Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 44 No. 4 (2023)

A Novel Security Model for Password Encryption Using Aadhaar and Amicable Number

Mohammed Shakeel[1], Akash Sanghi[2*], YDS Arya[3], Gaurav Agarwal[4]

Ph.D. Scholar, Department of CSE, Invertis University, Bareilly (U.P.), India[1]

Assistant Professor, Department of CA Invertis University, Bareilly (U.P.)[1]

Associate Professor, Department of CSE, Invertis University, Bareilly (U.P.), India[2]

Professor, Department of CSE, Invertis University, Bareilly (U.P.), India[3]

Associate Professor, Department of CSE, Invertis University, Bareilly (U.P.), India[4]

Abstract

In this day and age, data hacking has become a major source of fear. In order to avoid any further theft of data, we must stop this criminal hacking. Backups, encryption, access control, and other techniques are just a few that may be used to ensure data integrity. We did not combine the existing approaches when we announced our encryption process. The secret to encrypting user passwords is to utilize a mathematical calculation equation that is well-defined and a number conversion mechanism. Users must provide their password and Aadhaar number in order to register. Encrypted password has been recorded in the database. Every industry uses encryption to secure data, and there are many potential applications for it in the future.

Keywords-*Armstrong number, Aadhaar Number, Amicable Number, ASCII values, Encryption, Decryption, XOR operation, cryptography.*

1.Introduction

We have grown accustomed to using internet platforms in our daily lives. Online platforms are reproducing at an exponential rate. Everyone uses numerous online applications to communicate a lot of significant information. Online platforms provide access to a variety of information, from social to corporate to governmental. However, owing to adequate security, information is revealed to a third party. It has an effect on day-to-day living, damages an organization's reputation, and so on. Various organizations have various regulations or techniques to protect the users' sensitive information as well as to safeguard them. However, the consumers find it annoying since passwords and other critical data are absent.

Every time, the online application developer or owner does not consider how the user passwords will be stored in the database. As a result, many online applications employ the traditional approach to store login information in databases as user input. People will have to manage up to 300 billion passwords by the year 2020, and according to some studies, 100 passwords are stolen every second, or more than 8 million passwords each day [1].

So, we think about the problem and describe an algorithm that will encrypt the password and stores itinto the database in an encrypted form by using aadhaar as an extra factor for authentication as well as for encrypting the password. Normally most of the online platform demands only username and password for authentication, but by inputting aadhaar number, the approach become multifactor authentication.

1.1 Cryptography

The science of hidden writing is referred to as cryptography. The words "kryptos" and "logos," which both mean "hidden word" in Greek, are the origin of the term "cryptography". The ability of a user to utilize various keys for cryptographic operations is one of the characteristics used to classify cryptographic algorithms. There are two different kinds of cryptographic techniques: symmetric and asymmetric key encryption. Symmetric key encryption, often known as private or secret-key encryption, encrypts and decrypts data using the same key. By using a shared secret key to encrypt plaintext, the following expression is employed to produce cipher-text.

1.1.1 Plain-text

The actual, understandable data or message fed into the algorithm [2]. Anything that people can comprehend and/or relate to is considered plaintext.

1.1.2 Cipher-text: Original message that becomes unreadable and disorganized. Cipher-text is the result of the encryption algorithm's processing, and it serves as the decryption algorithm's input [3].

1.1.3 Secret Key: It is one of the inputs that an encryption and decryption algorithm takes into consideration for processing to produce cipher-text and plaintext as a deliverable. On the basis of the identical algorithm and plaintext, two unique secret keys generate two unique cipher-texts [3].

1.1.4 Cryptanalysis: It is a method of extracting a secret key or plaintext from an algorithm and cipher-text without being aware of the secret value.

1.1.5Amicable Number

A pair of numbers is said to be amicable if the sum of all of the first number's proper divisors (excluding it) equals the second number, and vice versa. For example, the numbers 220 and 284 are friendly. The sum of 220's proper divisors is 284, which is (1, 2, 4, 5, 10, 11, 20, 22, 44, 55, and 110). Similarly, the total of 284's appropriate divisors is 220, which is (1, 2, 4, 71, and 142). As a result, 220 and 284 are friendly numbers [4].

