Artificial Intelligence for Human-Technologies-Economy Sustainable Development
23–25 May 2024
Lublin, Poland
make learn
TIIM
PIConf
Management, Knowledge and Learning
International Conference
Technology, Innovation and Industrial Management
Sustainable Development

Enhancing Cybersecurity with Practical Cryptographic Hash Algorithms

Mohammed Nasir Salihu
Information and Communication Technology Department, Tetfund, Nigeria
salihumn@gmail.com

Abdulrasheed Jimoh
Department of Computer Science, Faculty of Science, Gombe State University, Gombe, Nigeria
jimohabdulrasheed1969@gmail.com

Suraj Salihu
Department of Computer Science, Faculty of Science, Gombe State University, Gombe, Nigeria
surajsalihu@gsu.edu.ng

Bala Modi
Department of Computer Science, Faculty of Science, Gombe State University, Gombe, Nigeria
bmodi@gsu.edu.ng

Abstract

The swift growth of cyberspace, facilitated by innovative networking and computing technologies, highlights cybersecurity's critical importance. With widespread reliance on web applications, intruders' persistent exploitation of vulnerabilities poses significant risks. As internet data transmission volumes soar, algorithms ensuring data confidentiality and integrity are urgently needed. This work addresses this need by reviewing two commonly used one-way cryptographic hashing algorithms, SHA and MD, along with their variants. The strengths and weaknesses of these algorithms are identified primarily based on their output lengths. Leveraging the findings from this review, suggestions are made to enhance cybersecurity measures in the face of evolving cyber threats. Utilizing Tools4Noobs, an online hash calculator, empirical testing is conducted to further validate the efficacy and resilience of these cryptographic hash functions. This investigation contributes to the ongoing discourse on cybersecurity, providing an understanding of the practical applications and implications of cryptographic algorithms in safeguarding information and computing resources.

Keywords: cyber-security, cryptography, hashing algorithm, SHA, MD

1.0 INTRODUCTION

Cyber-security is a collection of guidelines, methods, technologies, and procedures that combine to protect the integrity, secrecy, and accessibility of computing resources, software programs as well and data from breaches (Thakur, 2024). Many attackers continue to have an alternative because they only need to uncover one weakness or the other in the system that has to be protected (No & Vasarhelyi, 2017). The realm of cybersecurity is fraught with notable incidents where various entities, ranging from state actors to criminal organizations, have perpetrated attacks with significant consequences. For instance, specific incidents where breaches occurred were highlighted; in 2014, Hold Security experienced a breach affecting 5 million email accounts, attributed to Russian hackers (Plachkinova & Maurer, 2018). Around the same time, Google encountered a breach affecting 5 million passwords, also linked to Russia (Thomas et al., 2017).

The NotPetya attack in 2017, attributed to the Russian military, caused approximately $1 billion in losses by targeting critical infrastructure worldwide (Trautman, 2018). Similarly, the WannaCry ransomware attack, allegedly orchestrated by North Korea's intelligence, impacted PCs in 150 countries (Stoddart, 2022), with the UK's National Health Services bearing a heavy brunt (Akinola & Afonja, 2022). The SolarWinds cyber operation of 2020, linked to Russia as well, affected around 100 firms, showcasing the sophistication and scale of modern cyber threats (Trautman, Shackelford, Elzweig, & Ormerod, 2024). Subsequently, in March 2021, suspicions of Chinese cyber espionage arose following the compromise of 30,000 organizations in the United States by exploiting vulnerabilities in Microsoft systems (Milton, 2022). More recently, the Colonial Pipeline incident in May 2021, allegedly involving Russian perpetrators according to the FBI, disrupted the US gasoline supply chain, affecting millions of consumers (Lubin, 2022). These incidents highlight the ongoing challenges and complexities in safeguarding digital infrastructure against cyber threats from various actors on a global scale.

The motivation behind this work is the fact that on a regular basis, businesses and organizations around the world are victims of ransomware and other large-scale hacking and attacks (Swinford, Huesken, Mierow, Ariyaratne, & Ismail, 2023). The paper aims to review the weaknesses and strengths of the two common cryptographic hash functions, that is Message Digest (MD) and Secure Hash Algorithm (SHA) with their corresponding variants, as well as based on their output length, description to identify their vulnerabilities and offer meaningful suggestions in order to attain the security of information and computing devices. The study also aimed to determine the hash values for both short and long passwords, specifically "abc123" for short and "abcdefghijklmnopqrstuvwxyz" for long, across different algorithms.

2.0 LITERATURE REVIEW

2.1 HASHING ALGORITHM

Hashing algorithms, versatile tools employed across various domains such as cybersecurity, data management, and cryptographic applications, play a pivotal role in guaranteeing the integrity, security, and efficiency of data (Sahni, 2015). Hashing is a type of algorithm that accepts any size of input while converting it into a fixed-size output. Depending on the type of the algorithm employed, the result of hashing, also known as a digest, is always a fixed length, regardless of the length of the input, According to (Debnath, Chattopadhyay, & Dutta, 2017). Hash functions were originally developed to improve the efficiency of digital signatures, but they are today used to secure the basic foundations of information infrastructure (Dahal, Bhatta,& Dhamala, 2013).

Cryptography protocol, Password protection, digital signatures, message authentication, data integrity, pseudo-random number creation and key derivation are all examples of uses for hashfunctions (Sudha & Monica, 2012). The hashing algorithm should possess the following characteristics (Sahni, 2015)

1. Two distinct messages must not give the same hash value. (collision-free)
2. It must be challenging to retrieve a message with a specified hash value. (inversion free)

In order to retain the power of cryptographic algorithms when new threats to systems and information develop, NIST has been sponsoring a transparent, public challenge to establish a new, resilient cryptographic hash algorithm. When the new method is approved, it will take theplace of the current hash algorithm in FIPS 180-4. (Radack, 2012).

Hash functions employed in security-related applications are known as cryptographic hash functions (Merkle, 1979). Cryptographic hash functions are appropriate for a variety of security uses due to their many significant features. Cryptographic hash functions' primary objective is to improve security, key characteristics of cryptographic hash functions are as follows;
1. **Deterministic**: A cryptographic hash function will always yield the same result for the sameinput. For the sake of consistency and verification, determinism is essential (Preneel, 2010).

2. **Fast Computation**: In security protocols, where efficiency is vital, hash functions must be computationally faster to enable speedy data processing (Anwar, Apriani, & Adianita, 2021).

