Cyber Times International Journal of Technology & Management

Vol. 11 Issue 2, April 2018 – September 2018

TABLE OF CONTENTS

IMPORTANCE OF HASH VALUE IN SECURITY OF CLOUD STORAGE

Swati D. Ghule
Research Scholar, Tilak Maharshtra Vidyapeeth, Pune
Asst. Prof. P.E. S. Modern College of Engineering, Pune
gadeswati@rediffmail.com

Dr. Anup Girdhar
CEO-Founder
Sedulity Solutions & Technologies
anup@sedulitygroups.com

ABSTRACT

Cloud computing is an emerging technology that can allow user's large amount of data to store in cloud and it can be access from anywhere. There are two important aspects of cloud storage: Data integrity and Data compression. Maintaining data integrity is one of the major challenges in cloud computing because the user has no control over the security mechanisms that are used to protect the data. For assuring data integrity use of hash functions increases, which is also called as message digests or one-way encryption, have no key [1].

A cryptographic hash function compresses arbitrarily long messages to digests of a short and fixed length. Most of existing hash functions are designed to evaluate a compression function with a finite domain in a mode of operation, and the compression function itself is often designed from block ciphers or permutations. This modular design approach allows for a rigorous security analysis with the help of both cryptanalysis and provable security [2].

KEYWORDS: *Cloud Storage, Compression, Cryptography, Data security, Hash Functions.*

I. INTRODUCTION

Popularity of cloud computing is increasing day by day in distributed computing environment. There is a growing trend of using cloud environments for storage and data processing needs. Cloud computing is also called as an Internet-based computing, which makes resources, software, and information available for sharing with computers and other devices on pay per use. However, going for a cloud computing paradigm may have positive as well as negative effects on the data security of service consumers [5].

Data storage is a very crucial and valuable research field in cloud computing. The primary issue that has to be dealt with when talking about data security in a cloud is protection of the data. The idea is to construct a security model for preserving data in cloud storage where data sharing services can control the access and limit the usage of their shared data [5].

With rapid development of cloud computing, number of enterprises will outsource their sensitive data for sharing in a cloud. To keep the shared data legal against untrusted cloud service providers, a natural way is to store only the encrypted data in a cloud. The key problems of this approach include establishing access control for the encrypted data, and revoking the access rights from users when they are no longer authorized to access the encrypted data [5].

In recent years, with the cloud computing and Software as a Service (SaaS) the rise of cloud storage into information storage area of a hotspot. Compared with traditional storage, cloud storage is not just hardware, but network equipment, storage equipment, servers, applications, public access interface, the access network and the client program and other parts of the system. Cloud storage provides storage services, through the network data stored in local storage service provider (SSP) to provide online storage space. Need to store the service users no longer need to build their own data centers, storage services only apply to the SSP, thus avoiding the duplication storage platform, saving the expensive hardware and software infrastructure investments[4].

Maintaining data integrity is one of the major challenges in cloud computing because the user has no control over the security mechanisms that are used to protect the data. It is one of the important fundamental components for securing data.It can protect data in case of loss and damage due to hardware and software failure. Inconsistency and inaccuracy of data can occur by malicious attacks. The term data integrity refers to state, process and function. It applies on the field of data quality. It provides an assurance to the user that the data is not modified or corrupted by the cloud service provider or other users. The data will be stored in the cloud by a user and the integrity of the data will be checked by Auditing mechanism [1].

Data integrity performance can be measured based on the error detection rate. Apart from error detection rate, some data integrity mechanism also use for correcting data. In the field of data security, data integrity is an essential component. Now a day's security is a major concern in cloud computing. So for that reason data integrity mechanism needs to be improved. Data integrity requires because of growing demand of cloud in recent technological trends. The efficiency of data integrity is measured using the parameters like time for processing the data, storage cost, and memory for storage. For assuring data integrity which includes Data encryption, which encrypt and decrypt data by cipher, Data backup, which stores a copy of data in remote location, Access controls, permission to access of data, Data validation, to certify uncorrupted transmission and Using Error detection and correction of data when transmitting data. [1].

With the rapid growth of digital information, the cost of storage infrastructure, management cost and storage space also increases. Therefore it becomes a serious concern to reduce the large amount of data in order to be transferred, stored, and managed in cloud storage systems. People tends to store a lot of files inside their storage system which leads to waste of hardware resources and increases complexity of data center that will near future degrades the performance of cloud storage. When the storage reaches it limit, they then try to reduce those files size to minimum by using data compression methods [1].