Two numbers are called amicable if each equals the sum of the aliquot divisors of the other. The smallest pair of amicable number is 220 and 284. The sum of the aliquot divisors of 220 is 1+2+4+5+10+11+20+22+44+55+110=284 similarly; the sum of the aliquot divisors of 284 is 1+2+4+71+142=220. Therefore 220 and 284 are amicable numbers.

1.1.6Aadhaar Number

The aadhaar number is a unique 12-digit identification code issued to residents of India. Introduced in 2009, it serves as a digital identity for accessing various government and private services. Linked to biometric and demographic data, Aadhaar enhances efficiency in welfare distribution, financial transactions, and document verification. Each person will receive a single, distinctive Aadhaar ID number. Any identification-based application can leverage the universal identity infrastructure that Aadhaar will offer.

2.Literature Survey

Mandal *et. al.* published Fibonacci based position substitution (FBPS) in 2009, which generates encrypted text by substituting the locations of various bits inside a plain text character using Fibonacci numbers [5].

Udepal e*t.al.*, proposed a technique for ASCII value-based encryption where the cipher text is produced by combining the ASCII values of the characters in plain text with the randomization key using mathematical operations [6].

Snehal *et. al.* [7] proposed a Hybrid Data Encryption Scheme Using Color Code and Armstrong Numbers. This method uses color codes and Armstrong numbers to construct a secret key that is used as a password to decode the file.

Laiphrakpam *et al.* [8] proposed the concept of text encryption using elliptic curve cryptography. Corresponding ASCII values from plain text are paired together and those values are used as input for elliptic curve cryptography.

Nishtha *et.al*. proposed AES-based text encryption with dynamic key selection in 2016; the key length was extended to 192 bits, and the demonstration required 12 cycles to improve the performance of the old AES method [9].

Nath *et. al.* [10] suggested a technique in which they have updated the playfair approach into a new arena, where they have encrypted and decrypted text files. Depending on the initial text key the user provides, the key matrix's pattern will change. After determining the randomization number, encryption key, and shift parameter from the user's provided text key, they carried out various operations, including random left-shift, random right-shift, random up-shift, random down-shift, etc.

Yeme *et. al.* [11], suggested a two-stage verification method for user authentication in a Smartphone is suggested. First, there is the standard graphical text password. Next, a session-based 3D graphic with moving elements that serves as a password. In order to prevent dictionary attacks and guessing attacks, which are limitations of traditional text-based verification, they have included 3D picture environments and movement as an additional stage to the method of making passwords safer.

An approach known as the Modified RSA Encryption algorithm (MREA) was put out by Dhakar *et. al.* [12]. By using four prime numbers instead of two, this strategy is made more secure. As the modulus gets longer, it becomes more difficult to break it down into its component pieces. This lengthens the private key and, as a result, makes it harder to identify. The multiplicative inverse parameter has been included as a second parameter in the author's method. This extra parameter, which is a component of the private key, makes it harder for the attacker to get the private key using a brute force assault.

Aiswarya *et. al.* [13] presents a modification to the MREA (Modified RSA Encryption Technique) technique by translating the ciphertext that was obtained after encryption into binary code. Using a decimal format with digits ranging from 0 to 9, the ciphertext used in the MREA approach is encoded. Maximum 4-bit patterns are needed to represent a number when converting these digits to binary codes. To transform the ciphertext digits into a binary pattern of 1s and 0s, the author assigned binary codes to each digit and utilized these predetermined values. The recipient also receives the ciphertext in addition to the previously indicated pre-assigned codes. Inability to derive the plaintext from this binary pattern will frustrate the attacker or intrusive party.

Hamami *et. al.* [14] proposed a method in which the author recommended to utilize three prime integers rather than two to construct a public key and its matching private key. This will result in the variable N (N=p*q*r) having a big value. As N=p*q, factorization of N is more difficult than N in the original RSA technique, making it difficult for an intrusive party to locate the three prime integers. The examination of the variable N has become more difficult thanks to the recommended technique. Increasing the pace at which keys are created. It quickens the encryption and decryption procedures.

