**Aptisi Transactions on Technopreneurship (ATT)**
**Vol. 3 No. 2 September 2021**

**P-ISSN: 2655-8807**
**E-ISSN: 2656-8888**

# Hash Algorithm In Verification Of Certificate Data Integrity And Security

**Muhammad Rehan Anwar[1], Desy Apriani[2], Irsa Rizkita Adianita[3]**
[1]University of Agriculture Faisalabad (UAF), Computer Science, Pakistan
[2,3]University of Raharja, Indonesia
e-mail: rehan749@gmail.com[1], desy@raharja.info[2], irsa.rizkita@raharja.info[3]

Anwar, M. R. ., Apriani, D. ., & Adianita, I. R. (2021). Hash Algorithm In Verification Of Certificate Data Integrity And Security. *Aptisi Transactions on Technopreneurship (ATT)*, *3*(2), 67–74.

**DOI:** https://doi.org/10.34306/att.v3i2.212

**Author Notification**
22 August 2021
**Final Revised**
07 September 2021
**Published**
20 September 2021

***Abstract***

*The hash function is the most important cryptographic primitive function and is an integral part of the blockchain data structure. Hashes are often used in cryptographic protocols, information security applications such as Digital Signatures and message authentication codes (MACs). In the current development of certificate data security, there are 2 (two) types of hashes that are widely applied, namely, MD and SHA. However, when it comes to efficiency, in this study the hash type SHA-256 is used because it can be calculated faster with a better level of security. In the hypothesis, the Merkle-Damgård construction method is also proposed to support data integrity verification. Moreover, a cryptographic hash function is a one-way function that converts input data of arbitrary length and produces output of a fixed length so that it can be used to securely authenticate users without storing passwords locally. Since basically, cryptographic hash functions have many different uses in various situations, this research resulted in the use of hash algorithms in verifying the integrity and authenticity of certificate information.*

***Keywords:*** *Blockchain, Hash Algorithm, SHA-256, Certificate Data Integrity.*

## 1. Introduction

Authenticated encryption scheme is an important issue of certificate data security. It is necessary to ensure that data is sent to the specified recipient securely over the network. In general, such transactions should achieve good confidentiality and authenticity [1]. To overcome this convenience problem, a hash algorithm in the blockchain is used, where the digital platform allows the only valid and verified transactions. Hash is designed to find duplicate nodes on certificates issued in the same ledger and can achieve trusted transactions in P2P networks over the internet [2]. The hash algorithm is the most important cryptographic primitive function, an integral part of the blockchain data structure. Hashes are often used in cryptographic protocols, information security applications such as Digital Signatures and message authentication codes (MACs) [3]. A cryptographic hash function is a one-way function that converts input data to an arbitrary length and produces an output with a fixed length [4][5]. The output is usually referred to as a "hash value" which is presented in figure 1.

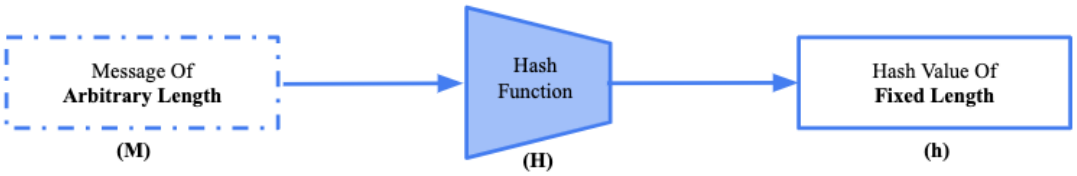


Figure 1. The basic form of a hash function

For a hash function to fulfill its purpose and be usable, a hash function needs to fulfill 5 (five) core properties such as:

1. Input can be a string of various sizes , but each output has a fixed length, say a 256-bit output or a 512-bit output.
2. The hash value must be efficiently computed for any given data.
3. Deterministic, in the sense that the same input when applied to a hash algorithm will return the same hash.
4. It is not possible to return and generate data from a hash value.
5. Every small change in the data must greatly affect the output hash, so that no one can correlate the new hash value with the old hash after the change.