Data compression is a method to reduce storage space by eliminating redundancies that occurs in most files. It is not only use for optimizing the limited storage space but also helpful in save time and optimal usage of resources which are insufficient in recent days. There are two types of compression, lossy and lossless. Lossy compression reduced file size by eliminating unwanted data after decoding, this is often used by video and audio compression. On the other hand, Lossless compression, manipulates each bit of data to minimize the size without losing any data after decoding. Data compression can be used for network processing technique to save energy. [1].

The main obstacle to the wide acceptance of cloud storage infrastructure is concern over the data confidentiality and integrity and possibility of access by untrusted users. Only when cloud providers can provide data security guarantees, the customer will be assured of data safety on the cloud from internal and external threats [3].

The idea is that the organizations must encrypt the data preceding storage of their data on a cloud server to guarantee confidentiality. But searching encrypted data is not a simple task. Usually a client may need to retrieve all data from the cloud server to the local device, decrypt it, and only then the search can be performed. Several hash functions can be considered for the generation of encryption keys [3]

II. LITERATURE REVIEW

Cryptographic hash functions have indeed proved to be the workhorses of modern cryptography [3].The use of hash functions has extended to numerous applications, such as password protection, pseudorandom bits and key generation, entropy extraction, etc. Historically, an important reason for the fast hash functions spread has been their efficiency in comparison [2]

Hash functions are normally many-to-one functions since they map arbitrary length inputs to fixed length outputs and the input is usually larger than the output (hash functions are compressing primitives). The output is referred to as the hashcode or the hash value or the message digest. Having a message digest, it is impossible to recover or find the original string. The hash code is a function of all the bits of the message and facilitates an error-detection capability: A change to any bits in the message results in a change to the hash code [4].

A variation on the message authentication code is the one-way hash function. As with the message authentication code, a hash function takes a variable-size message M as input and gives a fixed-size output, referred to as a hash code H (M). The hash code is also referred to as a Message Digest or Hash Value.

Generally, hash functions are classified as keyless or keyed. A keyless hash function takes a Variable length message and produces a fixed length hash value. Keyed hash functions, on the other hand, accept both a variable length message and a fixed length key to produce a fixed length hash value. Keyed hash functions can be further classified based on whether the key is private or public. [3].

When, the hash functions are publicly keyed, they are known as dedicated-key hash functions. Hash functions developed in the public key setting are families of hash functions where individual member functions are indexed by different keys. In this setting, if a member of the hash function family was broken, this should have minimal effect on the other members of the same family (this is not the case in the keyless hash function where a single attack over a function breaks the function entirely,

The drawback of hash functions in the dedicated-key setting, however, is a degraded efficiency as in this case the function is needed to process an extra input (the key) beside the message input. Secretly keyed hash functions are usually used to build Message Authentication Codes (MAC)[3].

III. STANDARD AND IDEAL SECURITY DEFINITIONS

The hash functions needs to fulfill standard security properties that prevent attacks on the application or functionality which makes use of the functions for deploying it. The main standard security properties a hash functions need to preserved, are, collision (col), second preimage (sec) and preimage security (pre)[2].

COLLISION RESISTANCE: A hash collision occurs when two (arbitrary) different messages hash to the same value.

PRE-IMAGE RÉSISTANCE: hash functions should be computationally non-invertible. When a message is hashed, it should be (computationally) not feasible to retrieve the original message from which the hash value was obtained. Pre-image resistance is also sometimes called One-wayness. Generally, collision resistance may not guarantee pre-image resistance, but it was shown that pre-image resistance can be implied by collision resistance if the hash function was sufficiently compressing (i.e. its domain is much larger than its range).

2^{ND} PRE-IMAGE RESISTANCE: 2nd pre-image resistance is also sometimes called Weak Collision Resistance.

A hash function is called secure if the following conditions are satisfied:

It is computationally infeasible to find a message that corresponds to a given hashcode. This is sometimes known as the one-way property of a hash function.

It is computationally infeasible to find two different messages that hash to the same hashcode value. This is also called as the strong collision resistance property of a hash function.

Informally, a hash function is collision secure if it is “hard" to find two randomly chosen distinct input messages mapping to the same output hash value. When one of these input messages is fixed in advance the target security property is second preimage. Preimage security is the “hardness" of finding the preimage message for a fixed output value [2].While hash function ensures data integrity, any change in the original message, however small, must cause a change in the digest, and, for any given file and digest, it must be infeasible for a untrusted users to create a different file with the same digest.

