International Journal of Emerging Technology and Innovative Engineering
Volume 4, Issue 2, February 2018 (ISSN: 2394 – 6598)

Paper Received on: 09.02.2018
Paper Published on: 17.02.2018

DATA INTEGRITY IN CLOUD COMPUTING USING SHA256 ALGORITHM

P.Priyanka Surana K.Rathika N.Renuga Devi C.Sankaresh

Department of Computer science and Engineering
Kalasalingam University

ABSTRACT

Cloud computing is defined as the computing which enables the term for delivery of hosted services over the internet. It has several attractive benefits for business and end users. Cloud computing services is categorized in three platforms **Iaas, Paas, Saas**. For securing data in cloud storage cryptographic hash functions are used. **SHA-1 (Secure Hash Algorithm)** is a hash function which is Slower in computation comparing MD5,7 for security vulnerabilities. The main disadvantage of **SHA-2** is consuming device which don't understand the SHA-2 algorithm except failure or an error message. **SHA-3** is designed as a good hashing function but it does not have a good password hashing scheme. **SHA-256** has two main components 1) SHA-256 compression function 2) SHA-256 message schedule. The advantage is that it is far more efficient.

I.INTRODUCTION

Cloud computing is a **platform independent** as this software is not required to be installed locally on the PC.The Cloud Computing is making our business applications **mobile** and **collaborative.**
Cloud Computing provides the applications as utilities over the internet. It allows us to create, configure, and customize the business applications online.

1.1. Data Integrity

Data integrity is an integrity in terms of data security is nothing but grants that data which can only be accessed or modified by only the authorized person. Data integrity is very important in cloud challenges. There is lot of research still going on in this field to provide secure and efficient data integrity in cloud computing. Researchers have given many solutions to focus on resolving the issues of data integrity but in cloud computing there are several techniques to solve this problem of data integrity checking, many researchers have proposed different systems and security models.

II.SERVICES OF CLOUD COMPUTING

There are various services offered by cloud computing :
IaaS: In this organization make use of unlimited cloud infrastructure.
SaaS: It allows user to access the software without worrying about storage.
Paas: It allows to run applications on the cloud service's platform without worrying about maintaining hard drives.
Managed Services: These are applications used by cloud service provider.
Utility Services: Companies that need to store a lot of data can store all of their data remotely and can even create a virtual data center.

III.TECHNIQUES OF DATA INTEGRITY

In cloud computing the techniques of data integrities taking a great task among these current challenges. They are

- **Provable Data Possession(PDP)**
- **Basic PDP based on MAC**
- **Scalable PDP**
- **Dynamic PDP**
- **HAIL**

Data integrity consists of some methods:

3.1Provable Data Possession(PDP):

Provable Data possession (PDP) is a technique in data integrity used in remote servers. In PDP if a client has stored data in an untruthful server it can verify the server possesses the original data without retrieving it. It uses Key generation algorithm.

Advantages:

- This technique gives a strong proof of data integrity.
- Support Block less verification.
- Allows public verifiability.

Limitations:

- No dynamic support
- Unbound no. of queries

3.2.Basic PDP Scheme based on MAC

Basic PDP Scheme based on MAC is one of the fine technique in data integrity using cloud computing. Message Authentication Code (MAC) of the whole file with a set of secret keys and stores them locally before outsourcing it to CSP. It Keeps only the computed MAC on local storage, sends the file to the CSP, and deletes the local copy of the file F. It uses Message authentication Code algorithm.

Advantages:

- Simple Technique.
- It has strong Integrity of data.

Limitations:

- It has Limited number of verifications with secret keys.
- The data owner has to retrieve the full file from the server in order to compute new MACs, which is not possible for large file.

3.3Scalable PDP:

Scalable PDP uses the symmetric encryption in data integrity original PDP uses public key to reduce computation overhead. It uses Cryptographic Hash Function algorithm.

Advantages:

- The main advantage of scalable PDP is it does not require bulk encryption.

Limitations:

- Number of updates with limited challenges.

3.4Dynamic PDP:

It allows several operations like update, delete, insert, modify etc. In this technique it uses rank based along with a skip list for inserting and deleting functions. It uses Rank-based authenticated skip list algorithm.

Advantages:

- It operates in full dynamic operations.

Limitations:

- It has computational complexity.

3.5.HAIL

It has high-availability and integrity layer in cloud storage. It uses Hash Functions.

Advantages:

- Data's can be stored in multiple cloud.

Limitations:

- It is applicable only for static data.

