Respected Sir/Ma’am,

It was very easy to crack with *rockyou.txt*, I would suggest that you use very strong password mechanism to create hashes for the password based on Secure Hash Algorithm.

Q1: What type of hashing algorithm was used to protect passwords?

MD5 or **MD4** (Raw Hash)

S.No	Hash File	Hash Type (Full Details)
1	e10adc3949ba59abbe56e057f20f883e	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
2	25f9e794323b453885f5181f1b624d0b	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
3	d8578edf8458ce06fbc5bb76a58c5ca4	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
4	5f4dcc3b5aa765d61d8327deb882cf99	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
5	96e79218965eb72c92a549dd5a330112	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
6	25d55ad283aa400af464c76d713c07ad	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
7	e99a18c428cb38d5f260853678922e03	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
8	fcea920f7412b5da7be0cf42b8c93759	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
9	7c6a180b36896a0a8c02787eeafb0e4c	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
10	6c569aabbf7775ef8fc570e228c16b98	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
11	3f230640b78d7e71ac5514e57935eb69	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
12	917eb5e9d6d6bca820922a0c6f7cc28b	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
13	f6a0cb102c62879d397b12b62c092c06	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
14	9b3b269ad0a208090309f091b3aba9db	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
15	16ced47d3fc931483e24933665cded6d	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))

16	1f5c5683982d7c3814d4d9e6d749b21e	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
17	8d763385e0476ae208f21bc63956f748	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
18	defebde7b6ab6f24d5824682a16c3ae4	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))
19	bdda5f03128bcbdfa78d8934529048cf	MD5, SHA1.Substr(0, 32), MD4, NTLM, md5(md5($plaintext))

Q2: What level of protection does the mechanism offer for passwords?

- MD5 is an “**iterative**” hash function.
- MD5 is generally a **considerable mechanism** for storing passwords in production.
- MD5, produces a **128-bit hash.**
- MD5 is born out of **RSA’s algorithm**.
- MD5 is a utility that can **generate a digital signature of a file**. MD5 belongs to a family of one-way hash functions called **message digest algorithms**. The MD5 system is **defined in RFC 1321**.
- The algorithm takes as input a message of **arbitrary length** and produces as output a **128-bit "fingerprint" or "message digest"** of the input. It is conjectured that it is **computationally infeasible** to produce two messages having the same message digest, or to produce any message having a given prespecified target message digest. The MD5 algorithm is **intended for digital signature applications**, where a large file must be **"compressed"** in a secure manner before being encrypted with a private (secret) key under a public-key cryptosystem such as **RSA.**

Q3: What controls could be implemented to make cracking much harder for the hacker in the event of a password database leaking again?

- One way of making the password hard to crack is by **maintaining credentials from multitude of services in a manager** like dash lane because they tend to use **varied hashing** algorithms & even hashing over hashed passwords [e.g., md5(md5($plaintext))] to store and keep the **strength high**, meeting to the rigidity of a strong case for an algorithm to process.
- **Reduce redundancy** across services such that in case of a leak out of one service doesn’t make the **other passwords vulnerable**.
- **Use alphanumeric character** with **special characters**.
- Reducing occurrence of an **adjective on noun or verb** which is an obvious prey to brute force attacks.

Q4: What can you tell about the organization’s password policy (e.g., password length, key space, etc.)?

It can be very well determined that the organization's **password policy is not up to the mark** as:

- A strong password must be at **least 8 characters** long.
- Although they do not allow spaces, the use of **special characters is probably resisted** to a set of common delimiters like ‘_’.
- The use of **numbers increases the resistance** of password by a factor of **10 times the digit appears**.
- The **lack of capital characters** splits the password strength by half.
- **Not avoiding the occurrence of English verbs** like book, popular, eating, hero, life, John Wick, interest, expert in turn making the password vulnerable to brute force attacks.
- It should not contain any of your **personal information**—specifically your real name, user name, or even your company name.
- It must be very **unique from your previously** used passwords.

Q5: What would you change in the password policy to make breaking the passwords harder?

- Keeping a **threshold on length**.
- **Caution** over use of **verbs are nouns or adjectives**.
- **Mandating** minimum **3 special characters and minimum one capital letter**.
- Applying a **hashing algorithm over another**, recursively to have a strong hashing function e.g., md5(strtoupper(md5($plaintext)))
- Your **password** should **not** be your **email, phone number or birthday**.

SUBMITTED BY:

NIKITA RAGHUWANSHI