A Comparison of the of the Security Provided by Window's Local Area Network Manager and Message Digest Five Hashes in the Application of Personal and Business Computers.

Abstract

The research conducted by the author of this paper clearly demonstrates that passwords hashed with the Window's Local Area Network Manager algorithm are significantly more susceptible to being compromised via several of the most common cryptographic attacks, than passwords hashed with the Message Digest Five algorithm. This paper will explore the relative strength of the Message Digest Five hashing algorithm compared to the Window's LAN Manager hashing algorithm by comparing the theoretical times needed to find a collision in the hashes generated by inputting plain text passwords of varying complexity into the hashing algorithms. By knowing the strengths and weakness of each algorithm, the end user can construct passwords that are less susceptible to being cracked via the methods described in the author's experiments.

Introduction

According to Vleck (2009), passwords have been used for years as a method of limiting access to computers and computer programs. One of the first recorded cases was in 1961 when the Compatible Time Sharing System was introduced; it had a log-in command that prompted the user to supply a password before it could be used (Vleck, 2009). Restricting access to a computer might have seemed hardly an issue back in the 1960's because only highly trained operators could use them. However, as of 2007, 71% of households in the United States have access to at least one computer (U.S. Census Bureau, 2007). Limiting who has access to a user's computer is a much bigger concern than it ever has been before.

In the professional world, unauthorized disclosure of information, such as billing information or corporate trade secrets, could have catastrophic results for the business. If the medical records for an entire clinic were leaked on the internet, the clinic’s reputation could be permanently tarnished. To help address the very fundamental need for privacy, most computer operating systems come with a built-in method for user authentication, and in most cases, this means a user name and password.

A vast majority of the time when the user creates a password to protect his account from unauthorized access, the password itself is not stored in what is known as “plain text” on the computer. The password is run through a “hashing algorithm” to generate a “hashed” version of the user’s password. The best way to define what a hashing algorithm does to a password is to describe some of the properties of the actual outputted hash.

In most hashing algorithms, a user can input text of any length and expect an output of a set number of characters and even the slightest change to the inputted text will result in a completely different output. According to the hash calculator found at lmcrack.com, the word “password” becomes “E52CAC67419A9A24A3B108F3FA6CB6D”, and the word “passwords” becomes

"20F40FEB2B32D3138E2648E6CDDFCAD4" after the Windows LAN Manager Hashing algorithm processes them (2009). One should note how the slightest change in the plain text input results in a completely different hashed output. The fundamental advantage to storing a password as a hash as apposed to clear text boils down to security(Pastore, 2006). For example, if a disgruntled network administrator had decided to sabotage his employers business by releasing sensitive documents stored on one of his co-worker's computers he would not be able to simply acquire the password to his co-worker's computer by using a tool like pwdump to dump the passwords from the computer to a CD or flash drive, so that he could comb through the computer's files after hours for sensitive documents to disclose. Instead, he would be given a hash like one of the ones provided in the example above. This would provide a significant obstacle for the disgruntled network administrator because he would have to figure out a way to decode the hash. According to McGlinn (2007), the one-way nature of most hashing algorithms makes it mathematically impossible to discover the user's password by attempting to "reverse the hash" by simply working the algorithm backwards (McGlinn, 2007).

This would leave the disgruntled network administrator with a few options to attempt to discover the plain text equivalent of the hashes he has retrieved. Some of the more common tools at a hacker's disposal that are used to help discover a user's passwords from a hash are Dictionary Attacks, Online Databases, and Brute Force Attacks says Pastore (Pastore, 2006), and are included in the author's experiments. The author attempts to determine which of these attacks appear to be the most effective against Windows LAN Manager hashes and Message Digest Five hashes and determine if there is a significant difference in the protection offered by each of these hashing algorithms. By knowing the strengths and weakness of each algorithm, the end user can construct passwords that are less susceptible to being cracked.

Subjects

The author had a third party create six passwords of increasing levels of complexity with the first password being appropriate to protect something that the user would openly disclose, and the sixth password being appropriate to protect data that should not be disclosed to anyone under any circumstances. The use of the third party prevented the possibility of the author consciously creating passwords that would be easily compromised by the software chosen by the author. The third party was informed that it could use upper and lower case letters, the numbers zero through nine, and any symbol that appears on a standard QWERTY keyboard. The passwords provided by the third party are as follows: david, robotcat, EuDGw, AsiZR78, F$[]T2%F34, and Mb%&Qa4)YwSx5. Each of the six passwords were processed into both Window's LAN Manager and Message Digest Five hashes as shown in the table below.

