AUTHENTICATED CLIENT PROVABLE DATA POSSESSION (AC-PDP) SECURITY SYSTEM FOR CLOUD COMPUTING

by

Memoona Javeria Anwar

2009-NUST-MSPHD-CSE (E)-03

MS-09 (SE)

Submitted to the Department of Computer Engineering in fulfillment of the requirements for the degree of

MASTER OF SCIENCE
in
SOFTWARE ENGINEERING

Thesis Supervisor

Prof Dr Muhammad Younus Javed

College of Electrical & Mechanical Engineering

National University of Sciences & Technology

2012

DECLARATION

I hereby declare that I have developed this thesis entirely on the basis of my personal efforts under the guidance of my supervisor Prof Dr Muhammad Younus Javed. All the sources used in this thesis have been cited and the contents of this thesis have not been plagiarized. No portion of the work presented in this thesis has been submitted in support of any application for any other degree of qualification to this or any other university or institute of learning.

_____

Memoona Javeria Anwar

ACKNOWLEDGMENTS

Innumerable words of praise and thanks to Allah, the Almighty, and the Creator of the universe for carving the path for me and always helping me out in the best possible way. Without His Will and Mercy, I would not have been able to accomplish this milestone.
I would like to thank my **mother**; it was because of her encouragement and prayers that I have achieved this milestone.

I offer my sincerest gratitude to my supervisor, **Dr. Muhammad Younus Javed**, who has supported me throughout my thesis with his guidance and knowledge whilst allowing me the room to work in my own way. I attribute the level of my master's degree to his encouragement and effort and without him this thesis, too, would not have been completed or written. One simply could not wish for a better or friendlier supervisor. I would like to express my deep and sincere gratitude to my committee members, **Dr. Shoab Ahmad Khan, Dr. Aasia Khanum,** and **Dr. Almas Anjum.**

In my daily work, I have been blessed with a friendly and cheerful group of fellow students. I wish to express my warm and sincere thanks to all of them. I also owe my thanks to my husband, **Inam Gull**, without his encouragement and understanding it would have been impossible for me to finish this work. My special gratitude is due for my brother **Usman Ghani** who helped and consoled me all through my MS study duration and this thesis.

In the memory of my loving father…

ABSTRACT

Extensive research has been conducted in the field of cloud computing. Both the academia and researchers have envisioned a broad range of applications for cloud networks. Success measurement of emerging e-technological applications is gauged by their effectiveness; ease of use and most notably by the gradation of information security and control. Concerns pertaining to information security ascend as the world switches towards applications that run beyond the designated firewall/private domain and move closer towards the public domain.

This thesis focuses on the design, implementation and analysis of a security protocol that fulfills the requirements of confidentiality, authentication and integrity in cloud environment. The proposed ***Authenticated Client Provable Data Possession (AC-PDP)*** meets requirements of high level security without compromising client authentication. AC-PDP architecture is based on the asymmetric encryption schemes (i.e. RSA and ElGamal). The designed protocol must consume minimum amount of resources while providing high levels of security in a quick and optimal fashion.

AC-PDP has been optimized by first eradicating the authentication overhead that incurs because of performing separate user authentication mechanism. The digital signature has been employed in such a way that it removes the need of running an independent authentication routine. In order to evaluate the system, several experiments have been carried out with respect to encryption/ decryption of various data blocks and integrity of file stored on server. Evaluation of AC-PDP has been done by conducting a comprehensive efficiency analysis of the proposed architecture. Furthermore, the result of execution time and memory usage for AC-PDP has been compared to its existing PDP scheme.

To fully test AC-PDP it has been evaluated from various aspects like processing time, space utilization and security provision. The results have been obtained by implementing a client/server environment in .Net. Extensive testing of AC-PDP has shown that this new cloud security system is complete and ensures almost complete security over cloud network, in terms of integrity. AC-PDP has been evaluated in comparison with other

protocols like PDP. The results demonstrate that AC-PDP has all the properties that are required by a highly acknowledged security protocol in cloud networks. The increased security offered by AC-PDP makes it a better choice as compared to the other PDP methods.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CSP	Cloud Service Providers
TPA	Third Party Auditor
PDP	Provable Data Possession
POR	Proof of Retrievability
DPDP	Dynamic Provable Data Possession
AC-PDP	Authenticated Client Provable Data Possession
HVT	Homo-Morphic Verifiable Tags
MsgD	Message-Digest
SHA	Secure Hashing Algorithm
chal	Challenge
Sig	Signatures
Tm	Homo-morphic verifiable tags
V	Proof
CC	Cloud Computing

CHAPTER 1

1. INTRODUCTION

Because of the immense opportunities that Cloud computing has to offer, it is one of the hot topics amongst the technology gurus and the reviewers. The research on market shares by a reputed firm highlights that the total market captured by Cloud computing was $16.5 billion in the year 2008 and is expected to be in excess of $42.5 billion by the end of the year 2012 (Gleeson, 2009). In terms of savings in expenditures, there is a 3 to 5 times savings in commercial applications while greater than 5 times savings for user-based applications (Lynch, 2008). As per a statement issued on their website, The Cloud environment is sure to dominate and it will be "*not less significant in comparison to e-business*" (Gartner, 2008).

Success measurement of emerging e-technological applications is gauged by their effectiveness; ease of use and most notably by the gradation of information security and control. Concerns pertaining to information security ascend as the world switches towards applications that run beyond the designated firewall/private domain and move closer towards the public domain.

Amid the most prominent e-technologies of the modern day, Cloud Computing has changed the manner in which IT architectural solutions are put forward, by shifting towards the theme of virtualization, be it in terms of data storage, infrastructure or software. While on one hand, the 'cloud' offers immense benefits for the users, yet on the

other hand, information breach / in-security is the foremost challenge that outweighs its colossal success factors.

1.1 PROBLEM OVERVIEW

The notion of cloud computing proposes a databank of e-resources that can be assigned or de-assigned at run time and can be used as a service. The core reason for cloud's success lies in the fiscal benefit it offers in term of initial expense (CapEx) and running expense(OpEx). However, the hindrance in its materialization lies in the challenges that await disentanglement. The predominant problems associated with cloud computing pertain to cloud security and appropriate implementation of cloud over the network. Security and trust issues top this list, because the data of the client that is released on the cloud is out of the shielded zone of the client. The central issue amongst the cloud computing security concerns is that of security of the data and pertains to data confidentiality, integrity and availability. Data confidentiality means that only legal persons can use the data. To ensure that only allowed list of people can manipulate the data is an enormous task, for which people have proposed different security models.

There are several data security models which deal with information handling; security and safe custody once it enters the public domain; each of them having their own benefits and concerns. Some of them pertain to aspects on the server where the application is hosted or the client where the data is accessed, using trusted and untrusted profiles.

1.2 RESEARCH AIM AND OBJECTIVES

Having this background in mind, the overall aim of my research is to study the various facets of security over the cloud and to provide a scalable and efficient mechanism to

ensure client's authenticity in cloud network along with the other attributes. To achieve this aim, I have set the following objectives for my research:

a. To get acquainted about the technicalities and subsequent issues that are generated as a result of employing cloud based services, with focus upon untrusted servers and untrusted clients.

b. To gain in-depth knowledge about the mechanism named 'provable data possession' (PDP).

c. To present a PDP prototype which will not need a separate user authentication procedure in addition to security routine and would allow a client go through the authentication of the originality of data without its repossession.

d. To take into account the consideration of untrusted client in addition to the notion of untrusted server.

e. To extend PDP with addition of authentication for untrusted clients and aligning it with goal of achieving security over the cloud

f. To prove that this improved PDP model would be efficient and complete compared to previous works on the subject, by implementing the model in C#.Net and comparing the results.

The research question that is designated to be answered at the culmination of dissertation would be:

"What is one complete cloud computing security model that takes into account both untrusted client and server"

1.3 THESIS OUTLINE

The structure of thesis is developed in a very logical pattern for an easy understanding of research case study. This thesis is structured into six chapters and document is organized as follows:

Chapter 1: It consists of general introduction, research aim and objectives, strategy, approach and thesis layout.

Chapter 2: This chapter provides literature review covering cloud computing architecture, its characteristics, benefit and threats. This will be succeeded by narration of the security mechanisms already available.

Chapter 3: Pertains to this research specifically on ***Provable Data Possession (PDP)***, a cloud security mechanism. This would also cover the working and extensions of PDP.

Chapter 4: Implementation aspects of proposed version of PDP

Chapter 5: Study of results obtained through existing PDP models and proposed PDP model.

Chapter 6: Conclusion and future work is the last chapter of the thesis in which overall summary of the thesis is given. Certain limitations and benefits of aimed work have also been discussed and finally areas for future work are identified and talked about.

1.4 SUMMARY

Introduction to cloud computing has been detailed in this chapter. Incidents that have happened in history of cloud computing are described. This thesis is about highlighting cloud computing security issues and providing an optimal solution to them. A brief elaboration of problem overview referring to this dissertation has been given in this chapter. Aims and objectives that make a baseline of this research have been narrated and a remit thesis outline has also been communicated.

Chapter 2

BACKGROUND

2.1 INTRODUCTION

As an introduction to cloud computing, following questions have been answered in this chapter:

- ***a.*** *What is cloud computing and how it works?*
- ***b.*** *What are the benefits of using cloud?*
- ***c.*** *What are characteristics of cloud?*
- ***d.*** *What does cloud computing mean to different people?*
- ***e.*** *What are security challenges in cloud computing?*

2.2 CLOUD COMPUTING AND HOW IT WORKS

The concept of computing over the cloud is based on its usage as a service and not as a standalone product. In this concept, shared assets and files are provided over the cloud based on the concept of billed-per-use service. It provides services such as computing, software-based applications, access to data and its storage in such a manner that the end user need not know how and where the data is being kept and used from. The data storage is usually in some far off country where storage is cheaper, while its access is through the internet based browser, which may be in the form of a desktop or tablet application. In the endeavor to capture more and more market, the developers and hosts of cloud oriented services try to provide the same or in many cases even better enactment than a user-desktop-based application.

The term "**Computing**" is defined as a simulated grouping of assets which entails the provision of computational services via the internet. The result is the provision of an environment that is able to initially commission, then assign / re-assign a resource at run-time, with the additional facility of observing the utilization of all resources over time.

2.3 THE COMPUTING WORLD BEFORE CLOUD

The computer age, though still evolving, has seen many drastic changes in the computer architectures. In the beginning, mainframes were predicted to be the future of computing. Some of these machines may still be in use today. Later, the same function was achieved by using clustered systems, however lately, the trend has shifted towards smaller personal computers (clients and servers) tied together to engender the so called cloud computing system, also known as the '**Cloud**'. The additional advantage of the 'cloud' is that it is more scalable, bears fault-tolerant services and enhanced performance. When necessitated, the cloud can provide unbounded computing resources being highly scalable, eliminating the need for cloud service providers to plan forward on hardware provisioning. The up-front obligation of building heavy structures can thus be disregarded and they can now start from small companies to which hardware resources are added only when necessitated.

Global giants like **Google** and **Microsoft** use cloud computing systems to enhance provision of their services to a larger audience. Business applications have always been complex and costly to build and the volume and diversity of hardware and software required to run them is arduous. An entire team of experts has to be engaged to install, configure, test, run, secure, and subsequently update these applications. Even the leading

companies with the finest IT departments cannot get the optimum application integration when these run into dozens or hundreds, while small and medium sized businesses do not stand a chance.

2.4 BENEFITS OF CLOUD

Using the services of the cloud means that we do not have to worry about hardware or software related issues, as they are taken care of by experienced service providers. As the infrastructure is shared, it works like a utility service, whereby we have to pay only for the service that we use. Furthermore, the upgrades are automatic and scaling up or even scaling down is easy. Therefore, cloud-based application solutions can be commissioned in very less time and with very less cost. With a cloud based application, it is as simple as opening the browser, logging in, customizing the application and using it.

To reap the benefits of the cloud, more and more companies are switching their large modules on cloud based servers. Cloud computing is a step forward to grid computing and clustering.

2.5 BUSINESS BENEFITS

A few benefits of building applications using cloud architecture are:

a. **Virtually negligible upfront infrastructure venture:** Building a large scale software solution requires setting up of huge back-end resources like hardware, resource management and teams for operationalization. In addition to that the bigger the system under design, the more cumbersome is the management's

approval process. Using utility style computing, these costs are comparably negligible.

b. **Just-in-time Infrastructure:** In the pre-cloud world, either the investment on software solution was huge as compared to the success of the company, or conversely, the software and hardware resources could not cope up with the success and pace of the company. The cloud environment allows us to forget about pre-procurement of large scale systems and obtaining just-in-time infrastructure instead. The risk factor in using cloud based solutions is quite low, as growth of business dictates scale of the solution. Furthermore, as per dictates of restructuring of the company, the infrastructure can be disposed of in a snap.

c. **Resource utilization:** Be it procurement of hardware resources upon running out of capacity or conversely the utilization of infrastructure when the system is under-consumed, the system administrators always have some worries on their desk. Cloud environment can prove a relief when it comes to managing resources by requesting or relinquishing services as needed.

d. **Costing based on actual usage:** The fundamental difference between desktop and web applications is the concept of utility based pricing, whereby, in case of web applications the customer is billed only for the substructure he or she accesses, not the entire infrastructure.

e. **Parallelization:** Since the cloud infrastructure offers the capability of acquiring additional resources according to the rules of business, therefore users can take advantage of this by utilizing the concept of parallelization. As an example, if a process takes 10 hours to complete, using the cloud's services, 10 parallel instances of the service can be used to complete the process in unit time.

2.6 CHARACTERISTICS OF CLOUD COMPUTING

a. **Large-scale:** Software giant **Google** has a very large cloud, spanning over more than a million servers. Next in line are names such as **Microsoft** and **Yahoo** having that number in the hundreds of thousands, while smaller entities; like an enterprise, have hundreds of them.

b. **Globalization:** As the services are cloud based, therefore users can access the services around the globe by just using a notebook and internet connection. In addition to that, the data can be shared when required. Some situations make this way of working a necessity, where work that has to be done is scattered over different locations around the globe.

c. **Reliability*:*** Data over the cloud has properties of being multi-transcript and fault tolerant, while isomorphism exchangeable computation mode is prevalent. In a nutshell, using a cloud based service is more reliable than a stand-alone PC.

d. **Flexibility:** Different types of applications can reside within a cloud and in a different scenario; an amalgamation of application can be supported by the cloud.

e. **Extendibility**: As mentioned in paragraphs above, the size of the cloud can grow dynamically according to the requirement of the application.

f. **Charged as a service:** A cloud is virtually a pool of resources from which the client can choose the amount of service to be used or engaged. The clients are billed according to their usage

2.7 THE ARCHITECTURE OF CLOUD

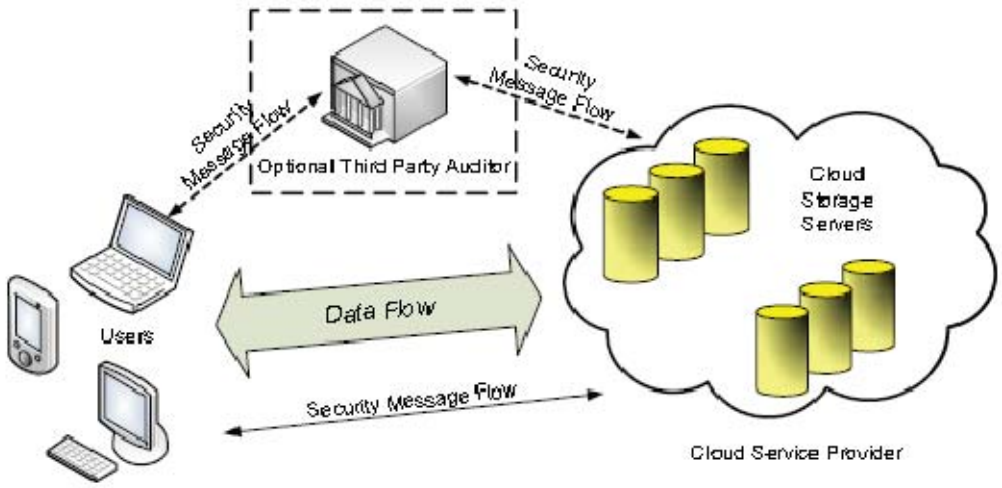


Figure 2.1: Network Architecture of Cloud Data Storage

Figure 2.1 shows the network storage mechanism over the cloud resource. There are three main entities:

a. **User:** They consist of individual clients or organizations. They use the cloud for storage of data and performing computations

b. **Cloud Service Provider (CSP):** The vendor of distributed system who has expertise in constructing and running distributed servers over the cloud. They operate live computing systems over the cloud

c. **Third-Party Checker or Auditor (TPA):** An independent body that possesses the requisite capability and skills to examine the data security over the cloud upon the client's request.

Data storage by a client takes place through a cloud service provider, who in-turn stores the client data on a set of distributed servers which are configured to run simultaneously in harmony with each other. Different data correction codes, such as erasure correction, are applied to data, that is important for the client, to withstand crashes of the server. At the user end, the machine interrelates with the CSP to store and retrieve data.

2.8 SERVICE MODELS

In the cloud's term, whenever we refer to "as a service" it implies that all services inside the classification are fully cohesive up to, and containing the respective level, thus combining any sub-levels, cloud computing environment has three primary variants, as under:

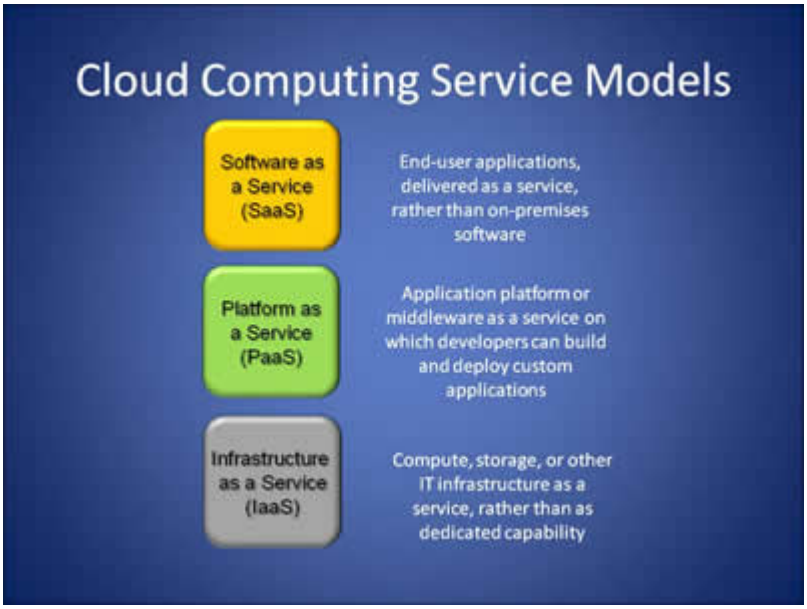


Figure 2.2: Cloud Service Models ***[21]***

a. Software as a Service (SaaS)

A facility provided by a firm or vendor which delivers a software solution to the system clients. The software may be core to a business, delivered over the internet or by other means.

b. Platform as a Service (PaaS)

A facility delivered by a firm, or vendor which provides a platform within which software applications can be developed. It is normally a web based service, with instantaneous constructs of the core infrastructure.

c. Infrastructure as a Service (IaaS)

A facility provided by a company or vendor which provides simple computer networking, load matching, content delivery linkages, routing, product data storage, and hoisting of virtualized operating systems.

2.9 DEPLOYMENT MODELS

The deployment model of the computing resource over the cloud is chosen basing on prescribes of resources in terms of technological, operational and business requirements. The core methods in which this deployment over the cloud is done is either ***private***, ***Community***, ***Public*** or ***Hybrid*** in nature and is termed as such.

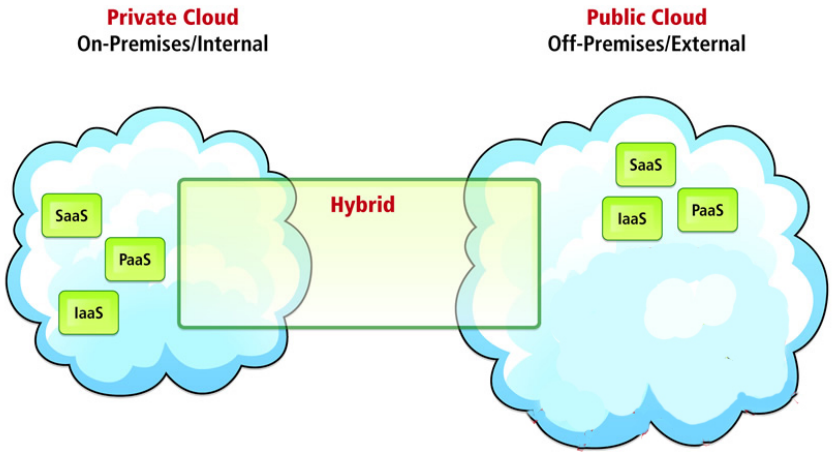


Figure 2.3: Hybrid Cloud *[22]*

a. **Private cloud:** The substructure is activated exclusively for a business. Its hosting can be through a third-party vendor or such organization which may keep its infrastructure spread over different countries which may be different to where it is being used.

b. **Community cloud:** The substructure is a combination of resources which may be the property of more than one company, thereby catering for a particular community having a united concern. For example: mission, security requirements, policy based and compliance considerations.

c. **Public cloud:** The substructure is globally accessible, usually catering for a very broad community or the general public. The provider of such services is called a cloud retailer.

d. **Hybrid Cloud:** The substructure is an amalgamation of multiple clouds amongst the types mentioned above (private, public, community, or hybrid), that does retain the individual properties of each technology, with the capability of data access / application sharing.

