SSRG International Journal of Electrical and Electronics Engineering
ISSN: 2348-8379/ https://doi.org/10.14445/23488379/IJEEE-V13I11P101

Volume 13 Issue 11, 1-13, November 2026
© 2026 Seventh Sense Research Group®

Original Article

Performance Benchmarking and Security Evaluation of HAVAL-128, MD5, and MMD5 Cryptographic Hash Algorithms for Protecting Data Integrity in Modern Systems

Shikha Srivastava[1], Md Afaque Alam[2], Mayank Srivastava[3]

[1,2]*Department of Mathematics, Aligarh Muslim University, Aligarh, Uttar Pradesh, India.*
[3]*Department of Computer Engineering and Applications, GLA University, Mathura, Uttar Pradesh, India.*

[3]*Corresponding Author: mayank.srivastava3103@gmail.com*

Received: 10 July 2026 Revised: 20 October 2026 Accepted: 30 October 2026 Published: 25 November 2026

Abstract *- Hash functions are a key ingredient of modern cryptography for secure digital communication. Hash algorithms are widely used for data authentication, integrity checks, password security, key generation, and digital forensics. The effectiveness and security of existing hash functions, despite their widespread use, remain active areas of research. This article proposes an enhanced padding technique based on a Modified MD5 hashing procedure (MMD5) to improve the overall hashing process. To evaluate its cryptographic strength, analysis based on the avalanche effect and statistical tests, including the Discrete Fourier Transform test, Binary Matrix Rank test, Approximate Entropy test, and Chi-Square test, was performed. Experimental results show that MMD5 achieves stronger diffusion than standard MD5 and HAVAL-128, while maintaining a balanced distribution of 0s and 1s, representing strong randomness in the generated hash outputs.*

Keywords *- Binary Matrix Rank test, Chi-Square test, Discrete Fourier Transform test, Hash function, Message integrity.*

1. Introduction

Among the most significant subclasses of cryptographic algorithms are cryptographic hash functions. A hash function is a mechanism that converts an input plaintext or message into a fixed-length binary string, said to be the message digest, hash value, or checksum [1]. These functions are fundamental in modern cryptography and are commonly used in digital signatures, data integrity, password protection, and blockchain systems [2-4].

Formally, a cryptographic hash function is defined as: $H: \{0, 1\}^* \rightarrow \{0, 1\}^n$, where $\{0,1\}^*$ denotes bit strings of any length, and $\{0,1\}^n$ desnotes a fixed finite-length bit hash of n bits. H is the hash function; it maps a binary input value ′x′ of any size into a compressed fixed-size value of 'h', i.e., h = H(x).

Cryptographic hash functions are generally divided into two segments: Keyed hash functions and Unkeyed hash functions [5].

The hash functions based on an input message and a secret key are known as keyed hash functions, whereas the hash functions which are based on just one parameter, an input message, are generally referred to as unkeyed hash functions.

The widely used family of hash functions, commonly referred to as the MD family, originated with early designs proposed by Ronald Rivest in the early 1990s [6, 7]. The MD4 algorithm, introduced in 1990, was designed to produce a 128-bit message digest with high computational efficiency.

This regulatory advantage, combined with its strong performance, led to significant interest in MD4 within both academic and industrial communities. However, subsequent cryptanalysis revealed notable structural weaknesses in MD4, culminating in the demonstration of highly efficient collision attacks in 1996, followed by further developments showing vulnerabilities to preimage attacks.

Although MD4 was never widely adopted and was not recommended for use after the early 1990s, several hash functions were influenced by its structure and design.

The most notable successor of MD4 is MD5, created by Ronald Rivest, which modified the structural design while

This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/)

enhancing security and computational efficiency. MD5 introduced multiple key structural refinements, which are given as follows:

- Added a fourth round, which increases the total number of operations from 48(MD4) to 64(MD5).
- Enhanced Boolean functions were added in specific rounds.
- Minimize configurational symmetry by altering the message word access order.
- Different additive constants were added: $T_i = [2^{32}|\sin(i)|]$, for $i = 1, 2,\ldots 64$
- Furthermore, enhanced diffusion using additional mixing and a rotational process.

The compression function is $Hi = f(H_{i-1}, M_i)$. Enhancing the internal function f was the main goal of MD5's improvement so that:

- Unpredictable propagation would result from a minor change in input.
- Regulating the differential path would become more difficult.
- The theoretical birthday bound would tend to the effective collision resistance: $2^{128/2} = 2^{64}$.

Therefore, MD5 offers a transformative improvement over MD4, with a small decrease in speed in exchange for a significantly enhanced security margin through extended multi-round operations and improved nonlinear configuration. Similar to its predecessor, MD5 produces a 128-bit hash, which limits its resistance to collisions due to the birthday paradox. Collisions may be possible with less computational work than expected. Although this drawback was not instantly applicable when it was first implemented, the research community was concerned about its long-term security. As a result, even though it is widely deployed across several applications, the need for hash functions with stronger security assurances prompted the research community to seek more reliable substitutes in later years.

We proposed an improved version of MD5 in this article, which consists of several rounds of input message modification and the Trailing Bit Complement (TRC) padding scheme. This framework enhances defence against differential and collision attacks and improves efficiency by enhancing nonlinearity and bit diffusion.

The research article is summarized below: Section 2 reviews the most recent related work on cryptographic hash functions with emphasis on MD5. Section 3 describes the properties of cryptographic hash functions, and Section 4 covers the construction and design of the proposed algorithm (MMD5). The results and discussion given in Section 5 present a comparison of the avalanche effect of the HAVAL-128, MD5 and MMD5 hashing algorithms, and a statistical analysis of MMD5. Finally, the conclusion is presented in Section 6.

2. Related Works

As the demand for reliable and efficient data authentication grows, many studies have examined the strengths and weaknesses of both existing and newly proposed hash algorithms.

The attacks on the MD5 hash algorithm using precomputed hash chains and the rainbow table technique are presented by Zyuzin et al. [8]. It describes the working principles of MD5, the generation of hash chains, and shows how to break the hash function using hash chains. It also introduces rainbow tables and discusses their impact on the efficiency of such attacks. An attack on a hash function is an attempt to reverse the hashing process and get back the original message. Nkouankou et al. [9] review existing methods for carrying out preimage attacks on MD5 by means of SAT solvers. It targets improvements of the encoding techniques as well as of the solving process.

