Design and Evaluation of a Password Quality Assessment Model

[1]Agholor, S., [2]Sodiya A. S., [3]Aborisade, D.O., [4]Osinuga, I. A. and [5]Adeniran, O. J.
[1]Department of Computer Science, Federal College of Education, Abeokuta
[2,3]Department of Computer Science, [4,5]Department of Mathematics
Federal University of Agriculture, Abeokuta

Article Information

Article # 02013
Received: 12th, Dec. 2020
Revision: 10th June 2021
Acceptance:17th June 2021
Published: 22nd June 2021

Key Words
Authentication, Password Meter, Password Quality Assessment Model, Password Strength

Abstract

Password, no doubt is the most widely used means for online authentication, although there are some concerns about weak passwords which can lead to weak security. To avoid this, most websites leverage on existing password quality assessment models (password meters) to show the end-user the strength of his/her chosen password with the overall aim of helping the end-user select strong password. Notwithstanding this development, most of the existing password meters still do not give realistic results due to inaccurate computations as found in the literature. To address these shortcomings, an improved password quality assessment model was developed that incorporated the attributes often ignored by most of the existing password meters. The efficiency of the improved password quality assessment model and three widely used models were analyzed using some selected passwords. The results showed that the improved password quality assessment model has a Root Mean Square Error of 0.30, followed by Microsoft (1.31), Todnem (1.88) and Yahoo (2.63). This showed that the improved password quality assessment model has the least error of accuracy in computing the strength of a given password. Thus, the error of misleading end-users into accepting a password as strong when indeed such a password is weak is highly reduced when using the improved password quality assessment model.

***Corresponding Author**: **Agholor, S.**; sagholor@fce-abeokuta.edu.ng

Introduction

In any human society, identification of individuals has been a basic requirement for security purposes and survival of the society. In very small villages, everyone knows and recognizes everyone else. Thus, it is possible to easily detect a stranger or identify a potential breach of security. In today's larger and more complex society, it is not that simple.

Similarly, as more interactions take place electronically, it becomes very difficult to identify each person, hence the need to have an electronic verification of a person's identity through an authentication scheme. Authentication, therefore, could be defined as one entity proving its identity to another (Forget, 2012). The system administrators and end-users rely largely on the passwords for digital authentication (Hu, 2018). Passwords remain popular because they have several advantages. According to Forget (2012), some of the advantages of passwords over other authentication systems are: they are easy to learn how to use them, they do not require any special hardware for implementation, they can easily be changed if compromised or forgotten and they provide adequate security to end-user's data, although there are some concerns about weak passwords which can lead to weak security.

However, the weakness according to Ma *et al*. (2007) is not within the password authentication itself but the choice of the passwords by human users. According to Yang *et al*. (2020), it is evident that many online users do not know how to choose strong password that can withstand fraudulent activities of hackers. To encourage end-users, generate strong passwords, a lot of websites employ proactive password checkers, which make use of password composition policies and or password strength meters (Xu and Han, 2019). Similarly, the findings of He *et al*. (2020) and Khern-am-nuai *et al*. (2020) showed that password meters help end-users to choose strong passwords. According to Pasquini *et al*. (2020), there is the need to design a password meter that can give a more accurate and reliable strength of a given password thereby guaranteeing the security of the password-based authentication system. This is the goal of this work.

Review of Existing Password Strength Assessment Meters

In this section, we examined some of the existing password strength (quality) assessment metrics and evaluate the metrics. This is necessary because password meters, which give feedback to the end-users are derived from these metrics. In other words, many websites relied on these password meters when using proactive password checker in assisting end-users toward creating strong passwords (Agholor and Sodiya, 2013; Agholor, 2017). Here, we briefly discuss some of the widely used password assessment formulas often used to measure the strength of a given password.

One of the earliest password assessment formulas was developed by Shannon (1951). The entropy (H) of a given password according can be calculated using the formula:

$H = Len * \log_2 C$ ……………………………….(1)

where:

H is entropy; C is the number of alphabets in the printed English; and Len is length of the password.

The formula was used to compute the strengths of some selected passwords and the results are shown in table 1.

