Computer Life (CPL) ISSN: 1819-4818
DELIVERING QUALITY SCIENCE TO THE WORLD

Audit Scheme ASA for Large Data Integrity Based on TPA

Wanle Chi [a]

Department of Information Technology, Wenzhou Vocational &Technical College, Wenzhou, China

[a]358455713@qq.com

Abstract: In order to achieve the integrity auditing of big data in cloud computing, a big data integrity audit scheme based on TPA was proposed, namely ASA. The Merkle hash tree (MHT) was extended to a numbered Merkle hash tree (IMHT). Bilinear pair mapping was used to construct the encryption and authentication of big data messages. The results showed that the scheme was better than other schemes in security. It had an absolute advantage in performance. Therefore, the scheme protects the privacy of large data, supports public auditing and updates of dynamic large data, and ensures the integrity and effectiveness of large data files.
Keywords:TPA; big data; audit program; ASA.

---

1. INTRODUCTION

Big data files for users on cloud servers may face security issues. For cloud service users, a safe, efficient and feasible approach is found without local data backup. The integrity of the large data files on the cloud server is audited [1]. The volume of large data is huge. There are many types of data. The value density is low. The size and number of large data files on the cloud server are very large. The user's own computing resources are very limited. Therefore, it is impossible for users to download all large data files from the cloud server when the integrity of large data is verified. The effective and correct method of verifying the user's large data integrity in the cloud server is an important security problem to be solved urgently [2, 3]. Most of the existing research schemes for data integrity audit of cloud servers are random sampling. Users can complete the integrity verification as long as they download some large data files. However, there are many problems in these solutions [4].

Based on the cloud computing environment of big data security storage mechanism, a TPA based data integrity audit plan ASA is proposed to solve the above problems. Based on the characteristics of large data, random sampling is adopted. The feasibility of the scheme is effectively verified. In the process of cloud server storage, the problem of large data integrity audit is solved. When using a large data file, the integrity and effectiveness of a legitimate user is ensured.

2. STATE OF THE ART

In recent years, the rapid development of information technology has brought unprecedented changes to human life. Among the many emerging information technologies, cloud computing and large data are the most typical two representatives. There is a close relationship between them. The architecture

of cloud computing provides a basic platform for information storage and data mining processing for large data. Large data technology has promoted the rapid development and improvement of cloud computing. However, the security issues involved in all phases of the whole large data chain based on cloud computing are becoming more and more prominent. In response to the existing problems, relevant scholars have proposed a provable data possession (PDP) scheme. The scheme uses a combination of RSA-based homomorphic verification labels and random sampling. The data owner can publicly verify without having to download all the data, that is, it only needs to use the public key to complete the verification process. However, the scheme is only applicable to verification of static data. After the data is stored to the cloud server, the data owner does not add, change or delete the data. In order to support the validation of dynamic and updatable data, many solutions are proposed. These schemes do not support public verification and do not have the function of batch auditing [5].

Public validation is supported. When users verify the large data integrity of the cloud server, the communication and computing overhead is greatly reduced. Many schemes have introduced the third party public auditor (TPA). It objectively provides a fair and credible audit result for data owners and cloud servers. However, there are still a lot of problems. Many schemes cannot provide data privacy protection, and data content may be leaked to auditors or attackers. The scheme lacks a validation process for the number of data blocks. Distrusted cloud servers can use other data blocks and their labels instead of challenging data blocks and their labels, and then send them to third party auditors. Part of the scheme does not support multiuser batch audits and requires an additional trusted direction auditor to send a commitment. However, it is not practical to add an additional trusted party in cloud storage services. These schemes only support the insertion, change, and deletion of fixed - size data blocks, which is called coarse-grained updates. The relevant scholars have proposed a large data integrity audit scheme that can support fine-grained updates (updates of arbitrary length data blocks). This scheme adds an authentication process between users and third-party auditors, to prevent third-party auditors from causing DDOS attacks by sending a lot of challenge information to the cloud server. However, this solution also lacks the verification of the data block number of the big data on the cloud server. It cannot provide big data privacy protection [6].

3. METHODOLOGY

3.1 System model

As shown in Figure 1, the ASA system model consists of three subjects: client, cloud service provider (CSP) and third-party auditor (TPA). Its definition is as follows:

Client: Users refer to massively large data files that are stored in a public cloud server in order to reduce the storage and maintenance burden. The users can be individual users or group users.

CSP: Cloud servers are managed by cloud service providers. It has a huge amount of storage capacity and computing resources to store the user's various data.

