All in a day's work: Password cracking for the rest of us

Jørgen Blakstad
ITEM, NTNU

Rune Walsø Nergård
ITEM, NTNU

Martin Gilje Jaatun
SINTEF ICT

Danilo Gligoroski
ITEM, NTNU

Abstract

The majority of computer systems are still protected primarily with a user name and password, and many users employ the same password on multiple systems. Additionally, some of the most popular operating systems such as Windows XP, Windows Vista and the upcoming Windows 7, still use ad-hoc constructed hash functions such as LM, while many Linux variants use the "broken" hash function MD5. This paper describes an experiment where we have tested the strength of a selection of passwords when converted to LM, NT and MD5 hashes, respectively, using commonly available tools. Our conclusion is that a large number of passwords can be cracked within a normal working day, and that all LM hash passwords can be recovered before morning coffee. The use of such weak hash functions in the process of user authentication in these operating systems poses a significant threat to an organization's security.

Keywords: Password, Security, Cracking, LM Hash, NT, MD5

1 Introduction

Most computer systems are still protected primarily by a username and password combination. Using passwords and password management routines for giving access rights is a technique that is as old as the history of operating systems. Generally speaking the development in the way how passwords were kept and used in operating systems went through these phases:

1. Keep passwords in pure text form [1].
2. Encrypt passwords with some naive encrypting algorithms or modifications of DES and keep just the encrypted parts [2, 3].
3. Encrypt passwords with DES and keep just the encrypted parts [2, 3].
4. Pre-pend the passwords with some added random value called "salt" and then encrypt them with DES. Keep just the encrypted parts [2].

This paper was presented at the NISK-2009 conference.

5. Hash the passwords with some cryptographic hash function and keep just the hash digests [4, 5, 6].

6. Pre-pend the passwords with some added random value called “salt” and then hash them with some cryptographic hash function. Keep just the hash digests [7, 8].

It is generally accepted that the techniques that combine salting and hashing of the passwords (item 6 in the previous list) are much more secure than the techniques described in items 1 – 5. The security of the techniques under items 5 and 6 also depend on the security of the cryptographic hash function employed. If the used hash function is weak (i.e., it is easy to find preimages, second preimages or collisions for that hash function), then protection of the passwords by that hash function is also weak. For the hash functions that we investigate in this paper it is very easy to find second-preimages.

In the area of cryptographic hash functions the most popular family of hash functions is the MD4 family, which includes functions like MD5, HAVAL, RIPEMD, RIPEMD-160, SHA-0, SHA-1, SHA-2 and many others [9, 10, 11]. Many of those cryptographic functions have been broken in practice (such as MD4, MD5, HAVAL, RIPEMD, SHA-0), and SHA-1 was theoretically broken in 2005 [12].

However, it is disappointing that popular operating systems still use weak cryptographic hash functions (ad-hoc constructed hash functions that do not belong even to the MD4 family) and many of them do not even use salting techniques. Typical examples are Microsoft Windows XP that uses an ad-hoc hash solution called “LM hash”, the more recent Microsoft Vista and the latest Windows 7 operating systems that use an ad-hoc hash function called “NT hash” (seen as an improved version of LM hash, partly based on the broken MD4 function), and some versions of the popular Linux operating system Ubuntu that use the broken MD5 hash function.

In this experiment dictionary attacks and attacks with the use of rainbow tables, which are both password cracking techniques, were applied.

Motivated by the fact that those most popular operating systems are still using very weak and practically broken hash functions for dealing with passwords and user authentication techniques, we chose 30 passwords of different strengths, and wanted to see how many could be cracked during a period of *1 day* (8 hours).

In our experiments we have included exactly the three aforementioned types of hash functions: LM hash, NT hash and MD5 hash. Note that although the LM and NT hashes were originally developed for network authentication, we will in this paper only consider them in the context of non-network password storage.

The remainder of the paper is structured as follows: In Section 2 we give a short introduction to the concept of rainbow tables, followed by a brief description of our laboratory environment in Section 3. Section 4 describes how our experiment was carried out, with results presented in Section 5. We discuss the results in Section 6, and offer our conclusions and suggestions for further work in Section 7.

2 A Short Introduction to Rainbow Tables

Rainbow tables contain a connection between a hash value and its corresponding password. By having this connection precalculated for all possible hash values, a quick search through the tables for a desired hash value can reveal the password.