Apart from the core properties, the hash must also satisfy the following 3 (three) security properties:

1. Collision resistance: Implies that it is impossible to find two different inputs, say, X and Y, with the same hash value [6]. Makes the hash function H() collision-resistant because neither can find X and Y, so H(X) = H(Y). Most online stores, such as the App Store, use this property to ensure file integrity.
2. Preimage Resistance: Where this property means that it is computationally impossible to reverse the hash function.
3. Second Preimage Resistance: This property indicates that if given the input X and returns a hash H(X), it is impossible to find Y, so H(X) = H(Y). It can be said that if the hash function is already collision resistant, then it is also resistant to the second preimage [7].

It can be noted that for the output of an n-bit hash value, an average effort of 2n is required to break the second preimage and preimage resistance, and 2n/2 for the collision resistance [8]. From the properties that the hash function needs to fulfill, it is clear that in verifying the integrity and security of certificate data, hash algorithms are indispensable and trustworthy.

## 2. Research Methods

In the current development of certificate data security, there are 2 (two) types of hashes that are widely applied, namely, MD and SHA. However, when it comes to efficiency, in this study the hash type SHA-256 is used because it can be calculated faster with a better level of security. SHA-256 belongs to the SHA-2 hash function family, which is used by Bitcoin and produces a 256-bit hash value [9]. The construction method is needed to avoid data collisions, where the construction method used in SHA-256 is the Merkle-Damgård construction as shown in Figure 2 [10].

P-ISSN: 2655-8807
E-ISSN: 2656-8888

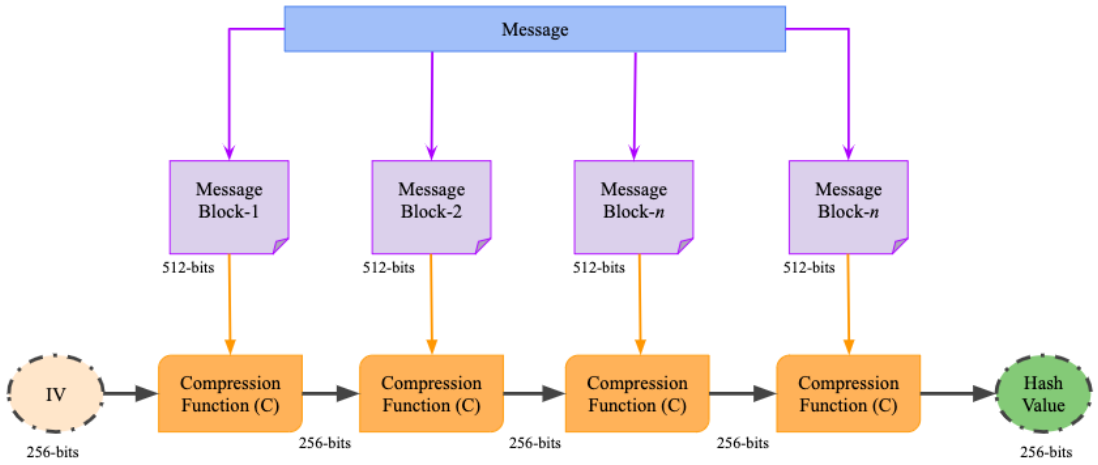


Figure 2. Merkle-Damgård Construction Method

Figure 2. Describes how a message is first divided into 512-bit blocks. When the non-last message is not an exact multiple of 512 bits, it will be filled with one 1 bit to reach a 512 bit block [11]. The 512 bit block is further divided into 16 32 bit word blocks (16 × 32 = 512). Each block goes through 64 rounds, where each 32-bit word goes through a series of combination operations of some common functions such as XOR, AND, OR, NOT [12].

Agile methods were also applied to demonstrate the significant benefits of simple, iterative and agile techniques for planning and controlling innovative product projects combined with traditional project management best practices [13]. This method is very supportive to verify the data integrity of the certificate using a hash algorithm.

## 3. Literature Review

Hash algorithms include features, such as one-way, deterministic cryptography, faster computation, avalanche effects, and must be impact-resistant. Judging from the many features available, 10 (ten) literature reviews on hash algorithms were carried out, as follows:

Access to a patient's medical history is essential for prescribing drugs correctly, with blockchain can dramatically improve the healthcare framework. In the research conducted by Sudeep Tanwar, several solutions were explored to increase the boundaries in the health care system using blockchain technology [14], but in this study we did not focus on hash algorithm research on educational institutions.

