provides the services dynamically regardless of the mechanism used (federated or multi) (Bernstein, Vij &amp; Diamond, 2011).</td>
</tr>
<tr>
<td><strong>Multi-cloud</strong></td>
<td>A multiple-cloud delivery model that occurs between two or more independent clouds. These clouds are used to implement the requested tasks by consuming the provided resources/capabilities via cloud service providers; here, the customers have the onus of resources/capabilities managing, task scheduling and load balancing. The multi-cloud model does not need prior agreement between the service providers, and it might allow a sort of advanced control over remote resources (partially coupled) (Ferrer et al., 2012; Goiri, Guitart &amp; Torres, 2012; Grozev &amp; Buyya, 2014).</td>
</tr>
<tr>
<td><strong>Multiple-clouds</strong></td>
<td>A set of geographically distributed clouds that could be consumed in a serial or simultaneous way (Hogan et al., 2011).</td>
</tr>
<tr>
<td><strong>Peer-to-Peer Federation</strong></td>
<td>A federation model where each cloud is connected to the other directly, and all the management, load balancing, communication, charging and monitoring processes are embedded in each independent cloud. This model offers a very high availability and redundancy level, but the operational cost is very high (Grozev &amp; Buyya, 2014).</td>
</tr>
<tr>
<td><strong>Centralized Federation</strong></td>
<td>A federation model that contains a central entity, which is used as a connection node between the federated-cloud. The central entity is responsible for resource allocation, resource optimisation, load balancing, scheduling, cloud interconnecting, monitoring and management. This model offers a high level of control, which might be tight (Villari et al., 2012; Grozev &amp; Buyya, 2014).</td>
</tr>
<tr>
<td><strong>Library-based Multi-cloud</strong></td>
<td>A multi-cloud model whereby the cloud service providers have to define and agree on the rules for the execution and delivery processes and develop a service (mediator) that is used to deliver the multi-cloud services (Goiri, Guitart &amp; Torres, 2012).</td>
</tr>
<tr>
<td><strong>Service-based Multi-cloud</strong></td>
<td>A multi-cloud model where the users develop their own service broker through a unified API (Grozev &amp; Buyya, 2014).</td>
</tr>
<tr>
<td><strong>Multi-tier Cloud</strong></td>
<td>Two or more clouds, with each running its own cloud OS and usually belonging to the same corporation. The clouds are managed by a third cloud (tightly coupled), and the resources/capabilities are available as a single cloud. This architecture is beneficial for corporations with geographically distributed cloud infrastructures because it provides uniform access (Moreno-Vozmediano, Montero &amp; Llorente, 2012).</td>
</tr>
<tr>
<td>Publications</td>
<td>Aim</td>
</tr>
<tr>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----</td>
</tr>
<tr>
<td>A Comber Approach to Protect Cloud Computing against XML DDoS and HTTP DDoS attack&lt;br&gt;Tarun Karnwal; T. Sivakumar; G. Aghila</td>
<td>1</td>
</tr>
<tr>
<td>A combined approach to ensure data security in cloud computing&lt;br&gt;Sood, Sandeep K.</td>
<td>1</td>
</tr>
<tr>
<td>A Comparison of Secure Multi-tenancy Architectures for File system Storage Clouds&lt;br&gt;Anil Kurmus, Moitrayee Gupta, Roman Pletka, Christian Cachin, and Robert Haas</td>
<td>1</td>
</tr>
<tr>
<td>A comprehensive approach to privacy in the cloud-based Internet of Thing&lt;br&gt;Martin Henzea, Lars Hermerschmidt, Daniel Kerpenb, Roger Häußling B, Bernhard Rumpec, Klaus Wehrle</td>
<td>1</td>
</tr>
<tr>
<td>A Comprehensive Review On Cloud Computing Security&lt;br&gt;G.Shanmugasundaram, V.Aswni, G.Suganya</td>
<td>1</td>
</tr>
<tr>
<td>A Comprehensive Survey on Security in Cloud Computing&lt;br&gt;Gururaj Ramachandra, Mohsin Iftikhar, Farrukh Aslam Khan</td>
<td>1</td>
</tr>
<tr>
<td>A data integrity verification scheme in mobile cloud computing&lt;br&gt;Chen Lin, Zhidongshen, Qianchen, Frederic.k.Sheldon</td>
<td>1</td>
</tr>
<tr>
<td>A design for cloud-assisted Fair-Play Management System of online contests with provable security&lt;br&gt;Wang W, Xu P, Yang LT, Li H</td>
<td>1</td>
</tr>
<tr>
<td>A First Step Toward Network Security Virtualization: From Concept To Prototype Seungwon Shin, Haopei Wang, And Guofei Gu</td>
<td>1</td>
</tr>
<tr>
<td>A framework for critical security factors that influence the decision of cloud adoption by Saudi government agencies&lt;br&gt;Madini O. Alaaasfi, Abdulrahman Alharthi, Robert J. Walters, Gary B. Wills</td>
<td>1</td>
</tr>
<tr>
<td>A framework for Data Security and Storage in Cloud Computing&lt;br&gt;Akshita Bhandari, Ashutosh Gupta, DebasisDas</td>
<td>1</td>
</tr>
<tr>
<td>A framework for establishing trust in Cloud provenance&lt;br&gt;Imad M. Abbadi</td>
<td>1</td>
</tr>
<tr>
<td>A Framework to Ensure Data Storage Security in Cloud Computing&lt;br&gt;Mrinal Kanti Sarkar, Sanjay Kumar</td>
<td>1</td>
</tr>
<tr>
<td>A framework to support selection of cloud providers based on security and privacy requirements</td>
<td>1</td>
</tr>
<tr>
<td>Title</td>
<td>Authors</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>A full lifecycle privacy protection scheme for sensitive data in cloud computing</td>
<td>Haralambos Mouratidis, Shareeful Islam, Christos Kalloniatis, Stefanos Gritzalis</td>
</tr>
<tr>
<td>A General Approach for a Trusted Deployment of a Business Process in Clouds</td>
<td>Elio Goettelmann, Nicolas Mayer, Claude Godart</td>
</tr>
<tr>
<td>A method of DDoS attack detection using HTTP packet pattern and rule engine in cloud computing environment</td>
<td>Junho Choi, Chang Choi, Byeongkyu Ko, Pankoo Kim</td>
</tr>
<tr>
<td>A Multilevel Encryption Technique in Cloud Security</td>
<td>Bappaditya Jana, Jayanta Poray, Tamoghna Mandal, Malay Kule</td>
</tr>
<tr>
<td>A new form of DOS attack in a cloud and its avoidance mechanism</td>
<td>Liu, Huan</td>
</tr>
<tr>
<td>A New Framework for Cloud Storage Confidentiality to Ensure Information Security</td>
<td>Deepak Singh, Harsh K Verma</td>
</tr>
<tr>
<td>A New Steganography Protocol for Improving Security of Cloud Storage Services</td>
<td>Kazuki Murakami, Qiangfu Zhao, Ryota Hanyu</td>
</tr>
<tr>
<td>A Quantitative Analysis of Cloud Users' Satisfaction and Data Security in Cloud Models</td>
<td>Munwar Ali Zardari, Low Tang Jung, Nordin Zakaria</td>
</tr>
<tr>
<td>A quantitative analysis of current security concerns and solutions for cloud computing</td>
<td>Gonzalez, Nelson; Miers, Charles; Redigolo, Fernando; Simplicio, Marcos; Carvalho, Tereza; Näslund, Mats; Pourzandi, Makan</td>
</tr>
<tr>
<td>A reliable recommendation and privacy-preserving based cross-layer reputation mechanism for mobile cloud computing</td>
<td>Lin H, Xu L, Mu Y, Wu W.</td>
</tr>
<tr>
<td>A Review on Cloud Security</td>
<td>Duygu Sinanc, Seref Sagiorglu</td>
</tr>
<tr>
<td>A Review on Cloud Security Threats and Solutions</td>
<td>Anu Maria Sebastian, Asst.Prof Jubilent J Kizhakkhettomam</td>
</tr>
<tr>
<td>A Robust Security Framework for Cloud-based Logistics Services</td>
<td>Kathiravan Srinivasan, Takshi Gupta, Punjal Agarwal, Anant Nema</td>
</tr>
<tr>
<td>A secure and efficient Ciphertext-Policy Attribute-Based Proxy Re-Encryption for cloud data sharing</td>
<td>Liang K, Au MH, Liu JK, Susilo W, Wong DS, Yang G, Yu Y, Yang A.</td>
</tr>
<tr>
<td>A Security Model for Preserving the Privacy of Medical Big Data in a Healthcare Cloud Using a Fog Computing Facility With Pairing-Based Cryptography</td>
<td>Hadeal Abdulaziz Al Hamid, Sk Md Mizanur Rahman, M. Shamim Hossain, Ahmad Almogren, And Atif Alamri,</td>
</tr>
<tr>
<td>A Study on Data Storage Security Issues in Cloud Computing</td>
<td>Naresh Vurukonda, B.Thirumala Rao</td>
</tr>
<tr>
<td>A survey of cloud computing data integrity schemes: Design challenges, taxonomy and future trends</td>
<td></td>
</tr>
<tr>
<td>Title</td>
<td>Authors</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>A Survey of Risks, Threats and Vulnerabilities in Cloud Computing</td>
<td>Kamal Dahbur, Bassil Mohammad, Ahmad Bisher Tarakji</td>
</tr>
<tr>
<td>A survey of security issues for cloud computing</td>
<td>Minhaj Ahmad Khan</td>
</tr>
<tr>
<td>A Survey on Automated Dynamic Malware-Analysis Techniques and Tools</td>
<td>MANUEL EGELE; THEODOOR SCHOLTE; ENGIN KIRDA; CHRISTOPHER KRUEGEL</td>
</tr>
<tr>
<td>A survey on cloud computing security: Issues, threats, and solutions</td>
<td>Saurabhsingh, Young-Sikjeong, Jonghyukpark</td>
</tr>
<tr>
<td>A survey on gaps, threat remediation challenges and some thoughts for proactive attack detection in cloud computing</td>
<td>Md. Tanim Khosheed, A.B.M. Shawkat Ali, Saleh A. Wasimi</td>
</tr>
<tr>
<td>A survey on securing the virtual cloud</td>
<td>Robert Denz and Stephen Taylor</td>
</tr>
<tr>
<td>A Survey on Security in Network Functions Virtualization</td>
<td>Wei Yang And Carol Fung</td>
</tr>
<tr>
<td>A survey on security issues and solutions at different layers of Cloud computing</td>
<td>Chirag Modi, Dhiren Patel, Bhavesh Borisaniya, Avi Patel, Muttukrishnan Rajarajan</td>
</tr>
<tr>
<td>A survey on security issues in service delivery models of cloud computing</td>
<td>Subashini, S.; Kavitha, V.</td>
</tr>
<tr>
<td>A systematic classification of security regression testing approaches</td>
<td>Michael Felderer · Elizabeta Fournier</td>
</tr>
<tr>
<td>A systematic review on cloud computing</td>
<td>Frederico Durao, Jose Fernando S, Carvalho, Anderson Fonseka, Vinicius Cardoso Garcia</td>
</tr>
<tr>
<td>A test-based security certification scheme for web services</td>
<td>MARCO ANISETTI, CLAUDIO A. ARDAGNA, ERNESTO DAMIANI, FRANCESCO SAONARA</td>
</tr>
<tr>
<td>A vision for cloud security</td>
<td>Hongwen Zhang</td>
</tr>
<tr>
<td>Title</td>
<td>Authors</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Achieving an effective, scalable and privacy-preserving data sharing service in cloud computing</td>
<td>Xin Dong a, Jiadi Yu , Yuan Luo , Yingying Chen , Guangtao Xue , Minglu Li</td>
</tr>
<tr>
<td>Addressing cloud computing security issues</td>
<td>Dimitrios Zissis , Dimitrios Lekkas</td>
</tr>
<tr>
<td>An analysis of security issues for cloud computing</td>
<td>Keiko Hashizume, David G Rosado, Eduardo Fernández-Medina and Eduardo B Fernandez</td>
</tr>
<tr>
<td>An Efficient and Secure Data Sharing Framework using Homomorphic Encryption in the Cloud</td>
<td>Bharath K. Samantha, Gerry Howser, Yousef Elmehdwi, and Sanjay Madria</td>
</tr>
<tr>
