ISSN: 2278 – 7798
International Journal of Science, Engineering and Technology Research (IJSETR)
Volume 2, Issue 4, April 2013

# A Secure Mass Data Storage and Integrity Verification Scheme in Cloud Computing

**Nayana.S.R, VenifaMini G, Jeya A. Celin J**

***Abstract*— Cloud computing, hitherto a novel technology in modern computing provide the users accessibility to a gamut of resource rich pool of shared computing applications that can be accessed on demand. The cloud data storage service relieves the users from the burden of voluminous local data storage and their maintenance by outsourcing mass data to the cloud. However data outsourcing results in the users losing their physical control of data. The accuracy of data stored in cloud is of paramount importance and this was earlier accomplished by involving a trusted third party to check the integrity of the data. The introduction of a trusted third party may lead to unauthorized exposure of the stored data to other users and also tampering of data. The proposed work envisages an encrypted data integrity checking scheme without a trusted third party auditor to verify the authenticity of the data stored in the cloud. This ensures that the data owners have the prospect of securing their sensitive data. Data integrity can be assured by using ELGAMAL digital signature scheme and also provides data dynamics by Merkle B-tree algorithm. Extensive security and performance analysis unequivocally proves that the proposed scheme is highly secure and grossly efficient.**

*Index Terms*— **Authenticated password key exchange, Data integrity, ElGamal encryption, Merkle B-tree, Third party auditor.**

## I. INTRODUCTION

Cloud computing has ushered in major advancements to the IT industries, enabling clients to run their software applications in remote computing clouds where data storage and processing resources could be achieved simultaneously. Cloud computing is a novel paradigm where organizations or individuals use the resources offered by cloud on rental basis and enjoy the advantage of cost saving on initial investment to setup private storage infrastructures, to purchase higher versioned processors or networking elements. The clients, world over are storing their important data in the remote servers in the cloud without having a backup copy in their local computers, as Storing data into the cloud relives the burden for storage management. But while taking pleasure of numerous benefits, the fact that the most important asset an organization or individual is that the confidential data is not under owners' direct physical control, depriving them of the benefit of not ensuring that it is corrupted. The main goals are to protect confidentiality and integrity of users' data stored on cloud.

Researches, galore in conjunction with third party auditor had been done for formulating remote integrity checking protocols which allow data integrity checking in conjunction with third party auditor. Normally third party auditor is a reliable independent component which is trusted by both the cloud users and server and has no incentive to conspire with either the cloud server or user during the auditing process. Trusted third party auditor is claimed to have the skill and competence that normal cloud users may not have. In order to save time and reduce communication overhead, many researchers recommend the support of third party auditor (TPA).

By leaving the resource consuming cryptographic operations on TPA for achieving confidentiality and integrity, cloud users can be worry-free**.** The issues such as TPA becoming bottleneck, data leakage, introduction of new vulnerabilities, scalability, accountability, performance overhead, dynamic data support, extra hardware cost incurred etc. have motivated many researchers to address the data storage security problems without using a third party auditor, where the cloud user may be comprised of extra application or tool which helps it periodically checks the data integrity.

The proposed scheme achieves data storage correctness in cloud computing without making use of a third party auditor. The central design goal of the proposed scheme is to enhance user's control in managing the various aspects related to the secrecy of sensitive data. This can be achieved by implementing data privacy categorization and protocols to provide security of data in different categories. The proposed model is user-centric, that is cloud customer can flexibly control and manage the different privacy mechanisms necessary to protect sensitive data and achieve legal compliance. Customers will be aware of all the operations carried out to secure the storage and processing of their sensitive information through a secure privacy auditing process.