Blockchain technology in the field of power systems seems to have been widely proposed through research conducted by Maria Luisa Di Silvestre, where the relationship with physical assets makes blockchain applications more complex but also more reliable and associated with measurable benefits. The application of blockchain technology in the field of power systems, clarifies some technical aspects of the promising technology, features and applications developed, and focuses on the future of innovative applications in electrical energy [15]. This research can be used as a reference for the application of blockchain technology through hash algorithms in the world of education.

Proposed a new hashing model in reinforcement with redundancy elimination, can fully utilize large-scale similarity information and eliminate redundant hash bits by deep gain learning, but currently the system under study is more about redundancy elimination for effective image capture [16]. Performing comparisons on the similarity of hashing algorithms manually is a complex task and requires a lot of time, therefore it needs to be further integrated to identify strengths and weaknesses [17].

As today's technology evolves, it has been found that many CMS frameworks and web applications use outdated hash functions, an arbitrary number of hash iterations. Notably, popular WordPress still uses MD5 with low hash iterations [18]. So in verifying the data security integrity of this certificate, SHA-256 hashing is applied which is more qualified.

The ideal image hashing method for authentication should provide the characteristics of uniqueness, compactness, perceptibility, robustness and uncertainty. Therefore, the integrity verification of the certificate data security that is currently under review needs to have strengths and limitations that depend on hashing algorithms [19]. The combination of color features and structure features also demonstrates strong hashing performance for discovering normal content manipulation, as well as finding tampered areas. This is confirmed by an experiment conducted by Shen Qi in 2019 regarding perceptual hashing for color images [20].

In addition to data integration verification, distribution and storage of certificate lists is also a major challenge due to the estimated high costs of distribution and storage in the network. Motivated by the need to maintain effective and scalable distribution and storage costs, a distributed management scheme was implemented by utilizing hashes that have been widely used in peer-to-peer (P2P) networks [21]. Research conducted by Anitta Patience Nama shows that hashes are effective in matching similar files, which have slight differences in content, thereby enabling optimal exchange and enabling the system to provide information that helps protect the system with an accuracy of at least 92% [22]. The main strength proposed in this study is that it does not require a trusted central authority to operate because it uses blockchain

a decentralized architecture, maintaining user privacy as a commitment to certificate data before being published and added to the public ledger [23].

## 4. Results and Discussion

As described in Figure 3, the first thing to do is to perform a hash algorithm function on the certificate, then sign it, because it is safer than signing first and then hashing. Ideally, authenticity verification would be performed before performing any other operations. Once the certificate has been signed, the signed hash is marked and sent to the recipient. The recipient can then check for authenticity and find the appropriate hash. Then check the received hash again, if there is a difference.

