A New Approach in Expanding the Hash Size of MD5

Esmael V. Maliberan, Ariel M. Sison, Ruji P. Medina

Graduate Programs, Technological Institute of the Philippines, Quezon City, Philippines

Abstract: The enhanced MD5 algorithm has been developed by expanding its hash value up to 1280 bits from the original size of 128 bit using XOR and AND operators. Findings revealed that the hash value of the modified algorithm was not cracked or hacked during the experiment and testing using powerful bruteforce, dictionary, cracking tools and rainbow table such as CrackingStation, Hash Cracker, Cain and Abel and Rainbow Crack which are available online thus improved its security level compared to the original MD5. Furthermore, the proposed method could output a hash value with 1280 bits with only 10.9 ms additional execution time from MD5.

Keywords*:* MD5 algorithm, hashing, client-server communication, modified MD5, hacking, bruteforce, rainbow table.

1. Introduction

Security has always been a concern in the realm of Information Technology, especially in cloud environment [1,24,20,7,9]. One of the primary roles in Information Security is Cryptographic hashes [10]. Hash Algorithm, which is also called message digest algorithm is being utilized to produce distinctive message digest for a random message [13]. Hashing algorithms are imperative elements in several cryptographic claims and security practices [19. Hashing has the property of being one way, and because of this property, they have major usage in providing message integrity and storing passwords in operating systems. Because of this, hash algorithms are used widely especially in login authentication and verifying message integrity [13,28,8].

With the rapid growth of internet, more communication applications, such as electronic mail or the use of World Wide Web browsers are receiving information [42]. While there is a huge amount of transferring data in the cloud system, the risk of accessing data by attackers raises [14]. Hence, there have been a lot of approaches and techniques in securing the client-server communication and other related online services and transactions. According to [33], the use of Short Message Service (SMS)-based One Time Password (OTP) is one of the most commonly used multi-factor authentication and authorization mechanism to address security issues on various online services. Another approach is the use of Radio Frequency Identification (RFID) technology for authentication. An enhancement was even proposed by [34] which utilized an efficient hash protocol to improve its security level. Among these security measures, hash algorithms are still widely used in cryptographic protocols and Internet communication in general. Several widely used hash algorithms exist. One of the most famous is the MD5 message digest algorithm developed by Ronald Rivest which is an improved adaptation of the MD4 algorithm. Other common algorithms are SHA-1 and its variants and RIPEMD-160. These hash algorithms are used widely in cryptographic protocols and internet communication in general. Among several hashing algorithms mentioned above, MD5 still surpasses the other since it is still widely used in the domain authentication security owing to its feature of irreversible [41]. This only means that the confirmation does not need to demand the original data but only need to have an effective digest to confirm the identity of the client. The MD5 message digest algorithm was developed by Ronald Rivest sometime in 1991 to change a previous hash function MD4, and it is commonly used in securing data in various applications [27,23,22]. It allows accepting any message input regardless of its length and generates a 128–bit hash value. However, MD5 is not perfect. The hashed value generated by the MD5 algorithm is not secured because its length is too short which can easily be hacked using brute force and rainbow table. Thus there is a severe trade-off regarding the security of the MD5 with its flaws having been broken in the field, most notoriously by the Flame malware in 2012. It was hacked using brute force attacks and rainbow tables. According to [5], a brute force attack is an approach which is associated with thoroughly inspecting all available keys until the appropriate one is achieved. On the other hand, a rainbow table [12] is a pre-calculated table for inversing hash functions usually for finding password hashes.

Because of these, there had been several modifications made in the algorithm to address its security issues [25,18,26,29,2] since MD5 suffers from the above-mentioned attacks due to its undersized hash value. For instance, in the study of [3], the author modified the MD5 algorithm by combining the SHA compression function to the algorithm and increase the hash code length of the MD5 up to 256-bit against collision attacks. Another modification made by [15] on MD5 by encrypting the password before it is being broadcast through a network using the six reserved bits of a TCP header. The technique of encryption which enhances the security of MD5 hashed password as Hashed Password Encryption (HPE) was used against brute force attacks and rainbow table. [4] Developed a unified architecture to modify the MD5 algorithm in improving its security level to prevent brute force attacks and rainbow table. [6] Enhanced the MD5 algorithm by applying multi techniques such as DNA coding, non- Linear Feedback Shift Register (NLFSR), and Logistic function of Chaos theory. It will expand of input MD5 algorithm to 1024 bits instead of 512 bits, and generates a 160 bits output instead of 128 bit. The modified MD5 algorithm can predict explicit enumeration through brute force Attack.

While there were modifications made by several researchers, however, no single modification is yet perfectly proven effective to solve that one and therefore there remains a challenge to address the problem against MD5 attacks. Hence this study will modify MD5 hash algorithm by using a new approach of hashing the password and extending its hash value to 1280 bit size with the use of XOR and AND operator. Furthermore, the extended hash value is proposed to prevent generic attacks such as brute force, dictionary attacks and rainbow table.

2. Literature Review

There had been many types of research made using the MD5 algorithm to secure client-server communication, particularly in a cloud environment. Some of these papers have been proposed to discover the flaws of different hash algorithms particularly MD5 [17]. Thus, several types of research have been proposed to enhance and modify the MD5 algorithm to increase its security level and performance.