## II. LITERATURE REVIEW

Recently, Ateniese et al. [15] propose two provable data possession (S-PDP, E-PDP) schemes to provide integrity protection for remote data. The S-PDP and E-PDP support data block append operation, and a variant of their main PDP scheme has public verifiability. Seb_e et al. [14] propose a remote data possession checking protocol for critical information infrastructures. Their protocol supports unlimited

***Manuscript received March, 2013.***

***Nayana.S.R*** *,B.Tech Computer science and Engineering, doing M.E Computer science and Engineering in Noorul Islam University, Tamilnadu ,India,phone: +918089701899.*

***G.Venifa Mini*** *,Asst Professor, Department of computer science and Engineering in Noorul Islam University, Tamilnadu ,India.*

***J.Jeya A Celin****, Professor, Department of computer science and Engineering in Noorul Islam University, Tamilnadu ,India*

times of file integrity verifications and has a tradeoff between the running time and the storage cost at the verifier. Their protocol can be easily adapted to support data dynamics, but it doesn't support public verifiability. After that several studies [12], [8] focus on providing data dynamics to provable data possession protocols. Ateniese et al. [12] propose a very efficient PDP protocol which is based on message authentication codes. Their protocol supports block modification, deletion and append. Erway et al. [8] propose two efficient PDP constructions with data dynamics by using rank-based skip lists and RSA trees. Later Wang etal. [3], [7] recognize the need of privacy against third-party verifiers and develop a random masking technique to deal with this problem. The proposed schemes in [3], [7] use a third party auditor to perform the verification. A third party auditor has certain special expertise and technical capabilities, which the clients do not have. Wang et al. [9] propose a protocol for ensuring remote storage security in the environment of multiple servers. Hao and Yu [2] propose a remote data possession checking protocol for the multiple replicas setting, which supports integrity verification performed by anyone other than client. The scheme proposed in [1], which independently envisages a RSA based HVT methods for remote data integrity checking with the help of third party verifier. The proposed protocol can be viewed as an adaptation of [14] to support more functionality. It inherits the support of data dynamics from [14], and supports integrity verification performed by anyone other than client and for providing security; it doesn't need to use a third-party verifier to check the integrity of data. It also uses ElGamal digital signature scheme for providing integrity verification [11], [16]. PAKE protocol [4], [17] is used to provide security to the passwords provided for authentication and also supports data dynamics by using Merkle B tree algorithm[13],[16].

## III. SYSTEM MODEL

The system model in this work is a typical cloud computing model with three participants as illustrated in fig 1. The three different entities are:

- A cloud service provider which manages and operates a cloud infrastructure of storage and computing services.
- A data owner that employs the cloud storage and computing resource facilities to remotely store and process data.
- A user who access the data from cloud storage.

Authentications of the users are made secure by an Authenticated Password Key Exchange protocol. The proposed secure mass data storage and integrity verification scheme envisages a user centric approach in which the cloud data owner has full control on the privacy mechanisms to be applied on the cloud data. The data owner classifies the data based on significance and sensitivity into three privacy categories.

- **Insensitive**: Data marked with this attribute is not sensitive and hence the provider is fully trusted to store it without any form of encryption.
- **Sensitive:** Data marked with this attribute is stored encrypted by the data owner- specific key.

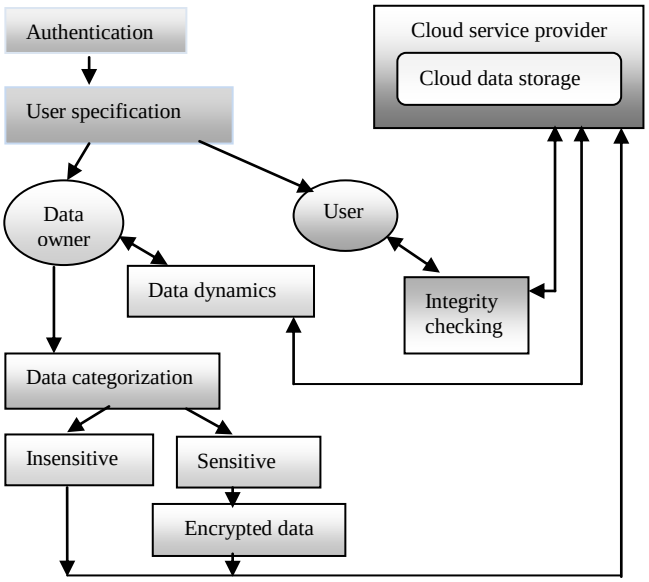


Fig 1: System architecture

The data owner will provide a unique identification number *fid* for each data that is to be stored in the cloud. With this unique file identification number, we can identify in which category the data will be.