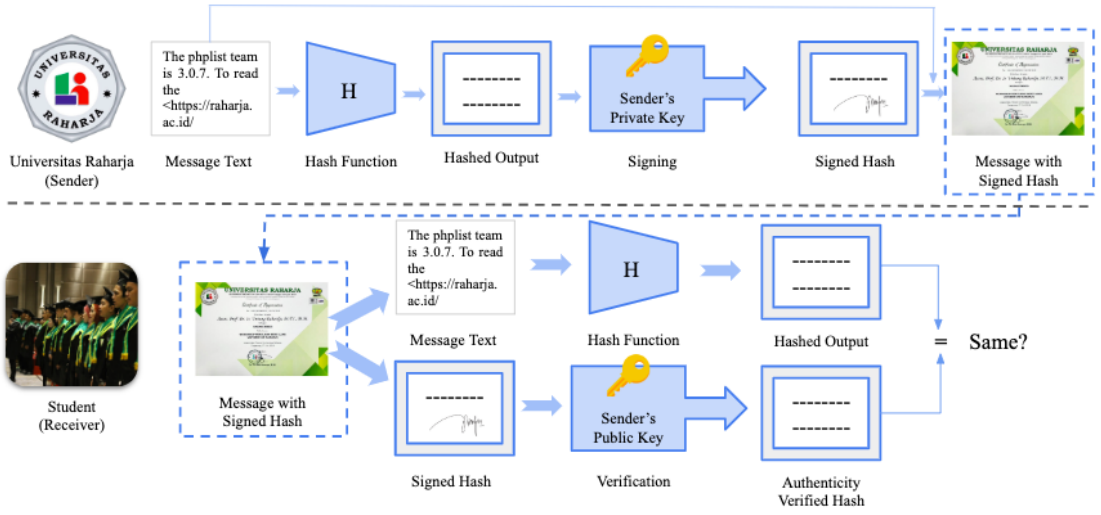


**Figure 3.** Digital Signature Algorithm (DSA)

In this way, the Digital Signature Algorithm (DSA) provides the following security properties:

1. Authenticity: Signed by private key and verified by public key
2. Data integrity: Hashes will not match if the data is changed.

3. Non-repudiation: Since the sender signed the certificate, it cannot deny that it did not send the message. Non-repudiation is the most desirable property where there is a possibility of data exchange disputes.

Resembling a hash pointer data structure, Merkle trees are also tamper-proof. Breaking at any level in the tree is the same as causing a mismatch of hashes that have been stored at one level in the hierarchy. It is very difficult for hackers to change all the hashes in the entire tree and the integrity of any certificate transaction sequence can be ensured.

The merkle tree provides a very efficient way to verify the specific transactions belonging to a particular block. If there are "n" transactions in the Merkle tree, then this verification takes only Log(n) time as shown in Figure 4.

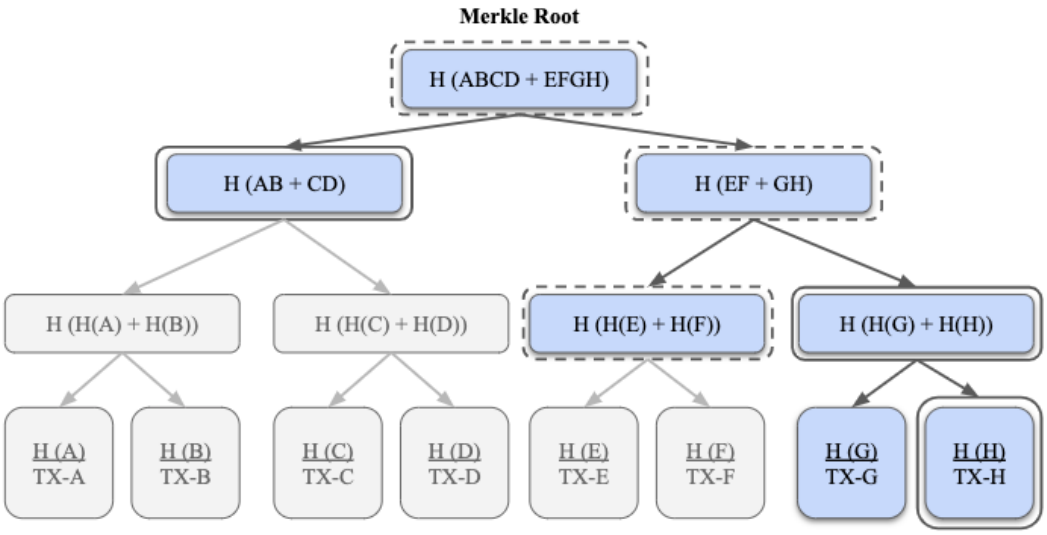


**Figure 4**. Verification on the Merkle tree

To verify whether a transaction or other certificate belongs to the Merkle tree, it is not necessary to check all items and the entire tree. It can be started by calculating the hashes of two concurrent transactions, seeing if they match the parent hash. Then proceed with verifying the parent and sibling hashes at that level to get another parent hash. Continuing this process all the way to the top root hash is the fastest possible way for transaction verification (only Log(n) time for n items). Since there are eight transaction elements (n = 8), only three calculations (log2 8 = 3) will be required for verification.