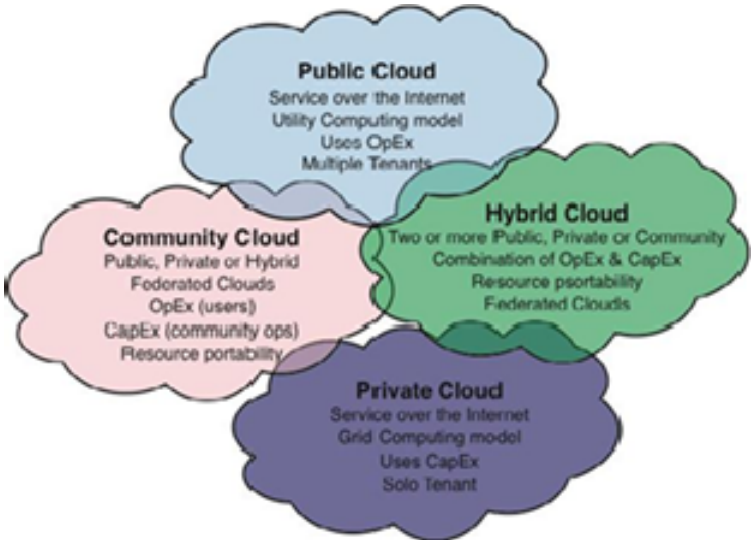


Figure 2.4: Cloud Deployment Models *[23]*

2.10 SECURITY CHALLENGES IN CLOUD

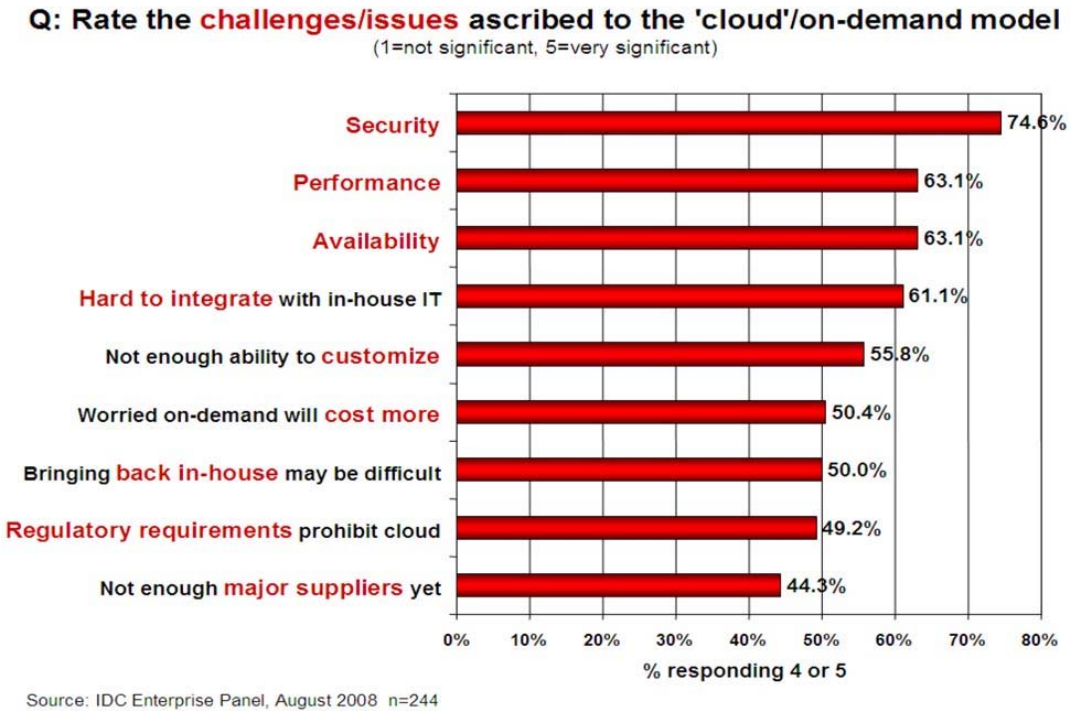


Figure 2.5: Security Challenges Analysis

There is no doubt about cloud having unlimited advantages but along with the advantages a few challenges are confronted as well, some of which are described below:

a. **Restricted user access**. There is an inherent level of risk associated with processing of data outside the premises of the enterprise due to the reason that subcontracted services bypass the "physical, logical and personnel controls".

b. **Supervisory compliance**. The responsibility of data security and integrity lies with the client due to business governs. Cloud service providers are subjected to external audits and security certifications.

c. **Data locality**. The client does not know about the location of the data storage. It may even be in a country with which the interest of the company or government clash.

d. **Data isolation**. Since the data is lying in a shared environment along with data of other clients, therefore only encryption might not be a comprehensive solution.

e. **Recovery**. In the event of a data loss or disaster, the CSP should be bound to tell the whereabouts of the data storage location. Normally CSPs hide data loss information.

f. **Exploratory support**. At times, investigation of unlawful activity with the stored data may not be possible due to the architecture and spread of the cloud.

g. **Long-term sustainability**. In an ideal environment, the CSP that has been engaged will continue to exist amid the swarm of globalization and expansion of giants in the field of CSP.

. **h.** **Types of attacks**. Examples of potential attacks to data stored over the cloud are:

(1) XML signature attack

(2) Cloud malware injection attack.

(3) Metadata spoofing attack.

(4) Flooding attacks

2.11 SUMMARY

In this chapter, study on cloud computing has been presented. The concept of cloud computing and its architecture has been discussed in detail. Furthermore, different service and deployment models of cloud computing have also been described along with the benefits that cloud computing offers. In the end, the security challenges to which the cloud is exposed are illustrated concisely.

.

Chapter 3

RELATED WORK

3.1 INTRODUCTION

This chapter centers upon the work already done in the area of data storage over the cloud. Different methods, techniques, algorithms and models suggested by researchers to enhance the efficiency and accuracy of existing algorithms have been discussed. At the end of this chapter, pros and cons of different algorithms affecting data storage are analyzed.

3.2 RELATED WORK

The integrity and precision of the data stored by the client over the cloud may be at risk because of the following factors:

a. The CSP has to hide data loss or reclaim space pertaining to the data which is either rarely accessed or not accessed at all.

b. A callous CSP may delete some of the data or might not store all the data on storage space it possesses, rather store some portion of it on some other media, tape or CD etc.

c. The cloud storage is susceptible to many external and internal threats.

To sum it up, the data stored over the CSP's server may be a cheaper or viable choice in monetary terms along with the ramifications of prolonged and huge quantity of data storage, yet the completeness and correctness of the data cannot be assured.

Since the client's data stored over the cloud passes through servers located worldwide, therefore the task of validating the completeness and correctness of this data is an uphill and difficult task to accomplish in the ambit of security of data.

Therefore clients need efficient practices to verify the integrity of their outsourced data with minimum computation, communication, and storage overheads. Consequently, many researchers have focused on the problem of '*provable data possession*' (PDP) and have proposed different schemes to audit the data stored on remote servers.

3.2.1 Provable Data Possession

Provable Data Possession (PDP), as the name implies, is a method to validate the integrity of data over servers that are located at remote places. ***Ateniese et a[1]l.*** have formalized a PDP model, in which the data owner pre-processes the data file to generate some metadata that will be used later for verification purposes through a challenge-response protocol with the remote cloud server. This file is subsequently stored upon an untrusted server after which the client can erase the copy held with him / her. In next step, the cloud server displays the file to be secure and un-tempered. It does so by answering the queries posed to it by the verifier, who may be the client or a trusted third-party data integrity checker. Researchers have proposed different variations of PDP

schemes under different cryptographic assumptions. Figure 3.1 and figure 3.2 depict the architecture of PDP.

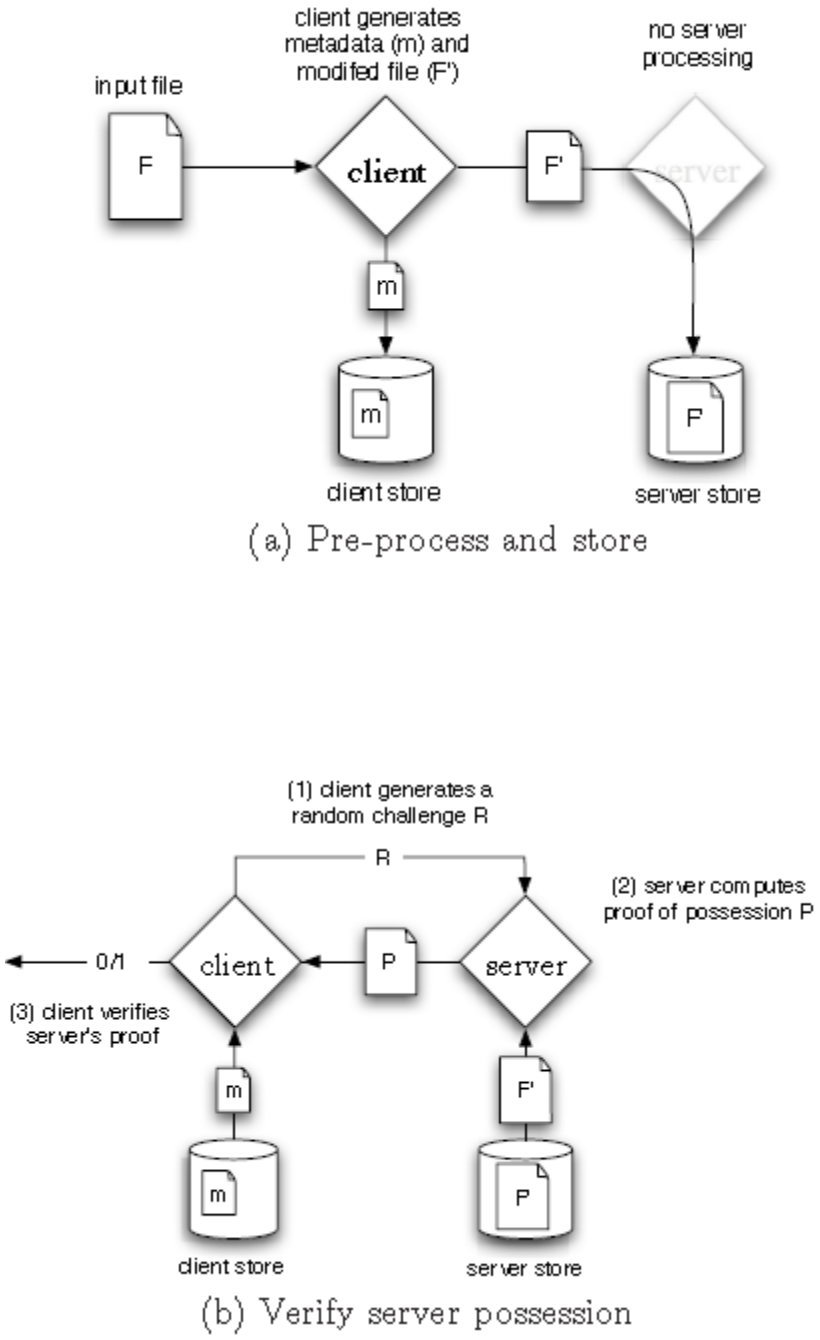


Figure 3.1: Provable Data Possession *[1]*

3.2.2 ARCHITECTURE

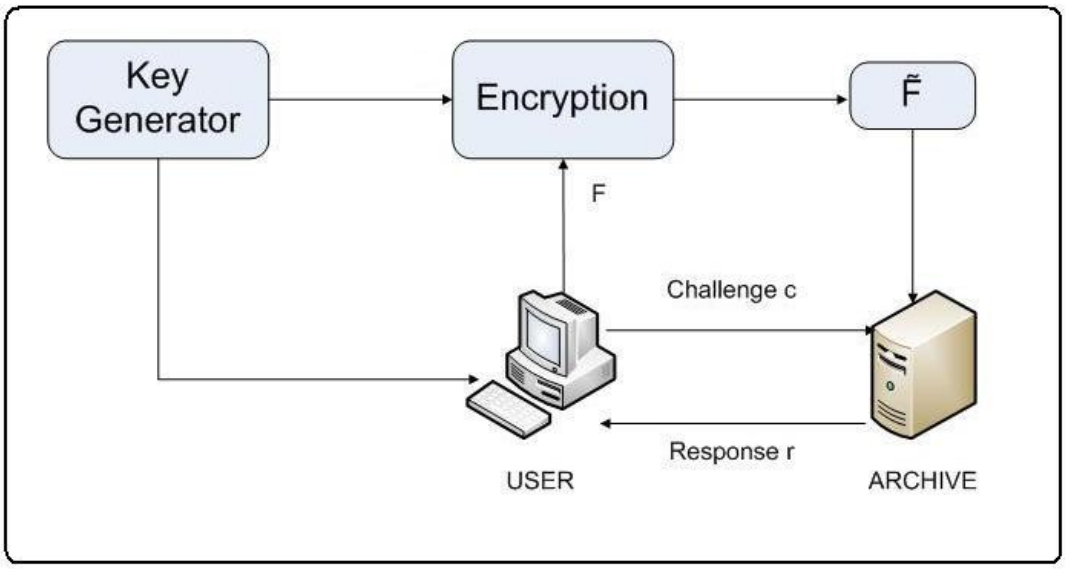


Figure 3.2: Architecture of Provable Data Possession *[1]*

3.2.3 THREAT MODEL

If the server does not answer the tasks assigned by client, it would mean a loss of data. In that case, even though the file may be partially or totally missing, but still the server may try to convince the client regarding its ownership and safe custody. The server's incentive for acting up in such a manner can be due to a variety of reasons, such as:

a. **Hiding the problem:** Not reporting a data loss incident that may be caused by:

   (1) Mistakes by the data handlers

   (2) Failure of physical storage resources

   (3) Data loss due to hacking attacks

b. **Monetary Reasons:** Reclaiming storage space by dumping data that is rarely or not accessed at all.

In contrast to this, the PDP model acquires evidence that the data is held and ascertains if the host had malfunctioned in-case of alteration of a portion of the data.

Deswarte et al. & ***Filho et al.*** have suggested methods using RSA-based hash functions to authenticate the storage of the data by the distant server. In contrast to other hash-based methods, their scheme allows the customer to undertake numerous checks through use of metadata created and stored initially on the customer's disk. The complication of both the communication connection and the storage with the client is $O(1)$. Their proposed algorithm suffers complications in terms of computability, where complete data file must pass through exponentiation procedure in which all blocks of the file structure are accessed. In addition to that, applying RSA protocol on the complete data file results in exceptionally sluggish performance i.e. approximately 20 sec/MB for 1 kb keys using processing power in excess of 2.8 GHz.

Schwarz & Miller have suggested a method that has the ability to authenticate data stored at numerous locations as per m/n scheme using erasure-coding. Using algebraic signature paradigm, the assurance of custody of files is verified. In this scheme, a packet of data is read and then the signature match of this packet record is certified against that of the parity check. Major disadvantage attributable to using this structure lies in the communication & computational complexities, which act in a linear fashion to the authentication requirement received against each packet of data. Moreover, the data security aspects of this proposed structure are yet to be established.

The transfer of data is authenticated at source according to bandwidth usage grid scheme, which is based on homo-morphic hash algorithms.

Krohn et al. have discussed various applications using the homo-morphic hashing technique. Using homo-morphic technique, they transform numerous input blocks into a singular block. The limitation in using this scheme is that it is based on erasure coding, because of which a sub-group of data packets can be composed with it. Furthermore, source validation methods do not apply to provable data possession.

Juels and Kaliski have proposed a scheme for ***Proof of Retrievability (POR)*** that permits a host to assure a customer about its ability to salvage data that was kept with that host earlier. In this scheme, lookout blocks are disguised amongst the regular file blocks to act as a spy in uncovering packet alteration by the host. The limitation of this scheme lies in its inability to tackle non-encrypted files and also in answering to a restricted query structure.

3.2.4 ANALYSIS. After analyzing these approaches, following limitations are identified:

a. These schemes are at burdensome in terms of bandwidth used for computation and communication operations on the entire data structure.

b. The desire of the customer that his/her data should be stored at one place in succession.

c. The security guarantees for data possession are not provided.

d. Repetitive outbreaks render the run-time routine as doubtful.

e. PDP and related schemes mentioned above are based on the ideal case of data that is stored in a linear storage area, with no outsourcing or changing within the file, rendering it as a classic case of a storage that is fixed and archival in nature.

3.3 DYNAMIC PDP

The application of fixed or seldom changing data is very limited, as compared to everyday use of data, in which the data owner has the liberty to perform data oriented operations of insert, modify and delete and still have the guarantee of non-adulteration of the packet data. This type of PDP storage scheme is popular for files, database and sharing of data.

Another Framework named 'Dynamic Provable Data Possession' (DPDP) was introduced by ***C. Chris Erway et al.*** to cater for the dynamic aspects of data. DPDP puts in use a newer version of valid wordlists in which the rank evidence is used for linking of wordlist records. The protocol claims to support proficient authentication processes upon the data at the basic packet level like validated delete & insert operations. There are some DPDP paradigms in the literature which satisfy different system requirements.

3.4 UNTRUSTED CLIENTS

An important thing to note in above analysis is that all these schemes address the issue of untrusted server, yet none of them {PDP (all variants), POR, DPDP, SPDP, EPDP, CPDP} addresses the problem of untrusted clients. A separate mechanism for client authentication and authorization is a must. This research work carries along the issue of untrusted clients as well in addition to untrusted servers. The proposed mechanism is complete, both with respects to client and server, hence it is named as ***Authenticated Client Provable Data Possession*** **(AC-PDP).**

3.5 SUMMARY

In this chapter a background study on cloud computing was presented. Details regarding cloud computing, security within the cloud network, and protocols to cater with security issues have been presented. The provable data possession and its extensions have been studied for its dedications towards providing security in cloud computing. Algorithms, techniques and various directions that have been discussed in this chapter form the foundation of the research work in this thesis.

Chapter 4

DESIGN & IMPLEMENTATION OF AC-PDP

4.1 INTRODUCTION

Researchers generally have a limited set of considerations pertaining to client authentication while designing security protocols for cloud based systems. Yet client authentication is one of the most basic requirements of cloud computing. Furthermore, the verification of client's authenticity is one of the most serious problem / concern for data storage and retrieval on unverified hosts. Such a situation may arise in:

a. Peer-to-Peer storage systems

b. Network files systems

c. Records that have been stored for an extensive duration

d. The storage of entities pertaining to web-based services

e. Catalogue based schemes

Therefore, these structures require authenticity checks by storage servers in order to take care of issues like misrepresentation or modification of data during access. This chapter presents the architectural design and implementation details of the proposed *Authenticated Client Proof of Data Possession* (ACPDP) *protocol* for Cloud networks (referred to as CSP). This protocol has been designed to meet the highest levels of security. In addition to conventional confidentiality, ACPDP also provides authentication and integrity. Furthermore, since cloud networks can be deployed in SOA environments,

therefore the importance of providing confidentiality, authentication and integrity within a single protocol intensifies manifold.

4.2 THREAT MODEL

A brief description of the threat to which PDP is susceptible has been narrated before a formal description of the proposed scheme is presented. Since PDP has addressed maximum challenges with reference to untrusted server but the area containing threats for untrusted clients still remains unanswered.

4.2.1 Threat Model-1

The malicious intruder threat has affected several businesses. In the cloud based environment, this threat is intensified due to the coupling of the client and the host services in a common managerial sphere, coupled to that are unverified processes & procedures of the service providers. Data can be compromised in multifarious ways, such as:

a. Modification or removal of data in the absence its reserve storage.

b. Storing a data packet out-of-context, due to which it cannot be salvaged.

c. The records are stored on an untrusted host.

d. Forgetting the private key by which the data had been encoded.

The chances of adulteration in data storage within the cloud environment grows many fold because of the abundance of the hazards and tests (peculiar to cloud's environment) and may increase further in linkage to the design structure or due to the method in which a cloud storage functions. Furthermore, the unintended user must be denied admittance to the data that is critical or sensitive in nature.