Clear Text	david	robotcat	EuDGw	AsiZR78	F$[]T2%F34	Mb%&Qa4)YwSx5
Windows LAN Manager	0C60300 E8A1677 14AAD3B 435B514 04EE	520659C 5296942 B5417EA F50CFAC 29C3	89957A1 506DA21 79AAD3B 435B514 04EE	D664D8C D65737D 93AAD3B 435B514 04EE	6F2991C FF111A3 073D846 4B9A7DB 91EA	4C03D7F AA7800A CEA9774 2F5C69B 60B8
Message Digest Five	172522E C1028AB 781D9DF D17EACA 4427	B58DA35 2651E5A 731AF16 D1D3818 05AD	10A6668 ABE82C8 4C057CA BD08CFD 93B0	890B433 A8ED8E5 FE44480 89C6A9C 2C1A	62DF27F 03F849E 07A1018 64D453F1 FBB9	A0F9B1B 192DF1B A276CAD 0BBB251 8873

Apparatuses

The tests were run on the author’s personal computer at home. The computer’s relevant specifications used in the tests are as follows: a single quad core AMD Phenom II X4 955 Processor over clocked to 3674.9 MHz, with 8192MB of dual channel DDR3 ram, the display adapter used was a GeForce GTX 285 with 1024MB of GDDR3 ram clocked at 648MHZ, the motherboard used was a Gigabyte GA-MA770T-UD3P, and the operating system used was Microsoft Windows 7 (6.1) Ultimate Edition (Build 7100). The password auditing tool “Cain and Abel”, available at http://www.oxid.it/cain.html, was used to calculate the Windows LAN Manager and Message Digest Five hashes, and to perform the dictionary and brute force attacks. The dictionary used for the attack came in a set and can be downloaded free of charge at http://oxid.netsons.org/phpBB3/viewtopic.php?f=15&t=3177. The particular dictionary used from the set for all of the dictionary attack test was the “Ultimate” dictionary. It consisted of over 65.2 million characters. And although the dictionary that was used was far from all-inclusive, it was essential to the experiment, because most hackers would try to use a dictionary attack to attempt to discover a cognitive password before attempting a brute force attack in order to save valuable time.

Finally, the online data bases http://www.lmcrack.com and http://hashkiller.com/index.php?action=md5webcrack were used to search for instances of the hash being previously calculated and broken. Finally, the system resources monitor “Process Explorer” available at http://technet.microsoft.com/en-us/sysinternals/bb896653.aspx was used to monitor the RAM and Processor load during the “dictionary” and “brute-force attack” test.

Procedures

The Window's LAN Manager hashes were first tested by being run through an online database. After calculating the hashes in Cain and Abel, the hashes were copied from the hash calculator and then pasted into the field labeled “LM (LanManager) Reverse Hash *via* Rainbow Tables” at www.lmcrack.com. If the database was able to find a match it was recorded into a spread sheet as “found” and if the database could not find a match it was recorded as “not found”.

The next test was the “dictionary attack”. The Windows LAN Manager hashes were copied from the hash calculator in Cain and Abel and inputted in the “LM/NTLM Hashes” field located under the “cracker” menu. After the hashes were inputted, the “LM Hashes” button was selected under the dictionary attack menu, which was selected by right clicking the hash under the “user name column”. This brought up the “dictionary attack window”. From here the “Ultimate” dictionary was selected by right clicking in the white space directly under the “File” column. Then clicking “add to list” and finally navigating to the dictionary file then selecting “open”. All of the additional options in the “dictionary attack window” were left checked with exceptions of “ Num. Sub. Perms.” and “Two numbers Hybrid Brute”. At this time a system resources monitor was enabled to keep track of the system RAM and Processor load and distribution. The test was started by clicking the start button. The system RAM and Processing power used was measured before and during the test. To determine how much of the available systems resources were actually being used by Cain and Abel and documented in the spread sheet as percent used. The results were of the “dictionary attack” were documented in the spread sheet as either “not found” or as the amount of time needed to find the clear text equivalent of the hash.

The last test performed on the Window's LAN Manager passwords was the “brute-force Attack”. The hashes were copied out of the hash calculator in Cain and Abel and then inputted in to the “LM/NTLM Hashes” field located under the “cracker” menu just like in the dictionary attack. This time however, the LM hash button was selected under the “Brute-Force Attack” sub menu brought up by right clicking on the hash to be attacked. After the “Brute-Force Attack” window was brought up, the password's length was inputted into the “password length” menu. The appropriate character set was selected under the “pre-defined” menu. At this time the system resources monitor was enabled once again so that the it could determined how much of the available systems resources were actually being used by Cain and Abel before and during the test. Finally, the test was started by clicking the start button. A screen capture was taken so that the time under the “Time Left” display could be recorded in the spreadsheet along with the RAM and Processor load.

The next tests were performed on the Message Digest Five hashes. For the online data base search, the site http://hashkiller.com/index.php?action=md5webcrack was used to search for previous instances of the Message Digest Five hashes being used and documented. The Message Digest Five hashes were copied out of Cain and Abel and pasted into the “MD5” field and submitted. The result for each hash was recorded in the spreadsheet as either “found” or “not found”.

