Password Encyption
Attacks & Prevention

Garrett Pearsall and Jeffrey Zhang

November 5, 2024
Version 1.2

Summary

We generally prefer to keep passwords that are simple and easy to remember but that ease helps attackers to crack it very easily. In this tutorial we can understand how passwords are exposed using different tools like John The Ripper. This guide helps us understand how important it is to have passwords that are strong enough and makes hacking difficult. We also go through different challenges and tasks for better understanding of the concept. A discussion on safe passwords is also included.

Contents

1 Problem Description   >

How can users access data securely and prevent unauthorized users from accessing their personal information?

It is important for all of us to understand why is it important to have a secure password. Every application needs some authentication verification to keep our information safe. Easy passwords give a lot of scope for an attacker to crack and steal private data. It is very important to have passwords that are complex so that they cannot be cracked quickly.

2 Real-World examples ⟨⟩

1. 500 million yahoo user accounts were hacked in 2014 [3]

2. GitHub has become the latest target of a password reuse attack [4]

3. LinkedIn is hacked in 2012 and the emails, unsalted SHA1 encrypted passwords, and personal information of 117 million users are released on the dark web [22]

---

1. Apart from the attack that happened on Yahoo in 2014 there was a different attack in 2013 which hacked around 1 billion accounts. The two attacks are largest known security breaches. The 2013 attack involved sensitive user information like names, phone numbers etc. Yahoo users were affected by both the attacks. Critics say that yahoo was very slow in adapting the security measures even after the breach. Yahoo's security officer, said in a statement that the attacker had stolen Yahoo's proprietary source code which gave access to its users accounts. [3]

2. GitHub has become the latest target of a password reuse attack. According to VP of Security at GitHub, an unknown attacker using a list of email addresses and passwords obtained from the data breach of other online services made a significant number of login attempts to GitHub's repository on June 14. Github administrators reviewed the logins and found that attackers got access to its users accounts. The megabreaches of Linkedln, MySpace, Tumblr, and the dating site Fling might be because of dumping 642 million passwords. The company adviced its users to practice good passwords and use two-factor authentication for keeping their account safe. [4]

3. In the LinkedIn attack, the attacker hacked most of the passwords in 72 hours and tried to sell the information. A security researcher reached out to some of the victims and got to know that they were using the same password at the time of breach. Officials of LinkedIn suggested not to store passwords in insecure manner, also to change the passwords once in a while and not to use same password for all the applications. They strongly recommended users to use strong passwords and two-factor authentication for keeping your account secure [22].

3 Salting Techniques <>

1. Salt is random information appended to a password to increase its hashed complexity

2. A proper salt helps mitigate both rainbow-table lookups and dictionary attacks

3. Each user should be given a unique salt. This one-use number is sometimes referred to as a nonce.

Salts are generally used to safeguard the passwords. The password and salt are concatenated which is processed with cryptographic hash function/ one-way hash function and the result is stored in salt database. *nix uses a password file (/etc/passwd) to store the hashes of salted passwords and a shadow file (typically /etc/shadow) holds additional user information including the salt

Problems with Salt:

Public salt/salt reuse: Using same salt for different hashed passwords. This will make rainbow tables useless. Using a single salt makes it easier to attack multiple accounts by cracking one hash

Short salt: If the salt is short , attacker can create rainbow tables with every possible combination so a large salt is always recommended.

[15]

Nonce: In cryptographic communication a nonce is a arbitrary number which can be used only once. Authentication Protocols uses nonces to ensure that old communications cannot be reused [18].

4 Windows Passwords (LM and NTLM)   <>

LM (Lan Manager):

1. Developed by IBM and Microsoft

2. Uses server message block protocol

To overcome some of the disadvantages of LM a new method called NTLM was introduced. Though it is better than LM, it is still weak and can be brute-forced easily.

LM hash algorithm:

1. Weak method of hashing

2. Can crack hashes in seconds using rainbow tables

Trouble with LM password Hashes:

- Passwords are truncated at 14 characters.

- Passwords are converted to all uppercase.

- Passwords of fewer than 14 characters are null-padded to 14 characters.

- The 14-character password is broken into two seven-character passwords that are hashed separately.