3. **Pre-image Resistance**: Here, it appears impractical in terms of computation to locate the initially entered data from a hash value (Primmer & D'Halluin, 2013). The hash function is supposed to resist attempts at reverse engineering or hash-based input discovery (Vasić, Kukec,& Mikuc, 2011).

4. **Collision Resistance**: Identifying two different inputs that produce the same encrypted value ought to be very challenging (Primmer & D'Halluin, 2013). Because collisions compromise the individuality of hash values, they diminish the security of hash functions (Aumasson, Meier, Phan, & Henzen, 2014).

5. **Avalanche Effect**: The hash output need to differ noticeably with a modest alteration in theinput. By preventing comparable inputs from yielding identical hash results, this improves security (Nechvatal et al., 2001).

6. **Fixed-Output-Size**: A cryptographic hash algorithm generates fixed-size function output irrespective of the quantity or dimensions of its input (Kamra, Sharma, & Leekha, 2019). Handling and comparing hash values is made easier by this feature (Khan et al., 2022).
7. **Efficient to Compute**: To produce the hash value, the hashing algorithm should be computationally effective, needing a suitable amount of processing power (Sadeghi-Nasab &

Rafe, 2023). On the other hand, finding collisions or reversing them must be computationally expensive (Chi & Zhu, 2017).

9. **Defense Against Birthday Attacks**: A birthday attack is the identification of two different inputs that result in a common hash value (Gauravaram, 2007). Hash functions for cryptography should withstand birthday attacks, making the computational search for these collisions challenging (Kamra et al., 2019).

10. **Non-reversible**: Hashing functions are intended to be a single direction, which means that obtaining the original input via the encrypted value would be computationally impossible if the process were reversed (Delfs, Knebl, Delfs, & Knebl, 2015).

11. **Keyless**: Usually, one can utilize hashing cryptography algorithms not using secret key. (Bouillaguet, 2011). Basic cryptographic hash functions can function without a key, in contrast to Hash-Based Message Authentication Codes (HMACs), which require a secret for added protection (Alabrah, 2014).

Central to the effectiveness of cryptographic hash functions are several key properties that govern their behavior and utility in safeguarding digital assets against malicious tampering and unauthorized access, among these properties, we have:

1. Ensuring authenticity: After receiving the message to be authenticated and the secret key as inputs, the hash function produces a Message Authentication Code (MAC). Any modifications to the message can be identified by recalculating the MAC at the user's end. This demonstrates integrity and authentication (Rizzardi, Sicari, & Coen-Porisini, 2022).

2. Digital signature: Alternative to generating a digital signature for the full message, It is more convenient to utilize the message's hash value (Srivastava, Baksi, & Debnath, 2023).

3. Password security: The message's hash value can be preserved if passwords are shielded from unauthorized users. The user-provided password's hash value is checked against to the stored hash value during user authentication (Huang, Ma, & Chen, 2011). The hash function essentially has three basic features. Nevertheless, other derived qualities also exist (Chi & Zhu,2017).

4. Pre-image resistance: A pre-image resistance is defined as a hash function "g" once it is extremely difficult to determine the input (x) based on a function's given output (h) of the function, that is, g(x) = h, another name for this characteristic is one wayness (Al-Kuwari, 2011).

5. 2nd Pre-image resistance: The function here is considered to have 2nd pre-image resistance when it is challenging to locate the second input (y) such that it yield the output, g(x) = g(y), for any given known input (x) (Al-Kuwari, 2011).

6. Collision resistance: Is a hashing algorithm g which becomes collision-resistant when it is difficult to identify a number of distinct inputs i.e two in a way that the hashing produces identical result for both inputs (Primmer & D'Halluin, 2013).

The aim is to study the popular cryptographic algorithms, MD and its variants, i.e., MD2, MD4,MD5 as well as SHA and its variants, i.e., SHA- (3, 2, 1 and 0) (Kundu & Dutta, 2020).

1. MD2 Algorithm

As opined by (Muller, 2004) that MD2 function was the first hash algorithm created by RonaldRivest to Secure RSA. It generates 128-bit message digests. The recommended security level of 2 is not met by MD2. The fundamental compression function was vulnerable to preimage assaults, the most sophisticated of which had a complexity of 2. Consequently, an attack on theentire MD2 hash in pre-image having a complexity of 2 is possible.

Description of the MD2 algorithm procedure, the message digest, or MD2 (Kaliski, 1992) can be expressed as follows

1. Adding bytes used for padding: The message of arbitrary length M is appended with "x"bytes of value "x" to make its length (in bytes) consistent with zero modulo 16. Let Mi,j represent the padded message's jth block.
2. Adding the message's checksum (which computes a 16-byte checksum, C1, C2,..., C16).
3. Initialization of 48-byte buffer to maintain the message digests initialized to zero.
4. Using the 16-byte block to process the message.

There is an attacking assault on Chouvoud and Rogier, which is capable of finding collision accordingto MD2's shortcomings (Rogier & Chauvaud, 1995)

2. MD4 Algorithm

MD4 message digest technique produces a 128-bit output in an input of any length message "message digest" or "fingerprint", so the production of second messages with a similar message digest (ideally) is computationally impractical to generate a message with the predetermined target message digest in place (Ronald Rivest, 1992). He further asserted that "with MD4, huge files can be safely "compressed" before being signed using RSA public-key cryptosystem as an example, making this technique perfect for digital signature applications."(Neforawati & Arnaldy, 2021)

Description of MD4 Algorithm Procedure

1. Adding bytes used for padding plus the size: A message M, which can have any length, is padded by one "1" bit accompanied by a sufficient number of "0" bits so the length (measured in bytes) of the padded message is equal to 448, modulo 512. The remaining length is then added to make it a multiple of 512 (in bits) by adding the size of M, which isjust the bare minimum of 64 significant bits in the original size (Mihailescu, Nita, Mihailescu, & Nita, 2021).

2. Initialization: To store a 128-bit message digest, a four-word buffer IV = (a, b, c, d) is initialized to a = 67452301, b = efcdab89, c = 98badcfe10325476 (in hexadecimal).

3. Breaking down the message into 16-word (512) chunks

4. Message digest: MD = (a, b, c, d) is the four-word (128-bit) output that is generated as the message digest (Khan et al., 2022).

Attack on MD4, Boer and Bosselaer have discovered an assault on the previous two rounds ofMD4. They are able to find collision in MD4, when the first circle of the technique is committed. Voudenay has demonstrated how to build collision if the last round of the

the algorithm is committed. When the entire MD4 is employed, his attack also discovered two similar digests according to Hamming distance (Den Boer & Bosselaers, 1991).