Shilpi*et. al.* [15] proposed a novel method by combining the two most complex algorithms, RSA and Diffie-Hellman, to increase security. The method in the suggested technique computes the values of "e" and "d" as in RSA, and then automatically generates three prime constants (let's say R, S, and G).These values are used to calculate the public numbers α and β using the following equation;

$\alpha = R^e \bmod P$

$\beta = R^d \bmod P$

Here, Diffie Hellman uses the following equation to calculate the session key:

$Key_A = \beta^A \bmod P$ (1)

$Key_B = \alpha^B \bmod P$ (2)

Such that $Key_A = Key_B = K$. (3)

Finally, the sender and receiver must exchange this key, represented as K, in order to carry out encryption and decryption. The encrypted text is created at the sender site by XORing the plaintext with the session key (K). The encrypted text was XORed with the session key (K) at the receiver site to produce the plaintext.

3. Research Objective

The goal of this scheme is to find a technique to create intelligent encryption systems that will work more reliably and with fewer dangers of data stealing.

4. Proposed Algorithm

Step 1: During registration, users have to provide the following credentials;

(a) Username
(b) Aadhaar Number
(c) Password
(d) Confirm password

After providing the unique username and appropriate password and aadhaar number, the user gets registered.

Step 2: During registration process, after providing the credentials, the algorithm takes the password as input and create the hash code (digest) using SHA256.

Step 3: In this step the algorithm convert each character of the digest into 8-bit binary representation.

Step 4: Now read the Aadhaar Number and perform the addition by taking a block of 4 digits and then take the square root of the sum.

Step 5: Take first two digits from the fraction part as key_base value.

Step 6: Select the Amicable number pair from list by considering the key_base as index value.

Step 7: If the key_base value is Even, take the first number from the selected Amicable number pair; otherwise second number.

Step 8: Now calculate the binary equivalent of the selected amicable number and store it in 24 bit representation.

Step 9: Divide the 24 bit pattern in 3 blocks each of size 8-bits.

Step 10: Perform Bitwise XOR operation between the 8-bit binary representation of password character and the blocks obtained in step 9.

Step 11: Convert the resultant binary pattern into its decimal equivalent and write the associated ASCII character as ciphertext character.

5. Experimental Setup

5.1Data Input and Password Conversion

In the proposed model firstly the user have to do the registration, in which user have to provide the credentials (username, password and Aadhaar Number).The following table (Table1) shows the username, password and Aadhaar Number of different users.

Table-1: User Data Table

Username	Aadhaar	Password
User-1	333238097839	12Y6W9P^te>7
User-2	528878359757	v03A>2t>*!Hr
User-3	947768918025	$oP#SEy?o418
User-4	990052148147	>!lwK4T>6t:6
User-5	959579349318	5.O>t9`D9>"d

User-6	462280505193	DZ5:v$z_n1H1
User-7	340402691136	drowssap
User-8	620125966734	F46"s|+yG)D1
User-9	763035634971	pb14q<B2Cple
User-10	408328135371	pwd@&*^786

While registration, the algorithm takes the password as input to hash function (SHA256) and computes the corresponding hash code of that password. The following table Table-2 shows the hash code of the passwords mentioned in Table-1.

Table-2 Hash Code of passwords

Password	Hash code (SHA256)
12Y6W9P^te>7	3704fa2db0a3d99f107f7b3fffcb4090e41fad2957b1f80efe02bb194307d903
v03A>2t>*!Hr	d27edb2ec0fbb552d4d85c3b535adfe7a1b318b112f36f1b4f023b07fb3f42e3
$oP#SEy?o418	ebf825cedc6f9616e799edd1c0eae2d75f39c09e81b4c4f102f0ab0b5e44badb
>!lwK4T>6t:6	0103bd5ffff92a0be3358efcbc356764e3464bc9f5eb6620a3df6afeaeecb91a
5.O>t9`D9>"d	388855f3cd00f52c8fca23f5d2670b25ac17af3e0ff2f204ac85182ed5e2d173
DZ5:v$z_n1H1	4eeb74cecb83c8c74e2bb81f55a67f18f551374adab2dfcaccadc37a41eb8a6b
drowssap	89174c9299dc17f114170bd5d98b0822ad8a3e1e6147be1a9bed0d2cd3b9990b
F46"s|+yG)D1	6d94e8ed09f410aa1ad66d9efe559182c84adb051f236aa413dd017b4bb96b9f
pb14q<B2Cple	23881c47489f0a6414f7352c50ef4921a7cdf369cce812f385757ddf0f4af470
pwd@&*^786	c2f79480fe4abc0e9d3901067475d28ca9f13dd551984e616c2f1e92b7536b6f