IV. CERTIFICATIONAL SECURITY DEFINITIONS:

In the certificational hash function properties are referred to as the minimal security requirements hash functions have to satisfy. These are more intuitive design principles, which do not necessarily amount to a full-edged attack on the hash function. Certificational security properties may contain: non correlation among input and output bits; strong output propagation of input bit differences known as the avalanche property; no input bits predictability given the output bits or local one-wayness; no left over input bits should be easy to recover given some of input bits or partial preimage resistance; pseudo-, free-start, near-collision attacks; and more.

V. APPLICATION OF HASH FUNCTION IN CRYPTOGRAPHY

Hash function is most imperative device or tool in cryptography, for achieving numerous security and objectives such as digital signatures, data integrity and authentication.

DATA SIGNATURES OR ELECTRONIC SIGNATURES

Historically, digital signatures were the first application of cryptographically secure hash functions. Rabin.(1978) introduced the idea of signing the hash of Cryptographic hash function. In the past, plain text signing a large message directly with a public key cryptosystem.

DATA INTEGRITY can compute hash value of the received data, and compare it with a hash value of original data, then send it through secure channel. If they are the same, then there is high security and they achieve confidentiality. Now a days, the use of MD5, SHA-1, and SHA256 increased. When data has been hashed, changing any of the data will bring about a totally diverse hash value. The primary application of hash functions in cryptography is message integrity.

VI. HASHING ALGORITHMS

Some examples of hashing algorithms are: MD5, SHA-1, SHA-2, and SHA-3.

The Merkle-Damgård construction used in designing hash functions such as MD5 (MD stands for Message Digest), SHA-1 (SHA stands for Secure Hash Algorithm), and SHA-2.There are many cryptography hash functions used to protect the data; the hash length ought to be sufficiently extensive to prevent an assault from find two or more messages that produce the same hash.

MD5 ALGORITHM

MD5 hash functions are mainly used in message integrity check and password shadow.
All people have known MD5 is the most widely used cryptographic hash functions.[6]

SHA-1

SHA-1 is cryptography hash functions are mainly used to provide data integrity and digital signature. SHA-1 was developed by NIST as US federal processing standard. The algorithm supports plain text any of length less than 264bit as input.

SHA-2

SHA-256 is cryptography hash functions are mainly used to provide data integrity and digital signature. SHA-256 was developed by NIST as US federal processing standard. SHA-256 uses a message block size of 512 bits and, as you may expect from its name, produces a message digest of size 256 bits.It converts an input message into the 256 bits message digest. Hence, must be input less than 264 bits and must be operated by 512 bits in groups. The algorithms SHA-256, SHA-384, and SHA-512 are collectively referred to as SHA-2.

CLOUD COMPUTING ATTACKS

There are different types of attacks on cloud computing are noted such as Denial of-Service attack(DoS), Malware-Injection Attack, Side Channel attack, Authentication attack and Man-In-The-Middle Cryptographic Attacks.

VII. DESIGN OF HASH FUNCTION

The design of a secure cryptographic hash function involves many aspects [2]

A hash function is cryptographically secure if it is computationally infeasible to find a collision, which is if it is computationally infeasible to construct meaningful messages whose hashcode would equal a specified value. Additionally, a hash function should be strictly one-way, in the sense that it lets us compute the hash code for a message, but does not let us figure out a message for a given hashcode — even for a very short messages.

Most secure hash functions are based on the structure proposed by Ralph Merkle in 1979. This structure becomes the basis of MD5, Whirlpool and the SHA series of hash functions.

In general, a hash function (keyed or keyless) is built on two components: a compression function f and a construction H. The compression function is a function mapping a larger (but fixed) sized input to a smaller fixed sized output. The output of a compression function to be used as input to the following compression function call.

The construction is the way, how the compression function is repeatedly called to process a message. We are not actually discussing the construction of compression functions because when we design a hash function, one would assume the presence of a “good" compression functions and design a construction accordingly [3]. Once the target security properties are identified, the next step is the hash function design itself.

HASH FUNCTIONS BASED ON BLOCK AND STREAM CIPHERS

Building hash functions based on block ciphers is the most popular and well-known approach. In this type, the compression function is a block-cipher with its two inputs representing a message block and a key.

Although the stream-cipher based approach is not much popular like the block-cipher based approach, in the recent SHA-3 competition, some of the successful second round candidates were based on stream-ciphers (e.g. CubeHash). The main differences between block-cipher-based and stream-cipher-based hash functions are the size of the block and the number of rounds required.