Modifications to MD5 Algorithm

According to [35], A 128-bit hash size output that is generated by the hash function is not extremely safe for some applications. Therefore, to offer more security, an algorithm that is MD5-512 has been proposed. MD5-512 works on the similar notion as MD5. It accepts the 512-bit input as that of MD5 and generates a 512-bit hash size output. The authors designed an enhanced MD5 algorithm that produces 640 bits. In this proposed algorithm, they divided 640 bits into 5 equal parts so that each block holds 128 bits and firstly execute operations on these 128 bits of each part and then combined the output of all these blocks to produce 640-bits hash value.

According to [36], their paper aims to propose and develop an enhanced MD5 algorithm for the secure web development. The MD5 algorithm has many flaws that make it susceptible to different generic attacks such as brute force, rainbow table, birthday, dictionary, etc. In spite of these, the MD5 algorithm is still employed in many web applications, login authentication, and security protocols and even in the diffusion and storage of digital data for confirmation, reliability, and safety of data with the use of a checksum. The research focuses on justifying the flaws inherent from the existing MD5 algorithm to protect web application and sustain data security and integrity. The research records the proposal and completion of an enhanced MD5 algorithm by changing its size and using a key to hash the message into its cipher form.

According to [37], the MD5 algorithm is an algorithm that is capable of converting the randomly dissimilar size of messages into a 128-bit hash value. Every message digest corresponding to the message is distinctive, and it is hard to compute the message itself by digest. It particularly complicates the understanding of the principle and the security of the MD5 algorithm. For the brute-force and collision attack, enhancement and execution of the MD5 algorithm in user security application is proposed. Altering the structure of the plaintext and ciphertext can guarantee the protection of the user.

[38] Proposed a novel algorithm which is an enhancement of the current MD5 algorithm to conquer the various generic attacks such as brute force, dictionary and rainbow table on MD5. In the proposed algorithm, the authors modified MD5 by utilizing the notion of Permutation Boxes and SALT. Once the current algorithm is enhanced, they obtained a new algorithm that generates a safer and complicated hash value as compared to the original one.

According to [39], the purpose of technical and related measures to maintain the protection of diverse documents while transferring on the medium is significant accountability in electronic data systems. Their study identifies the modification of the MD5 to protect perceptive data. Security of data during broadcast or while in a database may be essential to keep the integrity and confidentiality of data. The proposed algorithm exclusively identifies the algebraic steps necessary to convert data into a cryptographic code and also to convert the code back to its original form.

[40] Proposed a modified MD5 which generates an output of 512-bit to further enhance the security level during login authentication and sending messages in 3G and 4G network. The authors define eight-bit operation functions of variable J, K, L, M, N, O, P and Q respectively, in which x, y, z are three 32-bit integers. The generated output is a 512-bit hash message. With this, the sensitive data like passwords can be shared with the peer. There is another application of the proposed algorithm which is Message Authentication Code (MAC). This is a reliability check method based on cryptographic hash functions using a secret key.

[3] Proposed a novel enhanced MD5 Algorithm combined with SHA Compression Function that generates a 256-bit hash code. This modification was used to prevent brute force and rainbow table and birthday attacks. Their approach was to expand the hash size to 256–bit from its original size of 128-bit by expanding the compression function block size which will be applied in data reliability and signing applications. Complicated message enhancement approaches were used to attain the essential situations on the chaining variables and the message bits. Findings revealed that it is resistant to local collision and differential attack.

According to [15], hashed password occurs when it is encrypted using a hash algorithm. These hashed passwords are called data packet which passes through the internet. The authors proposed a novel technique to improve the safety of these hashed passwords by employing the six reserved bits of a TCP header before transmitting it through the internet. By applying the improved MD5 approach will diminish the risk of identifying the original password. Thus it will be impossible for the attacker to pre-calculate the hash for rainbow table and dictionary when Hash Password Encryption (HPE) is employed in the system. Furthermore, HPE makes an excellent utilization of six reserved bits in TCP header which entails proficient usage of network resources. Hence, the transmission of unused bits was decreased.

[16] Introduced a novel method of using MD5 by utilizing outside information, a computed salt and a random key to hash the password before the MD5 computation. By applying a key stretching method combined with XOR cipher in the final hash value, it will be impossible for hackers to attack using a rainbow table.

In the study of [11], the authors proposed the optimized MD5 function for the password recovery of an encrypted PDF file. The cracking performance was improved by reducing the number of instructions to be executed during the hashing operation in the recovery process.

[21] Proposed an improved hash algorithm based on the MD5 and SHA-256 that can be utilized in any signing

application and message integrity where its hash value was expanded to 256 bits. It revealed that the difficulty of the new algorithm is greater than that of MD5 and SHA-256.
Meanwhile, [4] proposed a new method for generating a hash value of any message. An evaluation was done between their proposed optimized algorithm and the existing cryptographic hashing tool MD5. Based on the proposed approach, the time lapsed to make a hash function from the original text is lesser than the MD5 hashing tool.

3. System Architecture of the Modified MD5

3.1 Expanding the hash size of MD5

The output of the MD5 algorithm which is 128 bit will be converted into hexadecimal value comprising of 32 characters. A function will be called to convert this value into decimal and allocate 16 blocks of an array consisting of 1 byte per block. There will be two (2) characters assigned and allowed for every block. Another function is created to allocate 160 blocks of an array of size 1 Byte.