The security of the users' passwords is enhanced by the protocol mentioned in [4], [17]. An **Authenticated password key exchange** is where based on the knowledge of their password, two or more parties establish a cryptographic key using message exchanging, such that an unauthorized party cannot involve in the process of brute force guessing the password. This method envisages that strong security can be obtained using weak passwords. Let $p, q$ be primes such that $q|p-1$, and let G be a subgroup of $Z_p^*$ of order q in which the DDH (Decisional Diffie-Hellman) assumption holds. During the initialization phase, generators $g_1, g_2, h, c, d \in G$ and a function H from a family of universal one-way hash functions [18] (which can be based on any one-way function) are chosen at random and published. Note that this public information is not an added assumption;"standard" Diffie-Hellman key exchange typically assumes that parties use a fixed generator g, and seem to require a public generator g for their proofs of security. However, we do require that no one know the discrete logarithms of any of the generators with respect to any other, and thus we need either a trusted party who generates the public information or else a source of randomness which can be used to publicly derive the information. As part of the initialization phase, passwords are chosen randomly for each client and assumed that all passwords lie in $Z_q$. For typical values of $|q|$, this will be a valid assumption for human-memorable passwords.

**Initialization:**

Step 1: Select $p, q$ prime with $|p| = k$ and $q|p-1$; this defines group $G$, subgroup of $Z_p^*$ of order $q$ in which DDH holds.

Step 2: Choose random generators $g_1, g_2, h, c, d \leftarrow G$

Step 3: $H \leftarrow UOWHF$

Publish parameters $(q, p, g_1, g_2, h, c, d, H)$

Step 4: Passwords for each client are chosen independently at random from set {1 … N}, where N is a constant, independent of the security parameter.

$(pw_c)c \in Client \leftarrow \{1, \dots\dots, N\}$

The execution of the PAKE protocol is given as follows:

When client C wants to connect to server S, the client first runs the key generation algorithm for the one-time signature scheme, giving VK and SK. Then, the client computes a client-encryption of $g_1^{pw_c}$. This, along with the client's name, is sent to the server as the first message. The server chooses random elements $x_2, y_2, z_2, w_2$ from $Z_q$, computes $\alpha'$ using the first message, and forms $g_1^{x_2} g_2^{y_2} h^{z_2} (cd^{\alpha'})^{w_2}$. The server then computes a server-encryption of $g_1^{pw_c}$. This is sent back to the client as the second message. The client selects random elements $x_1, y_1, z_1, w_1$ from $Z_q$, computes $\beta'$ using the second message, and forms $K = g_1^{x_1} g_2^{y_1} h^{z_1} (cd^{\beta'})^{w_1}$. Finally, $\beta'$ and K are signed using the signing key which was generated in the first step. The *sid* is defined as the transcript of the entire conversation.

**Procedure:**

**Client side:-**

$(V_k, S_k) \leftarrow key\ generation\ algorithm$

$Select\ random\ element\ r_1\ where\ r_1 \leftarrow Z_q$

$Computes\ \alpha$

$\alpha = H(V_k | Client | {g_1}^{r_1} | {g_2}^{r_1} | {h^{r_1} g_1}^{pw_c})$

$Send\ (Client | V_k | {g_1}^{r_1} | {g_2}^{r_1} | {h^{r_1} g_1}^{pw_c} | (cd^{\alpha})^{r_1})$

$Recieve\ from\ server:$

$Server | {g_1}^{x_2} {g_2}^{y_2} h^{z_2} (cd^{\alpha'})^{w_2} | {g_1}^{r_2} | {g_2}^{r_2} | {h^{r_2} g_1}^{pw_c} | (cd^{\beta})^{r_2}$