The next test was the “Dictionary Attack”. The Message Digest Five hashes were copied from the hash calculator in Cain and Abel and inputted in the “MD5 Hashes” field located under the “cracker” menu. After the hashes were inputted the, “Dictionary Attack” button was selected by right clicking the hash under the “MD5 Hash” column. This brought up the “dictionary attack window”. From here, the “Ultimate” dictionary was selected again by right clicking in the white space directly under the “File” column and then selecting “add to list” and finally navigating to the dictionary file then selecting “open”. Once again, all of the additional options in the

“Dictionary Attack Window” were left checked with exceptions of “Num. Sub. Perms.” and “Two numbers Hybrid Brute”.The system resources monitor was re-enabled again so that the RAM and processor load could be recorded before and during the test. The test was initiated by clicking on the “Start” button, and the results were recorded in the spreadsheet as either “Not Found” or if the password was found, the time to find the password was recorded.

The final test was the on the Message Digest Five hashes was the “Brute-Force Attack”. The hashes were copied out of the hash calculator in Cain and Abel and then inputted in the “MD5 Hashes” field located under the “cracker” menu just like in the preceding test. Next, the inputted hash was right clicked under the “MD5 Hashes” tab to bring up the “Brute-Force Attack” button. After the button was selected, the “Brute-Force Attack” window was brought up. Once at this window, the password's length was inputted into the “Password Length” field, and the appropriate character set was selected under the “Charset” tab. Next, the system recourse monitor was re-enabled so that readings could be taken before and during the test. Finally, the test was started by clicking on the “Start” button. A screen capture was taken at the beginning of each test so that the “Time Left” display could be accurately recorded in the spreadsheet along with the RAM and processor load readings.