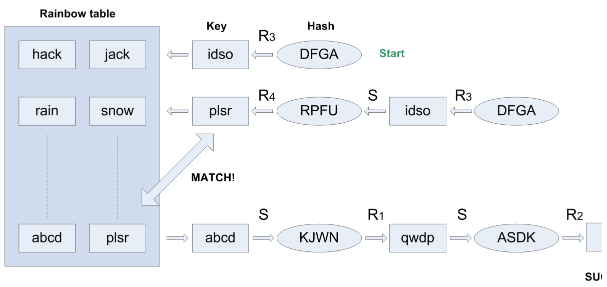

Figure 1: The rainbow table lookup process

Hellman [13] proposed a time-memory trade-off when applying a cryptanalytic attack. Hellman used something called hash chains to decrease the memory requirements. Only the first and the last element of the chain are stored in the memory, saving memory at the cost of cryptanalysis time. Additional memory trade off is achieved by reducing the hash values to "keys" with the help of a reduction function. Instead of using a single reduction function R, Oechslin [14] suggested using a sequence of related reduction functions R_1 up to R_{t-1}. He called the new chains rainbow chains, and the tables containing these chains rainbow tables.

In Figure 1, the hash value *DFGA* is subject to cracking. In the top right corner of the figure, *DFGA* is reduced with a reduction function (R) and the key is looked up in the rainbow tables. The key had no match with any of the endpoint keys, and by using the key a new hash is produced with the hash function (S) and then again further reduced with a new reduction function. The lookup process now finds a match and from the corresponding startpoint key *abcd* through a number of different reduction functions and the hash function the desired key *oier* is derived. The key *oier* is then identified as the key corresponding to the original hash *DFGA*.

3 Laboratory Environment

We made fresh installations of Windows XP, Windows Vista, Windows 7 RC and Ubuntu 8.10 on identical Dell OptiPlex GX270 systems with 2.6 GHz Pentium 4 processors and 1 GB RAM. The dictionary and the rainbow table attacks on the LM- and NT hashes were run in parallel on two Windows Vista systems. The dictionary attack on the MD5 hashes ran separately on a Ubuntu 8.10 system.

The rainbow tables used to crack the LM hash were much smaller than those for other hashes because of their weak cryptographic properties (as there are less combinations to try). Oechslin [14] also used the LM hash as an example when he introduced the concept of rainbow tables. We chose to include the NT hash because this is the new hash type included by Microsoft on newer Windows versions. The NT hash is supposed to solve many of the weaknesses of the LM hash, but still no salt is included. This is why the NT hash also is susceptible to the rainbow table attack. The MD5 hash was chosen because it is a common type of hash used in Unix systems. MD5 hashes are supported by John the Ripper [15], which is the tool that we used for the MD5 hashes in this experiment, and they may be cracked with the use of a dictionary attack. At RainbowCrack's web pages [16] rainbow tables for MD5 are available but we did not include these in the experiment due to time constraints.

4 Experimental Procedure

Table 1 gives an overview of which techniques (dictionary attack and/or rainbow table attack) that were applied to what type of hash. For all the dictionary attacks performed we used the wordlist that is already included in the Cain & Abel tool [17]; this list was also used with John the Ripper. For the LM hashes we used rainbow tables specifically designed to crack LM hashes and these tables, which in all consists of 64 subtables, were in total 64GB in size. The LM rainbow tables were downloaded for *free* from [18].

For the NT hashes we used rainbow tables that were specifically designed to crack NT hashes, but these tables were not free. It is possible to obtain free NT rainbow tables from Ophcrack's web pages [19], such as the one named *Vista free*, but these do not include as many characters as the versions you have to pay for. The NT rainbow tables that we bought are called *Vista special*, are 8GB in size and were obtained from Ophcrack's web page as well. The *Vista special* rainbow tables in all consist of 4 rainbow tables, and cover different character sets depending on password length: For passwords that are 6 characters or less they cover the same character set as the LMhash tables, while for 8 character password only numbers from 0-9 and lowercase characters are covered.

Type of hash	**Techniques**
LM hash	Dictionary attack and rainbow table attack
NT hash	Dictionary attack and rainbow table attack
MD5	Dictionary attack

Table 1: Applied techniques corresponding to their type of hash.

Selection of Passwords

In this experiment we wanted to test passwords of different strength. That is why we created three different password groups, and then added 10 password to each group. The ten passwords added to the *Most Popular* group are taken from Bruce Schneier's web page [20], 8 of the *Mnemonic Passwords* are taken from a mnemonic password generator [21] while 2 are created by us. The *Random Passwords* are collected from a random password generator [22] found on the Internet. The group labeled *Most Popular* contain some of the most common passwords used, and as we can see from Table 2, these passwords do not seem very secure. As a contrast, the *Random Passwords* are considered very secure because they are randomly generated with both lower-case- and upper-case letters, numbers and special characters. One disadvantage with the randomly generated passwords are that they are hard to remember, and that is why we have included some mnemonic passwords. A mnemonic password is a password where a user for example chooses a phrase that is easy to remember (mnemonic) and uses a character to represent each word in the phrase. As already mentioned, we chose to create two of the mnemonic passwords ourselves (5GCTw4tG@N and tcC!tW84s), in order to include mnemonic passwords of nine and ten characters in length. An overview of the 30 different passwords and to which group they belong is given in Table 2. Table 3 lists all the selected *Mnemonic Passwords* and their corresponding phrases.