3. MD5 Algorithm

According to (R Rivest & Group, 1992), the MD5 algorithm was developed by Professor Ronald L. Rivest in 1991. It is an extension of the MD4 message-digest algorithm. Compared to MD4, it is marginally slower but has a more "conservative" architecture. Because MD4 appears to be adopted for use more quickly than the current evaluation could support, MD5 was created, MD4 being "at the edge" when it comes to the possibility of a successful cryptanalytic attack due to its incredibly rapid speed. MD5 eases up a little, sacrificing a little speed in exchange for a far higher chance of ultimate security (Aljuffri, 2024).

Description of MD5 Algorithm Procedure

1. Addending the span and the padding bits: One "1" bit and sufficient "0" bits are used topad the message M, which can have any length so that the total length of the padded message in bits is equal to 448, modulo 512 (Obaida, Salman, & Zugair, 2022). In a lengthof M, or only the least 64 significant bits, is then added, making the length at the end a multiple of 512 (in bits) (Bertoni, Daemen, Peeters, & Van Assche, 2014).
2. **I**nitialization: A four-word buffer IV = (X1, X2, X3, X4) This sets the initialization (in hexadecimal) of a 128-bit message digest to:
   X1 = 67452301, X2 = efcdab89, X3 = 98badcfe, X4 = 10325476

3. Message processing in 512-word (16-word) blocks: let set a = x1, b = X2, c = X3, d = X4, e = X5.
4. Message digest: The produced MD is a four - word i.e 128 - bits output MD = (a, b, c, d).

Attacks on MD5, in 2004, (Wang, Feng, Lai, & Yu, 2004) presented the first MD5 collision.Later in, (Wang & Yu, 2005), presented their differential path and adequate condition forthis separate path to happen, as well they made this remark: “These days, MD5 is one of the most popular cryptographic hash algorithms. It was created in 1992 to enhance MD4, and numerous authors have since thoroughly investigated its security. The most well-known outcome to date was an accident that occurred during a semi-free start, i.e. collision,where a non-standard value, the outcome of the attack, replaces the hash function's original value. (Obaida et al., 2022)”

4. RIPEMD

RIPEMD stands for Race Integrity Primitives Evaluation Message Digest; this cryptographic hash algorithm was created by Hans Dobbertin, Bart Preneel and Antoon Bosselaers at the Katholieke Universiteit Liuven's COSIC research group. The first edition came out in 1996; typically, RIPEMD hashes are 40-digit hexadecimal digits (Sadeghi-Nasab & Rafe, 2023).

Attack on RIPEMD, in the 1997 (Dobbertin, 1997) discovered an attack on reduced ripemd which is similar to that of MD4. Thus, they made this remark: Collisions can be identified viaan attack that starts with an essentially very basic premise if the RIPEMD compress function'sinitial or final round is neglected. Therefore, we provide a RIPEMD attack that yields results similar to those previously reported for MD4, despite the fact that our approach requires moreprocessing power (Mihailescu et al., 2021).

5. SHA Algorithm Procedure

Appending the padding bits and length: A message M, which can have any length, it includes padding of "1" one bit with then enough “0” zero bits to guarantee that the padded message's length (measured in bytes) is equal to 448 of modulo 512. The span of M, or merely the least 64significant bits, is equally added, making the length in the end a multiple of 512 (in bits) (Khanet al., 2022).

1. Initialization: A five-word buffer IV = (X1, X2, X3, X4, X5) whereby a 128-bit MD isstored in hexadecimal (initialized) to
X1 = 67452301, X2 = efcdab89, X3 = 98badccfe, X4 = 10325476
3. Interpreting the message in blocks of 16 words (512): Assign values A, B, C, D,E, to and X1, X2, X3, X4 and X5 accordingly.

4. Message digest: The output MD = (X1, X2, X3, X4, X5) is a 5-word (160-bit) messagedigest.

i. SHA-0

SHA-0 is the initial cryptographic hashing that NIST disclosed in May 1993; the structure was modelled after Merkle Damgard. A 160-bit hash value is the result, whilea 512-bit input message is the input. A six-step local collision can begin at any stage in SHA-0. Local collisions that overlap can be used to easily generate a linear divergent path for an entire SHA-0 collision. Therefore, in order to identify the initial stages of local collisions, a disturbance vector is required. The attack's complexity is determined by the vector's real weight (Preneel, 2010).

Attack on SHA-0, according to (Biham & Chen, 2004), SHA-0 is vulnerable to the following types of attacks: There are two SHA-0 near-collisions complete compression process, whereby up to 142 out of the 160 resulting bits are identical. In addition, a substantial number of entire collisions with SHA-0 reduced by 65 rounds were discovered, a significant enhancement above the outcome of the 35 rounds. There are a lot of neutral bits in the messages; some of them don't change the differences for 15–20 rounds. It was also demonstrated that, although having extra rounds, the SHA-0 with 82 rounds becomes significantly less powerful than the SHA-0 (80 rounds). This fact shows that SHA-0's strength cannot be constant across a givennumber of rounds (Xu, 2020).

ii. SHA-1

In April 1995, NIST released the initial improved version of SHA-0. This version included an additional rotation and a modified process for extended message words. The foundation of the SHA-1 is derived from ideas shared by Professor Ronald L. Rivest of MIT, who was primarily the designer of the message digest algorithm [MD4], which is the algorithm that is modelled after it (Stevens, Bursztein, Karpman, Albertini, & Markov, 2017).

Attack on SHA-1, applying the strike-from-the-middle strategy to decreased SHA (0 and 1) hash algorithms, is made possible by the recently developed cryptanalytic techniques.

Utilising computations of the compression functions $2^{156.6}$ and $2^{159.3,}$ respectively, theassaults are able to find preimages in up to 52 and 48 steps for SHA-0 and SHA-1 accordingly, whereas the assault using brute force goes through 2^{160} computation of the compression function. Preimages up to 49 and 44 steps are foundby the prior best attacks, respectively (Aoki & Sasaki, 2009).

iii. SHA-2

As stated by (Martino & Cilardo, 2020), SHA-2 was initially released in 2001, when FIPS 180-2 specified its two primary variations, SHA-512 and SHA-256, which are the 64-bit and 32-bit editions of the same hash algorithm. The same version also included information on SHA-384, a reduced version of SHA-512. Other shortened versions were introduced to the pipeline by later SHS updates, like SHA-224 adopted by FIPS 180-3 in 2004, this is a SHA-256 version, while SHA-512/224 as well as SHA-512/256, two modifications of SHA-512, were added by FIPS 180-4 in 2012, along with the general guidelines for the SHA-512/t t-wide hash algorithm called SHA 512t (Dobraunig, Eichlseder, & Mendel, 2015).

