The Islamic University–Gaza
Research and Postgraduate Affairs
Faculty of Information Technology
Master of Information Technology

الجـامعــــــــة الإســــلاميــة – غـزة
شئون البحث العلمي والدراسات العليا
كـليــــــة تـكنــولــوجيـا المـعلـومــات
ماجستير تـكنــولــوجيـا المـعلـومــات

A Model to Detect the Integrity Violation of Shared File in the Cloud

نموذج للتحقق من تكاملية الملفات المشاركة على السحابة

Safaa Taher Lulu

Supervised by

Dr.Tawfiq S.Barhoom

Associate Professor – Applied Computer Technology

A thesis submitted in partial fulfilment of the requirements for the degree of
Master of Information Technology

Dec. /2016

إقــــــرار

أنا الموقع أدناه مقدم الرسالة التي تحمل العنوان:

نموذج للتحقق من تكاملية الملفات المشاركة على السحابة

A Model to Detect the Integrity Violation of Shared File in the Cloud

Student's name:	صفاء طاهر لولو	اسم الطالب:
Signature:	صفاء لولو	التوقيع:
Date:	2017/4/30	التاريخ:

بسم الله الرحمن الرحيم

الجامعة الإسلامية – غزة
The Islamic University - Gaza

الجامعة الإسلامية غزة
The Islamic University – Gaza

مكتب نائب الرئيس للبحث العلمي والدراسات العليا   هاتف داخلي: 1150

الرقم ج س غ/35/ Ref

التاريخ 2017/01/31 Date

نتيجة الحكم على أطروحة ماجستير

بناءً على موافقة شئون البحث العلمي والدراسات العليا بالجامعة الإسلامية بغزة على تشكيل لجنة الحكم على أطروحة الباحثة/ **صفاء طاهر أحمد لولو** لنيل درجة الماجستير في **كليـة *تكنولوجيـا المعلومـات*** **برنامج تكنولوجيا المعلومات** وموضوعها:

نموذج للتحقق من تكاملية الملفات المشاركة على السحابة

A Model to Detect the Integrity of Shared File in the Cloud

وبعد المناقشة التي تمت اليوم الثلاثاء 04 جمادي الأولى 1438هـ، الموافق 2017/01/31م الساعة الحادية عشر صباحاً، اجتمعت لجنة الحكم على الأطروحة والمكونة من:

د. توفيق سليمان برهوم	مشرفاً و رئيساً	[illegible]
د. إياد محمد الأغا	مناقشاً داخلياً	الأغا
د. محمد أحمد غزال	مناقشاً خارجياً	[illegible]

وبعد المداولة أوصت اللجنة بمنح الباحثة درجة الماجستير في **كليـة *تكنولوجيـا المعلومـات*/ برنامج تكنولوجيا المعلومات.**

واللجنة إذ تمنحه هذه الدرجة فإنها توصيها بتقوى الله ولزوم طاعته وأن يسخر علمها في خدمة دينها ووطنها.

والله ولي التوفيق،،،

نائب الرئيس لشئون البحث العلمي والدراسات العليا

أ.د. عبدالرؤوف علي المناعمة

البحث العلمي والدراسات العليا
الجامعة الإسلامية غزة
The Islamic University
Research & Graduate Affairs

يرفع الله الذين آمنوا منكم والذين أوتوا العلم درجات

Abstract

Nowadays, there is an increasing trend in outsourcing data to remote cloud servers, where users store their data at specific storage space offered by Cloud service provider (CSP) with low cost.

Many of cloud service provider pretend that user's data is saved securely with no data integrity violation. But as we know, anything saved in the internet environment can't be perfectly safe which reduce users trust that their files in safe situation, then they can’t store sensitive data in the cloud.

So, it's vital to ensure that data are kept properly in a consistent way and data integrity is guaranteed. Surely, this advantage will be more useful when data is shared between a group of users.

Some previous works treated with file integrity violation problem from the perspective of just one file without dealing with file sharing. On the other hand, some works enabled Third-Party Verifier which cause some security fears.

In this work, we work on the problem of integrity of shared data over the cloud and a new model was built with dynamic data operation support. This model contains an intermediate system to detect file integrity violation by hashing the file, using SHA-256 algorithm, before saving it on the cloud and after retrieving it, then compare the two hash values.

Several experiments were conducted to ensure the ability of detecting data integrity violation, different sizes of .doc files starting from 13KB, ending with 5MB were collected. On the other hand, different types of tests were conducted to evaluate the accuracy and concurrency. The evaluation process proves that our intermediate system is able to guarantee Files accuracy and detect all type of file modifications. Also, it is able to organize concurrency operations on shared file and enable just one user to modify File at the same point. As it depends on clear logic of comparing File Hash values, before storing to the cloud and after retrieving from the cloud, so there is no way to make mistakes in detecting file integrity violation and it is highly efficient to detect integrity violation of shared files.

Keywords-cloud computing, integrity detection, shared file, dynamic data operations, accuracy, concurrency.

الملخص

يوجد توجه كبير نحو تخزين المعلومات في خوادم سحابية بعيدة هذه الأيام، حيث يخزن المستخدم بياناته في ذاكرة تخزين مقدمة من مزود الخدمة السحابية مقابل عائد مادي بسيط.

العديد من مزودي الخدمة السحابية يدعون أن بيانات المستخدم محفوظة بأمان تام بدون أي انتهاك لسلامة هذه البيانات. ولكن وكما نعلم، فإن أي شيء محفوظ على الإنترنت فهو عرضة للانتهاك ولا يمكن الجزم بأنه آمن بشكل مطلق، مما يقلل من ثقة المستخدمين بأن ملفاتهم في وضع آمن لذا فإنهم لا يستطيعون حفظ بياناتهم الحساسة على السحابة.

لذا أصبح من الضروري التأكد من أن البيانات محفوظة بشكل مستقر وأن سلامة البيانات محققة. وبالتأكيد فإن هذه الفائدة ستعود بالنفع بشكل أكبر في حال كانت هذه البيانات مشتركة بين مجموعة من المستخدمين.

بعض الأعمال السابقة تعاملت مع مشكلة انتهاك سلامة البيانات من منظور ملف واحد بدون التعامل مع الملفات المشتركة. من جهة أخرى قام البعض بتفعيل طرف وسيط للتحقق من سلامة البيانات مماسبب عدة مخاوف أمنية.

في هذا العمل، نعمل على مشكلة الكشف على سلامة بيانات الملفات المشتركة في السحابة لذا قمنا ببناء نموذج يدعم العمليات الديناميكية على البيانات. هذا النموذج يحتوي على نظام وسيط لكشف على انتهاك سلامة البيانات عن طريقة تطبيق خوارزمية هاش تدعى SHA-256 على الملف قبل حفظه على السحابة وبعد استرجاعه منها ومن ثم مقارنة الناتج في المرتين.

وللتأكد من قدرة النظام على الكشف على انتهاك سلامة البيانات، أجرينا العديد من التجارب باستخدام ملفات بامتداد (.doc) بأحجام مختلفة تتراوح بين (13KB – 5MB). ومن جهة أخرى، أقمنا العديد من الفحوصات لتقييم الدقة والتزامن في عمليات النظام. برهنت التحاليل أن النظام الوسيط قادر على ضمان دقة الملفات في الكشف على كل أنواع التغيير على محتويات الملف، وبما أن النظام يعتمد على لوجيك واضح في مقارنة قيم الهاش الخاصة بالملف قبل تخزينه على السحابة وبعد استرجاعه منها فلا مجال لحدوث خطأ في كشف انتهاك سلامة البيانات، لذا فهو يتمتع بكفاءة عالية في كشف انتهاك سلامة البيانات المشتركة المخزنة على السحابة. وهو قادر أيضا على تنظيم العمليات المتزامنة على الملفات المشتركة ولايسمح لأكثر من مستخدم بالتغيير في محتويات الملف في نفس الوقت.

إهـــداء

إلى النور الذي ينير لي درب النجاح.. **أبي الغالي**

إلى الدرة البراقة التي لايخبو بريقها.. **أمي الغالية**

إلى رفيق دربي وتوأم روحي.. **زوجي الغالي**

إلى سندي وعضدي في الحياة.. **أخي وأخواتي**

إلى أبنائي وبناتي.. **يزن، ألمى، لجين وعمر**

إلى أساتذتي وأصدقائي وكل من ساندني

إلى أرواح الشهداء الأبرار وأسرانا البواسل وكل من حارب وضحى من أجل هذا الوطن العظيم

Acknowledgment

All thanks and praises to **Allah** who granted me the strength, support, guidance and eased the difficulties, which I faced during the accomplishment of this thesis.

I would like to express my deep gratitude to my supervisor, **Dr. Tawfiq S.M. Barhoom**, Associated Professor of Information Technology in the Islamic University for his guidance and for giving me the opportunity to accomplish this research. He advised me during my work to present the research as clearly as possible.

I would like to extend my thanks to many people, for their advices, especially my **academic staff** in Islamic university, for teaching me, and helping me with my courses in my master's degree.

Special thanks for **my brother: Eng. Ahmed**, for his special help to accomplish this work.

Also, I would like to thank my family: **father**, **mother**, **sisters** and a special thank for **my husband: Tawfiq**, for his support and patience during my study.

I cannot forget my colleagues at **Woman Affairs Ministry**, my friends, and all who supported and encouraged me to continue my study.

Safaa Taher Lulu

Table of Contents

List of Tables

List of Figures

List of Abbreviations

CA	Certificate Authority
CMU	Carnegie Mellon University
CSP	Cloud Service Provider
Iaas	Infrastructure As A Service
IBC	Identity Based Cryptography
MD5	Message Digest algorithm
Paas	Platform As A Service
PHP	Hypertext Preprocessor
PKI	Public Key Infrastructure
POR	Proofs of Retrievability
RSA	Rivest-Shamir-Adleman
Saas	Software As A Service
SHA-256	Secure Hash Algorithm
SLA	Service Level Agreement
TPA	Third Party Auditor
TPV	Third-Party Verifier

Chapter 1

Introduction

Chapter 1
Introduction

With the rapid development of computer hardware and software, cloud computing is seen as the important change of information industry and will make more impact on the development of information technology for the society.

Currently, the majority of cloud computing infrastructure consists of reliable services delivered through data centre that are built on servers with different levels of virtualization technologies. Which can be accessed anywhere in the world (NetworkWorld, 2016).

Resources in cloud systems can be shared among number of users. Users can communicate with cloud service providers to retrieve their data. In addition, users can perform operations on the cloud such as insert, delete and update.

Businesses who use cloud service providers (CSP) tend to have service level agreements (SLA) that act as contract and define the level of expected service from the CSP, including availability, performance and security. Nowadays, many researchers suggest the use of third party auditors (TPA) as a mean to monitor CSPs (A-Mourad, 2014).

But there are some security risks involved with releasing control over user's data. One of them is the cloud service provider could intentionally or accidentally modify some data from the cloud server. Because the user does not know, how the cloud provider control or monitor his data (Barhoom and Saqallah, 2014).

So, the model must have some sort of mechanism, to detect the data integrity violation especially when group of users share this data.

In this work, a model to detect the integrity violation of shared file in the cloud is introduced. Also it includes dynamic data operations support on shared files (including data update, append and delete).

An intermediate system is built in PHP 5 Language to detect file integrity violation by hashing the file before saving it on the cloud and after retrieving it, then compare the two hash values. Also, we will focus on how to keep dynamic operations working in concurrency way, so no user can coflict the work of another user on a shared file.

Several experiments are conducted on our implementation to demonstrate its ability to detect data integrity violation, and to evaluate its accuracy, by enabling a group of users to share one file many times consecutively and detect the integrity violation of this file at each time. Also, concurrency is evaluated by enabling multiple users to access one file at the same time.

To accomplish our experiment, we collected a dataset, with different sizes starting with 13 KB, ending with 5MB with (.doc) type files only. The evaluation process proves that our model has been performed at an acceptable level as it is able to guarantee Files accuracy detected all type of file modifications.

Also, it is able to organize concurrency operations on shared file and enable just one user to modify File at the same point.

As it depends on clear logic of comparing File Hash values, before storing to the cloud and after retrieving from the cloud, so there is no way to make mistakes in detecting file integrity violation and it is highly efficient to detect integrity violation of shared files.