<td>An efficient and secure privacy-preserving approach for outsourced data of resource constrained mobile devices in cloud computing</td>
<td>Pasupuleti SK, Ramalingam S, Buyya R.</td>
</tr>
<tr>
<td>An enhanced remote authentication scheme to mitigate man-in-the-browser attacks</td>
<td>Bin Mat Nor, F; Jailil, Kamarularifin Abd; Manan, J-LA</td>
</tr>
<tr>
<td>An Experimental Evaluation of Data Confidentiality Measures on the Cloud</td>
<td>Hussain Alijatef, Zaki Malik, Mohammed Alodib, Abdelmonaam Rezgui</td>
</tr>
<tr>
<td>An Investigation into Cloud Configuration and Security</td>
<td>K. Wood, E. Pereira</td>
</tr>
<tr>
<td>An Overview of Issues and Recent Developments in Cloud Computing and Storage Security</td>
<td>Everaldo Aguiar, Yihua Zhang, Marina Blanton</td>
</tr>
<tr>
<td>An overview of the commercial cloud monitoring tools: research dimensions, design issues, and state-of-the-art</td>
<td>Khalid Alhamazani , Rajiv Ranjan , Karan Mitra , Fethi Rabhi , Prem Prakash Jayaraman , Samee Ullah Khan , Adene Guabtni , Vasudha Bhatnagar</td>
</tr>
<tr>
<td>Analysis of Security and Privacy in Public Cloud Environment</td>
<td>Abdul Sattar Raja, Dr. Shukor Abd Razak</td>
</tr>
<tr>
<td>Analysis of VM Communication for VM-based Cloud Security Systems</td>
<td>Siyakha N Mthunzi, Elhadj Benkhelifa, Mohammad A. Alsmirat, and Yaser Jararweh</td>
</tr>
<tr>
<td>Authentication at scale</td>
<td>Grosse, Eric; Upadhyay, Mayank</td>
</tr>
<tr>
<td>Automatic detection and correction of web application vulnerabilities using data mining to predict false positives</td>
<td>Ibéria Medeiros; Nuno F. Neves; Miguel Correia</td>
</tr>
<tr>
<td>Beyond lightning: A survey on security challenges in cloud computing</td>
<td>Chunming Rong , Son T. Nguyen , Martin Gilje Jaatun</td>
</tr>
<tr>
<td>Building safe PaaS clouds: A survey on security in multitenant software platforms</td>
<td>Rodero-Merino,Luis; Vaquero,Luis M.; Caron,Eddy; Muresan,Adrian; Desprez,Frédéric</td>
</tr>
<tr>
<td>Title</td>
<td>Authors</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Characterizing hypervisor vulnerabilities in cloud computing servers</td>
<td>Perez-Botero, Diego; Szefer, Jakub; Lee, Ruby B</td>
</tr>
<tr>
<td>City on the Sky: Extending XACML for Flexible, Secure Data Sharing on</td>
<td>Tien Tuan Anh Dinh, Wang Wenqiang, Anwitaman Datta</td>
</tr>
<tr>
<td>the Cloud</td>
<td></td>
</tr>
<tr>
<td>Classifying malware attacks in IaaS cloud environments</td>
<td>Noëlle Rakotondravony, Benjamin Taumann, Waseem Mandarawi, Eva Weishäupl,</td>
</tr>
<tr>
<td></td>
<td>Peng Xu, Bojan Kolosnjaji, Mykolai Protsemba, Hermann De Meer And Hans</td>
</tr>
<tr>
<td></td>
<td>P. Reiser</td>
</tr>
<tr>
<td>Cloud Audit: A Data Integrity Verification Approach for Cloud</td>
<td>Rajat Saxena And Somnath Dey</td>
</tr>
<tr>
<td>Computing</td>
<td></td>
</tr>
<tr>
<td>Cloud based Secure and Privacy Enhanced Authentication &amp; Authorization</td>
<td>Umer Khalida, Abdul Ghafoor, Misbah Irum, Muhammad Awais Shibli</td>
</tr>
<tr>
<td>Protocol</td>
<td></td>
</tr>
<tr>
<td>Cloud computing — The business perspective</td>
<td>Sean Marston, Zhi Li, Subhajoti Bandypadhyay, Juheng Zhang, Anand</td>
</tr>
<tr>
<td></td>
<td>Ghalsasi</td>
</tr>
<tr>
<td>Cloud computing adoption framework: A security framework for</td>
<td>Victor Changa, Yen-Hung Kuob, Muthu Ramachandran</td>
</tr>
<tr>
<td>business clouds</td>
<td></td>
</tr>
<tr>
<td>Cloud computing and security challenges</td>
<td>Yu, Huiming, Powell, Nakia, Stembridge, Dexter, Yuan, Xiaohong</td>
</tr>
<tr>
<td>Cloud Computing Features, Issues and Challenges: A Big Picture</td>
<td>Deepak Puthal, B. P. S. Sahoo, Sambit Mishra, And Satyabrata Swain</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>cloud computing security by integrating classical encryption</td>
<td>Koorosh Goodarzi, abbas karimi</td>
</tr>
<tr>
<td></td>
<td>Manjima Bose, Kaushik Basu, Saunak Chaudhury, Prithika Sarkar</td>
</tr>
<tr>
<td>Cloud Computing: benefits, risks and recommendations for information</td>
<td></td>
</tr>
<tr>
<td>security</td>
<td>Catteddu, Daniele</td>
</tr>
<tr>
<td>Cloud concerns: the safe route to adoption</td>
<td>Julian Dyer</td>
</tr>
<tr>
<td>Cloud identity management security issues &amp; solutions: a taxonomy</td>
<td>Umme Habiba, Rahat Masood, Muhammad Awais Shibli and Muaz A Niazi</td>
</tr>
<tr>
<td>Cloud Security Challenges</td>
<td>Gurudatt Kulikarni, Nikita Chavan, Ruchira Chandorkar, Rani Waghmare,</td>
</tr>
<tr>
<td></td>
<td>Rajnikant Palwe</td>
</tr>
<tr>
<td>Cloud security defence to protect cloud computing against HTTP-DoS</td>
<td>Ashley Chonka, Yang Xiang, Wanlei Zhou, Alessio Bonti</td>
</tr>
<tr>
<td>and XML-DoS attacks</td>
<td></td>
</tr>
<tr>
<td>Title</td>
<td>Authors</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Cloud Security Engineering: Theory, Practice and Future Research</td>
<td>Kim-Kwang Raymond Choo, Omer F. Rana, And Muttukrishnan Rajarajan</td>
</tr>
<tr>
<td>Cloud security issues and challenges: A survey</td>
<td>Ashish Singh, Kakali Chatterjee</td>
</tr>
<tr>
<td>Cloud security monitoring and vulnerability management</td>
<td>M. Kozlovszky, L.Kovác, M. Tőrőcsik, G.Windisch, S.Ács, D.Prém, Gy. Eigner, P.I. Sas, T.Schubert, V. Pőserné</td>
</tr>
<tr>
<td>Cloud security: Emerging threats and current solutions</td>
<td>Luigi Coppolino, Salvatore D’antonio, Giovanni Mazzeo, Luigi Romano</td>
</tr>
<tr>
<td>Cloud Storage Encryption Security Analysis</td>
<td>Zhang Jing, Wang Jinsu, Zheng Zhuangfeng, Zhao Chongan</td>
</tr>
<tr>
<td>Cloud storage reliability for Big Data applications: A state of the art survey</td>
<td>Rekha Nachiappan, Bahman Javadi, Rodrigo N. Calheiros, Kenan M. Matawie</td>
</tr>
<tr>
<td>Cloud-Centric Multi-Level Authentication as a Service for Secure Public Safety Device Networks</td>
<td>Ismail Butun, Melike Erol-Kantarci, Burak Kantarci, And Houbing Song</td>
</tr>
<tr>
<td>Cloud Trust—a Security Assessment Model for Infrastructure as a Service (IaaS) Clouds</td>
<td>Dan Gonzales, Jeremy M. Kaplan, Evan Saltzman, Zev Winkelman, And Dulani Woods</td>
</tr>
<tr>
<td>Cloudvisor: Retrofitting Protection of Virtual Machines in Multi-tenant Cloud with Nested Virtualization</td>
<td>Fengzhe Zhang, Jin Chen, Haibo Chen and Binyu Zang</td>
</tr>
<tr>
<td>Commentary: Cloud computing e A security problem or solution?</td>
<td>P.G. Dorey, A. Leite</td>
</tr>
<tr>
<td>Comparative analysis of cloud data integrity auditing protocols</td>
<td>Neenugarg N, Seemabawa</td>
</tr>
<tr>
<td>Cross-Tenant Side-Channel Attacks in PaaS Clouds</td>
<td>Yingqian Zhang, Ari Juels, Michael K. Reiter, Thomas Ristenpart</td>
</tr>
<tr>
<td>Cross-VM side channels and their use to extract private keys</td>
<td>Zhang, Yinqian; Juels, Ari; Reiter, Michael K; Ristenpart, Thomas</td>
</tr>
<tr>
<td>CyberGuarder: A virtualization security assurance architecture for green cloud computing</td>
<td>Jianxin Li, Bo Li, Tianyu Woa, Chunming Hua, Jinpeng Huai, Lu Liu, K.P. Lam</td>
</tr>
<tr>
<td>Data and infrastructure security auditing in cloud computing environments</td>
<td>Hassan Rasheed</td>
</tr>
<tr>
<td>Data On-boarding in Federated Storage Clouds</td>
<td>Gil Vernik, Alexandra Shulman-Peleg, Sebastian Dippl, Ciro Formisano, Michael C. Jaeger, Elliot K. Kolodner, Massimo Villari</td>
</tr>
<tr>
<td>Data Security and Confidentiality in Public Cloud Storage by Extended QP Protocol</td>
<td>K.Priya, J.ArokiaRenjit</td>
</tr>
<tr>
<td>Data Security and Integrity in Cloud Computing Based on RSA Partial Homomorphic and MD5 Cryptography</td>
<td>Priyanka Ora, Dr.P.R.Pal</td>
</tr>
<tr>
<td>Title</td>
<td>Authors</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Data security and privacy preservation in cloud storage environments</td>
<td>Nesrine Kaaniche, Maryline Laurent</td>
</tr>
<tr>
<td>based on cryptographic mechanisms</td>
<td></td>
</tr>
<tr>
<td>Data Security and Privacy Protection Issues in Cloud Computing</td>
<td>Deyan Chen, Hong Zhao</td>
</tr>
<tr>
<td>Data Security Challenges and Its Solutions in Cloud Computing</td>
<td>R. Velumadhava Raoa, K. Selvamanib</td>
</tr>
<tr>
<td>Data Security Enhancement in Public Cloud Storage using Data Obfuscation</td>
<td>Dr. D.I. George Amalarethinam, B. Fathima Mary</td>
</tr>
<tr>
<td>and Steganography</td>
<td></td>
</tr>
<tr>
<td>Data Security in Cloud</td>
<td>Lalitha V.P, Sagar M.Y, Sharanappa S, Shredar Hanji, Swarup R</td>
</tr>
<tr>
<td>Data Security in Cloud Computing Using AES Under HEROKU Cloud</td>
<td>Bih-Hwang Lee, Ervin Kusuma Dewi, Muhammad Farid Wajdi</td>
</tr>
<tr>
<td>Data storage auditing service in cloud computing: challenges, methods</td>
<td>Kan Yang, Xiaohua Jia</td>
</tr>
<tr>
<td>and opportunities</td>
<td></td>
</tr>
<tr>
<td>DDoS attacks in cloud computing: Issues, taxonomy, and future</td>
<td>Gaurav Somani, Manoj Singh Gaur, Dheeraj Sanghi, Mauro Conti, Rajkumar</td>
</tr>
<tr>
<td>directions</td>
<td>Buyya</td>
</tr>
<tr>
<td>Decision-making in cloud computing environments: A cost and risk based</td>
<td>Benedikt Martens &amp; Frank Teuteberg</td>
</tr>
<tr>
<td>approach</td>
<td></td>
</tr>
<tr>
<td>Defending Web Services against Denial of Service Attacks Using Client</td>
<td>Suriadi, S.; Inf. Security Inst., Queensland Univ. of Technol. Brisbane,</td>
</tr>
<tr>
<td>Puzzles</td>
<td>Brisbane, QLD, Australia; Stebila, D.; Clark, A.; Hua Liu</td>
</tr>
<tr>
<td>DEPSKY: Dependable and Secure Storage in a Cloud-of-Clouds</td>
<td>ALYSSON BESSANI, MIGUEL CORREIA, BRUNO QUARESMA, FERNANDO ANDRE’, and</td>
</tr>
<tr>
<td></td>
<td>PAULO SOUSA,</td>
</tr>
<tr>
<td>Design and Implementation of Security System for Cloud Storage</td>
<td>Ju-Shu Chueh and Min-Te Sun</td>
</tr>
<tr>
<td>Detecting Flooding based DoS Attack in Cloud Computing Environment</td>
<td>Mohd Nazri Ismail, Abdulaziz Aborujilah, Shahruzniza Musa, AAmir Shahzad</td>
</tr>
<tr>
<td>Using Covariance Matrix Approach</td>
<td></td>
</tr>
<tr>
<td>Different facets of security in the cloud</td>
<td>Boampong, Philogene A; Wahsheh, Luay A</td>
</tr>
<tr>
<td>Distributed denial of service (DDoS) resilience in cloud: Review</td>
<td>Osanaiye O, Choo KK, Dlodlo M.</td>
</tr>
<tr>
<td>and conceptual cloud DDoS mitigation framework</td>
<td></td>
</tr>
<tr>
<td>Distributed Denial of Service Attacks in Software-Defined Networking</td>
<td>Qiao Yan And F. Richard Yu</td>
</tr>
<tr>
<td>with Cloud Computing</td>
<td></td>
</tr>
<tr>
<td>Effects of virtualization on information security</td>
<td>Shing-Han Li, David C. Yen, Shih-Chih Chen, Patrick S. Chen, Wen-Hui Lu,</td>
</tr>
<tr>
<td></td>
<td>Chien-Chuan Cho</td>
</tr>
</tbody>
</table>