$Select\ random\ elements\ x_1, y_1, z_1, w_1\ from Z_q$

$Computes\ \beta'$ is given as:

$H(Server | {g_1}^{x_2} {g_2}^{y_2} h^{z_2} (cd^{\alpha'})^{w_2} | {g_1}^{r_2} | {g_2}^{r_2} | {h^{r_2} g_1}^{pw_c})$

$K = {g_1}^{x_1} {g_2}^{y_1} h^{z_1} (cd^{\beta'})^{w_1}$

$Sig = Sign_{S_k}(\beta' | K)$

$Send\ (K | Sig)$ to server

$sk_c\ is\ calculated\ as$

$({g_1}^{x_2} {g_2}^{y_2} h^{z_2} (cd^{\alpha'})^{w_2})^{r_1} {g_1}^{r_2 x_1} {g_2}^{r_2 y_1} h^{r_2 z_1} (cd^{\beta})^{r_2 w_1}$

**Server side:**

$Recieve\ (Client | V_k | {g_1}^{r_1} | {g_2}^{r_1} | {h^{r_1} g_1}^{pw_c} | (cd^{\alpha})^{r_1}$

$x_2, y_2, z_2, w_2, r_2\ from Z_q$

$Computes\ \alpha'$

$\alpha' = H(V_k | Client | {g_1}^{r_1} | {g_2}^{r_1} | {h^{r_1} g_1}^{pw_c})$

$Computes\ \beta$

$\beta = H(Server | {g_1}^{x_2} {g_2}^{y_2} h^{z_2} (cd^{\alpha'})^{w_2} | {g_1}^{r_2} | {g_2}^{r_2} | {h^{r_2} g_1}^{pw_c})$

$Send\ (Server | {g_1}^{x_2} {g_2}^{y_2} h^{z_2} (cd^{\alpha'})^{w_2} |$

${g_1}^{r_2} | {g_2}^{r_2} | {h^{r_2} g_1}^{pw_c} | (cd^{\beta})^{r_2}$

$Recieve\ (K | Sig)\ from\ client$

$if\ Verify_{V_k}((\beta | K), Sig) = 1$

$sk_s\ is\ calculated\ as$

$({g_1}^{x_1} {g_2}^{y_1} h^{z_1} (cd^{\beta})^{w_1})^{r_2} {g_1}^{r_1 x_2} {g_2}^{r_1 y_2} h^{r_1 z_2} (cd^{\alpha})^{r_1 w_2}$

$else\ sk_s \leftarrow G$

**ELGAMAL INTEGRITY VERIFICATION**

After storing the data in the cloud the data owner and the user can check the integrity of the data by using ElGamal public key encryption algorithm along with ElGamal digital signature scheme. The data before storing in cloud, the client preprocess it and send to the cloud. The sending packet will contain the following:

*Dwid* :-This is the unique data owner ID provided by the cloud service provider upon registration.

*Uid* :- This is the unique user ID provided by the cloud service provider upon registration.

*Fid* :- This is the unique file ID provided by the data owner which contains field specifying whether the data is *Insensitive(IS) or Sensitive(SN).*

The ElGamal public key encryption algorithm gives the data owners public key and private key.

**ElGamal public key algorithm:**

Data user chooses

i. A large prime number p.
ii. A primitive element g modulo p.
iii. A (possibly random) integer x with $2 \le x \le p-2$.
iv. $y \equiv g^x \pmod p$.

Data user's public key is (p,g,y)≡ $PU_{Uid}$ and private key is (x)≡ $PR_{Uid}$.

User computes $y = g^x (mod\ p)$.

Data owner encrypts the message M where M<p.

A random integer 'k' is chosen by the data owner and computes $a \equiv g^k (mod\ p)$ and $t \equiv y^k M (mod\ p)$.