NTLM hashes are more difficult to crack than LM. Length and complexity of password matters. If the hashing function is complex it takes decades or even life time to hack and NTLMv2 uses RC4. NTLM is not salted and the types of encryption can be dictated by group policy and active directory settings. There are two versions of NTLM:

- LM and NTLMv1 use DES

- NTLMv2 uses MD4/MD5

[15]

Jeremi Gosney released a research paper called Exacerbating Global Warming at the Oslo Password Hacking Conference showing that any NTLM hash can be cracked in under 6 hours. [23]

5 LM and NTLM Hash Example [15]   <>

User Name: Administrator
ID: 500
LM Hash: e52cac67419a9a224a3b108f3fa6cb6d
NTLM Hash: 8846f7eaee8fb117ad06bdd830b7586c

User Name: Georgia Weidman
ID: 1000
LM Hash: aad3b435b51404eeaad3b435b51404ee
NTLM Hash: 8846f7eaee8fb117ad06bdd830b7586c

Note that the LM Hash aad3b435b51404eeaad3b435b51404ee signifies that it is empty and instead exclusively contains an NTLM hash.

6 Password Cracking Attacks <>

Passwords can be cracked using:

1. Brute-force attacks

2. Dictionary attacks

3. Precomputed tables

4. "Intelligent" brute-force attacks (e.g. Markov chains)

5. Group Policy Attacks

Passwords generally are (and should be) stored as a hashed message digest instead of plaintext [16]. As such, the attempts for each of these attacks will generally need to be hashed using the same algorithm to match the corresponding stored password hash. This is because cracking attacks are often run against password files that the attacker apprehended [7].

1. Brute-force attacks aim to crack passwords by trying every possible permutation of a password. For example, a brute-force attack against a four character password would begin with "aaaa" and attempt every possible iteration up until "zzzz". This attack is simple and effective, so long as there is enough time to run through the permutations. The drawback of brute-force method is, with each additional character added to the password, the time it takes to crack it grows exponentially. [16]

2. Dictionary attacks attempt to match words from a given wordlist against a set of passwords [16]. This method is more sophisticated than a brute-force search because, instead of generating the passwords character-by-character, a list of likely character combinations can be used (e.g. "password").

3. Rainbow tables represent one type of pre-computed dictionary attack. Tables store large amounts of passwords and their corresponding hashes. A rainbow table attack uses a hash value taken from the list and reduces it using a *reduction function* to match against passwords in the table. If a reduction matches a hash, the password may be found by generating a chain of hashes to find a match with one of the pre-computed hashes. Since rainbow tables incorporate the same hashing algorithm as the password storage, they can be defeated if the administrator includes a "salt" or a random value added to the password before hashing. This makes it so identical passwords can have different hashes, frustrating the attack.

4. An example of an intelligent brute-force attack is an attack using a Markov chain. By this method, probabilities for each position of the password can be determined by using previously-cracked passwords [16]. The rules we write in this tutorial for john the ripper are another example.

5. Group Policy Attacks A key component of active directory is group policy to manage security, user settings etc. Group policy settings are divided into user and computer sections. The vulnerability is a fundamental design flaw in Group Policy that remained undiscovered for at least a decade. They reported it to Microsoft in January 2014. To fix it, Microsoft had to re-engineer core components of the operating system and add several new features. Microsoft addressed the remote code execution flaw with the MS15-011 security bulletin, but also fixed a related Group Policy security bypass issue in MS15-014. Microsoft security engineers explained in a blog post that attackers would be most likely to exploit these vulnerabilities by using techniques like ARP spoofing on a local network in order to trick computers to accept and apply bad Group Policy configuration data from servers under their control. A longer password length is important to both administrators creating policy and users protecting their information [5] [6].

7 Password Security Guidelines <>

Administrators should:

- Never store plaintext passwords

- Use a slow hashing algorithm to hash *salted* passwords

- Incorporate complexity and minimum length requirements for password creation

Users should:

- Use strong, unique passwords

- Use password manager and two-factor authentication

Traditional advice for password complexity requires that the password be at least 12 characters long, consisting of letters (upper and lowercase), numbers, and special characters. In addition, it should not be a dictionary word and shouldn't have obvious substitutions. To be even more secure, users should consider using a *password manager*. Password managers allow you to store strong, randomized, unique passwords for every system you connect to. [8] To avoid having a single point of failure due to the master password, methods such as two-factor authentication can add an extra layer of security [9]. Additionally, many password managers which store encrypted passwords locally can be configured to require the use of a key file in conjunction with a master password to unlock all of a user's saved passwords. This is a common way to prevent a single point of failure when storing passwords on a local machine.