IV.SCALABLE PDP

It is the higher version of PDP. The main difference is that to reduce the computation PDP uses public key whereas scalable PDP uses symmetric encryption. It has dynamic operation on remote data.It has limited number of updates. The bulk of encryption is not required in this. It is based on symmetric key which is more efficient than public key so it does not offer public verifiability. It uses MHT and PDP algorithm.

Disadvantages of this algorithm

The main disadvantages of these algorithms are It lacks in randomness hence it results in the client can't easily deceive the server.

PROPOSED METHOD:

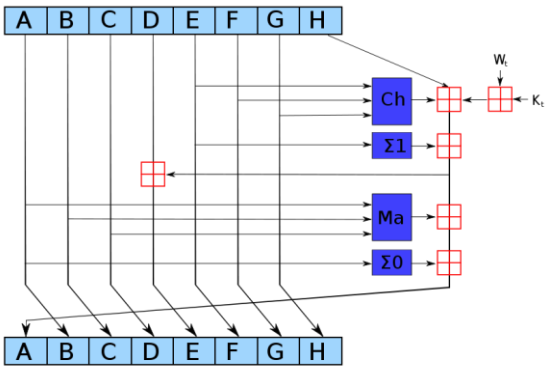

Fig 4.1SHA-2 Algorithm

This is a set of cryptographic hash functions. These functions are mathematical operations which run digital data.

Initialize variables

(first 32 bits of the fractional parts of the square roots of the first 8 primes 2..19):

h0 := 0x6a09e667
h1 := 0xbb67ae85
h2 := 0x3c6ef372
h3 := 0xa54ff53a
h4 := 0x510e527f
h5 := 0x9b05688c
h6 := 0x1f83d9ab
h7 := 0x5be0cd19
Initialize table of round constants
(first 32 bits of the fractional parts of the cube roots of the first 64 primes 2..311):
k[0..63] :=
0x428a2f98, 0x71374491, 0xb5c0fbcf, 0xe9b5dba5, 0x3956c25b, 0x59f111f1, 0x923f82a4, 0xab1c5ed5,
0xd807aa98, 0x12835b01, 0x243185be, 0x550c7dc3, 0x72be5d74, 0x80deb1fe, 0x9bdc06a7, 0xc19bf174,
0xe49b69c1, 0xefbe4786, 0x0fc19dc6, 0x240ca1cc, 0x2de92c6f, 0x4a7484aa, 0x5cb0a9dc, 0x76f988da,
0x983e5152, 0xa831c66d, 0xb00327c8, 0xbf597fc7, 0xc6e00bf3, 0xd5a79147, 0x06ca6351, 0x14292967,
0x27b70a85, 0x2e1b2138, 0x4d2c6dfc, 0x53380d13, 0x650a7354, 0x766a0abb, 0x81c2c92e, 0x92722c85,
0xa2bfe8a1, 0xa81a664b, 0xc24b8b70, 0xc76c51a3, 0xd192e819, 0xd6990624, 0xf40e3585, 0x106aa070,
0x19a4c116, 0x1e376c08, 0x2748774c, 0x34b0bcb5, 0x391c0cb3, 0x4ed8aa4a, 0x5b9cca4f, 0x682e6ff3,
0x748f82ee, 0x78a5636f, 0x84c87814, 0x8cc70208, 0x90befffa, 0xa4506ceb, 0xbef9a3f7, 0xc67178f2

Pre-processing:
append the bit '1' to the message
append k bits '0', where k is the minimum number >= 0 such that the resulting message
length (in *bits*) is congruent to 448 (mod 512)
append length of message (before pre-processing), in bits, as 64-bit big-endian integer

Process the message in successive 512-bit chunks:
break message into 512-bit chunks
for each chunk
break chunk into sixteen 32-bit big-endian words w[0..15]

Extend the sixteen 32-bit words into sixty-four 32-bit words:
fori**from** 16 to 63
s0 := (w[i-15] **rightrotate** 7) **xor** (w[i-15] **rightrotate** 18) **xor** (w[i-15] **rightshift** 3)
s1 := (w[i-2] **rightrotate** 17) **xor** (w[i-2] **rightrotate** 19) **xor** (w[i-2] **rightshift** 10)
w[i] := w[i-16] + s0 + w[i-7] + s1

Initialize hash value for this chunk:
a := h0
b := h1
c := h2
d := h3
e := h4
f := h5
g := h6
h := h7