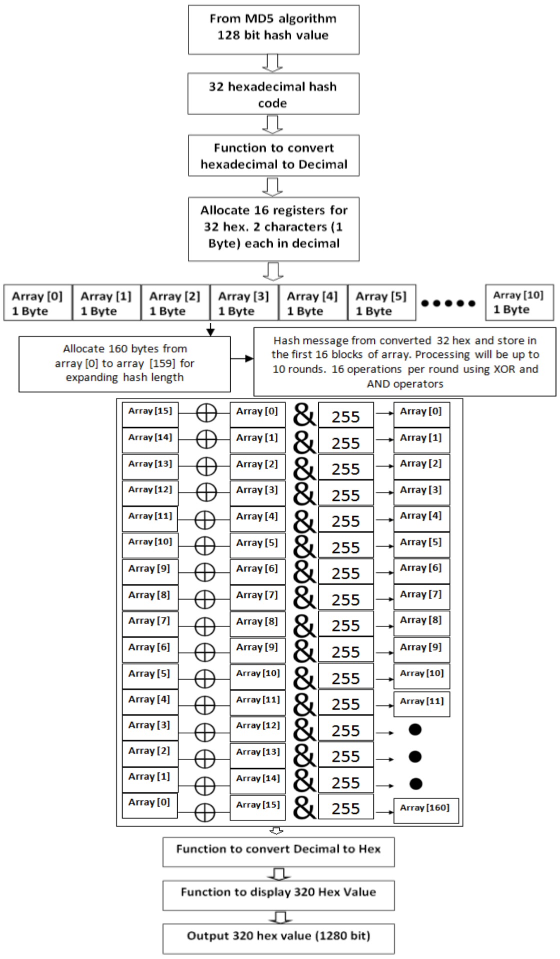

From MD5 algorithm 128 bit hash value
32 hexadecimal hash code
Function to convert hexadecimal to Decimal
Allocate 16 registers for 32 hex. 2 characters (1 Byte) each in decimal
Array [0] 1 Byte
Array [1] 1 Byte
Array [2] 1 Byte
Array [3] 1 Byte
Array [4] 1 Byte
Array [5] 1 Byte
●●●●●
Array [10] 1 Byte
Allocate 160 bytes from array [0] to array [159] for expanding hash length
Hash message from converted 32 hex and store in the first 16 blocks of array. Processing will be up to 10 rounds. 16 operations per round using XOR and AND operators
Array [15] ⊕ Array [0] & 255 → Array [0]
Array [14] ⊕ Array [1] & 255 → Array [1]
Array [13] ⊕ Array [2] & 255 → Array [2]
Array [12] ⊕ Array [3] & 255 → Array [3]
Array [11] ⊕ Array [4] & 255 → Array [4]
Array [10] ⊕ Array [5] & 255 → Array [5]
Array [9] ⊕ Array [6] & 255 → Array [6]
Array [8] ⊕ Array [7] & 255 → Array [7]
Array [7] ⊕ Array [8] & 255 → Array [8]
Array [6] ⊕ Array [9] & 255 → Array [9]
Array [5] ⊕ Array [10] & 255 → Array [10]
Array [4] ⊕ Array [11] & 255 → Array [11]
Array [3] ⊕ Array [12] & 255 → ●
Array [2] ⊕ Array [13] & 255 → ●
Array [1] ⊕ Array [14] & 255 → ●
Array [0] ⊕ Array [15] & 255 → Array [160]
Function to convert Decimal to Hex
Function to display 320 Hex Value
Output 320 hex value (1280 bit)

Figure 1. Operational Framework of the Study

It is intended for expanding the hash length to 160 bytes or 1280 bit as its target output. During this phase, the message will be hashed using XOR and AND operators and store the hashed message per 16 blocks of an array. XOR was used in the design because it has the best bit shuffling properties [30, 31]. AND operator was used to 255 decimal or FF in Hexadecimal hence it is the maximum value of 1 byte needed to be stored in an array. Processing will be up to ten (10) rounds and 16 operations per round. Another function will be called to convert decimal back to hexadecimal and finally outputs the 320 hexadecimal hash value or 1280 bit hash value.

3.2 Properties of a Hash Algorithm

Pre-image resistant: for any given h, computationally infeasible to find y such that H(y)=*h* while in second pre-image resistant, for any given x, it is computationally infeasible to find y ≠ x with H(y) = H(x). For strong collision resistant, it is infeasible to find a pair (x,y) such that H(x) = H(y).
These properties of a hash algorithm must be satisfied by the proposed algorithm. According to [32], for a given *n*-bit hash code, pre-image is calculated by the formula 2^n where n is the number of bits. Moreover, strong collision is calculated by the formula $2^{n/2}$ where n is the number of bits output of a hash algorithm.

Table 1 presents the comparison of the modified MD5 to the MD5 in calculating pre-images attack, second pre-image attack, and strong collision attack.

Name of Attacks	MD5 (128-bit output)	Modified MD5 (1280-bit output)
Pre-image attack	2^{128}	2^{1280}
Second Pre-image attack	2^{128}	2^{1280}
Strong Collision attack	2^{64}	2^{640}

As shown in Table 1, it would take 2^{128} permutations before a pre-image attack is calculated using MD5 while it would take 2^{1280} permutations for the proposed algorithm. Second pre-image attack would take 2^{1280} permutations, and strong collision attack would take 2^{640} permutations for the proposed algorithm. This means that the longer the length of the hash code output is, the more operations and permutations the attacker needs to perform to these attacks.

