Towards Efficient ECC Based Provable Data Possession Protocol with Data Dynamics for Secure Cloud Storage

D. Sudha Devi, K. Thilagavathy

Abstract- Cloud Computing is acquainted for its cost-effective on-demand services based on internet. The significant benefits of cloud services drive organizations and individuals to make hay by outsourcing data to cloud storages. Outsourcing data brings users' an easy and economical way of data management and also relieves users' from the burden of building and maintaining local data storage. However, data being residing in some third-party's premises, users have no full control over their data which necessitates ensuring confidentiality and integrity of data stored in untrusted cloud storage servers. To verify the correctness of data in cloud storage, this paper proposes an efficient ECC based Provable Data Possession (EPDP) Protocol with data dynamics. The proposed protocol preserves confidentiality of data stored in cloud storage and allows data owner to verify the integrity of data without retrieving the whole original data. Also the protocol is designed to perform stateless auditing and supports data dynamics at block level retaining the same security assurance. Security and Performance analysis proves the proposed protocol to be secure and highly efficient for secure cloud storage.

Keywords- confidentiality, cloud data storage, data integrity, elliptic curve cryptography, integrity verification, secure storage

I. INTRODUCTION

Cloud computing has been envisioned as a computing model that enables ubiquitous, convenient network access, location independent resource pooling, rapid elasticity and on-demand self-service that can be provisioned and released with minimal management effort or with service provider interaction [1].The service rendered by cloud that cannot be compared with other models comprises scalability, flexibility, multi-tenancy, Device & Location Independence, agility, pay for what you use etc. The major cloud service providers like Amazon[2], Google[3] offer these characteristics as services over the internet. Other such providers include Rack space, Microsoft, Sales force, VMware, Verizon, Citrix, IBM [4]-[10] etc. Migrating data into cloud storage provides data owners a great convenience which relieves owners from technical complexities and from the burden of investing, building and maintaining own infrastructure. Even though the benefits of cloud are significant and tremendous, one unique aspect that impedes the adoption of cloud by many individuals and enterprises is concern over data security [11]. Ensuring confidentiality and integrity of the outsourced data is very important since data are stored on shared servers at remote site. As data are handled by some third parties, the data owner has lack of full control over the outsourced data.

Revised Version Manuscript Received on January 30, 2016.

D. Sudha Devi, Coimbatore Institute of Technology, Coimbatore, Tamil Nadu, India.
Dr. K. Thilagavathy, Coimbatore Institute of Technology, Coimbatore, Tamil Nadu, India.

Confidentiality can be achieved by encrypting the data using industry standard algorithms before moving into cloud storage. And to prevent unauthorized access, authentication and access control protocols provides best solutions. To obtain fine-grained access to outsourced data a secure protocol is proposed in [12]. To utilize efficiently the pool of cloud services without fear, data owner must ensure that the outsourced data is not tampered and are handled as they expect. To ensure the correctness of stored data, data auditing service is essential. Many researchers have proposed schemes for confidentiality and integrity verification [13]-[24] for secure remote data storages. However, data integrity verification cannot be availed as a service from cloud providers since the cloud data storages are considered to be untrusted servers. Hence it is one of the essential responsibilities of data owners to audit how their sensitive data are handled in cloud environment. Ensuring integrity of data is a difficult task without a copy of data stored in data owner's server. Without having a local copy and retrieving whole data from cloud storage for integrity verification becomes cumbersome for data owner. Therefore, a secure and efficient remote data integrity verification protocol satisfying the above requirements is required to obtain the assurance that the outsourced data are handled as expected by data owners.

Ateniese et al. [25] was the first to introduce secure and efficient Provable Data Possession (PDP) schemes based on homomorphic verifiable tags. Wang et al. [26] designed a Privacy-Preserving Public Auditing for cloud data storage in which public key based homomorphic authenticator was utilized and random masking was used to achieve privacy-preserving public auditing scheme. Yang et al. [27] has proposed PDP for resource constrained mobile devices in cloud using bilinear signature and Merkle Hash Tree. Zhu et al. [28] constructed an interactive PDP protocol based on Diffie-Hellman computation. Natu and Pachouly[29] has compared various PDP schemes and has come out with advantages and disadvantages of various schemes. Liu et al. [30] has proposed a public auditing scheme based on BLS short signature method and homomorphic hash function. Worku et al [31] proposed a Privacy-preserving public auditing scheme in which the drawback of Wang et al.'s scheme has been overcome. Vanitha et al. [32] has designed a Privacy-preserving public auditing scheme based on elliptic curve digital signature algorithm for the shared data in cloud.