Most Popular	**Mnemonic Passwords**	**Random Passwords**
password1	hm71li	qaSt4@
abc123	k0fzug	vANe$a
myspace1	fp6jar	!aFu9ut
password	msi89a0	C7e++AV
blink182	%l41pc	w2U$atHe
qwerty1	m.f0vgk	$_Ch6cU5
fuckyou	v*qbt4un	Ke42A2Pe*
123abc	qtdra#	rA3$_ey*c
baseball1	5GCTw4tG@N	b*#D9*7yEG
football1	tcC!tW84s	8rA*_pHa$8

Table 2: An overview of the 30 selected passwords, divided into three categories

Mnemonic Passwords	**Phrase**
hm71li	Hefin manages Sven's first lugubrious identity
k0fzug	Kath's nothing frazzles Zoe's unexpected gadget
fp6jar	Frank promotes Zach's jocular absentminded rabbit
msi89a0	Morgan seizes if eighth Nina argues nothings
%l41pc	Percy liberates Fourier's first priceless cabbage
m.f0vgk	Morgan stops for only Vivian glamorises kilns
v*qbt4un	Victor's handy quarter boosts Tim's fourth unknown number
qtdra#	Quentin traps Dave's reputable aloof hash
5GCTw4tG@N	Fifth Grade Comm Tech waiting for their Graduation at NTNU
tcC!tW84s	The current crisis in the world fights for survival

Table 3: 10 mnemonic passwords with their corresponding phrase

Applied Methodologies for Password Cracking

For the LM and NT hashes, we first applied a dictionary attack to exclude some of the weakest passwords before we started to crack the rest of the passwords with the use of rainbow tables. For the MD5 hashes we only applied a dictionary attack. We created 30 users on a Windows XP machine, 30 users on a Windows Vista machine and 30 users on a Ubuntu 8.10 machine to be able to attack all of the three different types of hashes (LM hash, NT hash and MD5 hash). The user accounts, and thus also the corresponding password files, were created on virtual machines using a program called VirtualBox [23]. To create this many users we made scripts to automate the process. The script for adding users in a UNIX system adds 30 users with their passwords hashed with MD5, The result was a shadow file created in Ubuntu 8.10 containing 30 MD5 hash values.

5 Results

In this section we present the results of our empirical password study. Keep in mind that this experiment was supposed to reveal how many of the 30 selected passwords

Type of hash	Tools
LM hash	Cain & Abel
NT hash	Cain & Abel, Ophcrack
MD5	John the Ripper

Table 4: An overview of the tools used

that could be revealed during 1 day (8 hours).

Timing

Phase 1 of this experiment was a timing phase used to measure how long time both the dictionary attack and the rainbow table attack with the use of rainbow tables would take when trying to crack the LM and NT hash. This phase also measured the time of the dictionary attack on the MD5 hash. The results from this timing phase is presented below. We still separate between LM hash, NT hash and MD5 hash.

LM hash

For the LM hash we used Cain & Abel [17] for both the dictionary and the rainbow table attack, and for the former we used a wordlist (Wordlist.txt) which is included by default in the Cain & Abel tool. When the LM hash is constructed, the password is first divided into two segments with maximum 7 characters in each segment. The two segments are then hashed separately. The password used in this phase consists of 10 characters and therefore 2 LM hash values are constructed, one for the 7 first characters and one for the remaining 3 characters. As expected, none of these two hash values was cracked after running the dictionary attack, and thus the password was not revealed.

For the dictionary attack we had to open the *Windows Task Manager* and use the *CPU Time* to measure the time. The total time for the dictionary attack *1 minute and 8 seconds* (68 seconds). From this time we have to deduct 6 seconds, which is the time we estimate is used from program initiation. This time measurements show how long time it may take if the password is not found during the dictionary attack.

Now let us take a look at the timing of the *rainbow table attack*. We used only one table to be able to create an estimate instead of running the whole attack, as this would take longer time than we wanted to spend on the timing phase. The rainbow table attack, using just one of the tables, took 291,50 + 49,91 = 341,41 seconds = 5 minutes 41 seconds, when the password was not revealed by this single rainbow table. When calculating the total time for the single rainbow table we included *Total disk access time* and *Total rainbow table time*. Now we had to make an estimate for the maximum time the rainbow table attack might take when including all the 64 rainbow tables: 5 minutes 41 seconds * 64 tables = 364 minutes 10 seconds (21850 seconds) = *6 hours 4 minutes*. This estimate illustrates the worst case scenario, if no password is found during the rainbow table attack.