Scenario: A simple scenario could be when client is sending file to server for storing. While client sends file to server for storage, any intruder can interrupt the process in the middle and pose as the intended server. By utilizing this technique, the intruder can easily hack the file before it reaches the server. This security threat implies for a stronger and more effective client-server authentication mechanism. PDP currently does not have any verification mechanism to make sure that both the client and the server involved in the process are genuine.

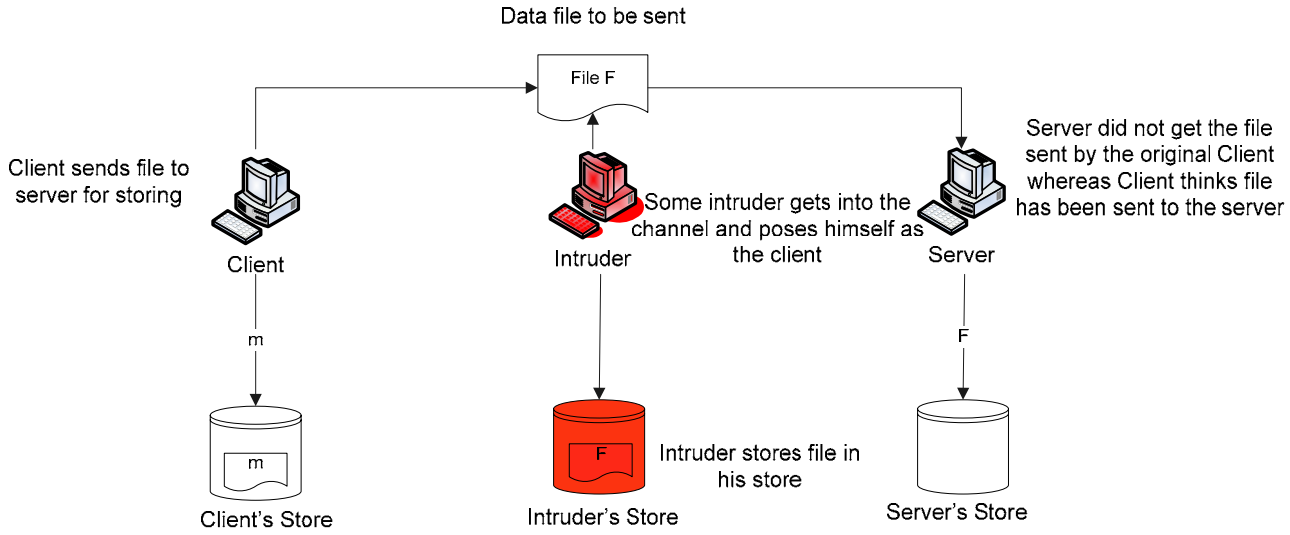


Figure 4.1: Threat Model-I

The threat perception may be individualistic in nature like a random hacker or may be bigger in size, ranging from groups of people who provide hacking services as a business, to those whose accounts are compromised only because they are part of a firm that was targeted to the biggest level in which a country employs its illegitimate means and resources to intrude the network of the other country. By merely adulterating the access protocol, the hacker gets access to tons of data that are stored in the cloud.

4.2.2 Threat Model-2

Data leakage or loss can have devastating impacts on businesses, including:

a. Damage to company's brand and reputation

b. Loss of morale and trust of employees, partners, and customers

c. Leakage of business secrets to rival parties, thereby affecting business

d. Compliance violations

e. Legal ramifications

In context of PDP where there is no way to authenticate a client, any malicious user can intrude and pose himself as the originator client and can get information from the server. When client sends a challenge to server, the hacker can hack the challenge and resultantly become the recipient of the proof, generated by the server. Such situations arise because of insufficient authentication, authorization, and audit (AAA) controls in existing PDP schemes.

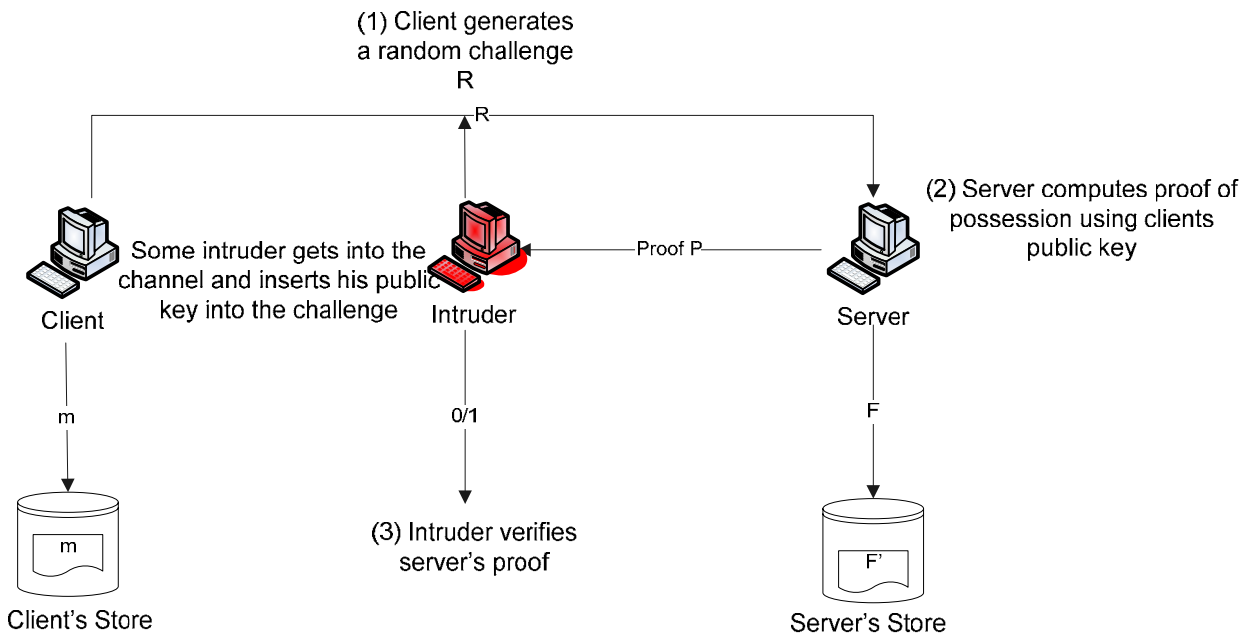


Figure 4.2: Threat Model-II

The concept of acquiring the services of professional hackers is ages old, which include Trojan horses, false identification and exploiting susceptibilities in code / hardware. Getting illegal access to an individual or organizations account creation information or their passwords means the attacker gets access to their data as a legitimate user. If the attacker ingresses to such resources, he may not only read the data, but, also modify and change the data to re-route the company's clients to illegal sites. This warrants the innovation of a mechanism to avoid such undesirable conditions.

4.3 AC-PDP DESIGN ELEMENTS

Formulated security protocol (AC-PDP) caters for the above described threats and is specifically intended for the cloud's environment. It has been designed for an optimum cloud server and provides the highest level of security in mission critical environment,

while consuming negligible resources. Augmented to that is the eradication of overheads incurred due to separate user authentication mechanisms. Furthermore, the digital signature has been employed in such a way so as to remove the need of running an independent authentication routine. The elements that make up AC-PDP successful are described below:

Figure below demonstrates the RSA key generation mechanism**.**

a. **Encryption.** The first step in AC-PDP is key generation using RSA, which is a public-key based algorithm built upon "*modular exponentiation*" invented by Rivest, Shamir and Adleman and named as such. Being a public key based system, the private key is different to the public key, but definitely has linkage to its corresponding public key. The notation for which is:

$$If\ A<N,\ then\ (A^{e}\ mod\ N)^{d}\ mod\ N = A$$

It implies that the encryption of '***A***' is done through '***e***', while the decryption is done using'***d***'. RSA is an asymmetric encryption algorithm which is considered more secure as compared to symmetric encryption schemes.

At this very step the plain text is not provided to the encryption program, because the generation of both keys is a pre-requisite for performing encryption.

b. Key Generation. Figure below demonstrates the RSA key generation mechanism

1. Pick two random prime numbers, p and q, p ≠ q

2. n = p * q

3. Select odd integer e, coprime to φ(n)
   - φ(n) = (p-1)(q-1)

4. Find d s.t. ed ≡ 1(mod φ (n))

5. Publish e and n as public key
   - Used to encrypt the message

6. Keep d (and n) secret as private key
   - Used to decrypt message

Figure 4.3 (a) Key Generation by RSA

Example

p = 941

q = 997

n = 941 * 997 = 938177

φ(n) = 936240

e = 896107

d = 144403

Public keys: e = 896107
n = 938177

Private key = d

Figure 4.3 (b) Example of Key Generation by RSA

4.3.1 RSA Encryption

Public-key cryptography refers to a cryptographic system that requires two separate keys, one to encrypt the plain-text and the other for decrypting the cipher-text. Both the functions are not done by one key; one of these keys is published or made public while the other is kept private. RSA is one of the examples of Public-key cryptography.

The working of public key cryptography (also known as *asymmetric* encryption) is shown below.

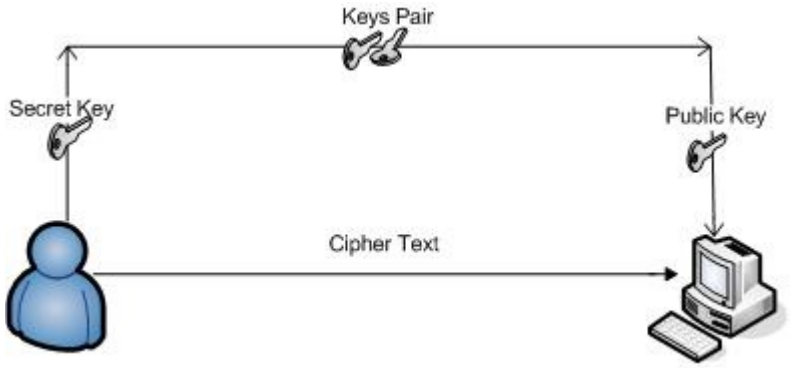


Figure 4.4: Asymmetric Encryption

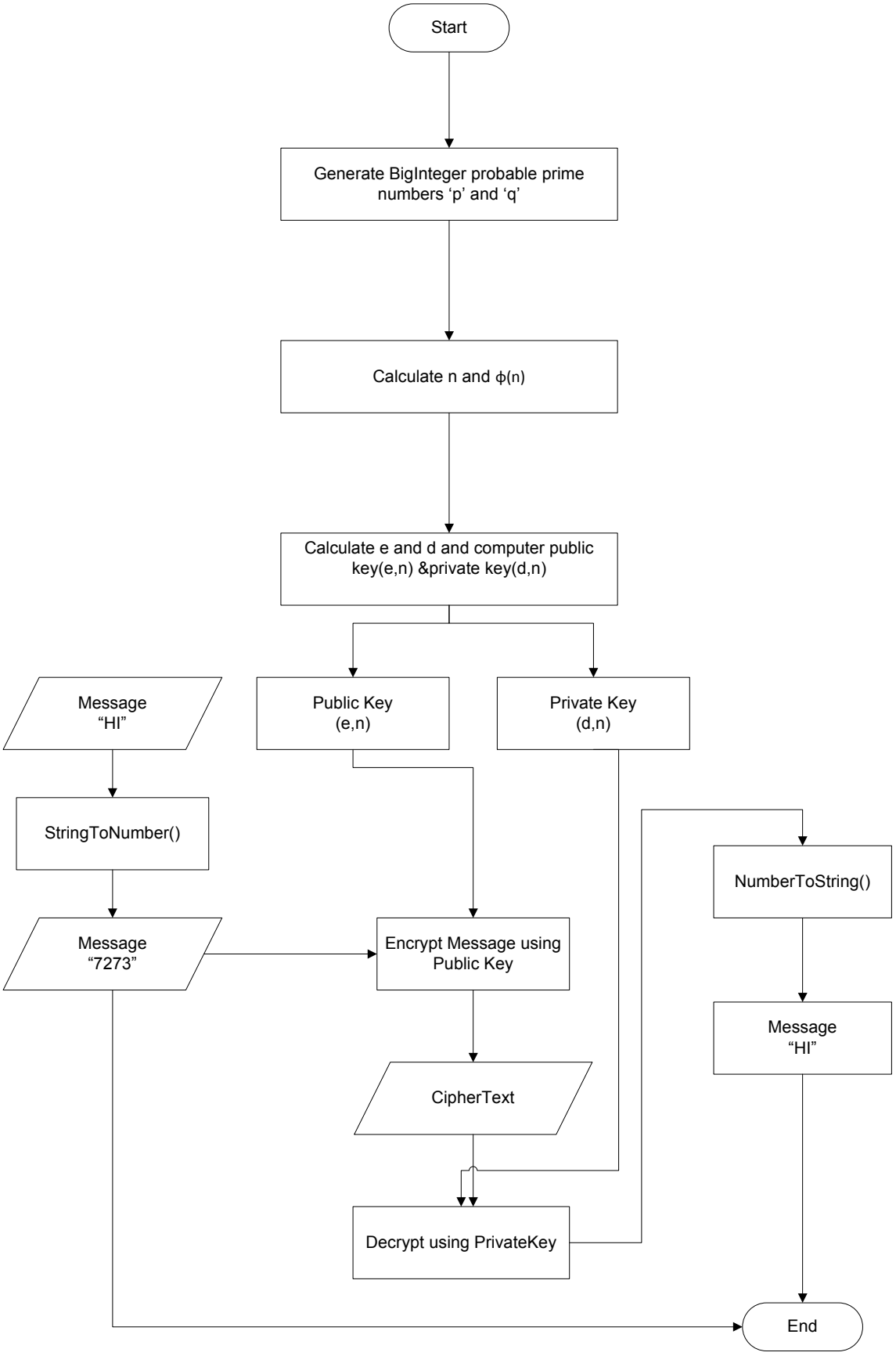


Figure 4.5: Flowchart of RSA Encryption

4.3.2 ElGamal encryption

Implementation of AC-PDP has also been done with another asymmetric encryption scheme named ElGamal. This scheme is firstly used for key generation and later for encryption-decryption and generation of digital signatures. ElGamal scheme has been incorporated for the following reasons:

a. ElGamal might be a bit stronger than RSA since calculating discrete logs is at least as tough as integer factorization and it has a stronger bit-to-bit encryption as compared to RSA.

b. Being different to RSA, the security in this scheme depends upon the complexity of discrete log problem, which opposes factoring problem.

c. Since deciphering is done more often than enciphering, the total cost of the operation is lesser in this scheme.

d. It is also faster because of the shorter exponents, while in RSA very short exponent (***e***) can be chosen for encryption but the exponent (***d***) for decryption will probably be 1024 bit long (for a modulus with 1024bit).

e. The exponents can both be chosen independently in case of ElGamal.

f. PGP suggests exponents to be of length 160bit, which makes ElGamal decryption faster even though an inverse has to be calculated next to the exponentiation.

ElGamal encryption system is a variant of D-H key swapping mechanism, composed of:

a. Key Generator

b. Encryption Algorithm

c. Decryption Algorithm.

A brief description of these is shown in the figure below.

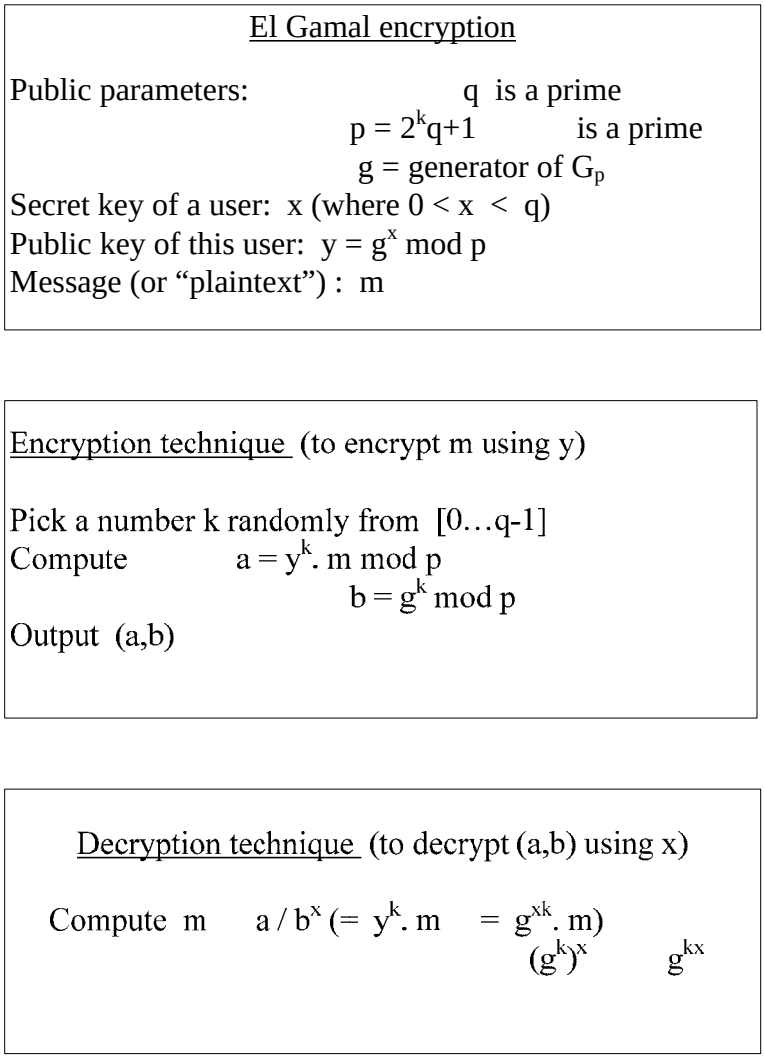


Figure 4.6: ElGamal Encryption/Decryption

4.3.3 Meta Data

Tag blocks are supposed to be created after the keys have been generated using asymmetric encryption scheme. The challenger station computes the authentication metadata and sends it to the server along with the modified file.

Homo-morphic Tags: Homo-morphic verifiable tags compose the structural blocks in PDP, which are used to verify the metadata of data storage packet. Metadata contains block number, start location of the block in the file and the location where the block ends. Analogous to the block of file, if a message is denoted by ***m***, its homo-morphic verifiable tag is denoted by ***tm***. These tags are saved on the server, linked to the corresponding file ***F***. The figure below demonstrates the process of metadata creation.