SHA-2 is currently the most widely used hash function at the time, and it has grown essential in many different applications. Historically, network security has been the primary application domain for hash functions, typically over the Internet. SHA-2 is not known to be vulnerable to any classical pre-image attacks (Amy et al., 2016).

vi. SHA-3

SHA-3 (NIST, August 2002) has a completely different design principle. Its primary feature, which is that it supports both variable length input and variable length output,is based on the sponge construction. The breadth of the permutation is determined by adding the two parameters, c (capacity) and b (bitrate), which form the basis of the construction (Kishore & Raina, 2019). According to the bit width of the permutation,SHA-3 was selected in 2012 following a public competition among non-NSA designers, and it was previously known as Keccak. Each of the 24 cycles in the mainfunction has 5 sub-steps. Three phases make up SHA-3's structure, which is based on sponge architecture: initialization, absorption, and squeezing. The state matrix has 0 initialization during the initialization phase, the first 24 rounds constitute the absorbing phase, and the necessary length hash value is obtained by truncating the state matrix during the squeezing phase. The SHA-3 family includes two extendable-output functions (XOFs), named SHAKE128 and SHAKE2562, and four cryptographic hash functions, called SHA3-224, SHA3-256, SHA3-384, and SHA3-512. (Kundu & Dutta, 2020).

Attack on Sha-3, According to (Zhou et al., 2020): The SHA-3 algorithm standard was approved by NIST in August 2015. SHA-1 and SHA-2 have similar construction and basic mathematical operations, whereas SHA3 is less susceptible to intrusions.

From the above, it can be asserted:

1. MD and SHA, with their variants, are the most frequently used cryptographic hashalgorithms.

1. Cryptographic hashing is being used to achieve confidentiality and availability of information and computing devices.

2. Cryptographic hash function has a wide range of real-life applications.

3. The vulnerability of a lower version to potential attacks leads to the creation of its higher version variant.

4. The exception of '3' above is the SHA - 3. It was not created with the goal of eliminating SHA-2 but to increase its efficiency. Its construction is quite different from that of SHA-1 andSHA-2. Hence, it is less vulnerable to attack.

5. All SHA - 2 family give rise to the same output word length as opposed to SHA family which give different output word length depending on the version being used.

6. The longer the output word length, the more difficult it is for attackers to hack and the more secure the algorithm will be.

3.0 RESEARCH PROCESS AND METHODOLOGY

The research process and methodology employed in the study involved testing the SHA and MD algorithms using the online hash calculator Tools4noobs. This tool allows for the computation of hash values using various algorithms such as ripemd128, ripemd160, md5, md4, md2, sha512, sha384, sha256, sha224, and sha1. To systematically evaluate the performance of these algorithms, descriptive statistics will be employed to compare the outputlengths of the various hash functions. This statistical tool will reveal differences in output lengths among the different algorithms, where some may generate shorter hash values than others.

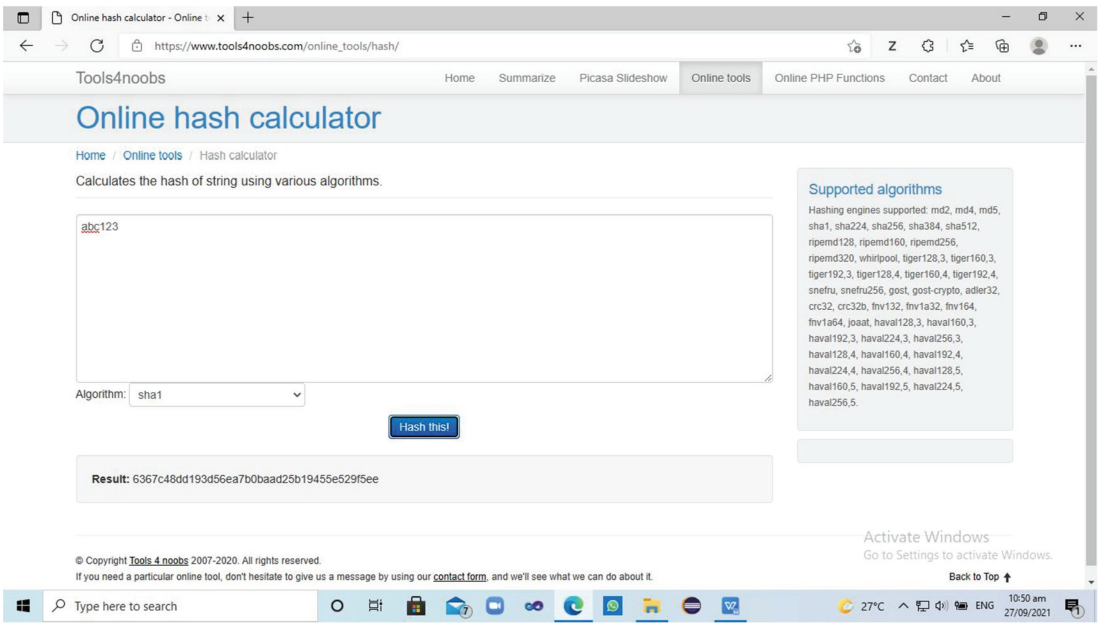
3.1 TESTING SHA AND MD ALGORITHMS USING TOOL4NOOBS

Tools4noobs hash (Figure 1.) is being used to generate outputs from the corresponding input on the various algorithms. i.e., MD and SHA with their variants.

"Tools4Noobs hash" is a hash calculator used for the computation of the string's hash values using a variety of algorithms such as ripemd128 and ripemd160, md5, md4, md2, sha512, sha384, sha256, sha224 and sha1.

Determining the string's hash using a variety of algorithms, including ripemd128 and ripemd160, as well as MD 2, md4, md5, sha1, sha224, sha256, sha384, and sha512

A descriptive statistic was used to present and compare the output lengths for the various hash algorithms.