5.2 Calculation of Key_base value for User-1

To compute the key_base value, the algorithm takes the Aadhaar number as input and calculates the sum of digits by dividing the 12 digit Aadhaar number into 3 blocks each of 4 digits. The next step is to take the square root of the sum up to six decimal places. The calculation of key_base of User-1 is shown below;

Username: User-1

Aadhaar Number: 3332 3809 7839

Sum of blocks of Aadhaar Number: 3332 + 3809 + 7839=14980

The square root of summation: 14980 =122.392810

Key_base (Round-1) =39

Key_base (Round-2) =28

Key_base (Round-3) =10

Similarly the above parameters for other users can be calculated. The following table Table-3 shows the corresponding key_base values of all the users for 3 rounds of encryption.

Table 3 Key_base values for password encryption

User name	Aadhaar	SOA	SR_SOA	Key_base (Round-1)	Key_base (Round-2)	Key_base (Round-3)
User-1	333238 097839	14980	122.392810	39	28	10
User-2	528878 359757	22880	151.261363	26	13	63
User-3	947768 918025	24660	157.035027	03	50	27
User-4	990052 148147	23261	152.515572	51	55	72
User-5	959579 349318	26847	163.850541	85	05	41
User-6	462280 505193	17865	133.660001	66	00	01
User-7	340402 691136	4809	69.346953	34	69	53
User-8	620125 966734	15531	124.623432	62	34	32
User-9	763035 634971	16164	127.137720	13	77	20
User-10	408328 135371	12267	110.756489	75	64	89

5.3 Selection of Amicable number

The algorithm then selects a particular amicable number pair on the basis of Key_base value for first round of encryption. For user-1 the key_base value for first round is 39, so the amicable number pair will be at index 39 i.e.898,216, 980,984. Here, 39 is ODD in nature so the algorithm will fetch the second number from the selected pair i.e. 980,984.For the second and third round of encryption the algorithm will take the second and third Key_base values and repeat the same process.

Table 4 Amicable Number Pair

Index	Amicable No.		Index	Amicable No.		Index	Amicable No.	
0	220	284	34	643,336	652,664	68	4,238,984	4,314,616
1	1,184	1,210	35	667,964	783,556	69	4,246,130	4,488,910
2	2,620	2,924	36	726,104	796,696	70	4,259,750	4,445,050
3	5,020	5,564	37	802,725	863,835	71	4,482,765	5,120,595
4	6,232	6,368	38	879,712	901,424	72	4,532,710	6,135,962
5	10,744	10,856	39	898,216	980,984	73	4,604,776	5,162,744
6	12,285	14,595	40	947,835	1,125,765	74	5,123,090	5,504,110
7	17,296	18,416	41	998,104	1,043,096	75	5,147,032	5,843,048
8	63,020	76,084	42	1,077,890	1,099,390	76	5,232,010	5,799,542
9	66,928	66,992	43	1,154,450	1,189,150	77	5,357,625	5,684,679
10	67,095	71,145	44	1,156,870	1,292,570	78	5,385,310	5,812,130
11	69,615	87,633	45	1,175,265	1,438,983	79	5,459,176	5,495,264
12	79,750	88,730	46	1,185,376	1,286,744	80	5,726,072	6,369,928
13	100,485	124,155	47	1,280,565	1,340,235	81	5,730,615	6,088,905
14	122,265	139,815	48	1,328,470	1,483,850	82	5,864,660	7,489,324
15	122,368	123,152	49	1,358,595	1,486,845	83	6,329,416	6,371,384
16	141,664	153,176	50	1,392,368	1,464,592	84	6,377,175	6,680,025
17	142,310	168,730	51	1,466,150	1,747,930	85	6,955,216	7,418,864
18	171,856	176,336	52	1,468,324	1,749,212	86	6,993,610	7,158,710
19	176,272	180,848	53	1,511,930	1,598,470	87	7,275,532	7,471,508
20	185,368	203,432	54	1,669,910	2,062,570	88	7,288,930	8,221,598
21	196,724	202,444	55	1,798,875	1,870,245	89	7,489,112	7,674,088
22	280,540	365,084	56	2,082,464	2,090,656	90	7,577,350	8,493,050
23	308,620	389,924	57	2,236,570	2,429,030	91	7,677,248	7,684,672
24	319,550	430,402	58	2,652,728	2,941,672	92	7,800,544	7,916,696
25	356,408	399,592	59	2,723,792	2,874,064	93	7,850,512	8,052,488
26	437,456	455,344	60	2,728,726	3,077,354	94	8,262,136	8,369,864
27	469,028	486,178	61	2,739,704	2,928,136	95	8,619,765	9,627,915
28	503,056	514,736	62	2,802,416	2,947,216	96	8,666,860	10,638,356
29	522,405	525,915	63	2,803,580	3,716,164	97	8,754,130	10,893,230
30	600,392	669,688	64	3,276,856	3,721,544	98	8,826,070	10,043,690
31	609,928	686,072	65	3,606,850	3,892,670	99	9,071,685	9,498,555
32	624,184	691,256	66	3,786,904	4,300,136			
33	635,624	712,216	67	3,805,264	4,006,736			

