SECURE VERIFIABILITY OF OUTSOURCED DATA WITH CLIENT VERIFICATION

By

Wajahat Sultan Malik

A thesis submitted to the faculty of Information Security Department,
Military College of Signals, National University of Sciences and Technology,
Islamabad, Pakistan, in partial fulfillment of the requirements for the degree of MS in
Information Security

September 2020

THESIS ACCEPTANCE CERTIFICATE

Certified that final copy of MS Thesis written by **Wajahat Sultan Malik**, Registration No. **00000280975**, of **Military College of Signals** has been vetted by undersigned, found complete in all respects as per NUST Statutes/Regulations/MS Policy, is free of plagiarism, errors, and mistakes and is accepted as partial fulfillment for award of MS degree. It is further certified that necessary amendments as pointed out by GEC members and local evaluators of the scholar have also been incorporated in the said thesis.

Signature: ___________________________________

Name of Supervisor: **Asst Prof Dr. Fawad Khan**

Date: _______________________________________

Signature (HOD): ____________________________

Date: _______________________________________

Signature (Dean/Principal) ______________________

Date: _______________________________________

ABSTRACT

The information security requirements are increasing due to huge expansion of data in various fields. For the Clients with the limited resources, dealing with huge data is always been a significant problem. Cloud computing offers a variety of interconnected resources to its users. Heavy computations are always been a problem for resource constrained devices. This problem is reduced to a greater limit as cloud computing provide ease to the users by providing the facility of outsourcing their different complex computational work along with utilizing the huge data storage of the Cloud servers. At the same time, the existence of untrusted service provider creates the problem of lack of effective controllability of outsourced data by the legitimate users, as clients may not be certain enough to have a complete smooth control over outsourced data. Hence the need to have secure outsourced data gets more strengthened. In this thesis, we have suggested a secure mechanism for outsourcing the data, its update, data / client verification along with data encryption. The cloud server is asked to have complex computations and return the desired data. In case of any change, corruption or dummy data is returned by the cloud server, the querying client identifies the error, and can change or update the existing data securely. This whole process is monitored by the client through generation of proofs. Moreover, in contrast to existing works in literature, the proposed scheme takes into consideration of the ownership verification, prior to entertaining any of the queries. The major intend is to have a higher level of efficiency and reduced computational costs for clients. The performance evaluation shows its effectiveness for resource constrained devices.

COPYRIGHT STATEMENT

DEDICATION

This thesis is dedicated to

MY LOVING FATHER, MALIK SULTAN MUREED (LATE) AND MY MOTHER

for their love, endless support and encouragement

ACKNOWLEDGEMENTS

I am grateful to Allah Almighty for giving me strength to keep going on with this thesis, irrespective of many challenges and troubles. All praises for HIM and HIM alone.

I am very grateful to my Project Supervisor Asst Prof. Dr. Fawad Khan, Co-Supervisor Asst Prof Dr Shahzaib Tahir and GEC members who supervised the thesis / research in a very encouraging and helpful manner. They always guided me with their profound and valuable support that have helped me in achieving my research aims.

I would also like to extend my feelings of gratitude towards Col (R) Irfan Hanif, Maj Faisal Munir Jatoi, Maj Imran Aziz and Syed Hassan Mussana for their continuous guidance, motivation and endless support.

Finally, I would like to express my appreciation to all the people who have provided valuable support to my study and whose names I couldn't bring to memory.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

Forward automatic update	FAU
Efficient privacy preserving outsourced computation framework	EPOC
Shared data integrity verification protocol	SDIVP
Third party auditor	TPA
Pseudo-random functions	PRFs
Decisional Diffie-Hellman	DDH
Provable data possession	PDP
Publically verifiable computation	PVC
Secure Collaborative PVC	SCPVC
Trusted third party	TTP
Public key infrastructure	PKI
Cryptographic hash function	CHF
Message authentication code	MAC
Manipulation detection code	MDC
One-way hash functions	OWHF
Universal one way hash function	UOWHF
Collision resistant hash function	CRHF
Message digest	MD
Simple Network Management Protocol	SNMP
Pseudo random number generator	PRNG
Linear feedback shift registers	LFSR
Verifiable outsource computation scheme	VOCS
Public key	PK
Secret key	SK
Public Parameters	PP
Cloud service providers	CSP
Outsource database	ODB
Merkle Hash Tree	MHT
Verifiable B-tree	VB-tree
Hybrid Authentication Tree	HAT

Embedded Merkle B-tree	EMB-tree
Verifiable database	VDB
Verifiable database update scheme	VDUS
Encrypted database	EDB
Oblivious transfers	OT
Personal identifiable information	PII
Cipher text	CT
Plain text	PT
Modular Multiplication	MA
Modular Exponentiation	ME
Modular Addition	MA
Man in the middle	MITM
Identity	ID

Chapter 1

INTRODUCTION

During the last few years, a very huge increase in collection of data sets has occurred. Due to very large size and complexity, the processing of big data is always been a problem [1]. Therefore different technologies, tools and applications are used for data processing and storage. The industry challenges and requirements are evolving with every passing day [2]. The major concern of the industry is customer/clients. Hence it is prospering in business and scientific applications. The significance lies in efficient analysis, professional processing, secure transit and storage, which also ensures data security, privacy and integrity [3]. Cloud computing is the usage of various kinds of servers, softwares, development platforms and storages by using internet. It is an addendum to the traditional computing [4]. Traditional computing have certain limitations as compared to that of cloud computing that has large computation power, processing resources and huge storage for data. The cloud has many distinct and distributed servers that continuously provide various desired services to their intended users. Cloud provides liberty to its users to have their cloud services through internet whenever and where ever they need using various terminals and the cloud is monitored and managed by different cloud service providers [5]. Clients are required to work in pay-per-use environment that enables them to have an access to the available network resources [6]. Clients have a quick and complete access to shared resources that may require sometime a little reciprocity with service providers. The client asks for the desired services or resources that are provided by the cloud rather than any firm entity. The desired application or service of the client will be running on some far distinct cloud but the client does not require any information regarding its physical location. By using various client devices like terminals, laptop, palmtop or mobile phone, client can access all network resources and perform different complex computation to achieve desired results.

With the exponential increase in cloud computing services and the intensity of users using this technology, this further leads to the limitations of computing, processing and storage. Due to the continuous development in cloud computing, the storage element associated with this technology is also becoming more and more significant. All storage services/devices are provided and managed by cloud storage [4]. Cloud storage allows various storage devices to interconnect and form a vast pool of storage system. All the data from different users are stored on this cloud storage. With many benefits it also bring some data security and privacy issues with it, as the client handover physical control of data to the untrusted remote cloud servers. Storage as a service allow its users to access their data anytime using internet. Therefore, ensuring data integrity is one of the major challenges. The functionalities of cloud storage is completely different than that of simple storage. Security, availability, integrity, reliability and performance are considered more important once we talk about the cloud computing. Outsource computation is one of the major cloud services that is utilized by the Resource constrained devices to have their complex computations to be executed at the cloud using cloud resources [7]. In this way the resource constrained clients easily complete their computational tasks and obtains the desired results in time. The cloud requires a valid input query to perform required computations on the already placed data and return the results to the asking client that further verify the results for further processing. In outsource data scenario, one of the major requirement is the amount of computational burden that a resource constrained client has to face and bear during initial / preparatory and verification phases. The computation cost at the client end must be reasonably less than that of original task computation cost, as the client is required / desired to perform these steps in reasonable amount of time.

At the same time there are certain issues related to the outsourced computation schemes. Cloud may give some wrong results by not using the correct data or by not performing the valid computations [8]. It may happen that it overwrites or delete some unused data in recent times, and client may asks for the same data after certain time, then cloud may send some old computed results or some fake output. The cloud may place the data at some wrong index, once client ask data for that particular index but get some other data or computed results. Hence these sort of problems give rise the importance of verifiability. The client must have a complete liberty and privileges to verify the data and computed results received from the cloud servers. The use of polynomials is increasing exponentially in various important fields like image processing, signal processing, information security, different types of data analysis, etc. However, problem of cost of computation arises with induction of high degree polynomials specifically for resource constrained devices and hence it is one of the major concerns in initial / verification process. Due to great advancements in internet, utility of cloud computing is expanding and getting more popularity. This increase in its utility spectrum give more importance to cloud storage [9]. The contents that are sent to the outsourced database require secrecy so as to cloud must not access the data entirely or even in parts [7]. Hence a secure mechanism is needed for secure transit of data and queries so they may not be revealed at any stage. Mostly the data being uploaded to cloud servers has been disguised by different techniques. Clients asks server to do computations on this disguised data rather than the original data. Due to the untrusted nature of the cloud servers, the correctness of the data and corresponding operations on the data are required to be intact [10]. The servers are also asked by clients to generate and send proofs to ensure the correctness and authenticity of the data. Moreover, the data has not been modified by the servers or any other unauthorized intruder. The original data has not been changed, added or deleted through any malicious activity by the cloud server. Hence the client must detect any of these problems through verifiability of proofs and ensure the data correctness.

1.1 Problem Statement

In this modern era of technology, number of users and devices are increasing exponentially and creating a large amount of data every second. The speedy development of cloud computing attract resource constrained clients to outsource this data to the cloud, as it reduces the storage, processing and computational overhead at their end. This also raises user's concern about the security and privacy of their data being stored outside their jurisdiction as the remote cloud server is semi honest but curious. Moreover, in this scenario the resource constrained clients keep only metadata of their outsourced encrypted data. Clients should have an accurate and complete control and trust over this outsourced data. There is a great requirement of provable verification mechanism regarding desired data updates have been made correctly or not. Hence the requirement of low cost data verification is also getting strengthened being the most integral part for ensuring data integrity. In this perspective, cost, memory constraints, privacy, maintenance, low processing power etc can create huge hurdles. Keeping in view these requirements, Outsourced databases can be chosen, employing polynomial computations as it is much more practical with appropriate computations on both sides, whereby maintaining the efficiency of whole system. Presently Pakistan has no cloud infrastructure. However, Pakistan Army has established a considerable setup of data centers. Application like eArms, Office Automation System, Hospital Management System etc. Therefore, we require a technique that can ensure the user's privacy in terms of data and different operations on data along with user authentication / validation over the cloud to ensure confidentiality and integrity. Designing a verifiable outsourced database with appropriate computations on either sides, is one of the major requirements for development in this particular field in Pakistan.

1.2 Motivation

In this thesis, we have explored different dimensions that can be utilized in these fields to cover the aspects and requirements of secure storage, data expansion, integrity and confidentiality in order to propose a verifiable outsourced database scheme that can help in increasing the overall performance of outsourced computations while keeping in mind the major drawbacks of existing schemes. Moreover, the verifiable outsourced database scheme could be easily used for devices having low processing and memory constraints with high degree of privacy, correctness and verifiability. The aim is to design a secure verifiable dynamic data outsourcing scheme which is better in performance and management.

1.3 Research Objectives

The main objectives of thesis are:

- Analysis of available verifiable outsourcing data techniques.
- Proposing a verifiable outsourcing data technique that encircle:
  - Client verification to make sure that the data is asked by, and going to the valid user.
  - No information about the stored data is exposed to the cloud server.
  - Computational results correctness is verified by the clients.
  - User be able to get data dynamically updated and verified.
- The computational cost for the resource constrained client should be low.
- Analysis of security and efficiency of the proposed scheme.

1.4 Contribution

The Secure Verifiable Dynamic Data Outsourcing with File based Client Verification will contribute in the following ways:

- The exponential factors for obtaining the verification vector by the client will be reduced while having no compromise on security. This will reduce the computation cost at initial stage for the client.
- The verification of the client (querying data) by the cloud will be introduced in the scheme to ensure the authenticity of user. The cloud will be able to differentiate between valid and invalid users.
- The security and privacy of original data along with verification information will be ensured at all stages of uploading, searching against queries or in updating process. Cloud will not be able to access data to deduce any useful information.

1.5 Thesis Outline

The research work has been organized and distributed in following chapters:

- **Chapter 1**: A brief introduction is given, problem statement is highlighted, followed by motivation behind the research and research objectives are identified / explained. Furthermore, the contributions made through this research are highlighted.
- **Chapter 2**: A birdseye view of existing / recent research that has already been carried out in the field of verifiable data / databases outsourcing using various techniques is discussed.
- **Chapter 3**: An introduction / brief description of preliminaries regarding data / database outsourcing paradigm, core elements and phenomenon of data outsourcing followed by the challenges and problems.
- **Chapter 4**: This chapter starts with the system model, followed by the system architecture objectives. In the end, proposed system algorithm is discussed.
- **Chapter 5**: : This chapter contains the security and performance analysis of our system. Comparative comparison of our thesis work with existing research, followed by the results based on performance parameters.

- **Chapter 6**: This final chapter summarizes the research, with conclusions drawn and presents future work objectives.

Chapter 2

EXISTING RESEARCH IN DATA OUTSOURCING PARADIGM

2.1 Introduction

In this chapter, various research works carried out at different times are discussed. The techniques used in proposed algorithms are highlighted. Various positive and negative aspects of these algorithms are discussed in context of their utilization and deployment to fulfill particular / intended requirements.

2.2 Existing Research

Data validation, updating and storage in client server environment is always been a challenging task. Researchers worked and proposed different concepts and protocols to solve different associated problems. Various verifiable computations schemes have been presented at different times. In [1], *Shen et al.* presented a public auditing protocol for outsourced database. The protocol addresses the issue of mistrust between data owners and remote cloud by introducing Global and Sampling verification technique to make both parties confident about each other behavior. Efficient auditing is achieved by structured relationship between blocks of data and their respective locations. The algorithm make use of doubly linked info tables and location arrays. In [2], *Chen et al.* identified a vulnerability (regarding Forward Automatic Update (FAU) Attack) in an efficient verifiable database framework presented by Catalano. The new proposed scheme was based on commitment binding. The construction was publically verifiable and secure against Forward Automatic Update (FAU) Attack. The scheme make use of bilinear pairing groups of prime orders.