Tools4noobs

Online hash calculator

Home / Online tools / Hash calculator

Calculates the hash of string using various algorithms.

abc123

Algorithm: sha1

Result: 6367c48dd193d56ea7b0baad25b19455e529f5ee

Supported algorithms

Hashing engines supported: md2, md4, md5, sha1, sha224, sha256, sha384, sha512, ripemd128, ripemd160, ripemd256, ripemd320, whirlpool, tiger128,3, tiger160,3, tiger192,3, tiger128,4, tiger160,4, tiger192,4, snefru, snefru256, gost, gost-crypto, adler32, crc32, crc32b, fnv132, fnv1a32, fnv164, fnv1a64, joaat, haval128,3, haval160,3, haval192,3, haval224,3, haval256,3, haval128,4, haval160,4, haval192,4, haval224,4, haval256,4, haval128,5, haval160,5, haval192,5, haval224,5, haval256,5.

© Copyright Tools 4 noobs 2007-2020. All rights reserved.
If you need a particular online tool, don't hesitate to give us a message by using our contact form, and we'll see what we can do about it.

Figure 1: Tools4Noobs: an online hash calculator

Following are the results of using short and long passwords, i.e., “abc123” was used for short, and “abcdefghijklmnopqrstuvwxyz” was used as a longer password, and the result compared onthe basis of the output lengths for the various algorithms tested. Tools4Noobs, an online Hashalgorithm calculator, was used for the test:

4.0 RESULT AND DISCUSSION

A descriptive statistic was employed to compare the output lengths of the various hash algorithms. This comparison revealed differences in output lengths for different algorithms, with some producing shorter hash values than others. The results obtained from the experiment are presented systematically for each algorithm tested. For instance, when using the MD2 algorithm with the short password "abc123," the hash value "e2cc4f546930ce0fcbce1246159cdb79" was generated, while for the long password "abcdefghijklmnopqrstuvwxyz," the hash value "4e8ddff3650292ab5a4108c3aa47940b" was obtained. As well, the MD2 algorithm yielded a 32-character output for both short and long passwords, while the SHA-512 algorithm produced a significantly longer 128-character output for the short password "abc123." Similar procedures were followed for the MD4, MD5, SHA-1,SHA-224, SHA-256, SHA-384, and SHA-512 algorithms, with corresponding hash values recorded for both short and long passwords.

4.1 The algorithms used are MD5, MD4, MD2, SHA512, SHA384, SHA256, SHA224and Sha1 respectively. INPUT1 is abc123 for all, and INPUT2 is lower-case alphabets a to z for all.

4.1.1. Using MD2 algorithm, a short password i.e., abc123 is being used as the input, and the following result is obtained.

ALGORITHM: MD2
INPUT1: abc123
RESULT1: e2cc4f546930ce0fcbce1246159cdb79

4.1.2. Using MD2 algorithm, a long password, i.e., a-z, is being used as the input, and the following result is obtained.
ALGORITHM: MD2
INPUT2: abcdefghijklmnopqrstuvwxyz
RESULT2:4e8ddff3650292ab5a4108c3aa47940b

4.1.3. Using the MD4 algorithm, a short password, i.e., abc123, is used as the input, and thefollowing result is obtained.

ALGORITHM: MD4
INPUT 1: abc123
RESULT 1:0ceb1fd260c35bd50005341532748de6

4.1.4. Using the MD4 algorithm, a long password, i.e., a-z, is being used as the input, and the following result is obtained.

ALGORITHM: MD4
INPUT2: abcdefghijklmnopqrstuvwxyz RESULT2:
d79e1c308aa5bbcdeea8ed63df412da9

4.1.5. Using MD5 algorithm, a short password i.e., abc123 is being used as the input, and the following result is obtained.

ALGORITHM: MD5
INPUT1: abc123
RESULT1: e99a18c428cb38d5f260853678922e03

4.1.6. Using the MD5 algorithm, a long password, i.e., a-z, is being used as the input, and the following result is obtained.

ALGORITHM: MD5
INPUT2: abcdefghijklmnopqrstuvwxyz RESULT2:
c3fcd3d76192e4007dfb496cca67e13b

4.1.7. Using SHA -1 algorithm, a short password i.e., abc123 is being used as the input, and the following result is obtained.

ALGORITHM: SHA-1
INPUT1: abc123
RESULT1:6367c48dd193d56ea7b0baad25b19455e529f5ee

4.1.8. Using the MD5 algorithm, a long password, i.e., a-z, is being used as the input, and the following result is obtained.

ALGORITHM: SHA-1
INPUT2: abcdefghijklmnopqrstuvwxyz
RESULT2: 32d10c7b8cf96570ca04ce37f2a19d84240d3a89

4.1.9. Using SHA -224 algorithm, a short password i.e., abc123 is being used as the input, andthe following result is obtained.

ALGORITHM: SHA-224
INPUT1: abc123
RESULT1:5c69bb695cc29b93d655e1a4bb5656cda624080d686f74477ea09349

4.1.10. Using the SHA-224 algorithm, a long password, i.e., a-z, is being used as the input, and the following result is obtained.

ALGORITHM: SHA-224
INPUT2: abcdefghijklmnopqrstuvwxyz
RESULT2: 45a5f72c39c5cff2522eb3429799e49e5f44b356ef926bcf390dccc2

4.1.11 Using the SHA -256 algorithm, a short password, i.e., abc123, is being used as the input, andthe following result is obtained.

ALGORITHM: SHA-256
INPUT: abc123
RESULT: 6ca13d52ca70c883e0f0bb101e425a89e8624de51db2d2392593af6a84118090

4.1.12. Using the SHA-256 algorithm, a long password, i.e., a-z, is being used as the input, and the following result is obtained.

ALGORITHM: SHA-256
INPUT 2: abcdefghijklmnopqrstuvwxyz
RESULT 2: 71c480df93d6ae2f1efad1447c66c9525e316218cf51fc8d9ed832f2daf18b73

4.1.13. Using SHA -384 algorithm, a short password i.e., abc123 is being used as the input, andthe following result is obtained.

ALGORITHM: SHA-384
INPUT1: abc123
RESULT1**:**853fb3c290346e14b009d6bfd1d2c28ca30cae71c8662a477b5bf8ec94805858d9fc
cd109754a5e7

4.1.14. Using the SHA-384 algorithm, a long password, i.e., a-z, is being used as the input, and the following result is obtained.
ALGORITHM: SHA-384
INPUT2: abcdefghijklmnopqrstuvwxyz
RESULT2:
feb67349df3db6f5924815d6c3dc133f091809213731fe5c7b5f4999e463479ff2877f5f2936fa6
3bb43784b12f3ebb4

4.1.15. Using the SHA-512 algorithm, a short password, i.e., abc123, is used as the input, andthe following result is obtained.