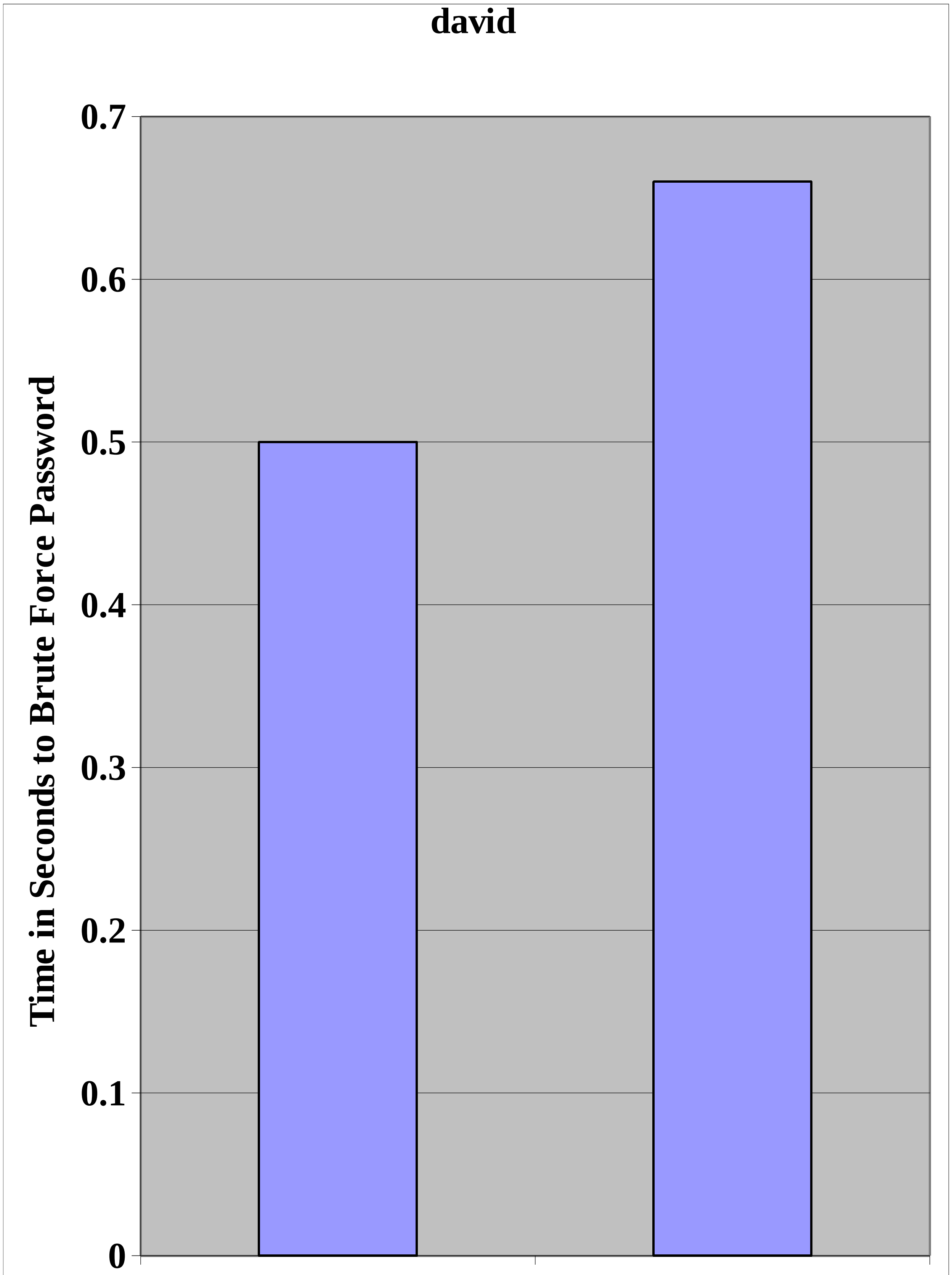
Results

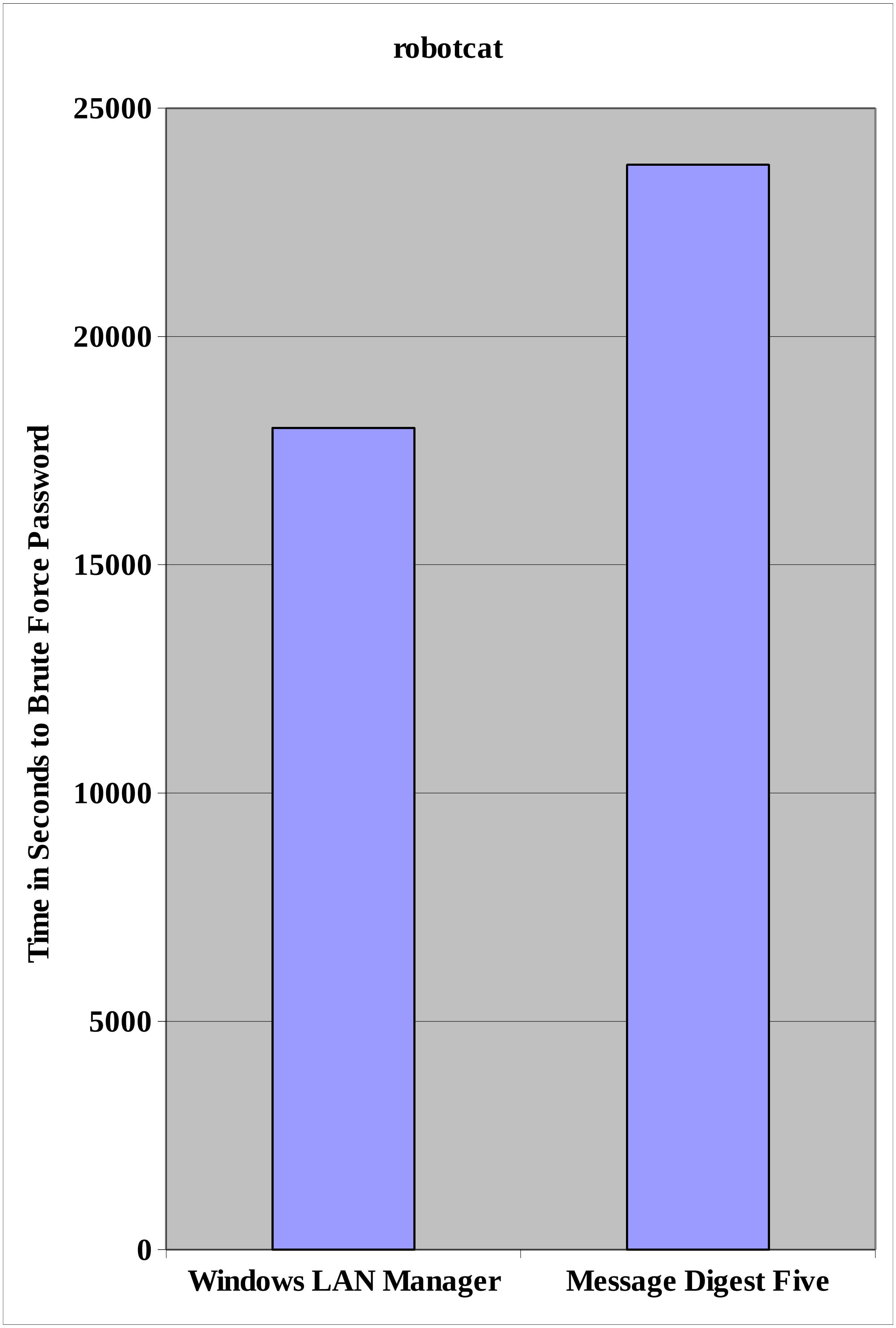
The online database search for the Windows LAN Manager hashes proved successful six out of six times, and all of the passwords were recovered as soon as the webpage could be refreshed. The online database search for the Message Digest Fives hashes was successful in finding three of the six passwords: “david”, “robotcat”, and “EuDGw”, and these were recovered as soon the webpage could be refreshed.