Message M sent as $(g^r, M * y^r) \equiv (g^r, M * g^{xr})$.

Decryption $(M * g^{xr})((g^r)^x)^{-1} \bmod p \equiv M$.

**ElGamal digital signature scheme**

$a \equiv g^k \pmod p$; gcd (k, p-1) = 1; else a ≡ 1?

Message *M*.

Public key (g, p, $y \equiv g^x \pmod p$)

$M \equiv (xa + kb) \bmod (p-1)$

*where x is the private key and k random secret value.*

Digital Signature (a,b) sent with M.

The sending packet will be

$PU_{Dwid}$,Dwid||Fid||E($PU_{Dwid}$,$k_{fid}$)||Timestamp

along with ES(a,b).

**ElGamal signature verification**

$y^a a^b \equiv g^M \pmod p$

$g^M \equiv g^{(xa+kb)} \bmod p$

$(g^x)^a (g^k)^b \equiv y^a a^b \pmod p$

If *M* was modified, congruence would be violated

**DATA DYNAMICS**

After storing data in the cloud the data owner can dynamically update it. The data dynamics offer data insertion, deletion and modification. This can be achieved by a novel technique called Merkle B tree. An MB-tree works like a B+ tree and also consists of ordinary B+ tree nodes that are extended with one hash value associated with every pointer entry. The hash values associated with entries on leaf nodes are computed on the database records themselves. The hash values associated with index node entries are computed on the concatenation of hash values of their children. An example MB-tree is depicted in figure 2. A leaf node entry is associated with hash value $h = H(r_i)$, while an index node entry with $h = H(h_1 | \dots\dots | h_{f_m})$, where $h_1, \dots\dots, h_{f_m}$ are the hash values of the node's children, assuming fan-out $f_m$ per node. After computing all hash values, the data owner has to sign the hash of the root using the private key.

The Merkle B-tree (e.g., insertion and deletion) are similar to those on B+ tree. The primary advantage of B+ tree is that it has a large fan-out, which can reduce the number of I/O operations when searching for an element [15].

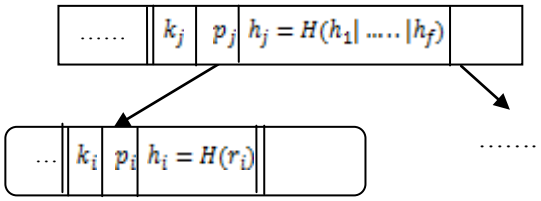


Fig 2: An MB-tree node

Let sig = (keygen, sign, verify) be a secure signature scheme. Let O be an ordered set. The Merkle B-tree scheme[13] consists of algorithms as follows:

- $(s_k, v_k) \leftarrow keygen(1^l)$: This algorithm runs $sig.keygen(1^l)$ to obtain a pair of secret and public keys$(s_k, v_k)$.
- (State, Au) ← setup( $s_k, O$ ) : This algorithm outputs a succinct signature which can be used for verification. The structure of Merkle B-tree T is similar to $B^+$tree, where the leaves store elements in ordered set O, and the values of internal nodes are computed from concatenation of the values of their children through an appropriate hash function. The root of the tree will be signed to produce the state information, denoted by state=sig.sign(T) and Au=T.
- Update: The update protocol fulfills update operations. Reference [15] gives the details about the insertion and deletion operations. Suppose Upd="update the element $E_i$ to $E_i'$".Upon receiving Upd from the data owner, the server updates $E$ to $E'$ by replacing $E_i$ with $E_i'$, and updates T to$T'$. The server provides a proof, a path of $E_i$ in T, namely a sequence including values of nodes from $E_i$ to the root of MB-tree as well as the values of these nodes' siblings. The data owner can hash the path of $E_i$ from the bottom to the top and verify whether the root is valid with respect to state or not. If so, the data owner updates the path from the bottom to the top by replacing $E_i$ with $E_i'$,which result in a new root, signs the new root, and sets $State' = sig.sign(T')$; otherwise, the data owner aborts.
- Verify: Given a range query qry(a,b), the server outputs a proof Prf showing that Rst contains all elements in [a,b].
  - If Rst is empty, which means there exists some s, such that $E_s < a, b < E_{s+1}$.The server returns the proof Prf including two paths: a path of $E_s$ and a path of $E_{s+1}$. The querier hashes each path from bottom to the top, and verify whether the roots match the state $state$, and $E_s$ is neighbor to $E_{s+1}$. If so, the one who queries returns the null set Rst , Prf, and accept. Otherwise, abort.
  - If Rst is not null, suppose the query result is $(E_s, \ldots \ldots, E_t)$,s≤ $t$. The server returns the proof Prf including two paths: one path of the left-most neighbor leaf of $E_s$ , and the other path of the right-most leaf of $E_t$. Then the querier uses Prf and the result Rst to construct a $B^+tree$, and verifies whether the root of the this $B^+$tree is valid for $state = sig.sign(T)$. If so, the querier returns (Rst,Prf,accept); otherwise abort.