Table 5 gives an overview of the timing results from the dictionary attack and from the estimate of the rainbow attack. The latter is based on the accomplishment of one of the 64 rainbow tables. Both these attacks were performed on one password (8rA*_pHa$8).

The idea with the timing phase was to find out how long time the dictionary attack and the rainbow table attack might take, and if it would be advantageous to include a dictionary attack to reveal the easiest passwords before running the rainbow table attack with the rainbow tables. The dictionary attack only used 0.24% of the available time and the rainbow table attack is estimated to constitue 76% of the time. This leaves us with plenty unused time in our 8-hour working day.

NT hash

For the NT hash we used Cain & Abel [17] for the dictionary attack and Ophcrack for the rainbow table attack. Let us first consider the dictionary attack. We used the same wordlist (Wordlist.txt) as for the LM hash.

The dictionary attack was measured to take 54 seconds, but from this we had to deduct 5 seconds as this was the time it took to start Cain & Abel. This makes the total time for the dictionary attack *49 seconds*. This gives an indication of how long time it might take if the password is not found during the dictionary attack.

The rainbow table attack on the NT hash, using just one of the tables, took *8 minutes 20 seconds*. The field displaying the time is labeled *Time elapsed*. In this case we do not need to deduct anything as Ophcrack has a build-in timing function which starts when the cracking process begins, and not when the program is loaded. As this was just an estimate when using one of the tables, we have to make an estimate of how long time it might take when all the 4 rainbow tables are included. The 4 NT rainbow tables mentioned here are the tables that together form the non-free *Vista special* tables. The total estimate for the use of rainbow tables on the NT hash is: 8 minutes 20 seconds (500 seconds) * 4 tables = *33 minutes 20 seconds* (2000 seconds). This estimate illustrates the case if no password is found during the rainbow table attack on the NT hash.

MD5 hash

For the MD5 hash we only performed a dictionary attack, and we used John the Ripper to find out how long time the dictionary attack would take on the selected password (8rA*_pHa$8).

We used the timing function included in the John the Ripper tool, and the figure shows that the dictionary attack, including the word mangling, took *51 minutes and 48 seconds* (3108 seconds) to complete on the selected password.

Table 5 gives an overview of the timing results from the dictionary attack performed on the MD5 hash value. Note that the dictionary attack was performed on just one password.

Hash	**Technique**	**Time**
LM	Dictionary attack	1 minute 8 seconds (68 seconds)
LM	Rainbow attack	6 hours 4 minutes (21850 seconds)
NT	Dictionary attack	49 seconds
NT	Rainbow Table attack	33 minutes 20 seconds (2000 seconds)
MD5	Dictionary attack	51 minutes 48 seconds (3108 seconds)

Table 5: Time measurement for the LM, NT and MD5 hash

Password Cracking

The LM hashes were obtained from a Windows XP machine, the NT hashes from a Windows Vista machine and the MD5 hashes from a Ubuntu 9.05 machine. We also performed an experiment on the side for the NT hashes, with the use of a machine running the newly released Windows 7 RC operating system. This was to be able to highlight possible improvements that Microsoft had done regarding the handling of the login password. A full outline of the selected passwords can be found in Table 2.

LM hash

Note that a password consisting of 7 characters or less will only have one LM hash value. *Nine passwords* were revealed from the dictionary attack, and are listed with the comment *(dict)* in Table 8. 30% av the passwords were revealed from the dictionary attack, while 70% of them will be a subject for the rainbow table attack.

The dictionary attack carried out with all the 30 selected passwords was measured to take 1 minute and 9 seconds. But also in this case we had to deduct the time it took for Cain & Abel to start as we used the *CPU Time* displayed in *Windows Task Manager.* Remember that the CPU time may not be completely accurate because of the influence from other processes. After deducting the 6 seconds that Cain & Abel used to start, the dictionary attack was estimated to take *1 minute 3 seconds.*

Now let us consider the *rainbow table attack* with the use of the LM rainbow tables. It should be noted that the users with a password that was revealed during the dictionary attack were moved from the list of passwords we wanted to reveal during the rainbow table attack. Examining the "Revealed LM?" column in Table 8, we can see that all the remaining *21 passwords* were revealed.

This means that we have revealed *all* the 30 selected passwords by using both a dictionary attack and a rainbow table attack on the LM hashes.

Even though we managed to reveal the 21 remaining passwords we still have to calculate how much time the rainbow table attack took to complete. To calculate the time we included *Total disk access time* and *Total cryptanalysis time* as reported by the Cain & Abel tool. The rainbow table attack, using all the rainbow tables, took $1724.58 + 6369.71 = 8094.29$ seconds = *2 hours 15 minutes.*

Table 6 gives an overview of the timing results from the dictionary attack and from the rainbow table attack. Recall that the rainbow table attack using the LM rainbow tables on the selected password was estimated to take 6 hours 4 minutes. The dictionary attack performed in this phase was carried out on all the 30 selected passwords, while the remaining 21 hash values were subject to the cryptanalysis attack.