1.1 Statement of the Problem

With increasing trend in outsourcing data to remote cloud servers with the ability of file sharing, most of these free cloud servers pretend that, it may keep files saved securely. But anything saved on the internet can't be guaranteed at all.

So, such shared files are on security risk, and nothing can detect how file integrity violation is on free cloud servers, also file integrity could be threatened from internal users who share this file. So, detecting integrity violation of shared files emphasis users trust that their files in safe situation and there is no illegal modification on it.

1.2 Objectives

1.2.1 Main Objective

The main objective of this research is to build a new model that detects the integrity violation of a shared file among a group of users in the cloud. Which is based on Hashing method and support dynamic data operations (data append, update and delete).

1.2.2 Specific Objectives:

1. Use a way to orchestrate the operation on the file.
2. Map the change on the file on a log file.
3. Use a proper and efficient hash algorithm (SHA256) to hash the file before sending it to the cloud.
4. Upload some files to the cloud as a dataset.
5. Design a cloud website which represents our model and supports dynamic data operations on files.
6. Implement the model and conduct the experiments in our cloud.
7. Evaluate the performance and the efficiency of the model in term of accuracy and concurrency.

1.3 Importance of the Project

In the context of cloud computing, where users outsource their files to remote cloud servers which may be controlled by untrustworthy cloud storage providers, it's vital to ensure that files are kept properly in a consistent way and file integrity is tracked. Surely, this advantage will be more useful when files are shared between a group of users, especially, because Integrity Violation Detection will be performed without the need to save a copy of the data locally. And will help in decreasing organizations fear toward outsourcing their files to cloud.

On the other hand, it could be a good way of File Archiving, as we can retrieve them after a long period of time and detect their integrity violation easily.

1.4 Scope and Limitations:

- In this research, we work on files only not structured and tabular data.
- The underline cloud to be used is a private one. (We built it).
- The shared data type to ensure its integrity is File type of text-based (doc, txt...).
- The integrity is the only security objective considered here.
- We will suppose that there is one admin and every other user in the group has the same permissions on shared file.
- We can't make a comparative study as we didn't find a model to compare with, our study is analytical.

1.5 Methodology

In order to achieve our objectives, the prototyping methodology will be followed:

- **Stage (1), Data Acquisition**: Reviewing the recent related papers about integrity of files on the cloud and study the existing models in order to overcome on its disadvantages such as hashing just part of file content which less integrity violation detection efficiency, or disable dynamic operation on files.

Also, we search about appropriate algorithms that can be used to hash the file before storing in the cloud, we will use a popular hash method called SHA256, and we will explain why we chose it.

- **Stage (2), Preparing File**: Before storing a file in the cloud (F), we will use a hash method (SHA256) to hash the file content. Then, hash value will be stored on the intermediate system database with the file ID, time of storing and file creator.

- Stage (3), Define Permissions: we looked for how Operating systems (like Linux) deal with permissions on shared files to find a way to orchestrate the operation on the file.

In principle, each user will have the same permissions on file, but when a service (File) is downloaded by a user (to update it), no other user can update it till this user or the file creator activate the update property or upload an updated version of the file.

If none of this user or file creator activate the update property, the system will activate it automatically after a pre-specified period of time, in our prototype, the pre-specified period is 15 minutes.

To delete a file, any user can delete it, actually it is not deleted, it is hidden to users, and only admin can see it and confirm deletion or restore it to the system. So Permission system will organize access to shared files between users.

- Stage (4), Support Dynamic Data Operation: Users can perform dynamic data operations (append, update or delete to modify the data file) under the restriction of this file permissions. For example, if a user open a File for writing, no one can write on it at that time.

- Stage (5), Develop a Log File: To map the change on the file, intermediate system will contain a log file, so it saves every dynamic operation type (insert, delete or update), its actor identity and time of operation.

- Stage (6), Integrity Violation Detection: To check the correctness of the file data in the cloud, we will retrieve a stored file, from the cloud, then, we will calculate the hash value of the retrieved file and finally, compare it with old hash value that stored in the intermediate system. If the two hash values are matched, then, there is no problem. If not, so Notification alert system will sends an alert to the user to know that the file integrity of the file is violated.

- Stage (7),Evaluate concurrency, we will allow more than one user to access a shared file in the cloud, if one open it in write mode, no other user can update it. After updating the file (upload a new version, update the old one or reactivate the update property), other users can re-update it again. But if one user open it in read mode, simply, other users can read this file and so on.

For this step, we will explore more about permissions on some Operating system as we mentioned before.

- Stage (8), Evaluate accuracy, we will enable a group of users to share files many times and detect the integrity violation of these files at each time.

- Stage (9), Results and discussions, analyzing the obtained results and justify the model feasibility.

1.6 Thesis Structure

This thesis consists of six chapters: Introduction, Theory Background, Related Works, Data integrity model, Experiments and Evaluation and finally, Conclusions and Future work.

The main points discussed in the chapters are listed below:

- **Chapter 1:** ***Introduction***, presents a general view about cloud computing need and brief description about the proposed model.
- **Chapter 2:** ***Theoretical Background***, presents an overview about cloud computing: definition, essential characteristics, benefits, challenges and security issues focusing into integrity.
- **Chapter 3:** ***Related Works,*** presents most previous related works and explains points of differences and similarities.
- **Chapter 4:** ***Data Integrity Violation Model,*** presents the proposed model that detect the integrity violation of share file in the Cloud. And describes the intermediate system with its main functions.

- **Chapter 5:** ***Experiments and Evaluation:*** the experimental works, evaluating and analyzing the experimental results.
- **Chapter 6:** ***Conclusions and Future Work:*** presents conclusions and possible future work.

Chapter2

Theoretical Background

Chapter 2

Theoretical Background

2.1 Introduction

In this chapter, we present an overview of cloud computing, its definition, essential characteristics, deployments model and cloud architecture key components. We also present cloud computing benefits, challenges and security issues.

2.2 Cloud Computing Overview

Cloud Computing is a generic term for anything that involves delivering hosted services over the Internet. The name cloud computing was inspired by the cloud symbol that is often used to represent the Internet.

Cloud computing emerges as one of the hottest topic in field of information technology. It is based on several other computing research areas such as virtualization, utility computing and grid computing.

The U.S. National Institute of Standards and Technology (NIST, 2011) has been developed the following definition of cloud computing:

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

In this section, we present cloud computing characteristics, categories, deployment models, types, benefits and architecture key components.

2.2.1 Cloud Computing Characteristics

Cloud computing exhibits the following key characteristics (NIST, 2011:

1. **Resource pooling**: The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. Examples of resources include storage, processing, memory, and network bandwidth.

2. **Productivity** may be increased when multiple users can work on the same data simultaneously, rather than waiting for it to be saved and emailed. Also, there is no need to install application software upgrades to their own devices.

3. **Cost reductions** claimed by cloud providers. A public-cloud delivery model converts capital expenditure to operational expenditure.

4. **Device and location independence**: users can connect their data from anywhere with regardless to what device they use (PC, laptop, mobile….).

5. **Maintenance** of cloud computing applications is easier, because they do not need to be installed on each user's computer, only cloud service provider is responsible of it and can access through APIs that don't require application installation onto PCs which reduces time and effort.

6. **On-demand self-service**: that enables users to consume computing capabilities as and when required. So, computer services such as email, applications, network or server service can be provided without the need to contact with each service provider (Mahmood, 2011).

2.2.2 Categories of Cloud Computing

There are mainly four models of cloud computing: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS) (Tumotech, 2014). Figure (2.1) presents the main differences between them.

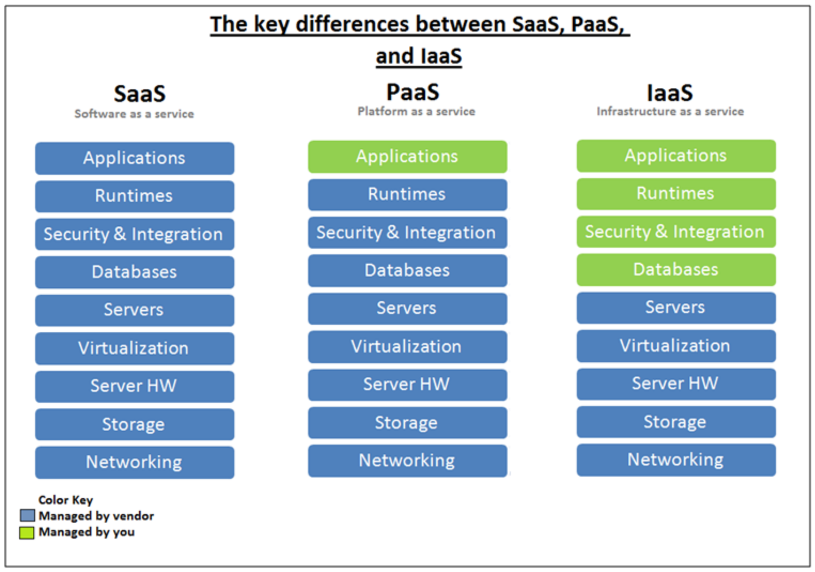


Figure (2.1): The difference between Saas,Paas and Iaas (Tumotech, 2014)

Software as a Service (SaaS):

- The cloud provides the user with access to already developed applications that are running in the cloud.
- The cloud users do not manage the infrastructure where the application resides.
- No need to install the application on the cloud user’s own computers.

Platform as a Service (PaaS):

- The cloud providers deliver to the user development environment services.
- The user can develop and run in-house built applications.

- The services might include an operating system, a programming language execution environment, databases and web servers.

Infrastructure as a Service (IaaS):

- This is the most basic cloud-service model, which provides the user with virtual infrastructure.

- User will manage everything from applications, to security & integration.

2.2.3 Cloud Deployment Models:

Deploying cloud computing can differ depending on requirements, each with specific characteristics that support the needs of services and users of the clouds in particular ways. (VictorVictories, 2015). Figure (2.2) presents Cloud computing deployment models.

1. **Private Cloud**

   Private cloud is cloud infrastructure operated only for a single organization, whether managed internally or by a third-party, and hosted either internally or externally.

2. **Public Cloud**

- Services are submitted over a network that is open for public use.

- Public cloud services may be free.

- Technically there may be little or no difference between public and private cloud architecture, however, security consideration may be substantially different for services.

- Public cloud service providers like Amazon AWS, Microsoft and Google own and operate the infrastructure at their data center and access is generally via the Internet.

3. **Hybrid Cloud**

- Is a composition of two or more clouds (private, community or public), that remain distinct entities but are bound together.
- Offering the benefits of multiple deployment models.

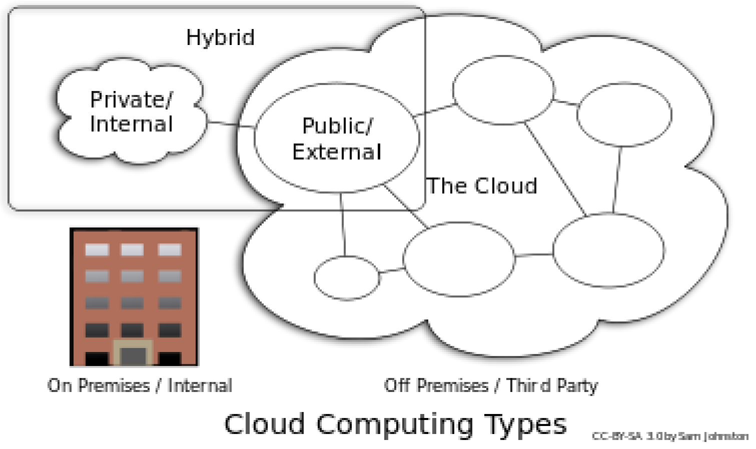


Figure (2.2): Cloud computing deployment models (Johnston, 2016)

2.2.4 Cloud Architecture Key Components

Cloud architecture includes three main components: (Barhoom and Saqallah, 2014)

- **User**: He/she may be the data owner who have a good large amount of data to be kept in the cloud or he/she may be a person need to access data in the cloud with proper access permissions.
- **Cloud Service Provider**: Is an Entity which includes significant storage space and computation recourse to keep up clients data.
- **Third party Auditor**: Has capabilities to manage outsourced data by the delegation relating to data owner.