Obaida et al. [10] propose a new keyed hash function that combines a hash function and the Henon map. The algorithm generates a fixed 128-bit hash value for inputs of any finite length. It ensures data security, integrity, and authentication in digital communications by using a secret key to prevent unauthorized entities from generating valid hash values. The article summarizes the structure, procedure, security features, and application aspects of the proposed algorithm.

During data transmission, integrity and authenticity are crucial for assuring the accuracy, stability, and reliability of information and for safeguarding it. In this regard, the hash functions SHA-1, MD5, and MD2 were analyzed by Al-Ghanimi et al. [11], which is based on factors such as execution time and bit variation.

Despite its limitations, MD5 is still largely employed for password protection in web login systems. Nugroho et al. [12] aim to improve security by combining MD5 and salted hashing. The research analyzes efficiency by incorporating salt values with MD5 using a quantitative experimental method for password protection. Boyanov [13] analyzes the role of cryptographic hashing algorithms for assuring the integrity and authenticity of files. It computes their practical use across 6 popular tools: CrypTool 1.4.40, MD5 Calculator, PowerShell, 7-Zip, BullZip, and HashCalc. The results offer an important insight into the role of hash functions play in preserving the dependability and authenticity of digital resources. The analysis outlines the advantages and disadvantages of each tool for producing and validating hash outputs while ensuring data security.

Kurunthachalam [14] addressed the inadequacy of conventional password systems in the face of modern cloud threats by proposing an MD5-ElGamal encryption model. Evaluated against traditional techniques for speed, memory, and security, the approach balances the fast execution of MD5 hashing with the robust, albeit computationally heavier, security of ElGamal asymmetric encryption. The findings confirm that this integrated framework offers a practical, high-integrity security solution for mobile cloud environments.

AlShuaibi et al. [15] proposed a block-based hashing scheme that integrates Hidden Markov Models with Genetic Algorithms (GA), leveraging evolutionary operations such as selection, crossover, and mutation to enhance adaptive security. To resist pre-image and collision attacks, the method incorporates a single-key Hill cipher. Experimental results indicate that this approach outperforms existing algorithms in both accuracy and overall effectiveness.

Gilbert et al. [16] investigated the security and efficiency of SHA-family hash functions for data integrity, highlighting vulnerabilities in conventional PoR schemes against modern exploits such as the Juels-Kaliski and RSA-SHA variants. Evaluating SHA-1, SHA-256, SHA-512, and SHA-3 through simulations, FPGA validation, and empirical testing, the authors found that SHA-3 is ideal for resource-constrained systems, while SHA-256 delivers the best overall balance of performance and security.

Amid the growing challenge for law enforcement to counter evolving cybercrime, Al-Khazalli et al. [17] examine the role of MD5 in digital forensics, highlighting its utility for evidence retrieval and device analysis. The study assesses the strengths and vulnerabilities of MD5, underscoring its pivotal yet nuanced function across forensic investigations.

Nurdin et al. [18] offer a detailed comparison of the performance and security of MD5 and SHA-256, along with well-supported recommendations for protecting the integrity of biometric data. As per the author, SHA-256, in contrast to MD5, provides a higher level of security but may be more computationally expensive.

Wang et al. [19] presented C implementations of the MD5 and SHA-1 hashing algorithms, providing an in-depth analysis of their underlying operations and workflows. Through software-based comparative testing, the authors demonstrated the structural differences, computational efficiency, and security profiles of both schemes.

Stevens et al. [20] demonstrated the first practical collision attack on the full SHA-1 hash function using advanced cryptanalytic techniques, shattering its assumed collision resistance. Their findings provided empirical evidence of SHA-1's vulnerabilities, reinforcing the urgent need to transition to more robust cryptographic alternatives. Recent studies on hash algorithms show several key challenges and research gaps. Numerous studies emphasize various cryptographic attacks against well-known hash families such as SHA, MD, and HAVAL; however, they don't thoroughly examine important properties such as the avalanche effect, randomness, and execution time. Some modified algorithm designs use a keyed hash function with Henon maps to increase the output's complexity. However, statistical analysis of the generated hash, like randomness and uniformity, is often missing from these approaches. The adoption of novel hash designs in realistic and safe systems is challenging due to the lack of standardized computation.

3. Properties of Cryptographic Hash Functions

A cryptographic hash function is defined as $H: \{0, 1\}^* \rightarrow \{0, 1\}^n$, where $\{0,1\}^*$ represents the collection of all bit sequences of any finite size and n is a fixed natural number. The function generates a fixed-length output of n bits from an input of any arbitrary length. If a hash function satisfies the following properties, it is considered secure.

Pre-image Resistance: A hash function is called preimage resistant if, for a given hash value $z \in \{0,1\}^n$, it is infeasible to obtain any input $t \in \{0,1\}^*$ such that $H(t) = z$. Although such a t may exist mathematically, finding it should require an impractically large amount of computation. This computation guarantees that an attacker can't get the original message, even if they know the message digest.

Second Pre-image Resistance: A hash function satisfies second pre-image resistance if, for a given input $t \in \{0,1\}^*$, it is impossible to determine another input $t' \neq t$ such that $H(t') = H(t)$. A one-way hash function guarantees both preimage and second-preimage resistance. This property prevents an adversary from changing an original message into a distinct message that generates the same message digest.

Collision Resistance: A hash function is called collision resistant if it is impossible to obtain any two different inputs t, $t' \in \{0,1\}^*$ such that $t \neq t'$ and $H(t) = H(t')$. A hash function is called a Collision-Resistant Hash Function (CRHF) if it satisfies both collision resistance and second-preimage resistance.

Among these security properties, ensuring collision resistance is generally regarded as the most difficult aspect of constructing a robust hash function [21]. However, a secure hash function ensures that finding a collision requires approximately $2^{n/2}$ operations (the birthday bound), which is infeasible for large n.

Deterministic Property: A hash function is deterministic, meaning that for all $t1, t2 \in \{0,1\}^*$, $t1 = t2 \Rightarrow H(t1) = H(t2)$. The output is always the same for the same input. There is no randomness in a standard cryptographic hash function.

Fixed Output Length: For every input t ∈ {0,1}*, the output length is always fixed at n bits: |H(t)| = n. Regardless of the input size, the hash output remains constant in length.

Avalanche Effect: The phenomenon in which a minor change in the input data causes a significant alteration in the output is called the avalanche effect [22]. If two inputs differ in only one bit, their Hamming distance can be represented as d(t, t′) = 1, then the Hamming distance between their hash outputs should be approximately d(H(t), H(t′)) ≈ n/2. The avalanche effect is strong, which means good diffusion and unpredictability of hash bits.