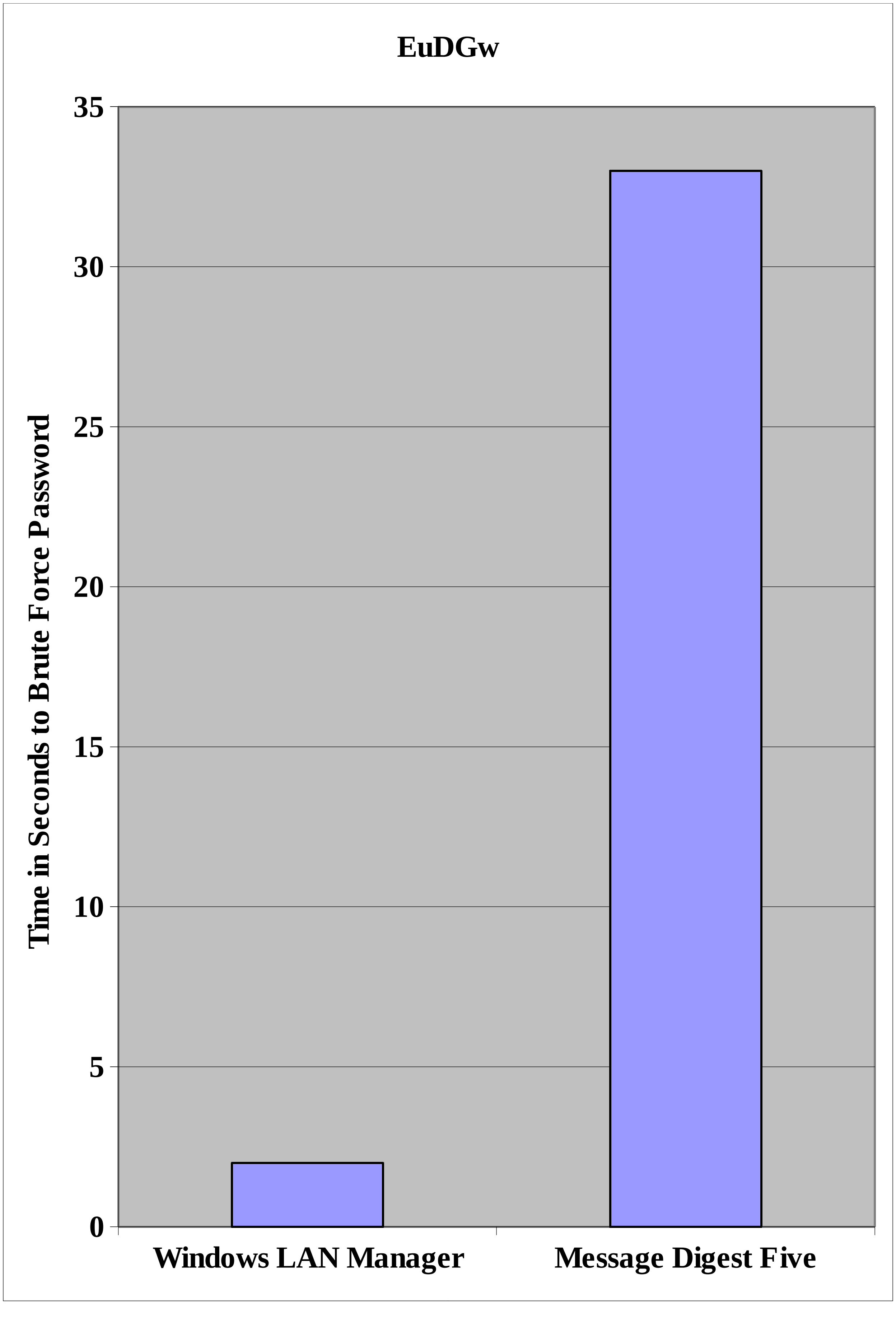
The dictionary attack was only successful in one instance against both of the hashing algorithms, and in both cases it was the password “david”. It was recovered in both instances in under twenty seconds after running through 8% of the dictionary. According to Process Explorer before the tests were executed 2% of the total processing power and 19% of the system RAM was used. During the tests 25% of the available processing power and 19.5% of the RAM was used. This indicated that only one of the processor’s cores was being completely utilized, while the other three cores were regulated to background processes.

