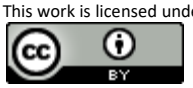
Available online at: https://ijact.in

Date of Submission	03/06/2020
Date of Acceptance	19/07/2020
Date of Publication	31/07/2020
Page numbers	3762-3767 (6 Pages)



ISSN:2320-0790

An International Journal of Advanced Computer Technology

ANALYSIS OF PASSWORD PROTECTED DOCUMENT

Dharavath Narendar[1], Sriramudu[2], VenuNalla[3], Padmavathi Guddeti[4*]

[1,2,3,4]C.R. Rao Advanced Institute of Mathematics, Statistics and Computer Science,
UoH Campus, Hyderabad, India
[1,2,3]Acharya Nagarjuna University, Guntur, India.
[*]Email: padmagvathi@gmail.com

Abstract: Nowadays, the documents are sent through electronics communications channels like email, WhatsApp, telegram etc., in which the document protection is of major concern. Passwords are used to encrypt the documents of different formats. In this research paper, we analyzed the encryption process involved in word documents (Procedure involved in document protection). We also discussed various password cracking possibilities and steps involved in the attacks including various password cracking tools for analysis of password for doc files and performed salt analysis on the same. We analyzed the randomness of the salt for the same key at different times, with different name and based on the size of the documents. We focused on John the Ripper (JtR) tool with single mode, word list, and incremental mode to reduce the file and memory complexity of brute force attack. We analyzed the randomness of the salt for the same key with same document with different time and the same documents with different name and size. We focused on John the Ripper (JtR) tool for reducing the file and memory complexity of brute force attacks. Also, we've discussed the performance analysis of password cracking based on CPU and GPUs with and without writing the dictionaries.

Keywords: password cracking; salt analysis; hash functions; cryptography; attacks

I. INTRODUCTION

String of characters known as passwords, can be used for the purpose of authentication and it is also known as access code or secret code. The string of characters may contain numeric, alphabets, symbols or special characters or combination of these. Passwords are mainly used for account protection in both dynamic and static modes. Dynamic mode stands on online authentication process which involves social engineering, bank transactions, and web logins. Static modes (standalone system) covers user logins, document protection of different formats i.e, doc, docx, excel, ppt, pdf, zip, rar, and mobile authentication.

Usually for the protection of any document we have to choose password in such a way that guessing of it is very difficult. Sometimes a very difficult one is not feasible to remember. So it is better to use key board patterns or some replacements of the alphabets with special characters to produce a secure password. Our research is mainly focused on analysis of document protection process involved in word document and its vulnerabilities. The same analysis can be used for pdf, zip and ppt formats. An outline of password cracking discussed in [1]. The authors of [2] and [3] have discussed the use of simulating password cracking for measuring password strength.

This paper is organized as follows; Section 2 contains Encryption process involved in protection of word document. Section 3 contains the flow of password attacking possibilities. Section 4 contains Password Cracking tools. Section 5 contains Password recovering of word document Using John the Ripper and Section 6 contains Performance analysis of creation of brute force dictionary .Section 7 contains conclusion and future work.

II. ENCRYPTION PROCESS INVOLVED IN PROTECTION OF WORD DOCUMENT

In providing security, cryptography plays major role. In document protection process, stream or block ciphers can be used for data encryption, hash functions can be used for authentication and key generation [4]. From Table I, shows

the cryptographic algorithms used in different versions of word documents.

Table I: MS Word Password Protection

Document	Password Protection	
	Hash Function	Encryption Algorithm
Word 97–2003	MD5	40-bit key RC4
Word 2010	SHA-1	AES and a 128-bit key
Word 2013	SHA-1	128-bit AES
Word 2016,2019	SHA-256	256-bit AES,CBC

Hashing is a mathematical method for producing an encoded string with fixed length for any given string. Hashing algorithms are not only used to store passwords but also used for checking the integrity of data.

Whenever a password is gone through a one-way mathematical algorithm, or process, that produces a completely different string, the password hash is born. Most popular hashing algorithms are MD5, SHA1, SHA2 and SHA3. No two data (i.e, password) will have the same hash. This is because, changing a single character of the data (i.e. password) will lead to production of a completely different hash value. But, if same password is used for two documents then hash value will be same. Due to this, in document protecting process passwords are used along with Salt i.e., passwords prepending with some random value salt (Salt + password) are used instead of password alone. From the salt analysis of word document and Table II, we can observe that small change in the document / file name, size, or content leads to a new or completely random value of Salt in the process of document protection. Security of password hashes and salt were mentioned in [5],[6] and [7]. Figure 1 demonstrates the Encryption process in password protection of doc.