In this paper an Efficient Provable Data Possession Protocol using elliptic curve cryptography is proposed. Without retrieving the original data, data owners can verify integrity of outsourced data in an efficient way. Data owner generates a secret and public key pair which is used in computing tags for the data and outsources data file and tags into cloud storage. Later the data owner or an entrusted third party auditor can send a challenge to cloud server to verify the integrity of data. The server in turn has to prove that the possession of data is safe by generating a proof for the challenge with the stored data and tags. Finally verifier can check the proof to verify integrity of data. Since the outsourced data are subject to updations, the proposed protocol is designed to support data dynamics also. The proposed EPDP protocol supports data operations such as insertion, deletion and modification at block level. Also the proposed system is assumed to be stateless, means that the verifier need not maintain states between all audits which is considered to be a desirable property in an auditing system. On the whole the proposed EPDP protocol gives assurance to data owners that their outsourced data are handled as they expect.

The rest of the paper is organized as follows: Section II describes the System Model, Section III elaborates the proposed Provable Data Possession protocol, Section IV deals with Data dynamics, Section V discusses Security and Performance analysis, and Section VI concludes with conclusion.

II. SYSTEM MODEL

A. Cloud Data Storage overview:

The cloud environment and the multilevel security system required to protect the data stored in cloud servers and the responsibilities of a data owner in securing data are discussed exhaustively in [33]. The cloud data storage model considered in this paper consists of the following entities as depicted in Fig.1.

1. Data Owner (DO): an entity that outsources data to cloud data storage.
2. Data User (DU): an authorized person who sends request for accessing the data stored in cloud storage.
3. Cloud Service Provider (CSP): an entity that provides storage as a service to users.
4. Third Party Auditor (TPA): an optional entity, expertise in verifying the integrity of the outsourced data on behalf of data owner.

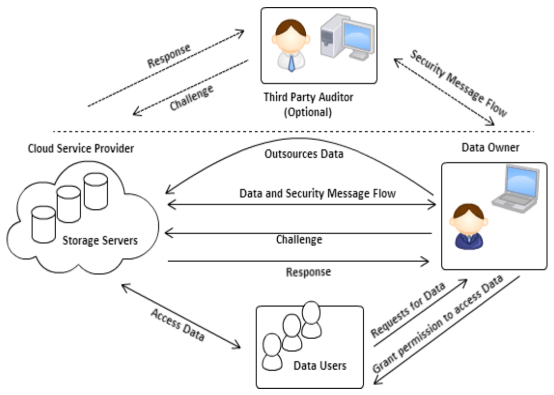

Fig. 1. Cloud Data Storage overview

B. Security Attacks

The data owners are recommended to encrypt data before outsourcing in order to protect the sensitive data hosted to cloud servers. Even though the data are encrypted, an attacker who is interested in knowing the data if not possible to decode it, may demolish the stored data by doing some manipulations or may delete some data which results in confusions to data owner. At such time there is an imperative need for an efficient data integrity verification mechanism which helps in verifying whether the stored data is intact. This data integrity verification can be done by data owner by himself or can assign a trusted TPA to perform the verification. The two types of threats in cloud storage environment are as follows:

Internal Attack:

The privileged users in the CSP site know how to handle the stored data and hence may either intentionally or accidently manipulate or leak data which results in heavy loss to data owner.

External Attack:

The malicious users from outside the cloud environment may compromise cloud servers and hacks data which may affect both data owner and CSP.