4. The Proposed Algorithm (MMD5)

This section presents the proposed algorithm, termed Modified Message Digest-5 (MMD5), which incorporates a novel TRC padding scheme and a multi-stage preprocessing transformation to enhance diffusion and input sensitivity.

4.1. TRC Padding Scheme

Let the input message be denoted by M, and its binary representation by a bit string of length L. Let the last bit of M be b_L. Define the padding bit p as:

$$\boldsymbol{p = \begin{cases} 1, & if\ b_L = 0 \\ 0, & if\ b_L = 1 \end{cases}}$$

Append the bit p repeatedly until the total length satisfies: L′ ≡ 448 (mod 512). Append the length bit, which is represented as a 64-bit value.

The resulting padded message is split into 512-bit chunks. Every chunk is represented as: K = K[0], K[1],..., K[63], where each K[i] is an 8-bit block.

4.2. Modified Hashing Process

In the proposed algorithm (MMD5), the message schedule word M[i] is replaced by the generated B[i] within the compression function. The generation of B[i] is performed through a sequence of modification processes, which is described below:

Modification I: Chaining XOR Computation

After getting the 512-bit input block from the TRC padding method, the binary string is split into equal 8-bit sub-blocks as follows: K = K[0], K[1],...,K[63]. The chaining XOR process is described in this way:

$K_0 = \{00000000\} \oplus K[0]$
$K_1 = K_0 \oplus K[1]$
$K_2 = K_1 \oplus K[2]$
$K_3 = K_2 \oplus K[3]$
$K_4 = K_3 \oplus K[4]$
$K_{63} = K_{62} \oplus K[63]$

This formulation introduces a sequential dependency in which each intermediate value incorporates the contributions of the previous state and the current message byte.

Modification II: Sequence-based Transformation

Let the integer value of the final output of Modification I be denoted as T. A sequence S is defined as follows:

$$S_i = \begin{cases} T + i, & if\ T + i \le 255, \\ T + i - 255, & otherwise \end{cases}$$

The transformed block is: K′[i] = K[i] ⊕ Si, i = 0,1,...,63

Modification III: Bit Rearrangement:

Each block K′[i] is expressed as: (b1, b2, b3, b4, b5, b6, b7, b8). It is transformed into (b3, b4, b5, b6, b1, b2, b7, b8). Let the resulting block be denoted by: K″ = K″[0], K″[1],...,K″[63].

This processing is performed across all 64 sub-blocks, resulting in a restructured 512-bit output. Now, B[i] is split into 16 sub-blocks, each consisting of 32 bits.

B[i] = 32×16

Expansion of Sequence B[i]

Now we expand B[i] up to 64 chunks as per the expansion rule given below.

$$\boldsymbol{B[i] = \begin{cases} B[i], & 0 \le i \le 15 \\ ROTR^{7}(B[i \bmod 16]) \oplus ROTR^{13}(B[i \bmod 16]), & 16 \le i \le 63 \end{cases}}$$

Where $ROTR^k(\cdot)$ denotes right rotation by k bits. After expansion, the final structure of the B[i] box becomes

B[i] = 32×64.

Compression Function of MMD5

The proposed algorithm (MMD5) performs the same number of operations as the original MD5 hash function. The author uses a generated B[i] instead of Mi to compute the compression function, which is shown in Figure 1. The computing process of the compression function is the same as the original algorithm and uses the same Boolean function as defined below:

F (D, C, B) = (D ∧ ¬B) ∨ (C ∧ B)
G (D, C, B) = (¬D ∧ C) ∨ (D ∧ B)
H (D, C, B) = D ⊕ C⊕ B
I (D, C, B) = (B ∨ ¬D) ⊕ C

∧, ¬, ∨, and ⊕ represent AND, NOT, OR, and XOR operations, respectively. The final hash is obtained by concatenating A, B, C, and D to generate a 128-bit digest.

In the proposed algorithm, various phases of modification, together with the expansion and transformation of B[i], increase the complexity of tracing the internal state back to the original input.

These operations introduce additional diffusion, making it infeasible to set up a direct correlation between the input and the resulting hash value. It is therefore impossible to find two distinct inputs that produce the same hash value under the modified structure.

It is worth mentioning that the introduction of the TRC padding technique and B[i] in the proposed algorithm improves the diffusion and complexity of the hashing process.

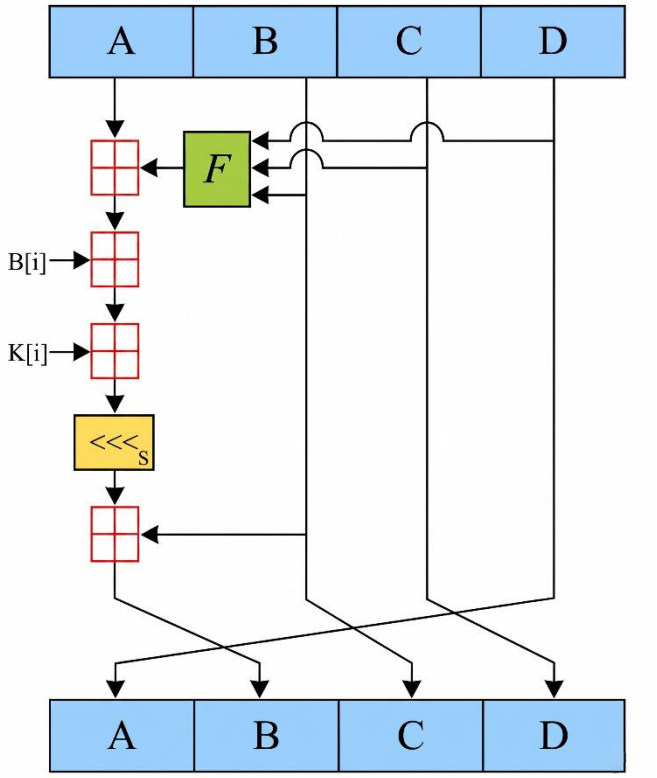


Fig. 1 Compression function of MMD5

5. Results and Discussion

In this section, the outcomes obtained from the avalanche effect and different statistical tests are discussed. The implementations of HAVAL-128, MD5, and MMD5 were carried out in Python (IDE: PyCharm).