227
<table>
<thead>
<tr>
<th>Title</th>
<th>Authors</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Efficient integrity verification of replicated data in cloud computing system</td>
<td>Mingxu Yi, Jinxia Wei, Lingwei Song</td>
<td>1111116</td>
</tr>
<tr>
<td>Efficient Privacy Preserving Integrity Checking Model for Cloud Data Storage Security</td>
<td>T.Subha and Dr.S.Jayashri</td>
<td>111116</td>
</tr>
<tr>
<td>Efficient Replica Migration Scheme for Distributed Cloud Storage Systems</td>
<td>Amina Mseddi, Mohammad A. Salahuddin, Mohamed Faten Zhani, Halima Elbiaze, and Roch H. Giltho,</td>
<td>111117</td>
</tr>
<tr>
<td>Enabling efficient access control with dynamic policy updating for big data in the cloud</td>
<td>Kan Yang; Xiaohua Jia; Kui Ren; Ruitao Xie</td>
<td>11115</td>
</tr>
<tr>
<td>Encryption as a Service for Data Healthcare Cloud Security</td>
<td>Abdelali El Bouchti, Samir Bahsani, Tarik Nahhal</td>
<td>1115</td>
</tr>
<tr>
<td>Enforcing Trust in Home Automation Platforms</td>
<td>Sarah Majoul, M. Richard-Foy, Aitor Agirre, Antonio Pérez, Antonio Kung</td>
<td>111017</td>
</tr>
<tr>
<td>Enhanced Cloud Data Security Using AES Algorithm</td>
<td>Akhil K.M, Praveen Kumar M, Pushpa B.R</td>
<td>111105</td>
</tr>
<tr>
<td>Enhanced Cloud Security by Combining Virtualization and Policy Monitoring Techniques</td>
<td>Loganayagi.Bb, S.Sujatha,</td>
<td>111118</td>
</tr>
<tr>
<td>Enhancing Cloud Security and Privacy: Time for a New Approach?</td>
<td>Bob Duncan, Alfred Bratterud, Andreas Happe</td>
<td>11116</td>
</tr>
<tr>
<td>Ensuring Cloud Security: Current Concerns and Research Challenges</td>
<td>Vesyropoulos, N., Georgiadis, C. &amp; Pimenidis, E.</td>
<td>11116</td>
</tr>
<tr>
<td>Ensuring Security and Privacy Preservation for Cloud Data Services</td>
<td>Jun Tang, Yong Cui And Qi Li, Kui Ren, Jiangchuan Liu, Rajkumar Buyya,</td>
<td>11115</td>
</tr>
<tr>
<td>Experimental analysis of attacks against Web services and countermeasures</td>
<td>Abdallah Ghourabi; Tarek Abbes; Adel Bouhoula</td>
<td>111105</td>
</tr>
<tr>
<td>Exploring Data Security Issues and Solutions in Cloud Computing</td>
<td>P. Ravi Kumar, P. Herbert Rajb, P. Jelcianac</td>
<td>11115</td>
</tr>
<tr>
<td>Expressiveness Considerations of XML Signatures</td>
<td>Meiko Jensen, Christopher Meyer</td>
<td>11115</td>
</tr>
<tr>
<td>From Security to Assurance in the Cloud: A Survey</td>
<td>Claudio A. Ardagna, Rasool Asal, Ernesto Damiani, Quang Hieu Vu,</td>
<td>111107</td>
</tr>
<tr>
<td>Full integrity and freshness for cloud data</td>
<td>Hao Jin, Ke Zhoua, Hong Jiang, Dongliang Lei, Ronglei Weia, Chunhua Li</td>
<td>111105</td>
</tr>
<tr>
<td>HASG: security and efficient frame for accessing cloud storage</td>
<td>Shenling Liu, Chunyuan Zhang1, Yujiao Chen</td>
<td>111117</td>
</tr>
<tr>
<td>Home is Safer than the Cloud: Privacy Concerns for Consumer Cloud Storage</td>
<td>Iulia Ion, Niharika Sachdeva, Ponnurangam Kumaraguru, Srdjan apkun</td>
<td>111116</td>
</tr>
<tr>
<td>Hourglass Schemes: How to Prove that Cloud Files Are Encrypted</td>
<td>Marten van Dijk, Ari Juels, Alina Oprea, Ronald L. Rivest, Emil Stefanov, Nikos Triandopoulos</td>
<td>111117</td>
</tr>
<tr>
<td>HyperSafe: A Lightweight Approach to Provide Lifetime Hypervisor Control-Flow Integrity</td>
<td></td>
<td>111105</td>
</tr>
<tr>
<td>Title</td>
<td>Authors</td>
<td>Citation</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>IaaS Cloud Security</td>
<td>Pragati Chavan, Pradeep Patil, Gurudatt Kulkarni, Ramesh Sutar, Shrikant Belsare</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Identifying Cloud Security Threats to Strengthen Cloud Computing Adoption Framework</td>
<td>Nabeel Khana, Adil Al-Yasirib</td>
<td>1 1 0 0 0 1 1 1 0 5</td>
</tr>
<tr>
<td>Implementation of Re-encryption Based Security Mechanism to Authenticate Shared Access in Cloud Computing</td>
<td>Ms. Neha Mahakalkar, Mrs. Vaishali Sahare</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Increasing virtual machine security in cloud environments</td>
<td>Roland Schwarzkopf, Matthias Schmidt, Christian Strack, Simon Martin and Bernd Freisleben</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Infrastructure as a Service Security: Challenges and Solutions</td>
<td>Wesam Dawoud, Ibrahim Takouna, Christoph Meinel</td>
<td>1 1 1 0 0 1 1 1 1 7</td>
</tr>
<tr>
<td>Insider Attacks in Cloud Computing</td>
<td>Duncan, A.J.; Creese, S.; Goldsmith, M.</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Integrating Trust with Cryptographic Role-Based Access Control for Secure Cloud Data Storage</td>
<td>Lan Zhou, Vijay Varadharajan, Michael Hitchens</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Lightweight secure storage model with fault-tolerance in cloud environment</td>
<td>Muhsra Ahmed, Quang Hieu Vu, Rasool Asal, Hassan Al Muhairi, Chan Yeob Yeun</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Load-based covert channels between Xen virtual machines</td>
<td>Okamura, Keisuke; Oyama, Yoshiiro</td>
<td>1 1 1 0 0 1 1 0 0 5</td>
</tr>
<tr>
<td>Locking the sky: a survey on IaaS cloud security</td>
<td>Vaquero, Luis M; Rodero-Merino, Luis; Morán, Daniel</td>
<td>1 1 1 0 0 0 0 1 1 5</td>
</tr>
<tr>
<td>Mechanisms of Polymorphic and Metamorphic Viruses</td>
<td>Xufang Li; Loh, P.K.K.; Tan, F.</td>
<td>1 1 1 0 1 1 0 0 5</td>
</tr>
<tr>
<td>Meta Cloud Data Storage Architecture for Big Data Security in Cloud Computing</td>
<td>Gunasekaran Manogarana, Chandu Thotab, M. Vijay Kumarc</td>
<td>1 1 0 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Mobile Cloud Computing Standard approach to protecting and securing of mobile cloud ecosystems</td>
<td>Ranbijay Kumar, Dr. S. Rajalakshmi</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Mobile security: a look ahead</td>
<td>Li, Qing; Clark, Greg</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Monitoring Intrusions and Security Breaches in Highly Distributed Cloud Environments</td>
<td>Montafred, Aryan Taheri; Jaatun, Martin Gilje</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Multi-cloud Performance and Security Driven Federated Workflow Management</td>
<td>Matthew Dickinson, Saptarshi Debroy, Prasad Calyam, Samaikya Vallripally, Yuanxun Zhang, Ronny Bazan Antequera, Trupti Joshi, Tommi White, Dong Xu</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Multi-level authentication technique for accessing cloud services</td>
<td>Dinesha, HA; Agrawal, VK</td>
<td>1 1 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Multilevel classification of security concerns in cloud computing</td>
<td></td>
<td>1 1 1 0 1 1 1 1 1 8</td>
</tr>
<tr>
<td>Title</td>
<td>Authors</td>
<td>CiteScore</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
<td>-----------</td>
</tr>
<tr>
<td>Multilevel Security framework for Cloud Data</td>
<td>Syed Asad Hussain A, Mehwish Fatima A, Atif Saeed B, Imran Raza A, Raja Khurram Shahzad</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Multitasking without compromise: a virtual machine evolution</td>
<td>Akshat Kumar Dixit, Dr. Charu</td>
<td>1 1 1 1 1 1 1 0 0 7</td>
</tr>
<tr>
<td>Next generation cloud computing: New trends and research directions</td>
<td>Blesson Varghese, Rajkumar Buyya</td>
<td>1 1 1 0 0 1 1 1 1 7</td>
</tr>
<tr>
<td>On cloud security attacks: A taxonomy and intrusion detection and prevention as a service</td>
<td>Salman Iqbal, laihamatkiah, Babakhdaghighe, Muzamil hussain, Suleman Khan, Muhammad khurramkhan, Kim-Kwangraymondchoo</td>
<td>1 1 1 0 1 1 1 1 1 8</td>
</tr>
<tr>
<td>On the effectiveness of XML Schema validation for counterering XML Signature Wrapping attacks</td>
<td>Jensen, M. ; Horst Gortz Inst. for IT-Security, Ruhr-Univ. Bochum, Bochum, Germany ; Meyer, C. ; Somorovsky, J. ; Schwenk, J.</td>
<td>1 1 1 0 0 1 1 0 0 5</td>
</tr>
<tr>
<td>Ontological Approach toward Cybersecurity in Cloud Computing</td>
<td>Takeshi Takahashi, Youki Kadobayashi, Hiroyuki Fujiwara</td>
<td>1 1 1 0 0 0 0 1 1 5</td>
</tr>
<tr>
<td>Privacy and Security for Cloud Computing</td>
<td>Pearson, Siani; Yee, George</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Privacy Risk, Security, Accountability in the Cloud</td>
<td>Marianithi Theoharidou, Nick Papanikolau, Siani Pearson and Dimitris Gritzalis</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Protecting the privacy and security of sensitive customer data in the cloud</td>
<td>Nancy J. King, V.T. Raja</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Proviable data transfer from provable data possession and deletion in cloud storage</td>
<td>Liang Xue, Jianbingni, Yannanli, Jianshen</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Providing Security, Integrity and Authentication Using ECC Algorithm in cloud storage</td>
<td>Akanksha Bansal, Arun Agrawal</td>
<td>1 1 1 0 0 0 1 1 1 0 5</td>
</tr>
<tr>
<td>Public integrity auditing for shared dynamic cloud data</td>
<td>Shubham Singha, Sumila Thokchomb</td>
<td>1 0 1 0 0 1 1 1 0 5</td>
</tr>
<tr>
<td>Remote data possession checking with enhanced security for cloud storage</td>
<td>Yu Y, Zhang Y, Ni J, Au MH, Chen L, Liu H.</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Research on Multi-tenant PaaS Cloud Security on Java Platform</td>
<td>Changyun Jia; Min Zhu; Xiaoming Liu</td>
<td>1 1 0 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>SDM: Smart deduplication for mobile cloud storage</td>
<td>Ryan N.S. Widodoa, Hyotaek Limb, Mohammed Atiquzzamanc</td>
<td>1 1 1 0 0 0 1 1 1 0 5</td>
</tr>
<tr>
<td>SDSecurity: A Software Defined Security Experimental Framework</td>
<td>Ala’ Darabseh, Mahmoud Al-Ayyoub, Yaser Jararwheh, Elhadj Benkhelifa, Mladen Vouk, Andy Rindos4,</td>
<td>1 1 1 0 0 1 1 1 0 0 5</td>
</tr>
<tr>
<td>Secure authentication scheme for IoT and cloud servers</td>
<td>Sheetal Kalra A, Sandeep K. Soodb</td>
<td>1 0 1 0 0 1 1 1 1 6</td>
</tr>
<tr>
<td>Secure cloud storage service for detection of security violations</td>
<td>Carlos Andr é e Batista de Carvalho, Miguel Franklin de Castro and Rossana Maria de Castro Andrade</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
</tbody>
</table>
Secure integration of IoT and Cloud Computing
Christos Stergiou, Kostas E. Psannis, Byung-Gyu Kimb, Brij Guptac