5.4Calculation of Secret Key

The next step is to create a secret key for first round of encryption. In the above step the selected Amicable number for first round is 980,984. Now the binary of 980,984 is 11101111011111111000. Perform the bit padding to make it 24-bit pattern and then divide it into three blocks each of size 8-bits. So the final bit pattern becomes 000011101111011111111000.

5.5 Bitwise XOR operation and Formation of Cipher Text

A cumulative bitwise XOR operation is performed between the 8-bit binary representation of the hash characters of the password and the 24-bit binary representation of the secret key. This process conducts in three iterations using different Amicable numbers on the basis of three derived key_base values.

The algorithm reads the user-1 password hash code character by character and converts it into 8-bit binary pattern. The 8-bit binary pattern is then get XORed with the first block of secret key, the resulting bit pattern is then XORed with the second block of secret key and then finally with the third block of secret key generating the first cipher character. The XOR operation is seen in Table 5.

Table 5 Character Generation for Cipher Text Using

Block wise Cumulative XOR operation between the secret key and the password's character

Round-1								
Binary of password character	0	0	1	1	0	0	1	1
	XOR OPERATION							
First block of secret key	0	0	0	0	1	1	1	1
Intermediate cipher text bits	0	0	1	1	1	1	0	0
	XOR OPERATION							
Second block of secret key	1	1	1	1	0	1	1	1
Intermediate cipher text bits	1	1	0	0	1	0	1	1
	XOR OPERATION							
Third block of secret key	1	1	1	1	1	0	0	0
Intermediate cipher-1	0	0	1	1	0	0	1	1

Round-2								
Intermediate cipher-1	0	0	1	1	0	0	1	1
	XOR OPERATION							
First block of secret key	0	0	0	0	0	1	1	1
Intermediate cipher text bits	0	0	1	1	0	1	0	0
	XOR OPERATION							
Second block of secret key	1	0	1	0	1	1	0	1
Intermediate cipher text bits	1	0	0	1	1	0	0	1
	XOR OPERATION							
Third block of secret key	0	0	0	1	0	0	0	0
Intermediate cipher-2	1	0	0	0	1	0	0	1
Round-3								
Intermediate cipher-2	1	0	0	0	1	0	0	1
	XOR OPERATION							
First block of secret key	0	0	0	0	0	0	0	1
Intermediate cipher text bits	1	0	0	0	1	0	0	0
	XOR OPERATION							
Second block of secret key	0	0	0	0	0	1	1	0
Intermediate cipher text bits	1	0	0	0	1	1	1	0
	XOR OPERATION							
Third block of secret key	0	0	0	1	0	1	1	1
Final Cipher Characters bits	1	0	0	1	1	0	0	1

The final cipher text will be obtained after doing all three rounds on each of the user-1 password's hash code characters. The Table-6 shows the final ciphertext of all the users' password.