**5. Listing Program**

The following is an example of code for different hash functions. This section is only intended to provide information on how to use hash functions programmatically. Sample code uses Python but will be very similar in different languages.

```
# -*- coding: utf-8 -*-
import hashlib
# hashlib module is a popular module to do hashing in python

#Constructors of md5(), sha1(), sha224(), sha256(), sha384(),
and sha512() present in hashlib
```

```
md=hashlib.md5()
md.update("The quick brown fox jumps over the lazy dog")
print md.digest()
print "Digest Size:", md.digest_size, "\n" , "Block Size: ",
Md.block_size

# Comparing digest of SHA224, SHA256,SHA384,SHA512
print "Digest SHA224", hashlib.sha224("The quick brown fox
jumps over the lazy dog").hexdigest()
print "Digest SHA256", hashlib.sha256("The quick brown fox
jumps over the lazy dog").hexdigest()
print "Digest SHA384", hashlib.sha384("The quick brown fox
jumps over the lazy dog").hexdigest()
print "Digest SHA512", hashlib.sha512("The quick brown fox
jumps over the lazy dog").hexdigest()
# All hash outputs are unique

# RIPEMD160 160 bit hashing example
h = hashlib.new('ripemd160')
h.update( "The quick brown fox jumps over the lazy dog")
h.hexdigest()

#Key derivation Algorithm:
#Native hashing algorithms are not resistant against brutefore
attack.
#Key deviation algorithms are used for securing password
hashing.
import hashlib, binascii
algorithm='sha256'
password='HomeWifi'
salt='salt' # salt is random data that can be used as an
additional input to a one-way function
nu_rounds=1000
key_length=64 #dklen is the length of the derived key
dk = hashlib.pbkdf2_hmac(algorithm,password, salt, nu_rounds,
dklen=key_length)
print 'derived key: ',dk
print 'derived key in hexadecimal :', binascii.hexlify(dk)

# Check properties for hash
import hashlib

input = "Sample Input Text"
for i in xrange(20):
      # add the iterator to the end of the text
      input_text = input + str(i)
      # show the input and hash result
      print input_text, ':', hashlib. sha256(input_text).
      hexdigest()
```

## 5. Conclusion

Since basically, cryptographic hash functions have many different uses in various situations, this study yields 6 (six) conclusions:

1. The hash function is used to verify the integrity and authenticity of certificate information. The blockchain data structure, using cryptographic hashes, and the use of Merkle trees make transaction verification easier and faster.
2. Hash functions can also be used to index data in a hash table. This can speed up the search process. Instead of the whole data, if we search by hash (assuming the hash value is much shorter compared to the whole data), then it should obviously be faster.
3. Hashes can be used to securely authenticate users without storing passwords locally. If the hacker hacks into the server, then it cannot get the password from the stored hash. Whenever a user tries to log in, the hash of the entered password is calculated and matched against the stored hash.
4. This study uses a hash function as a proof of work (PoW) algorithm to improve the security and privacy of the certificate.
5. Cryptographic functions are one-way and cannot be reversed. They are deterministic and produce the same output for a given input. However, any changes to the input will produce a completely different output when the final result is displayed.
6. The use of blockchain for verification purposes is in its early stages, and there are many research possibilities on hash algorithms that need to be considered in the future

Decentralized solutions where transactions are public are likely to have different types of attacks. Attempts at counterfeiting are the most obvious of all, especially when conducting transactions of value such as certificate verification. Hash algorithms can be used to ensure the system is resistant to certificate data falsification. If the transaction and signed hash has been done, no one can change the transaction, nor can anyone deny that the transaction has been made.

## Acknowledgment

This research was supported by University of Raharja and supported by Ristekdikti in the simlitabmas grant research project.