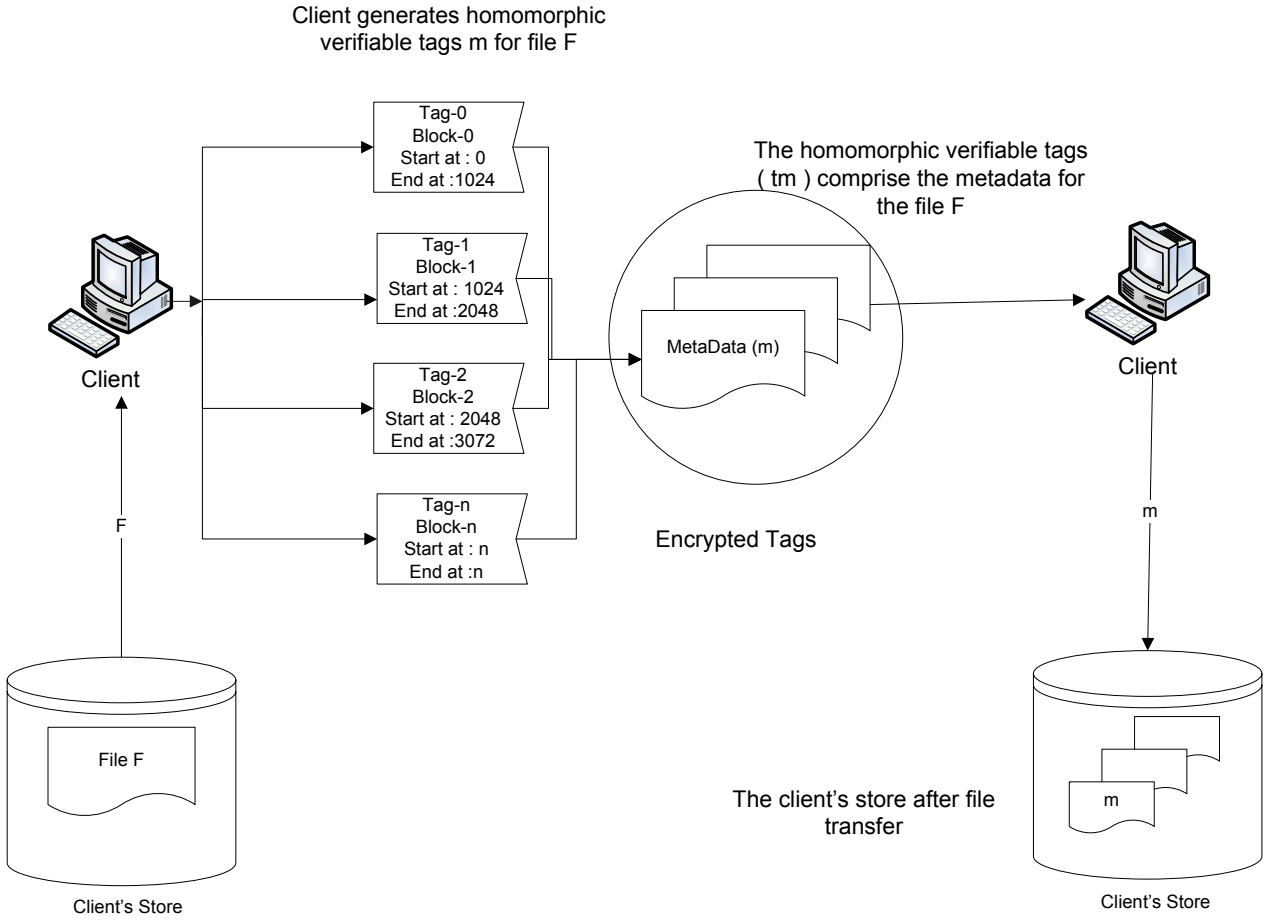


Figure 4.7: Metadata Encryption

4.3.4 Challenge

*A challenge "**chal**"* is generated by ***C,*** which designates specific data portions whose custody it wants to verify. It is this challenging phase where AC-PDP is considerably different from PDP schemes previously articulated. In current PDP schemes, the server receives the challenge without any tag about the sender. Whereas, in proposed AC-PDP scheme, the digital signatures of the sender also accompany the challenge. The client (**C**) is verified by the server (**S**) using

$$V \leftarrow \text{GenProof (pk, F, chal, } \Sigma\text{, Sig)}$$

Resultantly, it generates the proof of possession (*V)*. The client can verify the validity of the proof using:

$$\text{CheckProof (pk, sk, chal, } V\text{)}$$

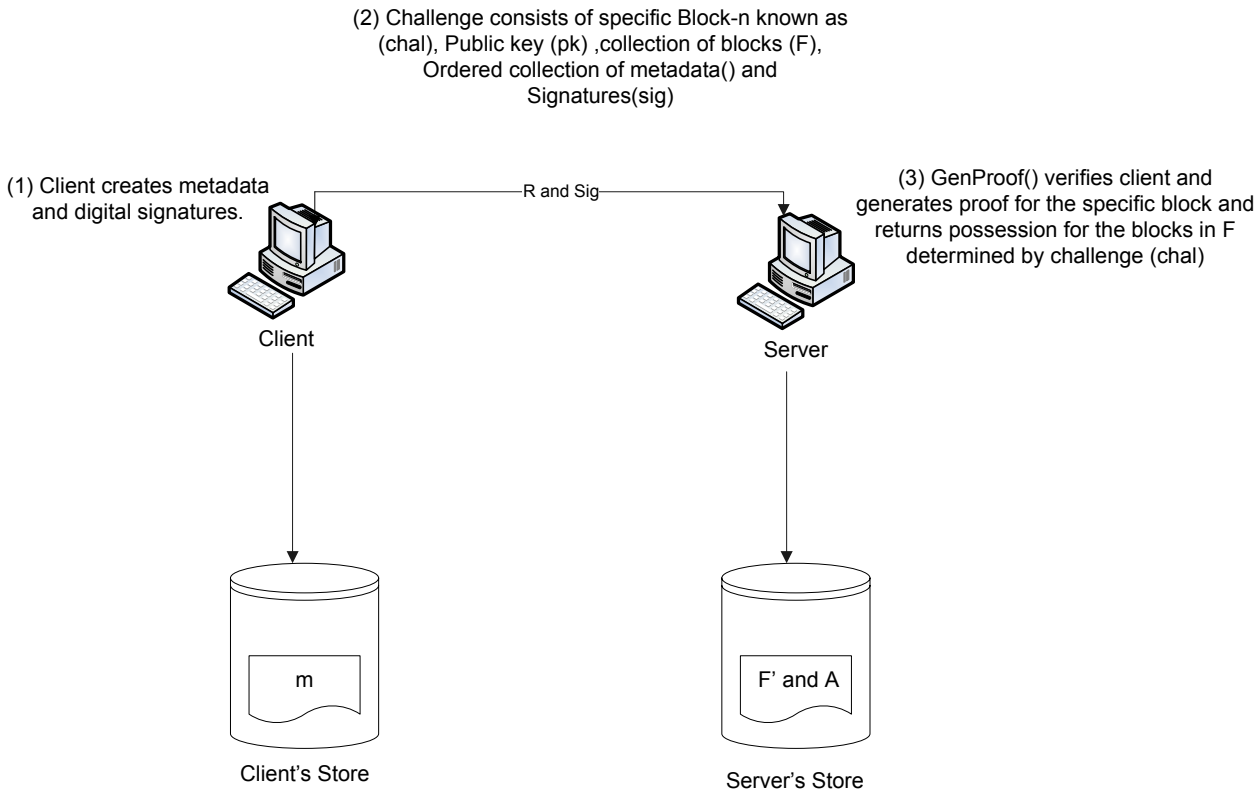


Figure 4.8: Challenge Creation

4.3.5 Digital Signatures

As valuable and sensitive data is stored over the cloud, therefore it must be made tamper proof against malicious attackers who are not the senders or the receivers of information. In addition to that, at times due to compulsions of data security, the data and its associated timestamp has to be made damage resistant for operations by both the owner and the host.

This is the point where **AC-PDP** differs from all other **PDP** schemes. No other scheme of PDP has client verification incorporated into it. The algorithm of AC-PDP happens to be more secure due to the addition of digital signatures for authentication of the client. The signatures added to the scheme bear the following characteristics:

a. Relative ease of establishing the authenticity of the signature.

b. Difficulty of forging the signature.

c. Non-transferability aspect for signature.

d. Mutability resistant aspect of signature.

e. Non-repudiation aimed at ensuring deny-proofing by signee of having signed the signature.

4.3.6. Principle

The digital signatures inside the AC-PDP protocol are computed on the basis of documents needing signatures, in which the signee withholds some secretive evidence. Message-Digest **(MsgD)** is used in place of the entire message; and is generated through application of hash utility. The hash utility used by AC-PDP is **SHA** that takes in messages of random or varying types and yields **MsgD**. This sophistication of algorithm

guarantees a high improbability that two unrelated messages are recorded against one hash utility level. The computation of digitally signed documents is either done by one of the following Crypto-systems:

a. Symmetric Key

b. Public Key

Public key cryptosystem is used in AC-PDP.

4.3.7 Public-Key Crypto System

A public key cryptosystem incorporates two keys:

a. Private-Key (***sk***): created and safe guarded by the creator

b. Public-Key (***pk***): shared with public and people who interact with the creator.

To ensure secrecy of under-transfer message being transmitted, the sender uses his public key (***pk***) to encrypt the data. The receiver upon receiving the file uses the matching private key (***sk***), shared to him by the sender, for decrypting the data. Validation is confirmed by encrypting a return message using sender's private key (***sk***) this time, that would be decrypted using the matching public key (***pk***) of the sender. Upon decryption, if the message is recreated again, it implies that the keys generated by the sender were correctly employed by the receiver and the message has been successfully transmitted in encrypted form.

4.3.8 Attributes of the Signature

The signature in AC-PDP contains following attributes:

a. **Public- Key generated by the sender:** Necessary for authentication and as the name suggests, is available to the public.

b. **Sender's name and contact information:** To check and keep track of the identity of the sender.

c. **Validity period of Public-Key (*pk*):** Puts a time limit on the validity of the signature, which ensures that the holder/acquirer does not misuse the signature beyond its validity period.

d. **Credentials of Sender:** Provides an identity to the creator of the digital signature.

e. **Signature's unique ID number:** This part is a unique number that is bundled to the signature for tracking and extra identification reasons.

f. **Certification Authority's (CA's) Signature:** The authentication signature which validates the ID of the authority that is liable to issue the digital certificate.

g. **Pass Phrase:** A random phrase sent by the sender as an additional ID.

The simplified model to illustrate the creation and validation mechanism of the signature is as under:

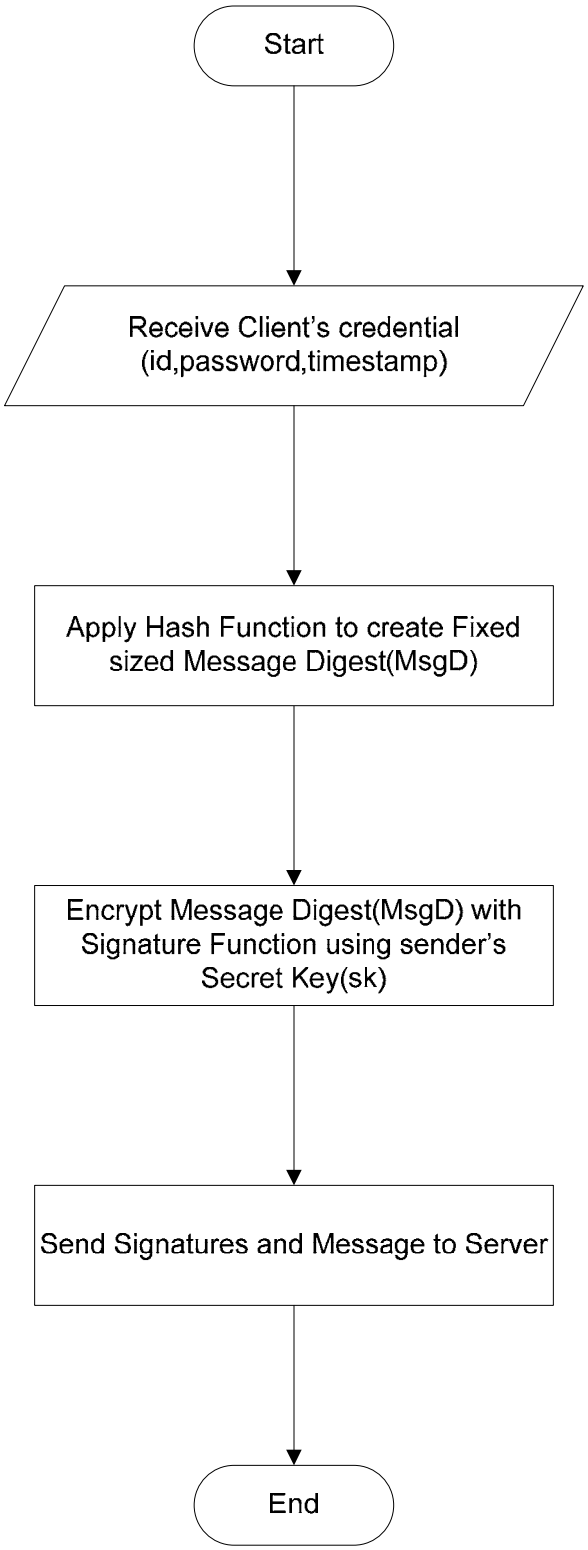


Figure 4.9 (a) Digital Signature at Client Side

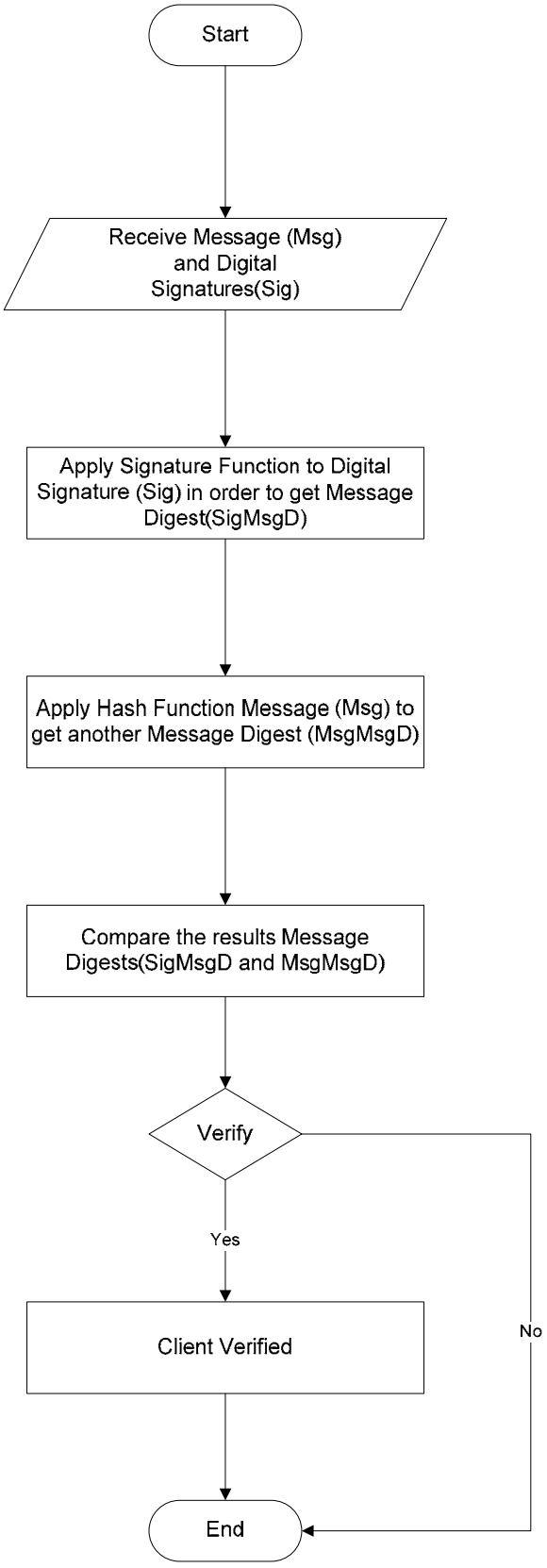


Figure 4.9 (b) Digital Signature at Server Side

4.3.9 Proof

The computation done by the server against the challenge posed to it by the client is called proof. The client creates a challenge which is represented as **'*chal*'**. This ***chal*** desires that server should prove its custody of blocks $m_{i1}, \ldots, m_{ic}$ determined through ***chal***, as per the condition that:

$$1 \leq ij \leq n, 1 \leq j \leq c, 1 \leq c \leq n$$

Proof's structure that is created by the server *S* is shown in the figure 4.10.

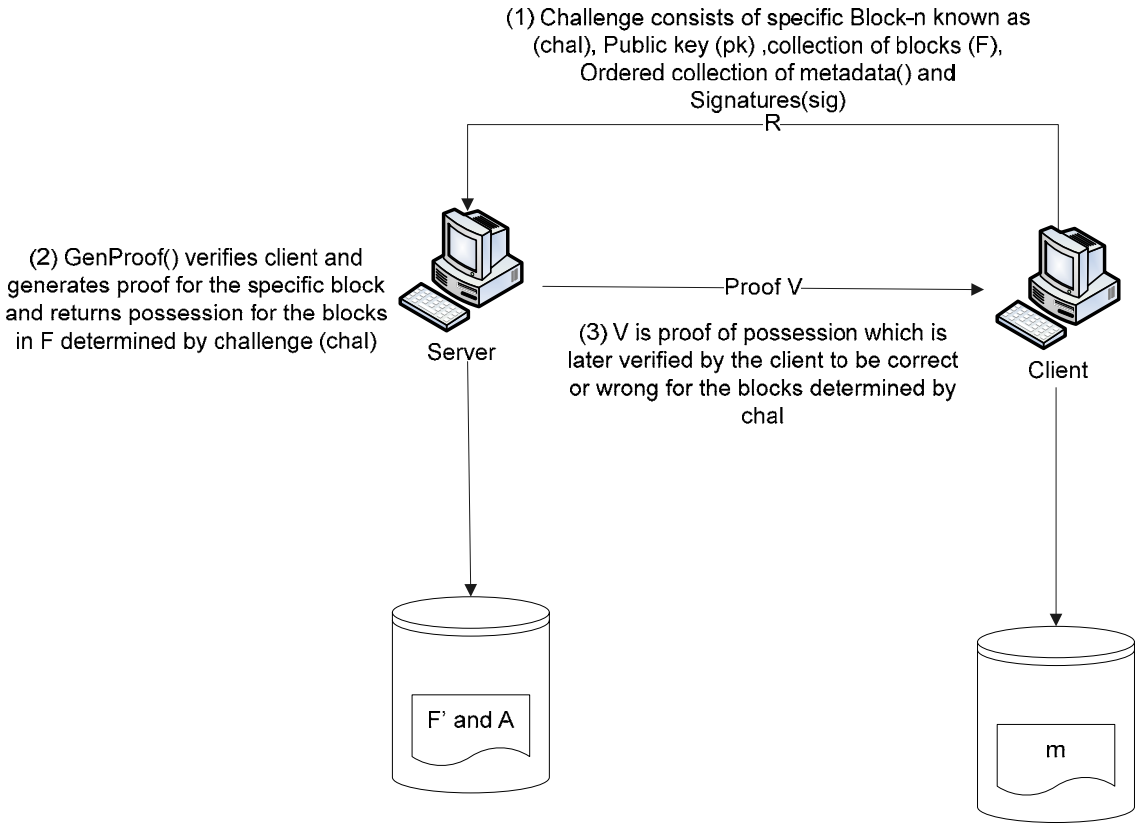


Figure 4.10 Proof Generation

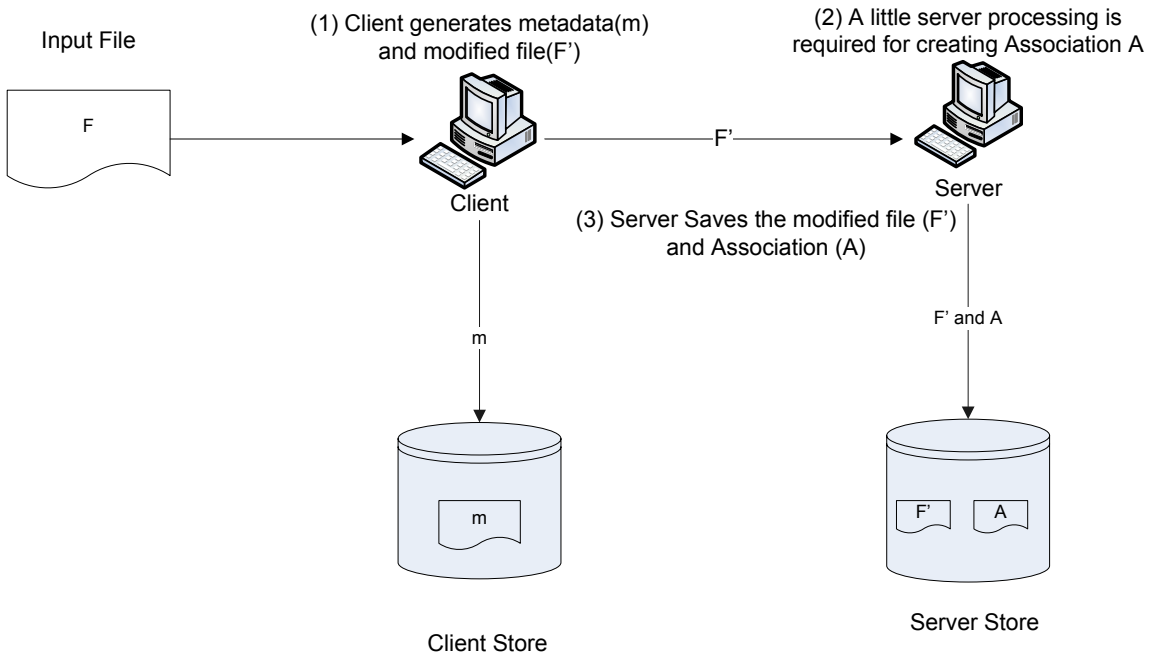
4.4 ARCHITECTURAL DIAGRAM OF AC-PDP

AC-PDP has been designed with the concept of PDP in focus. This algorithm works in two steps.