Table 1: Results of strengths of some selected passwords using Shannon entropy

SN	Passwords	Password Strength
1	password	38
2	password1	47
3	Password1	54
4	password123	57
5	password123!	73
6	aaaaaaaaaaaaaa	66
7	Password123!	79
8	2GWapWis	48
9	Wp8E&Ncc	53
10	1fi[d;jky.	61
11	;jhui-86jdgrq]	86

From table 1, the limitations of using the formula in computing the strength of a given password are:

The formula is somewhat faulty. For example, it reported 'aaaaaaaaaaaaaa' as a stronger password than '2GWapWis'.

Two or more passwords composed from the same character set/pool and same length have the same score even if one is composed with a variation of characters while the other(s) are composed with the same character.

Randomness as a measure of password strength is not accommodated by the formula. It can mislead end-users into choosing weak passwords as strong passwords.It does not ensure even distribution of the four-character set when creating a password.

The USA National Institute of Standards and Technology (NIST) published an electronic Authentication Guidelines in which they used the notion of Shannon entropy for estimating password strength (NIST Model,2006). The entropy model is given by:

$$H = \begin{cases} +4 & \text{for the first character} \\ +2 & \text{for each of the 2nd to 8th characters} \\ +1.5 & \text{for each of the 9th to 20th characters} \\ +1 & \text{for each of 21st character and above} \\ +6 & \text{use of uppercase and non-alphabetic characters} \\ +6 & \text{not in dictionaries} \end{cases} \quad \text{……………… (2)}$$

The formula was used to compute the strengths of passwords in table 1 and the results are shown in table 2.

Table 2: Results of strengths of some selected passwords using NIST entropy model

SN	Passwords	Password Strength
1	password	18
2	password1	20
3	Password1	26

4	password123	22.5
5	password123!	24
6	aaaaaaaaaaaaa	27
7	Password123!	30
8	2GWapWis	24
9	Wp8E&Ncc	24
10	1fi[d;jky.	21
11	;jhui-86jdgrq]	27

From table 2, the drawbacks of NIST entropy formula are:

The formula is somewhat faulty. For example, it reported 'aaaaaaaaaaaaa' as a stronger password than '2GWapWis'.

As reported by Weir *et al.* (2010), it is not effective metric for gauging password strength.

Randomness as a measure of password strength is not accommodated by the model. It does not ensure even distribution of the four character set when creating a password. It can mislead end-users into choosing weak passwords as strong passwords.

The use of 6 bits bonus for composition that requires both upper case and non-alphabetic characters forces the users to make use of these characters, but in many cases these characters will occur only at the beginning or the end of the password thereby reducing the total search space, hence the benefit is modest and nearly independent of the length of the password.

According to Todnem (2015), models that usually use days-to-crack approach were found to be severely lacking and unreliable for real world applications. This model for computing password strength does not utilize the typical "days-to-crack" approach for strength determination. Relatedly, Morgan (2016) confirmed the high rating of Todnem model in the computation of the strength of a given password, hence he recommended it for users of password-based authentication scheme. The formula is thus:

H=

Additions	Type	Rate
Number of characters	flat	+(n*4)
Uppercase letters	cond/incr	+((len-n)*2)
Lowercase letters	cond/incr	+((len-n)*2)
Numbers	cond	+(n*4)
Symbols	flat	+(n*6)
Middle numbers/symbols	flat	+(n*2)
Requirements	flat	+(n*2)
Deductions		
Letters only	flat	-n
Numbers only	flat	-n
Repeat chars (case insensitive)	comp	comp
Consecutive uppercase letters	flat	-(n*2)
Consecutive lowercase letters	flat	-(n*2)
Consecutive numbers	flat	-(n*2)
Sequential letters (3+)	flat	-(n*3)
Sequential numbers (3+)	flat	-(n*3)
Sequential symbols (3+)	flat	-(n*3)……..(3)

Using the Todnem password meter interface, we checked the password strengths of passwords found in table 1. The results of the feedbacks obtained are presented in table 3.