In [3], *Zissis and Dimitrios* presented general designs principles for cloud usage by evaluating cloud security along with their solutions to eliminate vulnerabilities and potential threats. The paper also discusses deployment models for cloud environment. In [5], *Ma et al.* presented a scheme that focuses on refereed computation delegation of sequence comparison. The algorithm make use of multiple servers. Users can get their desired correct results as far as system contain at least one honest server. Moreover, user can identify the malicious / misbehaving server in the system. Proposed scheme also preserve input / output privacy against malicious servers. In this paper [6], writers studied and presented another camouflage technique for data outsourcing, which allow users to hide their high degree computation polynomials along with all the inputs and outputs to the system. The client are given liberty to verify the results. Their source constrained clients use the high degree polynomials with fixed coefficients without homomorphic encryption. Disguised polynomial is constructed for outsourcing so that the server remains uninformed from polynomials and input / output. Clients present all inputs to the server in an encrypted manner. Authors have also presented a scheme for resource constraint client to verify the keyword searching by the server so that the client can verify that server has really performed the desired actions and has returned the valid results. The scheme introduces three parties i.e. client, server-1 for verification of the results if searching server (server-2) has returned the null results, a scenario in which server saves its computation resource and do not perform any computation and try to fool the client by sending a fake null value and server-2, the server that perform / implement the searching for desired value for client.

In article [7], the authors have introduced a concept of verifiability of outsourced data. This is achieved through a technique named as obfuscation technique. In this technique, the outsourced computation polynomial is obtained by multiplying outsourced polynomial with a hidden factors along with addition of a user defined polynomial. Server cannot get any useful information regarding multiplicative factor and user defined hidden polynomial from outsourced computation polynomial. Hence requirement of hiding of polynomial from server or any unwanted entity is obtained. The concept can also be implemented for information concealment. The protocol offer users to verify the received computation results from server. The time required for verification process is quite lesser than computation on original polynomials. This allows computationally weak clients to outsource and verify their data.

A privacy preserving framework with the name Efficient Privacy-Preserving Outsourced Computation framework EPOC [8] was presented by *Liu et al.* The framework was designed to secure the privacy of computation functions along with the output results. The user is required to define the computation functions. The protocol introduces a trusted party in the system having the responsibility of managing and distributing the keys within the system. The protocol allow its users to have multilevel privacy depending upon needs. Different levels of privacy can be achieved through basic enhanced and full EPOC, by outsourcing computation functions coefficients and exponents. Similarly protocol define its privacy levels to be meet by hiding / securing its functions parameters in order to ensure user's privacy. However, the EPOC does not provide verifiability of outsourced data computations. In paper [9], A loop hole in protocol, shared data integrity verification protocol (SDIVIP) proposed by *Yu et al.* was found. The major focus of the protocol was to ensure integrity of the data outsourced to remote untrusted servers. Authors have shown how an active adversary can execute man-in-the middle attack. Adversary can corrupt any data file by changing all blocks of data. Once untrusted server receive any query and after computation, send proof, adversary intercept proofs and modify them as per data changes. Writers have presented two possible ways to avoid this attack by introducing signature from the server or by using encryption of proof with TPA public key so that any change in data cannot by wrongly proved as legitimate by its corrupted proof. The security and source of data over public untrusted server has always been of prime importance. The previously presented multiple schemes deals with outsourced data from single source / user / entity. Author, in this paper [11] addresses this issue by introducing the cloud as an intermediate party along with a trusted authority that is used to add and authenticate the entities within the system. The user request the cloud to perform desired computations over a defined data set. The scheme makes use of bilinear mapping and bilinear map accumulation for the said purpose. As the user / clients cannot ask the workers for the entire data set. Therefore server does all computations. As the whole operation requires privacy protection, therefore certain different mechanisms are used for privacy preserving i.e. Elgamal encryption and keyed hashing for data privacy preserving, whereas the identity (ID) privacy is preserved through the ring signatures. The client receive computation results along with the computational proofs from server. Trusted authority is responsible for distribution of public and private keys to the system entities.

The computation cost at client end is that of intersection set verification. *Rosario et al.* [12] proposed a scheme for verifiable computation by using homomorphic encryption for random functions. They introduced a verifiable protocol between two sides. The client is required to perform some costly computation/preprocessing only once. The protocol require its client to perform this one time costly computations in a secure and protected manner/environment. In order to secure the auxiliary information related to computation function, these computations can also be outsource to some trusted entity. The client send the required public parameters to workers for computation of function. Workers send the computed results for verification to client. *Atallah et al.* [13] worked on outsource computations by making use of matrices, equation etc. The scheme does not allow its users to verify the results. Key exposure is an issue of prime importance once we talk about providing security, as the core defense lies in the secure placement of keys. The same problem is discussed in [14]. In proposed solution, client is required to update his secret keys in every session / period, but this solution invites another major problem of computation at resource constrained users as this increases the computation burden especially in frequent key update scenarios. The major focus of this article is to make the process of key updates more secure and private. In order to solve this additional problem of computations at client's end the authors have introduced a trusted authorized party (third party auditor, TPA). This TPA will be responsible for security, storage and key updates. TPA is given access to the encrypted versions of user secret keys. The client is required to download the new updated encrypted key along with the computational proof. The initial secret encrypted user key is generated at client end and is sent to the TPA, which generates the corresponding session key for the user. The protocol is based on binary tree structure. This tree structure is useful for fast key updates and instead of real keys it updates the encrypted keys to achieve privacy / security. Due to the increased use of cloud technology, outsourced computations with different requirements are also rising.

In article [15], the writers studied and presented a scheme, in which the user can verify the integrity and correctness of different required computations in a public way (without requirement of private key and any desired third party can verify the results). This scheme makes use of Matrix multiplication with high degree polynomials. The protocol rely on pseudo-random functions along with closed form efficiency to handle the multiple functionalities. It also uses PRFs based decisional linear assumptions for their employment in groups and bilinear maps. The writers than discussed multi functions, in which clients generates and send different input parameters to server for storing in advance, then client once required, send different desired queries for required functions. These inputs will remain blind to functions and servers. The multi functions verifiable technique can be delegated publically and then can be verified locally / privately. The above mentioned technique is suitable for retrievability proofing, keyword search, discrete Fourier transform and linear transformation. In paper [16], a solution to the problem of outsourced computation on large data that is available / stored on remote untrusted servers has been presented. They used amortized verifiable computation scheme. The scheme was presented for high degree polynomials. The functions used in this scheme are utilized to have predictions on a wide range of data sample space during required calculations. The same verifiable scheme is used for verifiable keyword search and retrievability proofs. To achieve security the whole construction was based upon DDH assumption along with its variants. The other focus of the paper was on verifiable databases. The resource constrained clients outsource their large tables / set of tables to an untrusted server and then as per requirement and make updates, add, delete and retrieval queries. Verifiable database scheme is proposed on the basis of hardness of subgroup membership problem in bilinear groups. The authentication computations is near to that of encryption of a data using Elgamal encryption due to its exponent operations. In verification, the computation is done under DDH assumptions that takes logarithmic number of power/exponent operations. In research article [17], the writer studied and presented a solution to a problem in which a resource constrained client can verify the correctness and originality of data placed on remote server without retrieving it. The model, provable data possession (PDP) take random set of data from the remote server and computes probabilities proofs and this ultimately decreases the input / output cost as the protocol need not to access the whole data / file, rather it randomly takes sample data from entire data set. The client is required to store a small amount of data regarding proofs. The output results in this phase transmit reasonably small amount of data which also decreases the transmission load on the channel. Both challenge and response use homomorphic verifiable tags, and then these computed tags are placed with their corresponding files to the server. Therefore the PDP model can be implemented on wide distributed storage / servers system. Another publically verifiable computation (PVC) scheme was presented by *Wang et al.* in their research paper [18]. The proposed scheme allow its users to have a correctness / integrity checking mechanism through witness vouching. The scheme does not require the resource constraint clients to outsource their computations polynomials and functions in plaintext or encrypted form before computation / at once. To advance this notion, a scheme called Secure Collaborative PVC (SCPVC) was presented. A trusted third party is made responsible to initialize the system and disseminate some required initial information to desired entities. After certain rounds, the untrusted public cloud receives certain required information. The private / intermediate cloud / party perform algebraic operations (using identifiers shared in untrusted public cloud and in private clouds). The target computation function is then generated jointly by these entities based on previous rounds attributes and algebraic operation. In this way the powerful untrusted server gets all information for desired computations which do not disclose any respective secrets, whereas the private cloud / party will have no information of target function. SCPVC generates the vouching parameters for integrity and correctness for the private cloud / TTP. TTP can check the correctness of target function and corresponding signatures. Communication over head in the network is one of the major reasons of the delay in network traffic.

The main target of the research by *Ma et al.* [19] is to address this issue by reducing the network traffic load. The researches have presented the communication efficient authentication scheme to verify the computation results received from server. Each result tuple has its authentication signature. To reduce the communication load, signatures of multiple result tuples are combined/aggregated to single signature. To authenticate the associated verification information, proposed mechanism uses tuple based Merkle hash trees. The proposed protocol also support union and intersection operation. The security and source of data over public untrusted server has always been of prime importance. Various previously presented multiple schemes deals with outsourced data from single source / user / entity. *Song et al.* in [20] proposed a new outsource computation scheme, which deals with the data from multiple sources. The scheme allow its all users to outsource their respective data to server at any time. All inputs parameters of desired function need not be signed by single entity in the system. The protocol evaluate polynomial functions over multiple data sources. The verification cost in this proposed scheme do not depend upon input parameters and polynomial size. In [21], *Pang et al.* proposed scheme that focuses on trustworthiness, entireness and correctness of desired outsourced data. The scheme supports both keyed and non-keyed input attributes and also make use of relational databases.

Xiaofeng et al. in [22], proposed outsourcing scheme for large linear equation system by making use of sparse matrix. The algorithm require single round communication between server and its clients. In [23], *Jiang et al.* identified and proposed a solution for major mutual authentication flaw in algorithm proposed by Tsai and Lo against impersonation attack by an untrusted server. *Yu et al.* proposed another outsourcing scheme [24] by making use of matrices. The algorithm hide original matrices and their inverses and allow server to identify any error while combining different random functions.

In [25] *Zhou et al.* presented polynomial based outsourcing scheme by introducing polynomial disguising. Another outsourcing algorithm was proposed by *Ye et al.* based on modular exponentiation [26] by making use of two non-interacting untrusted cloud servers. The input and output values remain secret to the servers. In [27] *Seung et al.* presented an outsourcing schemer for multi clients. The scheme allow their clients to interact directly with server with the help of PKI and counters.

Wang et al. presented a verifiable search scheme [28] for outsource databases without using pre-counting process. The scheme can be extended for multiple users by making use of multi part searchable encryption. *Jian Shen et al.* proposed an anonymous data sharing and storage scheme for multiple users with in the same group [29]. The scheme allows its users to communicate and store their data on cloud securely and anonymously, however identities can be traced back once required.

2.3 Conclusion

The chapter highlighted various data outsourcing schemes currently deployed / used in various client server environments. These algorithms are distributed in nature and are designed to address different issues / problems in data outsourcing paradigm. Most of the work is related to resource constrained clients in terms of computation, storage, processing etc.

Chapter 3

CRYPTOGRAPHIC PRELIMINARIES FOR VERIFIABLE DATA OUTSOURCING

3.1 Introduction

This chapter gives a broad view of preliminaries regarding verifiable data outsourcing. Different major cryptographic techniques used in such schemes are discussed. The security challenges, properties, threats along with some significant problems / limitations and their corresponding vulnerabilities are highlighted.

3.2 Hashing

The term hash function has been widely used in different fields of computer science since ages and employed to get various security services. It is a process that takes real data (or may be any desired data) as its input to have a certain processing and produce its fix stretched output. The output value is often quite shorter (smaller) than that of original input. The output of this process is strictly related to its corresponding input in such a way that a slight change in input may have a huge impact (produce a great difference in input). The resultant output is a reduced insinuation of the original input. A hash function must be having following properties in order to fall in a secure and trusted category.

- The underlying process of conversion of defined output from any given input I must be time efficient. The computation duration of the process should not be extensive time taking.

- The output O should not leak any handy information (may it be about complete, or in part) of original data about its corresponding input I.

- The process of getting back from output $O = h(I)$ to input I (from hash value to its corresponding input original data) be quixotic.

- Let the two distinct inputs be I and J, may not be having outputs like $h(I) = h(J)$, or

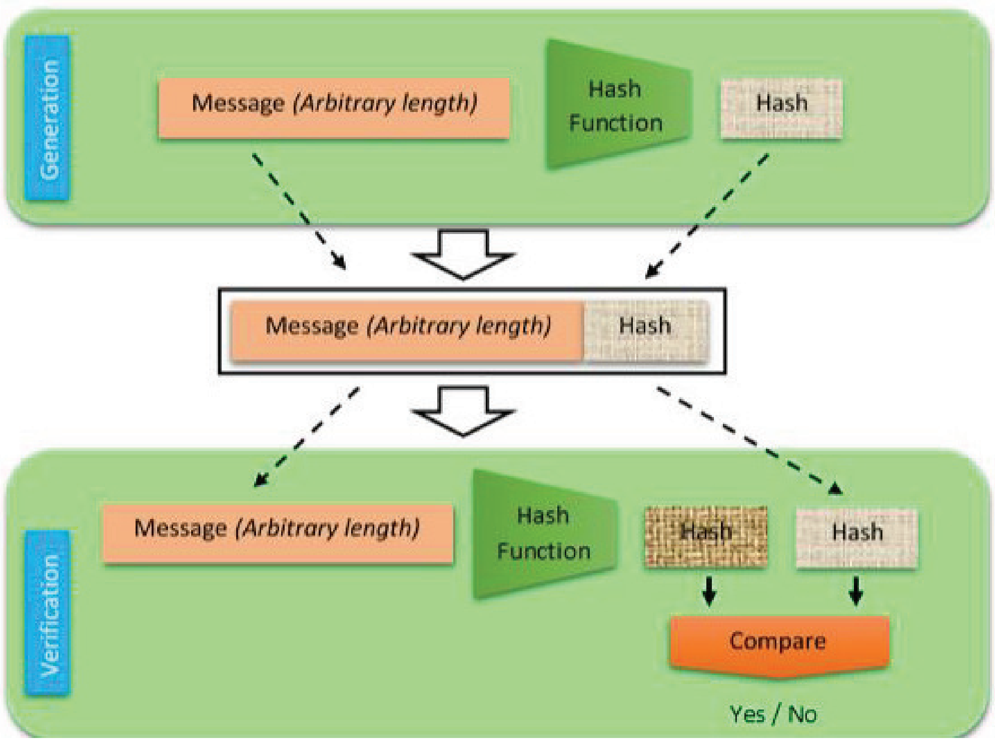

Figure 3.1: Hashing

by having any change in input I, such that I is transformed to I_1, their corresponding hash values may not be like $h(I) = h(I_1)$.