Table 2. Usage of the properties of a Hash Algorithm

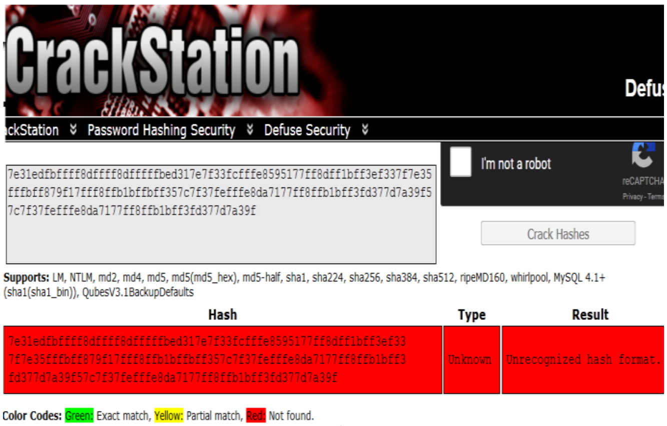
Applications	Pre-image Resistant	Second Pre-image Resistant	Collision Resistant
Hash + digital signature	Yes	Yes	Yes
Intrusion detection and virus detection		Yes	
Hash + symmetric encryption			
One-way password file	Yes		
MAC	Yes	Yes	Yes

4. Testing the reliability of the modified MD5

All generic attacks must be tested to measure its performance and reliability. To test the credibility of the modified hash algorithm, several attacks and tools have been used and tested.

CrackStation one of the powerful online cracking tools, utilizes huge pre-computed lookup tables to break password hashes. These tables store a plot between the password hash and the exact password for that particular hash. The hash codes are listed and indexed for searching the database immediately for a given hash. If the hash is found, the password can be obtained in a split of a second. This is applied only to hashes without salt and supports current hash algorithms such as MD4, MD5, sha256, sha512, etc.

As shown in Figure 2, the modified MD5 algorithm with an output of 1280 bit hash value was tested using the tool. The 1280 hash value will be searched in the databases of existing hash algorithms. The tool is configured to find a possible match. Although CrackStation accepted the hash code as input, however, it could not generate the equivalent plaintext value of the hash code in its database, and the result is the unrecognized hash format. It can be noted that the color code is red which means that the hash value is the unrecognized hash format and its type is unknown or not found in their database. It is because the tool will work only on existing known hashing algorithm such as MD5, sha256, sha512, MD160, etc.



```
7e31edfbffff8dffff8dfffffbed317e7f33fcfffe8595177ff8dff1bff3ef337f7e35
fffbff879f17fff8ffb1bffbff357c7f37feffffe8da7177ff8ffb1bff3fd377d7a39f5
7c7f37feffffe8da7177ff8ffb1bff3fd377d7a39f
```

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults

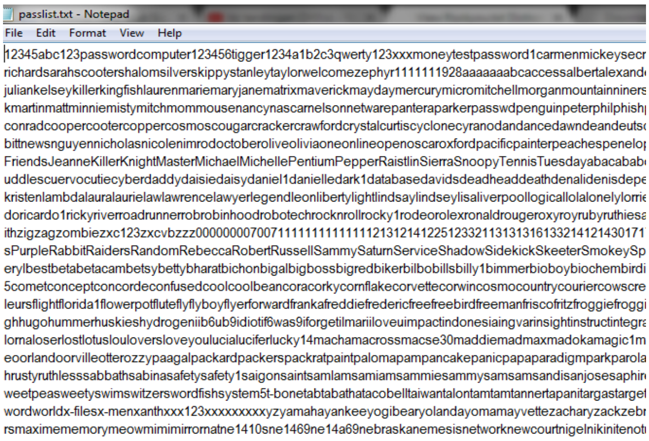
Hash	Type	Result
7e31edfbffff8dffff8dfffffbed317e7f33fcfffe8595177ff8dff1bff3ef337f7e35fffbff879f17fff8ffb1bffbff357c7f37feffffe8da7177ff8ffb1bff3fd377d7a39f57c7f37feffffe8da7177ff8ffb1bff3fd377d7a39f	Unknown	Unrecognized hash format.

Color Codes: 🟩 Green: Exact match, 🟨 Yellow: Partial match, 🟥 Red: Not found.

Figure 2. Cracking the Modified MD5 using CrackStation

4.1 Testing a Dictionary Attack

A dictionary attack is another form of attack in hashing algorithm. A dictionary attack tries to conquer an authentication mechanism by analytically using every word in a *dictionary* as the password.