Secure Logical Isolation for Multi-tenancy in Cloud Storage
Michael Factor, David Hadas, Aner Hamama, Nadav Har’el, Elliot K. Kolodner, Anil Kurmusy, Alexandra Shulman-Peleg, Alessandro Sorniotti

Secure virtualization for cloud computing
Flavio Lombardi, Roberto Di Pietro

Secure VM Backup and Vulnerability Removal in Infrastructure Clouds
Prabhjeet Kaur, Gaurav Somani

Secured Architecture for Inter-VM traffic in a Cloud environment
Karim Benzidane, Sad Khoudali, and Abderrahim Sekkaki

SecureSense: End-to-end secure communication architecture for the cloud-connected Internet of Things
Shahid Raza, Tómas Helgason, Panos Papadimitratos, Thiemo Voigt

Securing Authentication and Trusted Migration of Weblets in the Cloud with Reduced Traffic
John Panneerselvam, Stelios Sotiriadis, Nik Bessis, Nick Antonopoulos

Securing the virtual machine images in Cloud computing
Muhammad Kazim, Rahat Masood, Muhammad Awais Shibli

Securities and threats of Cloud Computing and Solutions
P.Gayatri, M. Venunath, V. Subhashini, Syed Umar

Security Algorithms for Cloud Computing
Akashdeep Bhardwaja, Gvb Subrahmanyamb, Vinay Avasthia, Hanumat Sastryd

Security and efficiency data sharing scheme for cloud storage
Ke Han, Qingbo Li, Zhongliang Deng

Security and Privacy Challenges in Cloud Computing Environments
Takabi, H.; Joshi, J.B.D.; Gail-Joon Ahn

Security and Privacy in Cloud Computing
Zhifeng Xiao, Yang Xiao

Security and privacy in cloud computing: A survey
Zhou, Minqi; Zhang, Rong; Xie, Wei; Qian, Weining; Zhou, Aoying

Zahir Tari, Kun Yi, Uthpala S. Premaratne, Peter Bertok, And Ibrahim Khalil, RMIT University

Security and Privacy of Big Data for Social Networking Services in Cloud
Christos Stergiou, Kostas E. Psannis, Theofanis Xifilidis, Andreas P. Plageras, Brij B. Gupta

Security and Trust in Cloud Scenarios
Mariagrazia Fugini, George Hadjichristofi

Security Challenges and Solutions in Cloud Computing
Eystein Mathisen

Security enhancement of cloud servers with a redundancy-based fault-tolerant cache structure
Hongjundai Shulinzha, Jiutianzhang Meikangqiu Lixintao