- The output hash value $O = h(I)$ should be arbitrary.

- Finding two distant inputs I and J such that their output hashes $h(I) = h(J)$, be computationally absurd.

Hash function if satisfy additional requirements (as described further), can be employed / used for various cryptographic applications and hence termed as Cryptographic hash functions.

Cryptographic hash function (CHF) is one of the tools that is of prime importance in the field of Cryptography. CHF are used to attain various security services like digital time stamping, digital signatures, authority, pseudo random number generation etc. In [30], *Ahmed et al.* have suggested that the use of Cryptographic hash functions to achieve different level of security in information processing application is much broader than that of block / stream cipher application.

Rompay [31] has defined CHF as:

A Cryptographic hash function is a function CHF: d → r, where domain $d = \{0,1\}^*$ *and range* $r = \{0,1\}^n$ *, where* $n \geq 1$.

Cryptographic hash function can be widely divided into two categories i.e., Non Keyed hash function or un-keyed hash function. This type of CHF don't make use of a secret or user defined key during its usage. The other type of CHF is keyed hash function; is the one which uses secret/ user defined key during process. In general the term keyed hash function are referred to message authentication codes (MAC) and the term Un-Keyed or Non Keyed hash functions are used for hash functions. Un Keyed hash functions are also sometimes referred as Manipulation Detection code (MDC) and can be further classified into different branches likes one-way hash functions (OWHF), Universal one way hash function (UOWHF) and Collision resistant hash function (CRHF).

- **One Way Hash Function** *Merkle* [32] has explained one way hash function (OWHF) is a function that satisfies the properties as explained below:

  - OWHF can be used for any data of arbitrary length.
  - It always give a fixed length output.
  - knowing function and its output, it is computationally infeasible to find two different inputs a and a' such that $H(a) = H(a')$

  The third property is called as one way property or pre image resistance and it explains that it is virtually impossible to get the actual input from received output and function. The fourth property is known as second pre image resistance property and states that no two different inputs can have the same output / hash value.

- **Universal One Way Hash Function** *Lin et al.* [33] proposed the idea of universal one way hash functions and not based on any trapdoor function presented a digital signature scheme by making use of definite number of one way hash functions with an equal same probability of being used.

- **Collision Resistant hash function** One of the definitions of Collision resistance function is explained as, [34, 35] CRHF is a hash function that appease all properties of

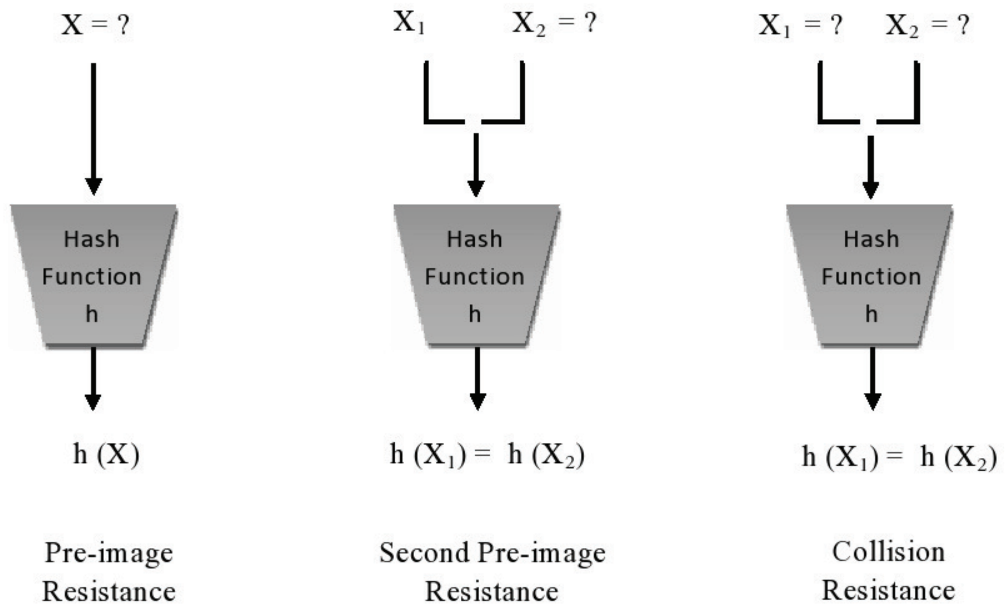

Figure 3.2: Basic Cryptographic Properties

OWHF and additionally satisfy the property; that it is computationally infeasible to find two district inputs (a, a') such that $H(a) = H(a')$.

3.3 Security properties of hash functions

Some of the major security properties of hash functions are described as following:

3.3.1 Basic Security Properties

The basic security provided (expected) by hash functions are collision resistance, pre-image resistance and second pre-image resistance. Collision resistance property is also known as strong collision resistance or collision freeness. Pre-image resistance is also called as one-wayness and second pre-image resistance is referred as weak collision [30]. Generally it is assumed that if a hash function is collision resistant than it will be second pre-image resistant as well, but second pre-image resistance and one wayness are incomparable. In practice, collision resistance property of hash function is most important [43, 44].

3.3.2 Avalanche criterion (Avalanche effect and completeness)

Avalanche effect described that a very small fluctuation in inputs bits has a huge and significant effect (turn around) on output bits. Whereas completeness, property representing that each input bit effects all output bits [45]. Avalanche criterion combines both properties i.e., Avalanche effect and completeness.

3.4 Security Services provided by Cryptographic Hash Function (CHF)

Some of the major security services provided by Cryptographic hash functions are described as following:

3.4.1 Users Authentication

Hash functions have been used by efficiently and since quite long time for authentication of users. The users passwords are saved / stored in the form of hashes (message digest). Users are asked to provide their passwords and then message digest of these passwords are calculated in order to be matched against the already stored hashes of original passwords. If it matches access is being granted else users are not authenticated.

3.4.2 Integrity and Authentication of data

Verification of data is one of the most important factors in modern days. It is now one of major necessities of networks. The two communicating parties are always interested and concerned about the authenticity of the data being received [36] it can be implemented in multiple ways. Symmetric encryption mechanism have drawbacks like speed, cost etc. [37,38]. Such methods are not efficient where encryption of complete message is not required as they combine confidentiality and authenticity/integrity e.g., in SNMP (Simple Network Management Protocol) the more important is to authenticate the incoming SNMP command and the concealment of this traffic is not require. MAC or hash functions are also been used for the same. The idea of constructing MAC from CHF is presented in [36]. Hash function are without symmetric encryption is used for authenticity and data integrity. The usage of hash function for message authenticity and integrity is rising because generally they are faster as compare to block ciphers in their implementation in different softwares.

3.4.3 Digital Time Stamp

Digital time stamping is employed to ensure temporal authentication [36] as when the particular digital document / data has been created or modified / changed. It also helps in protecting and ensuring auditing procedures, non- repudiation etc. *Haber et al.* [39] also shown the implementation of Digital time stamping through one way hash function.

3.4.4 Digital Signatures

Digital signatures are widely used for authenticity, integrity and non-repudiation in different ways. Hash functions are used to revamp the digital signature schemes. The major concept here is instead of signing the whole long message the sender only signs the digest of the message using digital signature algorithm, which upon receiving the receiver compute digest and then using the same hash function as reader and authenticate upon verification [30].

3.4.5 Pseudo Random Number Generations (PRNG)

Hash functions as one way hash function are used in implementation of PRNG in linear feedback shift Registers (LFSR) and in many other processes. The algorithms generally takes initial values known as seed of defined length to give certain outputs (the same may be used as inputs is certain other rounds in order to get final output) [40–42].

3.4.6 Other Applications

Hash functions are also widely used in various fields like indexing of data in hash tables, detecting of duplicate data, fingerprinting, as checksum for detection of deliberate / accidental data corruption and many more.

3.5 Verifiable Outsource Computations

Verifiable data outsourcing schemes are generally employed in distributed algorithms. A verifiable outsourcing computation scheme is generally a two party scheme / protocol i.e. Client and Server (number of clients and server may vary from scheme or as per requirement). The clients in such schemes are usually resource constrained in terms of power, computation, storage etc. These resource clients provide their required input data to servers along with their private / hidden functions and parameters. The servers upon receiving this information, respond with desired actions to send output with some affiliated proofs for result verification and correctness. The client then verify the received results that whether it is the desired result of actual input data and functions or not.

One of the major goals of verifiable outsource computation schemes is to reduce the verification cost of results at client end and to make this process more efficient and less time consuming. The verification process is required to be faster (less time consuming) than that of the actual computation time of original function. The resource constrained client is required to put one time extensive computation for hidden / private functions in order to generate public and private keys/ polynomials / parameters, which will be further used in the process of verifying results and outputs. These parameters will remain same for verifying results of multiple outputs against corresponding inputs [48]. Verifiable outsource computation schemes generally make use of following algorithms:

Verifiable outsource computation scheme [12] VOCS (*key generation, input coding, computation of output, verification of results, update*)

- **Key generation** $(F, SP) \to (PK, SK)$: Using defined security parameters, the client generates the Public and Private key pairs (PK, SK) for hidden functions F. The secret / private key is kept by the client for verification process while Public key is sent to server.

- **Input encoding** $(x, SK) \to (x', PP)$: Using secret key and input data, client encodes the desired data x to x' for sending it to server and some private parameters PP which is / are kept by the client.

- **Computation of output** $(x', PK) \to (y')$: Server using client pubic key and client generated encoded input to get the encoded output result / version of the functions F.

- **Verification of result** $(PP, y') \to$ (y or null): Client using his secret key, generates private parameters, verification algorithm converts the encoded output received from server, into desired output of functions F in order to verify whether it is the actual output or gives null value for invalid output.

- **Update** $(y, SK) \to U$: Client using secret key update any previous data / input into new data / input for further sending it to server for respective update.

3.5.1 Security

A verifiable data outsourcing scheme is supposed to be secure and correct. A scheme is said to be correct if values generated by client during input encoding algorithms allows an honest cloud server to compute results that can be further verified correctly at client end. [49, 50] Verifiable data outsourcing scheme is considered as secure if any malicious server cannot make client believe to accept random / corrupt outputs as correct.

Example (VOCS, F, SP)

```
$(PK, SK) \leftarrow$ Key generation $(F, SP)$
For $n = 1 \ldots l = polynomial(SP)$
    $x_i \leftarrow (x_1, x_2, \ldots, x_n)$
$(xi', PP) \leftarrow$ Input encoding $(x_i, SK)$
    $x_i' \leftarrow (x_1', x_2', \ldots, x_n')$
$y_i' \leftarrow$Computation of output $(x_i', PK)$
$y_i \leftarrow$ Verification of result $(PP, y_i')$
    $y_i(PP, SK, y_i')$
if $y = null \leftarrow y \neq F(x)$ else $y = F(x)$
```

3.5.2 Efficiency

Finally we assume that verifiable data outsourcing scheme be efficient [51–55], which means that time requires by the scheme to encode the input and verify the results being correct or otherwise, must be less than time required for actual computation of function F.

3.6 Outsource Database Verifiability

Verifiable outsource databases provide different mechanisms for data uploading, verification and updating. Through defined security specifications, the public and secret keys are generated. The client then specify the mechanisms for data encryption, decryption, how the received results to be verified, how different queries will be generated, how the desired data will be updated etc. The cloud in response will perform certain computations and retain the corresponding cipher text along with some proofs. The proofs will contain certain information that will verify and indicate the authenticity of required data along with its update history. For verification process client will make use of the secret keys in defined algorithms. Clients will also make sure the completion and verification of the updated data by verifying the proofs.

3.6.1 Verifiable Data Storage on Outsourced Databases

Due to huge data expansion all around, database outsourcing has emerged as one of the most important fields in research and practical usage. Initially database outsourcing concept was introduced by *Hacigumus et al.* [56]. This allow the owners of data to delegate the management of databases to Cloud Service providers (CSP), that are made responsible to provide their users with database services. Generally in outsource database (ODB) scenarios, resource constrained clients encode / encrypts their data locally with some private parameters including secret key along with some other corresponding metadata (identification data with respect to corresponding encoded / encrypted data) and then outsource this whole data to outsourced database that is being hosted by cloud service provider. The data owners upon requirement can ask data through queries. In ODB system, generally we have entities i.e; data owners, data users and cloud service provider. Authorized data users can query their required data through encrypted queries and CSP process these queries over encrypted data for desired computations and outputs.

3.6.2 Security Challenges

While having the number of various benefits, the paradigm of outsource databases come across few security challenges as well. Particularly due to hardware / software failures, interest issues computation etc. cloud may not perform complete computations honestly and may return some old / incorrect / corrupted result. As client do not have the data copy locally so data integrity is also emerges as one of major security challenges [57]. For integrity auditing of outsource databases researchers have presented two major categories of approaches, i.e. Authenticated Data structure based integrity verification. This approach is based on authenticated data structures, etc, Merkle Hash Tree (MHT), Verifiable B-tree (VB-tree), Hybrid Authentication Tree (HAT), Embedded Merkle B-tree (EMB-tree) [58–64] and the other approach is Signature Chaining Based integrity verification. This approach is based on signature chaining technique [65–68].