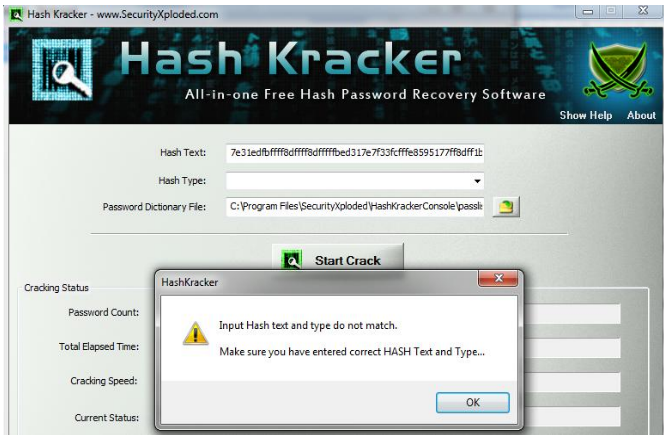


```
12345abc123passwordcomputer123456tigger1234a1b2c3qwerty123xxxmoneytestpassword1carmenmickeysecr
richardsarahscootershalomsilverskippystanleytaylorwelcomezephyr111111928aaaaaabcaccessalbertalexand
juliankelseykillerkingfishlaurenmariemaryjanematrixmaverickmaydaymercurymicromitchellmorganmountainniners
kmartinmattminniemistymitchmommousenancynascarnelsonnetwarepanteraparkerpasswdpenguinpeterphilphish
conradcoopercootercoppercosmoscougarcrackercrawfordcrystalcurtiscyclonecyranodandancedawndeandeuts
bittnewsnguyennicholasnicolenimrodoctoberoliveoliviaoneonlineopenoscaroxfordpacificpainterpeachespenelo
FriendsJeanneKillerKnightMasterMichaelMichellePentiumPepperRaistlinSierraSnoopyTennisTuesdayabacabab
uddlescuervocutiecyberdaddydaisiedaisydaniel1danielledark1databasedavidsdeadheaddeathdenalidenisdepe
kristenlambdalauralaurielawlawrencelawyerlegendleonlibertylightlindsaylindseylisaliverpoollogicallolalonelylorrie
doricardo1rickyriverroadrunnerrobrobinhoodrobotechrocknrollrocky1rodeorolexronaldrougeroxyroyrubyruthiesa
ithzigzagzombiezxc123zxcvbzzz000000007007111111111111121312141225123321131313161332141214301717
sPurpleRabbitRaidersRandomRebeccaRobertRussellSammySaturnServiceShadowSidekickSkeeterSmokeySp
erylbestbetabetacambetsybettybharatbichonbigalbigbossbigredbikerbilbobillsbilly1bimmerbioboybiochembirdi
5cometconceptconcordeconfusedcoolcoolbeancoracorkycornflakecorvettecorwincosmocountrycouriercowscre
leursflightflorida1flowerpotfluteflyflyboyflyerforwardfrankafreddiefredericfreefreebirdfreemanfriscofritzfroggiefroggi
ghhugohummerhuskieshydrogeniib6ub9idiotif6was9iforgetilmariiloveuimpactindonesiaingvarinsightinstructintegra
lornaloserlostlotuslouloversloveyoulucialuciferlucky14machamacrossmacse30maddiemadmaxmadokamagic1m
eoorlandoorvilleotterozzypaagalpackardpackerspackratpaintpalomapampancakepanicpapaparadigmparkparola
hrustyruthlesssabbathsabinasafetysafety1saigonsaintsamlamsamiamsammiesammysamsamsandisanjosesaphir
weetpeasweetyswimswitzerswordfishsystem5t-bonetabtabathatacobelltaiwantalontamtamtannertapanitargastarget
wordworldx-filesx-menxanthxxx123xxxxxxxxxyzyamahayankeeyogibearyolandayomamayvettezacharyzackzebr
rsmaximememorymeowmimimirrornatne1410sne1469ne14a69nebraskanemesisnetworknewcourtnigelnikinitenot
```

Figure 3. A dictionary that contains possible passwords

Hash Kracker is a simple-to-use software program that can calculate and reveal hash passwords from hash text using several algorithms. It offers support for MD5, SHA1, SHA256, SHA384, and SHA512. The tool used a dictionary named passlist.txt as shown in figure 3 to lookup for possible passwords that will match the hash code.

The tool accepted the 1280 bit hash value of the modified MD5, but a message appears which says "Input hash text and type do not match. Make sure you have entered correct HASH Text and Type. It is shown in figure 4.



Hash Text: 7e31edfbffff8dffff8dfffffbed317e7f33fcfffe8595177ff8dff1b

Hash Type:

Password Dictionary File: C:\Program Files\SecurityXploded\HashKrackerConsole\passli

Input Hash text and type do not match.

Make sure you have entered correct HASH Text and Type...

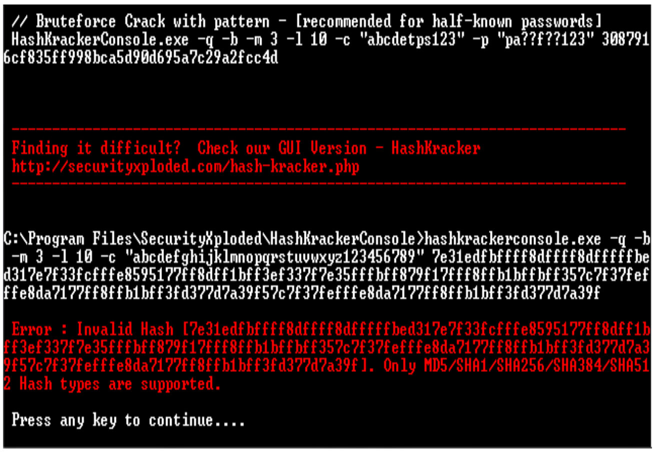
Figure 4. Using Hash Kracker tool to perform a dictionary attack on the Modified MD5

Again the test of Dictionary attack on the modified MD5 failed because it is only lookup to the tables of existing hash algorithms such as SHA1, SHA256, SHA512, MD5, MD160, etc.

4.2 Testing a brute force attack

Another common attack in a hash algorithm is brute force attack. A brute force attack is guessing and a trial-and-error way to acquire information such as personal identification number (PIN) or a password. In this type of attack, a system is used to produce a huge number of successive guesses regarding the content of the preferred information. The researchers used the hashkracker console program to perform this attack to measure its reliability. Hash Kracker Console is the all-in-one command-line tool to find out the password from the Hash. Currently, it supports password recovery from following popular Hash types such as MD5, SHA1, SHA256, SHA384, and SHA512.