Table 3: Password strengths using Todnem password meter

SN	Passwords	Password Strength
1	password	Very weak
2	password1	Weak
3	Password1	Good
4	password123	Good

5	password123!	Strong
6	aaaaaaaaaaaaa	Very weak
7	Password123!	Very strong
8	2GWapWis	Good
9	Wp8E&Ncc	Strong
10	1fi[d;jky.	Strong
11	;jhui-86jdgrq]	Very strong

From table 3, the limitations of using this password meter are:
(i) The formula is somewhat faulty. For example, it reports 'Password1', which is found in the dictionary as a good password.
(ii) Randomness as a measure of password strength is not accommodated.
(iii) It does not ensure even distribution of the four-character set when creating a password.

Password Meters by some Service Vendors
The password meters of some service vendors such as Google (2013), Yahoo (2013), and Microsoft (2014) were studied and reported by Agholor (2017). The result of the study indicating the strength of passwords used in table 1 is shown in table 4.

Table 4: Strengths of some selected passwords as obtained from the password meters

SN	Password	Password strengths of password meters		
		Google (2013)	Yahoo (2013)	Microsoft (2014)
1	password	Weak	Invalid	Medium
2	password1	Strong	Strong	Medium
3	Password1	Strong	Very strong	Medium
4	password123	Strong	Strong	Medium
5	password123!	Strong	Very strong	Medium
6	aaaaaaaaaaaaa	Fair	Invalid	Strong
7	Password123!	Strong	Very strong	Medium
8	2GWapWis	Strong	Very strong	Medium
9	Wp8E&Ncc	Strong	Very strong	Medium
10	1fi[d;jky.	Strong	Strong	Medium
11	;jhui-86jdgrq]	Strong	Strong	Strong

Source: Agholor (2017)

From table 4, it showed that the formulas used by these password meters are somewhat faulty. For example, Yahoo (2013) reported password1 as a strong password, Microsoft (2014) reported aaaaaaaaaaaaa as a strong password, and Todnem (2015) reported Password1 as a good password. However, the study of Agholor (2017) found these passwords and others that fall on serial numbers 1 to 7 easy to crack.

Measuring the Efficiency of a Password Meter using Root Mean Square Error
The Root Mean Square Error (RMSE) is used to measure the error that a password meter (password quality assessment model) reports a password as strong when indeed it is weak. This measurement is used to evaluate the efficiency of any given password meter. The RMSE is given by the formula:

$$\text{RSME} = \sqrt{\frac{\sum_{i=1}^{n} e_i^2}{n}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4)$$

where:
n is the number of passwords evaluated;
e is the error between the standard (acceptable) feedback point and model feedback point.

Methodology

Design of the Password Quality Assessment Model
As found in the literature, most of the existing password quality assessment models make use of formulas that utilize two attributes, namely, the length and the character set that make up the password in evaluating the password strength. A third attribute, the dictionary search was added by very few models. However, these attributes are not enough to give an accurate computation of a password strength hence the need for other necessary attributes.
In this work, the following attributes were added to the three existing attributes (length, character set and dictionary search) to get a robust password quality assessment model that could effectively determine the strength of a given password. The newly added attributes are:

i) The number of unique characters that make up the password;
ii) The randomness of the password
iii) The percentage of Numeric characters (N) that make up the password;
iv) The percentage of Lowercase characters (L) that make up the password;
v) The percentage of Uppercase characters (U) that make up the password;
vi) The percentage of Special characters (S) that make up the password;
vii) Whether the password can be found in the list of default passwords such as *password, default, admin, guest, 1111* etc.
viii) Whether the password is the same as;
a) user id or user's profile information;
b) a dictionary word appended with number(s)/special symbol(s) before or after such as *password1, deer2000, john1234, deer#,* etc;
c) a word with simple obfuscation, that is, letter substitution (LS) such as *p@ssw0rd, g0ldf1sh, pa$$w0rd* etc;
d) a dictionary double word such as *crabcrab, stopstop, passpass* etc; and
e) a common sequence from the keyboard row such as *qwerty, 12345, asdfgh etc.* This is often referred to as Keyboard Layout (KL)

The Model Development Process
This begins by developing the formula for computing the sub-entropies of the various identified attributes and summing the sub-entropies to give the improved entropy model as follows:

(a) **The Length and Character set that make up the password**; Using Shannon (1951) formula:

$$H1=Log_2C^{Len} = Len*Log_2C = Len * \frac{Log(C)}{Log(2)} \quad \ldots\ldots\ldots (5)$$

where:
H1 is the computed sub-entropy for the contribution of the attributes (Length and Character set) to the overall entropy;
Len is the Length of the password; and
C is the character set that make up the password.