Table II: Salt analysis

S. No	Word Document	Salt	Remarks
1.	Same file size &content , same password and file name also same	No change	Not effected with date and time
2.	Different content with same password	Changes	Minor changes can lead to change in the random value i.e., Salt which leads to completely different hash
3.	Same content with same password but filename is different		
4.	Same content with same password (Renamed the file)		
5.	No content with same password		
6.	Same content with same password (Removed one letter from the file)		
7.	Same content with same password (Removed one letter from the file in such a way that one bit difference of the content)		

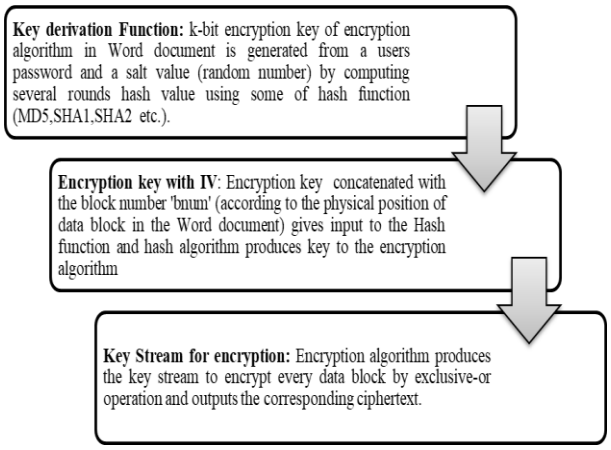


Figure 1. Encryption process involved in protection of word document

III. THE FLOW OF PASSWORD ATTACKING POSSIBILITIES

To crack password of protected document, the best attack is brute force attack but the complexity of attack is based on the password length. A brute force attack is one of the passwords cracking attack in which every possible combination in a range of characters is generated and used against the password hash. Once the libraries of hashes are obtained, they are compared to the target hash so that a matching hash is found.

If the length of the password is 4 and (i) Password uses only Latin small(a-z), then the number of lines or passwords in the brute force file or brute force dictionary file is $26.26.26.26=(26)^4$ (ii) Password uses only Latin small (a-z) and capital(A-Z), then the number of lines or passwords in the brute force file or brute force dictionary file is $52.52.52.52=(52)^4$ (iii) Password uses only Latin small (a-z) , capital(A-Z) an digits (0-9), then the number of lines or passwords in the brute force file or brute force dictionary file is $62.62.62.62=(62)^4$ (iv) Password uses only Latin small (a-z) , capital(A-Z) digits (0-9) and special character 33 (with space), so 26+26+10+33= 95 characters are

abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVZXYZ1234567890 !@#$%^&*(){}[]<>"?'.`;-+=,_|:/\~

Then the number of lines or passwords in the brute force file or brute force dictionary file is 95.95.95.95= $(95)^4$. And the length of the password is 5 then the number of lines or passwords in the brute force file is 95.95.95.95.95= $(95)^5$, so the size of the passwords file for brute force attack will increase based on the length of the password characters. Creation of Dictionaries can reduce the space and time. Table III demonstrates 2 character, 3 character, 4 character, 5 character and 6 Character passwords with all possible cases in a file which comes under brute force attack. But the file size of 5 characters is 47 GB. For 6 characters it can increase approximately 4TB memory. So, to apply brute force attack we have to use it in online password generation and checking mode. Using High Performance Computing (HPC) we can reduce time and memory by online generation mode. Generation of passwords up to 16 characters without storing can be observed from Table IV.