8 Types of Password Attacks   <>

- Online Password Attacks

  - Generally require more research and educated guesses from the attacker

  - These can be mitigated by limiting requests and protecting user information

- Offline Password Attacks

  - Involve an attacker gaining access to hashed passwords

  - This tutorial focuses primarily on offline attacks

It isn't easy because hashes are product of one-way hash function: Using hash function with an input an attacker can calculate the output but if the hashing algorithm is strong it cannot be similarly reversed. However, an educated guess can be made and hashed. This hash can be compared to the known hash.

Meterpreter has a `hashdump` command which attempts to print the hashed Windows passwords if it has access.

meterpreter >hashdump [15]

This method will trigger most antivirus software. A metasploit script called smart_hashdump exists which attackers can use to avoid this.

meterpreter >run post/windows/gather/smart_hashdump

The results will be saved in the `$user/.msf3/` folder. For convenience we will use this folder for the remainder of this tutorial.

9 Recovering Password Hashes from a Windows SAM File <>

- Windows Security Manager

- It stores hashed Windows passwords

- It is not that easy to recover password hashes from a SAM file

- The SAM file is encrypted by the Windows Syskey utility using a 128-bit RC4 Cipher

- Windows sometimes stores backups of the SAM file in the $Windows\repair folder which does not have elevated privileges and can be exploited by attackers.

[15]

10 Challenge 1: Crack a basic MD5 Password hash using John the Ripper <>

1. First, generate an MD5 hash of a simple password, e.g., "password123."

2. This will output something like: **482c811da5d5b4bc6d497ffa98491e38**.

3. Create a file (hashes.txt) containing the hash you just generated.

4. Run John with the following command: **john --wordlist=rockyou.txt hashes.txt**

(Hint: Try doing **echo -n "password123" | md5sum >> hashes.txt** as an example. If that doesn't work initially, try **touch hashes.txt** first.)

(Hint: pipe the output from the echo using **>>**. If you need to create the file use **touch hashes.txt**, then pipe the output in).

Use a pre-existing dictionary wordlist (e.g., the default rockyou.txt).

After running, John will attempt to crack the hash using the dictionary. If the password "password123" is in the wordlist, John will crack it and display the result.

Time: 5-15 minutes

11 John the Ripper <>

- A fast, free, multi-platform, open-source password cracker

- Will allow us to try out brute-force, dictionary, and rule-based attacks

Example modes:

- Brute Forcing

- Single

- Wordlist with rules

- Incremental

John the Ripper (hereafter referred to as "john") is a free, open source password cracking utility. It was initially designed to uncover weak passwords on Unix systems, but is also used in a wider scope for general password cracking. It is available for both *nix and Windows-based systems and can crack password hashes from different hashing algorithms (e.g. crypt(3) for Unix, LM for Windows). [10] john is typically run from the command line, but GUI interfaces exist for it (e.g. "Johnny" on Kali Linux).

In lieu of running any additional command line options, executing **john <pass.txt>** will run john in the following order of cracking modes: single crack, wordlist with rules, and incremental. Single crack mode will try passwords based off user names, word-mangling, and previously cracked passwords. Wordlist mode incorporates rules specified in john's configuration file (or by the user) and runs a wordlist against password hashes in the list. A is just a text file with each line representing a different word that can be a password. This is akin to the dictionary attack discussed previously. If a wordlist is not specified, john will use its default wordlist. Finally, incremental mode tries every different possible combination of characters for the password. It is essentially a brute-force attack. [10]

12 Example John the Ripper Commands `<>`

`--show`

`--single`

`--rules`

`--wordlist=FILE`

`--external=MODE`

`--incremental[=MODE]`

`--save-memory=LEVEL`

To get a list of other available options visit `http://www.openwall.com/john/doc/OPTIONS.shtml`

- Shows the cracked passwords for given password files. You can use this option while another instance of John is cracking to see what John did so far

- Enables the "single crack" mode, using rules from the configuration file section

- Enable word mangling rules for wordlist mode

- These are used to enable the wordlist mode that is read words from FILE, or from stdin

- Enables an external mode, using external functions defined in section

- Enables the "incremental" mode, using the specified configuration file definition. If MODE is omitted, the default is "ASCII" for most hash types and "LM_ASCII" for LM hashes.

- You might need this option if you don't have enough memory or don't want John to affect other processes too much or don't need it to load and print login names along with cracked passwords. Level 1 tells John not to waste memory on login names; it is only supported when a cracking mode other than "single crack" is explicitly requested. Levels 2 and 3 reduce use of performance optimizations involving large lookup tables, and thus have a negative performance impact.