As shown in figure 5, this attack failed in the modified MD5 because it cannot recognize the type of the hash value of the modified MD5. It is impossible to brute force on a certain kind of hash value if the attacker does not know its source code or how the modified MD5 was being developed. Furthermore, the attacker must first be familiar with the source code of the modified MD5 before he can perform brute force attack.

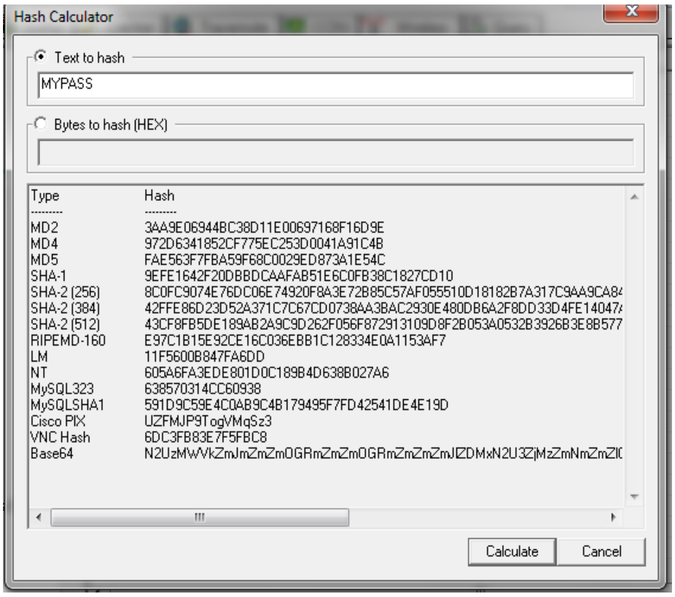


```
 // Bruteforce Crack with pattern - [recommended for half-known passwords]
 HashKrackerConsole.exe -q -b -m 3 -l 10 -c "abcdetps123" -p "pa??f??123" 308791
6cf835ff998bca5d90d695a7c29a2fcc4d



 ---------------------------------------------------------------------------
 Finding it difficult?  Check our GUI Version - HashKracker
 http://securityxploded.com/hash-kracker.php
 ---------------------------------------------------------------------------


C:\Program Files\SecurityXploded\HashKrackerConsole>hashkrackerconsole.exe -q -b
 -m 3 -l 10 -c "abcdefghijklmnopqrstuvwxyz123456789" 7e31edfbffff8dffff8dfffffbe
d317e7f33fcfffe8595177ff8dff1bff3ef337f7e35fffbff879f17fff8ffb1bffbff357c7f37fef
ffe8da7177ff8ffb1bff3fd377d7a39f57c7f37feffffe8da7177ff8ffb1bff3fd377d7a39f

 Error : Invalid Hash [7e31edfbffff8dffff8dfffffbed317e7f33fcfffe8595177ff8dff1b
ff3ef337f7e35fffbff879f17fff8ffb1bffbff357c7f37feffffe8da7177ff8ffb1bff3fd377d7a3
9f57c7f37feffffe8da7177ff8ffb1bff3fd377d7a39f]. Only MD5/SHA1/SHA256/SHA384/SHA51
2 Hash types are supported.

 Press any key to continue....
```

Figure 5. Using HashKracker Console to perform brute force attack on the Modified MD5

4.3 Testing using a hash calculator

Another tool was tested to test the credibility of the Modified MD5 using Cain and Abel password cracking tool. This tool is a password recovery tool for Microsoft Windows. It can recover many kinds of passwords using methods such as network packet sniffing, cracking various password hashes by using techniques such as dictionary attacks, brute force and cryptanalysis attacks as shown in figure 6. When the word "MYPASS" was converted into its equivalent hash value, findings revealed that it produces a hash value of "FAE563F7FBA59F68C0029ED873A1E54C" which is different from the hash value of the modified MD5.

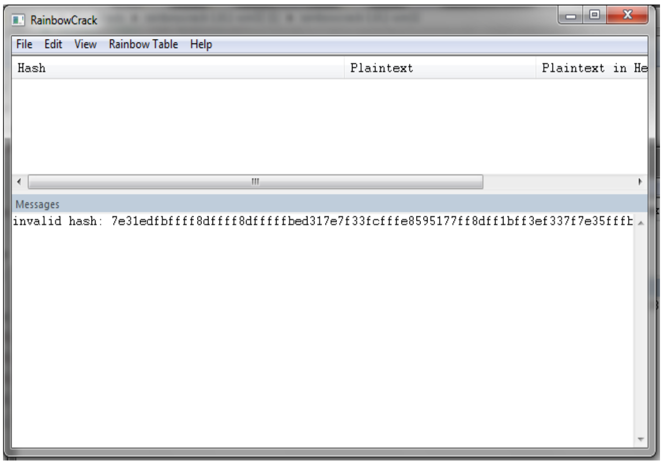


```
MYPASS

Type            Hash
---------       ---------
MD2             3AA9E06944BC38D11E00697168F16D9E
MD4             972D6341852CF775EC253D0041A91C4B
MD5             FAE563F7FBA59F68C0029ED873A1E54C
SHA-1           9EFE1642F20DBBDCAAFAB51E6C0FB38C1827CD10
SHA-2 (256)     8C0FC9074E76DC06E74920F8A3E72B85C57AF055510D18182B7A317C9AA9CA8
SHA-2 (384)     42FFE86D23D52A371C7C67CD0738AA3BAC2930E480DB6A2F8DD33D4FE14047
SHA-2 (512)     43CF8FB5DE189AB2A9C9D262F056F872913109D8F2B053A0532B3926B3E8B577
RIPEMD-160      E97C1B15E92CE16C036EBB1C128334E0A1153AF7
LM              11F5600B847FA6DD
NT              605A6FA3EDE801D0C189B4D638B027A6
MySQL323        638570314CC60938
MySQLSHA1       591D9C59E4C0AB9C4B179495F7FD42541DE4E19D
Cisco PIX       UZFMJP9TogVMqSz3
VNC Hash        6DC3FB83E7F5FBC8
Base64          N2UzMWVkZmJmZmZmOGRmZmZmOGRmZmZmZmJlZDMxN2U3ZjMzZmNmZmZl
```