5.1. Avalanche Effect

It is one of the key properties that ensures a small modification in the input results in a substantial change in the hash. Formally, if two input messages X and X′ differ by only one bit, then their corresponding hash outputs H(X) and H(X′) should differ in approximately half of their bits.

Let the total number of bits in the hash output be given by n. If d is the number of different bits between H(X) and H(X′), then the avalanche effect percentage is defined as: Avalanche Effect = (d/n) × 100%.

A good cryptographic hash function should ideally produce a value greater than 50%. Tables 1 and 2 show a comparison of the hash outputs obtained from the standard MD5, HAVAL-128, and the proposed algorithm (MMD5).

The analysis shows that MMD5 exhibits an improved avalanche effect, with more than 50% of the output bits changing in response to a minor alteration in the input character.

The average avalanche values computed over 15 different inputs were 50.002%, 46.45%, and 52.35% for HAVAL-128, MD5, and MMD5, respectively, which shows enhanced performance of the proposed algorithm.

5.2. Statistical Analysis

Statistical analysis is a means of assessing cryptographic integrity, ensuring that hash functions produce outputs indistinguishable from random noise. Without it, subtle patterns could allow attackers to predict outputs or find collisions.

The Binary Matrix Rank Test examines linear dependencies within a bit string, whereas the Discrete Fourier Transform Test looks for periodic repetitions, which indicate non-randomness. Meanwhile, the Chi-Square test and Approximate Entropy Test evaluate the randomness of hash outputs.

A hash function demonstrates sufficient entropy to protect confidential data and digital signatures by passing these statistical tests.

It is worth mentioning that the Discrete Fourier Transform Test and Binary Matrix Rank Test are widely used to assess the randomness of random number generation.

5.2.1. Binary Matrix Rank Test

Purpose: To determine whether the fixed-size substrings of the original string are linearly dependent.

Function call: n: Bit string length; ϵ: Bit sequences; R = Total rows in every matrix

Test description: Sequentially split the bit string into RC-bit-distinct blocks. The total number of blocks can be given as:

$$N = \left\lfloor \frac{n}{RC} \right\rfloor \quad (1)$$

Bits that are discarded will not be used in the computation of any block. Now compile the RC bits into R-by-C matrices. Now, we can find the rank of each matrix and category as follows.

Let F_R = Total matrices of rank R (full rank).
F_{R-1} = Total matrices of rank (R −1) (full rank −1)
N−F_R−F_{R-1} = Total remaining matrices

Now we compute χ^2(obs):

$$\boldsymbol{\chi^2(obs)} = \frac{(F_R - 0.2888N)^2}{0.2888N} + \frac{(F_{R-1} - 0.5776N)^2}{0.5776N} + \frac{(N - F_R - F_{R-1} - 0.1336N)^2}{0.1336N} \quad (2)$$

Further, we compute the p-value as follows:

$$\boldsymbol{p - value} = e^{-\chi^2(obs)/2} \quad (3)$$

Decision rule: If p-value ≥ 0.01, then the output string of bits is called random; else the string is non-random.

The experimental outcomes of the proposed algorithm (MMD5) are shown in Figure 2.

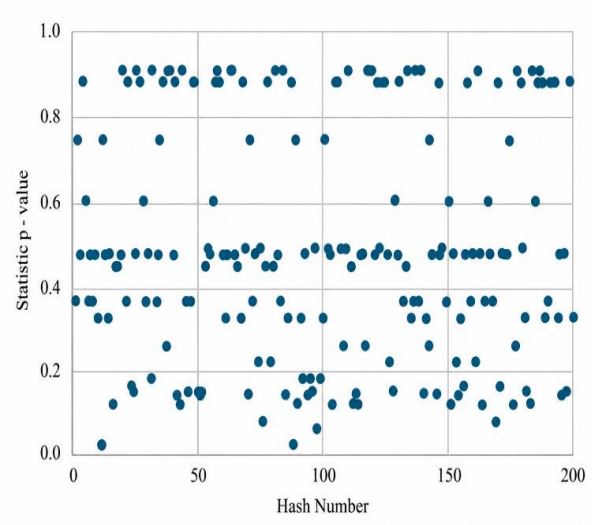


Fig. 2 Binary matrix rank test

5.2.2. Discrete Fourier Transform Test (DFT Test)

Purpose: The Fourier test identifies the periodic characteristics in the bit strings that would suggest a fluctuation from the randomness assumption.

In this, the initial binary sequence is converted to polar values, and the Fourier Test is applied to the resulting polar values to identify peaks in the power spectrum; if any peaks exceed the allowed threshold, the sequence fails the randomness test. This test is performed to determine whether the values or peaks surpassing the 95% threshold differ from 5%.

Function call: n: Bit string length; ϵ: Bit sequences (i.e. ϵ = (ϵ_1, ϵ_2,...,ϵ_n)

Test description: Transform the sequence in the form of x = x_1, x_2,..., x_n using Equation (4).

$$x_i = 2\epsilon_i - 1 \quad (4)$$

Now perform the DFT on X and get S = DFT(X). Let x_k be the kth bit of the sequence X, where k = 1, 2,...,n. Then, the terms of sequence S are obtained using the following formula:

$$\boldsymbol{f_j} = \sum_{k=1}^{n} \boldsymbol{x_k} . \boldsymbol{e}^{(2\pi i(k-1))\frac{j}{n}} \quad (5)$$

Where $\boldsymbol{e}^{\left(\frac{2\pi ikj}{n}\right)} = \boldsymbol{sin}\left(\frac{2\pi ikj}{n}\right) + \boldsymbol{cos}\left(\frac{2\pi ikj}{n}\right)$

$\boldsymbol{j = 0, 1, \ldots, n-1}$ and $\boldsymbol{i = \sqrt{-1}}$

Evaluate M = |S′|, where S′ is the substring containing the initial n/2 elements in S. Because of the symmetry, we only consider the values from 0 to $\frac{n}{2} - 1$. A sequence of peak heights generated by the modulus function. Let mod_j denote the modulus of f_j; then M = modj, and the threshold can be computed as follows.

$$\boldsymbol{T} = \sqrt{\boldsymbol{log}\left(\frac{1}{0.05}\right)\boldsymbol{n}} \quad (6)$$

For randomness, 95% of the test results shouldn't surpass T. Now, compute

$$\boldsymbol{N_0} = \frac{0.95 \times n}{2} \quad (7)$$

Where N_0 represents the expected theoretical (95%) number of peaks that are less than T, and N_1 is the observed number of peaks in M that are less than T.