Table III: Storage space for Brute force attack dictionary

Characters	operating system	Memory	Time
2	intel i7@3.20GHz,64 bit	27.1kB	0.19 sec
3	intel i7@3.20GHz,64 bit	3.4 MB	1.606 sec
4	intel i7@3.20GHz,64 bit	407.3 MB	2 min 29 sec
5	Using MPI Programming 3 CPU nodes each with 32 core.	46.4 GB (each core of data with 467MB)	2 min
6	Using MPI Programming 3 CPU nodes each with 32 core	4.7 TB (each core of data with 51GB)	1 hr

Table IV: Without Writing, Wordlist Generation

S. No	Memory	Time
2 Character	No I/O writing	0.004 sec
3 Character	No I/O writing	0.116 sec
4 Character	No I/O writing	6.084 sec
5 Character (Using MPI Programming)	No I/O writing	9 min 54.980 sec
6 Character (Using MPI Programming)	No I/O writing	33 min 49 sec
7 Character (Using OpenMP Programming)	No I/O writing	23 sec
16 Character (Using OpenMP Programming)	No I/O writing	21 min 33 sec

Dictionary attack is the next attached to be studied. A dictionary attack is one of the passwords' cracking attacks where dictionaries are prepared according to the partial information, logics, patterns, social engineering data, psychology etc., and after that, each word of the dictionary is used to produce the target password hash.

We downloaded available dictionaries from social media, electronic media and other different sources and observed the patterns involved in the dictionaries based on region, language, countries etc. The next doable attack is hybrid attack, which is a mixture of both dictionary and brute force attack. Under this attack dictionaries must be prepared based on some rules like 8 character length password brute force dictionaries can be appended with some numbers are characters. Table V demonstrates steps involved in different attacks for password cracking (PC).Method of Managing passwords securely was discussed in [8].

Table V: Steps involved in attacks of PC possibilities

Attack	Description of the attack	Steps involved in attack
Brute Force Attacks	World lists (word list is plain text consisting of one password per line) of all combination are collected in a file based on the password length. It checks all possible combinations of the passwords which match with the protected document hash value.	1) Enter the hashes to be cracked 2) Select Character set and length 3) Generate hash value 4) Compare with the document hash 5) Matches with one of the passwords and returns that password
Dictionary Attacks	Dictionaries must be prepared according to the available information, local region, local language related words logics, patterns, afterwards, each word of the dictionary is attempted against a password hash	1) Enter the hashes to be cracked 2) Load the dictionary 3) Generate hash value 4) Compare with the document hash 5) Hash recovered then password cracked. Otherwise go to step to and load another dictionary.
Hybrid Attacks	It is a mixer of brute force and dictionary attack.	1) Enter the hashes to be cracked 2) Define Brute force rules 3) Load the dictionary 4) Generate hash value 5) Compare with the document hash 6) Hash recovered then password cracked. Otherwise go to step to and load another dictionary
Rainbow Table attack	It is simply a completed brute force attack that can be reused as many times as needed without having to regenerate all those probable password combinations.	1) Rainbow tables are in form of lookup tables which contain almost all possible combinations of passwords within a given character set.
The time-memory trade-off attack	It involves using pre-computed, partially stored hash tables with the relevant missing parts that are computed at attack time.	1) Pre computations of hashes were stored 2) Missing part calculates at the time of cracking password of document.

IV. PASSWORD CRACKING TOOLS

In the literature there are many password cracking tools. Using tools we can crack passwords based on the dictionary provided to the tool. The following are popular password cracking network authentications and brute-forcing tool for UNIX, Linux, Windows OS, and Mac OS. These tools can be applicable for all formats of the documents i.e., doc, docx, pdf, rar, and zip but some of them have some limitations. Their lighter versions are free which can be useful for breaking weak passwords. The main disadvantage of these tools is free version has limited functionality.

4.1. *John the Ripper [9]:* It can perform brute-force attack with all possible passwords by combining text and numbers. We can also use it with a dictionary of passwords to perform dictionary attacks. It supports fifteen different platforms including Unix, Windows, DOS, BeOS, and OpenVMS.

4.2. *PassFab for Word:* Flexibility in the password recovery using three different password recovery options. Easy to use: recover passwords in two steps. Supports Word 2017/2016/ 2013 etc. Uses advanced algorithms and GPU technology to ensure fast recovery times.

4.3. *Recuva:*It supports multiple file systems which has compatibility with Windows family OSes including 10/8.1/8/7/XP/Vista/Server 2008 and 2003.

4.4. *SmartKey Office Password Recovery:*This tool is able to recover the passwords of MS Word 2017/2016/ 2013/ 2010/ 2007. One of brute-force, brute-force mask or dictionary attacks is used by it to recover passwords.

