

# CyberP3i Course Module Series

Spring 2017

Designer: Dr. Yeşem Kurt Peker, *Assistant Professor*

## Password Cracking



# Password Cracking

---

## Module Objectives

Upon completion of this module students will be able to

- Explain the mechanisms used for storing passwords
- Explain how on line, offline and non-electronic attacks can be used to recover passwords
- Compare and contrast different methods for cracking passwords
- Describe how salting when storing passwords helps prevent certain attacks

## Introduction

Passwords are the most common mechanism used for authenticating users in a system. All users of technology use passwords; in fact, most have several passwords. Almost always users choose their own passwords and have to remember the password they picked for a particular site when they need to authenticate themselves. This is an innate vulnerability with passwords because users need to choose passwords that they can remember. The attackers take advantage of this and exploit the common convenient ways users use to choose their passwords. Given that human is the weakest link in information security, the mechanisms and procedures for selecting and securely storing passwords become extremely important.

In cryptanalysis and computer security, password cracking is “the process of recovering passwords from data that have been stored in or transmitted by a computer system.” ([Wikipedia](#)) To understand the methods for recovering passwords, we need to first understand how the authentication via passwords works. Very simply, the password of the user needs to be stored in a database or file in some form and when the user enters their login name and password, the stored entry should be verified. If the entry verifies then the user is allowed access otherwise denied access to the system. In this module we will first describe the mechanisms used for storing passwords. Then we will give an overview of different methods for cracking, or more appropriately “recovering” passwords. The focus of password cracking will be on offline password cracking methods.

## How passwords are stored

Of ten we hear “passwords are encrypted” about storing passwords. This is not an accurate use of the word encryption as it calls for decryption. The more appropriate phrase would be “hash of a

password is stored". (In fact, the password is also salted before hashed. We will explain this in detail in the Offline Password Cracking section below). Hash of a password cannot be "decrypted", or in equivalent terms, inverted back to the plain password by an algorithm that reverses the hash function used. Indeed, it is the promise of hash functions that the hashes are NOT invertible. Note that there is no key involved with hash functions. This is in contrast to encryption/decryption where a key is required. (There is a separate concept called "keyed hash functions" which is used for message authentication codes but these are not used for passwords). Here is a brief description of hash functions and their properties and what they mean for passwords:

## Hash Functions

A hash function maps digital data of arbitrary size to digital data of fixed size. A cryptographic hash function is a hash function which is considered practically impossible to invert (one-wayness) or find collisions (i.e. two messages with the same hash value).

Characteristics of a cryptographically secure hash function

- *Variable input/ Fixed output size:* Can be applied to data of practically any size but the output is always a fixed number of bits
- *Pre-image resistant:* Computationally infeasible to find the input value for a given hash value (one-wayness)
- *Collision resistant:* Computationally infeasible to find two inputs that have the same hash value
- *Efficiency:* Easy (fast) to compute so practical on hardware and software
- *Pseudorandomness:* The outputs meet the tests for pseudorandomness (exhibits characteristics of random numbers)

Preimage resistance property implies two things:

1. One cannot recover the original password from the hash value. This is in contrast to encryption where with the help of the key one can recover the original.
2. Given a hash value, one cannot find a password that maps to the same hash value

Collision resistance property implies one cannot find two passwords with the same hash value. This property assures that the hashes for two distinct passwords will be different. The pseudorandomness property along with preimage and collision resistance assert that even if the two passwords differ in one bit their hashes will be very different –so different that looking at the hashes one cannot tell how the two passwords differ. The other two properties of fixed size output and efficiency are crucial for practical purposes.

Common hash functions used for storing passwords are SHA and MD5 algorithms.

In the Windows operating system, passwords on the local system are stored in the SAM file, while Linux stores them in the /etc/shadow file. These files are accessible only by someone with root/sysadmin privileges. In both cases, one can use a service or file that has root/sysadmin privileges to grab the password file.

## Cracking Passwords

The methods for cracking passwords can be divided into three areas of non-electronic, online, and offline. Non-electronic methods do not require technical knowledge. Online password cracking is necessary when one doesn't have access to the password hashes. Offline password cracking is possible when password hash(es) are available. After describing the methods for non-electronic and online password cracking this module will focus on offline password cracking methods.

### Non-Electronic Methods

Also known as non-technical methods, non-electronic methods do not make use of technology but social and stealthy skills. Some examples are:

1. Social engineering: Convincing people to reveal passwords
2. Shoulder surfing: Looking at either the user's keyboard or screen while he/she is logging in.
3. Dumpster diving: Searching for sensitive information at the user's trash-bins, printer trash bins, and user desk for sticky notes

### Online Password Cracking

Online password cracking usually means guessing the password when presented with a web form asking for a username and password combination. We refer to this as an "active" online password cracking method. When one has access to the network or the Internet, then other methods can also come into play to recover passwords. Below is a list of different types of online password cracking methods.