C. Design Goals

The proposed system is aimed in designing an integrity verification protocol that is efficient and secure enough to verify the data stored in cloud storage. The proposed EPDP protocol should encompass the following properties:

- Security: The proposed protocol should ensure confidentiality and integrity of the outsourced data. Confidentiality means the data is protected using encryption techniques and only authorized users should access the protected data and Integrity means verifying whether data has undergone unauthorized manipulations.
- Stateless verification: The proofs generated by the protocol should be based on randomly chosen challenges which relieve verifier from maintaining states between audits.
- Efficiency: The proposed protocol should be efficient in terms of low computation and storage overhead and the challenge posed should be unlimited which proves the efficiency of the proposed protocol.
- Audits without downloading: Data integrity verification should be done without retrieving the copy of whole original data.

III. PROPOSED EFFICIENT ECC BASED PROVABLE DATA POSSESSION PROTOCOL

To ensure the correctness of data outsourced to cloud storage, an efficient Provable Data Possession protocol using Elliptic Curve Cryptography is proposed. An elliptic curve E over Z_p is defined by the equation $y^2=x^3+ax+b$ where $a,b \in Z_p$ and $4a^2+27b^2 \neq 0 \pmod{p}$, with a special point O, called the point at infinity.

The set $E_p(a,b)$ consists of all pairs of integers (x,y) that satisfy $y^2 \bmod p=(x^3+ax+b) \bmod p$ together with a point at infinity. The co-efficient and variables are all elements of Z_p.A finite point G on elliptic curve having largest order n is chosen as base point [34]-[36]. The proposed EPDP protocol consists of the following five phases:

- Key Generation – given a security parameter as input, this algorithm generates a key pair (secret key, public key) as output.
- Tag Generation–given data block, using a hash function and keys as inputs, this algorithm gives tags as output.
- Challenge Generation – selecting random values, this algorithm generates challenge (chal) as output.
- Proof Generation – given a challenge ‘chal’,stored data file and tags as inputs, this algorithm generates a proof for data possession as output.
- Proof Verification– given generated proofs,chal and secret key as inputs, this algorithm verifies the proof of data possession and produces accept or reject as output.

Pre-Processing Phase

To ensure confidentiality of the outsourced data, before moving data into cloud storage it must be encrypted using industry standard encryption algorithms. Based on the sensitivity how data could be protected through multilevel security system and how data users in a hierarchy could access these protected data safely are discussed in [33, 12]. The encrypted data file is considered for processing in the proposed system. The phases of the proposed EPDP protocol are as follows:

1. Key Generation phase

The key generation algorithm, ‘GenKey’ is executed by data owner to generate secret and public key pair. For a given security parameter k, the GenKey algorithm provides a secret and public key pair (sk, pk) as output. Data owner selects a random integer k from [1,n-1],and P=kG is computed, where k is the secret key and P is the public key.

Algorithm 1: GenKey

```
1. Procedure: GenKey($1^k$)→(sk,pk)
2. select a random integer k $\epsilon$ [1,n-1]
3. compute P=kG
4. sk← k, pk←P
5. End Procedure
```

2. Tag Generation phase

The data owner computes tags over the encrypted data files by executing Tag generation algorithm ‘GenTag’. For generating tags, the data file F is divided into blocks, say $F = \{m_1,m_2,...,m_n\}$. For each block m_i, in data file, data owner selects a random integer d_i from [1,n-1] as the secret value (sv) of the block for which tag T_m is generated. The secret key(sk) and the coordinates of public key(pk) are used in computing the tags. Along with these keys a hash function $h:\{0,1\}^* \rightarrow Z_p$and the block secret value (sv) are used in tag computation. The data file and the generated tags for the data blocks $\{F, T_m\}$are sent to CSP.

Algorithm 2: GenTag

```
1. Procedure: GenTag(m,sv,sk,pk)→$T_m$
2. compute $\sigma_{i,1}= h(m_i)k\ d_iP_x$
3. compute $\sigma_{i,2}= d_iP_y$
4. $T_m \leftarrow \{\sigma_{i,1},\sigma_{i,2}\}$
5. End Procedure
```

3. Challenge Generation phase

Data owner can verify the integrity of the data stored in cloud storage by challenging CSP. The verifier creates a challenge by executing challenge generation algorithm ‘GenChal’ and sends it to server for proof generation. To generate a challenge, verifier selects $I=\{a_1, \ldots,a_c\}$, a random c-element subset of the set [1,n] and for each $i\in I$ $(1\leq i\leq c)$, a random value r_i is chosen. The challenge ‘chal’ indicates the positions and the blocks that are to be verified and is represented as $chal=\{(i,r_i)\}_{i\in I}$. The server on receiving ‘chal’ from the verifier, computes the response for the challenge and returns it to the verifier.