4.5. *Aircrack-ng*: It is a tool to crach the password for WiFi which can crack WEP or WPA passwords. It tries to crack passwords by analysing wireless encrypted packets using its cracking algorithm. It uses the FMS (Fluhrer, Mantin and Shamir attack) attack in parallel with other useful attack techniques to crack password. It is applicable for Windows and Linux systems.

4.6. *Ophcrack*:It is a free tool based on rainbow-table password cracking for Windows which is also useful for Linux. It cracks LanMan i.e., LM and New Technology LM i.e., NTLM hashes.

4.7. *Hashcat*:It is the most fastest and advanced tool for password cracking. It performs dictionary attack, Combinator attack, brute force attack, brute force attack and masking i.e., hybrid attack. It support multi OS i.e. Windows, Linux, macOS.

4.8. *THC Hydra*: It is a network logon password cracking tool and it is applicable for both Linux and Windows.

4.9. *Rainbow Crack*: Rainbow Crack generated MD5 rainbow tables, LM rainbow tables, NTLM rainbow tables, and Sha1 rainbow tables. And the tables are free so one download.

4.10. *Cain and Able*: It support windows and performs dictionary attack. It is mainly for network administrators, security professionals, forensics staff, and penetration testers.

4.11. *L0phtCrack*: It is Similar to Ophcrack using windows workstations and active dictionaries.

4.12. *SAMInside*: It supports windows and useful to unlock the system which are locked.

4.13. *DaveGrohl*: It supports macOS and cracks passwords.

4.14. *Ncrack*: It support multi OS i.e.. Windows, Linux, macOS, and cracks passwords.

4.15. *Wfuzz*: It is a web applicable password cracking tool

4.16. *Brutus*: Most flexible windows supported password cracking tool.

4.17. ElcomSoft: It support multi OS i.e.. windows, macOS

In the literature, there are many password cracking tools but few of them are suitable for windows among which we studied John the Ripper as base for our research towards cracking password from password a protected word document. The same analysis can be applied for others like zip, ppt, pdf and xls. Machine learning and neural networks can be used to create dictionaries based on passwords collected from social media [10-16].

V. PASSWORD RECOVERING OF WORD DOCUMENT USING JOHN THE RIPPER

Comparing with the other password cracking tools John the Ripper is more flexible tool for windows. When we compared with the password cracking time of the different formats doc, docx & Zip, docx file takes very less number of passwords per second and docx is very strong against brute force attack. Password recovering process involved using John the Ripper can understand from Figure 2.

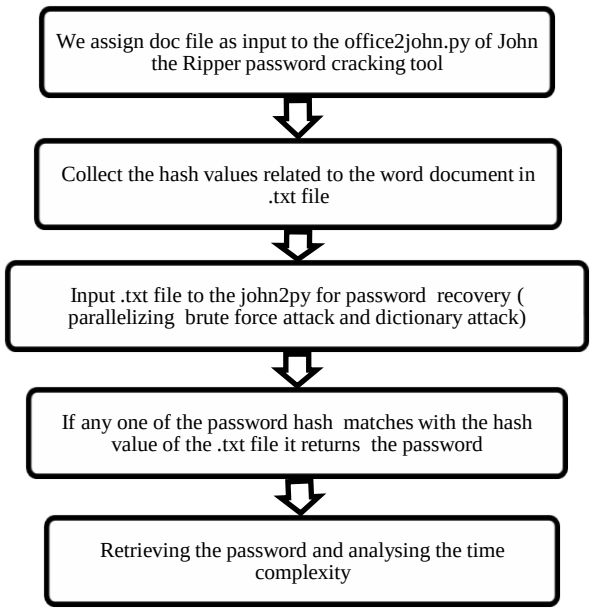


Figure 2. Password Recovering using John the Ripper

Using John the Ripper we can have hash of the document which can be stored in .txt file. For example, 2007 version of word document text.docx file is given to john the ripper

then the hash of text.docx file can be observed from Table VI.