Table-6 Encrypted password

<table>
<tr><th>Password</th><th>Encrypted password</th></tr>
<tr><td>12Y6W9P^te>7</td><td>²²*.³*)²)*»±². ¹²!»²±⁰¶³*"²+!±*⁰¶!·! ¹*±⁰!³±!*±*³+*»²)+⁰¹²¶!¸*⁰</td></tr>
<tr><td>v03A>2t>*!Hr</td><td>S60QSV6QT7RVV116S3S?1T4V141USRQ0U5V45?V556R42R5V3R76
4V70RV4R36Q4</td></tr>
<tr><td>$oP#SEy?o418</td><td>ÝÚÞ³⁰¹½ØÝßØ¾Þ±¾¹³¾Ý¼±±Ýßß¹Ø»ÝÚÝºß¼½Þ¸±Ø»±Ý³¹Ú¿Ø¿Þ¹»ºÞ»
ÙÚ»Ú½Ý¿¿ÚÙßÚ</td></tr>
<tr><td>>!lwK4T>6t:6</td><td>ÌÌÌ ªË«««««ÄÌ¬Ì ¨ÌÌËÆ¨«- -ÌËËËËË¨ÌËËË -Ä«Ë¨ ËËÌÌ¬Ìª«Ë¬«¨¬¨- ÄÌ¬</td></tr>
<tr><td>5.O>t9`D9>"d</td><td>/(((%%¶/¿´¸¸¶%.¿(¶¿½./¶%´.&';¾.%½¿-'½¶/µ,¶¶.¶.,$½¿(%-(.µ´%µ.´-'/</td></tr>
<tr><td>DZ5:v$z_n1H1</td><td>{-x{-w|wx{~--w}yyzx}wyy}|x{-~|x{}-wz-</td></tr>
<tr><td>drowssap</td><td>áæçä…á „æææäàãç…à à¸ç áæà„ƒƒá àäçã æàáàƒ</td></tr>
<tr><td>F46"s|+yG)D1</td><td>äŠ„æˆæä Šå„¸ ââ¸ää……äŠæåæãˆ„âää †¸ååâ„¸ƒääá„ááâŠå</td></tr>
<tr><td>pb14q<B2Cple</td><td>B@KKA GDGKI&C!FGAG&D@EBEC%&GIBA!D '&@FI
%KAB&@KEDED"&C&G!&GDC</td></tr>
<tr><td>pwd@&*^786</td><td>£Ä¥ÇËÄËÄ¥¦Ä¢¡£Ä¦Ë¤ÄËÄÄÄÄÇÄÇÆ¤ÄË£¢Ë¥ÄÄ¤¤ÆÆÄËËÄ¦ÄÄÄ£</td></tr>
</table>

6. Experimental Outcome and Results

After performing the simulation of proposed approach using python, the first interface that will appear is the "User Signup" page, in which the user have to provide the details including username, password and Aadhaar number. After the successful registration the password of the user is get encrypted using the input Aadhaar number in the database.

Figure 1 User Registration

At each login user have to enter the credentials, the password and the Aadhaar number will not appear in original form as shown in the figure 2. This will prevent the possibility of shoulder surfing attack.

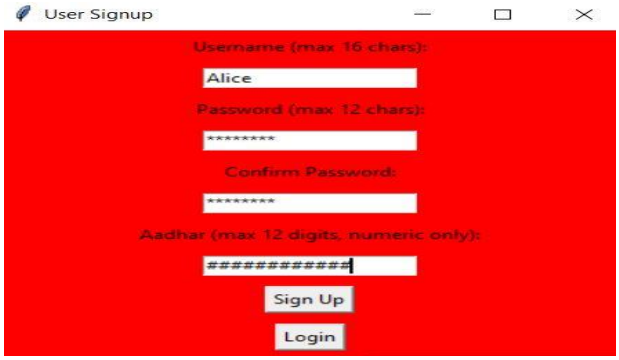

Figure 2 User Login page

After the successful login a dialog box of Signup Successful will appear as shown in figure 3

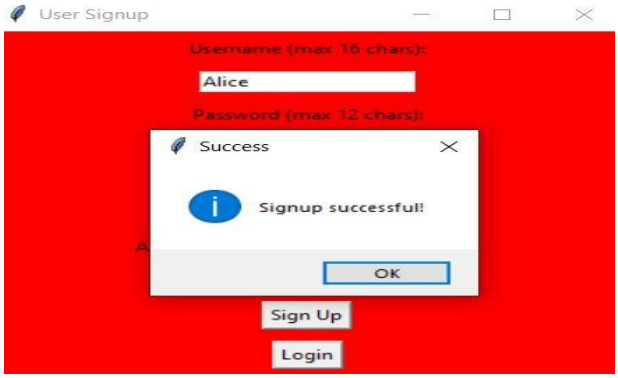

Figure 3 Successful Login

In case some credentials mismatches i.e. aadhaar is inputted with non-numeric values or password doesn't match, then the error dialog box will appear as shown in figure 4.