Security in Cloud Computing

<table>
<thead>
<tr>
<th>Title</th>
<th>Authors</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secure integration of IoT and Cloud Computing</td>
<td>Christos Stergiou, Kostas E. Psannis, Byung-Gyu Kimb, Brij Guptac</td>
<td>1 1 1 0 0 0 1 1 0 5</td>
</tr>
<tr>
<td>Secure Logical Isolation for Multi-tenancy in Cloud Storage</td>
<td>Michael Factor, David Hadas, Aner Hamama, Nadav Har’el, Elliot K. Kolodner, Anil Kurmusy, Alexandra Shulman-Peleg, Alessandro Sorniotti</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Secure virtualization for cloud computing</td>
<td>Flavio Lombardi, Roberto Di Pietro</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Secure VM Backup and Vulnerability Removal in Infrastructure Clouds</td>
<td>Prabhjeet Kaur, Gaurav Somani</td>
<td>1 1 1 0 0 0 0 1 1 5</td>
</tr>
<tr>
<td>Secured Architecture for Inter-VM traffic in a Cloud environment</td>
<td>Karim Benzidane, Sad Khoudali, and Abderrahim Sekkaki</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>SecureSense: End-to-end secure communication architecture for the cloud-connected Internet of Things</td>
<td>Shahid Raza, Tómas Helgason, Panos Papadimitratos, Thiemo Voigt</td>
<td>1 1 1 0 0 1 1 1 7</td>
</tr>
<tr>
<td>Securing Authentication and Trusted Migration of Weblets in the Cloud with Reduced Traffic</td>
<td>John Panneerselvam, Stelios Sotiriadis, Nik Bessis, Nick Antonopoulos</td>
<td>1 1 1 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Securing the virtual machine images in Cloud computing</td>
<td>Muhammad Kazim, Rahat Masood, Muhammad Awais Shibli</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Securities and threats of Cloud Computing and Solutions</td>
<td>P.Gayatri, M. Venunath, V.Subhashini, Syed Umar</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Security Algorithms for Cloud Computing</td>
<td>Akashdeep Bhardwaja, Gvb Subrahmanyamb, Vinay Avasthia, Hanumat Sastryd</td>
<td>1 1 1 0 0 1 1 1 0 5</td>
</tr>
<tr>
<td>Security and efficiency data sharing scheme for cloud storage</td>
<td>Ke Han, Qingbo Li, Zhongliang Deng</td>
<td>1 1 1 0 0 0 1 1 0 5</td>
</tr>
<tr>
<td>Security and Privacy Challenges in Cloud Computing Environments</td>
<td>Takabi, H.; Joshi, J.B.D.; Gail-Joon Ahn</td>
<td>1 0 1 0 0 1 1 1 1 6</td>
</tr>
<tr>
<td>Security and Privacy in Cloud Computing</td>
<td>Zhifeng Xiao, Yang Xiao</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Security and privacy in cloud computing: A survey</td>
<td>Zhou, Minqi; Zhang, Rong; Xie, Wei; Qian, Weining; Zhou, Aoying</td>
<td>1 0 1 1 0 1 1 1 7</td>
</tr>
<tr>
<td>Security and Privacy in Cloud Computing: Vision, Trends, and Challenges</td>
<td>Zahir Tari, Kun Yi, Uthpala S. Premaratne, Peter Bertok, And Ibrahim Khalil, RMIT University</td>
<td>1 1 1 0 0 1 1 1 1 7</td>
</tr>
<tr>
<td>Security and Privacy of Big Data for Social Networking Services in Cloud</td>
<td>Christos Stergiou, Kostas E. Psannis, Theofanis Xifilidis, Andreas P. Plageras, Brij B. Gupta</td>
<td>1 1 1 0 0 1 1 1 0 6</td>
</tr>
<tr>
<td>Security and Trust in Cloud Scenarios</td>
<td>Mariagrazia Fugini, George Hadjichristofi</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Security Challenges and Solutions in Cloud Computing</td>
<td>Eystein Mathisen</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Security enhancement of cloud servers with a redundancy-based fault-tolerant cache structure</td>
<td>Hongjundai Shulinzha, Jiutianzhang Meikangqiu Lixintao</td>
<td>1 1 0 0 0 1 1 1 0 5</td>
</tr>
<tr>
<td>Security in Cloud Computing</td>
<td></td>
<td>1 1 0 0 0 1 1 1 1 6</td>
</tr>
</tbody>
</table>

231
<table>
<thead>
<tr>
<th>Title</th>
<th>Authors</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Security in cloud computing: Opportunities and challenges</td>
<td>Mazhar Ali, Samee U. Khan, Athanasios V. Vasilakos</td>
<td>1 0 1 0 1 1 1 1 1 7</td>
</tr>
<tr>
<td>Security in Cloud- Computing-Based Mobile Health</td>
<td>Silas L. Albuquerque And Paulo R.L. Gondim,</td>
<td>1 0 1 0 0 1 1 1 0 5</td>
</tr>
<tr>
<td>Security in Multi-Tenancy Cloud</td>
<td>Amarnath Jasti, Payal Shah, Rajeev Nagaraj and Ravi Pendse</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Security issues in cloud computing</td>
<td>Tianfield, H.</td>
<td>1 1 1 1 0 1 1 1 1 8</td>
</tr>
<tr>
<td>Security Issues Over Some Cloud Models</td>
<td>Passent M. El-Kafrawya, Azza A. Abdoa, Amr. F. Shawish</td>
<td>1 1 1 0 0 1 1 1 1 7</td>
</tr>
<tr>
<td>Security of Cloud Computing, Storage, and Networking</td>
<td>Mohamed Hamdi</td>
<td>1 1 1 1 0 1 1 1 1 8</td>
</tr>
<tr>
<td>Semantic-aware multi-tenancy authorization system for cloud architectures</td>
<td>Jorge Bernal Bernabe, Juan M. Marín Perez, Jose M. Alcaraz Calero b, Felix J. García Clementec, Gregorio Martinez Perez, Antonio F. Gomez Skarmeta</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Server-aided anonymous attribute-based authentication in cloud computing</td>
<td>Liu Z, Yan H, Li Z.</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Server-Side Streaming Processing of WS-Security</td>
<td>Gruschka, N.; NEC Labs. Eur., Heidelberg, Germany; Jensen, M.; Iacono, L.L.; Luttenberger, N.</td>
<td>1 0 1 1 1 0 1 0 0 5</td>
</tr>
<tr>
<td>SOAP-Based vs. RESTful Web Services: A Case Study for Multimedia Conferencing</td>
<td>Belqasmi, F.; Singh, J.; Bani Melhem, S.Y.; Glitho, R.H.</td>
<td>1 0 1 1 1 0 1 0 0 5</td>
</tr>
<tr>
<td>Software defined cloud: Survey, system and evaluation</td>
<td>Yaser Jararweh, Mahmoud Al-Ayyoub, Ala’ Darabshe, Elhadj Benkhelifa, Miaden Vouk, Andy Rindos</td>
<td>1 1 1 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Splitting the HTTPS Stream to Attack Secure Web Connections</td>
<td>Prandini, M.; Ramilli, M.; Cerroni, W.; Callegati, F.</td>
<td>1 1 1 1 1 0 1 0 0 6</td>
</tr>
<tr>
<td>State-of-the-art Survey on Cloud Computing Security Challenges, Approaches and Solutions</td>
<td>Farrukh Shahzada</td>
<td>1 1 1 0 0 1 1 1 1 7</td>
</tr>
<tr>
<td>State-of-the-art Survey on Cloud Computing Security Challenges, Approaches and Solutions</td>
<td>Farrukh Shahzada</td>
<td>1 1 0 0 0 1 1 1 1 6</td>
</tr>
<tr>
<td>Still on the hook: the persistent problem of phishing</td>
<td>Steven Furnell</td>
<td>1 1 1 1 1 0 0 0 0 5</td>
</tr>
<tr>
<td>Study on the security models and strategies of cloud computing</td>
<td>Jianhua Chea, Yamin Duab, Tao Zhang, Jie Fana</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Survey of data and storage security in cloud computing</td>
<td>s.rajeswari, r.kalaiselvi</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Survey Paper On Cloud Computing Security</td>
<td>Wg Cdr Nimit Kaura Lt Col Abhishek Lal</td>
<td>1 1 0 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Title</td>
<td>Authors</td>
<td>References</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>------------</td>
</tr>
<tr>
<td>Surveying and Analyzing Security, Privacy and Trust Issues in Cloud</td>
<td>Dawei Suna, Guiran Chang, Lina Suna and Xingwei Wanga</td>
<td>1 1 1 1 1 1 1 1 9</td>
</tr>
<tr>
<td>Environments</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Targeted cyberattacks: a superset of advanced persistent threats</td>
<td>Sood, Aditya K; Enbody, Richard J</td>
<td>1 1 1 0 0 1 1 0 0 5</td>
</tr>
<tr>
<td>Taxonomy of Distributed Denial of Service mitigation approaches for</td>
<td>Alireza Shameli-Sendi, Makanpourzandi, Mohamedfekih-Ahmed, Mohamed</td>
<td>1 1 1 0 0 1 1 1 1 7</td>
</tr>
<tr>
<td>cloud computing</td>
<td>Cheriet</td>
<td></td>
</tr>
<tr>
<td>The Human Element of Information Security</td>
<td>Thompson, H.</td>
<td>1 1 1 0 0 1 1 0 0 5</td>
</tr>
<tr>
<td>The Impact of Virtualization on Network Performance of Amazon EC2</td>
<td>Guohui Wang; Ng, T.S.E.</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Data Center</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The State of Public Infrastructure-as-a-Service Cloud Security</td>
<td>Wei Huang, Afshar Ganjali, Beom Heyn Kim, Sukwon Oh, And David Lie,</td>
<td>1 1 1 0 0 0 0 1 1 5</td>
</tr>
<tr>
<td>The Threat in the Cloud</td>
<td>Green, M.</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>The Trusted Cloud Transfer Protocol</td>
<td>Mathias Slawik</td>
<td>1 1 1 0 0 1 1 1 1 7</td>
</tr>
<tr>
<td>Threat as a Service? Virtualization’s Impact on Cloud Security</td>
<td>Hsin-Yi Tsai, Ralf Steinmetz, Yu-Lun Huang, Melanie Siebenhaar and André</td>
<td>1 1 1 0 0 0 0 1 1 5</td>
</tr>
<tr>
<td>Miede,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Three Step Data Security Model for Cloud Computing based on RSA and</td>
<td>Vinay Kumar Pant, Jyoti Prakash, Amit Asthana</td>
<td>1 1 1 0 0 1 1 1 1 7</td>
</tr>
<tr>
<td>Steganography Techniques</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Toward a secure and usable cloud-based password manager for web</td>
<td>Rui Zhao, Chuan Yue</td>
<td>1 1 1 0 0 1 1 1 1 7</td>
</tr>
<tr>
<td>browsers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Toward publicly auditable secure cloud data storage services</td>
<td>Wang, Cong; Ren, Kui; Lou, Wenjing; Li, Jin</td>
<td>1 1 1 0 0 0 0 1 1 5</td>
</tr>
<tr>
<td>Towards Achieving Data Security with the Cloud Computing Adoption</td>
<td>Victor Chang And Muthu Ramachandran</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Framework</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Towards quantification and evaluation of security of Cloud Service</td>
<td>Talal Halabi, Martine Bellaiche</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Providers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Towards Secure Inter-Cloud Architectures</td>
<td>OSCAR ENCINA C, EDUARDO B. FERNANDEZ, RAÚL MONGE A</td>
<td>1 1 1 1 0 0 1 1 1 7</td>
</tr>
<tr>
<td>Trends and Research Directions for Privacy Preserving approaches on</td>
<td>Tanmay Sinha, VRNS Srikanth, Mangal Sain &amp; Hoon Jae Lee</td>
<td>1 0 1 0 1 1 1 1 1 6</td>
</tr>
<tr>
<td>the Cloud</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trust as a facilitator in cloud computing: a survey</td>
<td>Sheikh Mahbub Habib, Sascha Hauke, Sebastian Ries and Max Muhlhauser</td>
<td>1 1 1 1 0 1 1 1 1 8</td>
</tr>
<tr>
<td>Trust in the Cloud</td>
<td>Abbadi, Imad M.; Martin, Andrew</td>
<td>1 1 1 0 0 0 1 1 1 6</td>
</tr>
<tr>
<td>Twenty Security Considerations for Cloud-Supported Internet of Things</td>
<td>Jatinder Singh, Thomas Pasquier, Jean Bacon, Hajoon Ko, And David Eyers</td>
<td>1 0 1 0 0 0 1 1 1 5</td>
</tr>
<tr>
<td>Two layer Security for data storage in cloud</td>
<td></td>
<td>1 0 1 0 0 1 1 1 1 6</td>
</tr>
<tr>
<td>Authors</td>
<td>Title</td>
<td>Score</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-----------------------------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>A. Ashok Kumar, Santhosha, A.Jagan</td>
<td>Understanding the Security Challenges of Oblivious Cloud Storage with Asynchronous Accesses</td>
<td>1100011116</td>
</tr>
<tr>
<td>Cetin Sahin, Aaron Magat, Victor Zakhary, Amr El Abbadi, Huijia (Rachel) Lin, Stefano Tessaro</td>
<td>Verifying Cloud Services: Present and Future Sara Bouchenak, Gregory Chockler, Hana Chockler, Gabriela Gheorghe, Nuno Santos, Alexander Shraer</td>
<td>1110000115</td>
</tr>
<tr>
<td>Cetin Sahin, Aaron Magat, Victor Zakhary, Amr El Abbadi, Huijia (Rachel) Lin, Stefano Tessaro</td>
<td>Virtualization layer security challenges and intrusion detection/prevention systems in cloud computing: a comprehensive review Chirag N. Modi · Kamatchi Acha</td>
<td>1100111106</td>
</tr>
<tr>
<td>Cetin Sahin, Aaron Magat, Victor Zakhary, Amr El Abbadi, Huijia (Rachel) Lin, Stefano Tessaro</td>
<td>Virtualization security for cloud computing service Shengmei Luo; Zhaoji Lin; Xiaohua Chen; Zhuolin Yang; Jianyong Chen</td>
<td>1010001115</td>
</tr>
<tr>
<td>Cetin Sahin, Aaron Magat, Victor Zakhary, Amr El Abbadi, Huijia (Rachel) Lin, Stefano Tessaro</td>
<td>Virtualization: Issues, Security Threats, and Solutions MICHAEL PEARCE,SHERALI ZEADALLY, RAY HUNT.</td>
<td>1111011006</td>
</tr>
<tr>
<td>Cetin Sahin, Aaron Magat, Victor Zakhary, Amr El Abbadi, Huijia (Rachel) Lin, Stefano Tessaro</td>
<td>Virtualizing networking and security in the cloud Basak, Debasish; Toshniwal, Rohit; Maskalik, Serge; Sequeira, Allwyn</td>
<td>1010011116</td>
</tr>
<tr>
<td>Cetin Sahin, Aaron Magat, Victor Zakhary, Amr El Abbadi, Huijia (Rachel) Lin, Stefano Tessaro</td>
<td>Vulnerabilities and storage security in Cloud Computing Mostapha Derfouf, Amina Mimouni, Mohsine Eleuldj</td>
<td>1100011116</td>
</tr>
<tr>
<td>Cetin Sahin, Aaron Magat, Victor Zakhary, Amr El Abbadi, Huijia (Rachel) Lin, Stefano Tessaro</td>
<td>XSS-secure as a service for the platforms of online social network-based multimedia web applications in cloud Shashank Gupta &amp; B. B. Gupta</td>
<td>1010011105</td>
</tr>
</tbody>
</table>
APPENDIX C