3.6.3 Threats / Attacks

In outsource database paradigm, generally cloud service providers are referred as "Semi honest but curious". So it is doubted that whether CSP perform all actions honestly or not. Generally we assume two types of attackers:

- **External Attackers** An entity want to gain knowledge for which they are not authorized (partially or completely) through any unauthorized or hidden way.
- **Internal Attackers** An entity having some / complete knowledge of ODB and wants to send incomplete / corrupt results to the data owners.

3.6.4 Verifiable databases with updates

A database in general is considered as set of data tuples that contain a unique index and corresponding data item. A verifiable database (VDB) paradigm allow resource constrained clients / users to outsource their database containing huge data with the aim that upon requirement they can ask / get their required data anytime and can update record on server. Any change / corruption to the data by any unauthorized entity or dishonest server may be identified with larger probability. In order to achieve data integrity, data owners verify the result received from remote cloud server by making use of their secret key and private parameters. Data confidentiality is achieved through encryption of data using any desired encryption algorithm. Many schemes have been presented by researcher in this domain [69–75]. The formal definition of verifiable database with update is given below [76, 77].

Definition: A verifiable database update scheme *VDUS = (setup, Querying, verification, update)* consist of following algorithms as explained below:

- **Setup** $(PP, DB) \rightarrow (SK, PK, EDB)$: Client use his private security parameters to run the setup algorithm in order to create his secret key (in possession of client / user only), public key (that will be distributed to server and other users if required) and an encoded database (EDB).
- **Querying** $(PK, EDB, i) \rightarrow (\delta)$: server upon receiving query containing index i, query algorithm is run by the server to generate the desired output $\delta = (a, b \ldots z)$
- **Verification** $(PP, SK, i, \delta) \rightarrow (a)$: the verification algorithm is run by the client, using PP, SK, i and received output δ to verify whether the result is correct or not.
- **Update** $(SK, i, a) \rightarrow (a')$: in this update algorithm, the client using his secret key and desired index to update the data entry / value and will send it to server for storage.

3.7 Secure Outsourcing of Polynomials Data

Polynomials are generally used in many data outsourcing schemes. Due to their increased acceptability, they are often used in various applications e.g. data analysis, signal processing etc. In this regard *Benabbas et al.* Proposed an initial efficient algorithm [16] for secure polynomials outsourcing based on homomorphic encryption. Later *Gennaro et al.* [15] presented an algorithm for verifiable delegation of large polynomials. But the major common flaw of these both schemes were their inability to hide their inputs along with input polynomials and hence reveal both inputs and polynomials. Then *Ye et al.* [7] proposed a scheme that securely outsource polynomials by making use of separate polynomials for outsourcing and verification.

The verifiable calculations of various functions on outsourced data is convenient through verifiable computations [18]. The outsourcing of polynomials has been proved very effective and efficient in outsourced computation mechanisms. Verifiable computation schemes contains the private input functions and queries, so as to have them hidden and unidentified from the cloud service provider. The cloud is required to have the desired computations on required data set and return the computed results to the data asking clients. The client then verify the received results. The privacy of the data and queries are required at client end, cloud and during transit.

3.8 Applications

Polynomial Delegation and Computation has large number of applications [16, 78], few important are elaborated as under:

- **Verifiable keyboard search:** Let us assume a data file D containing n numbers of keyboards i.e. $D = w_1, w_2, \ldots, w_n$ (where w_i are the keywords). Using verifiable polynomial outsource computation, encoding / encrypting computation vector V, as having polynomial $P(.)$ of degree l such that $P(w_i) = 0$

- **Retrievability Proof:** If a remote user stores a data file D on a remote cloud server and after sometime want a proof regarding his data file D. By making use of polynomial delegation, the user encodes / encrypts the file D as a polynomial $P(x)$ of degree l and retrievability proof value as $P(r)$, with its correctness proof for any random input r by the user.

The basis of various data outsourcing protocols are delegation of polynomials due to their efficient and secure mechanism for data outsourcing and limited cost verifiability.

3.9 Verifiable Delegation / outsourcing of Polynomials

Verifiable delegation of polynomial is considered as one of the most important, efficient, practical and base point of verifiable data outsourcing schemes. A client generates a hidden large degree polynomial along with a mutated polynomial and other private parameters (PP) and secret key (SK). The mutated polynomial is sent to server for cloud computations. These polynomials then be used by the client during verification procedure for checking the correctness and authentication of the output received from server against a particular query.

3.9.1 Secure Computation Two Party Polynomial Protocol

The main aim of two party i.e. P_1 and P_2, secure computation protocol is that while evaluating a combined function i.e. $f(a, b) = f(y, z)$ by using their respective secrets S_1 and S_2, such that each party should get knowledge of its own output and gets no knowledge of other party's output / result. Such two party computation protocol was initially proposed by *Yan et al.* [79] and later this area of research gain a lot of attraction and a considerable work is done in this particular field [80–83]. A large number of protocols were designed to resolve various issues. The major issue faced by researchers in this field, that no generic protocol can address all issues collectively, rather different protocols were designed by researchers to address different problems / issues i.e. Separate protocols were designed to for special scenarios such as ranking [79, 84, 85], fairness [82, 86] and so on. Furthermore writers worked to address issues like special case polynomial manipulations i.e. polynomial multiplication and addition. A tailored protocol was developed [19] using 1 out of n oblivious transfers (OT) protocol. This secure two party multiplication protocol works by transforming the user generated two hidden polynomials multiplication into two completely other new polynomials addition such that the outcome of polynomial multiplication is same as that of polynomials addition. Same is the case with polynomial addition protocol that converts two secret polynomial addition into another new two polynomial multiplication.

3.10 Conclusion

Data outsourcing has emerged as very fruitful concept since last decade. Core concepts of verifiable data outsourcing have been discussed in this chapter being backbone of this phenomenon / paradigm. Moreover, various challenges and problems faced by verifiable data outsourcing in terms of security and deployment have also been discussed.

Chapter 4

VERIFIABLE DATA OUTSOURCING AND FILE BASED CLIENT VERIFICATION SCHEME

4.1 Introduction

In this chapter, our proposed scheme will be discussed in details containing all its algorithms (with suitable names as per their roles) along with their respective working at both client and server end. Our scheme is based on polynomials computations and their outsourcing. The following sections will be explaining the technical details of our proposed scheme.

4.2 System Model

As the client do not have full confidence over cloud or service provider, and cloud is itself considered as honest but curious. This increases the level of responsibility on the client side regarding keeping their data safe and secure from various malicious activities by attackers or even cloud sometimes. The client also verify the results to make sure that the data is safe and is not abused by the cloud service provider. However, after data uploading the cloud server is liable for data storage, data up keeping and data administration. Figure 1 shows the general illustration of our system. The proposed SVD^2OFBCV scheme comprises of five polynomial time algorithms Ω = *(Poly_Mutate, Comp_Vector, Enc, Comp_Result, Ver_Result),* such that:

- $H(y) \leftarrow$ **Poly_Mutate** $(P(y), k_1)$: is a deterministic algorithm that has to be run by the client. It takes user defined polynomial $P(y)$ and key k_1 as its input and it return a disguised polynomial $H(y)$ to the user for further operations.

- $(U) \leftarrow$ **Comp_Vector** $(P(y), k_2, k_3, g)$: is a deterministic algorithm, it requires the client defined polynomial $P(y)$, randomly selected keys k_2, k_3 and a group generator g as input to produce a vector U. This algorithm is run by client.

- $(\delta_x) \leftarrow$ **Enc** (x, k_4): is a deterministic algorithm, that takes data x and Encryption key k_4 as input to encrypt the data. The output is encryption of required data x that is to be sent to the cloud server. This algorithm is run by the client.

- $(\delta_y, V) \leftarrow$ **Comp_Result** $(H(y), \delta_x, U)$: is a deterministic algorithm, that is to be run by the Cloud server. It takes Mutated polynomial $H(y)$, Encrypted data δ_x and Computation vector U as input to produce the output value V and server's computed result δ_y over corresponding input δ_x. These values will be sent to respective client for verification process.

- $(A, B) \leftarrow$ **Ver_Result** $(\delta_x, \delta_y, k_2, k_3, g)$: is a deterministic algorithm, that takes Encrypted text δ_x, Server's computed result δ_y over δ_x, user's randomly selected values k_2, k_3 and a group generator g to produce the Verifiable results in the form of values A and B. This algorithm is run by the client.

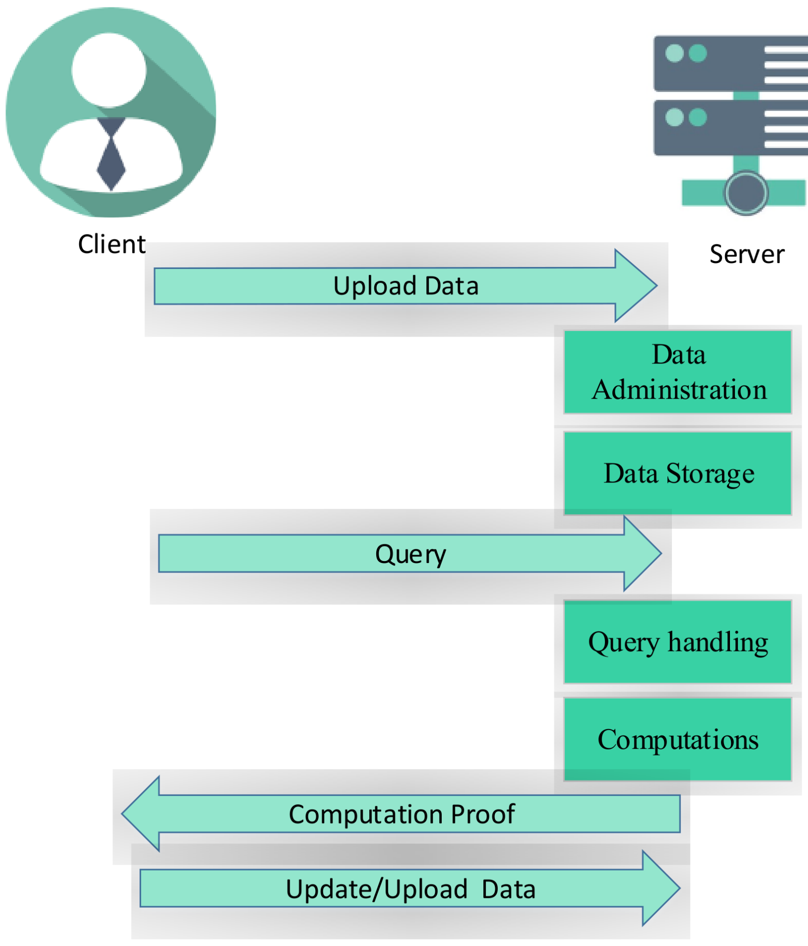

Figure 4.1: System model

4.3 System Architecture Objectives

Our system architecture objectives are as follow:

4.3.1 Data Privacy

Once data has been outsourced by a source constrained client to an external cloud server, data cannot be controlled by their owners being outside their jurisdiction. Since the data is generally sensitive and required to be properly protected. This protection is required from the cloud server itself and from all other external / internal unauthorized entities. Data privacy is related with how a piece of data / information is handled based on its importance and requirement. In this modern digital era the concept of data privacy is applied to various critical information / data like personal identifiable information (PII), personal health information, financial information etc. For different business organizations, the requirement of data privacy goes beyond its customers and employees PII. Data / information hiding (privacy) is one of the most important branches of data security that is related to proper data handing. Data privacy is also generally related to concerns like: method of sharing data with external / third parties, how and from where data is collected / stored, any other regularity restrictions etc. All the data uploaded on cloud will be fully encrypted. The encrypted data with other useful relevant information is sent to the cloud server in the form of cipher text. In this way all information along with original data is kept hidden from the cloud service provider. The cloud will not be able to access data and deduce any useful information related to original plaintext data. Moreover, the original polynomial is to be mutated so that it also remains undisclosed to cloud. Considering the data / information privacy objective in our algorithm design, ideally cloud server must not get any useful information of user's data except the data / information, for which the server is authorized i.e. queries information, computation results etc.

4.3.2 Less storage and computational cost

One of main problems faced by resource constrained clients / users is having less data storage space and huge computational cost in terms of processing and time. Such users cannot store their whole data with them and cannot have their own computations / processing being not equipped with huge resources. Clients are facilitated by allowing them to use very limited space for storing the verification credentials as the data in the form of cipher text is sent to cloud and will not share the original data. The encryption process require one time computation / processing on particular data. At the same time client generates corresponding verification credentials with respect to particular/concerned encrypted text using secret / private parameters. Resource constrained clients are required to keep / store this verification information only, upon querying the respective data from cloud server client use this particular information in verification algorithm with secret / public parameters to verify the correctness of data. This verification information is used by clients in the process of verifying the data through proofs received from the cloud. Hence less storage for verification information will be used. The client is required to have heavy computation only once during generating the polynomials / hiding polynomials. Moreover, to answer the limitations of client's processing and computation power, system provide low cost for computation involved either in sending or verifying data. The computation cost for verification is much lesser than that of computation the original function.

4.3.3 Client authenticity

A resource constrained client upload its data after encryption to the cloud server by keeping the corresponding verification data with itself. Upon requirement once the client query for a particular piece of data to the cloud server, the cloud server must verify the client querying data is a valid client. Once the data has been uploaded and certain client ask a particular data, system should give privilege to the cloud to verify the authenticity of the client (querying data). For this purpose the algorithm must have a mechanism to allow cloud server to authenticate the user. The uniqueness of system is that every time the client ask the data, client will be authenticated with unique corresponding credentials. This means that the client authentication must not be same for all data / pieces of data. For querying separate data, the client is required to get authenticated separately and with new / separate credentials in order to make sure that no unauthorized client / user impersonating as valid user for particular chunk of data. Along with the particular data verification information, client also store corresponding data user authentication information with itself. This information along with the query, sent to the server by the client.