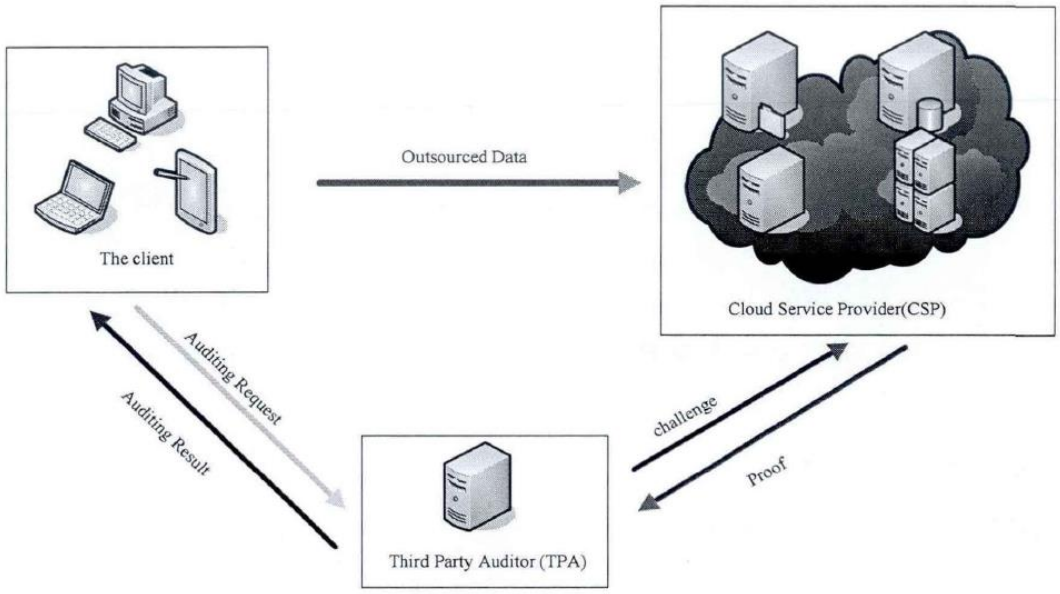

Figure. 1 ASA system model

TPA: The TPA sends a challenge message to the cloud server. Then, based on the server's reply message, the authentication process is completed, thereby performing auditing (verification) of the big data integrity stored in the cloud server.

3.2 The design of IMHT

The Merkle hash tree is expanded into a numbered Merkle Hash Tree (IMHT). The structure of IMHT is used to construct the verification information of the data block. It not only verifies the integrity of data blocks in big data files, but also verifies the number of data blocks to prevent semi-trusted or untrusted cloud servers. False audit evidence is falsified by using other legitimate data blocks as a challenge data block to deceive the verifier's attack. The main idea of MHT is to add numbered designs on the basis of MHT. That is to say, each node information contains two parts: one is the calculated hash value and the other is the number of the node. In a IMHT, each data block is calculated by hash. The obtained hash value and the number of the data block are used as information on the leaf node. All the hash values of other node information except for the leaf node are all calculated by the hash of the connection information of its child nodes. The numbering is added to the numbered part of the child's node information [7]. This structure of the MHT is ideal for quick verification of the integrity of user data blocks and data block numbers. When validated, only the information on a path from the leaf node to the root node needs to be processed. This information is called auxiliary authentication information, which is abbreviated as AAI. The root node information can be calculated using the leaf node and its corresponding AAI. If the root node information is correct, the integrity of the data block corresponding to the leaf node is not destroyed. The number of the verified data block is correct.

Figure 2 is an example of IMHT. The file m is divided into 8 data blocks {m1, m2, ..., m8}. The hash values {h(m1), h(m2), ..., h(m8)} and the corresponding numbers are successively used as the leaf nodes of the IMHT. The information of node C is {hc, Ic}. hc=h(h(m1) ||1||h(m2) ||2), Ic=1+2=3. The information of node A is {hA, IA}. hA=h(hc)||3||hD) ||7), Ic=3+7=10. The information of the root node is {hg, Ig}. hg=h(hA)||10||hg) ||26), Ig=10+26=36.