## References

[1] Tsai, JL (2009). Convertible multi-authenticated encryption scheme with one-way hash function. Computer Communications, 32(5), 783-786.
[2] Scholar, PG College Fees Transaction Using Hash Functions of Blockchain Model.
[3] Bos, JW, Halderman, JA, Heninger, N., Moore, J., Naehrig, M., & Wustrow, E. (2014, March). Elliptic curve cryptography in practice. In International Conference on Financial Cryptography and Data Security (pp. 157-175). Springer, Berlin, Heidelberg.
[4] Goyal, V., O'Neill, A., & Rao, V. (2011, March). Correlated-input secure hash functions. In Theory of Cryptography Conference (pp. 182-200). Springer, Berlin, Heidelberg.
[5] Lefebvre, F., Czyz, J., & Macq, B. (2003, September). A robust soft hash algorithm for digital image signatures. In Proceedings 2003 International Conference on Image Processing (Cat. No. 03CH37429) (Vol. 2, pp. II-495). IEEE.
[6] Steinberger, J. (2010, May). Stam's collision resistance conjecture. In Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 597-615). Springer, Berlin, Heidelberg.
[7] Contini, S., & Yin, YL (2006, December). Forgery and partial key-recovery attacks on HMAC and NMAC using hash collisions. In International Conference on the Theory and

Application of Cryptology and Information Security (pp. 37-53). Springer, Berlin, Heidelberg.
[8] Sobti, R., & Geetha, G. (2012). Cryptographic hash functions: a review. International Journal of Computer Science Issues (IJCSI), 9(2), 461.
[9] Mukhopadhyay, U., Skjellum, A., Hambolu, O., Oakley, J., Yu, L., & Brooks, R. (2016, December). A brief survey of cryptocurrency systems. In 2016 14th annual conference on privacy, security and trust (PST) (pp. 745-752). IEEE.
[10] Backes, M., Barthe, G., Berg, M., Grégoire, B., Kunz, C., Skoruppa, M., & Béguelin, SZ (2012, June). Verified security of merkle-damgård. In 2012 IEEE 25th Computer Security Foundations Symposium (pp. 354-368). IEEE.
[11] Sharief, SM (2013). Secure Hash Design & Implementation Based On Md5 & Sha-1 Using Merkle–Damgard Construction.
[12] Sharmila, S., & Umamaheswari, G. (2011, July). Detection of sinkhole attacks in wireless sensor networks using message digest algorithms. In 2011 International Conference on Process Automation, Control and Computing (pp. 1-6). IEEE.
[13] Conforto, EC, & Amaral, DC (2010). Evaluating an agile method for planning and controlling innovative projects. Project Management Journal, 41(2), 73-80.
[14] Tanwar, S., Parekh, K., & Evans, R. (2020). Blockchain-based electronic healthcare record system for healthcare 4.0 applications. Journal of Information Security and Applications, 50, 102407.
[15] Di Silvestre, ML, Gallo, P., Guerrero, JM, Musca, R., Sanseverino, ER, Sciumè, G., ... & Zizzo, G. (2019). Blockchain for power systems: Current trends and future applications. Renewable and Sustainable Energy Reviews, 109585..
[16] Yang, J., Zhang, Y., Feng, R., Zhang, T., & Fan, W. (2020). Deep reinforcement hashing with redundancy elimination for effective image retrieval. Pattern Recognition, 100, 107116.
[17] Breitinger, F., Stivaktakis, G., & Baier, H. (2013). FRASH: A framework to test algorithms of similarity hashing. Digital Investigation, 10, S50-S58.
[18] Ntantogian, C., Malliaros, S., & Xenakis, C. (2019). Evaluation of password hashing schemes in open source web platforms. Computers & Security, 84, 206-224.
[19] Du, L., Ho, AT, & Cong, R. (2020). Perceptual hashing for image authentication: A survey. Signal Processing: Image Communication, 81, 115713.
[20] Shen, Q., & Zhao, Y. (2020). Perceptual hashing for color image based on color opponent component and quadtree structure. Signal Processing, 166, 107244.
[21] Cebe, M., & Akkaya, K. (2019). Efficient certificate revocation management schemes for IoT-based advanced metering infrastructures in smart cities. Ad Hoc Networks, 92, 101801.
[22] His name is, AP, Cloud, IU, Disso, JP, & Younas, M. (2019). Similarity hash based scoring of portable executable files for efficient malware detection in IoT. Future Generation Computer Systems.
[23] Nosouhi, MR, Yu, S., Zhou, W., Grobler, M., & Keshtiar, H. (2020). Blockchain for secure location verification. Journal of Parallel and Distributed Computing, 136, 40-51.