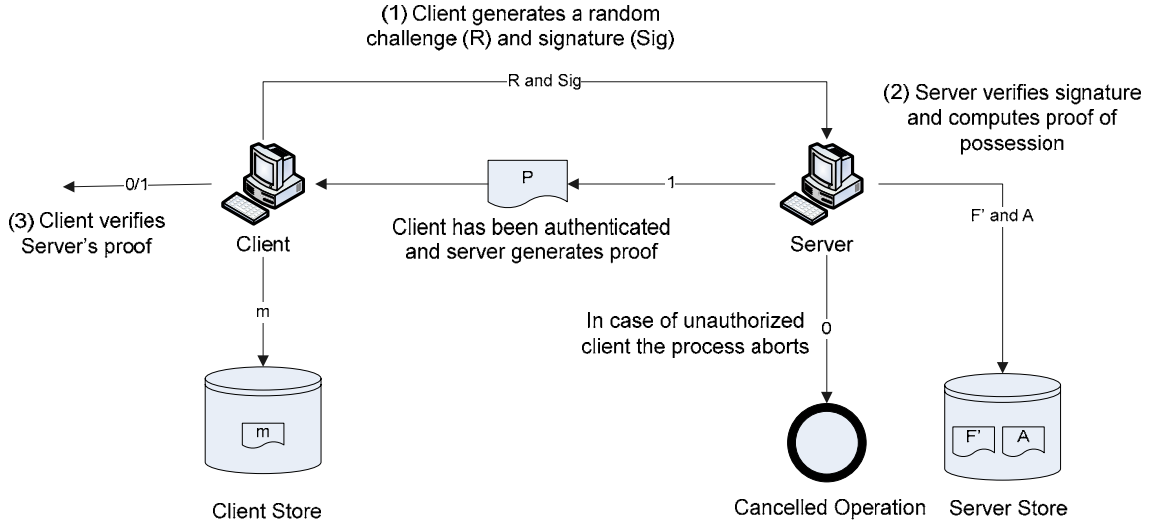
a. **Preprocess and Store phase:** The client C may delete its copy after it pre-processes the file whose output is a metadata file that is locally stored and subsequently transmitted to the server S.

b. **Verify Server Possession Phase:** The server answers the challenges issued by the client after validating the client by use of the stored file.

Figure 4.11 shows the two stages of the protocol.



(a) Preprocess and Store



(b) Verify server possession

Figure 4.11: AC-PDP Architecture

4.5 AC-PDP ALGORITHM

At first the generation of both types of keys is done as per asymmetric encryption scheme. Homo-morphic verifiable tags **(*Tm*)** are computed against every file block **(m)** using the public key. Association is then created by the server using metadata file **(F).** A challenge **(*chal*)** along with the digital signatures ***(Sig)*** is received from the client by the server which is later used for verifying the integrity of the stored file ***(F').*** A proof ***(V)*** is then generated by the server which is later verified by the client. The sequential flow of the algorithm is as under:

Notations

C = *Client*

S = *Server*

P_k = *Public key*

S_k = *Secret Key*

F= *Raw data file*

m = *Metadata*

T_m = *Homo-morphic verifiable tags*

F' = *Modified file*

A = *Association*

chal = *challenge*

Sig = *Digital signatures*

V = *proof*

atrList = *Client's Attributes List*

4.5.1 Algorithm

1. ***C** computes **(pk,sk)** through asymmetric encryption scheme **(RSA/ELGamal)***
2. ***C** computes homo-morphic verifiable tags **(Tm)** and saves metadata information **(m)***
3. ***C** sends modified file **F'** to **S***
4. *Upon receipt of file S stores **F'** and an association containing user information.*
5. ***C** prepares digital signatures **(Sig)***
6. ***C** sends random challenge **(R)** to server along with the digital signature **(Sig)** and Client's attribute list **(atrList)***
7. *Upon receipt of signatures **(Sig)** and challenge **(R), S** will first look into association **A** to check which user the mentioned file belongs to. S will then apply signature Function to the digital signature **(Sig)** and hash function to the Client's attribute list **(artList),** sent via challenge for verifying user.*
8. *If user is authentic **S** will generate a proof **V** and send it to **C***
9. ***C** then checks **V** to ensure the integrity of the file*

4.5.2 Flowchart

A detailed flowchart of the steps on client side and server side has been illustrated below.

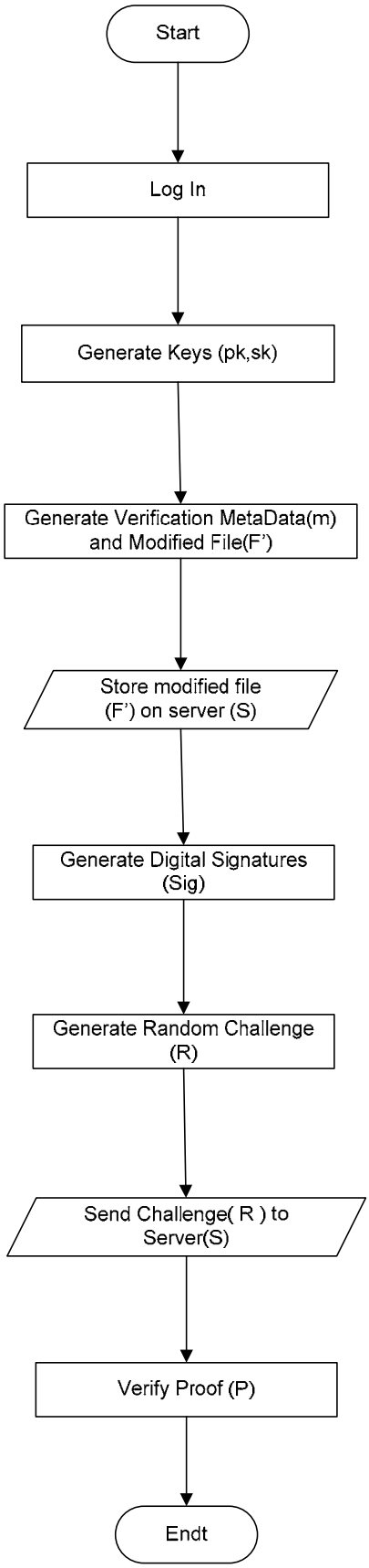


Figure 4.12: AC-PDP Flowchart, Client Side

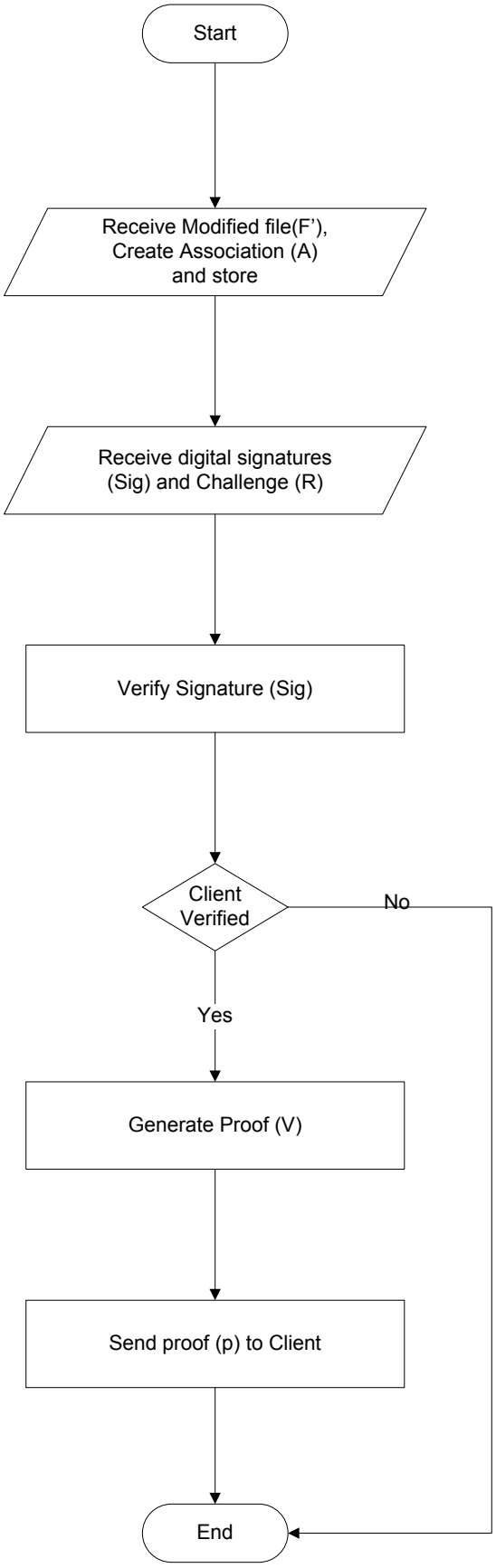


Figure 4.13: AC-PDP Flowchart, Server Side

4.6 COMPONENT DIAGRAM

Component diagram of AC-PDP is shown in the diagram below. The main components that make up the AC-PDP are as under:

a. ***ChalCreator*** creates the challenge with homo-morphic verifiable tags HVT through asymmetric encryption scheme.

b. ***SignatureCreator*** Creates signatures using encryption technique.

c. ***AssociationGeneration*** is required by ***ProofGenerator*** and is based on RSA/ELGamal encryption.

d. ***SignatureVerifier*** supports ***ProofGenerator*** in validating the user.

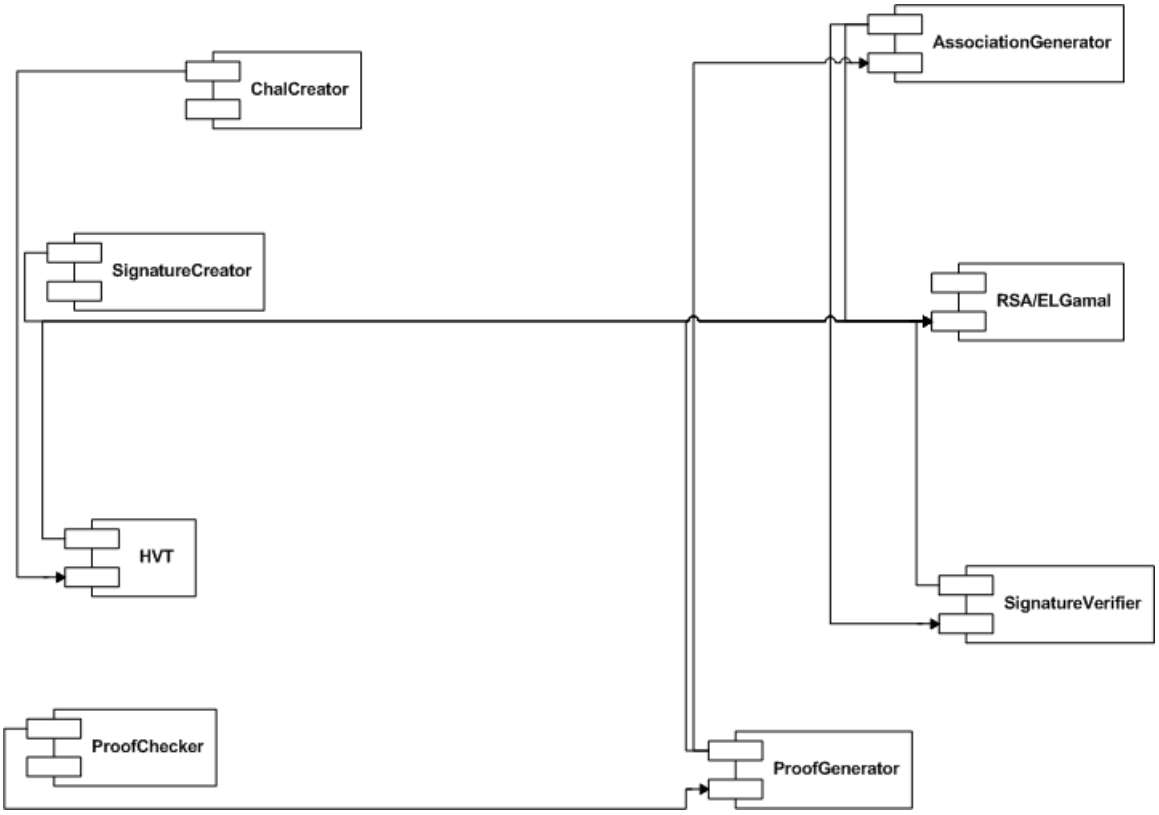


Figure 4.14: Component Diagram of AC-PDP

4.7 AC-PDP IMPLEMENTATION

The architecture of AC-PDP is made keeping in subconscious a high security environment. It works on asymmetric encryption and is intended for scenarios in which the measure of security guarantee is quite limited. For implementation part, a server and client class each using ElGamal and RSA has been simulated. These classes simulate the real world environment of the cloud.

a. **Step 1: Connection is established.**

```
try
{
   client.Connect(serverBox.Text.Trim(), 3001);
  btnConnect.Enabled = false;
   serverBox.Enabled = false;
   if (string.IsNullOrEmpty(serverBox.Text))
      serverBox.Text = "Localhost";
                        connectionMsg.Text = "You are connected to
                       "+serverBox.Text;
}
catch(Exception exp)
{
   MessageBox.Show(exp.ToString());

}
```

Figure 4.15: Code Snippet for Establishing Connection

b. **Step 2: Keys are generated:** This step is implemented using ElGamal and RSA. Two distinct prime numbers (of same bit length) p and q are chosen for key generation using RSA, as a first step.

c. **Step 3: Modulus is computed:** Next, modulus (n) is computed for both public and private key as:

$$n = pq$$

d. **Step 4: Euler's Totient Function is computed*:*** The computation is performed as below:

$$\varphi(n) = (p-1)(q-1)$$

e. **Step 5: Computation of *e* and *d*:** Integer e is chosen as per conditions below which is subsequently used for computing integer d as under:

$$1 < e < \varphi(n)$$

Where the GCD of following equation is co-prime

$$(e,\varphi(n)) = 1, \text{ i.e. } e \text{ and } \varphi(n)$$

In next step ***d*** is computed, as under:

$$d = e^{-1} \bmod \varphi(n)$$

Here, ***d*** being multiplicative inverse of:

$$e \bmod \varphi(n)$$

Public-Key is composed of modulus *n* along with encryption exponent *e*.

Private-Key is composed of modulus *n* along with decryption exponent *d* that has to be preserved and undisclosed.

f. **Step 6: Creation of *pk* and *sk*:** They are created using inner state of the code under RSA as:

```
RSACryptoServiceProvider RSAcsp = new RSACryptoServiceProvider(2048);
string pnp = "<BitStrength>2048</BitStrength>" + RSAcsp.ToXmlString(true);
string p = "<BitStrength>2048</BitStrength>" + RSAcsp.ToXmlString(false);

ASCIIEncoding encoder = new ASCIIEncoding();
TcpClient client = new TcpClient(serverBox.Text.Trim(),3000);
NetworkStream clientStream = client.GetStream();
byte[] pkey = encoder.GetBytes(p);
clientStream.Write(pkey,0,pkey.Length);

this.Invoke(new EnableAndSaveIt(this.EnableAndSave), new object[] { p, pnp });
```

Figure 4.16: Key Generation by RSA

g. **Step 7: Creation of Public Key:** The format of the key when both public and private key is generated by RSA looks like this.

```
<RSAKeyValue>
        <Modulus>…</Modulus>
        <Exponent>…</Exponent>
        <P>…</P>
        <Q>…</Q>
        <DP>…</DP>
        <DQ>…</DQ>
        <InverseQ>…</InverseQ>
        <D>…</D>
</RSAKeyValue>
```

A sample private key generated by RSA is like this:

<BitStrength>

2048 //RSA bit Length, The security of an algorithm cannot exceed its key length

</BitStrength>

<RSAKeyValue>

<Modulus>

nyJfSu3C0m72k8kzKak4njP7weQL6L/j7E+nWqJnPtFLoq8YeeQNSun0FU5lxTRN/wNOm/NUYY

cewgqryKNrnN2ATuUeJFIn5WtXWocjC9jO4o4WU1NOiPqehidWYhwxUWJzLoWAZ0MdvMdqfSf

1n010fC6jyXoGle/WHZwzyf/t3k2xucQLkaTLNVbhWIEf6UwRNEMNS0POkb9EIImQSiH1XeZjiEC

Ky0axWXkjoARQ60x37cuSpx4BIqAwpdjJIdW1aKNzbCQHpElOFsisyfAUq7NwpKF+yT+Nxb3TE

bPWSyXZ4QgE2aQVnbNai9HWTIDQra72afp3CnLtFZvNew==

</Modulus>

<Exponent>

// Public exponent e, 1 < e < φ(n), which is coprime to φ(n), that is, gcd(e, φ(n))=1

AQAB

</Exponent>

// First Prime number 'P' in base64

<P>

05U1aLMArsvdLE/Ml8Cf7zmwhKUrN2o8lHtE6l7s9xD6IGn7M4vhtpBu2ID9gH4RRKBlMyeWraf2L

J5yPvuZIr6ORB7mzzu9rFZcErPx8KmG4yv9TBcRgwhYCQ7gvCzT4PBsYH1eviHqT3YZh2j7dPw

WXTtGt++c6ljAKSJ8zmM=

</P>

// Second Prime number 'Q' in base64 such that P>Q

<Q>

wIp+zeY/eWFPrbiTYfuGjkaYi7gqBrjUhZYg0BFzh2xYYYKJBTaQZLBrEoaJ4qCjBhHF1LBXwM9d

BTA1Yqi377UkOx9QiN+0Aa2VlC0709VgZMTvvkCGhETVHeYgP7xkrVqCpydMK+ee6yrIs7Uc2n

PMLDhDMvpYbad7cQC1BAk=

</Q>

<DP> **// Prime Exponent 'P', DP = (1 / Exponent) mod (P - 1) is converted to base64**

N2WeqT4M1LdNvldaYvUtKs54+BHiwcyP9fRcc9zMwFdyleAJxFJAk7M3QnprZSQc01IHjSA1ZFx

QqRgHI5RuYNlxmEZa+jFTFPc7fi4dD3zjc0FGPZDgpeUeX6jEkKBQDVoFljHdB30LwyhRwNFs3

RFPrbNibbBP7k+0aFC+Sq8=

</DP>

<DQ> **// Prime Exponent 'Q', DQ = (1 / Exponent) mod (Q - 1) is converted to base64**

sLbsRyIKI+dyb+k2K/nkt3SH0cTHnb/KHol5uD4sIKEKRQS/qa84u5FH1IryxrPUrF9niRwIjohDwd1L

Qi/EEhfZFwJYsidPWTLuVD9n57axcAtoVt92FrPuPjDWbGkGpiyqOaxRtwIw03pDFt0ljEI3pGZHF

uS6gHn1S/PwL7E=

</DQ>

<InverseQ> **// crtCoefficient InverseQ = (1 / Q) mod P where P > Q is converted to base64**

ALBLGPM+b7xsgHUw18qsAzuMmDEDASvS1N01ybxDlgDbLihEkXPYO/Mt1jzKedooaNOWMBb

TA6BLKTnqzV5zql20jOzF5kDWT6EwAkHVInoi7//gA0WmcNntBaYbkaqwDl81xTEzNH1irvvnWzc

wck+k63ttBzJl2vujGPYl31M=

</InverseQ>

// Private Exponent in base64

<D>

RN6wVRI/LaleR6lc3hvhZ/5hZ5FVYh6h+qSibgQ5IKOd00NqGV6MLi7ANvRd8RHo64O34qBVlXv

P4PSTUdr/+LyYCgz4lYmEwLJK9N/lV/w11TDfEqLMK1YjgzXFGeLPqUQtSyX14zrVxNydBptjntT7

o77fnZOVBBDvHpN6imHs3yM3eWP65oFOPEKDDFThUwc8tsfmzqPbIG0fvMCZdoUVli2BC2iDN

2pKDi7NPGmwt8rz4lJ5lcF1Jwc8hNICSqzshUGSNMNrLIyizf1DqS2exPqWTaLaNjUUkCQaDegE

yirvo7Y7o3Lr8Vxj+K3BYvuRIeXkazxMKbS1zjn1sQ==

</D>

</RSAKeyValue>

h. **Step 8: Creation of Public Key:** It is also created on the same pattern.

```
<RSAKeyValue>
   <Modulus>…</Modulus>
   <Exponent>…</Exponent>
</RSAKeyValue>
```

4.7.1 ElGamal Encryption - Flow of Events

ELGamal Key generation has a different flow of events then RSA.

a. **Step-1: Generation of Keys:** Client produces a proficient narrative of a multiplicative-cyclic-group G having order q using generator g and then chooses random x from $\{0, \ldots, q-1\}$.

$h = g^x$ is computed next.

h, along with its description is published as a **public key** and x is retained as a **private key** that has to be kept secure.

b. **Step 2: Transferring file to the Server:** While file F is transferred tags are also created and transferred to the Server *S*. The code snippet for carrying out this process looks like:

```
while (count < file.Length)
{


   byte[] buffer=bReader.ReadBytes(2048);


  MetaDataTag tag = new MetaDataTag();
  tag.StartIndex = i * 2048;
  tag.Length = buffer.Length;
  tag.Tag = GenerateTags(encoder.GetString(buffer));
  tag.Block = encoder.GetString(buffer);

  Tags.Add(tag);

   clientStream.Write(buffer, 0, buffer.Length);
   count += 2048;
   i++;
}
```

Figure 4.17: Transferring File to Server

c. **Step 3: Creating Tags:** In the current implementation since the bit size has been taken to be 2048 which is why the input file has been divided into blocks of size 2048 bits. The tags contain following information.