Table VI: hash of text.docx

Hash value to be stored in .txt file	test.docx:$office$*2007*20*128*16*ea710f168a80f36e539578b72f88b314*6bdf30cbc904294a577b7e72b1832d39*3a2ba3a916a8f5d09dc7cca03ac2480d6cb55680 which has to be stored in .txt file.
test.docx	Word document name
$office$	Office document
2007	windows 2007 version
20	SHA1-20 bytes-160 bits
128	AES 128 bit key
16	16 bytes of Salt
ea 71 0f 16 8a 80 f3 6e 53 95 78 b7 2f 88 b3 14	Salt Value
6bdf30cbc904294a577b7e72b1832d39	Encrypted Verifier
3a2ba3a916a8f5d09dc7cca03ac2480d6cb55680	Hash value of Encrypted Verifier

VI. Performance Analysis of Creation of Brute Force Dictionary

Creating of the Dictionary using 94 characters (lowercase a-z, uppercase A-Z, numbers 0-9 and 32 (special characters without space) for password analysis of character length from 6 to 16. Created the dictionary of string length 5 data storage 47.8 GB in 3h1m36sec hours in Intel i7, 32GB RAM.

Profiled the code, generation of dictionaries will only take 1.96% and 98.04% time taken in I/O operation. Profiling Result, creating dictionary for 5 characters after exempted I/O operation and Parallelized the code of brute force dictionary using MPI and OpenAcc can be seen in Figure 3, 4 & 5 respectively.

```
======== CPU profiling result (bottom up):
Time(%)      Time  Name
 98.04%   231.02s  write
 98.04%   231.02s  | generate
 98.04%   231.02s  |   main
 98.04%   231.02s  |     ???
  1.96%  4.60999s  generate
  1.96%  4.60999s  | main
  1.96%  4.60999s  |   ???
  0.00%      10ms  __pthread_enable_asynccancel
  0.00%      10ms    write
  0.00%      10ms      generate
  0.00%      10ms        main
  0.00%      10ms          ???

======== Data collected at 100Hz frequency
```

Figure 3. Profiling code Result

```
[password@shavak-hrd1 cracking]$ time ./a.out
Using lenth = 5
Max combinations = 7339040224
Characters = abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVZXYZ1234567890!@#$%^&*()[]{}<>"?'.`;-+=,_|:/\~

real    9m5.584s
user    9m4.301s
sys     0m0.083s
[password@shavak-hrd1 cracking]$
```

Figure 4. Creating dictionary for 5 characters after exempted I/O operation (time: 9 min 5 sec)

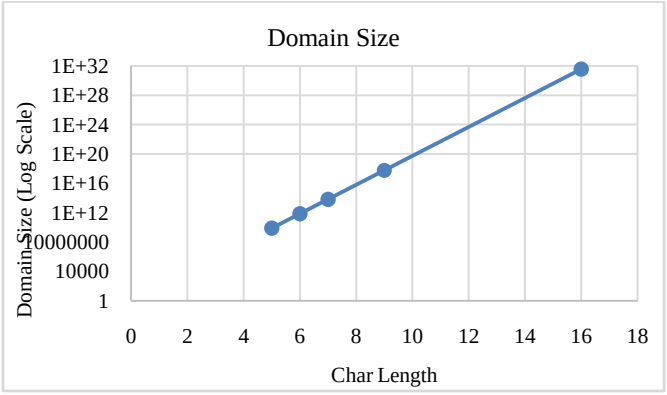


Figure 5. Parallelized the code of brute force dictionary using MPI and Open Acc

Using GPU, program time complexity decreased in MPI, OpenAcc. Memory-bound problem also solved. Comparison between MPI and OpenAcc (Speed up Vs Scalability) of brute force dictionary can be analyzed from Table VII. Using GPU, pprogram time complexity decreased in MPI, OpenAcc. Memory-bound problem also solved.

Table VII: Comparison between MPI and OpenAcc of brute force dictionary

Comparison with MPI and OpenAcc	OpenAcc Speedup wrt MPI (character length 6)	84.54 times
Word list generation	Openacc Scalability w.r.t. domain size (character length 6 vs 16)	5.38×10^{19}

VII. Conclusion and Future Work

Cryptographic algorithms involved in document protection play one of the major role in password cracking. Guessing of 40 bit key for RC4 can be done by exhaustive search or rainbow table attack. But in case of 128 bit, these procedures are very difficult. So we calculated the complexity of brute force attack with storing all passwords and without storing too along with salt analysis on word documents. And also analyzed the randomness of the salt for the same key with same content of the document, with different time, with different document name, by removing single character, by observing the change of the documents with single bit change of the content and based on the size of the documents. So salt is purely random such that identifying non randomness is a difficult task. We focused on John the Ripper (JtR) tool reducing the file and memory complexity of brute force attacks. This analysis can be useful for a new researcher to do research on password cracking possibilities and tools. Also did performance analysis of password cracking of word document based on CPU and GPUs. As a future work we are extending brute force attack and dictionary attack with less storage by using more computational power and also focussing on creation

of dictionaries using machine learning based on patterns, regions, social & electronic media.