4.3.4 Data verification

As data is generated by a resource constrained client, it is encrypted and sent to cloud for secure storage with some corresponding meta data. Resource constrained client don't have a copy of this data rather only has its verification and corresponding data client authentication information. Hence the client must have an authority to check data, whether its integrity is intact as desired (at the time of uploading data) or has been tampered. This change in data may be caused deliberately by some external / internal unauthorized entity with some malicious intent or unintentionally (may be caused due to bit lose or during transit). The client should deduce all required information from verification proofs received from remote cloud server about the data originality. The system allows its client to update any chunk of data once required and should also have an information whether data has been desirably updated earlier or not. Once the data has been updated, the new verification and corresponding data client authentication information / credentials will be generated and stored for future correspondence. If data has been updated, then its information regarding how many times this process has been done on certain chunk of data will also be require to maintain.

4.4 Proposed SVD^2OFBCV System Model

Our suggested system model will make use of polynomials. The masked polynomial will be outsourced to cloud. The original plaintext data is encrypted. All relevant information and encrypted data is combined together as final cipher text in order to send it to outsourced database. The Cipher text contains the encrypted data, verification values of corresponding data, Verification proof and the client authenticity proof. Once a valid client will ask for certain data, Cloud after verifying the data asking client, will return the desired cipher text. On receiving cipher text, client verify the required data, its security proof and updates. Figure 4.2, shows our suggested model. Notations used in our scheme are shown in Table 4.1.

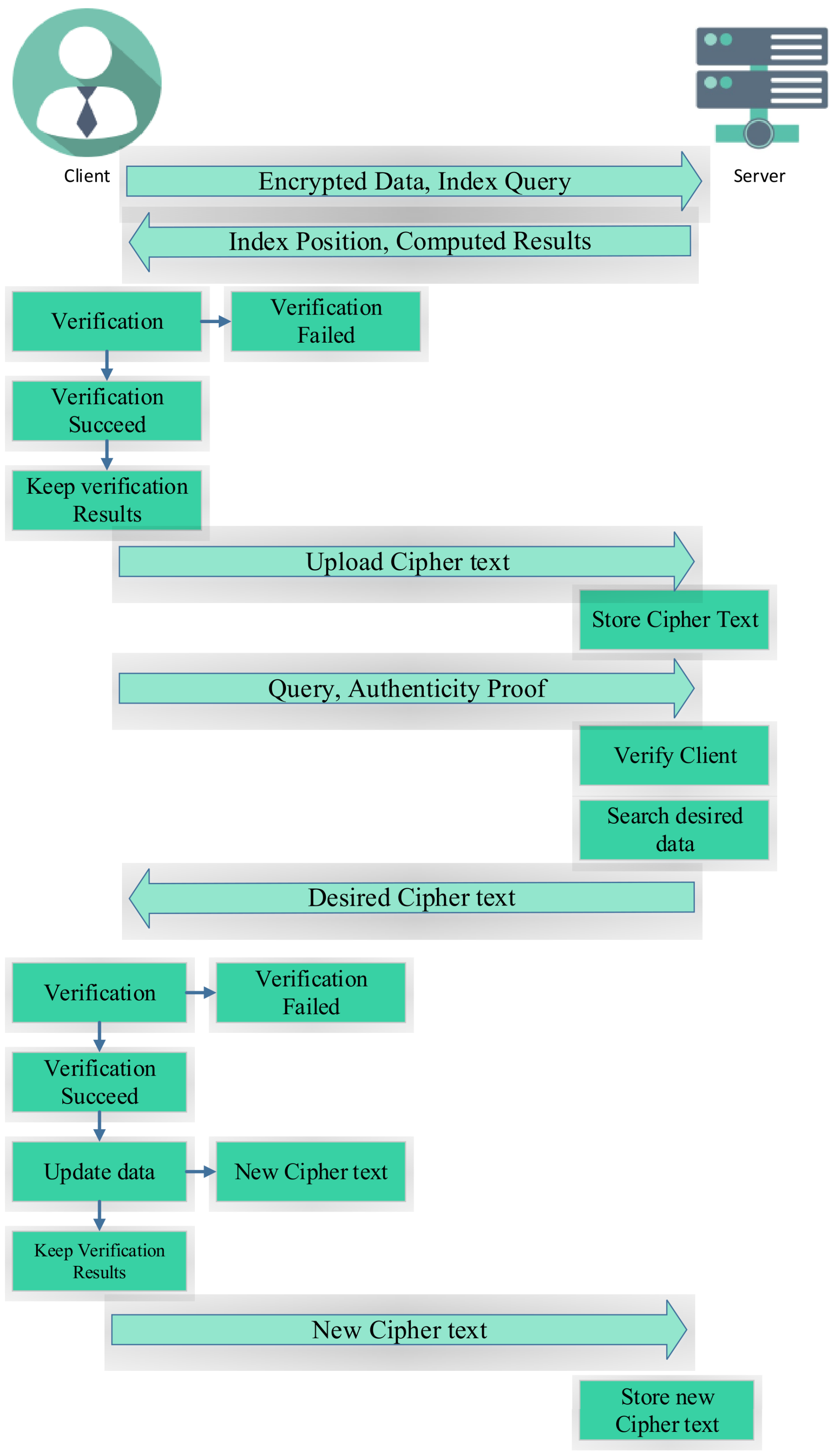

Figure 4.2: Proposed scheme

Table 4.1: Notations and their description

Notations	**Description**
$P(y)$	Client's original polynomial
$H(y)$	Disguised Polynomial
$C_0, C_1, C_2 \dots C_n$	Coefficients of client's original polynomial
$a_0, a_1, a_2 ... a_n$	Coefficients of disguised Polynomial
g	Group generator
m	Number of data updates
$U_0, U_1, U_2 ... U_n$	Client computed vector values
k_1	Polynomial conversion key
k_2, k_3	Client's randomly selected values for vector (U) generation
k_4	Data encryption key
k_5	Computation proof key
k_6	Client identity proof key
h_{k_n}	Keyed hash function
h	Hash function
δ_x	Data component cipher text
δ_y	Server's computed results over δ_x
V	Server's computed results over client computed vector U.
DP	Data proof
CP	Client proof for client identity (respective data identity)
CT	Complete cipher text uploaded on server

4.4.1 Proposed SVD^2OFBCV Scheme

Polynomial engendering. Client engenders two polynomials $P(y)$ and $P_1(y)$. Client mutates (mask) the polynomial $P(y)$, so that Cloud service provider gets no information about this polynomial, that will be used by the cloud during various computations.

$\boldsymbol{(H(y)) \leftarrow Poly\_Mutate(P(y), k_1)}$ **:**

Let, $P(y) = C_0 + C_1(y) + C_2(y)^2 + C_3(y)^3 + \ldots + C_n(y)^n$

The client randomly selects the value for k_1, k_2, k_3, k_4, k_5 and k_6 from Z_p, where p is a large prime. Client mutates polynomial $P(y)$ to $H(y)$ as follows:

$$\begin{aligned} a_0 &= C_0 k_1 \\ a_1 &= C_1 k_1^2 \\ a_2 &= C_2 k_1^3 \\ a_3 &= C_3 k_1^4 \\ &\vdots \\ a_n &= C_n k_1^{n+1} \end{aligned}$$

Hence, $H(y) = a_0 + a_1(y) + a_2(y)^2 + a_3(y)^3 + \ldots + a_n(y)^n$

$\boldsymbol{(U) \leftarrow Comp\_Vector(P(y), k_2, k_3, g)}$ **:**

Then, client compute the vector $U = (U_0, U_1, U_2, U_3, \ldots, U_n)$ as:

$$\begin{aligned} U_0 &= g^{a_0 k_2}.g^{k_3} \\ U_1 &= g^{a_1 k_2}.g^{k_3^2} \\ U_2 &= g^{a_2 k_2}.g^{k_3^3} \\ U_3 &= g^{a_3 k_2}.g^{k_3^4} \\ &\vdots \\ U_n &= g^{a_n k_2}.g^{k_3^{n+1}} \end{aligned}$$

Here the secret key will be,

$$SK = (k_1, k_2, k_3, k_4, k_5, k_6, P(y), P_1(y), g)$$

and the public key:

$$PK = (H(y), U).$$

$\boldsymbol{(\delta_x) \leftarrow Enc(x, k_4)}$ **:**

Conversion of Plaintext to Cipher text

Client encrypts the desired plaintext data x, as:

$$\delta_x = E_{k_4}(x)$$

The client sends δ_x to cloud for further processing and computing results.

$\boldsymbol{(\delta_y, V) \leftarrow Comp\_Result(H(y), \delta_x, U)}$ **:**

The cloud once receives the encrypted data δ_x , searches for its placing index i, and computes the following values:

$$\delta_y = \sum_{j=0}^{n} a_j \delta_x^j$$

and then finds,

$$V = \prod_{j=0}^{n} U_j^{\delta_x^j}$$

Then cloud sends δ_x, δ_y, V and index i to client.

$\boldsymbol{(A, B) \leftarrow Ver\_Result(\delta_x, \delta_y, k_2, k_3, g)}$ **:**

Client receiving this data, starts verification. Client will compute:

$$A = g^{k_2 \delta_y}$$

and

$$B = \prod_{j=0}^{n} g^{k_3^{j+1}\delta_x^j}$$

Client verify whether $V = AB$ or not.

Verification at cloud:

$$\delta_y = \sum_{j=0}^{n} a_j \delta_x^j$$

$$\delta_y = a_0 + a_1\delta_x + a_2\delta_x^2 + \cdots + a_n\delta_x^n$$

$$\delta_y = C_0k_1 + C_1{k_1}^2\delta_x + C_2{k_1}^3\delta_x^2 + \cdots + C_nk_1^{n+1}\delta_x^n$$

$$V = \prod_{j=0}^{n} U_j^{\delta_x^j}$$

$$V = U_0.U_1\delta_x.U_2\delta_x^2 \ldots U_n\delta_x^n$$

$$V = (g^{a_0k_2}.g^{k_3}).(g^{a_1k_2}.g^{k_3^2})\delta_x.(g^{a_2k_2}.g^{k_3^3})\delta_x^2 \ldots (g^{a_nk_2}.g^{k_3^{n+1}})^{\delta_x^n}$$

Verification at client:

$$A = g^{k_2}\delta_y$$

$$A = g^{k_2}(a_0 + a_1\delta_x + a_2\delta_x^2 + \cdots + a_n\delta_x^n)$$

$$B = \prod_{j=0}^{n} g^{k_3^{j+1}}\delta_{x^j}$$

$$B = g^{k_3}.g^{k_3^2}\delta_x.g^{k_3^3}\delta_x^2 \ldots g^{k_3^{n+1}}\delta_x^n$$

$$AB = g_3^k.g^{a_0k_2}.g^{k_3^2\delta_x}.g^{a_1k_2\delta_x}.g^{k_3^3\delta_x^2}.g^{a_2k_2\delta_x^2} \ldots g^{k_3^{n+1}\delta_{x^n}}.g^{a_nk_2\delta_{x^n}}$$

If the verification succeed, client sets data update counter $m = 1$, the next process for the client will be the generation of proofs. Client computes:

$$\phi = h_{k_5}(P_1(\delta_y))$$

where, h_k is a secure keyed hash function and then generates data proof DP as:

$$DP = h(i||m||\phi)$$

where, h is a secure hash function. Client then computes the client proof CP as:

$$CP = h_{k_6}(\phi)$$

And upload the final cipher text to the cloud, which contains the values:

$$CT = (i, CP, m, \delta_x, \delta_y, V, DP)$$

Update Data

After sometime, client wants to update data x to x' , that has been placed at index i on cloud storage. Client computes:

$$\phi = h_{k_5}(P_1(\delta_y))$$

and,

$$CP = h_{k_6}(\phi)$$

Now, the client queries for encrypted data δx that corresponds to data x which is placed at i, by sending i and CP' to cloud. Cloud upon receiving this query, matches CP' with CP placed at index location i, if verified then the cloud will return the desired cipher text.

$$CT_i = (i, CP, m, \delta_x, \delta_y, V, DP)$$

Client on receiving desired data, verifies

$V = AB$ (as described earlier)

and checks the data proof

$$DP' = h(i||m||\phi), if DP' = DP$$

Client updates data x to x' , and encrypts the updated data x' , as:

$$\delta_{x'} = E_{k_4}(x')$$

and sends to cloud. Cloud computes:

$$\delta_{y'} = \sum_{j=0}^{n} a_j \delta_{x'}^{j}$$

and then finds,

$$V' = \prod_{j=0}^{n} U_j^{\delta_{x'}^{j}}$$

Cloud will return both newly computed values to client. Client on receiving updated data computed results, verifies the computation by finding:

$$A' = g^{k_2 \delta_{y'}}$$

and

$$B' = \sum_{j=0}^{n} g^{k_3^{j+1} \delta_{x'}^{j}}$$

Now, client checks whether $V' = A' B'$ or not. Once verified, client computes the new updated proofs as:

$$\phi' = h_{k_5}(P_1(\delta_{y'}))$$

and the new data proof,

$$DP' = h(i||m+1||\phi')$$

Client then computes the corresponding client verifier as:

$$CP' = h_{k_6}(\phi')$$

finally client uploads the final updated cipher text to cloud, that contains:

$$CT' = (i, CP', m+1, \delta_{x'}, \delta_{y'}, V', DP')$$

4.5 Conclusion

This chapter discusses in detail the proposed scheme of our dynamic data outsourcing with file based client verification. The core working mechanism of our system design is polynomial computations and making use of cryptographic hashing. The system logic is modelled and tested in Intel core i7-8700k CPU @ 3.70 GHz using Python 3.8 and is found practically suitable for deployment in client server environment for verifiable data outsourcing.

Chapter 5

SECURITY AND PERFORMANCE EVALUATION

5.1 Introduction

In this chapter, we will start with the information leakage analysis and then will present the other security aspects of our propose SVD^2OFBCV scheme.