(1) ***Block number***

(2) ***Block Start offset***

(3) ***Block End offset***

d. **Step 4: Creating Association:** Once the tags have been created they are transferred to the server. When server receives the Tags ***Tm*** and File ***F'*** it stores

the modified file ***F'*** and Creates and stores Association ***A***. **S** contains modified file ***F'*** and association. The very purpose of this scheme is to ensure data integrity. For that matter Client ***C*** generates a challenge ***chal***.

e. **Step 5: Generating Challenge:** Challenge is generated as follows:

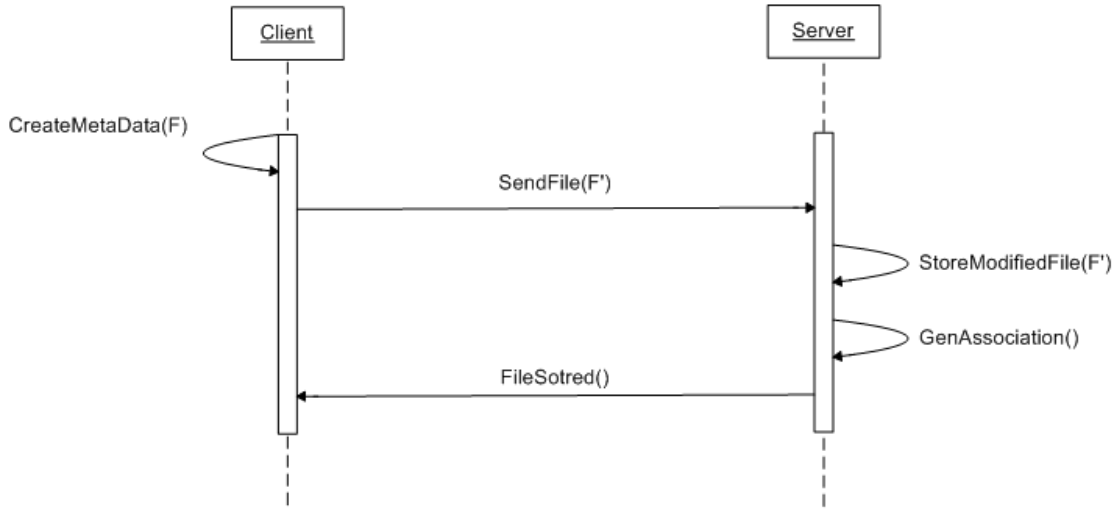


Figure 4.18: Transferring Metadata

f. **Step 6: Verifying user:** ***C*** sends this challenge to ***S*** which upon receipt of challenge authenticates the user. When server receives the challenge ***chal*** it first checks if this was the valid user who has initiated the challenge. The procedure to verify user is as follows.

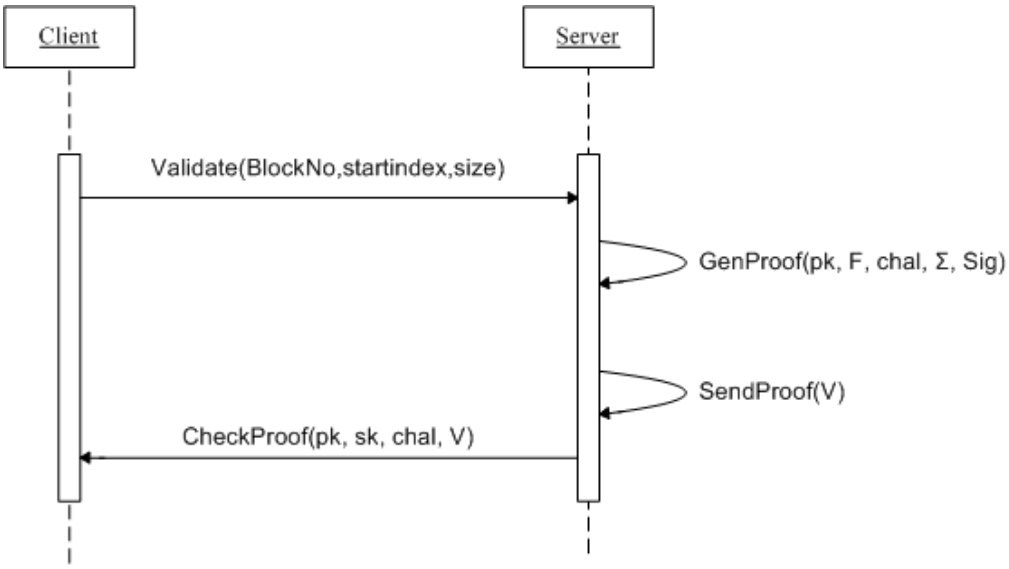


Figure 4.19: Verifying User

4.8 METHODS

Authentic Client-Provable Data Possession Scheme (AC-PDP) *It is a combination of 6 polynomial time algorithms* **(*KeyGen, TagBlock, GenAssociation, GenSignature, GenProof, CheckProof*)** *as under:*

a. ***KeyGen*** $(1k) \rightarrow (pk, sk)$**:** A probabilistic algorithm for generating keys which is used by the client for setting up the PDP scheme. It takes ***k*** as an input which is the security parameter, and its output consists of both the private and public keys (***sk***, ***pk***).

b. ***TagBlock(sk,m) → Tm:*** A procedure that is run by the ***C*** for generation of metadata that will be used for verification. It takes file block ***m*** and secret key ***sk***, and its output is the metadata file ***Tm***.

c. ***GenAssociation (F, ID, Pwd, email, exptime, passphrase) → A:*** A routine which is run by the ***S*** for creating an association ***A*** between file ***F*** and its initiator client ***C***. It takes File *F* and user attributes like *ID, pwd, email, exptime* and passphrase.

d. ***GenSignature (Pwd, email, exptime, passphrase) → Sig:*** This is a signature generation procedure that takes user attributes and creates digital signatures.

e. ***GenProof (pk, F, chal, Σ, Sig, atrList) → V:*** A routine for generation of proof-of-possession that is run by the server. Public key ***pk***, ordered collection ***F*** of blocks, challenge ***chal***, ordered collection ***Σ*** (which is the verification metadata corresponding to the blocks in ***F)***, digital signatures ***Sig*** and client's attribute list ***atrList*** are used as its input. It returns a proof of possession ***V*** for the blocks in ***F*** after verifying the user, and is determined by the challenge ***chal***.

f. **CheckProof(pk, sk, chal, *V*) → *{"success", "failure"}:*** Used for validation of proof-of-possession by the client. Public key ***pk,*** a secret key ***sk***, a challenge ***chal*** and a proof of possession ***V*** are used as its inputs. Its output determines if ***V*** is accurate proof of possession of blocks limited through ***chal***.

4.9 SUMMARY

In this chapter the architecture and implementation details of the newly proposed **A**uthentic **C**lient **P**rovable **D**ata **P**ossession have been described in detail. **AC-PDP** is a complete security protocol that fulfils the three most important requirements of security, namely confidentiality, authentication and integrity. It has been designed by using asymmetric encryption scheme along with digital signature scheme. A user authentication has been incorporated for the successful operation of this security protocol. The algorithm results in a complete protocol that takes into account untrusted servers as well as untrusted clients.

CHAPTER 5

TESTING AND EVALUATION OF AC-PDP

5.1 INTRODUCTION

AC-PDP is an ample protocol that provides high levels of security for resource over cloud. To finalize the protocol it has to be tested and evaluated in relation to a wide range of parameters. Since the protocol is designed for achieving security in cloud environment, it has to be tested against security features in cloud. Setting up a real cloud and making conditions for testing isn't free these days so a simulation has been made exactly like it is in cloud environment. As cloud is a client server environment at the end of the day, a TCP-Client and TCP-Server class has been implemented to take results for the proposed security protocol in terms of security and performance parameters. AC-PDP cannot be tested and evaluated in total isolation. It has to be tested and evaluated in comparison with other existing protocols.

5.2 AC-PDP TESTING ENVIRONMENT

To formally test and evaluate **AC-PDP** a proper environment had to be established. This environment provides support for visualization and result analysis.

5.2.1 Simulator

The simulator for testing **AC-PDP** has been implemented in dot Net 4.0. It is based on two modules, a **TCP-Client** and a **TCP-Server** class. The implementation works in the same manner like cloud does. Here is how **TCP-Client** and **TCP-Server** work.

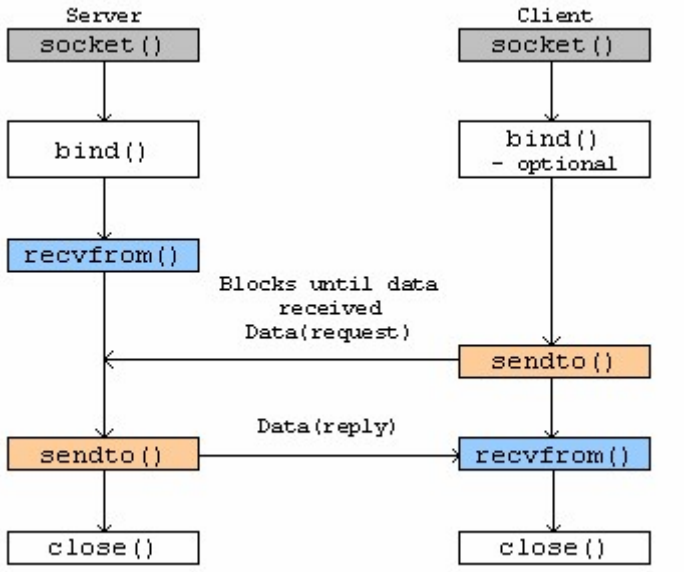


Figure 5.1- Working of TCP-Client and TCP-Server *[24]*

Client and server first connect to each other using local host. Then the client class sends modified file and metadata to server which is stored at server. Server also computes an association between user and file and stores it in its DB. Late on client calls upon server for proof of integrity which is computed and sent back to client on challenge request.

5.3 AC-PDP TESTING

Rigorous testing of **AC-PDP** is through the various test vectors that provide varying inputs to the simulator. Fundamentally **AC-PDP** is designed to take a file of any size as input and ensure its integrity over server. Each test vector is unique but fundamentally operates on three inputs i.e. Encryption scheme, file size and secret key. Varying these three inputs provides varying outputs. The purpose of this testing is to identify abnormal conduct and also to identify the usefulness of the whole system.

5.3.1 Test Vector-I (File Size versus Tag Size)

Tables 5.1 shows the tag size generated from files of different sizes keeping the key size constant which is 2048 in this case. It has been observed that changing the cryptosystem does not put any effect on the tag size but the input file size does matter. With increasing files size, the size of the tags that are being generated also increase. For a file of size 1mb AC-PDP(RSA) generates tags of size 9kb, AC-PDP(ElGamal) also generated 9kb file and PDP has also shown same kind of result for a file of size 1mb. As the file size increases, all three algorithms show a similar trend in the resultant tag file's size.

File Size (MBs)	Tag Size(KBs)		
	AC-PDP(RSA)	AC-PDP(ELGamal)	PDP(RSA)
1 mb	9 kb	9 kb	9 kb
2 mb	18 kb	18 kb	18 kb
4 mb	37 kb	37 kb	37 kb
8 mb	75 kb	75 kb	75 kb

Table-5.1 Comparison of File Size and Tag Size

5.3.2 Test Vector-II (File Size versus Number of Blocks)

Table 5.2 shows a relationship between file sizes and the number of blocks created as a result of applying a key of size 2048 bits. The behavior observed by the implementation suggests that there is no difference between the number of blocks created in each case

which implies that block number is not dependant upon file size; however by increasing file size, number of resulting blocks is also increasing. This behavior is common in all the algorithms. For file sizes 1mb, 2mb, 4mb and 8mb; number of blocks that are generated by each algorithm is 517, 1034, 3103 and 7231 respectively.

File Size (MBs)	Number of Blocks (n)		
	PDP	AC-PDP(RSA)	AC-PDP(ELGamal)
1 mb	517	517	517
2 mb	1034	1034	1034
4 mb	3103	3103	3103
8 mb	7231	7231	7231

Table- 5.2 File Size versus Number of Blocks

5.3.3 Test Vector-III (Tag Size versus Number of Blocks)

This test vector is the result of the test case conducted to relate tag size and number of blocks. Table 5.3 indicates that when there is an increase in tag size, there is a gradual increase in the corresponding number of blocks. However, no difference was observed among the manners shown by the three protocols.

Tag Size(kbs)	Number of Blocks(n) PDP	Number of Blocks(n) AC-PDP(RSA)	Number of Blocks(n) AC-PDP(ELGamal)
8.46	517	517	517
18	1034	1034	1034
110	3103	3103	3103
131	7231	7231	7231

Table 5.3 Comparison of Tag Size and Number Of Blocks

5.3.4 Test Vector-IV (Preprocess versus Challenge for different block sizes)

Here is where the difference lies. Since AC-PDP has an additional step added to it for client verification, it will certainly take longer pre-process time as compared to the previous PDP schemes. It will take time in signature creation and encrypting the signatures. Another behavior has been seen while conducting this test case is that ELGamal being a slow encryption algorithm (but of course a very secure one compared to RSA), takes longer time than RSA in encrypting the data, so preprocess time for AC-PDP with ELGamal has values quite larger than AC-PDP with RSA. Table 5.4 and 5.5 indicate that processing time is different for the same block size but challenge time remains the same. PDP has much lesser preprocessing time than AC-PDP(RSA) but the

challenge time for both is same. However, Table 5.6 shows that both the preprocessing time and challenge time is different in case of AC-PDP (ElGamal). Table 5.6 shows that AC-PDP (ElGamal) shows a much higher reading compared to the other two.

Block Size	Preprocess Time	Challenge Time (ms)
1024	1019	610103
2048	1218.75	610013
4096	1321	39111

Table-5.4 A comparison of block size, preprocess time and challenge time with PDP

Block Size	Preprocess Time	Challenge Time (ms)
1024	1250	610103
2048	1093	610013
4096	1046.87	39111

Table-5.5 a comparison of block size, preprocess time and challenge time with AC-PDP (RSA)

Block Size	Preprocess Time	Challenge Time (ms)
1024	1296.875	872483.125
2048	3218.75	872483
4096	18790.87	40671.87

Table-5.6 a comparison of block size, preprocess time and challenge time with AC-PDP (ElGamal)

5.4 PROCESSING TIME RESULTS

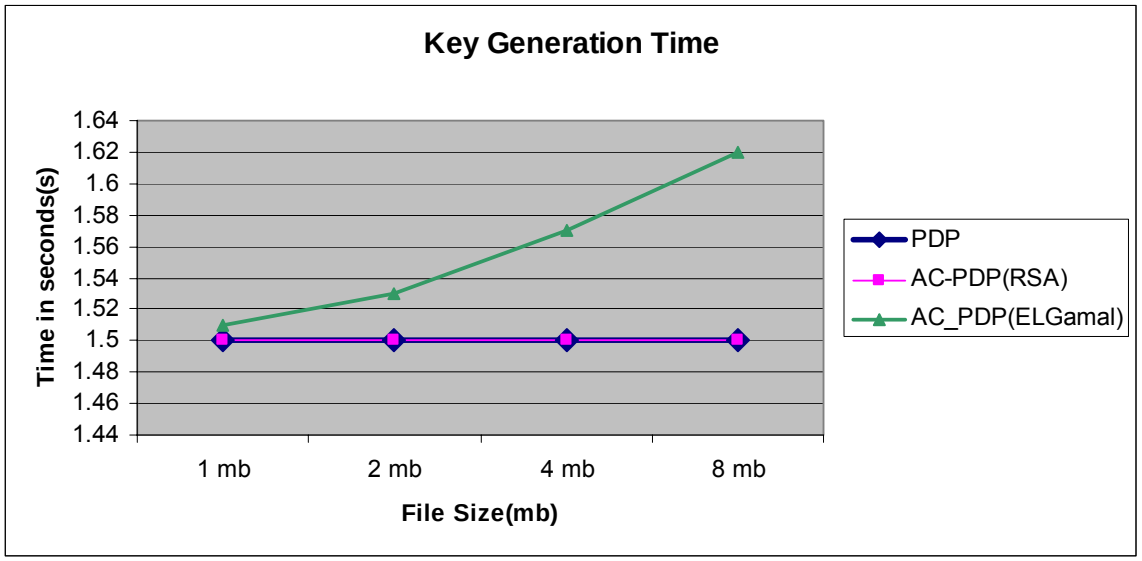


Figure 5.2- A comparison of Key Generation time

Figure 5.2 shows the key generation time taken by all three algorithms under observation. Key generation time is specific to the encryption scheme used hence it has got nothing much to do with the proposed PDP and already existing PDP. Key generation is directly proportional to the encryption scheme used as can be seen in Figure 5.2.

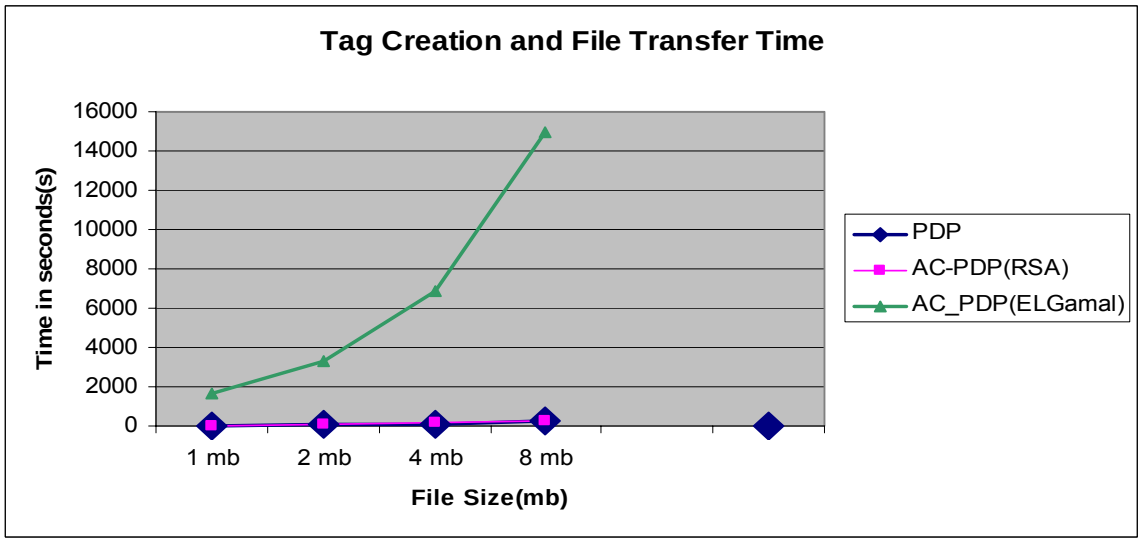


Figure 5.3- A comparison of Tag Creation and File Transfer time

Before file is transferred to the server, tags are created first and metadata information is sent to the file. In AC-PDP two encryption algorithms have been used namely RSA and ElGamal. ElGamal happens to be a very slow process where as it is thought to be stronger than RSA. Since PDP and AC-PDP (RSA) both have RSA for key generation, the graph shows close readings for the two whereas ElGamal raises way above the two because of its slower speed. Careful analysis clearly shows that processing time of all three methods is around 0.5 seconds when file size is 1mb. There come the difference in AC-PDP (RSA) and AC-PDP (ElGamal) when file size reaches to 8mb. RSA time remains between 0.05 to 0.1 seconds whereas ElGamal time increases to 0.22 seconds. Figure 5.4 shows this observation.