Technique	**Time**
Dictionary attack	1 minute 3 seconds (63 seconds)
Rainbow Table attack	2 hours 14 minutes (8094 seconds)

Table 6: Time measurements for the LM hashes

The dictionary attack uses only 0.2% of the available time of 8 hours, while the rainbow table attack constitues 28% of the time.

NT hash
We used Cain & Abel for the dictionary attack on the NT hashes, while Ophcrack was used for the rainbow table attack.

A complete list of the 8 revealed passwords from the dictionary attack is given in the "Revealed NT?" column in Table 8 (indicated by the text " Yes *(dict)*").

Here we experienced a minor difference between the SAM file obtained from the Windows Vista machine and the SAM file from the machine running Windows 7 RC. When we imported the SAM file that originated from Windows 7 RC we noticed that the loaded NT hash values were different than those that were loaded when using the Windows Vista SAM file, even though the same user account passwords were utilized, andwhen we tried to run the dictionary attack no passwords were revealed. At this point we wondered if Microsoft had improved their password handling for Windows 7 RC by for instance including a salt when generating the NT hash value. This turned out to not be the case. We tried to import the Windows 7 RC SAM file into Ophcrack to see if the hash values still were different from the hash values in the Windows Vista SAM file, and in Ophcrack they were the same. This sat aside our theory about improved password security. When we had loaded the Windows 7 RC SAM file in Ophcrack we chose *Save* from the top menu and then *Save to file*. The saved file was then imported into Cain & Abel, which was the tool used for the dictionary attacks. Now, the hash values were the same for both the Windows 7 RC SAM file and Windows Vista SAM file.

We estimated the time for the dictionary attack on the NT hash for the selected password to be 49 seconds. The dictionary attack carried out in this phase with all the 30 selected passwords was measured to take 58 seconds. After deducting the 5 seconds it took for Cain & Abel to start, as we used the *CPU Time*[1] displayed in *Windows Task Manager* to measure the time, the dictionary attack was measured to take *53 seconds*. This shows that it does not necessarily take significantly longer time to run a dictionary attack on a password file containing several NT hash values than it does with a password file containing just one NT hash value.

Let us now consider the *rainbow table attack* with the use of the NT rainbow tables. The users with a password that was revealed during the dictionary attack were moved from the list of passwords we wanted to reveal during the rainbow table attack. *10 passwords* of the 22 remaining passwords were revealed from the rainbow table attack. In other words, the rainbow table attack with the NT rainbow tables was able to reveal 45.5% of the 22 passwords that were subject to the rainbow table attack.

The results presented above show that 8 passwords were revealed during the dictionary attack and 10 passwords were reveled during the rainbow table attack, meaning that 18 of the 30 selected passwords were revealed in total. Table 8 lists all the 30 passwords to give an overview of which of the selected passwords that were revealed. This can be seen from the column labeled *Revealed NT?*. The 18 passwords that were found are listed with *Yes* in the *Revealed NT?* column.

As already mentioned, we have been able to reveal 18 of the 30 selected passwords by using both a dictionary attack and a rainbow table attack on the NT hash values. 27% of the 30 selected passwords were revealed during the dictionary attack while

[1]This was done for the lack of a better timing utility in Windows. Although this practice may be frowned upon, we did perform a sanity check using an ordinary stopwatch, which seemed to indicate that these timing figures were indeed reasonable.

33% of the passwords were revealed during the rainbow table attack. This means that 40% of the NT hash values were not cracked.

The rainbow table attack took *4 hours 26 minutes*. The rainbow table attack using the NT rainbow tables was estimated to take 33 minutes and 20 seconds. Table 7 gives an overview of the timing results from the dictionary attack and from the rainbow table attack. The dictionary attack performed in this phase was carried out on all the 30 selected passwords, while the remaining 22 hash values were subject to the cryptanalysis attack.

Technique	**Time**
Dictionary attack	53 seconds
Rainbow Table attack	4 hours 26 minutes (15960 seconds)

Table 7: Time measurements for the NT hashes

It should be added that we got almost the same results when using the Windows 7 RC SAM file as we did when we used the Windows Vista SAM file. The results from the dictionary attack were identical, and the only difference was that the rainbow table attack performed on the Windows 7 RC SAM file took 5 hours 15 minutes and 28 seconds to complete while, as already presented, the same attack took 4hours and 26 minutes on the Windows Vista SAM file. The exact same passwords were revealed in both cases. This means that the NT hash values stored in the Windows Vista SAM file and the Windows 7 RC file are identical, and that Microsoft has not increased the security to the newly released Windows 7 RC operating system regarding the login passwords that are hashed and stored in the SAM file. Microsoft has for example still not included a salt when generating the NT hash values.