Figure 6. Using Hash Calculator to convert the hash value of the word "MYPASS" in other hash algorithms

4.4 Testing a Rainbow Table Attack

Rainbow Table attack utilizes a pre-computed rainbow table. It consists of a database that has a large number of a hash function's input and equivalent output.



```
invalid hash: 7e31edfbffff8dffff8dfffffbed317e7f33fcfffe8595177ff8dff1bff3ef337f7e35fffb
```

Figure 7. Testing for rainbow table attack using Rainbow Crack

Its function is just to search and compare a password and its equivalent hash value inside the table. As shown in figure 7, a rainbow table attack is tested using RainbowCrack software. Results showed that the attack failed to crack the hash value of "MYPASS" which is 1280-bit generated from the modified MD5. This is because the database only contains hashes of the existing hash algorithms such as MD5, SHA2, SHA256, SHA512, etc.

4.5 Measuring the Performance of the Modified MD5

The simulation was done to test the time execution of the proposed algorithm using Intel Celeron @ 2.16 GHz, 2G DDR3 computer. The modified MD5 algorithm was written using C Language. The comparison of the time execution and hash size length of different hash algorithms and the modified MD5 algorithm of hashing the word MYPASS (6 bytes) are presented in table 3. It has been observed that the average execution time using the proposed algorithm during five attempts is 0.02975 seconds which produces 320 digits (1280-bit) long hash value. Even if the size of the MD5 was extended, still its performance is compelling with only additional 10.9 ms execution time from the original MD5.

Table 3 Comparison of the time execution of different hash algorithms and the Proposed MD5 Algorithm

Hash Algorithm	Time Elapse	Hash size length
MD5	0.01877 sec	32 digits
Proposed Algorithm	0.02975 sec	320 digits
SHA1	1.59740 E-5 sec	40 digits
SHA256	1.50203 E-5 sec	64 digits
SHA384	1.28746 E-5 sec	96 digits
SHA512	1.21593 E-5 sec	128 digits

5. Conclusion and Recommendation

In this paper, a new approach that expands the hash size of the MD5 algorithm to further secure against known hashing attacks. The new notion applied the combination of XOR and AND operators to produce a 1280 bit hash size to prevent generic attacks. The proposed modification also presents an avenue to explore to other hashing and expansion techniques that will require uncomplicated mathematical calculations.

References

[1] A. Arora, A. Rastogi, A. Khanna and A. Agarwal, "Cloud Security Ecosystem for Data Security and Privacy", 7th International Conference on Cloud Computing, Data Science& Engineering – Confluence, 2017.

[2] A. Bhandari, "Enhancement of MD5 Algorithm for Secured Web Development", Journal of Software, vol.12, no. 4, pp. 240-252, 2017.

[3] A. Kasgar, J. Agrawal and S. Sahu, "New Modified 256-bit MD5 Algorithm with SHA Compression Function", International Journal of Computer Applications, vol. 42, no. 12, 2012.

[4] A. Pandey and P. Bonde, "A Modified Approach For Cryptograpic Hash Function Based On MD5 Algorithm", International Journal of Engineering Research & Technology (IJERT),vol. 2, no. 8, 2013.

[5] C. Paar, J. Pelzl and B. Preneel, "Understanding Cryptography: A Textbook for Students and Practitioners", 2010.

[6] D.Farhan and M. Ali,"Enhancement MD5 Depend on Multi Techniques", International Journal on Software Engineering, 2015.

[7] E. Sediyono, K. Santoso and Suhartono, "Secure Login by Using One-time Password Authentication Based on MD5 Hash Encrypted SMS", IEEE, 2013.

[8] F. Wang, C. Yang, Q. Wu, and Z. Shi, "Constant Memory Optimizations in MD5 Crypt Cracking Algorithm on GPU

Accelerated Supercomputer Using CUDA", The 7th International Conference on Computer Science & Education (ICCSE 2012), 2012.

[9] G. Raj, R. Kesireddi and S. Gupta, "Enhancement of SecurityMechanism for Confidential Data using AES-128, 192 and 256bit Encryption in Cloud", 1st International Conference on Next Generation Computing Technologies, 2015.

[10] H. Kumar, S. Kumar, R. Joseph, D. Kumar, S. Singh, A. Kumar and P. Kumar, "Rainbow Table to crack Password using MD5 Hashing Algorithm", Proceedings of 2013 IEEE Conference on Information and Communication Technologies (ICT2013), 2013.

[11] K. Kim and U. Kyong, "Efficient implementation of MD5 Algorithm in Password Recovery of a PDF file", IEEE, 2013.

[12] K. Theoharoulis and I. Papaefstathiou, "Implementing Rainbow Tables in High end FPGAs for superfast password Cracking", International Conference on Field Programmable Logic and Applications, 2010.

[13] L. Zhong, W. Wan and D. Kong, " Javaweb Login Authentication Based On Improved Md5 Algorithm", IEEE, 2016.