Algorithm 3: GenChal

```
1. Procedure: GenChal (k)→chal
2. Choose a random c-element subset
   I from the set [1,n]
3. For each $i\in I$, $1\leq i\leq c$,
    select a random value $r_i$
4. $chal \leftarrow \{(i,r_i)\}_{i\in I}$
5. End Procedure
```

4. Proof Generation phase

The CSP on receiving a challenge from the verifier, computes a corresponding response as data integrity proof. The algorithm ‘GenProof ‘ is executed by CSP which takes the data file, tags, challenge ‘chal’ and public key as inputs and generates $\rho =\{\tau,\mu\}$ as proofs.

Algorithm 4: GenProof

```
1. Procedure: GenProof(F,$\sigma_{i,1}$,$\sigma_{i,2}$, chal, pk)→ρ
2. compute $\tau = \prod_{i=a_1}^{a_c} \sigma_{i,1}\ P_y\ r_i$
3. compute $\mu = \prod_{i=a_1}^{a_c} \sigma_{i,2}\ h(m_i)P_x\ r_i$
4. $\rho \leftarrow \{\tau,\mu\}$
5. End Procedure
```

5. Proof Verification phase

Data owner on receiving the response for the challenge from the CSP, verifies the integrity of the data by executing proof verification algorithm ‘Check Proof’. Verifier verifies whether the following condition holds for the received response using his secret key.

Check Proof algorithm takes the response ‘ρ’, challenge ‘chal’ and secret key(sk) as inputs and returns ‘true’ if the integrity of queried data blocks are verified as correct or returns ‘false’ otherwise. That is, the verifier can check for the condition $\tau = k\,\mu$. If the condition holds then the output is “true” otherwise is “false”.

Algorithm 5: CheckProof

```
1. Procedure: CheckProof(ρ, chal, sk)→ true / false
2. Check if ($\tau = k\,\mu$) then
       return(true)
   else
       return(false)
   End if
3. End Procedure
```

IV. DATA DYNAMICS

The proposed EPDP protocol supports various operations such as insertion, modification and deletion on blocks in the data file. The notations used for data dynamics are enumerated in Table I.

TABLE I. Notations – Dynamic data operations

Notations	Description
BO	Block Operation
1	Block Insertion
2	Block Modification
3	Block Deletion
p	Specified block to be processed
m_k^*	New/modified block

A. Block Insertion (BO=1)

The data owner suppose wants to insert a block in a specified position say, after block ‘p’ in the data file F, then algorithm 6 is executed to get the work done. In the proposed system, the block insertion operation can be performed without computing again the tags for the blocks which are moved backward due to insertion of a new block.

Algorithm 6: Block Insertion

```
1. Procedure: Block Insertion←(p,$m_k^*$ )
2. Select position p and new block to be inserted, $m_k^*$
3. Compute $\sigma_{k,1},\sigma_{k,2}$
4. $T_k \leftarrow \sigma_{k,1},\sigma_{k,2}$
5. Send request to server to process the update,
   ProcessUpdateRequest(1,p,$m_k^*$,$T_k$)
6. End procedure
```

B. Block Modification (BO=2)

The most frequently used data operation is data modification. The data owner suppose modifies a block say, $m_p$with m_k^* in the data file F, then algorithm 7 is executed to update data file. Once server receives block modification request, then it replaces the old block with the new block.

Algorithm 7: Block Modification

```
1. Procedure: Block Modification←(p,$m_k^*$ )
2. modify block$m_p$to $m_k^*$
3. Compute $\sigma_{k,1},\sigma_{k,2}$
4. $T_k \leftarrow \sigma_{k,1},\sigma_{k,2}$
5. Send request to server to process the update,
   ProcessUpdateRequest(2,p,$m_k^*$,$T_k$)
6. End procedure
```

C. Block Deletion (BO=3)

The data owner suppose wants to delete a block say, m_p in the data file F, then algorithm 8 is executed to perform deletion. Once server deletes the specified block, all other subsequent blocks are moved forward one step which is the reverse process of block insertion.