## IV. PERFORMANCE ANALYSIS

Implementation in .Net gives results as the ElGamal encryption takes longer time compared to RSA which leads to a delayed preprocess stage. Preprocess time for both the algorithms have been shown in figure 3 & 4.

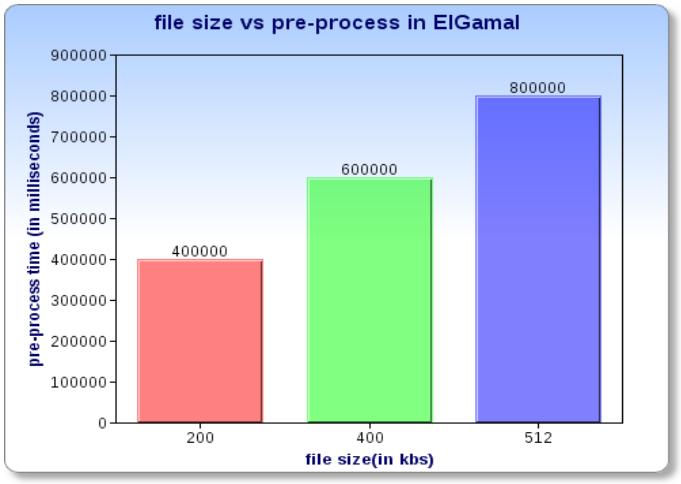


Fig 3: File size vs pre-process time in ElGamal

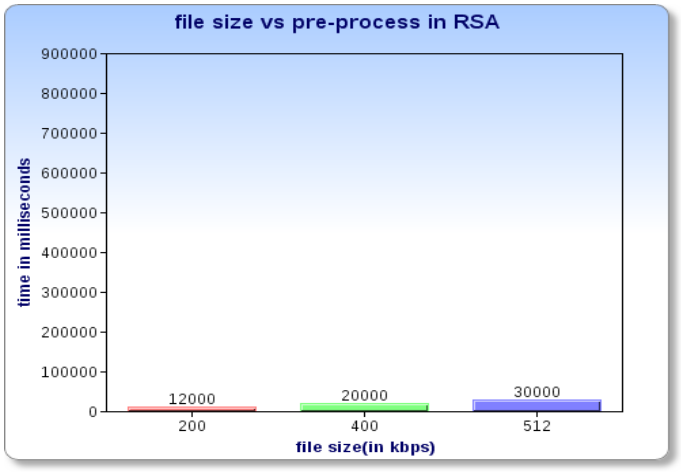


Fig 4: File size vs pre-process time in RSA

ElGamal shows exceptionally high readings compared to RSA. Preprocess includes key generation, metadata creation

and file transfer. The key generation time for both the algorithms is almost same. ElGamal is slower at encryption end but faster while decrypting. So the cryptosystem generates proof quickly compared to RSA as seen in Figure 3, 4 and Figure 5, 6.