Now calculate d and the p-value as follows.

$$\boldsymbol{d} = \frac{N_1 - N_0}{\sqrt{(0.95 \times 0.05)^n/_4}} \quad (8)$$

$$\boldsymbol{p - value} = \boldsymbol{2(1 - \phi(|d|)) = erfc}\left(\frac{|d|}{\sqrt{2}}\right) \quad (9)$$

Where $\boldsymbol{\phi}$(x) represents the cumulative probability function.

Decision rule: If p-value ≥ 0.01, then the output string of bits is called random; else the string is non-random.

The experimental outcomes of the proposed algorithm (MMD5) are shown in Figure 3.

5.2.3. Approximate Entropy Test

The Approximate Entropy test is used to evaluate the randomness of a binary sequence by examining the frequency distribution of overlapping bit patterns. The test analyse whether all possible overlapping blocks of length m and m+1 occur with approximately the same frequency one would expect from a random sequence. If the distribution is not uniform, then there are statistical dependencies or repetitive structures.

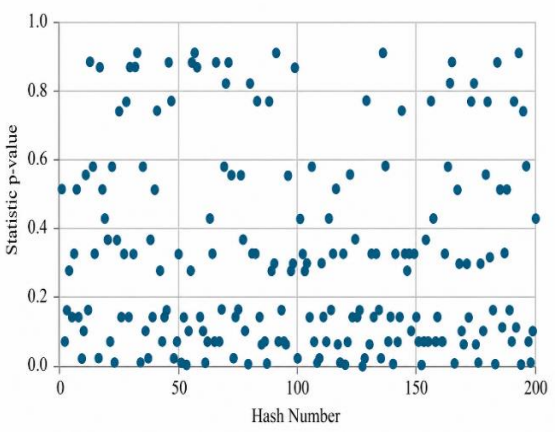


Fig. 3 Discrete Fourier Transform test

For a binary sequence of length n, overlapping blocks are created by shifting a window of size m one bit at a time, resulting in (n−m+1) patterns. The probability of each m-bit pattern is:

$$\boldsymbol{p_i = \frac{C_i}{n-m+1}} \quad (10)$$

Where C_i is the count of occurrences of the i^{th} pattern, then the pattern entropy is computed as:

$$\boldsymbol{\phi(m) = \sum_{i=1}^{2^m} p_i \, ln(p_i)} \quad (11)$$

The approximate entropy is computed by comparing block sizes m and m + 1:

$$\boldsymbol{ApEn = \phi(m) - \phi(m+1)} \quad (12)$$

The Chi-Square statistic for testing statistical significance is calculated as:

$$\boldsymbol{\chi^2 = 2n(ln - ApEn)} \quad (13)$$

and the corresponding p-value is calculated using:

$$\boldsymbol{p - value = Q(2^{m-1}, \frac{\chi^2}{2})} \quad (14)$$

The incomplete gamma complement function is represented by Q(.,.)

Decision rule: If p-value ≥ 0.01, then the output string of bits is called random; else the string is non-random.

Higher approximate entropy and an acceptable p-value indicate strong randomness, a uniform distribution of patterns, and improved resistance to statistical cryptanalysis. The experimental outcomes of the proposed algorithm (MMD5) are shown in Figure 4.

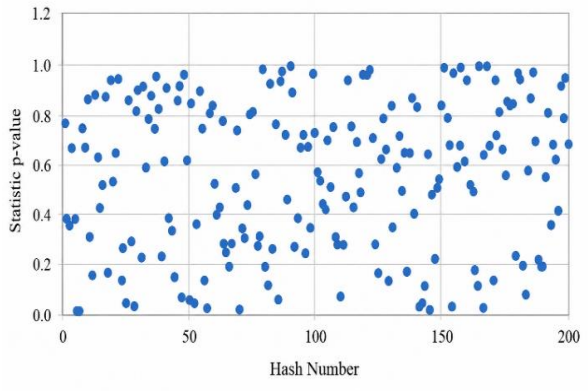


Fig. 4 Approximate entropy test

5.2.4. Chi-Square Test

Chi-Square ($\boldsymbol{\chi^2}$) test is a statistical method used to test the uniformity of hash outputs by analyzing the observed frequency of bits with the expected frequencies if they were random. In cryptographic analysis, a secure hash function should produce bits with nearly equal probabilities, hence showing no statistical bias.

Computation of the Chi-Square statistic is as follows:

$$\boldsymbol{\chi^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i}} \quad (15)$$

Where

O_i = Observed frequencies of the i^{th} category,
E_i = Expected frequencies of the i^{th} category,
k = Total number of categories.

For a binary hash sequence of length n, the expected frequencies are:

$$\boldsymbol{E_0 = E_1 = \frac{n}{2}} \quad (16)$$

For a 128-bit hash output:

$$\boldsymbol{E_0 = E_1 = \frac{128}{2} = 64}$$

The statistical significance of the computed value is determined using the p-value obtained from the chi-square distribution:

$$\boldsymbol{p-value = Q(\frac{df}{2}, \frac{\chi^2}{2})} \quad (17)$$

Where df = k −1 is the degree of freedom. For binary data: df = 2−1 = 1. Thus

$$\boldsymbol{p-value = Q(\frac{1}{2}, \frac{\chi^2}{2})} \quad (18)$$

Q(.,.) represents the incomplete gamma complement function.

Decision rule: If p-value ≥ 0.01, then the output string of bits is called random; else the string is non-random.

The experimental outcomes of the proposed algorithm (MMD5) are shown in Figure 5.

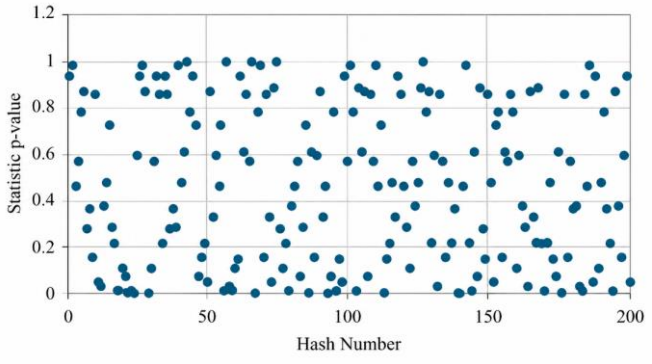


Fig. 5 Chi-Square test

Based on the computation of 200 generated hash samples, it is evident that most of the hashes pass the discrete Fourier Transform test, the Binary Matrix Rank test, the Approximate Entropy test, and the Chi-Square test. The average p-values of the discrete Fourier Transform test, Binary Matrix Rank test, Approximate Entropy test, and Chi-Square test are 0.321, 0.497, 0.556, and 0.457, respectively. The computed average results exceed the NIST-specified significance threshold (p-value ≥ 0.01) established for statistical randomness tests. Therefore, one can conclude that the proposed algorithm (MMD5) shows strong statistical randomness characteristics.