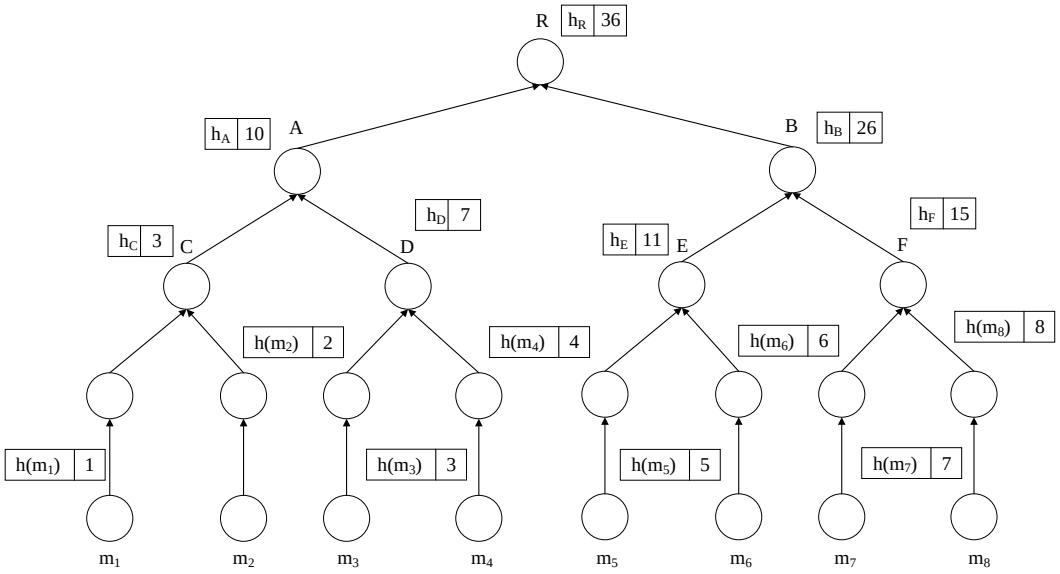

Figure. 2 IMHT instance

3.3 System initialization

Users randomly select x←Zq as their private key. x=gx is the public key of its own. The public private key pair of TPA is (y, Y), and the public private key pair of CSP is (z, Z). Users randomly select a set u={uj∈GI}, j∈[1,s], and calculate {e(uj,X)}j∈(1,s). In a word, in the proposed scheme, {x, y, z} are private parameter. {X, Y, Z, g{(uj, X)} j∈(1,s)} are public parameters.

3.4 File uploading

The user divides the big data file m into n data blocks m=)m1,mn). Each data block mi has the same length l bits. Then, each data block mi is divided into s[s=[l/|q|]] data sub-blocks mi=(mi1,,mis). The length of each data sub-block is |q| bits. Thus, the user's big data file is divided into n × s data sub-blocks in total. name is the name of the big data file. Users use their own private key to sign file names and related parameters. ft=name ‖ n ‖ u ‖ Ssigx (name ‖ n ‖u) is used as the label of the big data file m. Then, the user generates a homomorphic label for each data block. These homomorphic labels form a label set φ={δi} i=1n. It is called the signature of file m. Thereafter, the user constructs the IMHT using {h(mi), i}ni=1, as the leaf nodes, and calculates the root node information R. R and the current timestamp t are signed sig=(h(R‖t)) z with their own private keys. Finally, the user sends {m, φ, ft,sig,t} to the CSP, and removes { m, φ,sig,t } from the local store.

After the CSP receives the message sent by the user, the CSP first constructs the IMHT using {h(mi), i}ni=1, as the leaf node in turn, and calculates the root node information R. Through the following equation, CSP detects the illegal tampering and destruction of files in the process of uploading:

$$e(sig, g) \stackrel{?}{=} e(h(R^* || t), X) \tag{1}$$

3.5 Integrity audit

The integrity audit consists of three steps: User → TPA, TPA → CSP and CSP → TPA. The detailed description is as follows:

User → TPA: The user requests to know the TPA's identity information VID. AUTH=h (e (Y, Z) x,Q) is calculated by x (user private key), VID, ft (file tag), Y (TPA and public key) and Z (CSP public key). {AUTH, Q} is requested and sent to the TPA as an audit request. After receiving the audit request information sent by the user, the TPA first verifies the validity of the request information and the correctness of the sender of the information by judging the following equation:

$$AUTH \stackrel{?}{=} h(e(X,Z)^{y},Q) \tag{2}$$

The TPA then verifies the signature Ssigx(name||n||u) in ft with the user's public key X. As long as one of these two verifications fails, the TPA will send "FALSE" to deny the audit request. If these two tests are all passed, the audit process will continue.

TPA→CSP: The TPA randomly chooses k∈{0,l}x and randomly selects one of its C-element subset I={s1,s2,...sc} from the set [1, n]. chal= {AUTH, Q,I,k} is served as audit challenge information and sent to the CSP. After receiving the audit challenge information sent by the TPA, the CSP verifies the validity of the challenge information and the correctness of the sender of the message by judging whether the following equation holds:

$$AUTH \stackrel{?}{=} h(e(X,Y)^{Z},Q) \tag{3}$$

If the verification fails, the CSP will send "FALSE" to reject this challenge message. If the verification is successful, the audit process will continue.

CSP→TPA: Through calculation, CSP finally takes P={θ,δ,{h(mi),i,Ωi}i ∈ I,sig,t} as the TP information and sends it to TPA. After receiving the audit reply message sent by the CSP, the TPA firstly determines the correctness of the time stamp t in the reply message and the set I of the challenge data block number. If not correct, TPA will send "FALSE" to reject this reply message. If correct, the audit process will continue. TPA will calculate the IMHT root node constructed by it and then determine if the following equation holds:

$$e(sig,g) \stackrel{?}{=} e(h(R||t),X) \tag{4}$$

If equation (4) holds, the TPA will calculate vi=fk(i),i∈I. Then, the following equation is judged.

$$e(\delta,g) \stackrel{?}{=} \theta \bullet e(\prod_{i=s_i}^{s_g} h(m_i)^{v_i},X) \tag{5}$$

If both of these equations are true, the audit algorithm will output "TRUE". It means integrity audit. In other words, the user's big data integrity stored on the cloud server has not been compromised. Otherwise, the audit algorithm will output "FALSE". It means the integrity audit did not pass. Finally, the TPA sends the integrity audit result to the user.