Algorithm 8: Block Deletion

```
1. Procedure: Block Deletion ←(p )
2. Select block $m_p$ to be deleted
3. Send request to server to process the update,
   ProcessUpdateRequest(3,p )
4. End procedure
```

D. Process Update Request

The data operations get completed with the Process Update Request algorithm. Each data operation request in turn calls process update algorithm to complete the task. This algorithm finally updates blocks in server according to the block operation request send by the data owner.

Algorithm 9: Process Update Request

```
1. Procedure: Process Update Request→$F'$
2. If BO=1 then
3.     Insert $m_k^*$ after $m_p$ and move all blocks
       after $m_k^*$backward
       Store the corresponding tag $T_k$ in server
4. Else if BO=2 then
       Update block $m_p$ with $m_k^*$
       Store the corresponding tag $T_k$ in server
5. Else if BO=3 then
       Delete block $m_p$ and tag $T_p$
6. Update file F to $F'$
7. Return($F'$)
```

V. SECURITY AND PERFORMANCE ANALYSIS

A. Security Analysis

This section discusses the security analysis of the proposed protocol. The proposed EPDP protocol is proved to be correct and is also sound.

1. Correctness:

Theorem: If the outsourced data is stored honestly in cloud storage server, then whenever the server receives a challenge from verifier could compute proof for the challenge that will always be accepted by the verifier.

Proof: We have

$$\tau = \prod_{i=a_1}^{a_c} \sigma_{i,1}\ P_y\ r_i$$

$$\tau = \prod_{i=a_1}^{a_c} \sigma_{i,1}\ P_y\ r_i$$

$$\tau = \prod_{i=a_1}^{a_c} (h(m_i)kd_iP_x)\ P_y\ r_i$$

$$\tau = \prod_{i=a_1}^{a_c} (\ \sigma_{i,2}\ h(m_i)\ kP_x\ )\ r_i$$

$$\tau = \prod_{i=a_1}^{a_c} k\ (\sigma_{i,2}\ h(m_i)P_x\ r_i)$$

$$\tau = k\ \mu$$

From the above proof the proposed protocol is said to be correct or valid.

2. Soundness:

Theorem: For the dishonest server it is infeasible to confound the verifier to accept a false proof.

Proof: The server upon utilizing the already computed value or the guessed value cannot baffle the verifier in accepting a false proof because every time the challenge is chosen randomly.

B. Performance Evaluation of Probabilistic verification

The proposed protocol allows cloud storage server to prove data possession of selected blocks of data file F. This nature of sampling detects server misbehavior with high probability and also reduces the workload on storage server. Following are the assumptions used for detection probability.

- Out of n blocks of data file F, server disrupts d blocks (d/n)
- Verifier randomly selects the indices is uniform and therefore probability of selecting any block in the data file F is 1/n.
- Verifier asks proof for a challenge which on average has 'c' different blocks and detects misbehavior with high probability.

The detection probability P is as follows:

$$P = 1 - (1 - \frac{d}{n})^c$$

$$\text{where} c = \frac{\log(1 - P)}{\log(1 - \frac{d}{n})}$$

P specifies the detection probability that, if server disrupts 'd' blocks out of 'n' blocks of the data file F, then verifier can detect the disruption when a challenge is send for which the server has to compute proof for 'c' blocks. Fig.2 plots P for constant value of d and for different values of n and c. It shows the result for different detection probability from 0.5 to 0.99.

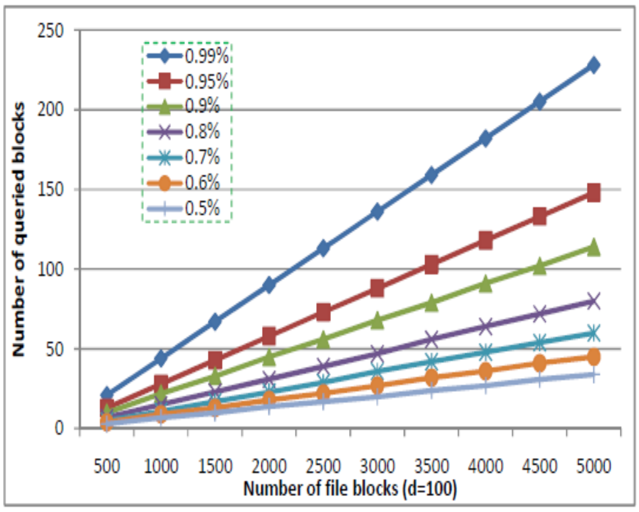

Fig. 2. Server misbehavior under different detection probability– number of data file blocks (n), number of queried blocks (c), disrupted block (d=100 blocks)