Figure 4 Authentication failed

For authentication of each user, the proposed algorithm allows the user to enter username, Aadhaar number and the password. The algorithm uses the Aadhaar number and the password as input to execute the algorithm, after generating the cipher text it will compare the newly generated cipher with the cipher stored in the database. If it matched, authentication passed otherwise failed. The database contains only username and encrypted password which is shown in the figure 5.

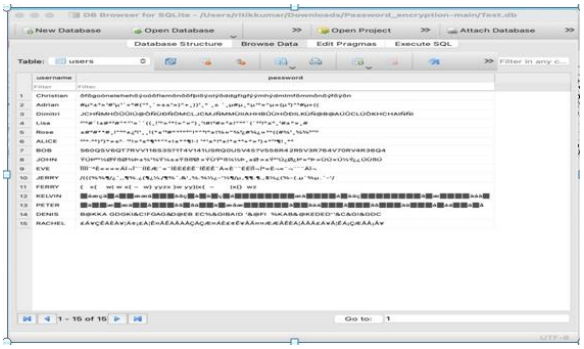

Figure 5 Encrypted password in database

7. Strength of ABES

(a) The size of each encrypted password is not fixed, so it does not reflect any relation with hashing.

(b) Some of the characters in encrypted password are non printable characters; this will definitely frustrate the attacker to perform cryptanalysis.

(c) In the database, only username and encrypted password is stored. The credentials which is used to perform encryption is not stored anywhere.

(d) At the time of login user have to enter the username, aadhaar number and password; this makes the approach as multifactor authentication.

(e) The encryption in the proposed approach is iterative in nature and comprises of three rounds.

(f) Amicable numbers are used in such a manner that it will increase the complexity of the proposed algorithm.

(g) The encryption time is less as compare to the DES and AES and other approaches proposed by the researchers. After comparing the encryption time of an approach Jumbling Salting (JS) proposed by Churi *et. al.*, Advanced Encryption Standard (AES), Data Encryption Standard (DES) and the proposed approach named as Amicable Based Encryption System (ABES) algorithm, the Encryption time of AAES is less as compare to JS, AES, and DES algorithms. The following graph in figure 7 illustrates the comparison of encryption time of above mentioned schemes.

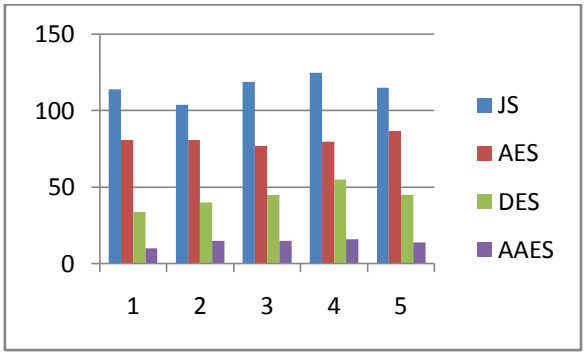

Figure 6 Encryption time of JS, AES, DES and ABES

(h) Though the algorithm is complex still the login time along with authentication is less as shown in the following graph (figure 7).

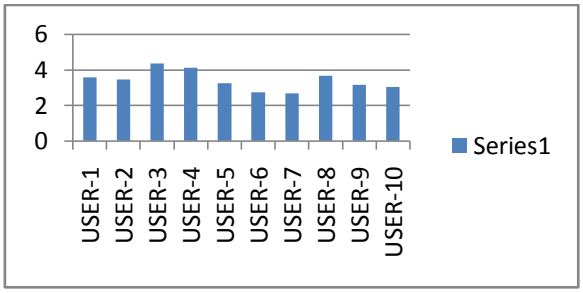

Figure 7 Login time

8. Conclusion and Future Scope

This study suggests a unique password encryption technique that might offer users who must use passwords to safeguard their accounts ease and security assurance. With this method, the database simply stores the encrypted username and password; no attributes that are needed during the whole encryption process are saved there, which makes it challenging for an attacker to do cryptanalysis. In the database the encrypted passwords are of different sizes which surely hide its relation with SHA256. In future we can enhance this approach by implementing an iterative encryption that will re-encrypt the stored encrypted password in the database after a specific period of time. The user does not need to change his/her password.