3.6 Update of dynamic data

The scheme uses the structure of MHT to construct the verification information of data blocks so as to achieve the purpose of fully supporting the efficient dynamic data updating operation. When big data is updated, the user does not need to retrieve the entire big data file, and only needs to send the data update information to the cloud server. Cloud server can help users update the appropriate data blocks and reduce the user's computing overhead. The update operations of dynamic data include data changes, data inserts, and data deletions. Suppose that the user's big data file m and its signature

Φhave been stored on CSP. Root node R and timestamp t and their signature sig have also been stored on CSP. The update operation of these three-dynamic data is described in detail.

For large data storage, data change is one of the most commonly used data update operations [8]. The data change operation means that only the data block is replaced. The logical tree structure of the data has not changed. It just needs to update the hash of all associated nodes on the path from the leaf node associated with this block to the root node of the tree and update the sign of the block's label and root node information. Finally, the user requests the TPA to perform a data integrity verification.

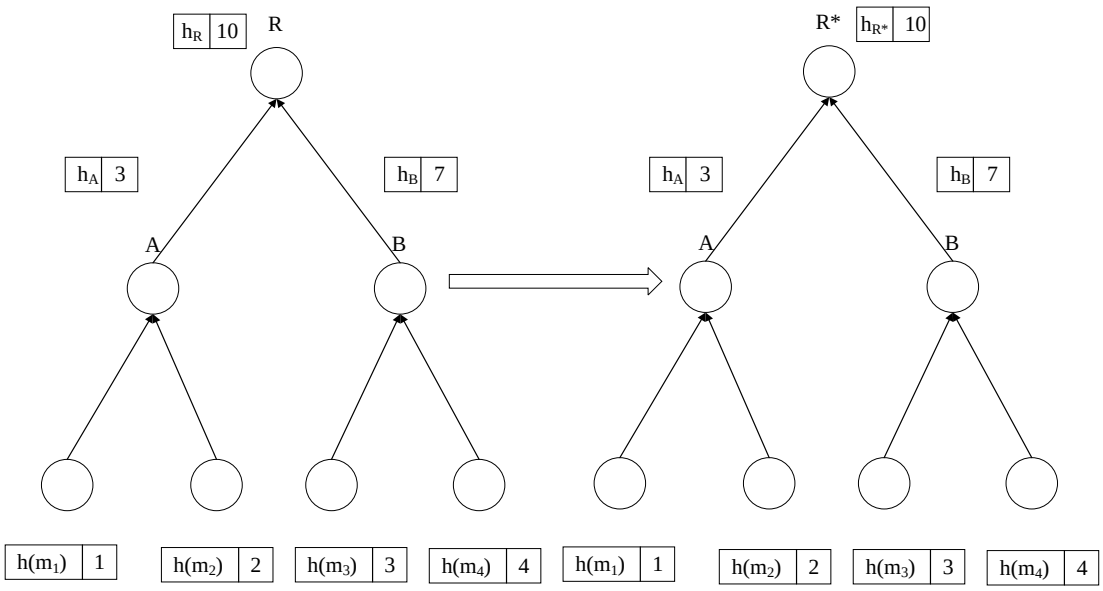

Figure. 3 Example of data changes

Figure 3 is an example of IMHT changes when performing data changes. Assuming that the user needs to change the second data block to m2, the CSP first needs to find the second leaf node {h(m2),2}. At the same time, its auxiliary verification information Ω2={h(m1),1, l}, {hB,7,r} is reserved. It is changed to {h(m2),2}. Then, the information about all the siblings of the second leaf node to the root node of this path is updated. That is, the number is constant and only the hash values of nodes A and R need to be recalculated.

Data insertion is the insertion of a new data at a specific data block location. All subsequent data blocks are shifted back by 1 bit, resulting in a change in the logical tree structure of the data.

Figure 4 is an example of IMHT variation when performing data insertion. The user needs to insert the data block m * after the second data block m2. The CSP first needs to find the second leaf node {h(m2),2} while preserving its auxiliary verification information Ω2={h(m1),1, l}, {hB,7, r}. The new leaf node {h(m*),3} is inserted. 1{h(m3),3} is changed to {h(m3),4} for all the leaf nodes after {h(m*),3}. {h(m3),4} is changed to {h(m3),5}. The number of all father nodes associated with these nodes is recalculated. ({hB,7} is changed to {hB,9}). After that, a node {hC, C} is added as the parent node of node h (m2), 2 and node {h (m *), 3}. hC=h(h(m2) ||h(m*) ||3), P=5. Then, the related information of all the brotherhood nodes on the path of the C node to the tree root node is updated. The information of the nodes A and R is recalculated.