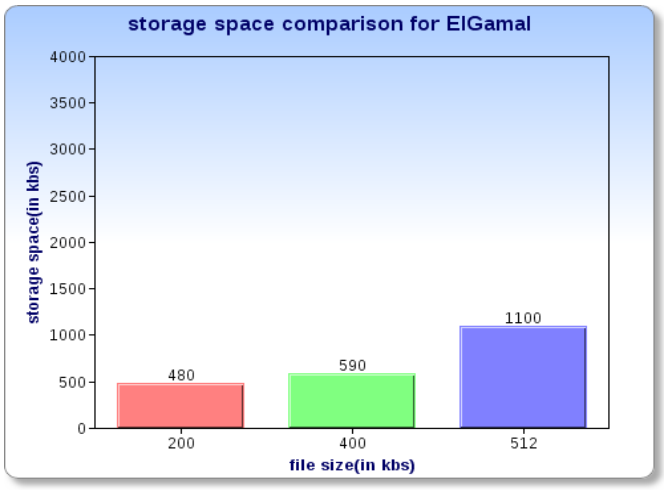


Fig 5: File size vs storage space in ElGamal

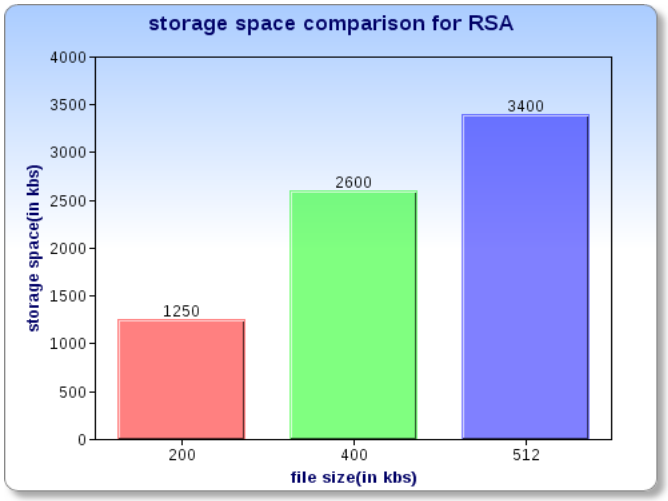


Fig 6: File size vs storage space in RSA

## V. CONCLUSION AND FUTURE WORK

The proposed work gives a new data integrity verification protocol for cloud storage providing integrity protection of customers' important data. It is proved to be secure against unauthorized users since it does not involve any trusted third party in data integrity checking operation. Only the data owner and authenticated users are allowed to check the integrity of data. ElGamal encryption scheme is for checking the integrity of data. User performing integrity verification avoids disputes between the client and server regarding data integrity. It has very good efficiency in the aspects of communication, computation and storage costs. Dynamic data updating is provided by Merkle B-tree algorithm.

The future work is focused on protocol support for data dynamics using Embedded Merkle Hash Tree (EMHT) [13] and a hardware support for data integrity checking.

## REFERENCES

[1] Zhuo Hao, Sheng Zhong, Neng-Hai Yu, "A privacy preserving data integrity checking protocol with data dynamics and public verifiability " Knowledge and Data Engineering, IEEE transactions on, vol 23, 2011,pp. 1432 - 1437

[2] Z. Hao and N. Yu, "A multiple-replica remote data possession checking protocol with public verifiability," in Data, Privacy, and E-Commerce, 2010.

[3]C. Wang, Q. Wang, K. Ren, and W. Lou "Privacy-preserving public auditing for data storage security in cloud computing " in *InfoCom2010, IEEE*.

[4] Feng Hao, Peter Ryan ," J-PAKE: authenticated key exchange without PKI"in Transactions on computational science XI, 2010 Pages 192-206.

[5]Li Xiao-fei, Shen Xuan-jing, Chen Hai-peng. "An Improved ElGamal Digital Signature Algorithm Based on Adding a Random Number". In 978-0-7695-4011-5/10 $26.00 © 2010 IEEE.