[17]

13 Activity: Physical Access <>

For this exercise (and forensic purposes) we will preserve the data integrity of our Windows 7 machine

1. Log into your Windows 7 VM and create 5+ user accounts of varying types and password strengths.

2. Log into your Kali-Everything VM and create 5+ user accounts of varying types as well.

There should be only the Windows 7 VM and the Kali Linux VM within your lab folder. We will be using WinSCP and OpenSSH to make the machines to talk to each other.

Duration: 10-15 minutes

14 Activity: Physical Access 2 <>

- On the Kali Everything machine, start the OpenSSH service (server-side):

  `sudo systemctl start ssh`

- On the Windows 7 machine, Run the WinSCP Shortcut. Input **192.168.14.201**(The Kali-Everything's IPv4 address), port 22

For WinSCP, we just left the protocol "SFTP" as the default, since it shouldn't matter as long as we specified port 22 that we opened up.

Duration: 5 minutes

15 Challenge 2: Saving the Windows 7 hashes <>

Without altering the integrity of the file system, transfer the contents of the hash dump to your other active Kali VM

- You can view the hashdump using `samdump2`:

  samdump2 SYSTEM SAM

- These files can be lengthy and you don't want to save anything locally

- You should have 2 files (system, sam) containing the password hashes in your Kali VM at the end of this exercise

Hint: Both of these systems are on the same network

Remember this is a forensic investigation. Be careful to not compromise the integrity of the Windows file system!

Duration: 10-15 minutes

16 Challenge 3: Offline Password Attack 2 <>

Now that you have the hash dump from the Win7 machine in Kali, use John the Ripper and wordlists to attempt to crack the passwords you created.

- This Windows 7 machine uses NTLM so you will have to change the **--format** flag when running john

- As before, write down what you were able to crack and with what settings. What led to each password being vulnerable?

- Are there any passwords you don't think could be cracked in a reasonable amount of time?

Hint: john –format = " " (refer john for commands)

Duration: 15 minutes

17 Linux Passwords <>

- Traditionally, UNIX account information and passwords are stored in **/etc/passwd**

- This creates a security concern because **/etc/passwd** is a world-readable file used by many command line tools

- To mitigate this risk, some distributions store password hashes in the(**/etc/shadow**) file which can only be accessed by a super user

- This does not prevent super users from directly accessing the password hashes

Windows does not allow access to SAM files while a user is logged in irrespective of the permissions of their account. While *nix has the advantage of salting passwords, Windows has a more effective permission scheme and both systems still have inherent flaws.

In addition Linux passwords can be encrypted using a number of algorithms. The hashdump will actually tell us which encryption method was used:

- $1: MD5

- $2: blowfish

- $2a: eksblowfish

- $5: SHA-256

- $6: SHA-512

- $y: Yescrypt

Traditional Unix systems store user names, one-way encrypted passwords, and user attributes (e.g. User ID, home directory, login shell) in **/etc/passwd**. Many Unix utilities use the information in the file to execute properly, therefore **/etc/passwd** needs to be *world-readable*. Since it is world-readable, this leaves the password hashes in a vulnerable position. To address this vulnerability, many Linux distributions store the password hash portion of **/etc/passwd** in a *shadow password* file, **/etc/shadow**. When a distro has shadow passwords enabled, **/etc/passwd** is used as before, except the password field is replaced by an 'x'. **/etc/shadow** also may include useful information regarding user passwords such as number of days since a password change and number of days after expiry before an account is disabled. [1]

18 Unshadowing the Password File   <>

- John the Ripper includes an `unshadow` command:
  $ unshadow /etc/passwd /etc/shadow

- This can be run while logged into a *nix system

Feel free to see what happens when you unshadow the passwords in Kali

john, being originally designed to detect insecure Unix passwords, includes a utility called **unshadow** that combines **/etc/passwd** and **/etc/shadow**, effectively unshadowing the passwords and matching them up with their associated user and user attributes. Once this file is cracked, the attacker has a lot of useful information at her fingertips. [10]

Cracked passwords are stored in **$JOHN/john.pot**, a non-human readable file. the **--show** tag is the best way to view cracked passwords [10].

19 Challenge 4: Cracking Linux Password   <>

- John the Ripper contains a different flag to decyrpt the linux hash file

- This will require a different format flag than when decrypting the sam hash file

- You need to find what encryption algorithm linux uses and the associated flag to decypher the hash

- Hint: Newer Linux distributions use a specific encryption algorithm that John the Ripper has a place in the "–format=" flag.