Main loop:
fori**from** 0 to 63
s0 := (a **rightrotate** 2) **xor** (a **rightrotate** 13) **xor** (a **rightrotate** 22)
maj := (a **and** b) **xor** (a **and** c) **xor** (b **and** c)
t2 := s0 + maj
s1 := (e **rightrotate** 6) **xor** (e **rightrotate** 11) **xor** (e **rightrotate** 25)
ch := (e **and** f) **xor** ((**not** e) **and** g)
t1 := h + s1 + ch + k[i] + w[i]

h := g
g := f
f := e
e := d + t1
d := c
c := b
b := a
a := t1 + t2

Add this chunk's hash to result so far:
h0 := h0 + a
h1 := h1 + b
h2 := h2 + c
h3 := h3 + d
h4 := h4 + e
h5 := h5 + f
h6 := h6 + g
h7 := h7 + h

Produce the final hash value (big-endian):
digest = hash = h0 **append** h1 **append** h2 **append** h3 **append** h4 **append** h5 **append** h6 **append** h7

From this above SHA-2 algorithm we can solve the disadvantages of existing algorithm.

V.RESULTS AND ANALYSIS

Algorithm and variant		Output size (bits)	Internal state size (bits)	Block size (bits)	Max message size (bits)	Rounds	Operations	Security (bits)
MD5 (as reference)		128	128 (4 × 32)	512	Unlimited[37]	64	And, Xor, Rot, Add (mod 2^{32}), Or	<64 (collisions found)
SHA-0		160	160 (5 × 32)	512	$2^{64} - 1$	80	And, Xor, Rot, Add (mod 2^{32}), Or	<80 (collisions found)
SHA-1		160	160 (5 × 32)	512	$2^{64} - 1$	80		<80 (theoretical attack[38])
SHA-2	*SHA-224*	224	256 (8 × 32)	512	$2^{64} - 1$	64	And, Xor, Rot, Add (mod 2^{32}), Or, Shr	112
	SHA-256	256						128
	SHA-384	384	512 (8 × 64)	1024	$2^{128} - 1$	80	And, Xor, Rot, Add (mod 2^{64}), Or, Shr	192
	SHA-512	512						256
	SHA-512/224	224						112
	SHA-512/256	256						128
SHA-3	*SHA3-224*	224	1600 (5 × 5 × 64)	1152	Unlimited[39]	24[40]	And, Xor, Rot, Not	112
	SHA3-256	256		1088				128
	SHA3-384	384		832				192
	SHA3-512	512		576				256
	SHAKE128	*d* (arbitrary)		1344				min(*d*/2, 128)
	SHAKE256	*d* (arbitrary)		1088				min(*d*/2, 256)

V. CONCLUSION

In cloud computing the data integrity is most challenging and it is a burning security issue. As considering the importance of data integrity, in this paper different techniques and their advantages and disadvantages are explained. The analytical study briefly compares this technique. From this survey paper it is concluded that there is need to design efficient, dynamic secure data integrity technique.

VI.FUTURE SCOPE

From the above study it is clear that all these techniques which are surveyed in this paper have some advantages as well as some disadvantages. All these are lack in proper data integrity mechanisms, supporting dynamic data operations, and by high resource and computation cost. The only drawback of this technique is, it works only for static data and the data is not prevented. So expanding the scope of this paper will be the future work.

VII.REFERENCE

[1]Hewitt, C. (2008) ”ORGs for scalable, robust, privacy friendly client Cloud Computing Environment in IEEE Proceedings Volume 12 Issue 5, September 2008.

[2] Chandran S. and Angepat M., “Cloud Computing: Analyzing the risks involved in cloud computing environments,” in Proceedings of Natural Sciences and Engineering, Sweden, 2010.

[3] Balachandra Reddy Kandukuri, Ramakrishna Paturi V, Dr.AtanuRakshit, "Cloud Security Issues”, in

Proceedings IEEE International Conference on Services Computing, September 2009.

[4] G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, “Scalable and Efficient Provable Data Possession,” in Proceedings of SecureComm ’2008.

[5] M. A. Shah, M. Baker, J. C. Mogul, and R. Swaminathan, “Auditing to keep online storage services honest,” in Proceedings of the 11th USENIX workshop on Hot topics in operating systems, 2007.

[6] Berkeley, CA, USA, 2007, pp. 1–6. C. Erway, A. K¨up¸c¨u, C. Papamanthou, and R. Tamassia. Dynamic provable data possession in Proceedings of the 16th ACM conference on Computer and communications security, CCS ’09, New York, NY, USA, 2009.