1. Passive Online Methods: Passive methods do not require communicating with the authorizing party to crack passwords. Some examples are:
  - a. Eavesdropping on network password exchanges: run packet sniffer tools on the local area network (LAN) to access and record the raw network traffic.
  - b. Man in the middle attacks: acquires access to the communication channels between victim and server to extract the information.
  - c. Replay attacks: Packets and authentication tokens are captured using a sniffer. After the relevant info is extracted, the tokens are placed back on the network to gain access.

2. Active Online attacks: Active methods require directly communication with the victim machine.
  - a. Guessing the password
  - b. Dictionary and Brute Forcing the password
  - c. Phishing
  - d. Malware (Keyloggers, Trojan, spyware)

In this module we focus on active online methods; in particular, guessing the password or automating the guessing process by use of a dictionary or brute forcing. Guessing the password is manually entering passwords in the login page.

This is not very effective because it is very slow. There are password-cracking tools, many freely available, to automate the task of guessing passwords. These tools can try out all combinations of possible characters that might appear in a password or all words in a dictionary of common passwords or their slight modifications. The downside of performing an online attack is that it can be very noisy, extremely slow and sometimes just not feasible. Many login forms have measures in place to prevent automatic password guessing. For example lockout feature locks the user out after a certain number of failed login attempts. The system may lock the account for a certain amount of time or completely block the IP address where the attempts are originating from in which case the user needs to contact the customer support to have the IP address unblocked. Even if the target site doesn't have a lockout feature, because of the noise so many wrong password attempts generate in the system, its log file will grow tremendously. It looks very suspicious when there are hundreds of wrong password attempts logged to the same IP address. One way around this would be to cover up the IP address via a proxy and use a different proxy for every so many guesses (as many as the system allows) or attempt a few guesses every so many minutes or so to look less suspicious. Many of the online password-cracking tools have these features available.

Online password cracking can be very slow because the speed of the method depends on the speed of the internet connection and the speed of the target server. Because of this and because brute forcing would include too many possibilities to try, the most effective way in an online method would be to use a dictionary of possible passwords. This can further be enhanced by using social engineering to customize the list of possible passwords for a user. More details on brute forcing and dictionaries are given in the Offline Password Cracking section below.

## Offline Password Cracking

As was described in the How Passwords Are Stored section, passwords are not stored in clear text in the system. The hashes of the passwords are stored. Offline password cracking assumes that the hash or hashes of the passwords that need to be cracked are available. In particular, this may be the password file from a system with the hashes of passwords of all users. The cracking is done on one's own system or on systems that the cracker have access to. Unlike an online attack, there are no locks or other restrictions to stop the cracker in an offline attack. The only limitation is the limits of the computer hardware the cracker is running on. So the better the processor and nowadays even graphics card, the more password guessing attempts a one can get per second. To speed up the cracking, distributed networks can be used where several machines run the cracker at the same or different times.

The main idea in offline password cracking is, given a hash value, to hash passwords one by one until a password whose hash matches with the given hash is found. Note that, even though highly unlikely due to the collision resistance property of hash functions, the actual password may not be the same as the password whose hash matches with the given hash. Nevertheless, the password found will work, because, for authentication into the system the hash of the password will be compared with the given hash.

So which passwords should one try?

### Brute Force Method

Brute force password cracking attempts all possibilities of all the letters, number, special characters that might be combined for a password. It is the most time consuming approach to password cracking. The more computing power one has, the more successful they will be this approach.

### Dictionary Method

In this method a dictionary of common passwords and possible passwords that are likely to be used by humans is used. Such dictionary files are available on the Internet. All words in the dictionary are tried out for a possible password. This method takes advantage of the fact that some users will choose convenience over security and choose usual passwords or common word or phrases as their passwords.

### Hybrid Method

This method is similar to the dictionary method but the words in the dictionary are expanded using common patterns that people use to choose their passwords. For example, adding a digit or an exclamation mark at the end of a dictionary word; or substituting Os by zeros, ls by ones, etc. If a dictionary attack looks for "password", a hybrid attack might look for "p@\$w0rd123".