Publications Years Distribution

<table>
<thead>
<tr>
<th>Years</th>
<th>Publications Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010</td>
<td>10</td>
</tr>
<tr>
<td>2011</td>
<td>25</td>
</tr>
<tr>
<td>2012</td>
<td>34</td>
</tr>
<tr>
<td>2013</td>
<td>37</td>
</tr>
<tr>
<td>2014</td>
<td>25</td>
</tr>
<tr>
<td>2015</td>
<td>29</td>
</tr>
<tr>
<td>2016</td>
<td>35</td>
</tr>
<tr>
<td>2017</td>
<td>41</td>
</tr>
<tr>
<td>2018</td>
<td>15</td>
</tr>
</tbody>
</table>

Publications Distribution (Subjects)

- Cloud Security: 64%
- Related Technology: 16%
- Both: 20%
Publications Distribution Based on Research Methodology

- Proposing New Framework/method/technique
- Comparing and Evaluating
- Analysing and Explaining
- Challenges and issues management
- Case Study

Publications Distribution Based on Contents

- Grounded Theory: 12
- Systemic Study: 64
- Case Study: 37
- Survey: 39
- Empirical Study: 39
- Not Clear: 59
<table>
<thead>
<tr>
<th>Category</th>
<th>Issue</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Software</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Platform</strong></td>
<td>Isolation</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>Accounting the pooled resources</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Safe termination</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td><strong>Front-end</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Unauthorised access</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>Direct internet connection</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Masked-code injection</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Open source software</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>Un-Guaranteed delivery</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>VMM</td>
<td>7</td>
</tr>
<tr>
<td><strong>Availability</strong></td>
<td>DoS</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2%</td>
</tr>
<tr>
<td><strong>Virtualisation</strong></td>
<td>large encrypted image</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>Image thieving</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Un-authorised image access</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td><strong>Virtualisation Management</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sprawl</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Virtual network</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>Virtual Machine monitor</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>Escape attack and Zero-day</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td><strong>Mobility</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>VM cloning</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>VM mobility</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td><strong>Malware</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>malware injection</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>side-channel and covert-channel</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td><strong>Protocols and standards</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Roll-back function</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Network-based cross tenant attacks</td>
<td>10</td>
</tr>
</tbody>
</table>

### Appendix D

237
<table>
<thead>
<tr>
<th>Category</th>
<th>Subcategory</th>
<th>Percentage</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web services</td>
<td>HTTP and HTTPS over PaaS</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td></td>
<td>XML wrapping attack</td>
<td>13</td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>Scanning and spoofing attack</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>Web Technologies</td>
<td>Unauthenticated session management</td>
<td>5</td>
<td>1%</td>
</tr>
<tr>
<td></td>
<td>Flood attacks</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>DoS attacks</td>
<td>12</td>
<td>6%</td>
</tr>
<tr>
<td></td>
<td>Bandwidth under-provision</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Connectivity</td>
<td>Monitoring</td>
<td>11</td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>Third party</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Social engineering</td>
<td>8</td>
<td>4%</td>
</tr>
<tr>
<td></td>
<td>Password attack</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Trust</td>
<td>Monitoring</td>
<td>11</td>
<td>5%</td>
</tr>
<tr>
<td></td>
<td>Third party</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Social engineering</td>
<td>8</td>
<td>4%</td>
</tr>
<tr>
<td></td>
<td>Password attack</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>Storage &amp; Computation</td>
<td>Data storage</td>
<td>28</td>
<td>12%</td>
</tr>
<tr>
<td></td>
<td>geographical distribution</td>
<td>27</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Loss of control</td>
<td>27</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cryptography</td>
<td>22</td>
<td>10%</td>
</tr>
<tr>
<td></td>
<td>insecure cryptography</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Algorithm failure</td>
<td>24</td>
<td>10%</td>
</tr>
<tr>
<td></td>
<td>Malware</td>
<td>20</td>
<td>4%</td>
</tr>
<tr>
<td></td>
<td>Syncing via data</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Sanitisation</td>
<td>5</td>
<td>1%</td>
</tr>
<tr>
<td></td>
<td>cloud recycling</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Unauthorised access</td>
<td>30</td>
<td>12%</td>
</tr>
<tr>
<td></td>
<td>Unauthorised access</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Malicious attacks</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Physical access</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Malicious attacks</td>
<td>28</td>
<td></td>
</tr>
<tr>
<td>Access</td>
<td>Authentication</td>
<td>22</td>
<td>22%</td>
</tr>
<tr>
<td></td>
<td>Password attack</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Account lock out</td>
<td>13</td>
<td>10%</td>
</tr>
<tr>
<td></td>
<td>Identity management</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>

238
APPENDIX E

Main Activity

// the main activity

public class MainActivity extends AppCompatActivity {

// buttons to move to protected file list, CS, and file selection

    private Button csButton = null;
    private Button flButton = null;
    private Button pcButton = null;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        csButton = (Button) findViewById(R.id.goCS);
        csButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), CS.class);
                startActivity(i);
            }
        });

        flButton = (Button) findViewById(R.id.goFileSelection);
        flButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), Pass_Code_File.class);
                startActivity(i);
            }
        });

        pcButton = (Button) findViewById(R.id.goProtectedFiles);
        pcButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), Pass_Code.class);
                startActivity(i);
            }
        });
    }
}
public class CS extends AppCompatActivity {

    private Button mainButton = null;
    private Button flButton = null;
    private Button pcButton = null;
    private Button dropboxButton = null;
    private Button boxButton = null;
    private Button onedriveButton = null;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_cs);

        dropboxButton = (Button) findViewById(R.id.goDropBox);
        dropboxButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), DropBoxActivity.class);
                startActivity(i);
            }
        });

        onedriveButton = (Button) findViewById(R.id.goOnedrive);
        onedriveButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), OneDriveActivity.class);
                startActivity(i);
            }
        });

        boxButton = (Button) findViewById(R.id.goBox);
        boxButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), BoxActivity.class);
                startActivity(i);
            }
        });

        mainButton = (Button) findViewById(R.id.goMain);
        mainButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), MainActivity.class);
                startActivity(i);
            }
        });

        flButton = (Button) findViewById(R.id.goFileSelection);
    }
}
public class Pass_Code  extends AppCompatActivity {
    // the passcode activity the generate the dynamic keypad that changes every time when the activity established

    Button mainButton = null;
    Button B1 = null;
    Button B2 = null;
    Button B3 = null;
    Button B4 = null;
    Button B5 = null;
    Button B6 = null;
    Button B7 = null;
    Button B8 = null;
    Button B9 = null;
    Button B0 = null;
    Button clr = null;
    Button show_pass = null;
    EditText Pass = null;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.pass_code);
        mainButton = (Button) findViewById(R.id.goMain);
        B1 = (Button) findViewById(R.id.button2);
        B2 = (Button) findViewById(R.id.button3);
        B3 = (Button) findViewById(R.id.button4);
        B4 = (Button) findViewById(R.id.button5);
        B5 = (Button) findViewById(R.id.button6);
        B6 = (Button) findViewById(R.id.button7);
        B7 = (Button) findViewById(R.id.button8);
        B8 = (Button) findViewById(R.id.button9);
        B9 = (Button) findViewById(R.id.button10);
        B0 = (Button) findViewById(R.id.button12);
        clr = (Button) findViewById(R.id.button11);
        show_pass = (Button) findViewById(R.id.button13);
        Pass = (EditText) findViewById(R.id.editText);
    }
}
mainButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        Intent i = new Intent(view.getContext(), MainActivity.class);
        startActivity(i);
    }
});

// generate the values and assign these values to an array in order to sort the keypad

// the loops aim to avoid button duplication

int i = 0;
int j = 0;
int[] Array_Key = new int[10];
Random generator = new Random();
int rand_nu = 0;

for (i = 0; i < 10; i++) {
    rand_nu = generator.nextInt(10);
    for (j = 0; j <= i; j++) {
        if (rand_nu == Array_Key[j]) {
            rand_nu = generator.nextInt(10);
            j = 0;
        }
    }