5.2.5. Execution Time

The efficiency of MMD5 is compared with MD5 and HAVAL-128, considering execution time for various input sizes of 1 KB, 5 KB, 10 KB, and 20 KB. The experimental evaluation shows that MD5 and HAVAL-128 require more execution time than MMD5 to generate the hash output, as shown in Figure 6. Based on these results, one can conclude that MMD5 can be used with any runtime application to achieve better security. In the proposed algorithm (MMD5), the author replaces the conventional padding technique with Trailing Bit Complement (TRC) padding and uses the generated B[i] instead of Mi to compute the compression function. B[i] incorporates several rounds of modification stages and advanced bit-mixing computations, which increase complexity and enhance the overall performance of MMD5. Consequently, the incorporation of the TRC padding technique and the generated B[i] box contributes to improved performance in avalanche effect, execution time, and randomness, demonstrating the efficacy of the proposed algorithm as compared to other benchmark hash functions.

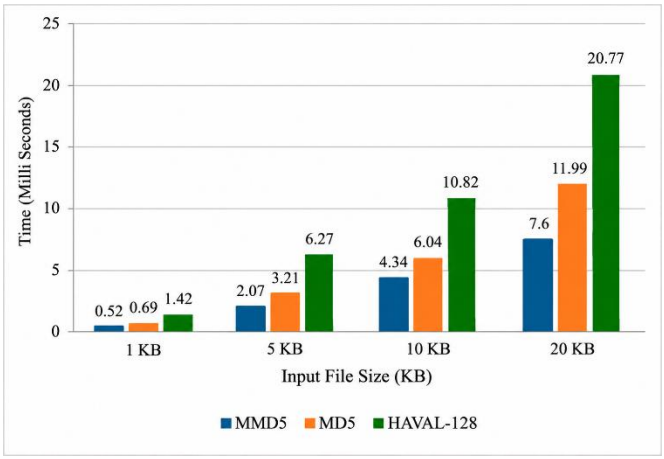


Fig. 6 Comparative execution time of MMD5, MD5, and HAVAL-128

6. Conclusion

The paper presents an optimized hashing technique that incorporates a new padding method, Trailing Bit Complement (TRC), combined with 16 multiplication-modification rounds, to enhance computational efficiency while preserving data integrity. The performance analysis of MMD5 shows an average avalanche effect of 52.35%, exceeding the standard threshold of 50% and exhibiting stronger diffusion properties than MD5 and HAVAL-128. The mean p-values obtained from the Binary Matrix Rank test, Discrete Fourier Transform test, Approximate Entropy test, and Chi-Square test were 0.497, 0.321, 0.556, and 0.457, respectively, all substantially greater than the NIST-recommended significance criterion of p> 0.01 for randomness. The proposed algorithm, MMD5, has the shortest execution time, making it suitable for runtime applications. One can conclude that the proposed algorithm (MMD5) generates a cryptographically reliable hash that meets established statistical randomness criteria.

Conflicts of Interest

There is no conflict of interest with respect to this paper.

Acknowledgments

The authors would like to thank GLA University, Mathura (UP), India, and Aligarh Muslim University, Aligarh (UP), India, for providing the academic environment and research facilities necessary for conducting this study.

References

[1] Bart Preneel, "Hash Functions," *Encyclopedia of Cryptography, Security and Privacy*, Springer Cham, pp. 1096-1109, 2025. [CrossRef] [Publisher Link]

[2] Muhammad Rehan Anwar, Desy Apriani, and Irsa Rizkita Adianita, "Hash Algorithm in Verification of Certificate Data Integrity and Security," *Aptisi Transactions on Technopreneurship*, vol. 3, no. 2, pp. 181-188, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[3] Susila Windarta et al., "Lightweight Cryptographic Hash Functions: Design Trends, Comparative Study, and Future Directions," *IEEE Access*, vol. 10, pp. 82272-82294, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[4] Gongling Zhang, "Cryptographic Techniques in Digital Media Security: Current Practices and Future Directions," *International Journal of Advanced Computer Science and Applications*, vol. 15, no. 8, pp. 1-9, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[5] Roayat Ismail Abdelfatah, Esraa Abdelkhalek Baka, and Mohamed E. Nasr, "Keyed Parallel Hash Algorithm based on Multiple Chaotic Maps (KPHA-MCM)," *IEEE Access*, vol. 9, pp. 130399-130409, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[6] Carlos Cid, "Recent Developments in Cryptographic Hash Functions: Security Implications and Future Directions," *Information Security Technical Report*, vol. 11, no. 2, pp. 100-107, 2006. [CrossRef] [Google Scholar] [Publisher Link]

[7] Han Zhuoyu, and Li Yongzhen, "Design and Implementation of Efficient Hash Functions," *2022 IEEE 2nd International Conference on Power, Electronics and Computer Applications (ICPECA)*, Shenyang, China, pp. 1240-1243, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[8] Vladislav Dmitrievich Zyuzin et al., "Attack on Hash Functions," *Eurasian Journal of Biosciences*, vol. 14, no. 1, pp. 907-913, 2020. [Google Scholar]

[9] Aboubakar Nkouankou et al., "Pre-Image Attack of the MD5 Hash Function by Proportional Logic," *International Journal of Research and Innovation in Applied Science*, vol. 7, no. 8, pp. 20-25, 2022. [Google Scholar] [Publisher Link]

[10] Tameem Hameed Obaida, Hanan Abbas Salman, and Hasan Najim Zugair, "Improve MD5 Hash Function for Document Authentication," *Webology*, vol. 19, no. 1, pp. 7223-7234, 2022. [Google Scholar] [Publisher Link]

[11] Hayder Al-Ghanimi et al., "Detection of Attacks by Assessing the Performance of MD2, MD5 and SHA-1 Hashing Algorithms," *2023 6th International Conference on Contemporary Computing and Informatics (IC3I)*, vol. 6, pp. 2729-2733, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[12] Agung Nugroho, and Teddy Mantoro, "Salt Hash Password using MD5 Combination for Dictionary Attack Protection," *2023 6th International Conference of Computer and Informatics Engineering (IC2IE)*, Lombok, Indonesia, pp. 292-296, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[13] Petar Boyanov, "Practical Applications of Hash Functions MD5, SHA-1, and SHA-256 using Various Software Tools to Verify the Integrity of Files," *Journal Scientific and Applied Research*, vol. 27, no. 1, pp. 120-137, 2024. [Google Scholar] [Publisher Link]