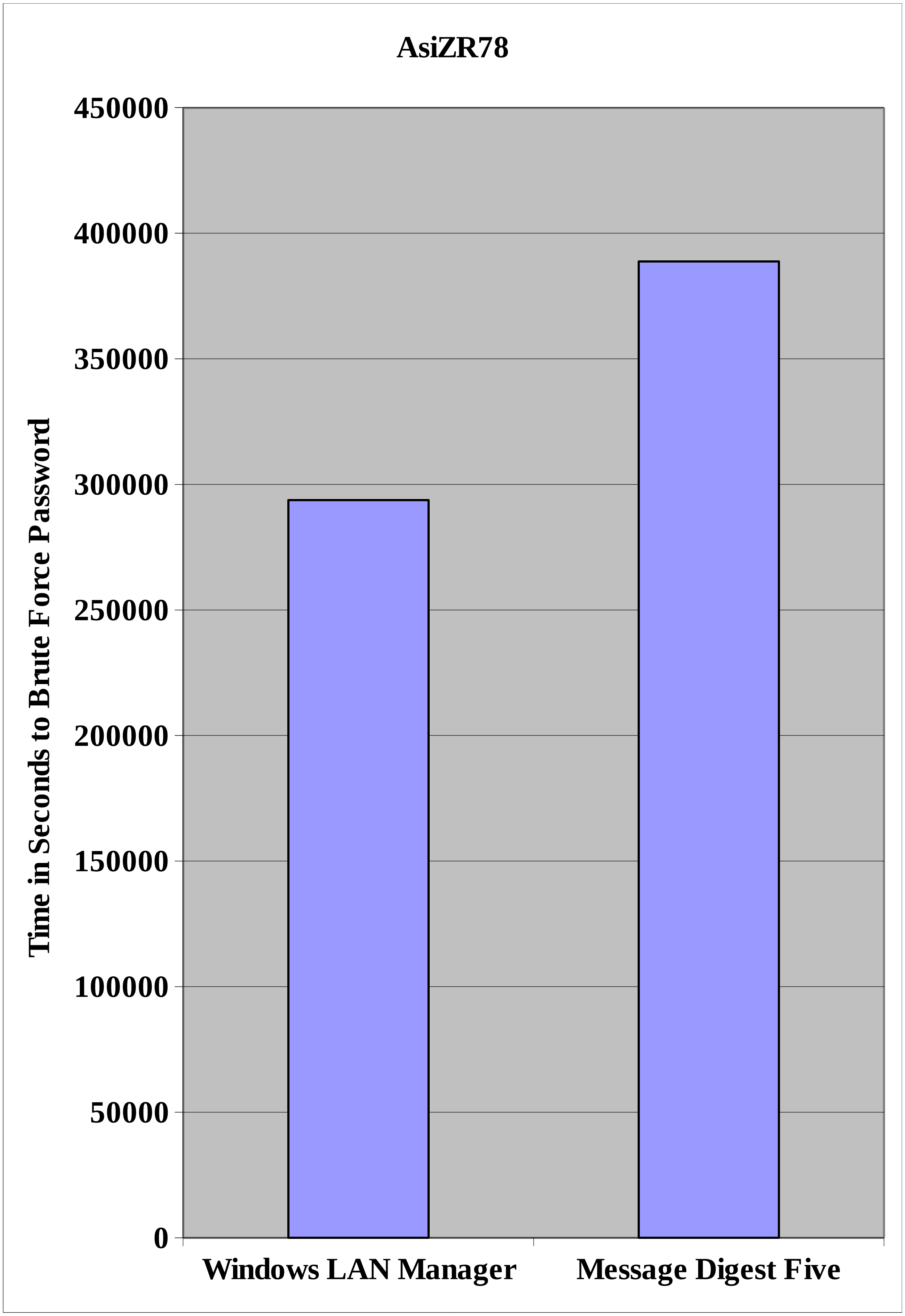
The Brute-Force attacks would have taken longer on average to perform with the times ranging from less than one second to crack the simplest password “david”, all the way to many thousands of years to theoretically crack “Mb%&Qa4)YwSx5”. The theoretical times needed to crack the passwords via brute-force Attacks are represented in the charts on the next page. When the data from Process Explorer was analyzed, it revealed again that one of the processor’s cores was loaded to maximum capacity while the remaining three cores were mainly regulated to background processes. There was not a significant increase in the amount of system RAM utilized.