2.2.5 Cloud Computing Benefits

Using cloud computing provide many facilities and benefits. So, it can help you focus more on your core business competency. The following are some of the cloud computing benefits (Tsagklis, 2013):

1. **Cost Efficiency:** which can be achieved by the elimination of the investment in stand-alone software or servers. Companies can reduce their capital expenditures and use operational expenditures for increasing their computing capabilities. So, they can save on licensing fees and at the same time eliminate overhead charges such as the cost of data storage, software updates, management etc.

2. **Continuous Availability:** Public clouds offer services that are available wherever the end user might be located. This approach enables easy access to information and accommodates the needs of users in different time zones and geographic locations. As a side benefit, collaboration rooms since it is now easier than ever to access, view and modify shared documents and files.

3. **Backup and Recovery:** The process of backing up and recovering data is simplified since those now reside on the cloud and not on a physical device. The various cloud providers offer reliable and flexible backup/recovery solutions. In some cases, the cloud itself is used solely as a backup repository of the data located in local computers.

4. **Scalability and Performance:** Cloud instances are deployed automatically only when needed and as a result, you pay only for the applications and data storage you need. Hand in hand, also comes elasticity, since clouds can be scaled to meet your changing IT system demands.

5. **Quick Deployment and Ease of Integration:** A cloud system can be up and running in a very short period, making quick deployment a key benefit. A user is

allowed to choose the services and applications that best suit his preferences, while there is minimum effort in customizing and integrating those applications.

6. **Device Diversity and Location Independence:** With the cloud, the “Bring your own device” (BYOD) policy can be easily adopted, a user might decide not only which device to use, but also where to access the service from. You can access your applications and data anywhere in the world at any time you want.

2.3 Type of Outsourcing Data

Outsourcing is the process of contracting with another company or person to do a particular work (Roseindia, 2016). Outsourcing of data is a useful feature through which resources are grouped to serve many users at the same time. Lack of security on data outsourcing is an important issue and becomes as a wide area in the Information science research field (Thangavell, 2014).

Data outsourcing relieves the responsibility of local data storage and maintenance, but introduces security fears. So, most of outsourced data is less important, which greatly limits the applicability of cloud computing.

Outsourced data could be one of several types such as Files, Tabular data, Structured data…etc. In this work the proposed model design to consider the shared file. The Files are the most common type used in data sharing and used by different types of users regardless to their technology knowledge.

2.4 Cloud Computing Security Issues

Despite the potential advantages achieved from the cloud computing, some organizations are slow in accepting it due to security issues and challenges associated with it.

This section, presents some cloud challenges and related security issues (Apostu et al.,2012) :

1. **Prone to Attack:**
   Nothing on the Internet is 100% secure. So, storing data in the cloud could make your company vulnerable to external hack attacks and threats.
2. **Dependency and vendor lock-in:**
   Implicitly you are dependent on the provider. This is what the industry calls “vendor lock-in” since it is sometimes difficult to migrate from a provider once you have rolled with him, you should carefully and thoroughly contemplate all options when picking a vendor.
3. **Possible Downtime:** The whole setup is dependent on internet access, thus any network or connectivity problems may late your business.
4. **Security and Privacy**: Perhaps two of the more “hot button” issues surrounding cloud computing relate to storing and securing data, and monitoring the use of the cloud by the service providers. You should know that you will be surrendering all your company’s sensitive information to a third-party cloud service provider. So, you need to make absolutely sure that you choose the most reliable service provider, who will keep your information totally secure.

Cloud computing security is the set of control-based technologies and policies designed to adhere to regulatory compliance rules and protect information, data applications and infrastructure associated with cloud computing use (Rouse, 2014).

Security in cloud computing refers to a broad set of policies, technologies, and controls deployed to protect data, applications, and the associated infrastructure of cloud computing (Chenxi Wang, 2009). Security contains many policies and rules which should be applied to protect data such as, confidentiality, availability, integrity…etc. as shown in Figure(2.3).

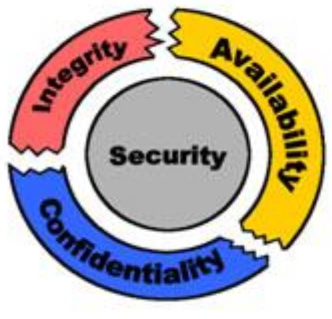


Figure (2.3): The most important security rules (Carabotx, 2009)

Data Integrity refers to protecting data from unauthorized deletion, modification or fabrication (Lekkas, 2012). In cloud systems, Data integrity is defined as the consistency and accuracy of stored data in absence of any alteration to the data between two updates of a file or record.

Although, there is increasing trend in outsourcing data to remote cloud servers, there is lack of offering a strong evidence of data integrity.

In this work, the model is to detect the integrity violation of shared file on the cloud. As the integrity of the file content is a very important to have a right information and as consequence has a right decision. And, as it is a very big issue to convince the data owner that his files are stored correctly and stored consistently without the existence of local copies in their devices. If the data stored in the system is manipulated in any way in an unauthorized way then all the work put in that system is wasted.

File Sharing:

As we work on shared files, there is a need to get each user an account and permissions or privileges on these files, this is similar to permission on some operating systems like LINUX, that support access control lists which allow assigning read, write and execute permissions for more than one user and more than one group for each file in the system.

Linux permissions allow a file or directory owner to restrict access based on the user's relationship to each file. This allows for control schemes that provide varying levels of access to different people.

Each permissions category (owner, group owner, and other) can be assigned permissions that allow or restrict their ability to read, write, or execute a file. For a regular file, read permissions are required to read the contents of a file, write permissions are necessary to modify it, and execute permissions are needed to run the file as a script or an application (Ellingwood, 2013).

- **Dynamic data operations** like insertion deletion and updating on shared files on the cloud.
- **Dynamic concurrency:** "In database systems, it refers to the ability of a database to allow multiple users to affect multiple transactions. This is one of the main properties that separates a database from other forms of data storage like spreadsheets. " (Techopedia, 2016) Concurrency refers to the assurance that users can access data at the same time (Ashdown and Kyte, 2016) .

In this work, the meaning of dynamic concurrency is to enable users to share a file with some dynamic operation on it. In the other hand taking in account permissions of the users on the share file without conflict.

In cloud environment, security plays an important role because users can save their data on remote cloud servers, which are controlled and managed by untrustworthy cloud providers.

So, when a group of related users store their shared files on the cloud servers, they are should be able to return to it and find it in a consistent state (as it is previously stored). Otherwise, they will be in a big security problem which refers to integrity issues.

Most of prior researches deal with data integrity issue in the cloud from the side of just one user, but as we know, nowadays business needs work to be accomplished as soon as possible so sharing files between work team (or users) will have a significant effect on finishing tasks soon .

Also File Sharing enables a number of users to use the same file wherever they are, whenever they want and they can share this file between each other easily.

In terms of security, we can see that online file sharing is a great option for protecting these files. Since files can only be accessed with a secure login. So, only authorized users can access them. Also, any viruses that affect your hard drive do not affect your stored files because the files are stored in the cloud which is independent of hardware failures, loss, spyware, or other mishaps that can happen to business users on a daily basis (Mount, 2014).

Many of cloud service provider pretend that user's data is saved securely with no data integrity violation. But as we know, anything saved in the internet environment can't be perfectly safe. So, such shared files are on security risks, and nothing can guarantee file integrity on free cloud servers, also file integrity could be threatened from internal users who share this file.

2.5 Hashing process:

Hashing is the process of transformation of a string of characters into a fixed-length value that represents the original string. It could be used in many encryption algorithms (Rouse,2005).

So, it is a method of creating unpredictable, irreversible output from a string input. There are many different hashing algorithms such as MD2, MD5, SHA…etc.

In this work, SHA256 algorithm will be used, version of SHA (standing for secure hash algorithm) family, as it is a strong hash algorithm, resistant to collision attacks and doesn’t broken by any attack.

Also, we will work on the problem of integrity of shared data over the cloud. Integrity violation detection will include two areas: fears from outside the cloud (as internet is not totally safe) and fears from not-authorized use from internal users.

2.6 Summary

We presented in this chapter a cloud computing overview, its definition, essential characteristics, deployments model, benefits and Cloud architecture key components. In addition, we presented cloud computing benefits, challenges and security issues, also we represent some type of outsourcing data.

We discussed the security problems in the cloud and declared that data security is becoming a fundamental issue in Cloud Computing. Data integrity is critical for most of users, and it is important to ensure that their data is stored securely.

Chapter 3

Related Works

Chapter 3

Related Works

3.1 Introduction

In cloud environment, large numbers of threats are raised. Data integrity is one of the main threats. A lot of researchers focused on providing data correctness within the cloud and introduce many answers to decrease the threat related to data integrity among the cloud.

Chalse, Selokar and Katara (Chalse, Selokar and Katara, 2013) presented a detailed analysis of the cloud security problem and they believe that data storage security in Cloud Computing is an area full of challenges as we mentioned before, so, they made a study to analyse the security requirement and highlight the existing threats in cloud computing to help the researchers to identify security requirements and their design was mainly based on the usage of Public and Private key encryption system.

Here, we will present related work according to:

1. Used techniques in verifying File integrity.
2. Enabling Third-Party Verifier (TPV):
3. Enabling File sharing.
4. Supporting Dynamic operations.

3.2 Used Techniques in Verifying File Integrity

Kavuri et al. (Kavuri et al., 2014) proposed an improved hash based message integrity verification process to ensure the confidentiality of the user’s information in the remote storage, users can access the required files using their identity along with the message integrity value.

Most of the existing hash methods integrate hash value to the original data. But Kavuri et al. (Kavuri et al., 2014) method is to store both the encrypted data and its hash value separately in the cloud storage.

Teli and Nida (Teli and Nida, 2015) proposed a novel heterogeneous online and offline signcrypt model for a cloud network for the issue pertaining to identities and trust management, firstly, it set ups the secure, trustworthy connection between the cloud user and cloud data center, secondly, it allows a cloud user in an Identity based cryptography (IBC) to send a request message to an internet host in public key infrastructure (PKI) to provide trustworthiness, authentication and data non-repudiation.

In Traditional Model for identity management user will access the cloud service by submitting his request to the cloud broker, which acts as an interface between CSP and user .So, Teli and Nida (Teli and Nida, 2015) added a trusted authority (TA) also known as certificate authority (CA) in PKI. So, users who want to access Data centres must register to TA and get a certificate access token then a request message is sent to service broker.

Kumar and Saxena (Kumar and Saxena, 2011) proposed a scheme which gives a proof of data integrity in the cloud by encrypting only few bits of data per data block not the whole data ,so reduce computational overhead on the clients also the client storage overhead is minimized as it does not store any data with it. In this data integrity protocol the verifier needs to store a single cryptographic key regardless of the size of the data file F. The verifier before storing the file at the archive, appends some Meta data to the file and stores at the archive. The verifier uses this Meta data at the time of verification to verify the integrity of the data. But this scheme cannot be applied when the data need to be dynamically changed also it just checks the integrity of data but it does not prevent the archive from modifying the data.

3.3 Enabling Third-Party Verifier (TPV)

Some Researchers enable a third-party verifier to check data integrity, Lianhong and Hua

(Lianhong and Hua, 2010) presented a secure and efficient scheme, which enables not only the data owner but also a third-party verifier to check data integrity. The proposed scheme is based on RSA assumption. But this scheme cannot realize the dynamics data for remote data integrity check. Also, Shah et al. (Shah et al., 2007) proposed a solution that contain a third party auditing as it allows customers to evaluate risks, and increases the efficiency of insurance based risk mitigation by encrypting the data then sending a number of pre-computed symmetric-keyed hashes over the encrypted data to the auditor. Their solution has three phases: initialization, verification, and extraction. In initialization, the storage service commits to storing a document on behalf of the customer, and the auditor initializes long-term state. During verification, the auditor repeatedly checks the stored contents. Finally, in extraction, the auditor assists in returning the data to the customer in case of doubt or dispute.

Kumar Subramanian (Kumar Subramanian, 2011) proposed an efficient and secure protocol for ensuring data storage security in cloud computing, their method allows third party auditor to periodically verify the data integrity stored at CSP without retrieving original data. It generates probabilistic proofs of integrity by challenging random sets of blocks from the server, which drastically reduces the communication and I/O, this solution removes the burden of verification from the user, minimizes both the user’s and storage service’s fear about data leakage and data corruptions.