5.2 Security Analysis

We will start with the information leakage analysis and then will present the other security aspects of our propose SVD^2OFBCV scheme.

5.2.1 Leakage Profiling

It is not attainable to produce a data outsource verifiable scheme that does not leak any information outside. However the most important part is to analyze the extent, that the leaked information is how much important to the adversary and whether that leaked information is helpful in directly or indirectly extracting any relevant information. The proposed scheme leaks very less information as compared to previous schemes. Our proposed scheme information leakage analysis is as follows:

- **Information Leakage** IL_1

  This leakage is about the information exposed by the Mutated Polynomial $H(y)$ i.e, The information about the number of coefficients contained by it.

$$IL_1 = Number\ of\ Coefficients\ in\ H(y)$$

- **Information Leakage** IL_2

  This leakage point towards the information related with the relationship of index i and the relative Client proof CP during query phase (or the number of times same data asked by the client).

$$IL_2 = \begin{Bmatrix} i \leftrightarrow CP \\ Number\ \ of\ \ times\ \ same\ \ data\ \ is\ \ asked \end{Bmatrix}$$

5.2.2 Data Corruption

In case of occurrence of any deliberate (by any internal or external malicious user's activity) or un-deliberate data corruption, user will be able to verify it and will not accept any undesirable data as explained below:

Postulate: Fraudulence of cipher text must not take place.

Assumption: If a cloud service provider generates a fake cipher text

$$CT_f = (i, CP, m, \delta_x, \delta_y, V, DP)_f$$

and places it at index position i, instead of a valid cipher text

$$CT = (i, CP, m, \delta_x, \delta_y, V, DP)$$

Consider the probabilistic pseudo example shown in 1:

Algorithm 1

```
1: $(H(y)) \leftarrow$ Poly_Mutate $(P(y), k_1)$
2: $(U) \leftarrow$ Comp_Vector $(P(y), k_2, k_3, g)$
3: $(\delta_x) \leftarrow$ Enc $(x, k_4)$ Ciphered_data_sent_to_server ()
4: for 1 $\leq$ b $\leq$ p  do
5:     $(\delta_{b'})$ Enc $(\delta_{x'}$ , $k_b)$
6:     Guess_Poly $P_1$_Hidden()
7:     for 1$\leq$ t $\leq$ p do
8:         for 1$\leq$s $\leq$ p do
9:             if $(a_s(y)s! = a_i(y)^i)$ then
10:                $a_s(y)^s + +$
11:                $t + +$
12:            end if
13:        end for
14:        Guess_Poly $P_1$_Hidden()
15:        for 1$\leq$ d $\leq$ p do $\phi_d = h_{kd}(P_1(\delta_{b'})$
16:        end for
17:    end for
18: end for
```

For this, the cloud must pass the verification process at client's end with CT_f. To do this, the cloud will be requiring the encryption key k_4 , for generating $\delta_{x_f} = E_{k_4}(x_f)$, for guessing k_4, the probability will be $1/p$ (a prime p is very large, so $1/p$

will be very negligible).

In order to forge the data proof DP, with corresponding to δ_{x_f} , the cloud is required to have k_5 , again for this the probability will be $1/p$. Moreover the hidden polynomial $P_1(y)$, that is only known to the client (the probability of getting this hidden polynomial is also very unlikely) is also a requirement for producing fake value of DP.

$$\phi_f = h_{k_5}(P_1(\delta_{y_f}))$$

and then further computing:

$$DP_f = h(i||m||\phi_f)$$

So, the overall probability will be
Pr [Valid cipher text] = $(1/p)(1/p)$ (Pr of guessing $P_1(y)$).
Which is very very negligible. Hence data fraudulent cannot be possible.

5.2.3 Client Verification

Postulate: Mischievous client cannot get authenticated by the server.
Assumption: If some mischievous client approaches cloud server for data placed at index position i.
Then he is required to send the query containing index i, and the corresponding client proof CP_i. The mischievous client must pass the client verification at server end, for that he will be requiring δ_y corresponding to index i having negligible probability. Then he will be requiring $P_1(y)$, which is only known to the client. Further for valid

$$\phi = h_{k_5}(P_1(\delta_y))$$

Probability of getting k_5 is $1/p$. Therefore, the overall probability of passing the verification process at cloud for any mischievous client is very negligible. Hence the mischievous client cannot get the data from cloud.

5.2.4 Data update counter value

Postulate: Manipulation of data update counter value must be identified.
Assumption: If cloud changes the data update counter value $m_f = m + z$, the cipher text placed at index position i will be

$$CT_f = (i, CP, m_f, \delta_x, \delta_y, V, DP)_f$$

After this, the cloud server also has to change the corresponding data proof DP to DP_f , and for this the cloud has to compute

$$DP_f = h(i||m_f||\phi)$$

but here ϕ is unknown to the cloud. For getting ϕ, cloud require k_5 having probability of $1/p$ and the hidden polynomial $P_1(y)$. Hence the overall probability of manipulating the data update counter value and passing client verification process is very negligible.

5.2.5 Misplaced / Wrong Cipher text

Postulate: Misplaced/Wrong indexed cipher text must be identified.

Assumption: If the cloud has intentionally or unintentionally deleted the cipher text at index position i, and place some other cipher text at i.

Later, the client asks for original data which was actually placed at index position i. But the cloud this time returns the cipher text

$CT_j = (j, CP_j, m_j, \delta_{x_j}, \delta_{y_j}, V_j, DP_j)$, instead of

$CT_i = (i, CP_i, m_i, \delta_{x_i}, \delta_{y_i}, V_i, DP_i)$

For this, consider the probabilistic pseudo example 2:

Algorithm 2

```
1:  query($i \leftarrow CP_i$)
2:  query_response ($CT_j$)
3:  for 1 do ≤ s ≤ p $\phi_s = h_{ks}(P_1(\delta_{b'})$
4:      Guess_Poly $P_1$_Hidden()
5:      for do$1 \leq t \leq p$
6:          for do$1 \leq q \leq p$
7:              if $(aq(y)^q \neq a_i(y)^i)$ then
8:                  $a_s(y)^q + +$
9:                  $t + +$
10:                 Guess_Poly $P_1$_Hidden()
11:             end if
12:         end for
13:     end for
14: end for
```

On verifying data proof DP_j , client computes

$$DP_i = h(i||m_i||\phi_i) = h_{k_5}(P_1(\delta_{y_i}))$$

Which will be the valid value (required value), but not equals to (which is received in CT_j)

$$DP_j = h(j||m_j||\phi_j) = h_{k_5}(P_1(\delta_{y_j}))$$

Hence the cloud cannot cheat the client by sending the wrong indexed or fake cipher text.

5.3 Efficiency and Comparison

The comparisons between our proposed scheme and some other listed data outsourcing schemes are presented in table 5.1.

Table 5.1: Comparison of existing data outsourcing schemes

	Input Privacy	**Homomorphic Encryption**	**Output/Response Privacy**	**File based Client Verifiability**
Guo et al [4]	✓	✗	✓	✗
Ye et al [6]	✓	✗	✓	✗
Jun Ye et al [7]	✓	✓	✓	✗
Fiore et al [16]	✗	✓	✗	✗
Benabbas et al [17]	✗	✓	✗	✗
Our Scheme	✓	✗	✓	✓

Now we will talk about the computational cost during initial stage at client end. Client generates two polynomials, and masks (mutate) the outsourced polynomial $P(y)$, using which cloud performs computations and generate the vector U. For these two elements, the client is required to use certain complex mathematical operations. The experimental platform is constructed to analyze our scheme and its comparison with other similar model [4]. Computational cost of client in terms of Modular multiplication (MM), Modular Exponentiation (ME) and Modular Addition (MA) is shown in Table 5.2.

Table 5.2: Computation cost at client

Parameter	Masking of polynomial $P(y)$ to $H(y)$		Computation of Vector U	
	This Work	[4]	This Work	[4]
MM	n	n (with no additional coefficient)	$2n$	$2n$
ME	n	$2n$	$2n$	$2n$ (with additional exponent)
MA	0	n	0	0

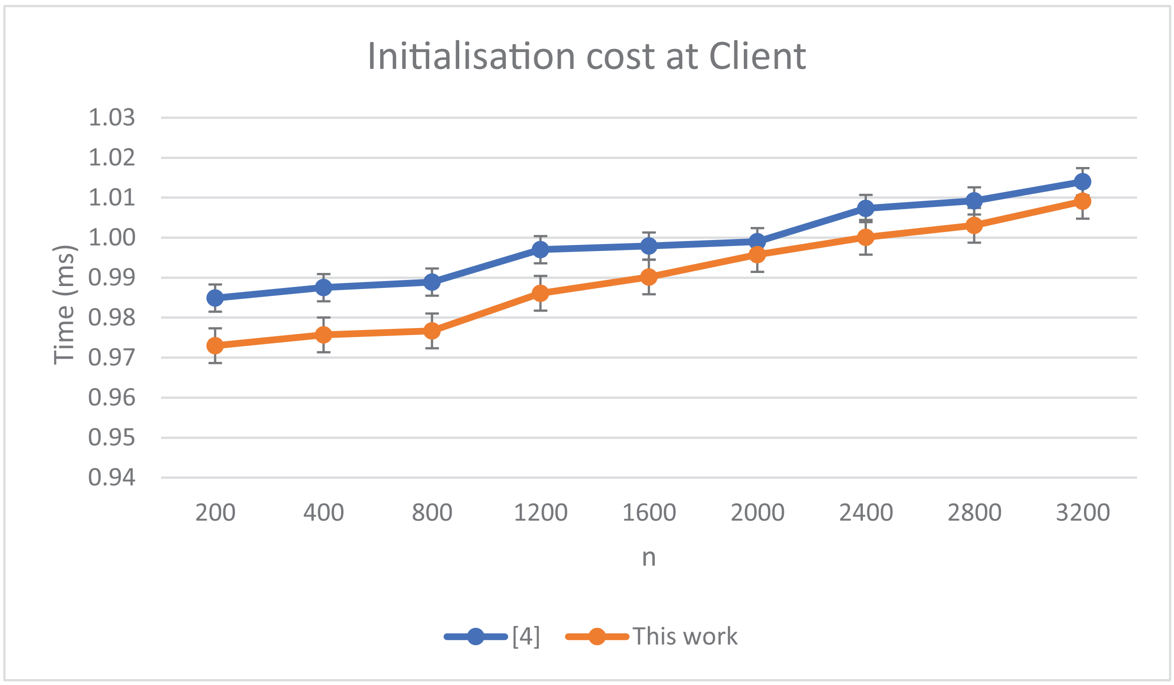

Figure 5.1: Initialization cost at client

Table 5.2 and Figure 5.1, showing the comparison result of our suggested model with the other similar scheme [4]. In Modular Multiplication, our model uses one less operation than that of the other scheme [4]. In Modular exponentiation, during polynomial masking our model reduces the operations to certain percentage along with no modular addition (MA) in terms of computation. Similarly for generating Vector U, our model uses one less exponent than that of other similar scheme [4]. The computational cost of cloud service provider in verification and computation of results is shown in Table 5.3.

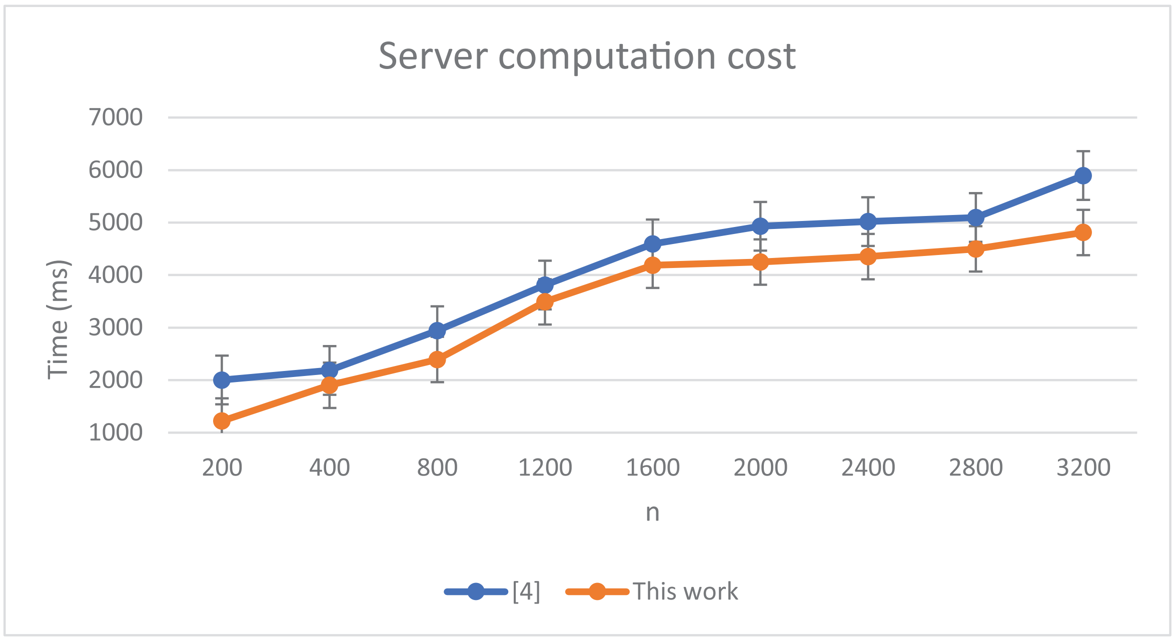

Figure 5.2: Server computation cost

Table 5.3: Computation cost at cloud

Parameter	This Work	[4]
MM	$n + (n-1)$	$3n$
ME	$(n-1) + (n-2)$	$4n$
MA	n	n

Table 5.3 and Figure 5.2 show that our suggested model uses less exponents in process of modular multiplication and exponentiation.