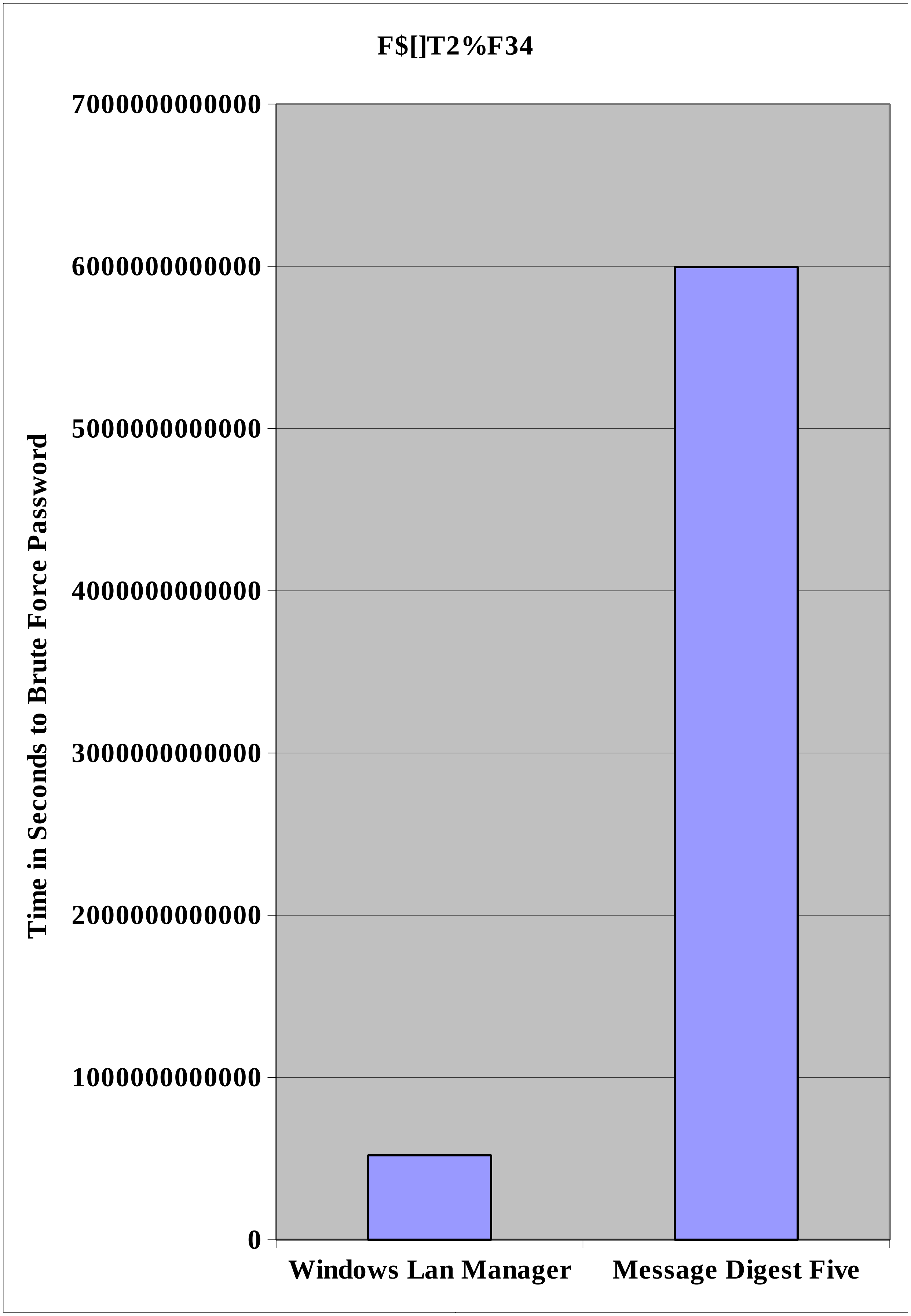
Figures

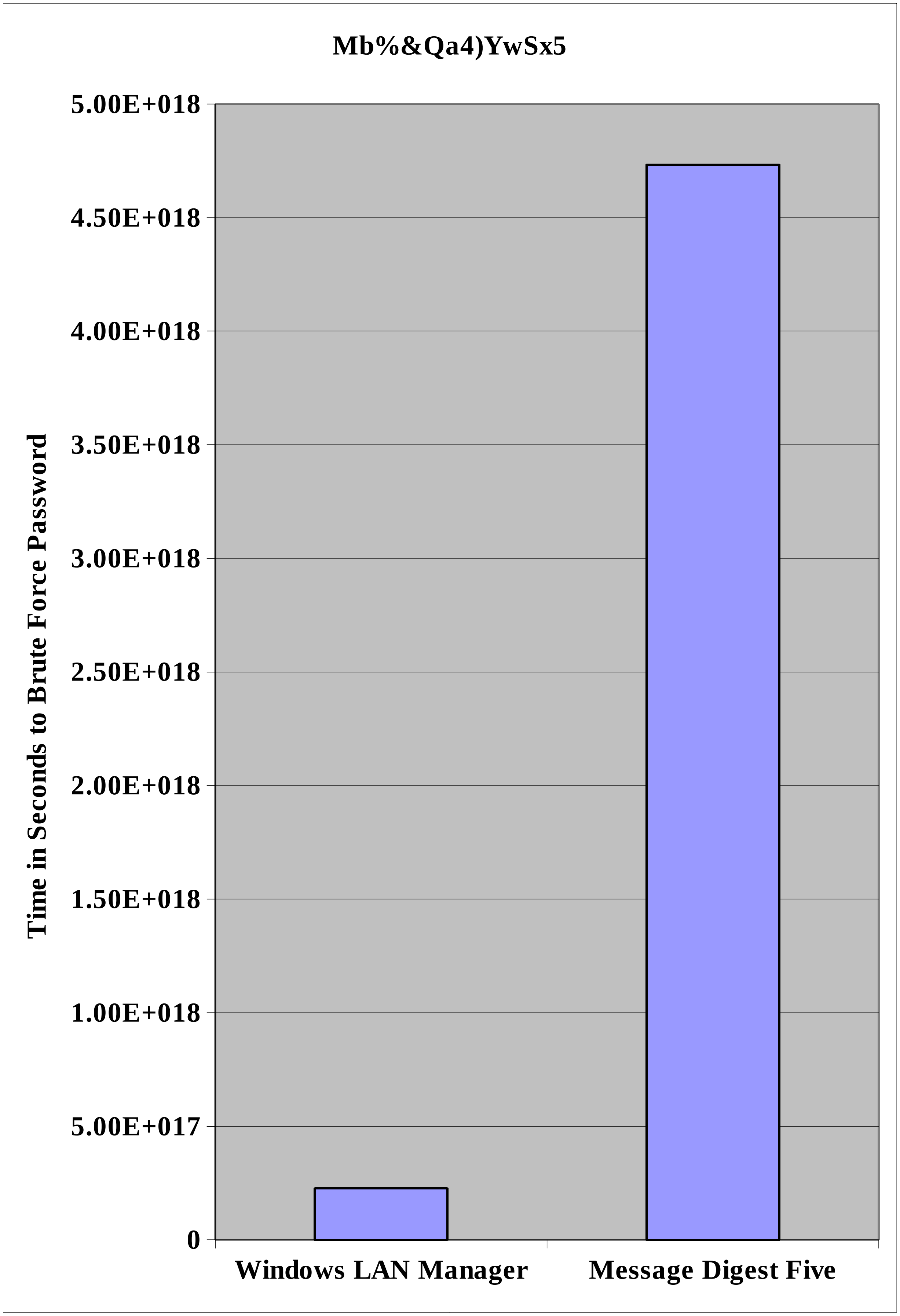

david
Time in Seconds to Brute Force Password
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Windows LAN Manager
Message Digest Five


robotcat
25000
20000
15000
10000
5000
0
Time in Seconds to Brute Force Password
Windows LAN Manager
Message Digest Five


EuDGw
Time in Seconds to Brute Force Password
35
30
25
20
15
10
5
0
Windows LAN Manager
Message Digest Five


AsiZR78
450000
400000
350000
300000
250000
200000
150000
100000
50000
0
Time in Seconds to Brute Force Password
Windows LAN Manager
Message Digest Five


F$[]T2%F34
Time in Seconds to Brute Force Password
7000000000000
6000000000000
5000000000000
4000000000000
3000000000000
2000000000000
1000000000000
0
Windows Lan Manager
Message Digest Five



Mb%&Qa4)YwSx5
Time in Seconds to Brute Force Password
5.00E+018
4.50E+018
4.00E+018
3.50E+018
3.00E+018
2.50E+018
2.00E+018
1.50E+018
1.00E+018
5.00E+017
0
Windows LAN Manager
Message Digest Five

Discussion

The research shows that passwords hashed with the Message Digest Five hashing are significantly more secure than passwords hashed with the Windows Local Area Network Manager hashing algorithm. Not only were fewer of the passwords found in the online data base searches, but also the time difference to discover the plain text equivalent of the hashes via a brute-force attack of the passwords was vast. The only exception was for the password “david” which was discovered in less than a second in both instances.

While it was true that fewer of the passwords hashed with the Message Digest Five algorithm were discovered by the online database search, half of the passwords tested were still found, which still shows a significant vulnerability. The data showed that the passwords hashed with the Message Digest Five algorithm were only safe from this attack once the passwords were comprised of upper and lower case letters, the numbers 0-9, and had a length of at least seven characters. However, it is still better to have a password length of at least ten characters to thwart most brute-force attacks.