[14] Aravindhan Kurunthachalam, and Nisar Ahmed L, "The Secure User Authentication and Data Sharing for Mobile Cloud Computing using MD5 and ElGamal Encryption," *International Journal of Advances in Computer Science and Engineering Research*, vol. 1, no. 2, pp. 104-117, 2025. [Publisher Link]

[15] Aseel AlShuaibi, Muhammad Waqas Arshad, and Mohammed Maayah, "A Hybrid Genetic Algorithm and Hidden Markov Model-based Hashing Technique for Robust Data Security," *Journal of Cyber Security and Risk Auditing*, vol. 2025, no. 3, pp. 42-56, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[16] Chris Gilbert, and Mercy Gilbert, "Exploring Secure Hashing Algorithms for Data Integrity Verification," *SSRN Electronic Journal*, 2025. [Google Scholar] [Publisher Link]

[17] Ahmed Ali Talib Al-Khazalli et al., "Utilizing MD5 Hashing in Digital Forensics: A Comprehensive Analysis," *Engineering Headway*, vol. 35, pp. 295-303, 2026. [CrossRef] [Google Scholar] [Publisher Link]

[18] Nurdin Nurdin, Al Habbal, and Fajriana Fajriana, "Comparative Analysis of MD5 and SHA-256 Hash Algorithms for Fingerprint File Integrity Verification," *Journal of Applied Informatics and Computing*, vol. 10, no. 2, pp. 1495-1504, 2026. [CrossRef] [Google Scholar] [Publisher Link]

[19] Zhenqi Wang, and Lisha Cao, "Implementation and Comparison of Two Hash Algorithms," *2013 International Conference on Computational and Information Sciences*, Shiyang, China, pp. 721-725, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[20] Marc Stevens et al., "The First Collision for Full SHA-1," *Advances in Cryptology – CRYPTO 2017*, Springer Cham, vol. 10401, pp. 570-596, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[21] Peyman Ayubi, Saeed Setayeshi, and Amir Masoud Rahmani, "Chaotic Complex Hashing: A Simple Chaotic Keyed Hash Function based on Complex Quadratic Map," *Chaos, Solitons and Fractals*, vol. 173, pp. 1-20, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[22] Darshana Upadhyay et al., "Investigating the Avalanche Effect of Various Cryptographically Secure Hash Functions and Hash-based Applications," *IEEE Access*, vol. 10, pp. 112472-112486, 2022. [CrossRef] [Google Scholar] [Publisher Link]

Table 1. Comparative analysis of avalanche effect between Haval-128 and MMD5

No.	Input Size	Message Digest (Haval-128)	Avalanche Effect	Message Digest (MMD5)	Avalanche Effect
1	The window in my room is closed.	91a244adde26b57153e3f16575d70f8e	58.59	a2e536c3adf85404a81cd0292f8f6b92	52.34
	The Window in my room is closed.	76ace2f4a91adb7f982cdc0a825b90c7		ae42e97b8d0f9c880b7be75536919b5c	
2	number theory is queen of mathematics	7ee57e97c46c82e8b5c3aac8c31e9f5a	49.22	1b876e3ea59369befa29407862c27184	49.22
	Number theory is queen of mathematics	ed2e8c03dd0f89607dad9417d88d1e82		af446d13e285904cd18b7f9544e63f4c	
3	Puma is a shoe brand.	21c18f6817f2c81be1ec48f8b8f6b8e1	53.12	8650ed9112305b18d46bfe7788775978	55.47
	Puma is a Shoe brand.	f41c115025ef77d46be69fa43cd0ca4a		9b7e0d328f114960c5710752b449d3b2	
4	I have a laptop bag.	bd13b686294b6e4ea2c3c0cdd9f21448	59.38	c112e35f72b07eab25f10847955aa10f	51.56
	I have a Laptop bag.	2a7721d5d4536f1f431e2bde10d05d2e		877de22c8379927aaff9a950be58a8d2	
5	Badminton is an indoor sport.	a77909c15c59e2c511c521560568e571	45.31	78c621674a8c13f3feabdae57acde33b	52.76
	Badminton is an Indoor sport.	fe7d690509337978c6c91378df828535		4931dbb001d17a551f9d9eff5f8f1c96	
6	Football is an outdoor sport.	feb05747b068ef8da5665398e13d3191	53.91	78cca634f3d918864932a7566a3028d7	51.97
	Football is an Outdoor sport.	37f1f3b889fb79c74f56aa446f4ba82c		65234213d6196d5b463c3b29f11b2e96	
7	There are several railway tracks in aliagrh.	24ffe4dba6840d363a6b87394fea43b6	46.09	e69dcadb76a25ef8c1a029c69a17e3ba	53.12
	There are several Railway tracks in aliagrh.	ec11844d53bd2fa520e1d437bb2eeb3b		137bf6e455f0d19a635a0c3a1c60cee9	
8	We have semester exam next month.	fc3227ca3741ad665b9f8ad513a02c8a	54.69	8d740aa1f434b5db15855c19f82ddb68	53.91
	We have semester Exam next month.	b0a6bed5017736b5ef23312c0da7f4e7		5a91d05bc056ae5a0a0ee90d6bf2791d	
9	There are 20 departments in our university.	54f414d7c989975ab15523e134c7bb14	55.12	9d7770461b11014157c564762eb08847	52.67
	There are 20 Departments in our university.	7994ee0b0c0968ad8b70349e888e26d4		213a39c221384f271dfdc59ab5b3d5d9	
10	It is getting hotter and hotter in this city.	ec4f1149b5b242e326a74d0ef2290487	45.31	55924ee3468e9e46bdc42338ae1ba795	54.69
	It is getting Hotter and Hutter in this city.	95515d7994cf778948a9ae0e298b6c0d		fb887b5c3d2ca6eb33842d5b2e81520f	
11	Rockstar was a film ahead of its time.	26168d9b9c032538ba268c0358a271fe	55.56	7fb3688f2b2f870c39213adea787f14c	50.78
	Rockstar was a Tilm ahead of its Fime.	1cc0c13411b24fab421353269734e53d		8a85c3d6ec51d1a9ae83680e30250816	