Their proposed method is mainly suitable for thin users who have less resources and limited computing capability but for a big business organization it will be less efficient.

In the other hand, Al-Saiyd and Sail (Al-Saiyd and Sail, 2013) proposed a cloud computing security development lifecycle model to achieve safety and enable the user to take advantage of this technology and face the risks that may be exposed to data, they proposed a data integrity checking algorithm to eliminates the third party auditing .The cloud user has to deal with the cloud provider through managing the data integrity checking and evaluation in efficient way using set of hash functions. The data and its corresponding hash value are retrieved back when the cloud user needs cloud data, and checked for any data altercation by re-generating and comparing the hash result with the pre-generated hash value.

Some previous works didn't check data integrity only, but it also localized where the error is using different technique such as Erasure coding which creates a mathematical function to describe a set of numbers so they can be checked for accuracy and recovered if one is lost (Rouse, 2014).

Wang et el. (Wang et el., 2009) proposed a scheme with distributed verification of erasure-coded data, this scheme achieves the integration of storage correctness insurance and data error localization also it supports dynamic operations on data blocks, including: data update, delete and append.

Gunjal and Jeny (Gunjal and Jeny, 2013) relied on erasure-correcting code in the file distribution preparation to provide the redundancies and guarantee data dependability, their scheme achieves the integration of data error localization and storage correctness insurance.

And other researchers, used “proof of retrievability" models for error localization, Juels and Kaliski (Juels and Kaliski, 2007) introduce (POR) model in which the verifier stores only a single cryptographic key enables a user (verifier) to determine that an archive (prover) “possesses” a file or data object F ,POR protocol encrypts F and randomly embeds a set of randomly-valued check blocks called sentinels. F is then encrypted to protect the positions of these special blocks.

This scheme involves the encryption of the file F using a secret key it becomes computationally cumbersome especially when the data to be encrypted is large. Also it does not support dynamic data operations.

3.4 Enabling File Sharing

Few of researchers deal with integrity problem taking in consideration data sharing, Marimuthu et al. (Marimuthu et al., 2014) proposed a secure multi-owner data sharing scheme, for dynamic group in the cloud. They found that One-Time Password is one of the easiest and most popular forms of authentication that can be used for securing access to accounts and they integrated Image based authentication with it to achieve high level of security in authenticating the use. So, they provided groups with signatures and dynamic broadcast encryption techniques for secure sharing between users.

Also, Aarthi and Indira (Aarthi and Indira, 2016) proposed a methodology that provides data sharing using TLS and SSH. If user want to share data, encrypted data is read from cloud database, then perform under re-encryption by shared user encryption key, because a unique encryption key is maintained for each user. Then that re-encrypted data Stored in cloud database server.

But when a user want to access data, he/she should sends a download request to the third part (that is responsible for key management, encryption, decryption, and access control) which could be one of threats on data.

3.5 Supporting Dynamic Operations

Few of works consider dynamic data files and support dynamic data operation on it, Barhoom and Saqallah, (Barhoom and Saqallah, 2014) proposed a model with dynamic data operations support including data append, update and delete, to ensure the integrity of user's data in the cloud based on verification tokens and erasure-coded data without the need to keep the data locally at the client side. This model reduces the computational overhead of the client but it doesn't ensure the integrity of a shared data by a group of users.

Khatri and Jethava (Khatri and Jethava, 2013) proposed a scheme that provides data integrity checking frequently and supports dynamic operation using RSA signature Merkle Hash Tree construction. But also, it doesn’t detect integrity of shared data and another shortcoming is that used SHA-1 hash algorithm which is broken. In this research, we will consider dynamic operations and file sharing.

Finally, after an advance step of working, we found GitHub, which is a web-based version-control and collaboration platform for software developers. Was created by Linus Torvalds in 2008 as a software-as-a-service (SaaS) business model software building faster (Rouse, 2016).

GitHub can actually be used for any types of files (including word documents) so if you have a team that is constantly making changes to a document, you can actually use GitHub as your version control system which support sharing data with dynamic operations on it. Also it keeps track of all the changes that have been pushed to the repository (as a log file).While GitHub is a software service, our work is specialized for files.

Table (3.1): Summarization of most related previous works

No.	Authors	Dynamic operations	Shared File	Technique
1.	(Kavuri et al., 2014)	No	No	Hashing - Encrypted data and its hash value separately in the cloud
2.	(Marimuthu et al., 2014)	No	Yes	One-Time Password - Image based authentication
3.	(Barhoom and Saqallah, 2014)	Yes	No	Verification tokens and erasure-coded data
4.	(Al-Saiyd and Sail, 2013)	No	No	Hashing - eliminates the third party auditing
5.	(Aarthi and Indira, 2016)	-	Yes	TLS and SSH techniques, key share is stored by third party
6.	(Khatri and Jethava, 2013)	Yes	No	SHA-1 hash algorithm which is broken
7.	GitHub.	Yes	Yes	software service, Trust issue

As shown in Table (3.1), some related works like Kavuri et al. (Kavuri et al., 2014) and Al-Saiyd and Sail (Al-Saiyd and Sail, 2013) used hashing to detect integrity in the cloud and store hash value separately, not inside the stored file. But they didn’t support Dynamic Operations or File Sharing.

Others, like Marimuthu et al. (Marimuthu et al., 2014) and Aarthi and Indira (Aarthi and Indira, 2016) used different types of hashing and encryption and supported File Sharing but didn’t support on these files.

Although Khatri and Jethava (Khatri and Jethava, 2013) and Barhoom and Saqallah (Barhoom and Saqallah, 2014) supported Dynamic Operations on files stored in the clouds of their models, but they didn’t apply integrity detection on shared files.

On the other hand, we can notice that Github has supported both Dynamic Operations and File Sharing. But it is a public cloud and we should take in consideration, trust issues especially when file contents are sensitive and can’t assume any error.

Chapter 4

Shared Files Integrity Violation Detection Model

Chapter 4

Shared Files Integrity Violation Detection Model

4.1 Introduction

In this chapter, Shared Files Integrity Violation Detection Model is presented with its architecture and flow chart descriptions, to achieve the specific objectives of our research, then in the second section methodology is explained with some examples of how this model will work, finally in the third section, model implementation is described in details.

4.2 The Proposed Model

As known, Sharing Files over the internet may causes many security fears as anything saved in the internet environment can't be perfectly safe. In addition, file integrity could be threatened from internal users who share this file. So, whatever cloud service providers pretend that user's data is saved securely ,there is fear of happening file integrity violation at any moment.

In this work, Integrity detection violation includes two areas: fears from outside the cloud (as internet is not totally safe) and fears from not-authorized use from internal users.

Our model contains an intermediate system to detect file integrity violation by hashing the file before saving it on the cloud and after retrieving it, then compares the two hash values. The proposed model supports dynamic data operation on shared files, including append , update and delete. Also, we focus on how to keep dynamic operations working in concurrency way, so no user can coflict the work of another user on a shared file.

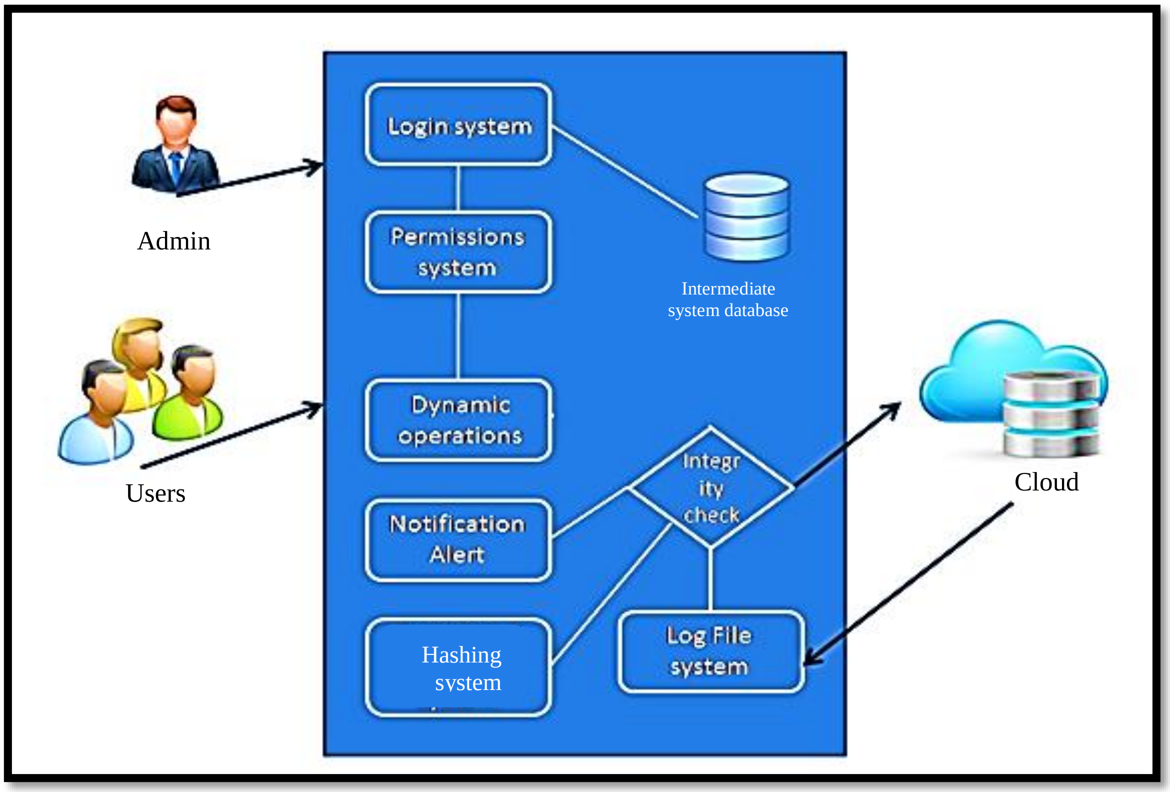


Figure (4.1): The Proposed Model Architecture

Figure (4.1) presents our proposed model which include the following:

- **Users**: One admin and other users, each one has his own account in the system.
- **Login system:** each user log in with his own account, (all users in the group except admin are with same privileges).
- **Permission system:** in principle each user will have the same permissions on file, but each resource (File) has a specific permissions on it.
- **A log file** in the intermediate system stores the actor of each operation on shared data in the cloud.

- **Integrity Check**: verifies the correctness of the data in the cloud, including Hashing process to calculate hash value of the file.
- **Notification Alert:** sends alert to the user if data integrity of the file is violated.
- **Dynamic Data Operation Support:** where a user can perform dynamic data operations of append, update or delete to modify the data of the file.

The presented flow chart in Figure (4.2), illustrates how the intermediate system works to detect integrity violation of shared files on the cloud.