[14] N. Khanezaei abd Z. Hanapi, "A Framework Based on RSA and AES Encryption Algorithms for Cloud Computing Services", Conference on Systems, Process and Control, 2014.

[15] M. D. A. Chawdhury and A. Habib, "Security Enhancement of MD5 Hashed Passwords by Using the Unused Bits of TCP Header", Proceedings of 11th International Conference on Computer and Information Technology (ICCIT 2008), 2008.

[16] M. Kioon, Z. Wang and S. Das, "Security Analysis of MD5 algorithm in Password Storage", Proceedings of the 2nd International Symposium on Computer, Communication, Control and Automation (ISCCCA-13), 2013.

[17] M. Stevens, A. Lenstra and B. Weger, "Chosen-prefix Colisions for MD5 and applications", International Journal on Applied Cryptography, vol. 2, no. 4, 2012.

[18] Mehrabani and Eshghi, "Design of an ASIP Processor for MD5 Hash Algorithm", 20th Telecommunications Forum TELFOR, 2012.

[19] P. Walia and V. Thapar, "Implementation of New Modified MD5-512 bit Algorithm for Cryptography", International Journal of Innovative Research in Advanced Engineering (IJIRAE), vol. 1, no. 6, 2014.

[20] Q. Kester, L. Nana, A. Pascu and S. Gire, "A New Encyption Cipher for Securing Digital Images of Video Surveillance Devices using Diffie-Hellman- MD5 Algorithm and RGB shuffling", European Modelling Symposium IEEE, 2013.

[21] R. Roshdy, M. Fouad and M. Aboul-Dahab, "Design And Implementation A New Security Hash Algorithm Based On Md5 And Sha-256", International Journal of Engineering Sciences & Emerging Technologies, vol. 6, no. 1, pp. 29-36, 2013.

[22] S. Sonh, "A Study on Area-Efficient Design of Unified MD5 and HAS-160 Hash Algorithms", The Journal of the Korean Institute of Information and Communication Engineering, vol. 16, no. 5, pp. 1015-1022, 2012.

[23] V. Kapoor, "A new security system using ECC and MD5", IJERT, 2014.

[24] V. Mahalle and A. Shahade, "Enhancing the Data Security in Cloud by Implementing Hybrid (Rsa & Aes) Encryption Algorithm", IEEE, 2014.

[25] V. Mishra and V. Pandey, "Architecture based on MD5 and MD5-512 Bit Applications. International Journal of Computer Applications", 2013.

[26] Vidhya and Sasilatha, "Performance analysis of Black Hole attack Detection scheme using MD5 algorithm in WSN", International Conference on Smart Structures & Systems, 2014.

[27] W. Xijin and F. Linxiu,"The Application Research of MD5 Encryption Algorithm in DCT Digital Watermarking", Physics Procedia, 2012.

[28] X. Zheng and J. Jin, "Research for the Application and Safety of MD5 Algorithm in Password Authentication", IEEE, 2012.

[29] Y. Sasaki, "Improved Single-Key Distinguisher on HMAC-MD5 and Key Recovery Attacks on Sandwich-MAC-MD5 and MD5-MAC", IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, pp. 26-38, 2015.

[30] C. Dobre, "Hash Function XOR". Information Security Journal, 2013.

[31] E. Vandierndonck, and K. De Bosschere. "XOR- based Hash Function", IEEE Transactions on Computers, vol. 54, no.7, 2005.

[32] G. Gordon, "Properties of Hash Functions", Sirindhorn International Institute of Technology, Thailand:Thammasat University, 2016.

[33] A. Reyes, E. Festijo and R. Medina, "Securing One Time Password (OTP) for MultiFactor Out-of-Band Athentication through a 128- bit Blowfish", International Journal of Communication Networks and Information Security (IJCNIS), vol. 10, no. 1, 2018.

[34] S. Nashwan, "SE-H: Secure and Efficient Hash Protocol for RFID System", International Journal of Communication Networks and Information Security (IJCNIS, vol. 9, no. 3, 2017.

[35] M. Jain and N. Agrawal, "Improved 640 Bit Message Digest Cryptographic Algorithm", 4th International Conference on System Modeling & Advancement in Research Trends (SMART), 2015.

[36] A. Bhandari, M. Bhuiyan and P. Prasad, "Enhancement of MD5 Algorithm for Secured Web Development", Journal of Software, vol. 12, no. 4, 2017.

[37] G. Liu and H. Qi, "Improvement and Implementation of an MD5 Algorithm", Information Technology and Computer Application Engineering, 2014.

[38] S. Tyagi and S. Luthra, "A New Improved and Secure Version of MD5", International Journal of Engineering And Computer Science, vol. 3, no. 7, pp. 7257-7261, 2014.

[39] A. Santra and N. S, "A Modified MD5 Algorithm for Wireless Networks", International Journal of Advanced Research in Computer Science, vol. 3, no. 2, 2012.

[40] Smith and S. Bhati, "A New Modified 512-bit Approach for MD5 Algorithm", International Journal of Latest Technology in Engineering, Management & Applied Science, vol. 2, no. 8, 2013.

[41] .Wang, C. Yang, Q. Wu and Z. Shi, "Constant Memory Optimizations in MD5 Crypt Cracking Algorithm on GPU-Accelerated Supercomputer Using CUDA", The 7th International Conference on Computer Science & Education, 2012.

[42] G. Singh and Supriya, "Modified Vigenere Encryption Algorithm and its Hybrid Implementation with Base64 and AES", Second International Conference on Advanced Computing, 2013