The passwords hashed, using the Windows Local Area Network Manager hashing algorithm, were found to be extremely vulnerable to the online database search. In every instance no matter how complex the passwords were, the clear text was always found. This shows that if at all possible avoid using the Windows Local Area Network Manager hashing algorithm to store your passwords.

The only test that showed both of the algorithms having a shared weakness was the dictionary attack. Of the two passwords that could be considered cognitive, “david” and “robotcat”, only david was found. There is still a 50/50 chance that a cognitive password of any length could be found in a relatively short time frame. To avoid any possibility of a password being broken using a dictionary attack, do not use cognitive passwords.

In summary, the research reflects that the best passwords that can be constructed for personal and business computers, are at least eight characters long, and should be composed of both upper and lower case letters, include the numbers zero through nine, special keyboard symbols such as “%”. The passwords should not be cognitive, and should be hashed by the Message Digest Five hashing algorithm instead of the Windows Local Area Network Manager hashing algorithm.

References

Vleck, T.V. (2009, September 1). *Story: The Password File*.
Retrieved from
http://www.multicians.org/thvv/7094.html

U.S. Census Bureau (2007, October). *Computer and Internet Use in the United States*.
Retrieved from
http://www.census.gov/population/www/socdemo/computer/2007.html

McGlinn, J. (2007). *Password hashing*.
Retrieved from
http://phpsec.org/articles/2005/password-hashing.html

Pastore, M., & Dulaney, E. (2006). *CompTIA security+ study guide: exam SY0-101*.
Hoboken,NJ : Wiley

(2009). *Hash calculator*. Retrieved from http://lmcrack.com/