MD5 hash

For the MD5 hash a dictionary attack was the only attack that was applied, using the John the Ripper tool for the whole 8 hours period.

The 8 passwords that were revealed during the dictionary attack constitute 27% of all the 30 selected passwords. This means that the remaining 22 passwords (73%) were not found since the dictionary attack was the only attack performed on the MD5 hash values. Which passwords were revealed can be seen from the column labeled *MD5 Revealed?* in Table 8.

We estimated the time for the dictionary attack on the MD5 hash for the selected password to be 51 minutes and 48 seconds. The dictionary attack carried out in this phase with all the 30 selected passwords was aborted after 8 hours, as this was the time limit set for the experiment. This means that the attack did not complete.

6 Discussion

Figures from the company Net Applications [24] indicate that Windows XP still has more than 60% of the total market share for desktop operating systems. Although there are large uncertainties associated with these figures, it is clear that a very large proportion of laptops in the world today are running Windows XP, and these laptops will by default have LM hashes enabled. Our small experiment revealed that not only can all LM hashes on a computer be cracked in a day, but at two and a quarter hours it can even be performed before most office workers have finished their morning coffee.

Number	Password	**Length**	**Revealed LM?**	**Revealed NT?**	**Revealed MD5?**
1	password1	9	Yes *(dict)*	Yes *(dict)*	Yes *(dict)*
2	abc123	6	Yes *(dict)*	Yes *(dict)*	Yes *(dict)*
3	myspace1	8	Yes *(rbow)*	Yes *(rbow)*	No
4	password	8	Yes *(dict)*	Yes *(dict)*	Yes *(dict)*
5	blink182	8	Yes *(dict)*	Yes *(rbow)*	No
6	qwerty1	7	Yes *(dict)*	Yes *(dict)*	Yes *(dict)*
7	fuckyou	7	Yes *(dict)*	Yes *(dict)*	Yes *(dict)*
8	123abc	6	Yes *(dict)*	Yes *(dict)*	Yes *(dict)*
9	baseball1	9	Yes *(dict)*	Yes *(dict)*	Yes *(dict)*
10	football1	9	Yes *(dict)*	Yes *(dict)*	Yes *(dict)*
11	hm71li	6	Yes *(rbow)*	Yes *(rbow)*	No
12	k0fzug	6	Yes *(rbow)*	Yes *(rbow)*	No
13	fp6jar	6	Yes *(rbow)*	Yes *(rbow)*	No
14	msi89a0	7	Yes *(rbow)*	Yes *(rbow)*	No
15	%l41pc	6	Yes *(rbow)*	Yes *(rbow)*	No
16	m.f0vgk	7	Yes *(rbow)*	No	No
17	v*qbt4un	8	Yes *(rbow)*	No	No
18	qtdra#	6	Yes *(rbow)*	Yes *(rbow)*	No
19	5GCTw4tG@N	10	Yes *(rbow)*	No	No
20	tcC!tW84s	9	Yes *(rbow)*	No	No
21	qaSt4@	6	Yes *(rbow)*	Yes *(rbow)*	No
22	vANe$a	6	Yes *(rbow)*	Yes *(rbow)*	No
23	!aFu9ut	7	Yes *(rbow)*	No	No
24	C7e++AV	7	Yes *(rbow)*	No	No
25	w2U$atHe	8	Yes *(rbow)*	No	No
26	$_Ch6cU5	8	Yes *(rbow)*	No	No
27	Ke42A2Pe*	9	Yes *(rbow)*	No	No
28	rA3$_ey*c	9	Yes *(rbow)*	No	No
29	b*#D9*7yEG	10	Yes *(rbow)*	No	No
30	8rA*_pHa$8	10	Yes *(rbow)*	No	No

Table 8: The results obtained from the various attacks.

LM hash

The results from our attacks on the LM hash show that in this case it did not matter to which category the password belongs because all the passwords were revealed after the completion of both attacks. As intended, almost all of the weakest passwords were revealed during the dictionary attack. As this was the purpose of running the dictionary attack, this part of the experiment has to be considered successful. It is however important to add that in this case it was not really necessary to run the dictionary attack at all, since the character set used for the cryptanalysis attack included every character that is possible to employ in the password. In other words, this means that the rainbow table attack would have revealed all the 30 passwords.