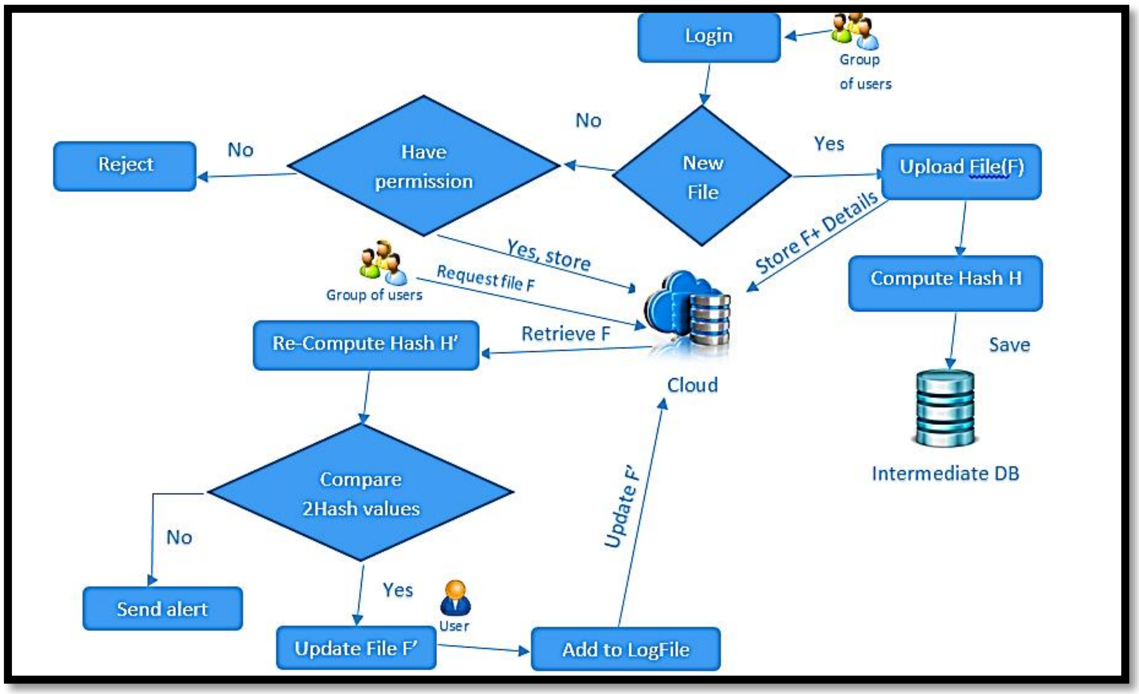


Figure (4.2): Intermediate system flow chart

4.3 Methodology

In order to achieve the functions of the model, these steps of prototyping methodology were followed:

Step (1), Data Acquisition: we have reviewed the most recent related papers about integrity of files on the cloud and study the existing models in order to overcome on its disadvantages such as hashing just part of file content which less integrity violation detection efficiency, or disable dynamic operation on files.

Actually, this stage helped us to find some contributions that can be represented in this research like enlarging integrity violation detection of one file to include group of files (shared) and enable applying dynamic operations on these files.

Also, we search about appropriate algorithms that can be used to hash the file before storing in the cloud, A popular hash method called SHA256 has been used, as it is a strong hash algorithm and doesn't broken by any attack.

At the beginning MD5 was used, which stands for **Message Digest** algorithm, its main idea is to take up a random data (text or binary) as an input and generate a fixed size "hash value" as the output. The input data can be of any size or length, but the output "hash value" size is always fixed. Figure (4.3) presents some examples of MD5 hash algorithm.

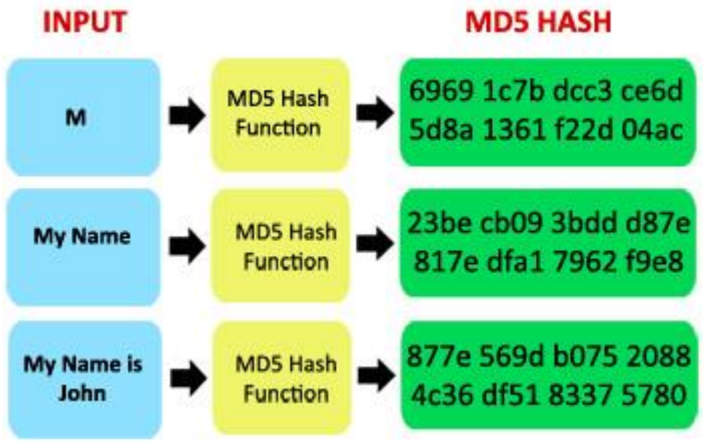

INPUT
MD5 HASH
M
MD5 Hash Function
6969 1c7b dcc3 ce6d 5d8a 1361 f22d 04ac
My Name
MD5 Hash Function
23be cb09 3bdd d87e 817e dfa1 7962 f9e8
My Name is John
MD5 Hash Function
877e 569d b075 2088 4c36 df51 8337 5780

Figure (4.3): Examples of MD5 hash algorithm (Ramesh, 2016)

But after some searching on this algorithm, it is clear that it was broken and not collisions resistance so The CMU Software Engineering Institute considers MD5 essentially "cryptographically broken and unsuitable for further use" (Chad R. Dougherty, 2008).

So we go to the most common hash algorithm family SHA (standing for secure hash algorithm), there are several versions of SHA: SHA0 (obsolete because vulnerable), SHA1, SHA2 and finally SHA3 introduced in 2012 (ibnu, 2016). Table (4.1) presents the differences of SHA's algorithms.

Table (4.1): Differences of SHA's algorithms (ibnu, 2016)

Algorithm	**Output size (bits)**	**Internal state size (bits)**	**Block size (bits)**	**Max message size (bits)**	**Word size (bits)**	**Rounds**	**Operations**	**Collision**
SHA-0	160	160	512	2^{64}-1	32	80	+,and,or,xor,rotl	Yes
SHA-1	160	160	512	2^{64}-1	32	80	+,and,or,xor,rotl	2^63 attack
SHA-256/224	256/224	256	512	2^{64}-1	32	64	+,and,or,xor,shr,rotr	None yet
SHA-512	512	512	1024	2^{218}-1	64	80	+,and,or,xor,shr,rotr	None yet

So from table (4.1), it is clear that SHA-256 rounds are shorter than SHA-512. So it is faster speed than SHA 512, and the result of some observations shows that none of the currently known attack methods can be successfully applied to SHA-256. Table (4.2) represents some examples hash values using SHA-256.

As a conclusion, SHA 256 has a good security, and it will be a good choice to use in our model as a hash algorithm.

Table (4.2): SHA-256 Examples

Text	**Hash value in SHA-256**
My	8ed6791bdf3d61a1e6edcbb253979b0a6bef7f3d99dda0fb49cffe96923514b6
My name is	deb83494817def510d32bbe2498b88af969d3e4325b65102eab7daab94399f5f
My name is Safaa	4fa916fc5c37fcd0c85bb8b45675475ef21b64d09be9dda828287404fdcce2e2

Step (2), Preparing File: Before storing the file in the cloud (F), a hash method (SHA256 as we mentioned before) will be used to hash the file content. Then, hash value will be stored on the intermediate system database with the file ID, time of storing and file creator.

SHA256 Hash method is applied to the whole file as we found that hashing different file sizes doesn't exceed one second, about 0.8 second, and in this way any integrity change will be detected more accurately. Figure (4.4) shows the Preparing File part of the model.

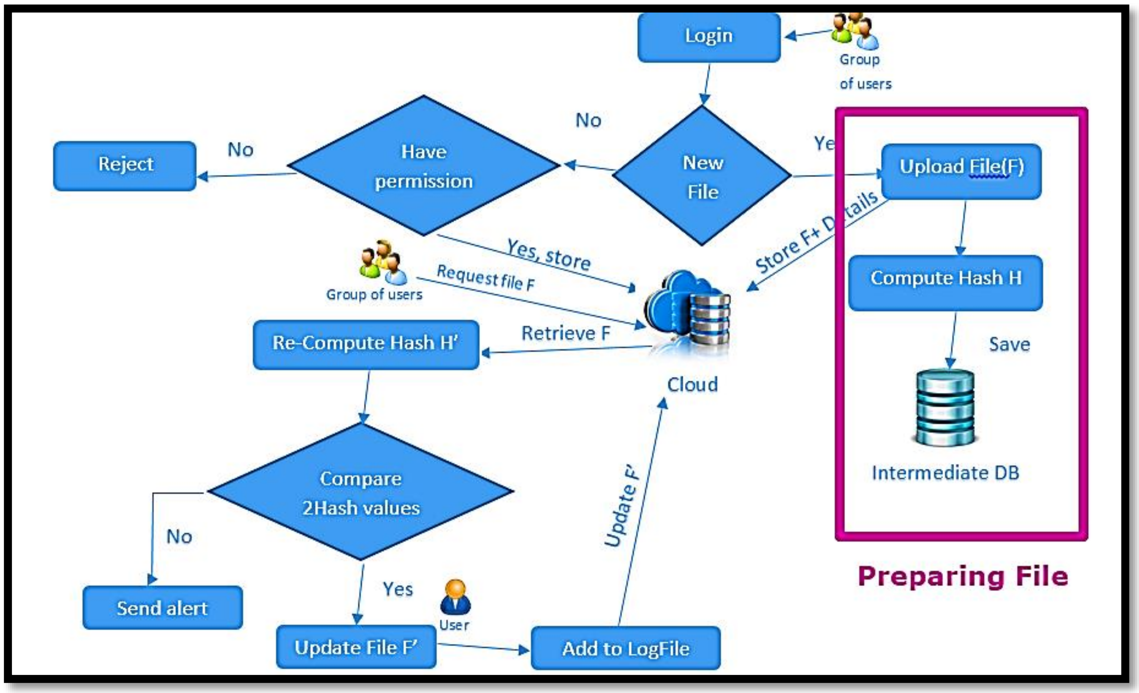


Figure (4.4): Preparing File Step

Step (3), Define Permissions: In principle, each user will have the same permissions on file, but when a service (File) is downloaded by a user (to update it), no other user can update it till this user or the file creator re-activate the update property or upload an updated version of the file.

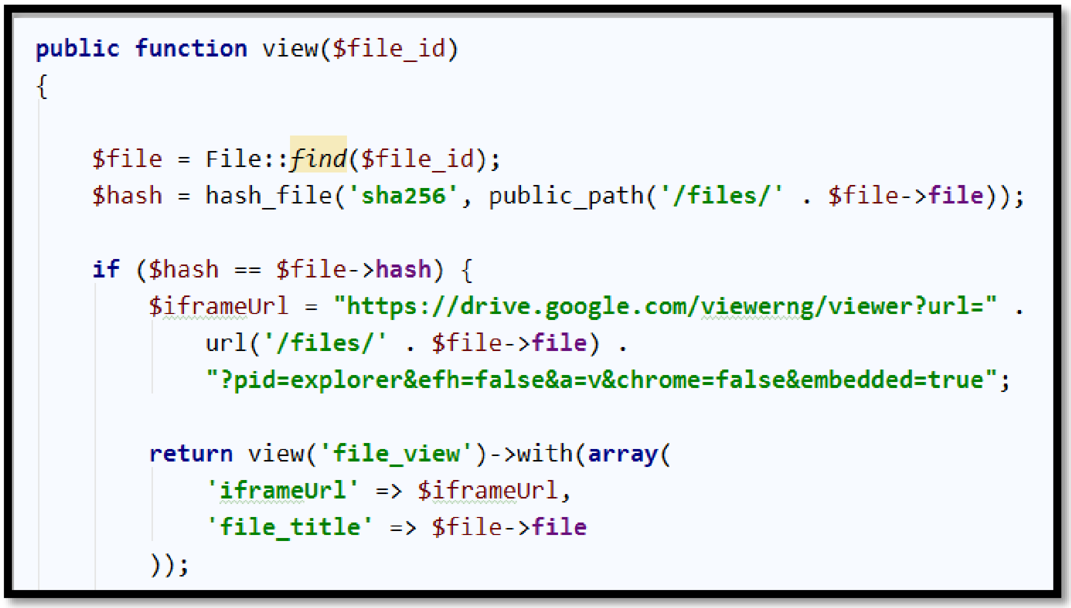
If none of them (user or file creator) activated the update property, the system will activate it automatically after a pre-specified period of time, in our prototype, the pre-specified period is 15 minutes. In future prototype, Admin can specify the period as he wants and when he wants.

In case of more than one user (n users) want to access the same file at the same time, we have two mechanisms:

1. **One of the n users is the file creator**, so when the n users try to update the same file at the same time, only the file creator can download the file to update it and

then update ability will be hidden for the other users. After updating the file and re-uploading it, users can see it, read or update it again.

2. **In case of none of the n users is the file creator** (all have the same priority). We use a way to choose the fastest one, which is Google Drive way, which enables the fastest one to access the wanted file. Figure (4.5) presents a piece of code that uses Google Drive code to enable just one user to access the file.



```php
public function view($file_id)
{

    $file = File::find($file_id);
    $hash = hash_file('sha256', public_path('/files/' . $file->file));

    if ($hash == $file->hash) {
        $iframeUrl = "https://drive.google.com/viewerng/viewer?url=" .
            url('/files/' . $file->file) .
            "?pid=explorer&efh=false&a=v&chrome=false&embedded=true";

        return view('file_view')->with(array(
            'iframeUrl' => $iframeUrl,
            'file_title' => $file->file
        ));
```

Figure (4.5): Google Drive code

- **To delete a file**, any user can delete it, actually it is not deleted, it is hidden to users, and only admin can see it and confirm deletion or restore it to the system.