Table 5.4: Client verification computation cost

Parameter	This Work	[4]
MM	$n + 1$	$n + 1$
ME	$n + 1$	$n + 1$ (with additional exponent)
MA	n	n

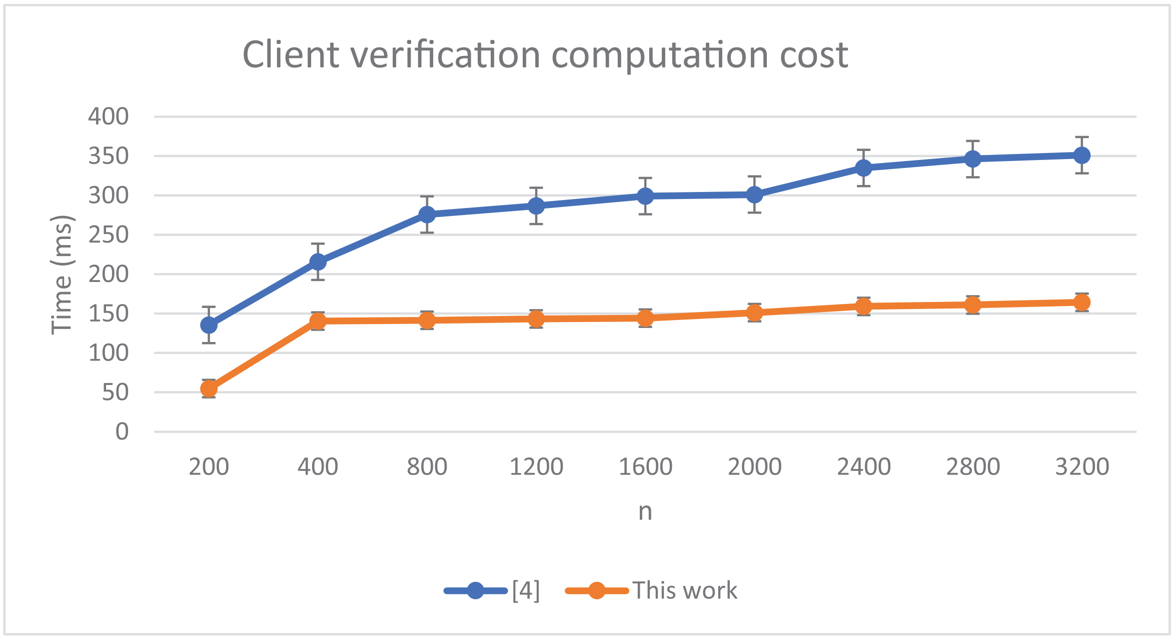

Figure 5.3: Client verification computation cost

The Comparison table 5.4 and Figure 5.3 showing the efficiency of our proposed scheme in terms of computation. The complete Computation comparison of our proposed scheme with that of [4] is shown in Figure 5.4.

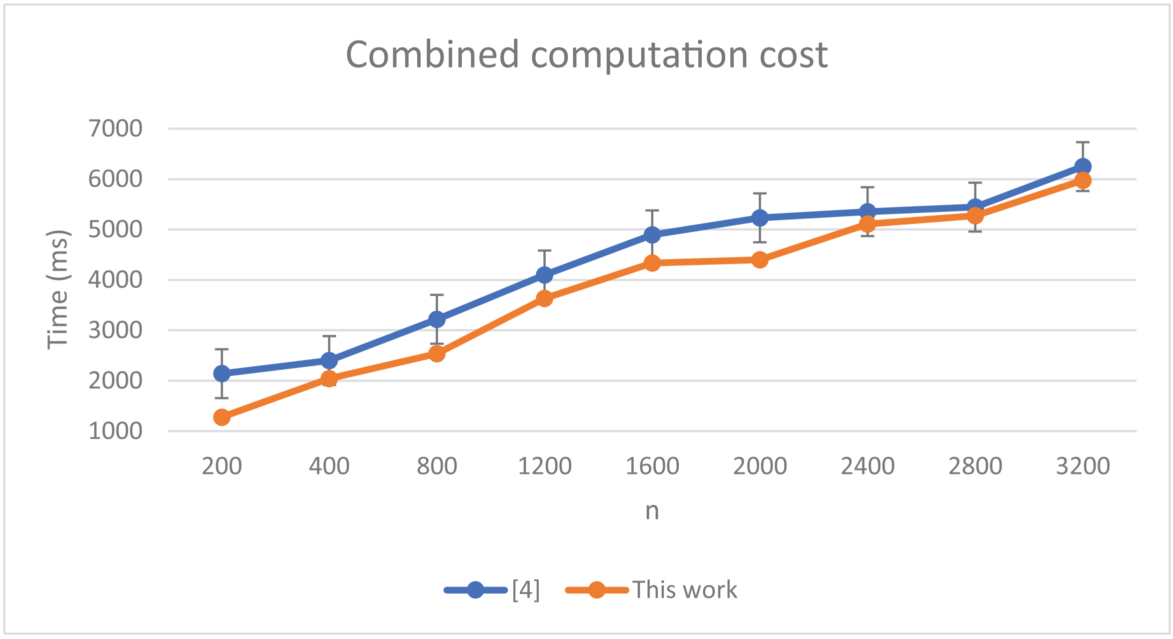

Figure 5.4: Combined computation cost

The main computational cost at client end is in initial stage, where these computations are done at start. The computed results are stored and used in verification and generations of proofs. The major part of the computations are outsourced to cloud service provider.

5.4 Conclusion

This chapter discusses various security, performance and efficiency aspects of our proposed scheme and explains different mechanisms used to achieve higher level of security and efficiency in order to make the system more secure in practical environments.

Chapter 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion

Present era has been more demanding towards clients due to a great and continuous rise in data. Data up keeping, storage and administration always hinders the smooth functioning. Clients with the limited resources, dealing with huge data is always been a significant problem. Cloud computing has introduced many positive aspects and open new horizons to answer these problems. The problem of users being resource constrained is reduced to a significant limit as cloud computing provide ease to the users by providing the facility of outsourcing their different complex computational work along with the facility of making use of huge data storage of the cloud servers. The data and computation outsourcing property for resource constrained users is proved to be very successful. However its security and computational cost are always been under consideration. The user is not required to keep huge data rather only some verification credentials.

In this paper, we suggested a model for secure and correct verification of data. The security of data at all stages has been taken care of. During initial computations and verification, the computational cost at client end is also reduced. The data will not be disclosed during uploading or updating process. Data corruption, modification, misplacing or tampering will be detected at the client end with simple verification to avoid any fraud or cheating. Moreover, in contrast to existing works in literature, the proposed scheme takes into consideration of the ownership verification, prior to entertaining any of the queries as our model gives liberty to the cloud server to verify the authenticity of the client to make sure that data is provided to its intended and valid user. The aim is to have a higher level of security, efficiency and reduced computational costs for clients. The performance evaluation construction shows its effectiveness for resource constrained users / clients.

6.2 Future Work

The proposed dynamic data verifiability scheme introduces a novel approach to explore further new capabilities of data outsourcing paradigm for enhancing overall security, efficiency, user privacy and reducing computational cost for clients. Following are the possible future research avenues:

- The proposed / existing outsource database scheme (ODB) supports only private data verifiability (only data owner can verify his data as secret key lies only with data owner). Designing a publically verifiable ODB can be an interesting future research area.
- Benchmarking the presented algorithm for deployment in corporate sector by performance analysis.
- The proposed scheme be enhanced to support probabilistic query and verifiability proof between both client and server

BIBLIOGRAPHY

[1] Shen, Jian, Jun Shen, Xiaofeng Chen, Xinyi Huang, and Willy Susilo. "An efficient public auditing protocol with novel dynamic structure for cloud data." IEEE Transactions on Information Forensics and Security 12, no. 10 (2017): 2402-2415.

[2] Chen, Xiaofeng, Jin Li, Xinyi Huang, Jianfeng Ma, and Wenjing Lou. "New publicly verifiable databases with efficient updates." IEEE Transactions on Dependable and Secure Computing 12, no. 5 (2014): 546-556.

[3] Zissis, Dimitrios, and Dimitrios Lekkas. "Addressing cloud computing security issues." Future Generation computer systems 28, no. 3 (2012): 583-592.

[4] Guo, Zhen, Hui Li, Chunjie Cao, and Zhengxi Wei. "Verifiable algorithm for outsourced database with updating." Cluster Computing 22, no. 3 (2019): 5185-5193.

[5] Ma, Xu, Jin Li, and Fangguo Zhang. "Refereed Computation Delegation of Private Sequence Comparison in Cloud Computing." IJ Network Security 17, no. 6 (2015): 743-753.

[6] Ye, Jun, Xianlin Zhou, Zheng Xu, and Yong Ding. "Verifiable outsourcing of high-degree polynomials and its application in keyword search." Intelligent Automation & Soft Computing (2017): 1-6.

[7] Ye, Jun, Haiyan Zhang, and Changyou Fu. "Verifiable delegation of polynomials." IJ Network Security 18, no. 2 (2016): 283-290.

[8] Liu, Ximeng, Baodong Qin, Robert H. Deng, and Yingjiu Li. "An efficient privacy-preserving outsourced computation over public data." IEEE Transactions on Services Computing 10, no. 5 (2015): 756-770.

[9] Wu, Tsu-Yang, Yueshan Lin, King-Hang Wang, Chien-Ming Chen, and Jeng-Shyang Pan. "Comments on Yu et al's shared data integrity verification protocol." In The Euro-China Conference on Intelligent Data Analysis and Applications, pp. 73-78. Springer, Cham, 2017.

[10] Zhou, Kai, M. H. Afifi, and Jian Ren. "ExpSOS: secure and verifiable outsourcing of exponentiation operations for mobile cloud computing." IEEE Transactions on Information Forensics and Security 12, no. 11 (2017): 2518-2531.

[11] Zhuo, Gaoqiang, Qi Jia, Linke Guo, Ming Li, and Pan Li. "Privacy-preserving verifiable set operation in big data for cloud-assisted mobile crowdsourcing." IEEE Internet of Things Journal 4, no. 2 (2016): 572-582.

[12] Gennaro, Rosario, Craig Gentry, and Bryan Parno. "Non-interactive verifiable computing: Outsourcing computation to untrusted workers." In Annual Cryptology Conference, pp. 465-482. Springer, Berlin, Heidelberg, 2010.

[13] Atallah, Mikhail J., Konstantinos N. Pantazopoulos, John R. Rice, and Eugene H. Spafford. "Secure outsourcing of scientific computations." Adv. Comput. 54, no. 6 (2001): 215-272.

[14] Yu, Jia, Kui Ren, and Cong Wang. “Enabling cloud storage auditing with verifiable outsourcing of key updates.” IEEE transactions on information forensics and security 11, no. 6 (2016): 1362-1375.

[15] Fiore, Dario, and Rosario Gennaro. “Publicly verifiable delegation of large polynomials and matrix computations, with applications.” In Proceedings of the 2012 ACM conference on Computer and communications security, pp. 501-512. 2012.

[16] Benabbas, Siavosh, Rosario Gennaro, and Yevgeniy Vahlis. “Verifiable delegation of computation over large datasets.” In Annual Cryptology Conference, pp. 111-131. Springer, Berlin, Heidelberg, 2011.

[17] Ateniese, Giuseppe, Randal Burns, Reza Curtmola, Joseph Herring, Lea Kissner, Zachary Peterson, and Dawn Song. “Provable data possession at untrusted stores.” In Proceedings of the 14th ACM conference on Computer and communications security, pp. 598-609. 2007.

[18] Wang, Qiang, Fucai Zhou, Chunyu Chen, Pengkai Xuan, and Qiyu Wu. “Secure collaborative publicly verifiable computation.” IEEE Access 5 (2017): 2479-2488.

[19] Ma, Di, Robert H. Deng, Hweehwa Pang, and Jianying Zhou. “Authenticating query results in data publishing.” In International conference on information and communications security, pp. 376-388. Springer, Berlin, Heidelberg, 2005.

[20] Song, Wei, Bing Wang, Qian Wang, Chengliang Shi, Wenjing Lou, and Zhiyong Peng. “Publicly verifiable computation of polynomials over outsourced data with multiple sources.” IEEE Transactions on Information Forensics and Security 12, no. 10 (2017): 2334-2347.

[21] Pang, HweeHwa, Arpit Jain, Krithi Ramamritham, and Kian-Lee Tan. “Verifying completeness of relational query results in data publishing.” In Proceedings of the 2005 ACM SIGMOD international conference on Management of data, pp. 407-418. 2005.

[22] Chen, Xiaofeng, Xinyi Huang, Jin Li, Jianfeng Ma, Wenjing Lou, and Duncan S. Wong. “New algorithms for secure outsourcing of large-scale systems of linear equations.” IEEE transactions on information forensics and security 10, no. 1 (2014): 69-78.

[23] Jiang, Qi, Jianfeng Ma, and Fushan Wei. “On the security of a privacy-aware authentication scheme for distributed mobile cloud computing services.” IEEE systems journal 12, no. 2 (2016): 2039-2042.

[24] Yu, Jianhua, Xueli Wang, and Wei Gao. “Improvement and applications of secure outsourcing of scientific computations.” Journal of Ambient Intelligence and Humanized Computing 6, no. 6 (2015): 763-772.

[25] Zhou, Xianlin, Yong Ding, Zhao Wang, Xiumin Li, Jun Ye, and Zheng Xu. “Secure Outsourcing Algorithm of Polynomials in Cloud Computing.” In SEKE, pp. 46-51. 2016.

[26] Ye, Jun, Zheng Xu, and Yong Ding. “Secure outsourcing of modular exponentiations in cloud and cluster computing.” Cluster Computing 19, no. 2 (2016): 811-820.

[27] Choi, Seung Geol, Jonathan Katz, Ranjit Kumaresan, and Carlos Cid. “Multi-client non-interactive verifiable computation.” In Theory of Cryptography Conference, pp. 499-518. Springer, Berlin, Heidelberg, 2013.