[15]

20 Challenge 5: Windows 7 Password Manager<>

In the Window 7 policy manager you can change the requirements needed for password creation. Try to find and edit these policies to make sure that users create and change secure passwords in your windows server.

- Find where the password policies are stored and look at the minimum requrements that windows require for passwords.

- Change and play with the policies so that you can keep your users secure and that when a user creates a password it will not accept weak passwords.

Hint: Look in Computer Configuration program and navigate to where password policies are stored to find and change the policies.

Duration: 10-15 minutes

21   Questions

1. What is salting?

2. Can you crack any possible password with a brute-force attack? If so, what would this require?

3. What encryption is used by Windows? Linux?

22 Rainbow Tables

- Another way to crack passwords is to calculate all possible hashes for passwords of a certain length and hash function and to store the passwords and their associated hashes in a table for future lookup.

- This trades speed for storage capacity. To have rainbow tables for the most common hash functions and for passwords of moderate length terabytes of storage is commonly required.

- The downside to cracking with rainbow tables is that if a salt is added to a password it reduces its effectiveness to the point where it gives no advantage.

[14]

Specialized software is available for cracking passwords with rainbow tables. Rainbow tables and the required software to use them can be downloaded from many places, but one place is Project RainbowCrack http://project-rainbowcrack.com/index.htm. [14] (Rainbow crack, or RCrack, is also a program included in Kali that performs rainbow table attacks in a similar fashion to JTR)

23 Pass-The-Hash [20]   <>

- Technique that allows attacker to authenticate access using LM and NTLM Hash without needing cleartext password

- Any Windows machine that uses communications protocols are vulnerable

- Very difficult to defend against, requires defense in depth (see below)

Mitigations include heavily using least privilege principle, firewalls, disk encryption, removal of credential caching, active directory usage, limiting administrator logins to specific domains, patching, and so on.

24 Conclusion <>

- Passwords are fragile. Passwords that are easy for people to remember are even easier for computers to guess.

- Even "secure" passwords can be vulnerable to more sophisticated attacks.

- It seems likely that passwords will ultimately be replaced by a newer more secure mechanism, that will combine better security with greater ease of use.

25 Appendix: Setting Up the VM, Solutions, and Change-log <

1. Steps for setting up the virtual machine

2. Solutions to the challenges and questions

3. Change-log

Steps for Virtual Machine setup:

1. This Tutorial requires following VM's

   - Windows7 VM with WinSCP preloaded
   - Kali-Everything VM with OpenSSH and John The Ripper preloaded

Solutions:

- Challenge 1: Crack a basic MD5 Password hash using John the Ripper

    ○ find rockyou.txt in /usr/share/wordlists

    ○ unzip the file with gzip -d rockyou.txt.gz

    ○ touch hashes.txt

    ○ echo -n "password123" — md5sum ¿¿ hashes.txt

    ○ john –wordlist=rockyou.txt hases.txt

- Activity 1: Physical Access

    ○ Login into windows 7 and and create five user accounts

    ○ go to contol pannel

    ○ go to add or remove user accounts and add five new accounts

    ○ Log into kali vm and add 5 new accounts

    ○ sudo adduser **<name>**

    ○ sudo passwd **<name>**

- Activity 2: Physical Access 2

    ○ sudo systemctl start ssh

    ○ open WinSCP

    ○ connect to 192.168.14.201

    ○ username: kali password: kali

- Challenge 2: Saving windows7 hashes

    ○ reg save hklmc:

    ○ reg save hklmc:

    ○ then transfer contents via WinSCP

- Challenge 3: Offline Password Attack2

    ○ touch out.txt

    ○ samdump2 SYSTEM SAM **–o** out.txt

    ○ john –format **=** NT

- Challenge 4: Cracking Linux Passwords

    ○ Get linux hashdump

    ○ john –format **=** crypt

    ○ if passwords do not show use –show to see cracked passwords

- Challenge 5: Windows 7 Password Manager

  ○ open contol pannel

  ○ open system and security

  ○ open administrative tools

  ○ open local security policy

  ○ expand account policies

  ○ change policies

Questions

- The difference between a hash with salt and without salt is its complexity. Because a strong password with random data appended to it is going to make the work of extracting information really tough even when your hash is exposed.

- Yes. For the really tough ones, all you need is massive amounts of time and computing resources.