NT Hash

The results from the rainbow table attack on the NT hash show that 10 passwords (33%) were revealed. After the completion of both the dictionary attack and the rainbow table attack, 18 passwords (60%) were revealed in total. This left us with 12 unrevealed passwords (40%). 6 of the 10 passwords that were revealed through the rainbow table attack on the NT hash belong to the category named *Mnemonic Passwords*, and in addition also two passwords from the *Most Popular* category and two passwords from *Random Passwords*were revealed. The 4 remaining passwords from the *Mnemonic Passwords* category that were not revealed are *m.f0vgk* (7 characters), *v*qbt4un* (8 characters), *5GCTw4tG@N* (10 characters) and *tcC!tW84s* (9 characters). The last two are the passwords that were produced by us, and both of them have corresponding phrases that are easy for us to remember even though they consist of many characters. The maximum password length that is supported by the *Vista special* tables is 8 characters, and the corresponding character set is quite limited as only numbers from 0 to 9 and lowercase letters are supported. As *5GCTw4tG@N* is 10 characters long it was expected that this password would not be revealed during the rainbow table attack. The *tcC!tW84s* password consists of 9 characters. This is also too long for the *Vista special* tables to crack. The other two passwords (*m.f0vgk* and *v*qbt4un*) that were not revealed by the *Vista special* rainbow tables are both within the supported password length of the *Vista special* tables, but they have different character sets because they have different length. Both of these passwords consist of lowercase letters, one number and one special character. The only thing that prevents both of them from being revealed by the rainbow tables is the special characters.

If we take a look at the *Random Passwords* category there were only two passwords (*qaSt4@* and *vANe$a*) that were revealed by the rainbow table attack. The reason why these two passwords were revealed is that they both consist of only 6 characters. If they had been one character longer, they would not have been revealed because the character set for passwords of 7 characters in length do not include special characters. Both of the passwords contain a special character. They are also the only passwords in the *Random Passwords* category that are just 6 characters in length, and the other passwords are not revealed by the rainbow table attack.

The two passwords *myspace1* and *blink182* (that were not revealed by the dictionary attack) were revealed by the rainbow table attack. Both *myspace1* and *blink182* consist of 8 characters, which is the longest password length supported by the *Vista special* tables. These passwords were revealed because they only consist of lowercase letters in addition to numbers. If for example only one of the characters had been an uppercase letter, the passwords would not have been revealed. It has now been shown that small differences might determine whether a password is going to be revealed or not.

Windows 7: Nothing New Under the Sun

The same results were obtained when the dictionary attack and the cryptanalysis attack were performed on the SAM file that originated from the Windows 7 RC operating system. This means that also the same discussions applies for the NT hashes stored in the Windows 7 RC SAM file.

MD5 hash

It may be interesting to note that although the MD5 hash function is considered “broken”, this of course does not impact the success rate of performing a dictionary attack against MD5 hashes. Thus, we see that the dicrionary attack against MD5 revealed exactly the same passwords as were revealed by the dictionary attack against the NT hashes.

Time and resource contraints prevented us from also performing a rainbow table attack against MD5.

Dictionary Attacks vs. Rainbow Table Attacks

If we were to compare the use of a dictionary attack with a rainbow table attack we would highlight that while the dictionary attack requires a relatively small wordlist, the rainbow tables are generally a lot bigger, requiring a quite large storage capacity. Rainbow tables must either be downloaded or generated in advance of the attack, and this may take quite a long time depending on your download speed and the system you are using. It is important to be aware that attackers with ill intentions probably would invest a big effort to recover the password.

Another observation is that when it comes to cracking passwords with the use of rainbow tables, it does not matter if the password is strong or weak as long as the password is within the requirements sat for the particular rainbow tables.

The results obtained from our experiment show that even though it is important to choose a strong password, this may not be sufficient by itself. The underlying system also influences how easily the password can be recovered. A good example of this is Windows XP, which utilizes the LM hash together with the NT hash. Because of the weaknesses with the construction of the LM hash, the rainbow tables can support a character set consisting of all the characters necessary to reveal all the passwords. The results obtained from our study, showing that all the 30 selected passwords may be revealed using rainbow tables, confirms this statement. It may also be confirmed by investigating the character set as well. To avoid the case illustrated in this example you should either disable the use of LM hash or you should use a password longer than 14 characters if you are using Windows XP. When the password is 15 characters or longer an operating system that uses both LM hash and NT hash will not store the LM hash, only the NT hash.

This study shows that a big portion of the selected passwords were revealed, even with computers that are not particularly powerful. As the storage capacity and processor power of mainsteam computers keeps increasing, more computations can be performed in a shorter amount of time. Even though the use of a salt in the generation of the password hash will probably offer sufficient protection from password recovery for a while, it will only be a temporary respite. It has been generally known even before Klein's studies [25] that it is important to choose a good password – our findings indicate that in the near future, it will not matter how good your password is; it'll get cracked anyway.

7 Conclusions and Further Work

The password as an authentication mechanism is headed for obsolence, as the password lengths required to thwart rainbow table attacks are rapidly approaching unmanageable (or unrememberable) proportions. If you are still using the LM hash on your laptop, you might as well put your passord in a cleartext file and call it "password.txt" - according to our results, anyone who wants your password will have it by the end of the working day. Even though we were not able to crack all the NT hashes, it seems that it is only a matter of time before rainbow tables for all practical password lengths will be generally available also for the NT hash.