Data deletions are the opposite of data insertion. The delete operation of the data means deleting a specific block of data. The location of all the back blocks will move forward 1 bits, and the logical tree structure of the data changes.

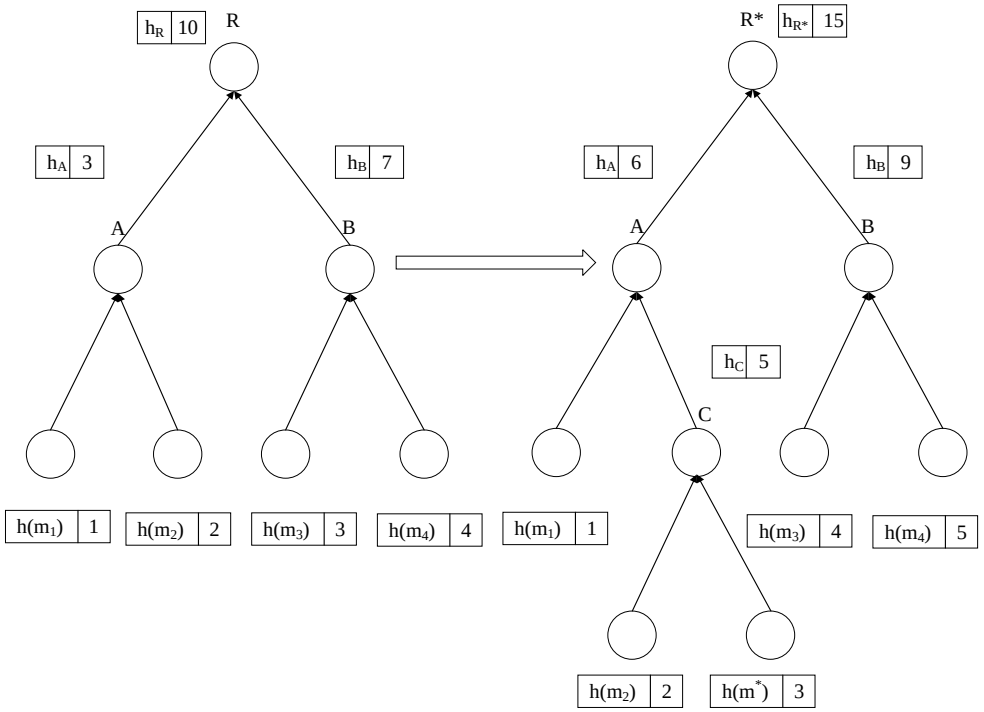

Figure. 4 Example of data insertion

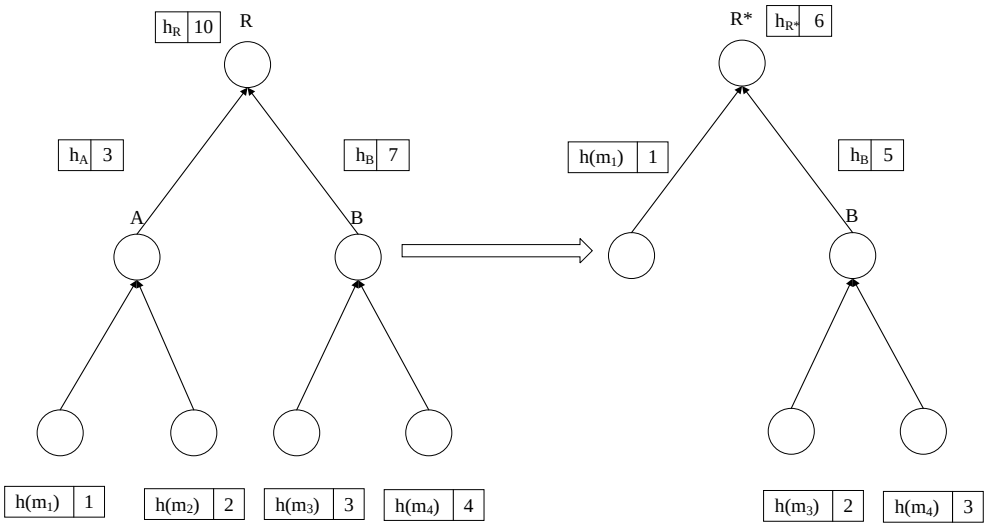

Figure. 5 Examples of data deleting

Figure 5 shows an example of MHT changes when data deletion is performed. The user needs to delete the second data block m2. First, the CSP needs to find the second leaf node {h(m2),2}, retains its auxiliary verification information Ω2={h(m1),1, l}, {hB,7, r}, and deletes it and its parent node {hA,3}. 1{h(m3),3} is subtracted {h(m3),2} from all the leaf nodes after {h(m2),2}. {h(m4),4} is changed to {h(m4),3}. The number of all father nodes associated with these nodes is recalculated

from the bottom to the ground. ({hB,7} is changed to {hB,5}). Then, the sibling node {h (m1), 1} is updated with information about all sibling nodes on the path to the root node. That is, the information of the node is recalculated.