ALGORITHM: SHA-512
INPUT1: abc123 RESULT1:
c70b5dd9ebfb6f51d09d4132b7170c9d20750a7852f00680f65658f0310e810056e6763c34c9a
00b0e940076f54495c169fc2302cceb312039271c43469507dc

4.1.16. Using the SHA-384 algorithm, a long password, i.e., a-z, is being used as the input, and the following result is obtained.

ALGORITHM: SHA-512
INPUT 2: abcdefghijklmnopqrstuvwxyz
RESULT2: 2:
4dbff86cc2ca1bae1e16468a05cb9881c97f1753bce3619034898faa1aabe429955a1bf8ec483d7
421fe3c1646613a59ed5441fb0f321389f77f48a879c7b1f1

4.2 RESULT SUMMARY

Table 2 below summarises the results of the input using a short password, i.e., abc123

Table 2: Comparison of outputs of algorithms based on word length

Algorithm	**Input**	**Output**	**Number of Character (output)**
MD2	abc123	e2cc4f546930ce0fcbce1246159cdb79	32
MD4	abc123	0ceb1fd260c35bd50005341532748de6	32
MD5	abc123	e99a18c428cb38d5f260853678922e03	32
SHA - 1	abc123	6367c48dd193d56ea7b0baad25b19455e529f5ee	40
SHA - 224	abc123	5c69bb695cc29b93d655e1a4bb5656cda624080d6 86f74477ea09349	56
SHA - 256	abc123	6ca13d52ca70c883e0f0bb101e425a89e8624de51d b2d2392593af6a84118090	64
SHA - 384	abc123	853fb3c290346e14b009d6bfd1d2c28ca30cae71c8 662a477b5bf8ec94805858d9fccd109754a5e7	80
SHA - 512	abc123	c70b5dd9ebfb6f51d09d4132b7170c9d20750a785 2f00680f65658f0310e810056e6763c34c9a00b0e9 40076f54495c169fc2302cceb312039271c434695 07dc	128

In Table 2 above, the same password yielded a variety of outputs depending on the type of algorithm used. MD family has the shortest word count output, while the corresponding SHA family has the longer output. Also, for the SHA family, the higher the version, the longer the output, which obviously means the harder it would be for attackers to be able to crack. SHA -1, for instance, has a 40-character output, while the corresponding SHA 512 (a higher version) has a 128-character output. Table 3 below describes the output of a long input password, i.e., lowercase letters a to z.

Table 3: Comparison of output using longer input**.**

algorithm	**Input**	**Output**	**No. of char (output)**
MD2	Abcdefghijklm nopqrstuvwxyz	4e8ddff3650292ab5a4108c3aa47940b	32
MD4	Abcdefghijklm nopqrstuvwxyz	d79e1c308aa5bbcdeea8ed63df412da9	32
MD5	Abcdefghijklm	c3fcd3d76192e4007dfb496cca67e13b	32

	nopqrstuvwxyz		
SHA - 1	Abcdefghijklm nopqrstuvwxyz	32d10c7b8cf96570ca04ce37f2a19d84240d3a8 9	40
SHA 2- 224	Abcdefghijklm nopqrstuvwxyz	45a5f72c39c5cff2522eb3429799e49e5f44b356 ef926bcf390dccc2	56
SHA 2- 256	Abcdefghijklm nopqrstuvwxyz	71c480df93d6ae2f1efad1447c66c9525e31621 8cf51fc8d9ed832f2daf18b73	64
SHA 2- 384	Abcdefghijklm nopqrstuvwxyz	feb67349df3db6f5924815d6c3dc133f0918092 13731fe5c7b5f4999e463479ff2877f5f2936fa6 3bb43784b12f3ebb4	80
SHA 2- 512	Abcdefghijklm nopqrstuvwxyz	4dbff86cc2ca1bae1e16468a05cb9881c97f1753 bce3619034898faa1aabe429955a1bf8ec483d7 421fe3c1646613a59ed5441fb0f321389f77f48a 879c7b1f1	128
SHA 3- 224	Abcdefghijklm nopqrstuvwxyz	5cdeca81e123f87cad96b9cba999f16f6d41549 608d4e0f4681b8239	56
SHA 3- 256	Abcdefghijklm nopqrstuvwxyz	7cab2dc765e21b241dbc1c255ce620b29f527c6 d5e7f5f843e56288f0d707521	64
SHA 3- 384	Abcdefghijklm nopqrstuvwxyz	fed399d2217aaf4c717ad0c5102c15589e1c990 cc2b9a5029056a7f7485888d6ab65db2370077 a5cadb53fc9280d278f	96
SHA 3- 512	Abcdefghijklm nopqrstuvwxyz	af328d17fa28753a3c9f5cb72e376b90440b96f 0289e5703b729324a975ab384eda565fc92aade d143669900d761861687acdc0a5ffa358bd0571 aaad80aca68	128

From Table 3 above, the input is the 26 lower-case English alphabets used for the test. The outputs, however, only differ in the contents and not in length compared to the correspondingshorter input of abc123 in Table 2. Both SHA-2 512 and SHA-3 512 algorithms produce the same length of output (128 characters) for the given input "Abcdefghijklmnopqrstuvwxyz," thechoice between them would depend on other factors such as cryptographic security, performance, and specific requirements of the application. It's essential to consider factors like the algorithm's resistance to attacks, speed, and suitability for the intended use case to determine the best option. Therefore, without additional information about these aspects, it's challenging to conclusively declare one algorithm as superior to the other solely based on the output length.

4.3 DISCUSSION

The results obtained from the analysis of various hashing algorithms using different input strings present interesting insights into the behavior and characteristics of these algorithms. The study employed a range of hashing algorithms, including MD2, MD4, MD5, SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512. Each algorithm was tested with two different inputs: a short password ("abc123") and a longer one consisting of lowercase English alphabets("a-z").

Output Variations: the Input "abc123" across all algorithms, this short password consistently generated outputs of varying lengths. For instance, MD family algorithms produced 32-character outputs, while SHA family algorithms yielded longer outputs, ranging from 40 to

128 characters, while for Input "a-z" interestingly, when the longer input string was used, the output lengths remained consistent across most algorithms, despite the variation in the contentof the outputs. This consistency suggests that the length of the input string might not always directly correlate with the length of the output hash.

In terms of the MD family vs. SHA Family, the output lengths suggest that the MD family algorithms consistently produced shorter output lengths compared to the SHA family algorithms. This difference is particularly evident when comparing the outputs generated fromthe same input strings. SHA family version and output length, the version impact within the SHA family, a noticeable pattern emerges, higher versions of SHA algorithms tend to produce longer output lengths. For example, SHA-1 generates a 40-character output, whereas SHA-512, a higher version, generates a significantly longer 128-character output.