(b) **The number of Unique characters that make up the password**: A character is said to be unique if it appears only once in the password. For example, assuming the end-user's password is "*character*" then the unique characters are h, t and e which implies that it has 3 unique characters as c, a, and r appear more than once. Thus:

$$H2 = \frac{Unique}{Len} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (6)$$

where:
H2 is the computed sub-entropy for the contribution of the attribute (Unique characters) to the overall entropy;
Unique is the number of unique characters that make up the password; and
Len is the Length of the password.

(c) **The randomness of the password**
Many password policies require users to compose their passwords using different character sets and most end-users just put characters from the same character set together rather than mixing them up to create random passwords. For example, it is more likely to have a password *python685* rather than *py6th8on5*. The latter is more random than the former and more difficult to crack. The substrings in *python685* is 2, that is, *python* and *685*, while the substrings in *py6th8on5* is 6, that is, *py*, *6*, *th*, *8*, *on* and *5*. The more the number of the substrings in a password, the more randomness the password is. Therefore, the number of substrings are used to measure the randomness of a password as:

$$H3 = \frac{NSUB}{Len} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (7)$$

where:
H3 is the computed sub-entropy for the contribution of the attribute (randomness) to the overall entropy;
NSUB is the number of substrings from the same character set that make up the password; and
Len is the Length of the password.

(d) **The percentage of Numeric characters that make up the password**; For this attribute to have a maximum point, it must allow for equal distribution of other character sets. For example, assuming that the password is a 20-character password, then the numeric should be five characters, lowercase should be five characters, uppercase should be five characters and special symbols should be five characters. From this analysis, it follows that the percentage of each contributing attribute is 25%. Since it is not easy for an end-user to get this ideal situation of 25% in real life password creation, we take a situation that allows for an evenly distribution of all the character sets.
To achieve this, we take a percentage range. Statistically, taking alpha level of 5% implies that we are 95% confident that the percentage range will give an evenly distribution of the four-character sets. Thus, we take a starting point of 5% lower than the ideal percentage of 25% as the lower bound and take the terminal point of 5% above the ideal percentage of 25% as upper bound. Hence, we have an interval of 10%, that is, 5% below the ideal percentage of 25% and 5% above the ideal percentage of 25%. To obtain

the lower bound, we therefore subtract 5% from the ideal percentage of 25% to get 20%. From the 20%, we count 10% interval to get 29% as the upper bound, both lower bound and upper bound inclusive. Hence, the percentage range that can give an evenly distribution of the four character sets at 95% confidence interval is 20% - 29%.

Using a 5-point Likert scale, the percentage range and the associated points are as follows:
20% - 29% for 5 points (P5)
30% - 39% for 4 points (P4)
40% - 49% for 3 points (P3)
50% - 59% for 2 points (P2)
60% - 100% and 0% - 19% 1point (P1)

From the above percentage range, higher points are earned for getting as close as possible to within 20% - 29%. This is so because a password is harder to crack when the percentage of numeric, lowercase, uppercase and special characters are all as close as possible. This means the password consists of roughly as many numbers, lowercase, uppercase and special characters. To get the sub-entropy of this attribute, we must first compute the percentage range. Let Num% be the percentage range and H3 be the sub-entropy for this attribute, then:

$$\text{Num\%} = \frac{\text{NN}*100}{\text{Len}} \quad \ldots\ldots (8)$$

and

$$\text{H4}=\text{NN}*P_i \quad \ldots\ldots (9)$$

where:

H4 is the computed sub-entropy for the contribution of the attribute (percentage of Numeric characters that make up the password) to the overall entropy; Num% is the percentage range; NN is the number of numeric characters in the password; Len is the length of the password; P_i is the points of the percentage range after computation using equation (8); i = 1.5

(e) The percentage of lowercase characters that make up the password. In the same vein and using the same method as in (d), we have:

$$\text{Low\%} = \frac{\text{NL}*100}{\text{Len}} \quad \ldots\ldots (10)$$

and

$$\text{H5}=\text{NL}*P_i \quad \ldots\ldots (11)$$

where:

H5 is the computed sub-entropy for the contribution of the attribute (percentage of Lowercase characters that make up the password) to the overall entropy; Low% is the percentage range of lowercase character; NL is the number of lowercase characters in the password; Len is the length of the password; P_i is the points of the percentage range after computation using equation (10); i = 1..5

(f) The percentage of uppercase characters that make up the password. Again, applying the same method used in (d), we have:

$$\text{Upp\%} = \frac{\text{NU}*100}{\text{Len}} \quad \ldots\ldots (12)$$

and

$$\text{H6}=\text{NU}*P_i \quad \ldots\ldots (13)$$

where:

H6 is the computed sub-entropy for the contribution of the attribute (percentage of Uppercase characters that make up the password) to the overall entropy; Upp% is the percentage range of uppercase character; NU is the number of uppercase characters in the password; Len is the length of the password; P_i is the points of the percentage range after computation using equation (12); i = 1.5

(g) The special character that make up the password. Using the same method as in (d), we have:

$$\text{Sp\%} = \frac{\text{NS}*100}{\text{Len}} \quad \ldots\ldots (14)$$

and

$$\text{H7}=\text{NS}*P_i \quad \ldots\ldots (15)$$

where:

H7 is the computed sub-entropy for the contribution of the attribute (percentage of Special characters that make up the password) to the overall entropy; Sp% is the percentage range of special character; NS is the number of special characters in the password; Len is the length of the password; P_i is the points of the percentage range after computation using equation (14);i = 1.5

(h) Whether the password is in the dictionary or falls on attributes (vii) and (viii)
In this case, the entropy is classified as zero because it can easily be cracked. In other words, it has very low entropy which makes cracking attacks very easy.

The Improved Password Quality Assessment Model : Summing the computed sub-entropies for equations (5), (6), (7), (9), (11), (13), (15) and combining it with that of (h), an improved password quality assessment model to calculate the strength of a given password is hereby proposed as follows:

$$\text{H}= \begin{cases} \text{Len} * \frac{\text{Log(C)}}{\text{Log(2)}} + \frac{\text{Unique}}{\text{Len}} + \frac{\text{NSUB}}{\text{Len}} + \text{NN}*P_i + \text{NL}*P_i \\ + \text{NU}*P_i + \text{NS}*P_i \\ 0 \text{ elsewhere} \end{cases} \quad \ldots\ldots (16)$$

where the symbols retain their usual meanings as defined in preceding equations; and H is the entropy of the given password. Equation (16) is the improved entropy model designed to compute the strength of a given password.

Analysis of Strength Feedbacks

The various existing feedbacks displaying the strengths of various passwords were studied and analyzed. A 5-point Likert scale used in assigning weights to these various feedbacks was developed. This 5-point Likert scale was validated by a purposefully selected team of experts from the Information Security Group of the Nigeria Computer Society. The 5-point Likert scale is shown in table 5.

Table 5: The 5-Point Likert Scale

Feedback	Point
Very Weak or Weak or Invalid	1
Fair or Medium or Average	2
Good	3
Strong	4
Very Strong	5

Implementation and Evaluation

Implementation

The improved password quality assessment model was implemented using Microsoft Visual Studio 2012 Edition. The snapshot of the interface is shown in figure 1.