Fig.3 depicts the changes in ratio under different detection probabilities where the number of disrupted blocks is 1% of n. The verifier, to achieve P of 99% should ask for minimum 458 blocks for verification. To achieve P of 95%, 90%, and 80%, the verifier should ask for at least 298, 229 and 160 blocks for verification respectively.

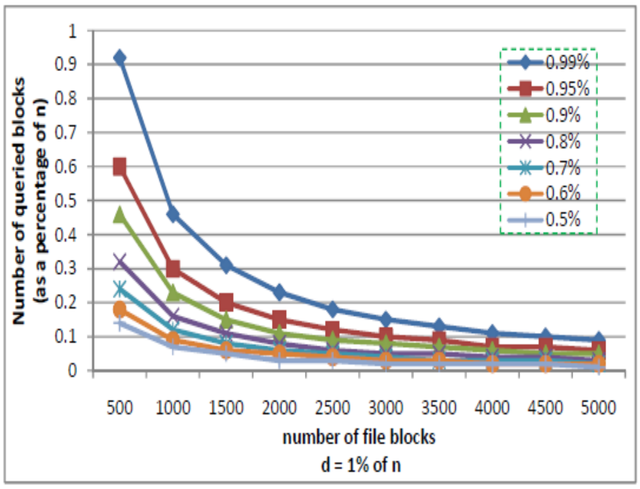

Fig. 3. Ratio of queried blocks (c) in total number of file blocks (n) under different detection probability

(d=1% of n)

Performance Evaluation

The performance of proposed EPDP protocol is discussed in this section. Table II shows the notations used for evaluation of cryptographic operations. The protocol is built using cryptographic hash function and elliptic curve cryptography.

TABLE II. Notations used for performance evaluation

Notations	Descriptions
T_{mul}	Time complexity for executing modular multiplication

T_{hash}	Time complexity for executing hash function
T_{ec_mul}	Time complexity for executing multiplication in an elliptic curve point

On the data owner side, a pair of keys are generated and are used in tag generation. The computation cost for this process is $nT_{hash} + 4nT_{mul} + T_{ec\_mul}$, where n is the number of file blocks. On the server side, proof is generated and sent to verifier for integrity verification. The computation cost for this process is $cT_{hash} + 5cT_{mul}$. Upon receiving proof for the queried blocks, verifier evaluates the proof for integrity verification. The computation cost for this process is cT_{mul}, where c is the number of queried blocks.

A comparison between schemes proposed in [26], [30] and the proposed EPDP protocol is as follows: To check the efficiency of the proposed protocol, the simulation is performed using Intel Core i3 processor with 4GB RAM and Java cryptography library is used. The schemes proposed in [26] and [30] are based on bilinear pairing which involves computation cost for multiplication operation, exponentiation operations and bilinear pairs. Since the proposed EPDP protocol is built without pairing, the computation cost for exponent operations and bilinear pairs are reduced and the computation cost for modular multiplication, elliptic curve point multiplication and hash functions are involved. The comparison is performed, say the number of queried blocks c = 32 and the computation costs of the protocol in [26], [30] and the proposed protocol are shown in the following Fig.4.