4. RESULT ANALYSIS AND DISCUSSION

The ASA solution can support dynamic big data updates, protect big data privacy, counterfeit and replay attacks. It has the authentication function. In order to better reflect the superiority of the security performance of the ASA scheme, Wang's scheme of the two latest research programs in this field, the scheme of Liu and the safety performance of the ASA scheme are compared. The results are shown in Table 1. In the table, "√" means that the solution can resist such attacks or achieve the corresponding performance goals. "×" means that the program cannot resist such attacks or fail to achieve the corresponding performance targets.

Table. 1 Comparison of safety performance

Scheme	Update of dynamic large data	Privacy protection of large data	Forgery attack	Replay attack	Authentication
Wang	√	√	×	×	×
Liu	√	×	×	×	√
ASA	√	√	√	√	√

Using NS2, Wang's solution, Liu's solution and ASA solution were simulated and compared. The whole process of integrity auditing is mainly the interaction between TPA and CSP. Therefore, the object of performance simulation mainly focuses on three aspects: communication overhead between TPA and CSP, CSP calculation overhead, and TPA calculation overhead. The simulation results and detailed analysis are illustrated as follows.

4.1 Communication overhead

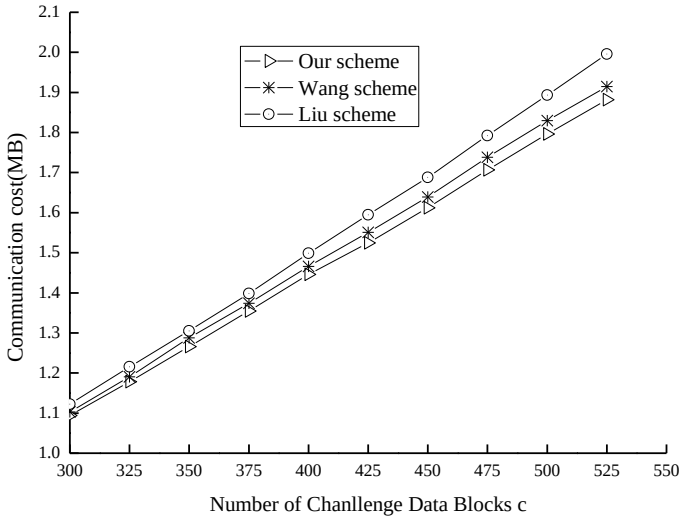

Figure. 6 Comparison of communication overhead between TPA and CSP in three schemes

Table. 2 The communication overhead between TPA and CSP and the linear relationship between c

Scheme	Communication overhead
Wang	3588c+25070
Liu	3666c+9488
ASA	3568c+2036

As shown in Figure 6, a comparison is given between Wang's solution, Liu's solution, and the communication overhead of TPA and CSP in the integrity audit process of the ASA solution. It is clear that the communication overhead between TPA and CSP is directly proportional to the number C of challenge data blocks. The ASA scheme has lower communication overhead than the other two schemes. The linear relationship between TPA and CSP communication overhead and C is shown in Table 2. The following are the reasons for the simulation results.

For the ASA scheme, TPA will challenge the information in the integrity audit process. chal= {AUTH, Q,I,k} is sent to CSP. Then, CSP sends the reply information P={θ,δ,{h(mi),I,Ωi}i∈l,sig,t} to TPA. AUTH, Q, θ, δ, h(mi), sig, and t are points in the elliptic curve. Both of their coordinate values are |p| bits. The entire file contains n data blocks. Therefore, the maximum length of chal={AUTH, Q,I,k} is clog2 n+k+4|p| bits. The maximum length of P={θ,δ,{h(mi),I,Ωi}i ∈ l,sig,t} is clog2(2n)·[2|p|+log2n]+8|p| bits. Therefore, the maximum communication overhead between TPA and CSP is [2clog2(2n) +12] |p|+clogn(4n)·log2n+k bits.

For Wang's scheme, in the integrity audit process, the TPA sends the challenge information chal={I,vi}i∈l to CSP. After that, the CSP sends the reply information P={δ,{μj, Rj}i≤j≤s, {h(mi), Ωi}i∈l} back to the TPA. It is calculated that the maximum communication overhead between TPA and CSP is [2s+2clog2(2n) +2] |p|+s|q|+cd+clog2n bits.

For Liu's scheme, in the integrity audit process, the TPA sends the challenge information to the CSP with chal={sigAUTH,{VID}PKcss,{I,vi,j}i∈I}. After that, the CSP sends the reply information P={{μk}k∈[1,w], δ,{ h(mi),Ωi}i∈I,sig} back to the TPA. It is calculated that the maximum communication overhead between TPA and CSP is (2c+8)|p|+(w+c)|q|+c(2p+1)log2n bits.