12	India and Pakistan were separated a long time ago.	d63a1634d39992ee76fdf033fb874c89	43.75	c42d762108f100f12e09399f760f882c	54.33
	India and Pakistan were separated a long time ago.	80717600d95e6ab2a4bd3bc6f1a22c61		7e386d5329df7afb87f1929c52aa4bc3	
13	Doctor's recommend drinking a lot of water daily.	7e8ae0527939fb9b6a816ef9eeb00ef8	37.01	ba2bdd7c958af6db8e78d7139b9a7f65	51.56
	Doctor's recommend Drinking a lot of water daily.	053f19f6b59715f9caf0003e637fe6cc		45d422836a7509244e78d7139b9a7f65	
14	Avalanche effect is important in secure hash function design	dfa6904c6846169b963f2a4aa98184b2	43.75	9069c6bab2f9cd8df76325373ba8cd1d	50.22
	Avalanche effect is important in secure hash function designs.	09ec1f5c094a1d27420d851df65f99d3		20af7e38d0218052da890c366ae54856	
15	Even one bit change should flip many bits in output hash value	bf0b9e6845bbc7d28a7ff54050002866	49.22	9eaf0d0d387f54e4bf14b0fbd355c8ee	50.78
	Even one bit change should flip many bits in output hash values.	a4f3f8e998493f912ebae1d33e513891		57c952b179252a633b8beba90b26dbac	

Table 2. Comparative analysis of avalanche effect between MD5 and MMD5

No.	Input Size	Message Digest (MD5)	Avalanche Effect	Message Digest (MMD5)	Avalanche Effect
1	The window in my room is closed.	418348b7bcef12a5b383e05fb6304686	42.52	a2e536c3adf85404a81cd0292f8f6b92	52.34
	The Window in my room is closed.	eae11a3708e92f64098a38aa33d077d2		ae42e97b8d0f9c880b7be75536919b5c	
2	Number theory is queen of mathematics	5c86300a28e7d3cdbf4a9a84119249ec	47.24	1b876e3ea59369befa29407862c27184	49.22
	Number Theory is queen of mathematics	029064221bfa750ab9363e0da7e4e524		af446d13e285904cd18b7f9544e63f4c	
3	Puma is a shoe brand.	04085cb5a8d39973f025a17c9a76b571	51.59	8650ed9112305b18d46bfe7788775978	55.47
	Puma is a Shoe brand.	c82ca16086771c9c47581f861d50f58e		9b7e0d328f114960c5710752b449d3b2	
4	I have a laptop bag.	f23305a7dcab78d80c33522fb837c1ed	44.53	c112e35f72b07eab25f10847955aa10f	51.56
	I have a Laptop bag.	cb879325639599df74e02a835ed0baf7		877de22c8379927aaff9a950be58a8d2	
5	Badminton is an indoor sport.	7ef5a5295921656d4b37e10a764a2952	47.66	78c621674a8c13f3feabdae57acde33b	52.76
	Badminton is an Indoor sport.	cd71fbc811a057382f927b7ee06d0338		4931dbb001d17a551f9d9eff5f8f1c96	
6	Football is an outdoor sport.	2155f27c8dad87f025547fb8b1145f6d	43.75	78cca634f3d918864932a7566a3028d7	51.97

	Football is an Outdoor sport.	c239b228d5bc12fd3e3eed6a5186eb97		65234213d6196d5b463c3b29f11b2e96	
7	There are several railway tracks in aliagrh.	ab50be0cb82b079dc390636c7dc66b72	44.53	e69dcadb76a25ef8c1a029c69a17e3ba	53.12
	There are several Railway tracks in aliagrh.	395f0d6ccaf2f5244014c3a9a703d33a		137bf6e455f0d19a635a0c3a1c60cee9	
8	We have semester exam next month.	ad0c40adf2020b0d707fea4f3f4274ee	54.69	8d740aa1f434b5db15855c19f82ddb68	53.91
	We have semester Exam next month.	be124f120d854425cdbfa02438754a88		5a91d05bc056ae5a0a0ee90d6bf2791d	
9	There are 20 departments in our university.	a18c270a655cdb118cf537e4cb3b6001	49.22	9d7770461b11014157c564762eb08847	52.67
	There are 20 Departments in our university.	482a37cdc5bde0e545eb291a487b9a42		213a39c221384f271dfdc59ab5b3d5d9	
10	It is getting hotter and hotter in this city.	641731caa0ca1fb01a3ed2cbf67b12d7	46.88	55924ee3468e9e46bdc42338ae1ba795	54.69
	It is getting Hotter and Hutter in this city.	f38ef172f2a03eddea57e8029140f00d		fb887b5c3d2ca6eb33842d5b2e81520f	
11	Rockstar was a film ahead of its time.	15602e9da4c66dfdc6fba76f7f5f1ba4	43.75	7fb3688f2b2f870c39213adea787f14c	50.78
	Rockstar was a Tilm ahead of its Fime.	919a7bcd50b165248a16411386b7a81f		8a85c3d6ec51d1a9ae83680e30250816	
12	India and Pakistan were separated a long time ago.	c2871963e616255a26b39b315bf6d23e	45.67	c42d762108f100f12e09399f760f882c	54.33
	India and Pakistan were separated a long time ago.	407d8d6d7f89115ba41d313f3b15dbb2		7e386d5329df7afb87f1929c52aa4bc3	
13	Doctor’s recommend drinking a lot of water daily.	f9eea0d97046fce865a7c73b541bc017	44.53	ba2bdd7c958af6db8e78d7139b9a7f65	51.56
	Doctor’s recommend Drinking a lot of water daily.	9ad1e221b6912433068f705883889f48		45d422836a7509244e78d7139b9a7f65	
14	Avalanche effect is important in secure hash function design	07fb2acbff49cad90d7ee46175c5f3cc	42.97	9069c6bab2f9cd8df76325373ba8cd1d	50.22

	Avalanche effect is important in secure hash function designs.	1438f47210c2473201cdffa07104b1a7		20af7e38d0218052da890c366ae54856	
15	Even one bit change should flip many bits in output hash value	2a58f939974dbc2ea0585c03ea08718b	47.24	9eaf0d0d387f54e4bf14b0fbd355c8ee	50.78
	Even one bit change should flip many bits in output hash values.	66dffabe65f45455df5fdc1562db33ac		57c952b179252a633b8beba90b26dbac	