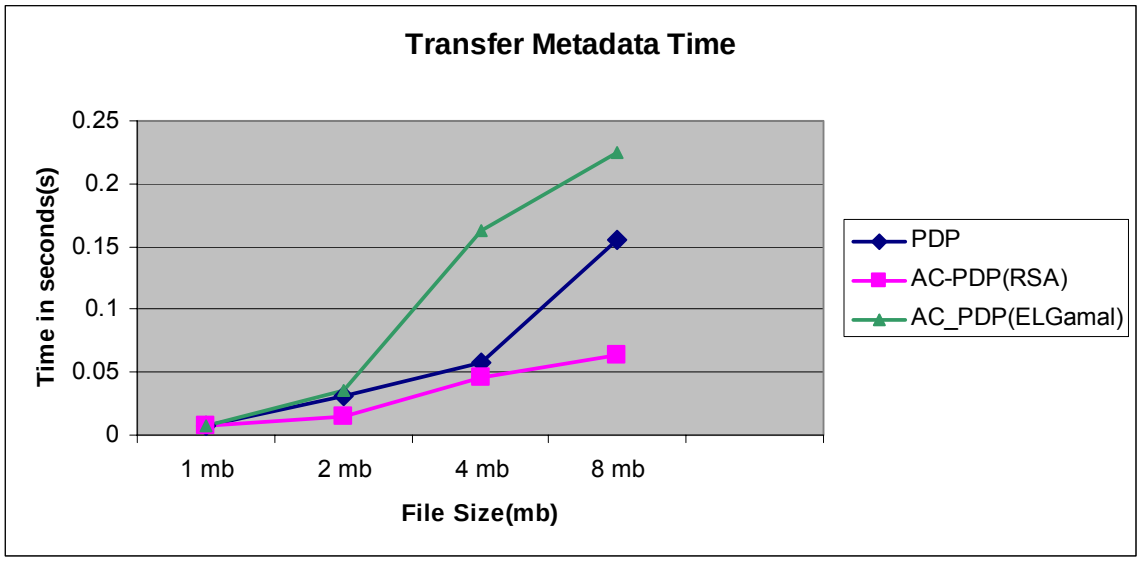


Figure 5.4 - PDP, AC-PDP (RSA) and AC-PDP (ElGamal) Metadata transfer time

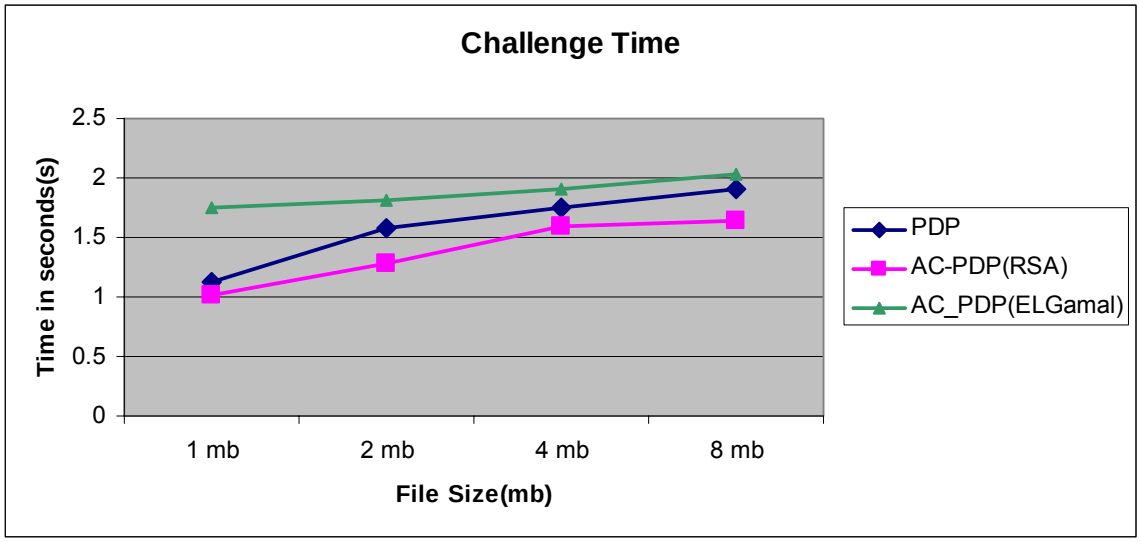


Figure 5.5 - PDP, AC-PDP (RSA) and AC-PDP (ElGamal) Challenge time

Challenge time for AC-PDPs is slightly higher because it requires authenticating the client before generating proof. According to Figure 5.5 AC-PDP takes longer than PDP which is because of the signature verification at server end before generating proof. While challenging certain encryption and decryption is also performed; so the time is again dependant upon the encryption scheme used.

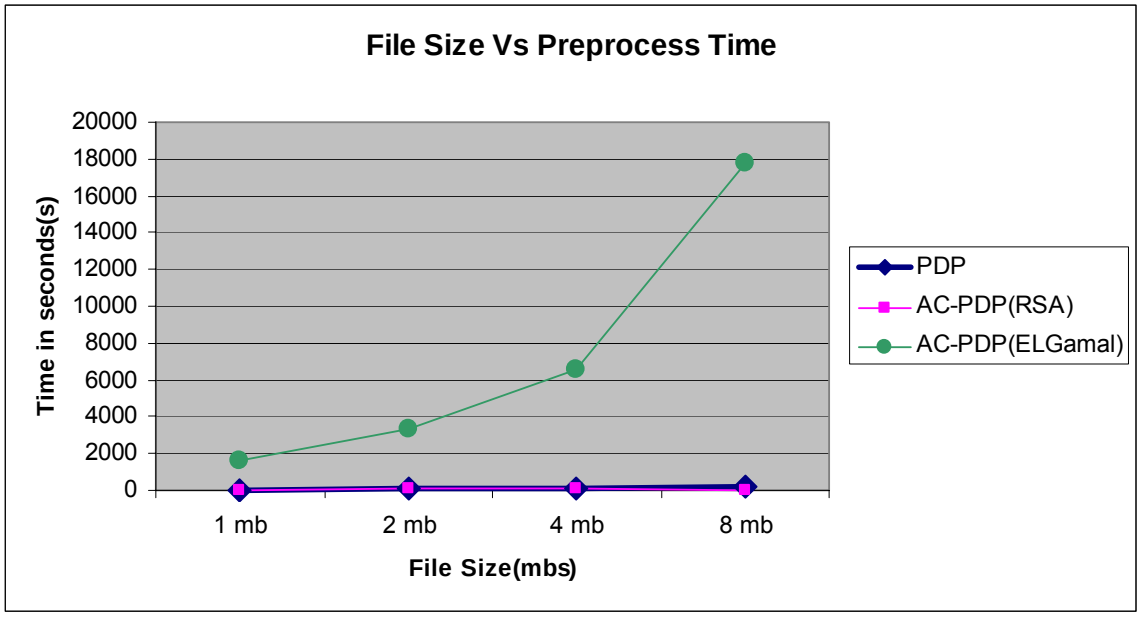


Figure 5.6 – File Size Vs Pre Process time with PDP, AC-PDP (RSA) and AC-PDP (ElGamal)

The preprocess time for ElGamal is higher than the other two because of its slower key generation mechanism. As seen in Figure 5.6 ElGamal's graph goes up. This is a known fact about ElGamal and also one of its disadvantages that its encryption is slow. There is certain situation because of which ElGamal encryption happens to be slow. When an ElGamal key pair is generated, an option of specifying a prime modulus is always there. If the prime modulus is not specified, one will be generated. Since the modulus must be a safe prime which means a prime ***p*** such that ***(p-1)/2*** is also prime, and those are much rarer than regular primes, it takes a long time. So it can be concluded that the root cause because of which key generation takes longer time if ***p*** and ***g*** are not provided, is that the key generator first generates ***p*** and ***g*** values for the given key size and then generates the key which is expensive. ***p*** and ***g*** values can be reused, although it should be known that in trying to attack ElGamal private key the most expensive computation depend on the value of ***p***, so these computations can be reused for any key derived from the same ***p***

value. For large values of ***p*** this is still an awful lot of work. Taking care of these tiny things can lead to a faster ElGamal key generation. Using existing well known safe prime is a good idea.

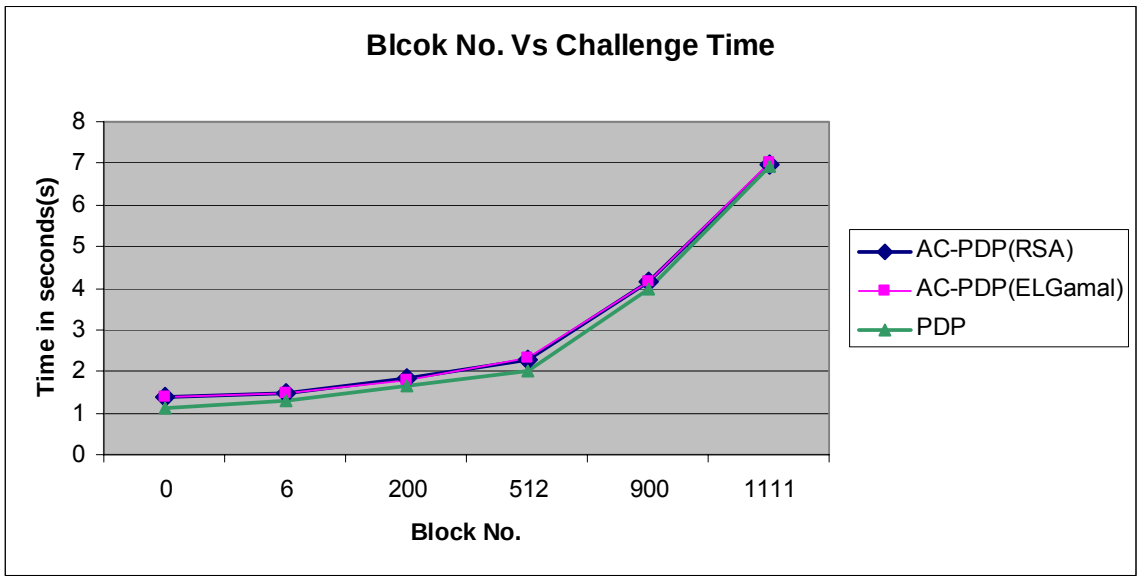


Figure 5.7 – Block No Vs Challenge Time with PDP, AC-PDP (RSA) and AC-PDP (ElGamal)

Figure 5.7 shows that as we go for validating late blocks in the file; the response time keeps on increasing. The behavior is uniform in all three algorithms. For PDP, challenging block '0' requires 1.39 seconds, block '6' requires 1.53 seconds and block 111 needs 6.992 seconds. The behavior has been seen to be consistent in the other two algorithms as well.

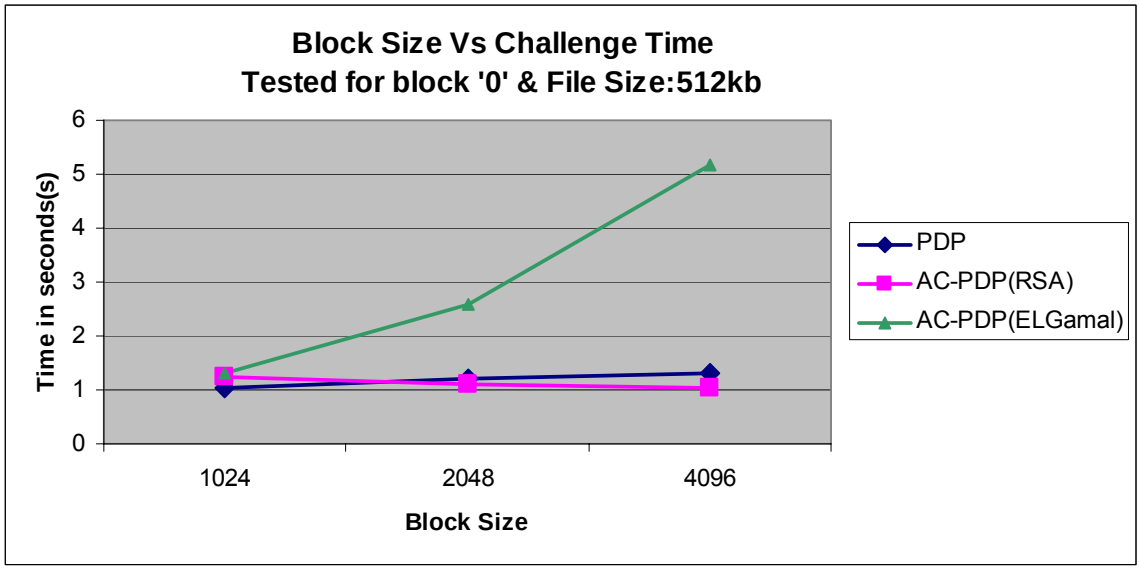


Figure 5.8 – Block Size Vs Challenge Time with PDP, AC-PDP (RSA) and AC-PDP (ElGamal)

Block size is what we define flexibly. Figure 5.8 demonstrates this. The algorithm has been tested upon three block sizes **'1024', '2048'** and **'4096'** and a variety in results has been observed. The time for challenge will increase with increasing block size. Challenging block with 1024 bits takes 1.01, 1025 and 1.29 seconds for PDP, AC-PDP (RSA) and AC-PDP (ElGamal), whereas if the block size increases to 4096 bits per block the algorithms are going to take 1.32, 1.04 and 5.18 seconds in the same order.

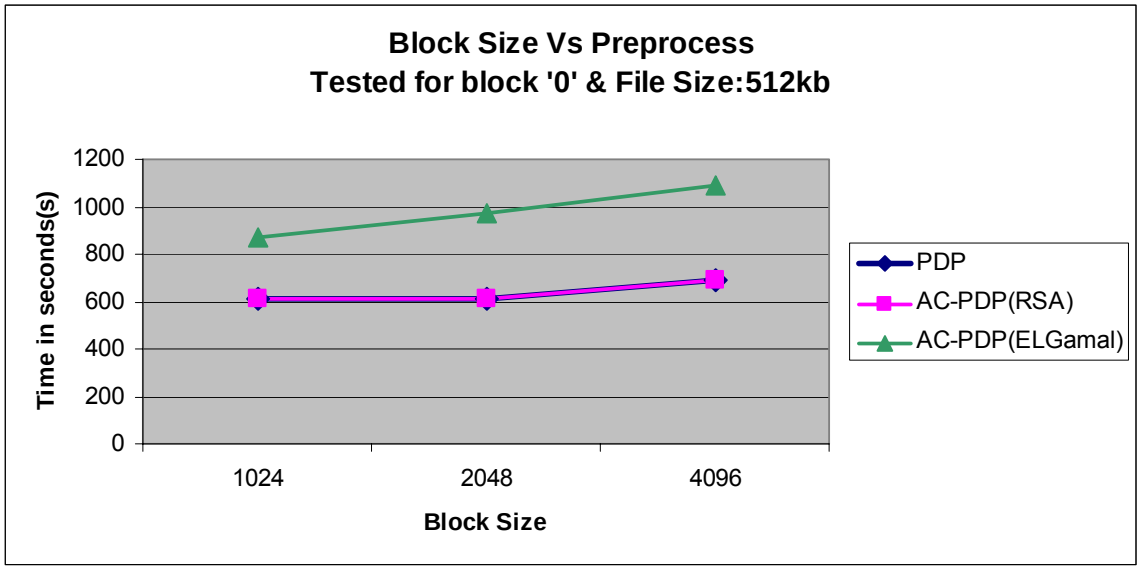


Figure 5.9 – Block Size VS Pre Process Time with PDP, AC-PDP (RSA) and AC-PDP (ElGamal)

There also is some logic associated with block size and preprocess time. ElGamal being slow at the encryption end takes longest time than the other two. Though the time for preprocess increases as the block size is increased but the transition is almost negligible as shown in Figure 5.9. ElGamal has been observed to consume longest time in preprocess that refers to its slow encryption part.

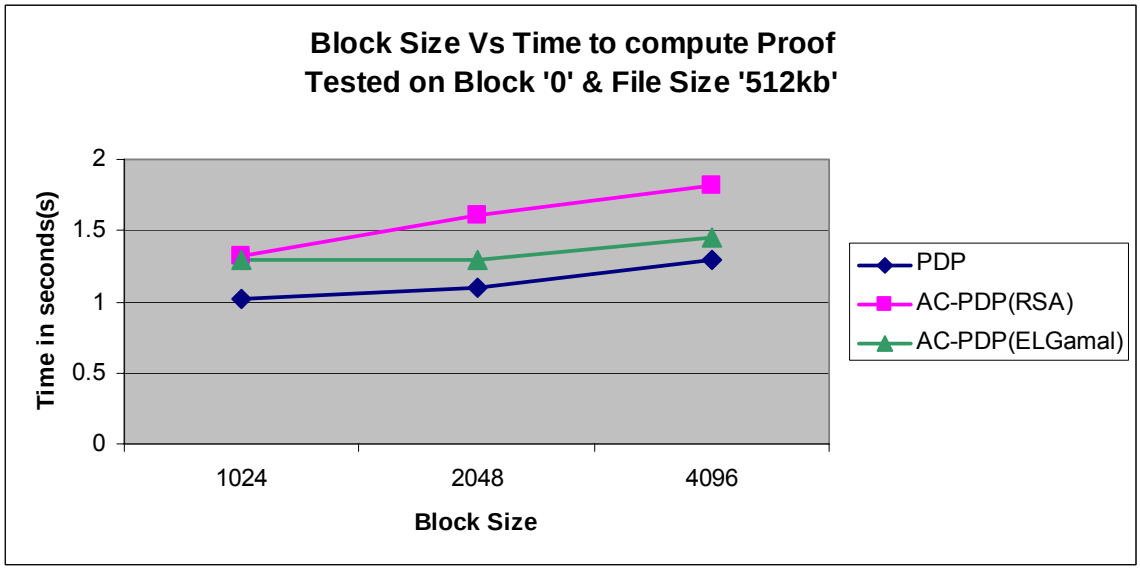


Figure 5.10 – Block size Vs Time to Compute Proof with PDP, AC-PDP (RSA) and AC-PDP (ElGamal)

Increasing block size also causes an increase in the resultant time to compute proof but as in the case of block size versus challenge time, the transition is negligible here as well. So it can be said that changing the block size shall not effect time to compute proof as well as challenge time. However, the transition between the three algorithms is less as compared to preprocess time. Figure 5.10 sums it up.

5.5 STORAGE SPACE RESULTS

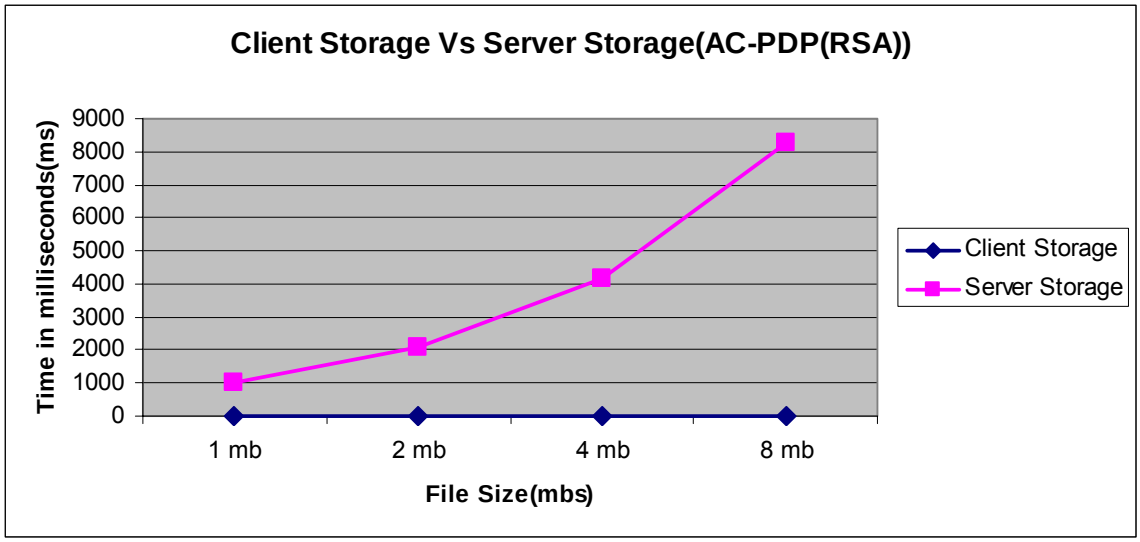


Figure 5.11 – Client Storage Vs Server Storage with AC-PDP (RSA)

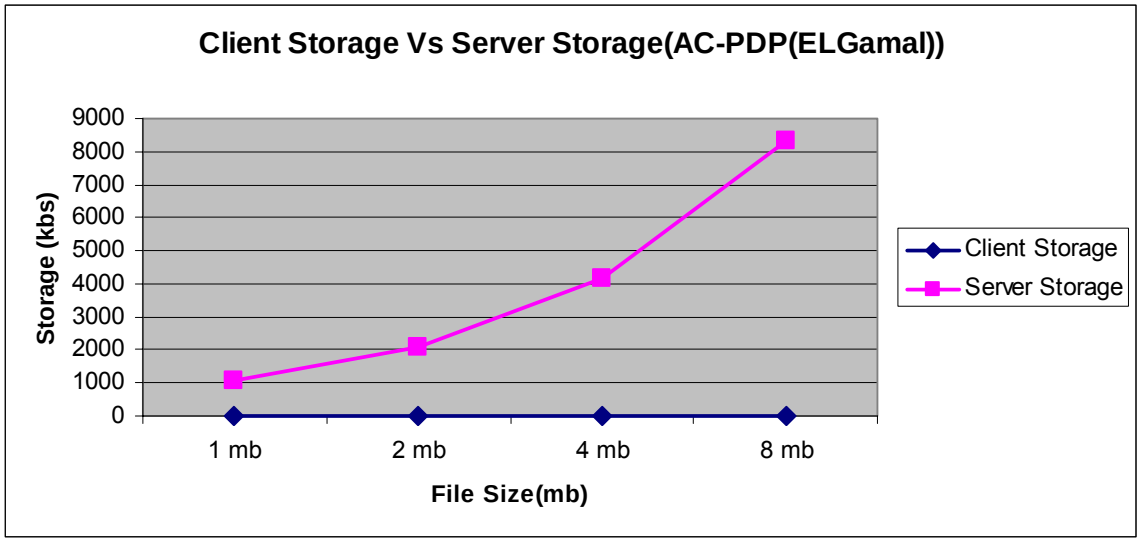


Figure 5.12 – Client Storage Vs Server Storage with AC-PDP (ElGamal)

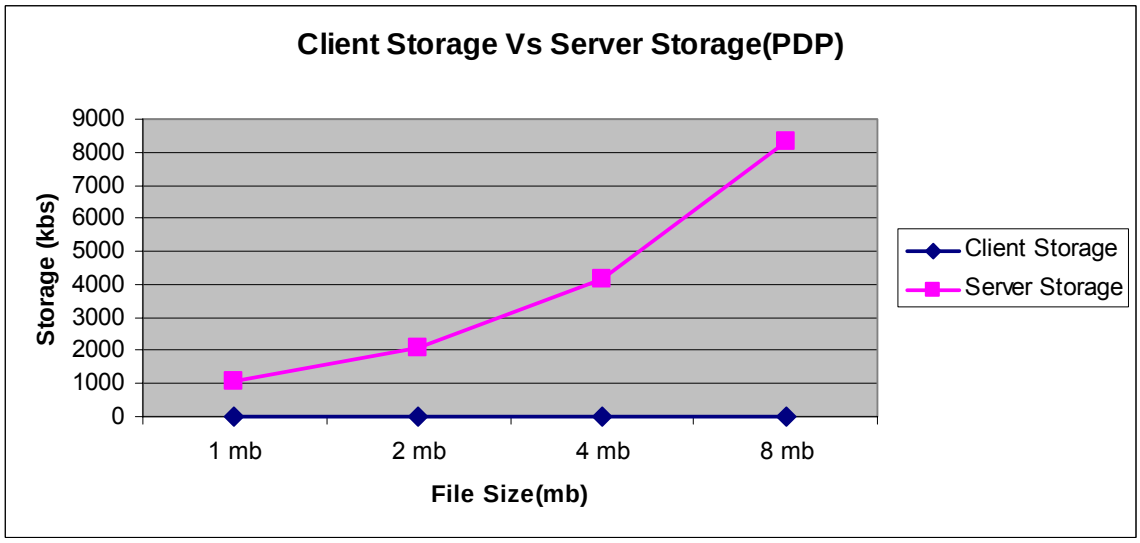


Figure 5.13 – Client Storage Vs Server Storage with PDP

When we talk about storage space, be it **PDP, AC-PDP (RSA)** or **AC-PDP (ElGamal),** server is always going to consume larger space than client. Since client deletes the file after transferring to the server, also in case of AC-PDPs no association file has to be stored at client side. So the over head of storing many extra elements is reduced at client's end.