Overall, the Liu scheme has higher communication overhead than the ASA scheme. The reply message sent by the CSP to the TPA contains a series of information {μi}j ∈ [1,s]. The communication overhead of the Wang scheme is more than that of the ASA scheme. In order to protect the user's data privacy, CSP sends the response information to the TPA. The information related to user data is encrypted using mask technology. After encryption, the information becomes {μj,Rj}j∈[1,s], which increases the traffic. In the ASA scheme, the CSP is sent to the TPA response information. The information related to the user's data is encrypted by the way of bilinear pairing. θ=Πe(μj,X)uj only needs to be sent to the TPA, thus greatly reducing the communication overhead.

4.2 The computing overhead of CSP

As shown in Figure 7, a comparison of the computational costs of CSP in Wang's solution, Liu's solution and ASA solution in the integrity audit process is given. It can be clearly seen from the figure that the CSP calculation overhead is directly proportional to the number C of challenge data blocks.

The ASA scheme and the Wang scheme are basically the same for the computing overhead of CSP. It is slightly higher than the Liu scheme. The computing overhead of CSP and the linear relationship between C are shown in Table 3. The following are the reasons for the result of the simulation.

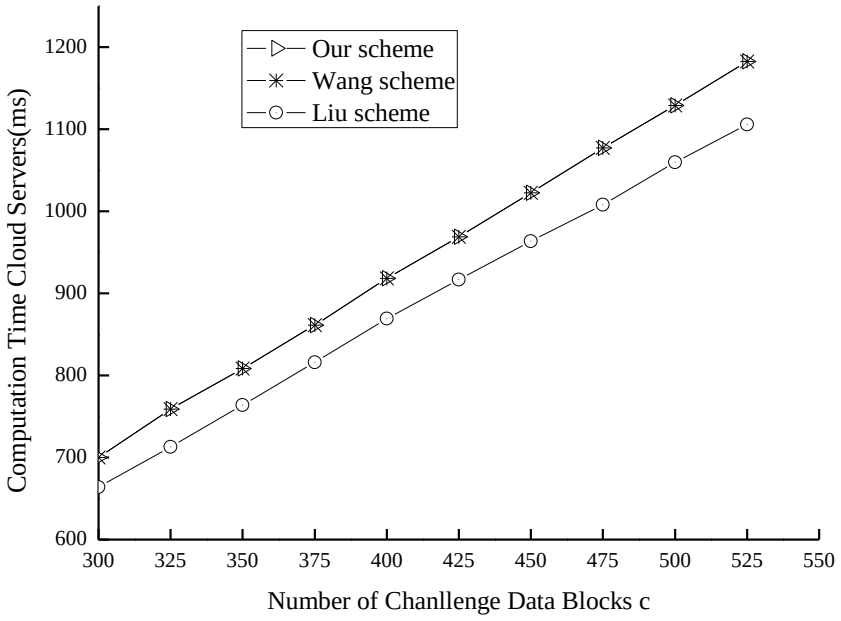

Figure. 7 Comparison of CSP's computing overhead in three schemes

Table. 3 The computation overhead of CSP and the linear relation of c

Scheme	The computing overhead of CSP
Wang	2.15c+56.83
Liu	2.15c-0.71
ASA	2.15c+54.19

After analysis, for the ASA scheme, the CSP needs to compute $\delta=\Pi\delta ivi$ and μj in the integrity audit process. Finally, all calculation overhead of CSP is (c-1) sTadd(|q|) +csTmul(d,|q|)+(c+s-2)Tmul(G1)+cTexp(d,G1)+sTexp(|q|,G1).

After analysis, for the Wang scheme, all the computing overhead of CSP is csTadd(|q|) +(c+1)sTmul(d,|q|)+(c-1)Tmul(G1)+cTexp(d,G1).

After analysis, for the Liu scheme, all the computing overhead of CSP is (c-1) sTadd(|q|)+csTmul(d,|q|)+(c-1)Tmul(G1)+cTexp(d,G1).

In general, in order to protect user data privacy, the ASA scheme is divided into s data blocks for each data block mi. mi=(mi1.....,mis). The CSP needs to encrypt the aggregated data set $\{\mu j\}j\in[1,s]$. As a result, a small amount of computation is added.

4.3 The computing overhead of TAP

As shown in Figure 8, a comparison of the calculation overhead of TPA in Wang's solution, Liu's solution, and the ASA solution in the integrity audit process is given. It can be clearly seen from the figure that the TPA calculation cost is proportional to the number C of challenge data blocks. The ASA scheme has a lower TPA calculation cost than the other two schemes. The linear relationship between TPA calculation cost and C is shown in Table 4. The following are the reasons for the result of the simulation.