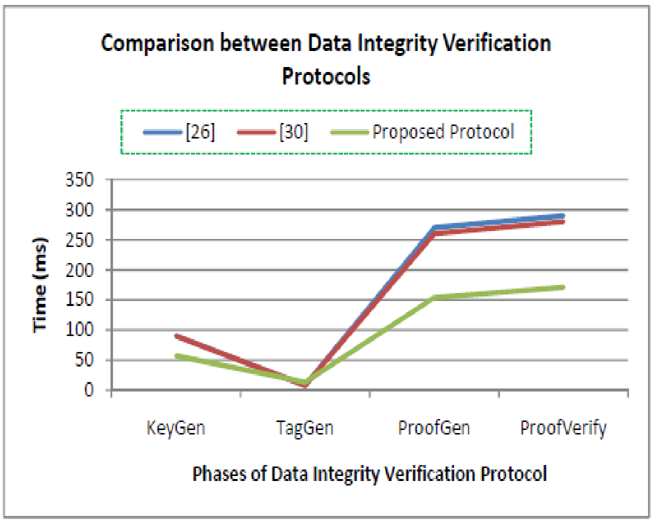

Fig. 4. Comparison between [26], [30] and the proposed protocol

Fig.4 depicts the computation cost for each algorithm in [26], [30] and the proposed EPDP protocol. From the analysis, it is determined that the computation cost is minimized and thereby the comparison result proves the efficiency of the proposed protocol.

VI. CONCLUSION

The Cloud Computing paradigm provides users a number of valuable services that are required in today's global electronic village. The most widely used cloud services include data storage and computing services. Data owners hosting sensitive data in cloud storage servers have to overcome data security problems. To ensure confidentiality and integrity of data outsourced to cloud storage, in this paper, an efficient provable data possession protocol using elliptic curve cryptography is proposed. The data owner can perform stateless auditing to verify the correctness of data stored in cloud. The integrity verification can be done without retrieving the whole original data from cloud server which minimizes communication overhead. Also the proposed protocol supports dynamic data operations at block level maintaining the same security assurance. Through security and performance analysis it is proved that the proposed protocol is secure and efficient in terms of confidentiality and integrity.

REFERENCES

1. P.Mell and T.Grance,"Draft NIST Working Definition of Cloud Computig"[Online]. Available: http://csrc.nist.gov/groups/SNS/cloudcomputing/index.html, 2012.
2. Amazon Web Services (AWS), [Online]. Available: http://aws.amazon.com/ec2.
3. Google AppEngine, [Online]. Available: http://googcloudlabs.appspot.com.
4. Rackspace Cloud sites, [Online]. Available: http://www.rackspace.com/cloud/sites.
5. Microsoft Azure, [Online]. Available: https://azure.microsoft.com/en-in.
6. Salesforce, [Online]. Available: http://www.salesforce.com/in.
7. VMware vCloud, [Online]. Available: http://vcloud.vmware.com.
8. Verizon, [Online]. Available: http://www.verizonwireless.com/support/verizon-cloud.
9. Citrix Cloud Services, [Online]. Available: https://www.citrix.com/solutions/cloud-services.
10. IBM Cloud, [Online]. Available: http://www.ibm.com/cloud-computing.
11. D. Sudhadevi and K. Thilagavathy, "A novel approach to enhance cloud data defense," Asian Journal of Information Technology, Vol. 12, No. 9, 2013, pp. 305–311.
12. D. Sudha Devi, K. Thilagavathy, "An Elliptic Curve Cryptography based adaptive and secure protocol to access data outsourced to cloud server", International Journal of Applied Engineering Research, Vol. 10, No. 18, 2015, pp 39443-39450.
13. H. Shacham and B.Waters, "Compact Proofs of Retrievability", Proc.14th International Conference Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT), LNCS 5350,2008, pp.90-107.
14. C. Erway, A. Kupcu, C. Papamanthou, R. Tamassia, "Dynamic provable data possession", In: Proceedings of the 16th ACM conference on computer and communications security, CCS 2009, pp. 213–22.
15. G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson,and D. Song, "Remote Data Checking using Provable Data Possession", ACM Transactions on Information and System Security,Vol. 14, No. 1, 2011, pp. 12.1–12.34.
16. Q. Wang, C. Wang, K. Ren W. Lou, and J. Li, "Enabling public verifiability and data dynamics for storage security in cloud computing," IEEE Transaction on Parallel and Distributed Computing, Vol.22, No.5, 2011, pp.847-859.
17. Z. Hao, S. Zhong, N. Yu. "A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability", IEEE Transaction on Knowledge Data Engineering, Vol.23, Issue. 9, 2011, pp.1432-1437.
18. M. van Dijk, A. Juels, A. Oprea, R. L. Rivest, E. Stefanov, and N. Triandopoulos, "Hourglass schemes: how to prove that cloud files are encrypted" In Proceedings of the 2012 ACM conference on Computer and communications security, 2012,pp. 265–280.
19. Q. Zheng, S. Xu,"Secure and efficient proof of storage with deduplication", In: Proceedings of the second ACM conference on data and application security and privacy, CODASPY 2012,pp. 1–12.