As we work on shared files, there is a need to with permissions or privileges on these files, this is similar to permission on some operating systems like LINUX, that support access control lists which allow assigning read, write and execute permissions for more than one user and more than one group for each file in the system. Each permissions category (owner, group owner, and other) can be assigned permissions that allow or restrict their ability to read, write, or execute a file. For a regular file, read permissions are required to read the contents of a file, write permissions are necessary to modify it, and execute permissions are needed to run the file as a script or an application (Ellingwood, 2013). Table (4.3) represents the basic permissions on files in this work.

Table (4.3): File Permissions

File	Mode	New user	
		Read	Write
F1	Read	Yes	No
F2	Write	No	No
F3	Violated	No	No

- File is in **READ MODE** (one user opened the File to read it): any other user can read it easily, but no one can update it till all users close it.
- File is in **WRITE MODE** (one user downloaded the File to update it): no other user can read nor update it, till this user finishes updating it.
- File is **VIOLATED**: no user can read or update it except admin, who can delete it from the system or upload another version of it.

So Permission module in the model will organize access to shared files between the users.

Step (4), Support Dynamic Data Operation: Users can perform dynamic data operations (append, update or delete to modify the data file) under the restriction of this file permissions. For example, if a user open a File for writing (updating), no one can write on it at that time. As explained in Step (3).

Step (5), Develop a Log File: To map the change on the file, intermediate system will contain a log file, so it saves every dynamic operation type (insert, delete or update), its actor identity and time of operation so if any file modification operation will be stored, so every user will be responsible of the operation he/she made on this file. Table (4.4) represents a sample of Log File content.

Table (4.4): Sample of Log File Content

The action	File name	User	Time
Upload	Diagram.doc	Ahmed Lulu	2016-10-02 04:32:49
Download	Test.docx	Yazan Mahani	2016-10-02 23:20:55
Delete	Sample.doc	Alma Wadi	2016-10-03 21:312:22

Step (6), Integrity Violation Detection: To check the correctness of the file data in the cloud, we will retrieve a stored file, from the cloud, then, the hash value of the retrieved file will be re-calculated and finally, compare it with shared hash value that stored in the intermediate system. If the two hash values are matched, then, there is no problem. If not, so Notification alert system will sends an alert to the user to know that the file integrity of the file was violated. Figure (4.6) shows how the Integrity Violation Detection step of the model will be.

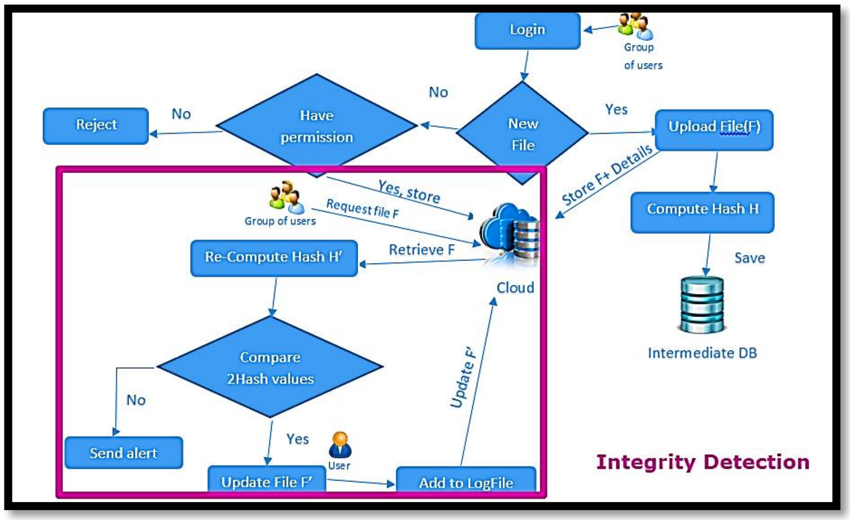


Figure (4.6): Integrity Violation Detection process

Step (7),Evaluate concurrency, the model allows more than one user (about three users because more than two is considered as a group) to access a shared file in the cloud, if one open it in write mode, no other user can update it at the same time. The updated file (upload a new version, update the old one or reactivate the update property with no change), may be updated by other users again. But if one user open it in read mode, simply, other users can read this file and so on.

Step (8), Evaluate accuracy, by enabling a group of users to share file many times consecutively and detect the integrity violation of these files at each time. So, many updates could be done on one file and before each update integrity will be detected to ensure accuracy of the file.

Step (9), Results and discussions, analyzing the obtained results of each experiment on the model and justify the model feasibility and how degree it matches its specific objectives this will help us to make a future vision of this work to improve it in the next prototype.

4.4 Implementation

In this section the implementation of our model is discussed and the intermediate system which we built is described, with introducing used programing language and main parts of the model.

As a programming language to be used there were some choices like PHP, Python …etc., in this work, PHP is used as a programming language because it is free and widely-used and this model is just a prototype so we can develop it in the future using another programming language.

PHP is a server scripting language, and a powerful tool for making dynamic and interactive Web pages, and it is a widely-used open source general-purpose language and can be embedded into HTML. So, it is used in this work (Cowburn, 1997).

Figure (4.7) represents the main modules of the intermediate system which consists of (upload file, view file, update file, delete file, organize file operations sequence and store Log File).

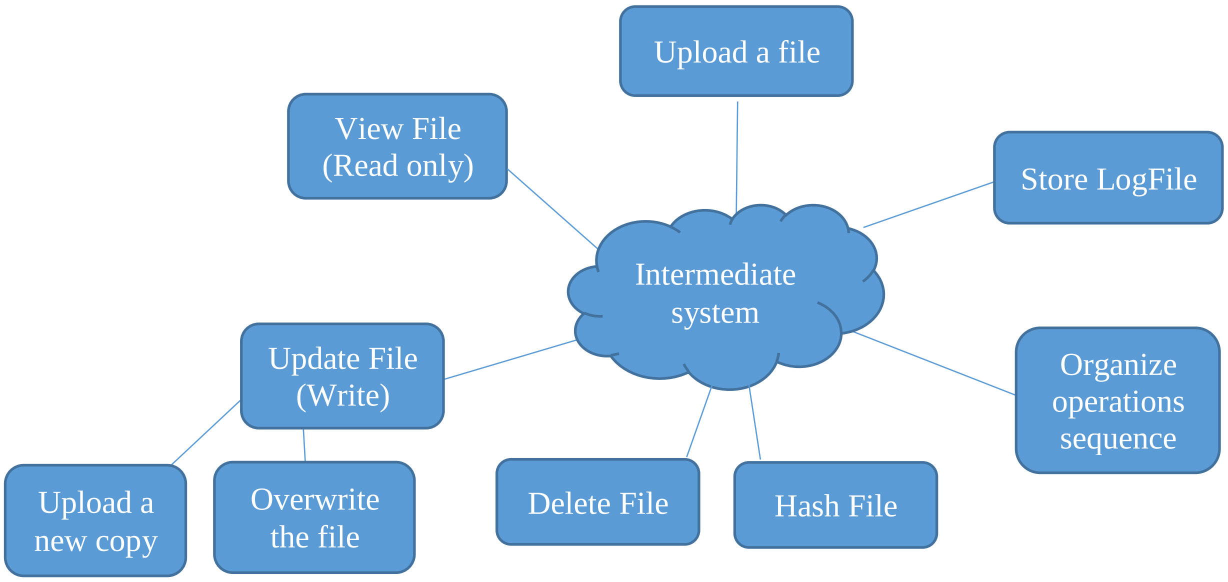


Figure (4.7): Intermediate System Modules

The intermediate system contains these functionalities:

1. **Upload File:**

   User can choose a file (just with .doc extension) from his device and upload it simply to the intermediate system.

2. **View file:**

   Any user in the system can view any shared time at any time he/she wants except when the file is in Write mode.

3. **Update file:**

   If File is in Write mode, a user has downloaded it for updating, then he/she has three choices to do:

a. Overwrite the file: updated file will be stored in the system and old file will be erased. So, just one version of this file will be cloud (with the last update).

b. Upload a new version: old file will remain and a new version of the file will exist also but with a modified name.

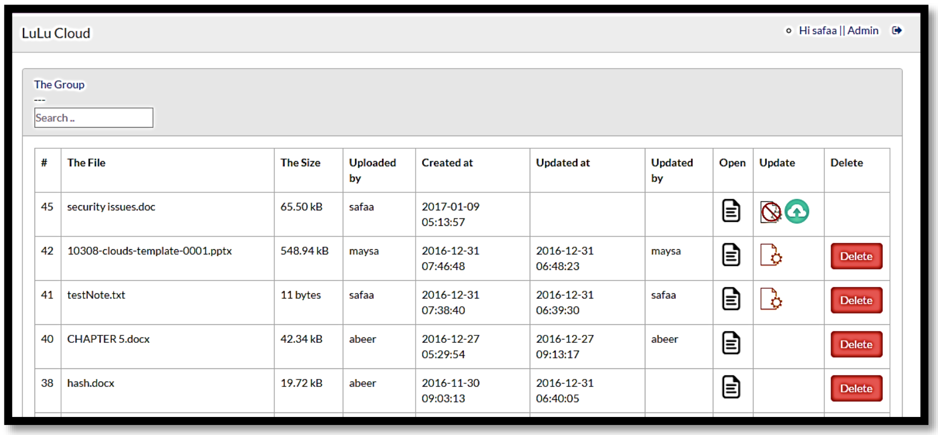
c. No changes: if user feels that he/she doesn't want to change anything of file contents, he/she can easily re-active update function for this file without the need to re-upload it again.

Figure (4.8) represents three cases of updating a shared file in the system, for example:

- **File with ID 38**: it has been uploaded by (Abeer) and now it is downloaded by a user (we don't know who till now) so update function is unavailable at this moment.

- **File with ID 41:** it has been updated by (Safaa) and no one access it now. So, anyone can update it (update function is available).

- **File with ID 45:** it has been downloaded by (Safaa) Now. No one else can update it at this moment, when (Safaa) finished her work, she has two choices to do:


LuLu Cloud

Hi safaa || Admin

The Group

Search ..

#	The File	The Size	Uploaded by	Created at	Updated at	Updated by	Open	Update	Delete
45	security issues.doc	65.50 kB	safaa	2017-01-09 05:13:57					
42	10308-clouds-template-0001.pptx	548.94 kB	maysa	2016-12-31 07:46:48	2016-12-31 06:48:23	maysa			Delete
41	testNote.txt	11 bytes	safaa	2016-12-31 07:38:40	2016-12-31 06:39:30	safaa			Delete
40	CHAPTER 5.docx	42.34 kB	abeer	2016-12-27 05:29:54	2016-12-27 09:13:17	abeer			Delete
38	hash.docx	19.72 kB	abeer	2016-11-30 09:03:13	2016-12-31 06:40:05				Delete

Figure (4.8): Updating File Cases

4. **Delete file:**

   When file is no longer needed any user can delete it, actually, it is just hidden and only Admin can see it and he/she can do two actions:

   a. Confirm deletion: no one can retrieve the file anymore.

   b. Retrieve it, and will be seen by other users.

5. **Organize file operations sequence:**

   a. **Read Mode**: File can be read by a group of users at the same time.

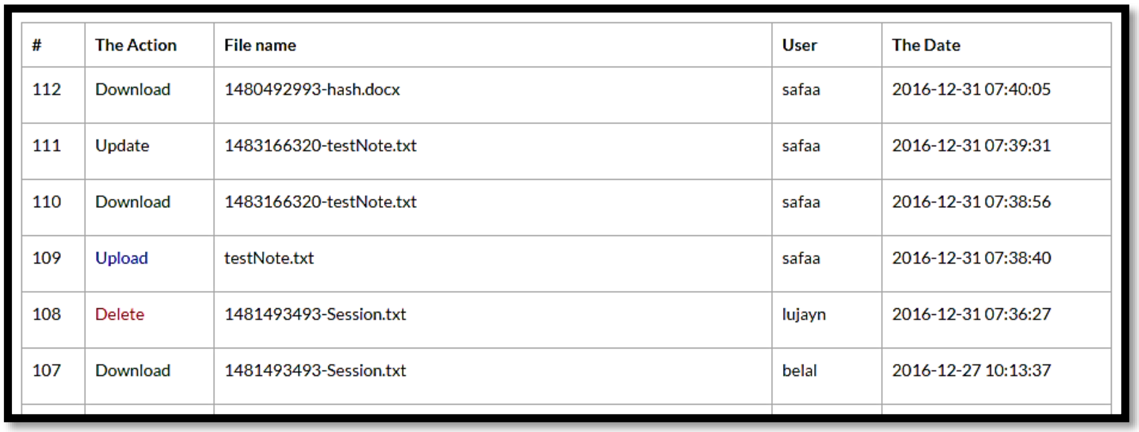
   b. **Write Mode**: Intermediate system enables one and only one user to update File at the same time and apply some steps to ensure this point: When a user downloads a file to update it, UPDATE function is disabled for others. But when more than one user try to download it at the same time:

      ▪ If one of them is the file creator, he/she has the priority to download the file to update it and then update ability will be hidden for the other users.