VIII. ACKNOWLEDGMENTS

The authors would like to thank Prof. C.E.Venimadhavan, Dept. of Computer Science and Automation, Indian Institute of Science, Bangalore for his valuable suggestions.

REFERENCES

[1] Yu, F. and Huang, Y. 2015. An overview of study of password cracking. In 2015 International Conference on Computer Science and Mechanical Automation (CSMA). 25-29, IEEE.

[2] Kelley, P.G., Komanduri, S., Mazurek, M.L., Shay, R., Vidas, T., Bauer, L., Christin, N., Cranor, L.F. and Lopez, J. 2012. Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms. In 2012 IEEE symposium on security and privacy. 523-537. IEEE.

[3] Chanda, K. 2016. Password security: an analysis of password strengths and vulnerabilities. International Journal of Computer Network and Information Security.8(7), p.23.

[4] Zhang, L.J., Yu, F. and Ji, Q.B. 2017. An Efficient Recovery Method of Encrypted Word Document. In Current Trends in Computer Science and Mechanical Automation. 1,40-48. Sciendo Migration.

[5] Ah Kioon, M.C., Wang, Z.S. and Deb Das, S. 2013. Security analysis of MD5 algorithm in password storage. In Applied Mechanics and Materials. 347, 2706-2711. Trans Tech Publications Ltd.

[6] Gauravaram, P. 2012, Security Analysis of salt|| password Hashes. In 2012 International Conference on Advanced Computer Science Applications and Technologies (ACSAT).25-30, IEEE.

[7] Idris, Y.B., Ismail,S.A.,Azmi, N.F.M.,Azmi, A. and Azizan, A. 2017. Enhancement Data Integrity Checking Using Combination MD5 and SHA1 Algorithm in Hadoop Architecture. Journal of Computer Science & Computational Mathematics.7(3), 99-102.

[8] Halderman, J.A., Waters, B. and Felten, E.W. 2005. A convenient method for securely managing passwords. In Proceedings of the 14th international conference on World Wide Web. 471-479.

[9] Peslyak, A. 1996. John the ripper. URL http://www.openwall.com/john.

[10] Hitaj, B., Gasti, P., Ateniese, G. and Perez-Cruz, F. 2019. Passgan: A deep learning approach for password guessing. In International Conference on Applied Cryptography and Network Security.217-237, Springer, Cham.

[11] Melicher, W., Ur, B., Segreti, S.M., Komanduri, S., Bauer, L., Christin, N. and Cranor, L.F. 2016. Fast, lean, and accurate: Modeling password guessability using neural networks. In 25th {USENIX} Security Symposium ({USENIX} Security 16). 175-191.

[12] Pal, B., Daniel, T., Chatterjee, R. and Ristenpart, T. 2019. Beyond credential stuffing: Password similarity models using neural networks. In 2019 IEEE Symposium on Security and Privacy (SP). 417-434, IEEE.

[13] Hitaj, B., Gasti, P., Ateniese, G. and Perez-Cruz, F. 2019. Passgan: A deep learning approach for password guessing. In International Conference on Applied Cryptography and Network Security. 217-237, Springer, Cham.

[14] Alpatskiy, M.A., Borzunov, G.I., Epishkina, A.V. and Kogos, K.G. 2020. New Approach in the Rainbow Tables Method for Human-Like Passwords. In 2020 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus). 2035-2040, IEEE.

[15] Glory, F.Z., Aftab, A.U., Tremblay-Savard, O. and Mohammed, N. 2019. Strong Password Generation Based On User Inputs. In 2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON). 0416-0423, IEEE.

[16] Kaloudi, N. and Li, J. 2020. The ai-based cyber threat landscape: A survey. ACM Computing Surveys (CSUR).53(1), 1-34.