In block-cipher-based, the message blocks are usually large, and iterated a small number of rounds, while in stream-cipher-based, the block size is small, with more rounds. Thus, in block cipher-based, a comparatively good compression function is necessary but in stream-cipher-based, even a weak compression function may provide sufficient security [3].

HASH FUNCTIONS BASED ON PERMUTATIONS

Block cipher based hash functions are distinguish by the fact that the key input to the cipher depends on the input values; this implies that the key schedule has to be powerful and that it needs to be executed for every encryption call, which entails a substantial computational cost. An alternative approach is to fix one or more keys, and restrict the hash function design to use the block cipher for these keys only.

The usage of fixed-key block ciphers, or on the other hand permutations, as well offers the benefit that one does not need to implement an entire block cipher but only a limited number of instantiations of it. This approach was introduced by Preneel et al. Black et al. were the first to formally study these constructions, indicating that a 2n-to-n-bit compression function f using one n-bit permutation p. The development of a hash function H from a permutation p where the compression function (based on p) need not be secure. Throughout, p is assumed to be an ideal permutation [2].

The simplest hash function consists of the first n-bit block, XORing it bit-by-bit with the second n-bit block, XORing the result with the next n-bit block, and likewise. We will call this as the XOR hash algorithm. The hashcode generated by the XOR algorithm can be useful as a data integrity check in the presence of completely random transmission errors.

Remember, that one of the main functional requirements for a hash function is the ability to contain arbitrarily (or at least very) long inputs. To achieve this, most hash functions in the literature use a compression function in a mode of operation [2] Comparatively, compression functions are often easier to design and analyze, and the problem of designing a hash function is shifted to the problem of designing a compression function.

Thus, the next design step lies in the development of a secure compression function [2] Because of their extensive deployment and hence paramount security importance, hash functions should be supported with a powerful security foundation. Such a statement supports the claim that all hash function designs should have and come with a rigorous security analysis [2].

VIII. CONCLUSION

Cloud providers need to maintain the privacy and security of personal data that they hold on behalf of organizations and users. Responsible management of personal data is a central part of creating the trust that underpins adoption of cloud based services – without trust, customers will be reluctant to use cloud-based services. Authentication is necessary in Cloud Computing. The main goal is to securely store and manage data that is not controlled by the owner of the data. The data are stored in cloud environment Cloud security here is solved by providing a Hash Function for data in the cloud [4].

The hash value improves the security of the information systems by permitting an arbitrary identifier to be replaced by a standard fixed-length code. This code is used as an index for referencing related data in response to a user's query. Index creation by the data owner will make search in cloud computing easier and faster.

In addition, this method simplifies the system's structure, increases search efficiency, and reduces the size of data and communication delays.

REFERENCES

[1] Abhijit Choudhury, Santanu Kumar Misra, Bijoyeta Roy, " A brief Survey on Data Integrity and Compression in Cloud Computing", International Journal of Computer Applications (0975 – 8887), Volume 156 – No 13, December 2016.

[2] Elena Andreeva, Bart Mennink, and Bart Preneel, "Open Problems in Hash Function Security", This work was supported in part by the Research Council KU Leuven: GOA TENSE (GOA/11/007).

[3] Hasan Omar Al-Sakran, "ACCESSING SECURED DATA IN CLOUD COMPUTING ENVIRONMENT", International Journal of Network Security & Its Applications (IJNSA) Vol.7, No.1, January 2015.

[4] Paridhi Singhal, Alok Garg, Manoj Diwakar, " Security in Cloud Computing- Hash Function",*International Journal of Computer Applications (0975 – 8887) Volume 68– No.14, April 2013.*

[5] Ranjita Mishra, Sanjit Kumar Dash, Debi Prasad Mishra, Animesh Tripathy, "A Privacy Preserving Repository for Securing Data across the Cloud", 978-1-4244-8679-3/11/$26.00 ©2011 IEEE.

[6] Saif Al-Kuwari, James H. Davenport, Russell J. Bradford, "Cryptographic Hash Functions: Recent Design Trends and Security Notions", eprint.iacr.org/2011/565.

[7] Simon Waddington, Jun Zhang, Gareth Knight, Jens Jensen, Roger Downing and Cheney Ketley, "Cloud repositories for research data – addressing the needs of researchers", Journal of Cloud Computing: Advances, Systems and Applications Advances, Systems and applications 2013**2**:13 https://doi.org/10.1186/2192-113X-2-13 [Accessed 2 March 2018].