Security Implications, output length as a security measure, and the length of the output hash is often considered a crucial factor in assessing the security of a hashing algorithm. Longer output lengths generally imply a higher level of security, as they increase the computational complexity for attackers attempting to reverse-engineer the original input from the hash. However, another factor is Resistance to Attacks; while longer output lengths provide a degree of security, it's important to consider other factors, such as the algorithm's resistance to cryptographic attacks. Some algorithms, despite shorter output lengths, may offer robust security features that make them resistant to common attack methods.

Choice of Algorithm, consideration factors when selecting a hashing algorithm for a specific application, developers must consider various factors beyond just output length. These factors may include the algorithm's resistance to attacks, computational efficiency, performance requirements, and compatibility with existing systems.

SHA-2 vs. SHA-3, the comparison between SHA-2 and SHA-3, highlights the importance of evaluating the specific requirements of each application. While both SHA-2 and SHA-3 algorithms produce the same output length for the given input string, other factors, such as resistance to emerging cryptographic attacks and performance characteristics, may influence the choice between them.

5.0 SUMMARY

A review was done on the common techniques being employed to safeguard the integrity and authenticity of documents' digital signatures over cyberspace; a brief overview of MD and SHA algorithms was made. SHA version and its variants seem more productive due to its longer output compared to its MD counterpart. Although the cryptographic hash algorithm offers a fixed length output for any length of input, users are also advised to avoid commonly used easy-to-guess passwords and shorter passwords in order to prevent unauthorised. Other areas of further research include techniques such as salting, adding pepper, and other spices to make a cryptographic hash function more difficult for attackers, even with more sophisticated tools like brute force attacks and so on.

6.0 CONCLUSION

In conclusion, information security could be achieved by developers choosing a more effective and efficient cryptographic hash algorithm to protect their clients' data. The review also revealed that the longer the output length of an algorithm, the harder it is for hackers to penetrate and the more secure the algorithm is said to be. SHA-512 algorithm appears to be more reliable concerning the safety of information and less vulnerable to potentialattacks by intruders.

7.0 REFERENCE

Akinola, A., & Afonja, A. (2022). *Introduction to Cyber-Security*. ChudacePublishing.

Al-Kuwari, S. M. S. A. (2011). Integrated-Key Cryptographic Hash Functions. University ofBath.

Alabrah, A. (2014). Improved Internet Security Protocols Using Cryptographic One-Way HashChains.

Aljuffri, A. A. M. (2024). Securing Power Side Channels by Design.

Amy, M., Di Matteo, O., Gheorghiu, V., Mosca, M., Parent, A., & Schanck, J. (2016). Estimating the cost of generic quantum pre-image attacks on SHA-2 and SHA-3. In *International Conference on Selected Areas in Cryptography* (pp. 317–337). Springer.

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), 181–188.

Aoki, K., & Sasaki, Y. (2009). Meet-in-the-middle preimage attacks against reduced SHA-0 and SHA-1. In *Advances in Cryptology-CRYPTO 2009: 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings* (pp.70–89). Springer.

Aumasson, J.-P., Meier, W., Phan, R. C.-W., & Henzen, L. (2014). The hash function BLAKE.Bertoni, G., Daemen, J., Peeters, M., & Van Assche, G. (2014). The making of KECCAK. *Cryptologia*, *38*(1), 26–60.

Biham, E., & Chen, R. (2004). Near-collisions of SHA-0. In *Advances in Cryptology–CRYPTO2004: 24th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 2004. Proceedings 24* (pp. 290–305). Springer.

Bouillaguet, C. (2011). Algorithms for some hard problems and cryptographic attacks againstspecific cryptographic primitives. Université Paris Diderot (Paris 7).

Chi, L., & Zhu, X. (2017). Hashing techniques: A survey and taxonomy. *ACM Computing Surveys (Csur)*, *50*(1), 1–36.

Dahal, R. K., Bhatta, J., & Dhamala, T. N. (2013). Performance analysis of SHA-2 and SHA-3finalists. *International Journal on Cryptography and Information Security (IJCIS)*, *3*(3), 720–730.

Debnath, S., Chattopadhyay, A., & Dutta, S. (2017). Brief review on journey of secured hash algorithms. In *2017 4th International Conference on Opto-Electronics and Applied Optics (Optronix)* (pp. 1–5). IEEE.

Delfs, H., Knebl, H., Delfs, H., & Knebl, H. (2015). Symmetric-key cryptography. *Introduction to Cryptography: Principles and Applications*, 11–48.

Den Boer, B., & Bosselaers, A. (1991). An attack on the last two rounds of MD4. In *Annual International Cryptology Conference* (pp. 194–203). Springer.

Dobbertin, H. (1997). RIPEMD with two-round compress function is not collision-free. *Journal of Cryptology*, *10*, 51–69.

Dobraunig, C., Eichlseder, M., & Mendel, F. (2015). Security evaluation of sha-224, sha-512/224, and sha-512/256. *Institute for Applied Information Processing and Communications, Graz University of Technology*.

Gauravaram, P. (2007). Cryptographic hash functions: cryptanalysis, design and applications. Queensland University of Technology.

Huang, C.-Y., Ma, S.-P., & Chen, K.-T. (2011). Using one-time passwords to prevent password phishing attacks. *Journal of Network and Computer Applications*, *34*(4), 1292–1301.

Kaliski, B. (1992). RFC1319: The MD2 Message-Digest Algorithm. RFC Editor.

Kamra, S., Sharma, M., & Leekha, A. (2019). Secure hashing algorithms and their comparison.In *2019 6th International Conference on Computing for Sustainable Global Development (INDIACom)* (pp. 788–792). IEEE.

Khan, B. U. I., Olanrewaju, R. F., Morshidi, M. A., Mir, R. N., Kiah, M. L. B. M., & Khan, A. M. (2022). Evolution and Analysis Of Secure Hash Algorithm (Sha) Family. *Malaysian Journal of Computer Science*, *35*(3), 179–200.

Kishore, N., & Raina, P. (2019). Parallel cryptographic hashing: Developments in the last 25 years. *Cryptologia*, *43*(6), 504–535.

Kundu, R., & Dutta, A. (2020). Cryptographic Hash Functions and Attacks-A Detailed Study. *International Journal of Advanced Research in Computer Science*, *11*(2).