[28] Wang, Jianfeng, Xiaofeng Chen, Jin Li, Jiaolian Zhao, and Jian Shen. “Towards achieving flexible and verifiable search for outsourced database in cloud computing.” Future Generation Computer Systems 67 (2017): 266-275.

[29] Shen, Jian, Tianqi Zhou, Xiaofeng Chen, Jin Li, and Willy Susilo. “Anonymous and traceable group data sharing in cloud computing.” IEEE Transactions on Information Forensics and Security 13, no. 4 (2017): 912-925.

[30] Ahmad, Musheer, Shruti Khurana, Sushmita Singh, and Hamed D. AlSharari. “A simple secure hash function scheme using multiple chaotic maps.” 3D Research 8, no. 2 (2017): 13.

[31] Preneel, B., J. Vandewalle, and Bart Van Rompay. “Analysis and design of cryptographic hash functions, Mac Algorithms and block ciphers.” (2004).

[32] Merkle, Ralph Charles. “Secrecy, authentication, and public key systems.” Stanford University, 1979.

[33] Lin, Zhuosheng, Simin Yu, and Jinhu Lu. “A novel approach for constructing one-way hash function based on a message block controlled 8D hyperchaotic map.” International Journal of Bifurcation and Chaos 27, no. 07 (2017): 1750106.

[34] Akhavan A, Samsudin A, Akhshani A (2013) A novel parallel hash function based on 3D chaotic map. EURASIP Journal on Advances in Signal Processing 1:1-12

[35] Akhavan, Amir, Azman Samsudin, and Afshin Akhshani. “A novel parallel hash function based on 3D chaotic map.” EURASIP Journal on Advances in Signal Processing 2013, no. 1 (2013): 126.

[36] Yang, Yu-Guang, Peng Xu, Rui Yang, Yi-Hua Zhou, and Wei-Min Shi. “Quantum Hash function and its application to privacy amplification in quantum key distribution, pseudo-random number generation and image encryption.” Scientific reports 6 (2016): 19788.

[37] Wazid, Mohammad, Ashok Kumar Das, Rasheed Hussain, Giancarlo Succi, and Joel JPC Rodrigues. “Authentication in cloud-driven IoT-based big data environment: Survey and outlook.” Journal of Systems Architecture 97 (2019): 185-196.

[38] Stallings, William. “Cryptography and network security”, 4/E. Pearson Education India, 2006.

[39] Haber, Stuart, and W. Stornetta. “How to Time-Stamp a Digital Document, Crypto’90, LNCS 537.” (1991): 437-455.

[40] Sharma, Aakriti, Bright Keshwani, and Pankaj Dadheech. “Authentication issues and techniques in cloud computing security: a review.” In Proceedings of International Conference on Sustainable Computing in Science, Technology and Management (SUSCOM), Amity University Rajasthan, Jaipur-India. 2019.

[41] Andreeva, Elena, Gregory Neven, Bart Preneel, and Thomas Shrimpton. “Seven-property-preserving iterated hashing: ROX.” In International Conference on the Theory and Application of Cryptology and Information Security, pp. 130-146. Springer, Berlin, Heidelberg, 2007.

[42] Kim, Hangi, Do-won Kim, Okyeon Yi, and Jongsung Kim. “Cryptanalysis of hash functions based on blockciphers suitable for IoT service platform security.” Multimedia Tools and Applications 78, no. 3 (2019): 3107-3130.

[43] Lin, Zhuosheng, Simin Yu, and Jinhu Lu. “A novel approach for constructing one-way hash function based on a message block controlled 8D hyperchaotic map.” International Journal of Bifurcation and Chaos 27, no. 07 (2017): 1750106.

[44] Dodis, Yevgeniy, Thomas Ristenpart, and Thomas Shrimpton. “Salvaging Merkle-Damgård for practical applications.” In Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 371-388. Springer, Berlin, Heidelberg, 2009.

[45] Hong, Deukjo, Dong-Chan Kim, Daesung Kwon, and Jongsung Kim. “Improved preimage attacks on hash modes of 8-round AES-256.” Multimedia Tools and Applications 75, no. 22 (2016): 14525-14539.

[46] Liu, Hongjun, Abdurahman Kadir, Xiaobo Sun, and Yanling Li. “Chaos based adaptive double-image encryption scheme using hash function and S-boxes.” Multimedia Tools and Applications 77, no. 1 (2018): 1391-1407.

[47] Biham, Eli. “New techniques for cryptanalysis of hash functions and improved attacks on Snefru.” In International Workshop on Fast Software Encryption, pp. 444-461. Springer, Berlin, Heidelberg, 2008.

[48] Yiu, Man Lung, Yimin Lin, and Kyriakos Mouratidis. “Efficient verification of shortest path search via authenticated hints.” In 2010 IEEE 26th International Conference on Data Engineering (ICDE 2010), pp. 237-248. IEEE, 2010.

[49] Wang, Qiang, Fucai Zhou, Chunyu Chen, Pengkai Xuan, and Qiyu Wu. “Secure collaborative publicly verifiable computation.” IEEE Access 5 (2017): 2479-2488.

[50] Chen, Xiaofeng, Jin Li, Jianfeng Ma, Qiang Tang, and Wenjing Lou. “New algorithms for secure outsourcing of modular exponentiations.” IEEE Transactions on Parallel and Distributed Systems 25, no. 9 (2013): 2386-2396.

[51] Fu, Zhangjie, Xingming Sun, Qi Liu, Lu Zhou, and Jiangang Shu. “Achieving efficient cloud search services: multi-keyword ranked search over encrypted cloud data supporting parallel computing.” IEICE Transactions on Communications 98, no. 1 (2015): 190-200.

[52] Yang, Kan, and Xiaohua Jia. “Data storage auditing service in cloud computing: challenges, methods and opportunities.” World Wide Web 15, no. 4 (2012): 409-428.

[53] Thiranant, Non, Mangal Sain, and Hoon Jae Lee. “A design of security framework for data privacy in e-health system using web service.” In 16th International Conference on Advanced Communication Technology, pp. 40-43. IEEE, 2014.

[54] Xinyue Cao, Zhangjie Fu, Xingming Sun, “A Privacy-Preserving Outsourcing Data Storage Scheme with Fragile Digital Watermarking-Based Data Auditing”, Journal of Electrical and Computer Engineering, vol. 2016, Article ID 3219042, 1-7.

[55] IM, Yusuf, Mustafa Kaiiali, Adib Habbal, A. S. Wazan, and Auwal SI. “A secure data outsourcing scheme based on Asmuth–Bloom secret sharing.” Enterprise Information Systems, 10:9 (2016), 1001-1023.

[56] Hacigumus, Hakan, Bala Iyer, and Sharad Mehrotra. “Providing database as a service.” In Proceedings 18th International Conference on Data Engineering, pp. 29-38. IEEE, 2002.

[57] Hao, Zhuo, Sheng Zhong, and Nenghai Yu. “A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability.” IEEE transactions on Knowledge and Data Engineering 23, no. 9 (2011): 1432-1437.

[58] Li, Yantao. “Collision analysis and improvement of a hash function based on chaotic tent map.” Optik 127, no. 10 (2016): 4484-4489.

[59] Wazid, Mohammad, Ashok Kumar Das, Vanga Odelu, Neeraj Kumar, and Willy Susilo. “Secure remote user authenticated key establishment protocol for smart home environment.” IEEE Transactions on Dependable and Secure Computing (2017).

[60] Li, Feifei, Marios Hadjieleftheriou, George Kollios, and Leonid Reyzin. “Dynamic authenticated index structures for outsourced databases.” In Proceedings of the 2006 ACM SIGMOD international conference on Management of data, pp. 121-132. 2006.

[61] Challa, Sravani, Mohammad Wazid, Ashok Kumar Das, Neeraj Kumar, Alavalapati Goutham Reddy, Eun-Jun Yoon, and Kee-Young Yoo. “Secure signature-based authenticated key establishment scheme for future IoT applications.” IEEE Access 5 (2017): 3028-3043.

[62] Teh, Je Sen, Azman Samsudin, and Amir Akhavan. “Parallel chaotic hash function based on the shuffle-exchange network.” Nonlinear Dynamics 81, no. 3 (2015): 1067-1079.

[63] Mouratidis, Kyriakos, Dimitris Sacharidis, and Hweehwa Pang. “Partially materialized digest scheme: an efficient verification method for outsourced databases.” The VLDB Journal 18, no. 1 (2009): 363-381.

[64] Lin, Zhuosheng, Christophe Guyeux, Simin Yu, Qianxue Wang, and Shuting Cai. “On the use of chaotic iterations to design keyed hash function.” Cluster Computing (2019): 1-15.

[65] Jiteurtragool, N., P. Ketthong, C. Wannaboon, and W. San-Um. “A topologically simple keyed hash function based on circular chaotic sinusoidal map network.” In 2013 15th International Conference on Advanced Communications Technology (ICACT), pp. 1089-1094. IEEE, 2013.

[66] Narasimha, Maithili, and Gene Tsudik. “DSAC: integrity for outsourced databases with signature aggregation and chaining.” In Proceedings of the 14th ACM international conference on Information and knowledge management, pp. 235-236. 2005.

[67] Pang, HweeHwa, Arpit Jain, Krithi Ramamritham, and Kian-Lee Tan. "Verifying completeness of relational query results in data publishing." In Proceedings of the 2005 ACM SIGMOD international conference on Management of data, pp. 407-418. 2005.

[68] Lin, Zhuosheng, Christophe Guyeuxy, Qianxue Wang, and Simin Yu. "Diffusion and confusion of chaotic iteration based hash functions." In 2016 IEEE Intl Conference on Computational Science and Engineering (CSE) and IEEE Intl Conference on Embedded and Ubiquitous Computing (EUC) and 15th Intl Symposium on Distributed Computing and Applications for Business Engineering (DCABES), pp. 444-447. IEEE, 2016.

[69] Chen, Xiaofeng, Hui Li, Jin Li, Qian Wang, Xinyi Huang, Willy Susilo, and Yang Xiang. "Publicly Verifiable Databases with All Efficient Updating Operations." IEEE Transactions on Knowledge and Data Engineering (2020).

[70] Chen, Xiaofeng, Jin Li, Xinyi Huang, Jianfeng Ma, and Wenjing Lou. "New publicly verifiable databases with efficient updates." IEEE Transactions on Dependable and Secure Computing 12, no. 5 (2014): 546-556.

[71] Chen, Xiaofeng, Jin Li, Jian Weng, Jianfeng Ma, and Wenjing Lou. "Verifiable computation over large database with incremental updates." In: Kutyłowski M., Vaidya J. (eds) Computer Security - ESORICS 2014. ESORICS 2014. Lecture Notes in Computer Science, vol 8712.

[72] Zhang, Liang Feng, and Reihaneh Safavi-Naini. "Verifiable delegation of computations with storage-verification trade-off." In European symposium on research in computer security, pp. 112-129. Springer, Cham, 2014.

[73] Chen, Xiaofeng, Jin Li, Jian Weng, Jianfeng Ma, and Wenjing Lou. "Verifiable computation over large database with incremental updates." IEEE transactions on Computers 65, no. 10 (2015): 3184-3195.

[74] Miao, Meixia, Jianfeng Wang, Jianfeng Ma, and Willy Susilo. "Publicly verifiable databases with efficient insertion/deletion operations." Journal of Computer and System Sciences 86 (2017): 49-58.

[75] Miao, Meixia, Jianfeng Ma, Xinyi Huang, and Qian Wang. "Efficient verifiable databases with insertion/deletion operations from delegating polynomial functions." IEEE Transactions on Information Forensics and Security 13, no. 2 (2017): 511-520.

[76] Benabbas, Siavosh, Rosario Gennaro, and Yevgeniy Vahlis. "Verifiable delegation of computation over large datasets." In Annual Cryptology Conference, pp. 111-131. Springer, Berlin, Heidelberg, 2011.

[77] Catalano, Dario, and Dario Fiore. "Vector commitments and their applications." In International Workshop on Public Key Cryptography, pp. 55-72. Springer, Berlin, Heidelberg, 2013.

[78] Benabbas, Siavosh, Rosario Gennaro, and Yevgeniy Vahlis. "Verifiable delegation of computation over large datasets." In Annual Cryptology Conference, pp. 111-131. Springer, Berlin, Heidelberg, 2011.

[79] Yao, Andrew C. ”Protocols for secure computations.” In 23rd annual symposium on foundations of computer science (sfcs 1982), pp. 160-164. IEEE, 1982.

[80] Nielsen, Jesper Buus, Peter Sebastian Nordholt, Claudio Orlandi, and Sai Sheshank Burra. ”A new approach to practical active-secure two-party computation.” In Annual Cryptology Conference, pp. 681-700. Springer, Berlin, Heidelberg, 2012.

[81] Xia, Feng, Wei Wang, Teshome Megersa Bekele, and Huan Liu. “Big scholarly data: A survey.” IEEE Transactions on Big Data 3, no. 1 (2017): 18-35.

[82] Shen, Jian, Tianqi Zhou, Debiao He, Yuexin Zhang, Xingming Sun, and Yang Xiang. ”Block design-based key agreement for group data sharing in cloud computing.” IEEE Transactions on Dependable and Secure Computing (2017).

[83] Huang, Yan, David Evans, Jonathan Katz, and Lior Malka. “Faster secure two-party computation using garbled circuits.” In USENIX Security Symposium, vol. 201, no. 1, pp. 331-335. 2011..

[84] Li, Jin, Yinghui Zhang, Xiaofeng Chen, and Yang Xiang. “Secure attribute-based data sharing for resource-limited users in cloud computing.” Computers & Security 72 (2018): 1-12.

[85] Li, Ping, Jin Li, Zhengan Huang, Chong-Zhi Gao, Wen-Bin Chen, and Kai Chen. “Privacy-preserving outsourced classification in cloud computing.” Cluster Computing 21, no. 1 (2018): 277-286.

[86] Sookhak, Mehdi. “Dynamic remote data auditing for securing big data storage in cloud computing” PhD diss., University of Malaya, 2015.