    Array_Key[i] = rand_nu;
}

for (i = 0; i < 10; i++) {
    for (j = 0; j < 10; j++) {
        if (Array_Key[i] == Array_Key[j] && i != j) {
            Array_Key[j] = 0;
        }
    }
}

B1.setText("" + Array_Key[0]);
B2.setText("" + Array_Key[1]);
B3.setText("" + Array_Key[2]);
B4.setText("" + Array_Key[3]);
B5.setText("" + Array_Key[4]);
B6.setText("" + Array_Key[5]);
B7.setText("" + Array_Key[6]);
B8.setText("" + Array_Key[7]);
B9.setText("" + Array_Key[8]);
B0.setText("" + Array_Key[9]);

B1.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v1) {
        Pass.setText(Pass.getText().toString() + B1.getText());
    }
});

B2.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v2) {
        Pass.setText(Pass.getText().toString() + B2.getText());
    }
});

B3.setOnClickListener(new View.OnClickListener() {"
public void onClick(View v) {
    Pass.setText(Pass.getText().toString() + B3.getText());
}

B4.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v4) {
        Pass.setText(Pass.getText().toString() + B4.getText());
    }
});

B5.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v5) {
        Pass.setText(Pass.getText().toString() + B5.getText());
    }
});

B6.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v6) {
        Pass.setText(Pass.getText().toString() + B6.getText());
    }
});

B7.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v7) {
        Pass.setText(Pass.getText().toString() + B7.getText());
    }
});

B8.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v8) {
        Pass.setText(Pass.getText().toString() + B8.getText());
    }
});

B9.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v9) {
        Pass.setText(Pass.getText().toString() + B9.getText());
    }
});

B0.setOnClickListener(new View.OnClickListener() {
    public void onClick(View v0) {
        Pass.setText(Pass.getText().toString() + B0.getText());
    }
});

clr.setOnClickListener(new View.OnClickListener() {
    public void onClick(View vclr) {
        Pass.setText(""");
    }
});

show_pass.setOnClickListener(new View.OnClickListener() {
    public void onClick(View vshow) {
        if (Pass.getText().toString().equals("1111")) {
            Intent ii = new Intent(vshow.getContext(), TodoListActivity.class);
            startActivity(ii);
        } else {
            Toast.makeText(Pass_Code.this, "WRONG PASSWORD ...", Toast.LENGTH_SHORT).show();
        }
    }
});

public class DBHelper extends SQLiteOpenHelper {

    public static final String TABLE_NAME = "phdtest3";

    // declaring the table items
    public static final String _ID = "_id";
    public static final String ORIGNAL_PATH = "file_location";
    public static final String CHUNK1_PATH = "chunk1";
    public static final String CHUNK2_PATH = "chunk2";
    public static final String CHUNK3_PATH = "chunk3";
    public static final String PHONE_PATH = "phone_path";
    public static final String CLOUD1_PATH = "cloud1";
    public static final String CLOUD2_PATH = "cloud2";
    public static final String CLOUD3_PATH = "cloud3";

    // declaring the database version
    static final int DB_VERSION = 1;

    // creating table SQL query
    private static final String CREATE_TABLE = "create table " + TABLE_NAME + "(" + _ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " + ORIGNAL_PATH + " TEXT, " + CHUNK1_PATH + " TEXT, " + CHUNK2_PATH + " TEXT, " + CHUNK3_PATH + " TEXT, " + PHONE_PATH + " TEXT, " + CLOUD1_PATH + " TEXT, " + CLOUD2_PATH + " TEXT, " + CLOUD3_PATH + " TEXT);";

    public DBHelper(Context context) {
        super(context, DB_NAME, null, DB_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(CREATE_TABLE);
    }

    // avoid duplications
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
        onCreate(db);
    }

    // insert function
    public void insert(String filepath, String chunk1path, String chunk2path,
                        String chunk3path, String key, String phonepath, String cloud1path, String cloud2path, String cloud3path) {
        ContentValues contentValue = new ContentValues();
        contentValue.put(DBhelper.ORIGNAL_PATH, filepath);
        contentValue.put(DBhelper.CHUNK1_PATH, chunk1path);
        contentValue.put(DBhelper.CHUNK2_PATH, chunk2path);
        contentValue.put(DBhelper.CHUNK3_PATH, chunk3path);
        contentValue.put(DBhelper.PHONE_PATH, phonepath);
        contentValue.put(DBhelper.CLOUD1_PATH, cloud1path);
        contentValue.put(DBhelper.CLOUD2_PATH, cloud2path);
        contentValue.put(DBhelper.CLOUD3_PATH, cloud3path);
    }
```java
public Cursor fetch() {
    String[] columns = new String[] {
        DBhelper._ID, DBhelper.ORIGINAL_PATH,
        DBhelper.CHUNK1_PATH, DBhelper.CHUNK2_PATH, DBhelper.CHUNK3_PATH,
        DBhelper.PHONE_PATH,
        DBhelper.CLOUD1_PATH, DBhelper.CLOUD2_PATH,
        DBhelper.CLOUD3_PATH
    };
    Cursor cursor = database.query(DBhelper.TABLE_NAME, columns, null, null, null, null, null);
    if (cursor != null) {
        cursor.moveToFirst();
    }
    return cursor;
}

// update query
public int update(long _id, String filepath, String chunk1path, String chunk2path, String chunk3path,
                  String phonepath, String cloud1path, String cloud2path, String cloud3path) {
    ContentValues contentValues = new ContentValues();
    contentValues.put(DBhelper.ORIGINAL_PATH, filepath);
    contentValues.put(DBhelper.CHUNK1_PATH, chunk1path);
    contentValues.put(DBhelper.CHUNK2_PATH, chunk2path);
    contentValues.put(DBhelper.CHUNK3_PATH, chunk3path);
    contentValues.put(DBhelper.PHONE_PATH, phonepath);
    contentValues.put(DBhelper.CLOUD1_PATH, cloud1path);
    contentValues.put(DBhelper.CLOUD2_PATH, cloud2path);
    contentValues.put(DBhelper.CLOUD3_PATH, cloud3path);
    int i = database.update(DBhelper.TABLE_NAME, contentValues, DBhelper._ID + " = " + _id, null);
    return i;
}

// delete query
public void delete(long _id) {
    database.delete(DBhelper.TABLE_NAME, DBhelper._ID + "=" + _id, null);
}
```
public class BoxActivity extends ActionBarActivity implements BoxAuthentication.AuthListener {

    // sessions initialising
    BoxSession mSession = null;
    BoxSession mOldSession = null;

    public String uploadfilepath = Environment.getExternalStorageDirectory();

    // box client configuration
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_box);
        mListView = (ListView) findViewById(android.R.id.list);
        mAdapter = new BoxItemAdapter(this);
        mListView.setAdapter(mAdapter);
        BoxConfig.IS_LOG_ENABLED = true;
        BoxConfig.CLIENT_ID = "8wsh9q14m4iwxl6q6p6e6y2hsbtjpc";
        BoxConfig.CLIENT_SECRET = "S4T93CH2ucF4pselHT8YrEh64ZBypjRY";
        initialize();
    }

    public void initialize() {
        mAdapter.clear();
        mSession = new BoxSession(this);
        mSession.setSessionAuthListener(this);
        mSession.authenticate();
    }

    public void loadRootFolder() {
        new Thread() {
            @Override
            public void run() {
                try {
                    final BoxListItems folderItems = mFolderApi.getItemsRequest("0").send();
                    runOnUiThread(new Runnable() {
                        @Override
                        public void run() {
                            mAdapter.addAll(folderItems);
                        }
                    });
                } catch (BoxException e) {
                    e.printStackTrace();
                }
            }
        }.start();
    }

    // file uploading
    public void uploadboxFile(final String up_path) {
        mDialog = ProgressDialog.show(BoxActivity.this, null, "Uploading...", false, true);
        // file uploading logic
    }
}
new Thread() {
    @Override
    public void run() {
        try {
            InputStream fis = new FileInputStream(up_path);
            String destinationFolderId = "0";
            String uploadName = "";
            BoxRequestsFile.UploadFile request = mFileApi.getUploadRequest(fis, uploadName, destinationFolderId);
            final BoxFile uploadFileInfo = request.send();
            showToast("download " + uploadFileInfo.getName());
            loadRootFolder();
        } catch (IOException e) {
            e.printStackTrace();
        } catch (BoxException e) {
            e.printStackTrace();
            BoxError error = e.getAsBoxError();
            if (error != null && error.getStatus() == HttpStatus.SC_CONFLICT) {
                ArrayList<BoxEntity> conflicts = error.getContextInfo().getConflicts();
                if (conflicts != null && conflicts.size() == 1 && conflicts.get(0) instanceof BoxFile) {
                    uploadboxNewVersion((BoxFile) conflicts.get(0));
                    return;
                }
            }
            showToast("Upload failed");
        }
    }
}).start();

// file downloading
// some prints in order to facilitate tracking the requests and identifying
public void downloadboxfile() {
    mDialog = ProgressDialog.show(BoxActivity.this, getText(R.string.boxsdk_Please_wait), getText(R.string.boxsdk_Please_wait));
    new Thread() {
        @Override
        public void run() {
            try {
                OutputStream fis = new FileOutputStream(Environment.getExternalStorageDirectory());
                ByteArrayOutputStream outstream = new ByteArrayOutputStream();
                String destinationFolderId = "";
                BoxRequestsFile.DownloadFile fileDownload = mFileApi.getDownloadRequest(outstream, destinationFolderId);
                System.out.println("***************************     download started    ***********************");
                fileDownload.send();
                outstream.writeTo(fis);
                fis.close();
                loadRootFolder();
            } catch (IOException e) {
                System.out.println("*******************    File Not Found *******************");
                e.printStackTrace();
            } catch (BoxException e) {
                System.out.println("***************");
                e.printStackTrace();
            }
        }
    }.start();
}
public boolean clearAdapter() {
    runOnUiThread(new Runnable() {
        @Override
        public void run() {
            mAdapter.clear();
        }
    });
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.box_menu, menu);
    return true;
}

@Override
public boolean onPrepareOptionsMenu(Menu menu) {
    int numAccounts = BoxAuthentication.getInstance().getStoredAuthInfo(this).keySet().size();
    menu.findItem(R.id.logoutAll).setVisible(numAccounts > 1);
    menu.findItem(R.id.logout).setVisible(numAccounts > 0);
    menu.findItem(R.id.switch_accounts).setTitle(numAccounts > 0 ? R.string.switch_accounts : R.string.login);
    return true;
}