Lubin, A. (2022). Cyber plungers: colonial pipeline and the case for an omnibus cybersecurity legislation. *Ga. L. Rev.*, *57*, 1605.

Martino, R., & Cilardo, A. (2020). SHA-2 acceleration meeting the needs of emerging applications: A comparative survey. *IEEE Access*, *8*, 28415–28436.

Merkle, R. C. (1979). *Secrecy, authentication, and public key systems.* Stanford university.

Mihailescu, M. I., Nita, S. L., Mihailescu, M. I., & Nita, S. L. (2021). Hash functions. *Cryptography and Cryptanalysis in MATLAB: Creating and Programming Advanced Algorithms*, 83–102.

Milton, L. (2022). The History and Implications of Cyberwarfare for Corporations, Governments, People, and Society: Case Studies Examining How Individuals Create Vulnerabilities in the Information Security Field.

Muller, F. (2004). The MD2 hash function is not one-way. In *International Conference on theTheory and Application of Cryptology and Information Security* (pp. 214–229). Springer. Nechvatal, J., Barker, E., Bassham, L., Burr, W., Dworkin, M., Foti, J., & Roback, E. (2001).Report on the development of the Advanced Encryption Standard (AES). *Journal of Research of the National Institute of Standards and Technology*, *106*(3), 511.

Neforawati, I., & Arnaldy, D. (2021). Message Digest 5 (MD-5) Decryption Application usingPython-Based Dictionary Attack Technique. In *2021 4th International Conference of Computer and Informatics Engineering (IC2IE)* (pp. 424–428). IEEE.

No, W. G., & Vasarhelyi, M. A. (2017). Cybersecurity and continuous assurance. *Journal of Emerging Technologies in Accounting*, *14*(1), 1–12.

Obaida, T. H., Salman, H. A., & Zugair, H. N. (2022). Improve MD5 Hash Function For Document Authentication. *Webology (ISSN: 1735-188X)*, *19*(1).

Plachkinova, M., & Maurer, C. (2018). Security breach at target. *Journal of Information Systems Education*, *29*(1), 11–20.

Preneel, B. (2010). The first 30 years of cryptographic hash functions and the NIST SHA-3 competition. In *Cryptographers' track at the RSA conference* (pp. 1–14). Springer.

Primmer, R., & D'Halluin, C. (2013). Collision and preimage resistance of the Centera contentaddress. *ArXiv Preprint ArXiv:1306.6020.*

Radack, S. (n.d.). Itl Bulletin for May 2012 Secure Hash Standard: Updated Specifications Approved And Issued As Federal Information Processing Standard (Fips) 180-4.

Rivest, R, & Group, M. message-digest algorithm N. W. (1992). MIT Laboratory for ComputerScience and RSA Data Security. *Inc., RFC1321.*

Rivest, Ronald. (1992). *The MD5 message-digest algorithm.*

Rizzardi, A., Sicari, S., & Coen-Porisini, A. (2022). Analysis on functionalities and security features of Internet of Things related protocols. *Wireless Networks*, *28*(7), 2857–2887.

Rogier, N., & Chauvaud, P. (1995). The compression function of MD2 is not collision free. In *Selected Areas in Cryptography* (Vol. 95, pp. 18–19).

Sadeghi-Nasab, A., & Rafe, V. (2023). A comprehensive review of the security flaws of hashing algorithms. *Journal of Computer Virology and Hacking Techniques*, *19*(2), 287–302.

Sahni, N. (2015). A review on cryptographic hashing algorithms for message authentication. *International Journal of Computer Applications*, *120*(16).

Srivastava, V., Baksi, A., & Debnath, S. K. (2023). An overview of hash based signatures. *Cryptology EPrint Archive*.

Stevens, M., Bursztein, E., Karpman, P., Albertini, A., & Markov, Y. (2017). The first collisionfor full SHA-1. In *Advances in Cryptology–CRYPTO 2017: 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20–24, 2017, Proceedings, Part I 37* (pp. 570–596). Springer.

Stoddart, K. (2022). On Cyberwar: Theorizing Cyberwarfare Through Attacks on Critical Infrastructure—Reality, Potential, and Debates. In *Cyberwarfare: Threats to Critical Infrastructure* (pp. 53–146). Springer.

Sudha, M., & Monica, M. (2012). Enhanced security framework to ensure data security in cloud computing using cryptography. *Advances in Computer Science and Its Applications*,*1*(1), 32–37.

Swinford, S., Huesken, B., Mierow, A., Ariyaratne, H., & Ismail, S. (2023). Computer Crimes. *Am. Crim. L. Rev.*, *60*, 579.

Thakur, M. (2024). Cyber Security Threats and Countermeasures in Digital Age. *Journal of Applied Science and Education (JASE)*, 1–20.

Thomas, K., Li, F., Zand, A., Barrett, J., Ranieri, J., Invernizzi, L., … Moscicki, A. (2017). Data breaches, phishing, or malware? Understanding the risks of stolen credentials. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security* (pp. 1421–1434).

Trautman, L. J. (2018). How Google Perceives Customer Privacy, Cyber, E-Commerce, Political and Regulatory Compliance Risks. *Wm. & Mary Bus. L. Rev.*, *10*, 1.

Trautman, L. J., Shackelford, S., Elzweig, B., & Ormerod, P. (2024). Understanding Cyber Risk: Unpacking and Responding to Cyber Threats Facing the Public and Private Sectors.*University of Miami Law Review*, *78*(3), 840.

Vasić, V., Kukec, A., & Mikuc, M. (2011). Deploying new hash algorithms in secure neighbor discovery. In *SoftCOM 2011, 19th International Conference on Software, Telecommunications and Computer Networks* (pp. 1–5). IEEE.

Wang, X., Feng, D., Lai, X., & Yu, H. (2004). Collisions for hash functions MD4, MD5, HAVAL-128 and RIPEMD. *Cryptology EPrint Archive*.

Wang, X., & Yu, H. (2005). How to break MD5 and other hash functions. In *Annual international conference on the theory and applications of cryptographic techniques* (pp.19–35). Springer.

Xu, L. (2020). Design and Implementation of a Credible Blockchain-based E-health Records Platform.

Zhou, T., Zhu, Y., Jing, N., Nan, T., Li, W., & Peng, B. (2020). Reliable SoC Design and Implementation of SHA-3-HMAC Algorithm with Attack Protection. In *2020 IEEE International Conference on Smart Cloud (SmartCloud)* (pp. 88–93). IEEE.