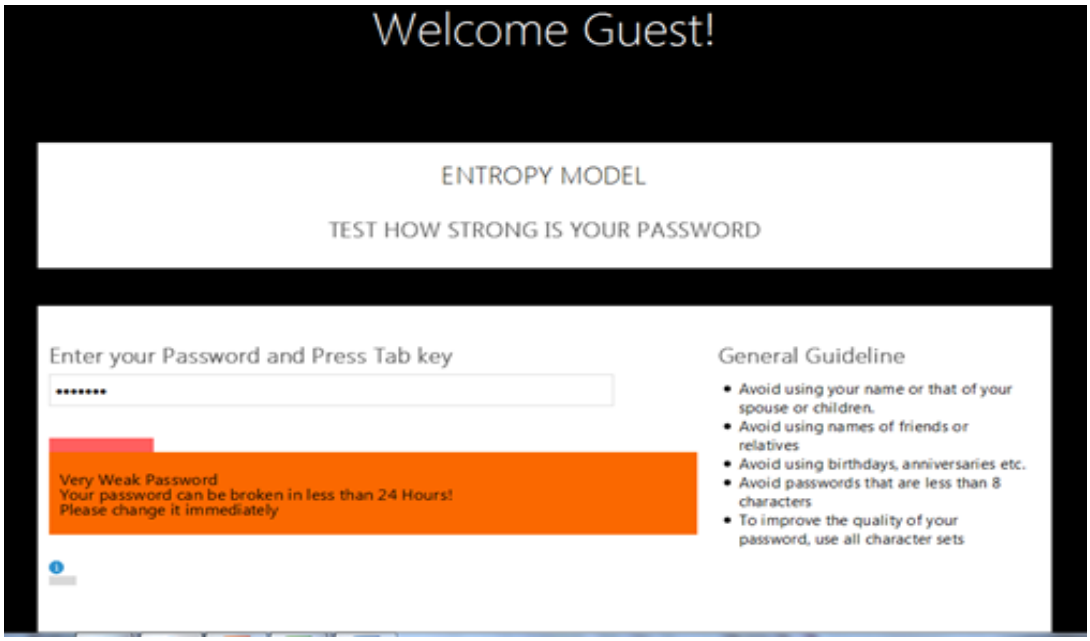

Figure 1: Snapshot of the improved model. Source: From the model software developed

Evaluation

Some selected passwords were used to test the computational results and the efficiency of the improved password quality assessment model and three (3) widely used password quality assessment models. The results were analyzed using equation (4), which is the Root Mean Square Error (RMSE) formula. The summary of the results are presented in tables 6, 7 and 8.

Table 6: Comparison of strength feedbacks

Passwords	Yahoo (2013)	Microsoft (2014)	Todnem (2015)	Improved Model
password	Invalid	Medium	Very Weak	Very Weak
password1	Strong	Medium	Weak	Very Weak
Password1	Very Strong	Medium	Good	Very Weak
password123	Strong	Medium	Good	Very Weak
password123!	Very Strong	Medium	Strong	Very Weak

aaaaaaaaaaaaa	Invalid	Strong	Very Weak	Very Weak
Password123!	Very Strong	Medium	Very Strong	Very Weak
2GWapWis	Very Strong	Medium	Good	Fair
Wp8E&Ncc	Very Strong	Medium	Strong	Good
1f[d;jky.	Strong	Medium	Strong	Good
;jhui-86jdgrq]	Strong	Strong	Very Strong	Good

Table 7: Summary of Results (Yahoo (2013) and Todnem (2015))

Passwords	**Standard Point**	**Yahoo Point**	**Yahoo e^2**	**Todnem Point**	**Todnem e^2**
password	1	1	0	1	0
password1	1	4	9	1	0
Password1	1	5	16	3	4
password123	1	4	9	3	4
password123!	1	5	16	4	9
aaaaaaaaaaaaa	1	1	0	1	0
Password123!	1	5	16	5	16
2GWapWis	3	5	4	3	0
Wp8E&Ncc	3	5	4	4	1
1f[d;jky.	3	4	1	4	1
;jhui-86jdgrq]	3	4	1	5	4
TOTAL		-	76	-	39
RMSE		-	2.63	-	1.88

Table 8: Summary of Results (Microsoft (2014) and Improved Model)

Passwords	**Standard Point**	**Microsoft Point**	**Microsoft e^2**	**Improved Model Point**	**Improved Model e^2**
password	1	2	1	1	0
password1	1	2	1	1	0
Password1	1	2	1	1	0
password123	1	2	1	1	0
password123!	1	2	1	1	0
aaaaaaaaaaaaa	1	4	9	1	0
Password123!	1	2	1	1	0
2GWapWis	3	2	1	2	1
Wp8E&Ncc	3	2	1	3	0
1f[d;jky.	3	2	1	3	0
;jhui-86jdgrq]	3	4	1	3	0
TOTAL		-	19	-	1
RMSE		-	1.31	-	0.30

From tables 7 and 8, the results showed that the improved model has a Root Mean Square Error of 0.30, followed by Microsoft (1.31), Todnem (1.88) and Yahoo (2.63). This showed that the improved model has the least error of accuracy in computing the strength of a given password. Thus, the error of misleading end-users into accepting a password as strong when indeed such a password is weak is highly reduced when using the improved model.