      ▪ If none of them is the file creator (all have the same priority). The intermediate system will choose the fastest one using Google Drive way, which enables the fastest one to access the requested file. Figure (4.6) presents a piece of code that uses Google Drive code to enable just one user to access the file.

6. **Store Log File:**

   Each operation applied to Files (Upload, Download, Delete, Update) are saved in a log file which can be seen by the Admin to control the cloud and to detect any integrity violation, also, it helps Admin when a file is deleted so admin can confirm deletion or restore the file again. Figure (4.10) represents a sample of Log File.



#	The Action	File name	User	The Date
112	Download	1480492993-hash.docx	safaa	2016-12-31 07:40:05
111	Update	1483166320-testNote.txt	safaa	2016-12-31 07:39:31
110	Download	1483166320-testNote.txt	safaa	2016-12-31 07:38:56
109	Upload	testNote.txt	safaa	2016-12-31 07:38:40
108	Delete	1481493493-Session.txt	lujayn	2016-12-31 07:36:27
107	Download	1481493493-Session.txt	belal	2016-12-27 10:13:37

Figure (4.9): Log File

4.5 Summary

In this chapter, we presented and built our model to detect the Integrity Violation of shared files in the Cloud with its architecture and flow chart descriptions. Then methodology is explained with some examples of how the model works, in implementation section, the intermediate system was described, which is built using PHP language with presenting its main functions including (upload file, view file, update file, delete file, organize file operations sequence and store Log File).

Chapter 5

Experiments and Evaluations

Chapter 5

Experiments and Evaluations

5.1 Introduction

In this chapter, our experiments are presented, with the evaluation of the intermediate system, to detect the integrity violation of shared data in the cloud using hash algorithm, and detect any unauthorized modifications. As we didn't find a model to compare with, a comparative study can't be done, so an analytical study is applied in this work.

5.2 Experiments:

Five experiments were performed on our implementation to demonstrate its ability to detect shared files integrity violation, these experiments can be listed as:

1. Using different file sizes.

2. Test all model modules.

3. Using a way to change some bytes in any file (hack way).

4. Update a file many times consecutively by different users.

5. Update a file by many users at the same time.

5.2.1 Experimental Environment and Tools:

Our experiments are conducted with a laptop (msi), running windows 8 x86 operating system with core i5, with 8 GB memory RAM.

PHP 5 (Laravel Framework) is used as a programming language

5.2.2 Setup a Private Cloud:

We built a website that enables users to upload their files, update, read or delete these files. This website is hosted on an Intel(R) Xeon(R) server with CPU E3-1231 v332 Gb ram.

Figure (5.2) represents the home page of this website.

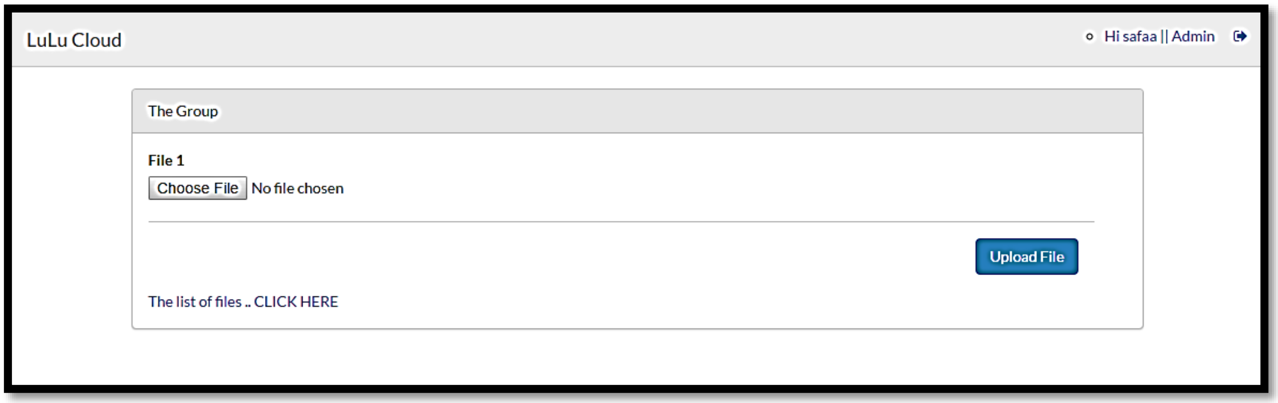


Figure (5.1): Home Page

5.2.3 Dataset:

To accomplish the experiments, a dataset must be collected, so, different sizes starting with 13 KB, ending with 5MB were chosen. The dataset includes different types of file size classification according to Microsoft Windows Explorer File Size Classification as seen in Figure (5.1). Type of files used is only (.doc).

At the beginning, files were grouped by their sizes to measure how much time every file needs to accomplish hash process.

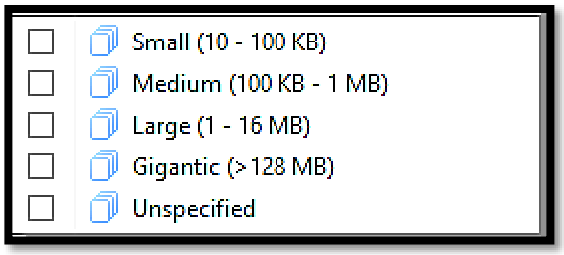

- ☐ Small (10 - 100 KB)
- ☐ Medium (100 KB - 1 MB)
- ☐ Large (1 - 16 MB)
- ☐ Gigantic (> 128 MB)
- ☐ Unspecified

Figure (5.2): Microsoft Windows Explorer File Size Classification (Barhoom and Saqallah, 2014)

According to this classification:

1. 38 small files (10- 100 KB), 26 medium files (100 KB- 1MB) and 6 large files (1-16 MB) were chosen.
2. SHA-256 hash algorithm was applied on these files.
3. As a result: we concluded that there is no need for this grouping because hash process doesn't exceed one second (less than 0.8 second) for any different size and there is no difference between hashing time of files despite their different sizes. So, grouping step was cancelled.

5.2.4 Model Experiments:

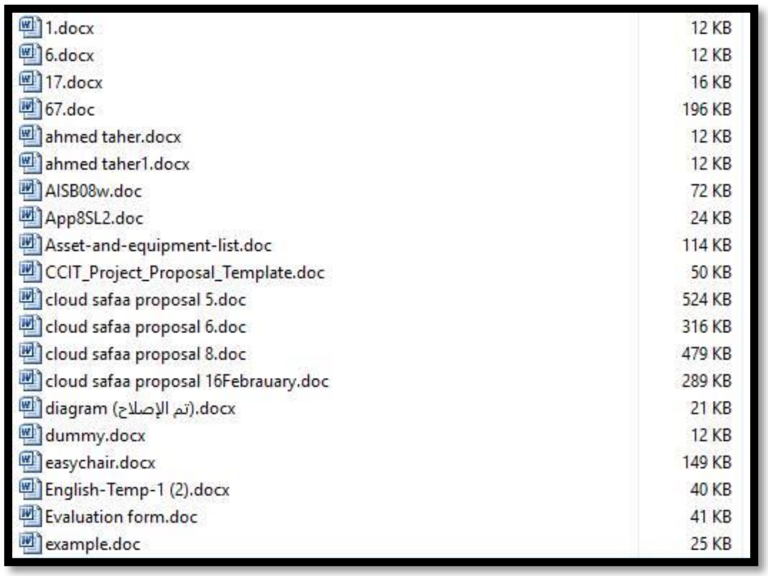
After collecting suitable data set (containing different sizes), four different types of experiments have been constructed.

- ✓ **Experiment 1**: Detect files integrity violation after intended hacking: try to modify file content without modifying hash value
- ✓ **Experiment 2**: Update same file many times consecutively: to test if hash value is re-calculated after each update with no conflicts.
- ✓ **Experiment 3**: Update same file concurrently: to test if model will allow more than one user to access the same file.

- ✓ **Experiment 4**: Hacking by different types of modification: modify very small parts of file contents to test if hash valued changed or not.

5.2.4.1 Experiment 1: Detect files integrity violation after intended hacking

The first experiment is to upload 70 files to our cloud, then 20 files of them was hacked by modifying their contents without rehashing files after updating them. Figure(5.3) presents the 20 hacked files.

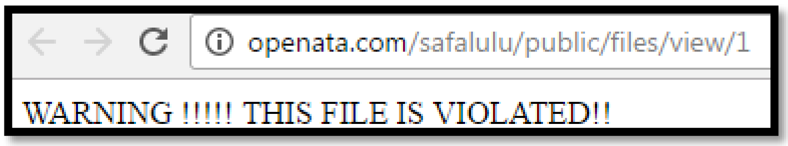


1.docx	12 KB
6.docx	12 KB
17.docx	16 KB
67.doc	196 KB
ahmed taher.docx	12 KB
ahmed taher1.docx	12 KB
AISB08w.doc	72 KB
App8SL2.doc	24 KB
Asset-and-equipment-list.doc	114 KB
CCIT_Project_Proposal_Template.doc	50 KB
cloud safaa proposal 5.doc	524 KB
cloud safaa proposal 6.doc	316 KB
cloud safaa proposal 8.doc	479 KB
cloud safaa proposal 16Febrauary.doc	289 KB
diagram (تم الإصلاح).docx	21 KB
dummy.docx	12 KB
easychair.docx	149 KB
English-Temp-1 (2).docx	40 KB
Evaluation form.doc	41 KB
example.doc	25 KB

Figure (5.3): Hacked Files

Then, we tried to download files (update in the normal way) to detect if the intermediate system can found the hacked files or not.

After applying this experiment, we found that the intermediate system didn’t allow users to update these 20 hacked files and it displayed a warn message to inform the user that these files integrity is VIOLATED!!! And the message displays that there is violation. As shown in figure (5.4).


openata.com/safalulu/public/files/view/1

WARNING !!!!! THIS FILE IS VIOLATED!!

Figure (5.4): Violation Alert Message

5.2.4.2 Experiment 2: Update same file many times consecutively

The second experiment is to let a user to update an uploaded file, then re-upload it and let number of users (three users) to update it again , one after one, to see if any conflict occurs or not.

This experiment approved that a file can be shared and updated consecutively by number of users with no conflicts or problems.

5.2.4.3 Experiment 3: Update same file concurrently

The first experiment is to let three users to try to access (to update) the same file at the same time. Our model has a property that gives a priority to the file creator.

So, we divided this experiment into two experiments: (file creator is one of the three users, file creator isn't one of the three users).

Experiment 3.a: File creator update File with others:

When one of the three users is the file creator, and they try to update the same file at the same time, only the file creator can download the file to update it and the file UPDATE property will be hidden immediately for the other users (including the two users in the experiment). After updating the file and re-uploading it, users can see it, read or update it again.

Experiment 3.b: Group of users with same priority update File:

When none of the three users is the file creator (all have the same priority). At the beginning of this experiment, when more than one user try to update the same file at the same time, sometimes, it was successful to let just one user to update, and stop the other users from updating the file, but in other times the model let more than one user to update, so this make a conflict problem.

After research, we used a Google technique code (that is used in Google Drive service) to permit to just one and only one user to update the file and the file UPDATE property will be hidden immediately for the other users (including the two users in the experiment). As a result, because of using this code, intermediate system enable one and only one user to update the file at the same time.

5.2.4.4 Experiment 4: Hacking by different types of modification:

We tried to modify little parts of file contents to see if hash value is effected or not so some files were hacked by modifying different part of the content:

a. Add a space to the file content.
b. Remove just one letter.
c. Remove one word.
d. Modify/ remove one line/ paragraph.