20. P. Williams and R. Sion, "Single round access privacy on outsourced storage", In Proceedings of the 2012 ACM conference on Computer and communications security, 2012, pp. 293–304.
21. C. Wang, Q. Wang, K. Ren, N. cao and W. Lou , "Towards Secure and Dependable Storage Services in Cloud Computing", IEEE Transaction on Cloud Computing, Vol.5, Issue.2, 2012, 220-232.
22. Y. Zhu, H. Hu, G. Ahn and M. Yu, "Cooperative Provable Data Possession for Integrity Verification in Multicloud Storage", In Proceedings of IEEE Transactions on Parallel Distributed Systems, 2012, pp. 2231-2244.
23. Y. Ren, J. Xu, J. Wang and J. Kim, "Designated-Verifier Provable Data Possession in Public Cloud Storage",International Journal of Security and Its Applications, Vol.7, No.6, 2013, pp.11-20.
24. D. Koo, J. Hur, H. Yoon,"Secure and efficient data retrieval over encrypted data using attribute-based encryption in cloud storage",Computers and Electrical Engineering, 39 (2013), pp. 34–46.
25. G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, "Provable data possession at untrusted stores", In: Proceedings of the 14th ACM conference on computer and communications security, 2007, pp. 598–609.
26. C. Wang, Q. Wang, K. Ren, W. Lou," Privacy-preserving public auditing for data storage security in cloud computing",IEEE proceedings, INFOCOM, 2010,pp. 1–9.
27. J. Yang, H. Wang, J. Wang, C. Tan and D. Yu, "Provable Data Possession of Resource-constrained Mobile Devices in Cloud Computing", Journal of Networks, Vol. 6, No. 7, 2011, pp.1033-1040.
28. Y. Zhu, H. Hu, G. Ahn, S. Yau, "Efficient audit service outsourcing for data integrity in clouds", The Journal of Systems and Software, 85 (2012), pp. 1083– 1095.
29. P.Natu, S.Pachouly, "A Comparative Analysis of Provable Data Possession Schemes in Cloud", International Journal of Computer Science and Information Technologies, Vol. 5, No.6 , 2014, pp.7927-7931.
30. H. Liu, P. Zhang, J. Liu, "Public Data Integrity Verification for SecureCloud Storage", Journal of Networks, Vol.8, No.2, 2013, pp.373-380.
31. S.Worku, C. Xu, J. Zhao, X. He, "Secure and efficient privacy-preserving public auditing scheme for cloud storage", Computers and Electrical Engineering, 40 (2014), pp. 1703–1713.
32. M. Vanitha, T. Jayapratha, K. Subramani, T. Pradeepa, "Elliptic Curve Cryptography Digital Signature Algorithm For Privacy-Preserving Public Auditing For Shared Data In The Cloud", International Journal on Recent and Innovation Trends in Computing and Communication, Vol. 3 Issue. 3, pp.1497 – 1502.
33. D. SudhaDeviand K. Thilagavathy, "An Adaptive Multilevel Security Framework for the Data Stored in Cloud Environment," The Scientific World Journal, (2015), pp. 1-11.
34. A.J. Menezes, V. Oorschot, and S.A. Vanstone, "Handbook of Applied Cryptography", CRC Press Series on Discrete Mathematics and its Applications, CRC Press, Boca Raton, FL, 1997, pp. 320- 383.
35. N. Koblitz, A. Menezes, and S.A. Vanstone, "The state of elliptic curve cryptography", Designs, Codes and Cryptography (19), 2000, pp. 173-193.
36. D. Hankerson, A.J. Menezes, and S. Vanstone, "Guide to Elliptic Curve Cryptography", Springer-Verlag, New York, USA, 2004, pp.78-80.