Recommendation

We recommend the use of the improved model in computing the strength of a given password since it gives a more accurate result.

Conclusion

The improved model used several attributes that were ignored by most of the existing password meters in computing the strength of a given password. The experimental results showed that the improved

entropy model corrected the identified flaws found in most of the existing password meters.

Conflict of Interest

The authors declare no funding sources and no conflict of interest.

References

Agholor, S. (2017). An Improved Approach for Managing Multiple Passwords. An unpublished Ph.D. thesis submitted to the Department of Computer Science, Federal University of Agriculture, Abeokuta, Nigeria. 291pp.

Agholor, S. and Sodiya, A. S. (2013). An Assessment of Feedback Mechanism of some selected websites towards improved end users' password. Proceedings of the 11th International Conference on e-Governmant and National Security. NCS, Iloko, Nigeria, pp. 44 – 49.

Forget, A. (2012). A World with Many Authentication Schemes. An Unpublished Ph.D. thesis submitted to the Faculty of Graduate and Postdoctoral Affairs, School of Computer Science, Carleton University, Ottawa, Ontario, Canada. 244pp.

Google . (2013). Google Password Meter (Online). http://google.com. Accessed on 12/05/2013.

He, Y., Alem, E. E. and Wang, W. (2020). Hybritus: a password strength checker by Ensemble Learning from query feedbacks of websites. Springer Frontiers Computer Science, 14, 143802

Hu, G. (2018). On Password Strength: A Survey and Analysis. Springer International Publishing AG 2018, pp. 165 – 186.

Khern-am-nuai, W., Hashim, M. J., Pinsonneault, A. Yang, W. and Li, N. (2020). Enhancing Operational Security by Redesigning Password Strength Meters: Evidence from Randomized Experiments (Online). https://ssrn.com/abstract=2800499. Accessed on 21/10/2020.

Ma, W., Campbell, J., Tran, D. and Kleeman, D. (2007). A Conceptual Framework for Assessing Password Quality. *International Journal of Computer Science and Network Security* 7 (1):179-185.

Microsoft. (2014). Microsoft Password Meter (Online). http://microsoft.com/athome/ security/ privacy/passwordchecker.mspx. Accessed on 12/05/ 2014.

Morgan, S. (2016). Worst Passwords of 2015, Best Passwords of 2016 (Online). http://spleshdata.com. Accessed on 14/03/2016.

NIST. (2006). Special Publication 800-63. Electronic Authentication Guideline, Technical Report, National Institute of Standards and Technology (NIST), Computer Security Division, Gaithersburg, USA. 45pp.

Pasquini, D., Ateniese, G. and Bernaschi, M. (2020). Interpretable Probabilistic Password Strength Meters via Deep Learning. Proceeding of European Symposium on Research in Computer Security, Italy, pp. 1-14.

Shannon, C. E. (1951). Prediction and Entropy of Printed English. *Bell Systems Technical Journal* 30:50–64.

Todnem, J. (2015). The Password Meter (Online). http://passwordmeter.com. Accessed on 12/05/ 2015

Weir, M., Aggarwal, S., Collins, M. and Stern, H. (2010). Testing Metrics for Password Creation Policies by attacking large sets of Revealed Passwords. In: Al-Shaer, E. (ed) Proceedings of the 17th Conference on Computer and Communications Security. ACM, Chicago, IL, USA, pp. 162–175.

Xu, M. and Han, W. (2019). An Explainable Password Strength Meter Addon via Textual Pattern Recognition. Security and Communication Network 2019: 1-10.

Yahoo. (2013). Yahoo Password Meter (Online). http://yahoo.com.

Yang, S., Yeo, K. C., Azam, S., Karim, A., Ahammad, R. and Mahmud, R. (2020). Empirical Study of Password Strength Meter Design. Proceedings of 5th IEEE International Conference on Communications and Electronics Systems, Combatore, India, pp. 436-442.