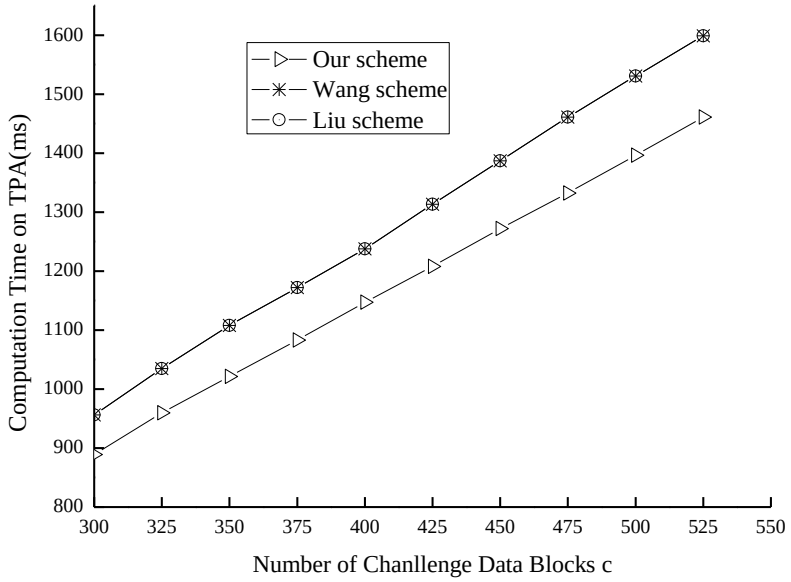

Figure. 8 Comparison of TPA's computing overhead in three schemes

Table. 4 The computation overhead of TPA and the linear relation of c

Scheme	The computing overhead of TPA
Wang	2.87c+97.12
Liu	2.87c-94.83
ASA	2.87c+39.52

After analysis, for the ASA scheme, in the integrity audit process, all the computing overhead of TPA is cTprng(d)+cTmap(G1) +cTexp (d, G1) +(clog2n+1) Thash(G1) +4Tmap (G1, G2).

After analysis, for the Wang scheme, in the integrity audit process, all the computing overhead of TPA is (c+2s-1) Tmul(G1) +(c+1) Texp(d, G1)+(s+1) Texp (|q|,G1)+(clog2n+2)Thash (G1)+4Tmap (G1,G2).

After analysis, for the Liu scheme, in the integrity audit process, all the computing overhead of TPA is (c+s-1) Tmul(G1) +cTexp (d, G1) +sTexp(|q|, G1)+(clog2n+1)Thash(G1)+4Tmap(G1,G2).

In general, the ASA scheme transfers part of the computational overhead required by the audit process from TPA to CSP through the design of the scheme algorithm. CSP has more computing resources and capabilities. This transfer is acceptable.

Comparing the computation overheads of the three schemes CSP and TPA, it can be found that the total computational cost of the ASA scheme is lower than that of Wang's scheme. It is basically the same as Liu's plan. Considering the communication overhead and computational overhead, it can be concluded that the ASA scheme is higher than the other two schemes.

5. CONCLUSION

The ASA scheme is introduced in detail. The various stages of the program such as IMHT design, system initialization, file upload, integrity audit, dynamic data updates are described in detail. The ASA program was analyzed for safety by comparison with other methods. The superiority of the ASA scheme is proved. Finally, the performance simulation and comparison of the ASA scheme and similar scheme are carried out. From the results, the security of the ASA solution is improved compared with other solutions. Communication costs and computing costs have a good efficiency.

REFERENCES

[1] Sookhak, M., Talebian, H., Ahmed, E., Gani, A., & Khan, M. K. (2014). A review on remote data auditing in single cloud server: taxonomy and open issues. Journal of Network & Computer Applications, 43(5), 121-141.

[2] Buribayeva, G., Miyachi, T., Yeshmukhametov, A., & Mikami, Y. (2015). An autonomous emergency warning system based on cloud servers and sns. Procedia Computer Science, 60(1), 722-729.

[3] Chen, L., Hu, Y., Zhang, F., Duan, W., & Yu, P. (2013). Performance improving design on cloud computing for agricultural products safety traceability system. Transactions of the Chinese Society of Agricultural Engineering, 29(24), 268-274.

[4] Nagar, N., & Suman, U. (2017). Reliable and enhanced third party auditing in cloud server data storage. International Journal of Security & Its Applications, 11(7), 59-72.

[5] Fujinoki, H. (2013). In-line auditing and real-time lineage summaries to maintain ownership of information stored in cloud servers. Journal of Network & Information Security, 1(2).

[6] Marshal, M. S. V. (2013). Secure audit service by using tpa for data integrity in cloud system. International Journal of Innovative Technology & Exploring Engineering, 3(4).

[7] Appelbaum, D. (2015). Securing big data provenance for auditors: the big data provenance black box as reliable evidence. Journal of Emerging Technologies in Accounting, 13(1), 17-36.

[8] Kalaivani, A., & Ananthi, B. (2015). One step integrity management approach based public auditing in shared data in cloud environment for privacy preservation. International Journal of Applied Engineering Research, 10(15), 35936-35940.