- Windows uses NTLM and Linux uses newcrypt

Changelog:

Ver.	Date	Authors	Changes
— Password (In)security: Attacks & Prevention			
v1	March 08th 2017	Harika Vadapalli and Nicholas Valenti	Modified slides and added several new slides. Problem Description is modified. New real world examples in slide two and Salting technique was added. Added slides on LM and NTLM hashes, Password Cracking attacks, security guidelines,Types in attacks. Slides on John Thr Ripper were modified from previous tutorial. Added new challenges and tasks. Linux description was modified from previous one and added new slides about Rainbow tables, hashcat and pass-the-hash by us.
v1.1	May 30th 2018	Ananth Jillepalli	Updated network diagram
v1.2	October 18th 2024	Jeffrey Zhang and Garrett Pearsall	Modified slides and added several new slides. Removed and Refactored Several Challenges and Knowledge checks. Updated Kali/Win7 Cracking Activities. Now requires WinSCP and OpenSSH. Removed Ubuntu and WinXP requirements for VMs. Removed Out of Date Network Diagram.

References

[1] Frampton, S. *Linux Password & Shadow File Formats*. Linux Administration Made Easy. `http://www.tldp.org/LDP/lame/LAME/linux-admin-made-easy/shadow-file-formats.html`.

[2] Openwall. *John the Ripper usage examples.* `http://www.openwall.com/john/doc/EXAMPLES.shtml`

[3] Vindu Goel and Nicole Perlroth. *Yahoo Says 1 Billion Accounts Were hacked.* `https://www.nytimes.com/2016/12/14/technology/yahoo-hack.html?_r=1`

[4] Swati Khandelwal. *Github accounts Hacked in 'Password reuse attack'.* `http://thehackernews.com/2016/06/github-password-hack.html`

[5] Lucian Constantin. *Critical vulnerability in Group Policy puts Windows computers at risk.* `http://www.computerworld.com/article/2883152/critical-vulnerability-in-group-policy-puts-windows-computers-at-risk.html`

[6] Hwong, Jenko. *Group Policy Security Risks and Best Practices* `https://www.giac.org/paper/gsec/4138/group-policy-security-risks-practices/104227`

[7] Microsoft Corporation *The Importance of Using Strong Passwords.* Hot for Security. `https://msdn.microsoft.com/en-us/library/ms851492(v=winembedded.11).aspx.` .

[8] Hoffman, C. *How to Create a Strong Password (and Remember It).* How-To Geek `http://www.howtogeek.com/195430/how-to-create-a-strong-password-and-remember-it/.` .

[9] Johnson, D. *How secure are password managers?.* `http://www.cbsnews.com/news/in-wake-of-lastpass-hack-how-safe-are-password-managers/.` .

[10] John the Ripper *John the Ripper Documentation.* `http://www.openwall.com/.`

[11] Skull Security. *Passwords.* rockyou.txt. `https://wiki.skullsecurity.org/Passwords.`

[12] Count Upon Security *JTR CHEAT SHEET.* `https://countuponsecurity.files.wordpress.com/2015/06/jtr-cheat-sheet.pdf.`

[13] *Microsoft LAN Manager Hash.* `https://en.wikipedia.org/wiki/LM_hash.`

[14] *Project RainbowCrack* `http://project-rainbowcrack.com/index.htm`

[15] Weidman, Georgia. *Penetration testing: a hands-on introduction to hacking.* No Starch Press, 2014.

[16] Goodin, D. *How Crackers Make Minced Meat Out of your Passwords.* Ars Technica. `http://arstechnica.com/security/2013/05/how-crackers-make-minced-meat-out-of-your-passwords/1/.` .

[17] Openwall. *John the Ripper's command line syntax* `http://www.openwall.com/john/doc/OPTIONS.shtml`

[18] Cryptographic Nonce. `https://en.wikipedia.org/wiki/Cryptographic_nonce`

[19] Hashcat. `https://en.wikipedia.org/wiki/Hashcat`

[20] Pass The Hash. `https://en.wikipedia.org/wiki/Pass_the_hash`

[21] OCCUPYTHEWEB. *How to Crack Passwords, Part 3 (Using Hashcat)* `https://null-byte.wonderhowto.com/how-to/hack-like-pro-crack-passwords-part-3-using-hashcat-0156543/`

[22] LinkedIn Hack. *Another Day, Another Hack: 117 Million LinkedIn Emails And Passwords* `https://motherboard.vice.com/en_us/article/another-day-another-hack-117-million-linkedin-emails-and-password`

[23] Naked Security News. *Windows passwords: "Dead in Six Hours" – paper from Oslo password hacking conference* `https://nakedsecurity.sophos.com/2012/12/17/windows-passwords-dead-in-six-hours-paper-from-oslo-password-hacking-con`