## Rainbow Tables

The offline approaches we have described so far, namely brute force, dictionary, and hybrid methods are very time- and CPU-intensive. A faster approach is to take a table with all the words in the dictionary already hashed and compare the hashes from the password file to the list of hashes in the table. If there is a match, the password is found. The table of pre-computed hash values is called a rainbow table. There are rainbow tables available on the Internet as well as tools to create rainbow tables (for example rtgen, Winrtgen)

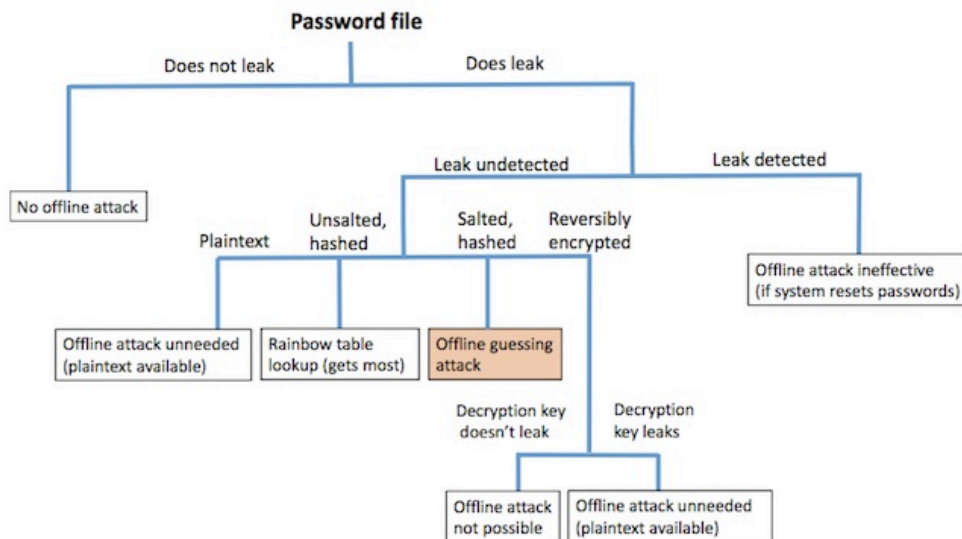
Rainbow tables reduce the complexity of cracking even strong passwords considerable; it reduces the process to essentially a search (of sorted items). To protect against rainbow hash tables, a technique called salting is used.

## Salting

Password salting is a technique where random string of characters are added to the password before calculating their hashes. The salt is also stored along with the hash of the password in the password file or database because, otherwise, the system would not know which random number was added to the string before hashing. The advantage is that with the salt the pre-computed hash tables, i.e. rainbow tables are useless. The tables would need to include hashes of all possible salted passwords which becomes impractical quickly. The size of the salt is usually 8 bytes (hence  $2^{64}$  different salt values) which requires to increase the size of the table by  $2^{64}$ . If each hash value is 160 bits (assuming SHA-1 is used), this would require  $2^{64} \times 160$  bits of storage which is 368,934,881 terabytes. Note that this is the amount of data that needs to be stored for one password only.

If salting is used in storing passwords in a system, then one goes back to brute forcing , dictionary, and hybrid attacks and use the salt that comes with the password hash as they are applying the hash functions.

The diagram below depicts the process of cracking passwords offline:



Source: <http://www.securityweek.com/brute-force-attacks-crossing-online-offline-password-chasm>

## Some Password Cracking Tools

- **John the Ripper:** Free open source password cracking tool for Linux, Unix and Mac OS X. A Windows version is also available. This tool can detect weak passwords. A pro version of the tool is also available.
- **L0phtCrack:** L0phtCrack is a password auditing and recovery application packed with features such as scheduling, hash extraction from 64-bit Windows versions, and networks monitoring and decoding.
- **Ophcrack:** Ophcrack is a Windows password cracker based on rainbow tables. It comes with a Graphical User Interface and runs on multiple platforms.
- **Cain & Abel:** It allows recovery of various kind of passwords by sniffing the network, cracking encrypted passwords using dictionary, brute-force, and cryptanalysis attacks.
- **RainbowCrack:** RainbowCrack cracks hashes with rainbow tables. It uses time-memory tradeoff algorithm to crack hashes.
- **THC Hydra:** Fast network logon password cracking tool



## References

<https://null-byte.wonderhowto.com/how-to/hack-like-pro-crack-passwords-part-1-principles-technologies-0156136/>

<https://null-byte.wonderhowto.com/how-to/hack-like-pro-crack-passwords-part-1-principles-technologies-0156136/>

[https://ktflash.gitbooks.io/ceh\\_v9/51\\_cracking\\_passwords.html](https://ktflash.gitbooks.io/ceh_v9/51_cracking_passwords.html)

## Assessment Questions

1. What are the advantages of storing hashes of passwords instead of passwords themselves? Can you think of any disadvantages? Explain.
2. What are the advantages of salting the password before hashing it for storage? Can you think of any disadvantages of salting? Explain.
3. Describe three attacks that do not require any technical skills to crack passwords.
4. Explain the limitations in online password guessing.