ElGamal scheme suffers from expansion of message by twice the size in the encryption process. This behavior has also been observed in my implementation. The concept is illustrated in Figures 5.11, 5.12, and 5.13 above.

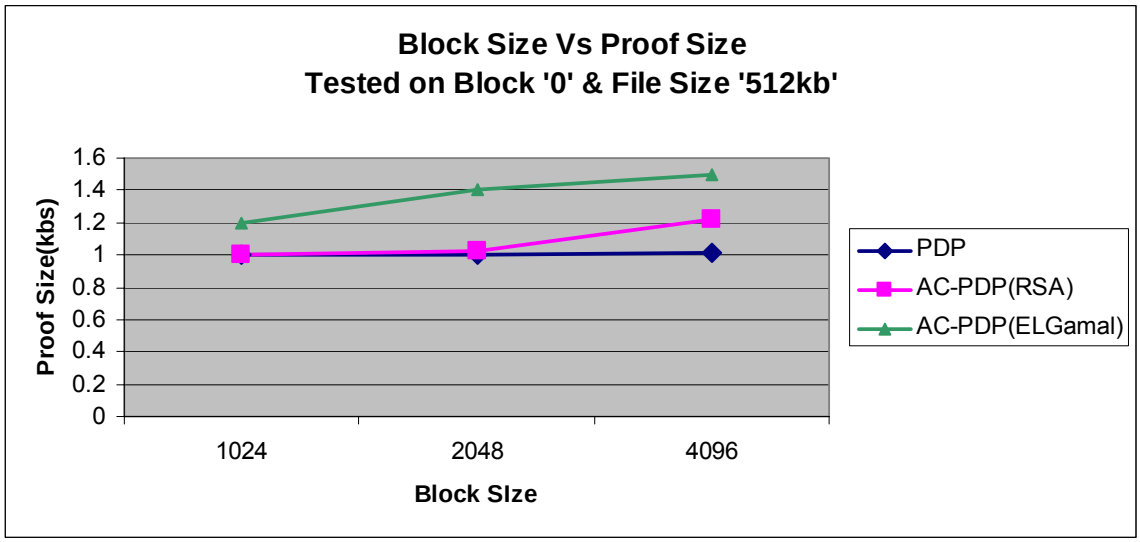


Figure 5.14 – Block size VS Proof Size with PDP, AC-PDP (RSA) and AC-PDP (ElGamal)

Bigger the block, bigger would be the proof size. Block size and proof size happen to be directly proportional to each other. The results taken for a block size of 1024 show a smaller sized proof, which gives the conclusion that proof size varies as block size varies. For a block sized 1024 PDP and AC-PDP (RSA) generate a proof of size 1 kb whereas AC-PDP (ElGamal) results into a proof of size 1.2 kb which is absolutely comparable to the other two algorithms. For a block of size 2048, slightly bigger sized proofs are produced with all three algorithms and so is the case with a block size 4096. The implementation in this dissertation concludes that increasing block size will affect sequent proof size.

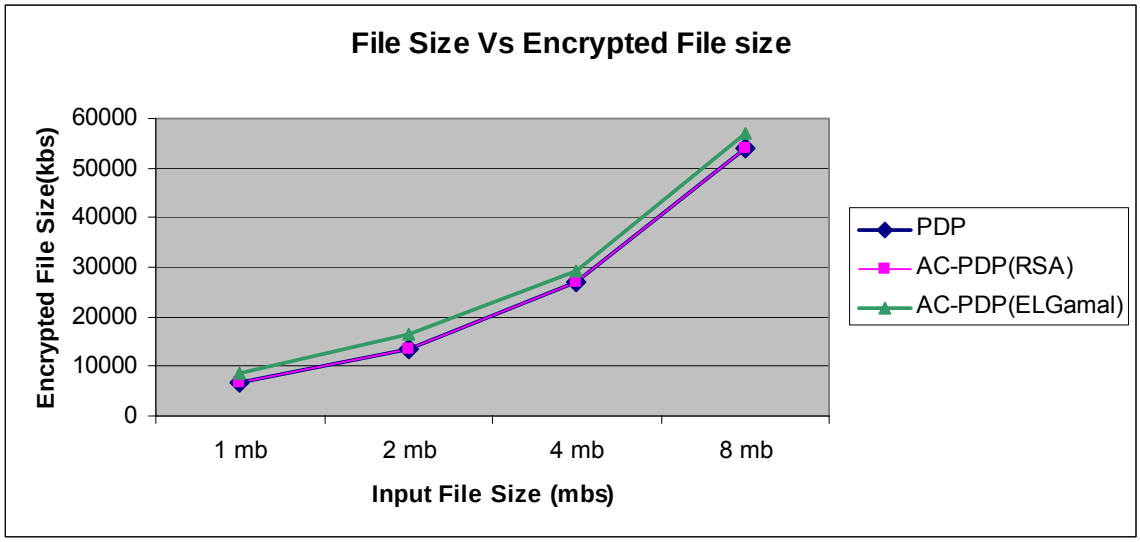


Figure 5.15 – File size Vs Encrypted File Size with PDP, AC-PDP (RSA) and AC-PDP (ElGamal)

Encrypted file size with ElGamal happens to be the larger. The size of the encrypted data doubles once encrypted. A file originally sizing 1024 kbs the resultant encrypted file sizes 8785 kbs, a 2048 kbs file results into a 16569 kbs and so on, with ElGamal.

5.6 SUMMARY

To fully test AC-PDP it has been evaluated from various aspects like processing time, space utilization and security provision. The results have been obtained by implementing a client/server environment in .Net. Extensive testing of AC-PDP has shown that this new protocol is complete and ensures almost complete security over cloud network, in terms of integrity. AC-PDP has been evaluated in comparison with other protocols like PDP. The results demonstrate that AC-PDP has all the properties that are required by a highly acknowledged security protocol in cloud networks.

Chapter 6

CONCLUSIONS & FUTURE WORK

6.1 CONCLUSIONS

As data storage requirement grow with each passing day, more and more customers are switching towards storage in the 'Cloud' using cloud service providers (CSPs), whereby resources are provisioned and de-provisioned as needed. While cloud storage provides the benefit of global data access, it does bear the burden of data security and access control. If data access is wide-spread, then security issues are bound to arise. Security in this context means confidentiality of information coupled with progressive features such as authentication and integrity. Most research effort was however focused on security of data over untrusted server, because of the fact that provision of security was considered exclusively to be the server's concern. Emerging trends in cloud security however promise high levels of security without compromising authentication for untrusted clients.

This research work was focused on the development of a Complete Security Protocol ***(AC-PDP)*** that would be widely accepted and easily deployed. Having studied the incorporation of client authentication within the existing security protocol, this study eliminates the need / overheads of running separate routines for client authentication. The comprehensive security protocol presented ***(AC-PDP)*** would cater for untrusted client in addition to the server. Furthermore, in order to validate this research, an extensive comparative analysis between proposed scheme and other existing PDP schemes (being

the only scheme that had previously addressed the same problem). Results have proved that the proposed scheme overcomes the fundamental limitations of the PDP model, which does not allow only authorized users to seamlessly access the owner's files.

In this dissertation, an attempt has been made to develop a new protocol that provides features of confidentiality and integrity without compromising the client's authentication scheme over the Cloud.

6.2 FUTURE WORK

There is much more to be done to refine this model and further improve its performance and make it dynamically changeable. Few highlighted areas are stated below:

a. **Performance Optimization:** Although ACPDP has been fully tested and is considered as a complete solution, best speed limits are yet to be achieved. The protocol can be finalized by implementation of a complete cycle in smallest possible time interval, which requires refinement and improvement of proposed model pertaining to encryption phase. This suggests that by employing a fast encryption scheme will contribute towards achieving best speed results.

b. **Memory Optimization:** ACPDP has demonstrated good results in terms of memory and time requirements as per the implementation done so far. For further optimization the encrypted file size and proof size can be tweaked and adjusted to obtain improved results and performance. Keeping key size as per requirements will fine tune the memory requirements.

c. **Asymmetric Encryption Techniques:** ACPDP is fundamentally a security protocol based on the RSA and ELGamal encryption hence space for research is available by combining AC-PDP architecture with any encryption scheme other than RSA and ELGamal to acquire the best possible results.

d. **Symmetric Encryption Technique:** Symmetric algorithms have the advantage of not consuming too much computing power. ACPDP can also be extended to propose and implement a protocol that attempts to incorporate symmetric encryption techniques for achieving optimization in terms of time and computing power but in that case security has to be negotiated.

e. **Hybrid Encryption Techniques:** AC-PDP can also be implemented with a combination of symmetric and asymmetric encryption techniques. This way, advantages of both the techniques can be taken. The asymmetric keys are used for authentication and after this have been successfully done; one or more symmetric keys are generated and exchanged using the asymmetric encryption.

BIBLIOGRAPHY

[1] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "***Provable Data Possession at untrusted stores",*** In *CCS*, pp. 598–609, 2007.

[2] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "***Scalable and Efficient Provable Data Possession,*** in *SecureComm*, pp. 1–10, 2008.

[3] R. Tamassia, "***Authenticated data structures***", In *ESA*, pp. 2–5, 2003.

[4] M. T. Goodrich, C. Papamanthou, R. Tamassia, and N. Triandopoulos. Athos, "***Efficient authentication of outsourced file systems***", In *ISC*, pp. 80–96, 2008.

[5] H. Shacham and B. Waters, "***Compact Proofs of Retrievability",*** In ASIACRYPT, pages 90–107, 2008.

[6] S. Y. Ko, I. Hoque, B. Cho, and I. Gupta***, " On availability of intermediate data in cloud computations***", In Proc. 12th Usenix Workshop on Hot Topics in Operating Systems (HotOS XII), 2009.

[7] K. Zeng, **"*Publicly verifiable remote data integrity*",** in *ICICS*, 2008, pp. 419–434.

[8] D. L. G. Filho and P. S. L. M. Barreto, "***Demonstrating data possession and uncheatable data transfer***", Cryptology ePrint Archive, Report 2006/150, 2006.

[9] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, ***"MR-PDP: Multiple-Replica Provable Data Possession,"*** in *28th IEEE ICDCS*, 2008, pp. 411–420.

[10] R. Rivest, A. Shamir, and L. Adleman, ***"A method for obtaining digital signatures and public-key cryptosystems",*** *Commun. ACM*, vol. 26, no. 1, 1983.

[11] F. Li, M. Hadjieleftheriou, G. Kollios, and L. Reyzin, ***"Dynamic authenticated index structures for outsourced databases",*** in *SIGMOD '06: Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, New York, NY, USA, 2006, pp. 121–132.

[12] Ramgovind S, Eloff MM, Smith E, "***The Management of Security in Cloud Computing",*** in Information Security for South Africa (ISSA), 2010

[13] R. Curtmola, O. Khan, and R. Burns, ***"Robust remote data checking",*** in *StorageSS '08: Proceedings of the 4th ACM International Workshop on Storage Security and Survivability*, New York, NY, USA, 2008, pp. 63–68.

[14] Cong Wang and Kui Ren, Illinois Institute of Technology Wenjing Lou, ***"Toward Publicly Auditable Secure Cloud Data Storage Services",*** **in** IEEE Network, The Magazine of Global Internetworking archive Volume 24 Issue 4, July-August 2010

[15] K.Mukherjee, G.Sahoo, ***"A Secure Cloud Computing",*** in Recent Trends in Information, Telecommunication and Computing (ITC), 2010 International Conference ,12-13 March 2010

[16] *Xiaojun Yu, Qiaoyan Wen,* ***"A VIEW ABOUT CLOUD DATA SECURITY FROM DATA LIFE CYCLE"****, in* Computational Intelligence and Software Engineering (CiSE), 2010.

[17] Youngmin Jung, Mokdong Chung, *"**Adaptive Security Management Model in the Cloud Computing Environment**"*, in Advanced Communication Technology (ICACT), 2010.

[18] Balachandra Reddy Kandukuri, Ramakrishna Paturi V, Dr. Atanu Rakshit, **"Cloud Security Issues",** in 2009 IEEE

[19] Uma Somani, Kanika Lakhani, Manish Mundra, ***"Implementing Digital Signature with RSA Encryption Algorithm to Enhance the Data Security of Cloud in Cloud Computing",*** in 2010 IEEE.

[20] http://www.transactionlevelanalysis.com/the-cloud/

[21] http://www.esri.com/technology-topics/cloud-gis/service-models.html

[22] http://cloudingworld.com

[23] http://cloudcomputingtechnologybasics.blogspot.com/2011/08/cloud-computing-deployment-models.html

[24] http://www.tenouk.com

[25] E. Mykletun, M. Narasimha, and G. Tsudik, ***"Authentication and integrity in outsourced databases,"*** *Trans. Storage*, vol. 2, no. 2, 2006.

[26] M. A. Shah, M. Baker, J. C. Mogul, and R. Swaminathan, ***"Auditing to keep online storage services honest,"*** in *HOTOS'07: Proceedings of the 11th USENIX workshop on Hot topics in operating systems*, Berkeley, CA, USA, 2007, pp. 1–6.

[27] F. Seb´e, J. Domingo-Ferrer, A. Martinez-Balleste, Y. Deswarte, and J.-J. Quisquater, ***"Efficient remote data possession checking in critical information infrastructures,"*** *IEEE Trans. on Knowl. and Data Eng.*, vol. 20, no. 8, 2008.

[28] Melissa Helgeson, ***"Security and Applications of ElGamal's Encryption Algorithm".***

[29] Zhang lianhong, Chen Hua, ***"Secuirty Storage in the Cloud Computing: A RSA-based Assumption Data Integrity Check without Original Data",*** *in Applied Mechanics and Materials (2011)*

[30] Ramgovind S, Eloff MM, Smith E, ***"The Management of Security in Cloud Computing",*** in Information Security for South Africa (ISSA), 2010

[31] Rocco Aversa, Beniamino Di Martino, Massimiliano Rak, Salvatore Venticinque, ***"Cloud Agency: A Mobile Agent Based Cloud System",*** in Complex, Intelligent and Software Intensive Systems (CISIS), 2010.

BIBLIOGRAPHY

[1] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "***Provable Data Possession at untrusted stores",*** In *CCS*, pp. 598–609, 2007.

[2] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "***Scalable and Efficient Provable Data Possessio",*** In *SecureComm*, pp. 1–10, 2008.

[3] R. Tamassia, "***Authenticated data structures***", In *ESA*, pp. 2–5, 2003.

[4] M. T. Goodrich, C. Papamanthou, R. Tamassia, and N. Triandopoulos. Athos, "***Efficient authentication of outsourced file systems***", In *ISC*, pp. 80–96, 2008.

[5] H. Shacham and B. Waters, "***Compact Proofs of Retrievability",*** In ASIACRYPT, pages 90–107, 2008.

[6] S. Y. Ko, I. Hoque, B. Cho, and I. Gupta***, " On availability of intermediate data in cloud computations***", In Proc. 12th Usenix Workshop on Hot Topics in Operating Systems (HotOS XII), 2009.

[7] K. Zeng, ***"Publicly verifiable remote data integrity",*** in *ICICS*, 2008, pp. 419–434.

[8] D. L. G. Filho and P. S. L. M. Barreto, "***Demonstrating data possession and uncheatable data transfer***", Cryptology ePrint Archive, Report 2006/150, 2006.

[9] R. Curtmola, O. Khan, R. Burns, and G. Ateniese, ***"MR-PDP: Multiple-Replica Provable Data Possession,"*** in *28th IEEE ICDCS*, 2008, pp. 411–420.

[10] R. Rivest, A. Shamir, and L. Adleman, ***"A method for obtaining digital signatures and public-key cryptosystems",*** *Commun. ACM*, vol. 26, no. 1, 1983.

[11] F. Li, M. Hadjieleftheriou, G. Kollios, and L. Reyzin, ***"Dynamic authenticated index structures for outsourced databases",*** in *SIGMOD '06: Proceedings of the*

2006 ACM SIGMOD International Conference on Management of Data, New York, NY, USA, 2006, pp. 121–132.

[12] Ramgovind S, Eloff MM, Smith E, "***The Management of Security in Cloud Computing",*** in PROC 2010 IEEE International Conference on Cloud Computing 2010

[13] R. Curtmola, O. Khan, and R. Burns, ***"Robust remote data checking",*** in *StorageSS '08: Proceedings of the 4th ACM International Workshop on Storage Security and Survivability*, New York, NY, USA, 2008, pp. 63–68.

[14] Cong Wang and Kui Ren, Illinois Institute of Technology Wenjing Lou, ***"Toward Publicly Auditable Secure Cloud Data Storage Services",*** **in** IEEE Network • July/August 2010

[15] K.Mukherjee, G.Sahoo, ***"A Secure Cloud Computing",*** in © 2010 IEEE

[16] *Xiaojun Yu, Qiaoyan Wen,* ***"A VIEW ABOUT CLOUD DATA SECURITY FROM DATA LIFE CYCLE****", in* 978-1-4244-5392-4/10/$26.00 ©2010 IEEE

[17] Youngmin Jung, Mokdong Chung, "**Adaptive Security Management Model in the Cloud Computing Environment**", in Feb. 7-10, 2010 ICACT 2010

[18] Balachandra Reddy Kandukuri, Ramakrishna Paturi V, Dr. Atanu Rakshit, **"Cloud Security Issues",** in 978-0-7695-3811-2/09 $26.00 © 2009 IEEE

[19] Uma Somani, Kanika Lakhani, Manish Mundra, ***"Implementing Digital Signature with RSA Encryption Algorithm to Enhance the Data Security of Cloud in Cloud Computing",*** in 978-1-4244-7674-9/10/$26.00 ©2010 IEEE.

[20] http://www.transactionlevelanalysis.com/the-cloud/

[21] http://www.esri.com/technology-topics/cloud-gis/service-models.html

[22] http://cloudingworld.com

[23] http://cloudcomputingtechnologybasics.blogspot.com/2011/08/cloud-computing-deployment-models.html

[24] http://www.tenouk.com