// box menu items
public boolean onOptionsItemSelected(MenuItem item) {
    int id = item.getItemId();
    if (id == R.id.upload) {
        downloadboxfile();
        return true;
    } else if (id == R.id.switch_accounts) {
        switchAccounts();
        return true;
    } else if (id == R.id.logout) {
        mSession.logout();
        return true;
    } else if (id == R.id.logoutAll) {
        new Thread() {
            @Override
            public void run() {
                BoxAuthentication.getInstance().logoutAllUsers(getApplicationContext());
            }
        }.start();
        return true;
    }
    return super.onOptionsItemSelected(item);
}

public void switchAccounts() {
    mOldSession = mSession;
mSession = new BoxSession(this, null);
mSession.setSessionAuthListener(this);
mSession.authenticate().addOnCompletedListener(new BoxFutureTask.OnCompletedListener<BoxSession>() {
    @Override
    public void onCompleted(BoxResponse<BoxSession> response) {
        if (response.isSuccess()) {
            clearAdapter();
onAuthCreated(mSession.getAuthInfo());
        }
    }
});

@Override
public void onAuthCreated(BoxAuthentication.BoxAuthenticationInfo info) {
    mFolderApi = new BoxApiFolder(mSession);
mFileApi = new BoxApiFile(mSession);
    loadRootFolder();
}

// authentication and logout functions
@override
public void onAuthFailure(BoxAuthentication.BoxAuthenticationInfo info,
Exception ex) {
    if (ex != null) {
        clearAdapter();
    } else if (info == null && mOldSession != null) {
        mSession = mOldSession;
mOldSession = null;
onAuthCreated(mSession.getAuthInfo());
    }
}

@Override
public void onLoggedOut(BoxAuthentication.BoxAuthenticationInfo info,
Exception ex) {
    clearAdapter();
    initialize();
}

public class BoxItemAdapter extends ArrayAdapter<BoxItem> {
    public BoxItemAdapter(Context context) {
        super(context, 0);
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        BoxItem item = getItem(position);
        if (convertView == null) {
            convertView = LayoutInflater.from(getContext()).inflate(R.layout.boxsdk_list_item, parent,
false);
        }

        TextView name = (TextView) convertView.findViewById(R.id.name);
        name.setText(item.getName());
        ImageView icon = (ImageView) convertView.findViewById(R.id.icon);
        if (item instanceof BoxFolder) {
            icon.setImageResource(R.drawable.boxsdk_icon_folder_yellow_private);
        } else {
            icon.setImageResource(R.drawable.boxsdk_generic);
        }
    }
}
Dropbox activity

public class DropBoxActivity extends AppCompatActivity {

// dropbox main activity
private Button mainpage = null;
public Button updrobbox = null;

private static final int IMAGE_REQUEST_CODE = 101;
public String ACCESS_TOKEN;

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_dropbox);
    mainpage = (Button) findViewById(R.id.goMainPage);
    mainpage.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent i = new Intent(view.getContext(), MainActivity.class);
            startActivity(i);
        }
    });
    // ask for sign in if there is no available active token
    if (!tokenExists()) {
        Intent intent = new Intent(DropBoxActivity.this, LoginActivity.class);
        startActivityForResult(intent);
    }
    ACCESS_TOKEN = retrieveAccessToken();
    getUserAccount();

    FloatingActionButton fab = (FloatingActionButton) findViewById(R.id.fab);
    fab.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            upload();
        }
    });
    // dropbox sign in activity
    public void getUserAccount() {
        if (ACCESS_TOKEN == null) return;
    }
}
new UserAccountTask(DropboxClient.getClient(ACCESS_TOKEN), new UserAccountTask.TaskDelegate() {
    @Override
    public void onAccountReceived(FullAccount account) {
        Log.d("User data", account.getEmail());
        Log.d("User data", account.getName().getDisplayName());
        Log.d("User data", account.getAccountType().name());
        updateUI(account);
    }

    @Override
    public void onError(Exception error) {
        Log.d("User data", "Error receiving account details.");
    }
}).execute();

public boolean tokenExists() {
    SharedPreferences prefs = getSharedPreferences("com.example.hassan.v1", Context.MODE_PRIVATE);
    String accessToken = prefs.getString("access-token", null);
    return accessToken != null;
}

public String retrieveAccessToken() {
    SharedPreferences prefs = getSharedPreferences("com.example.hassan.v1", Context.MODE_PRIVATE);
    String accessToken = prefs.getString("access-token", null);
    if (accessToken == null) {
        Log.d("AccessToken Status", "No token found");
        return null;
    } else {
        Log.d("AccessToken Status", "Token exists");
        return accessToken;
    }
}

// dropbox upload function
public void dropboxupload() {
    File file = new File(FileDialog.START_PATH, "/sdcard");
    if (file != null) {
        //Initialize UploadTask
        new UploadTask(DropboxClient.getClient(ACCESS_TOKEN), file, getApplicationContext()).execute();
        Toast.makeText(DropBoxActivity.this, "UPLOAD ... DONE !!!", Toast.LENGTH_LONG).show();
    }
}

// dropbox download function
public void dropboxdownload() {
    File afile = new File("/storage/sdcard/newdropbox.zip");
    new DownloadTask(DropboxClient.getClient(ACCESS_TOKEN), afile, getApplicationContext()).execute();
}
OnClickListener loadButtonListner = new OnClickListener() {
    public void onClick(View v) {
        // start browsing the device file when LOAD button clicked, and make the root is 
        // "/sdcard"
        Intent intent = new Intent(v.getContext(), FileDialog.class);
        intent.putExtra(FileDialog.START_PATH, "/sdcard"علوم
        startActivityForResult(intent, SELECT_FILE);
    }
};

// update the displayed information to match with the selected file
public void changeFileNameText(String newFileName) {
    TextView t = (TextView) findViewById(R.id.fileText);
    t.setText(newFileName);
}

public void changeMD5Text(String newMD5) {
    TextView t = (TextView) findViewById(R.id.mdfive);
    t.setText(newMD5);
}

public void updateFileSizeDialog(long fileSizeInBytes) {
    float fileSizeInKilos = fileSizeInBytes / 1024;
    float fileSizeInMegs = fileSizeInKilos / 1024;
    TextView t = (TextView) findViewById(R.id.fileSize);
    t.setText(fileSizeInMegs + " Megabytes");
}

// calculate and display the file and segments sizes
// calculating the file size in kilobytes
    float fileSizeInKilos = fileSizeInBytes / 1024;
// calculating the file size in megabytes
    float fileSizeInMegs = fileSizeInKilos / 1024;
    int dotposition;
// declaring a float to save the segment size
    float ll;
// displaying the file size in megabytes
    TextView t = (TextView) findViewById(R.id.fileSize);
    t.setText(fileSizeInMegs + " Megabytes");
    EditText ff = (EditText) findViewById(R.id.splitsize);
// calculating the segments size
    ll = fileSizeInMegs / 3;
    String hh = Float.toString(ll);
// defining the "." Position
    dotposition = hh.lastIndexOf(".");
    String cutval;
    cutval = hh.substring(dotposition, 5);
    ff.setText(cutval);
// declaring an array to hold the splitted bytes
byte[] kilobuffer = new byte[1024];

try {

// starting the splitting process while the input stream not empty
while (is.read(kilobuffer) >= 1) {
    kilobyteCounter++;
}

// if statement to split the stream base on the defined size
if (kiloByteCounter > splitValInKiloBytes) {
    kiloByteCounter = 1;
    suffixCounter++;
}

// name the segment
newfilePath = filePath
    + getStringSuffixFromInt(suffixCounter, splitValInKiloBytes);

// writing the segment path to array before inserting to database
chunk_path_list[suffixCounter-1] = newfilePath;

// creating a record and inserting the file and segments information into the database
insert_intent.putExtra("Original File Path", filePath);
insert_intent.putExtra("CHUNK 1", chunk_path_list[0]);
insert_intent.putExtra("CHUNK 2", chunk_path_list[1]);
insert_intent.putExtra("CHUNK 3", chunk_path_list[2]);
insert_intent.putExtra("PHONE PATH", chunk_path_list[3]);
startActivity(insert_intent);
os.close();
}

// declaring array input stream, and write one segment to each variable
ByteArrayInputStream bais1 = new ByteArrayInputStream(ff1);
ByteArrayInputStream bais2 = new ByteArrayInputStream(ff2);
ByteArrayInputStream bais3 = new ByteArrayInputStream(ff3);
ByteArrayInputStream bais4 = new ByteArrayInputStream(ff4);

// combining the input streams into a single sequence in order to join the segments
SequenceInputStream sis1 = new SequenceInputStream(bais1, bais2);
SequenceInputStream sis2 = new SequenceInputStream(bais3, bais4);
SequenceInputStream sis = new SequenceInputStream(sis1, sis2);

// declaring and creating a new file to write and save the combined segments
newFileName = (Environment.getExternalStorageDirectory() + "/joind.jpg");

// reading the single sequence stream till the end, and write it to joining file
while ((ch = sis.read()) != -1) {
    try {
        os.write(ch);
    }
}

// error catching
catch (IOException e) {
    System.out.println("Error write files");
lastererror = 1;
    break;}}
Toast.makeText(getBaseContext(), "Successfully Joined Into File "+ ".", Toast.LENGTH_SHORT).show();
}

File encrypt and decrypt

// encrypting the file by passing the secret key and the file

private byte[] encrypt(byte[] secretk, byte[] efile) {
    SecretKeySpec skeySpec = new SecretKeySpec(secretk, "AES");
    Cipher cipher;

    // declaring an array to hold the encrypted file
    byte[] encrypted = null;
    try {
        // selecting the wanted encryption algorithm AES
        cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        // switch to the encryption mode
        cipher.init(Cipher.ENCRYPT_MODE, skeySpec, new IvParameterSpec(iv));

        // encrypting the file and hold the outcomes
       encrypted = cipher.doFinal(efile);

        // try to catch error
        } catch (Exception e) {
            e.printStackTrace();
        }

    return encrypted;
}

// decrypting the file by passing the secret key and the file stream

private byte[] decrypt(byte[] secretk, FileInputStream finputs) {
    SecretKeySpec skeySpec = new SecretKeySpec(secretk, "AES");
    Cipher cipher;

    // declaring and initiating variables to hold and save the decrypted data
    byte[] dData = null;
    CipherInputStream cinputs = null;
    try {

        // selecting the decryption algorithm AES
        cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        cipher.init(Cipher.DECRYPT_MODE, skeySpec, new IvParameterSpec(iv));

        // declaring and saving the input stream for decrypting
        cinputs = new CipherInputStream(finputs, cipher);

        // declaring and write the encrypted data to buffer (outputstream)
        ByteArrayOutputStream buffer = new ByteArrayOutputStream();
        byte[] data = new byte[8];
        while ((cinputs.read(data)) != -1) {
            buffer.write(data);
        }
        buffer.flush();

        // saving the decrypted data from the buffer to dData
        dData = buffer.toByteArray();
    }
// trying to catch error
   } catch (Exception e) {
       e.printStackTrace();
   }
finally{
    try {

      // closing the opened data stream
      finputs.close();
      cinputs.close();

      // try to catch error
      } catch (IOException e) {e.printStackTrace();}}

    return decryptedData;
}