[6] Wassim Itani, Ayman Kayssi, Ali Chehab " Privacy as a Service: Privacy-Aware Data Storage and Processing in Cloud Computing Architectures" in Proceedings of the 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing, Pages 711-716 .

[7] C. Wang, S. S.-M. Chow, Q. Wang, K. Ren, and W. Lou,"Privacy preserving public auditing for secure cloud storage." Cryptology ePrint Archive, Report 2009/579.

[8] C. Erway, A. K¨upc¸ ¨u, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," in *CCS '09*: Proceedings of the 16th ACM conference on computer and communications security , 2009, pp. 213–222.

[9] Wang C, Wang Q, Ren K, and Lou W, "Ensuring data storage security in cloud computing," in Proc. of IWQoS'09, 2009.

[10]Kyrakos Mouratidis, Dimitris Sacharidis"Partially materialized digest scheme: an efficient verification method for outsourced databases" in the international journal on very large data bases archieve,Vol 18 Issue 1, 2009,pages 363-381.

[11]. Marco Bodrato, "Public key cryptography. ElGamal, hints on implementation", Tokyo University of Science,2008.

[12] G. Ateniese, R. Di Pietro, L. V. Mancini, and G. sudik, "Scalable and efficient provable data possession," in SecureComm '08:proceedings of 4th international conference on Security and privacy in communication networks,
2008, pp. 1–10.

[13] Qingji Zheng, Shouhuai Xu, Giuseppe Ateniese "Efficient Query integrity of outsourced dynamic databases"in Proceeding of the 2012 ACM Workshop on Cloud computing security workshop, 2012, Pages 71-82

[14] F.Sebe,J.Domigo-Ferrer,A.Martinez-Balleste et.al "Efficient remote data possession checking in critical information infrastructures", IEEE Transactions on, vol. 20, 2008,pp. 1034-1038.

[15]G. Atneiese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song,"Provable data possession at untrusted stores" in CCS '07: Proceedings of the 14th ACM conference on Computer and communications security, pp. 598-609.

[16]Li, F., Hadjieleftheriou, M., Kollios, G., Reyzin, L "Dynamic authenticated index structures for outsourced databases." In: SIGMOD Conference, 2006, pp. 121–132

[17]. Andreas .V.Meier, "The ElGamal Cryptosystem" 2005.

[18] Katz, J., Ostrovsky, R., Yung, M. :"Efficient password-authenticated key exchange using human-memorable passwords". In: Pfitzmann, B. (ed.) EUROCRYPT 2001.LNCS, vol. 2045, 2001, pp. 475–494. Springer, Heidelberg.

[19]M. Naor and M. Yung. "Universal One-Way Hash Functions and Their Cryptographic Applications", 1989, STOC '89.

[20]R.C. Merkle, "Protocols for public key cryptosystems" Proc. Of IEEE Symposium on Security and Privacy'80, 1980, pp. 122–133.

**Nayana.S.R** was born in Trivandrum in 1987. She graduated in B-Tech Computer Science and engineering from Lourdes Matha College of Science & Technology, Kuttichal under University of Kerala in 2009. Currently she is pursuing M.E Computer Science and Engineering in Noorul Islam University, Thuckalay, Tamilnadu. Her research interest includes Wireless Communication, cloud computing.

**G.Venifa Mini** is Assistant Professor, Department of computer science and engineering at Noorul Islam University, Thuckalay, Tamilnadu. She graduated in B.E Computer Science and Engineering and post graduation in M.E Computer science and Engineering. Currently she is pursuing Ph D in Noorul Islam University. Her research interest includes Wireless Networking, Cloud Computing.

**J.Jeya A Celin** is Professor, Department of Computer science and Engineering at Noorul Islam University, Thuckalay, Tamilnadu. She graduated in B.Sc Computer Science and post graduated in MCA and a Ph D holder. Her research interest includes Computer Applications, Data Mining, Wireless Networking, and Cloud Computing.

.