And after applying this experiment and trying to access these files, a warn message was appeared that File is violated!

5.3 Evaluation of the Model

The most important measure in our model is accuracy, in terms of security; it is measured by enabling a group of users to share one file many times consecutively and detect the integrity violation of this file at each access.

The other measure is to evaluate the concurrency, by enabling multiple users to access the shared file at the same time to detect any file access conflict.

5.3.1 Accuracy Evaluation

Accuracy refers to whether the data correctly records the business object or event it represents. It has two requirements: it must be the right value and it must represent the value in a consistent form with all other representations of the same value (Olson, 2002).

Our focus is to detect the integrity violation of shared file amongst multiple users; the users' access to the shared file is consecutively. Three experiments were conducted, in order to evaluate the accuracy:

1. Update one file directly in the server side for many times: (**normal way**)
    a. Change some bytes in the file (one word, one line, one paragraph).
    b. Insert some bytes to the file.
    c. Delete some bytes from the file.
2. Also, we repeat the previous experiment but in an illegal way (**Hacking**), mean to update any file without re-computing a new hash value.
3. **Collision resistant**: We checked our dataset SHA256 hash value for all stored files (more than 100 file), after files were hashed. We found that SHA256 is resistant to collision attacks, as there isn't any matching between two files hash values.

After we conducted all these tests, we found that our intermediate system is able to guarantee Files accuracy and detected all type of file modifications.

5.3.2 Evaluate Concurrency

Concurrency refers to the assurance that users can access data at the same time (Ashdown and Kyte, 2016). Our model enables just one user to update File at the same time, so when multiple users try to update one file at the same time, the intermediate system should enable just one user and stop others.

To ensure this property, the following experiment were conducted many times:

1. More than one user (about three users, sometimes four users) tried to access a shared file in the cloud to update it.

2. Intermediate system enable just one user to access it (using two ways as mentioned before in section 4.1 *define permission*) and notify others that File is turned on Write Mode now by disabling update property temporary.

After we conducted all these experiments, it is clear that our intermediate system is able to organize concurrency operations on shared file and enable just one user to modify File at the same point.

5.4 Summary

In this chapter, conducted experiments on our model were presented and results were analyzed, also, the evaluation of the intermediate system, to detect the integrity violation of shared data in the cloud using a hash algorithm (SHA-256).

Chapter 6
Conclusion and Future Work

Chapter 6

Conclusion and Future Work

6.1 Introduction

In this chapter, we concluded our work, results, and the future work directions. In this research, the problem of detecting the integrity violation of shared file in the cloud is discussed.

We mentioned how there is an increasing trend in outsourcing data to remote cloud servers, it's vital to ensure that data are kept properly in a consistent way and data integrity is guaranteed, especially, when data is shared between a group of users.

Many researchers discussed this problem and introduced some solutions to detect data integrity violation, which is stored in the cloud. But most prior works consider static data files and not supporting dynamic data operation or don't support file sharing. Few of them discussed the problem of detecting shared file integrity violation with dynamic operation support, but as known, many of cloud service provider pretend that user's data is saved securely with no data integrity violation. Actually, anything saved in the internet environment can't be perfectly safe.

On the other hand, we found that Github (Rouse, 2016) has supported both Dynamic Operations and File Sharing. Table (6.1) presents some of our similarities and differences between Github and our work.

Table (6.1): Differences and Similarities Between Github and Our Work

Property	Our solution	Github
Single update	Yes	Yes
Multiple update	No	Yes
Private cloud	Yes	No

It is clear that Github outweighs on us in multiple update, but it is a public cloud and we should take in consideration, trust issues especially when file contents are sensitive and can't assume any error.

So, we built the model to detect the integrity violation of shared files in the cloud with dynamic operation support. This model contains an intermediate system to detect file integrity violation by hashing parts of the file, using SHA-256 algorithm, before saving it on the cloud and after retrieving it, then compare the two hash values, if they are matched, then there is no problem, else integrity is violated and alert message is appeared.

Several experiments were conducted with different (.doc) files sizes (from 13KB to 5MB) to ensure the ability of detecting data integrity violation, and to evaluate the accuracy and concurrency.

The evaluation process proves that our model has been performed at an acceptable level and is highly efficient against malicious data modification attack.

6.2 Future work

As future challenges the research approach can be updated through adding the following:

1. Support more different types of files (image, voice, and video).
2. Make a periodically check on less used files in the cloud, to detect their integrity violation.
3. Work on other types of data such as structured and tabular data.
4. Enlarge work scope to include error localization and error recovery.

The Reference List

The Reference List

Aarthi, N. Indira. (25 Feb 2016) *Enabling Efficient and Protected Sharing of Data in Cloud Computing*. Paper presented at the International Conference On Information Communication And Embedded System (ICICES 2016).

Apostu, A. ; Ularu, G.; Suciu, G.; Todoran, G. (2012). *Study on advantages and disadvantages of Cloud Computing – the Advantages of Telemetry Applications in the Cloud.* Gartner's Symposium/ITxpo in Orlando.

Ashdown, L. (2016). *Concurrency.* Retrieved Dec. 30, 2016 from https://docs.oracle.com/database/121/CNCPT/consist.htm#GUID-7AD41DFA-04E5-4738-B744-C4407170411C

Barhoom, D. T. a. S., Zakaria Kh. (2014). "*A Model to Ensure the Integrity of User's Data in the Cloud.*". Islamic Univercity.

Carabotx, E. (2009).*Security Rules*. Retrieved Feb. 12, 2016 from http://techtalk.gfi.com/taking-security-seriously

Chad R. Dougherty, *(2008)*. *Vulnerability Note VU#836068 MD5 vulnerable to collision attacks. CERT Carnegie Mellon University Software Engineering Institute.* Retrieved Nov-15, 2016, from https://www.kb.cert.org/vuls/id/836068

Chalse,R.; A. S. a. A. K. (27 Sep 2013). *"A New Technique of Data Integrity for Analysis of the Cloud Computing."*. Paper presented at the CICN '13 Proceedings of the 2013 5th International Conference on Computational Intelligence and Communication Networks.

Chenxi Wang. (2009). *Cloud Security Front And Center*. Retrieved March 22, 2016 from http://blogs.forrester.com/security_and_risk/2009/11/cloud-security-front-and-center.html

Cowburn, P. (1997).*PHP*. Retrieved March. 22, 2016, from http://php.net/manual/en/intro-whatis.php

Ellingwood, J. (2013). *Linux Permessions*. Retrieved Feb. 12, 2016, from https://www.digitalocean.com/community/tutorials/linux-permissions-basics-and-how-to-use-umask-on-a-vps.

Gunjal, Y. a. J., J.Rethna Virjil. (2013). "Data Security And Integrity Of Cloud Storage In Cloud Computing.". *International Journal of Innovative Research in Science, Engineering and Technology 2(4).*

Ibnu (2016).*SHA*. Retrieved Dec. 10, 2016 from https://steemit.com/steemit/@ibnu/nice-selection-of-steemit-in-choosing-256-as-hash-algorithm

Johnston, S. (2016). *Cloud Computing Chart* Retrieved Feb. 22, 2016, from https://simple.wikipedia.org/wiki/Cloud_computing.

Juels, A.; Kaliski, B. (2007). *"PORs: Proofs of Retrievability for Large Files."*. Paper presented at the The 14th ACM conference on Computer and communications security.

Kavuri, S.; Kancherla, S.; Rao, G. and Rao,B (27 Sep 2014). "*Data Authentication and Integrity Verification Techniques for trusted/Untrusted Cloud Servers.*" Paper presented at the International Conference on Advances in Computing,Communications and Informatics (ICACCI).

Khatri, T. ; Jethava, G. (2013). *Improving Dynamic Data Integrity Verification in Cloud Computing.*

Kumar, S.; Saxena, A. (4 Jan 2011). "*Integrity Proofs in Cloud Storage.*". Paper presented at the Communication Systems and Networks, Third International Conference.

Lekkas, D. (2012). *"Addressing cloud computing security issues"*. Future Generation Computer Systems.

Lianhong, Z.; Chen,H. (2010). *"Secuirty Storage in the Cloud Computing: A RSA-based Assumption Data."*. Paper presented at the International Coriference on Educational and Information Technology.

Lou, W. ; Ren,K.; Wang, Q.; Wang, C. (2009). *" Ensuring data storage security in Cloud Computing."* Quality of Service, 2009. IWQoS. 17th International Workshop.

Mahmood, Z. (31 August 2011). *Cloud Computing: Characteristics and Deployment Approaches.* 11th IEEE International Conference on Computer and Information Technology.

Marimuthu,K.; Sashi, K.; Setty,S.; Tainwala, K. (10 May 2014). "*Scalable and Secure Data Sharing for Dynamic Groups in Cloud."* Paper presented at the IEEE

International Conference on Advanced Communication Control and Computing Technologies (ICACCCT). .

Mehul A. Shah, M. B., Jeffrey C. Mogul, Ram Swaminathan. (2007). *"Auditing to Keep Online Storage Services Honest."*.

Mount, Ch. (2014). *FileSharing* Retrieved Feb. 22, from http://cloudtweaks.com/2014/01/online-file-sharing-and-the-importance-of-security.

Al-Mourad, M. H. a. (2014). *"Effective Third Party Auditing in Cloud Computing."*. Advanced Information Networking and Applications Workshops (WAINA).

National Institute of Standards and Technology. (2011). *"The NIST Definition of Cloud Computing".*

NetworkWorld. (2016). *Cloud computing*. Retrieved Feb. 22, 2016, from http://www.networkworld.com/news/2008/072808-open-source-cloud-computing.html?page=2

Nida and Teli, B. K. (20 March 2015). *"Efficient and Secure Means for Identity and Trust."* Paper presented at the International Conference on Advances in Computer Engineering and Applications (ICACEA).

Olson, J. (2002). *Accuracy*. Retrieved Dec. - 23, 2016, from http://www.information-management.com/infodirect/20021108/6019-1.html

VictorVictories. (2015). *Types of Cloud Computing Deployment Model.*
. Retrieved Dec. - 23, 2016, from https://www.ibm.com/developerworks/community/blogs/722f6200-f4ca-4eb3-9d64-8d2b58b2d4e8/entry/4_Types_of_Cloud_Computing_Deployment_Model_You_Need_to_Know1?lang=en

Ramesh, S. (2016).*MD5*. Retrieved Nov -15, 2016, from http://www.gohacking.com/what-is-md5-hash

Roseindia. *Outsourcing*. Retrieved Feb. 10, 2016, from http://www.roseindia.net/services/outsourcing/different-types-of-outsourcing.shtml.

Rouse, M. (2005).*Hashing*. Retrieved 15-April, 2016, from http://searchsqlserver.techtarget.com/definition/hashing

Rouse, M. (2014). *Erasure Coding*. Retrieved Feb. 21, 2016, from http://searchstorage.techtarget.com/definition/erasure-coding.

Rouse, M. (2016). *Github*. Retrieved Dec. 14, 2016 from http://searchitoperations.techtarget.com/definition/GitHub

Al-Saiyd, D. N. A. a. S., Nada. (Dec 2013). "Data Integrity in cloud computing security". *Journal of Theoretical and Applied Information Technology.58*(3)

Subramanian, R.; Kumar, S. (2011). An Efficient and Secure Protocol for Ensuring Data Storage Security in Cloud Computing ".". *IJCSI International Journal of Computer Science Issues. 35* (9), p8

Techopedia. *Concurrency.* Retrieved Dec. 10, 2016 from https://www.techopedia.com/definition/27385/concurrency-.

Thangavell, M ;Varalakashmi, P. (December 2014). *"A Survey on security over data outsourcing."* Paper presented at the Sixth International Conference on Advanced Computing (ICoAC).

Tumotech (2014). *Growth of cloud computing.* Retrieved Feb. 22, 2016, from http://www.tumotech.com/2014/04/21/what-you-need-to-know-about-the-explosive-growth-of-cloud-computing

Tsagklis, Ilias. (2013). *Cloud Computing pros and cons* from https://www.javacodegeeks.com/2013/04/advantages-and-disadvantages-of-cloud-computing-cloud-computing-pros-and-cons.html