An obvious opportunity for further work would be to employ rainbow tables for MD5 to see how a vanilla Linux distribution bears up under a cracking attack. It could also be interesting to see if discovered MD5 vulnerabilities could be translated into a practical attack. More fundamental contributions could be made in finding alternatives to passwords that are more secure, but with the same level of social acceptance and ease of use.

Acknowledgements

The research for this paper was conducted as part of an MSc thesis at the Norwegian University of Science and Technology (NTNU).

References

[1] M. Wilkes, *Time - Sharing Computer Systems.* New York, NY, USA: American Elsevier, 1968.

[2] R. Morris and K. Thompson, "Password security: a case history," *Commun. ACM*, vol. 22, no. 11, pp. 594–597, 1979.

[3] M. Luby and C. Rackoff, "A study of password security," *J. Cryptol.*, vol. 1, no. 3, pp. 151–158, 1989.

[4] E. Glass, "The ntlm authentication protocol and security support provider," sourceforge.net, 2006. [Online]. Available: http://davenport.sourceforge.net/ntlm.html\#theLmResponse

[5] M. G. Jaatun, "LMHASH SPOOFING VHA SAMBA KLIENT," *FFI note*, 2000.

[6] J. M. Johansson, "The great debates: Pass phrases vs. passwords. part 1 of 3," 2004. [Online]. Available: http://technet.microsoft.com/en-us/library/cc512613.aspx

[7] *FreeBSD Handbook*, 2009. [Online]. Available: http://www.freebsd.org/docs.html(June2009)

[8] U. Drepper, "Unix crypt using sha-256 and sha-512," people.redhat.com, 2008. [Online]. Available: http://people.redhat.com/drepper/SHA-crypt.txt

[9] R. Rivest, "The MD4 message-digest algorithm," IETF Request for Comments (RFC) 1320.

[10] ——, "The MD5 message-digest algorithm," IETF Request for Comments (RFC) 1321, Internet Activities Board, Internet Engineering Task Force.

[11] *Secure Hash Standard*, National Institute of Science and Technology Std. Federal Information Processing Standard (FIPS) 180-1.

[12] X. Wang, Y. L. Yin, and H. Yu, "Finding collisions in the full SHA-1," *Lecture Notes in Computer Science*, vol. 3621, pp. 17–36, 2005, Proceedings of Crypto 2005.

[13] M. E. Hellman, "A cryptanalytic time - memory trade-off," *IEEE TRANSACTIONS ON INFORMATION THEORY*, vol. IT-26, pp. 401–406, 1980. [Online]. Available: www-ee.stanford.edu/~hellman/publications/36.pdf

[14] P. Oechslin, "Making a Faster Cryptanalytic Time-Memory Trade-Off," in *The 23rd Annual International Cryptology Conference, CRYPTO '03*, ser. Lecture Notes in Computer Science, vol. 2729, 2003, pp. 617–630. [Online]. Available: http://infoscience.epfl.ch/getfile.py?recid=99512\&mode=best

[15] "John the ripper password cracker," openwall.com, Openwall Project, 2009. [Online]. Available: http://www.openwall.com/john/

[16] RainbowCrack, "Project rainbowcrack," project-rainbowcrack.com, 2009. [Online]. Available: http://project-rainbowcrack.com/

[17] M. Montoro, "Cain & abel," oxid.it, 2009. [Online]. Available: http://www.oxid.it/cain.html

[18] "Rainbow tables," shmoo.com, The Shmoo Group, 2006. [Online]. Available: http://rainbowtables.shmoo.com/

[19] Ophcrack, "What is ophcrack?" sourceforge.net, 2009. [Online]. Available: http://ophcrack.sourceforge.net/

[20] B. Schneier, "Myspace passwords aren't so dumb," schneier.com, 2006. [Online]. Available: http://www.schneier.com/essay-144.html

[21] "The university's password generator page," aber.ac.uk, Aberystwyth University, 2000. [Online]. Available: http://users.aber.ac.uk/auj/portfolio/mnemonic.shtml

[22] "Secure password generator," pctools.com, PC Tools, 2009. [Online]. Available: http://www.pctools.com/guides/password/

[23] "Virtualbox," virtualbox.org, Sun Microsystems, Inc, 2009. [Online]. Available: http://www.virtualbox.org/

[24] "Operating System Market Share." [Online]. Available: http://marketshare.hitslink.com/

[25] D. V. Klein, "Foiling the cracker: A survey of, and improvements to password security, (revised paper with new data)," in *14th DoE Computer Security Group*, May 1991. [Online]. Available: http://www.klein.com/dvk/publications/passwd.pdf