Referneces

[1] "Password Security Report: 83% of Users Surveyed Use the Same Password for Multiple Sites - Cyclonis", Cyclonis, 2019. [Online]. Available: https://www.cyclonis.com/report-83-percent-userssurveyed-. [Accessed: 29- Oct- 2019].

[2] William Stallings.2011. Cryptography and Network Security Principles and Practice, 5th Edition.Pearson Education.

[3] Ramesh Yegireddi, R Kiran Kumar, "A survey on conventional encryption algorithms of Cryptography," 2016(IEEE) International Conference on ICT in Business Industry & Government (ICTBIG), Indore, 2016, pp. 1-4,

[4] R. Das, S. Dutta, "A Private Key Encryption Scheme based on Amicable Number with User defined Cipher Block Sequencing Techniques", International Journal of Computer Sciences and Engineering, Vol.6, Issue.5, pp.34-41, 2018

[5] J. K. Mandal and Mangalmay Das, "Fibonacci Based Position Substitution (FBPS) Encoder for Secured Message Transmission", IEEE International Advance Computing Conference (IACC) Patiala, India, pp.964-970, 6-7 March 2009.

[6] Udepal Singh and UpasnaGarg, "An ASCII value-based text data encryption System", "International Journal of Scientific and Research Publications (ISSN 2250-3153)", Volume 3, Issue 11, pp. 1-5,November 2013.

[7] SnehalSherkhane, Amit Waghmare, SurajDalvi, Shreya Bamne, "Hybrid Data Encryption using Colour code and Armstrong number", "International Journal of Engineering Science and Computing", Volume 7, Issue 4, pp. 10300-10305,April,2017.

[8] Laiphrakpam Dolendro Singh and Khumanthem Manglem Singh, "Implementation of Text Encryption using Elliptic Curve Cryptography", "Eleventh International Multi-Conference on Information Processing-2015 (IMCIP-2015)"," Procedia Computer Science", doi: 10.1016/j.procs.2015.06.009, Volume 54, pp. 73-82,2015.

[9] NishthaMathur and Rajesh Bansode, "AES Based Text Encryption Using 12 Rounds with Dynamic Key Selection", "7th International Conference on Communication, Computing and Virtualization 2016"," Procedia Computer Science", doi: 10.1016/ j.procs. 2016 .03.131, Volume 79, pp. 1036-1043, 2016.

[10] A.Nath, S. Ghosh and M. Mallick, "Symmetric Key Cryptography Using Random Key Generator.", in Proceedings of the 2010 International Conference on Security & Management, Las Vegas, Nevada, USA, 2010, pp. 234-242.
[11] A. S. Yerne and F. I. Z. Qureshi, "Design 3D Password with session based technique for login security in Smartphone," 2016 Online International Conference on Green Engineering and Technologies (ICGET), Coimbatore, 2016, pp.1-4.doi: 10.1109/GET.2016.7916769
[12] Dhakar, R. S., Gupta, A. K., & Sharma, P. (2012, January). "Modified RSA Encryption Algorithm (MREA)". In Advanced Computing & Communication Technologies (ACCT), 2012 Second International Conference on (pp. 426-429). IEEE.
[13] P. M. Aiswarya, A. Raj, D. John, L. Martin, and G. Sreenu, "Binary RSA encryption algorithm," 2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), Kumaracoil, 2016, pp. 178-181.
[14] Al-Hamami, A. H., &Aldariseh, I. A. (2012, November). Enhanced Method for RSA Cryptosystem Algorithm. In Advanced Computer Science Applications and Technologies (ACSAT), 2012 International Conference on (pp. 402-408).
[15] Gupta, S., & Sharma, J. A Hybrid Encryption Algorithm based on RSA and Diffie-Hellman. 978-0-7695-4587-5/11.

Mohammed Shakeel is a Ph.D. research scholar and also working as an Assistant Professor in the Department of Computer Applications at Invertis University, Bareilly, Uttar Pradesh, India. He is MCA, PGDCA. He has 18 years of teaching experience. He has 8 research papers published in various National and International journals. He has participated in numerous conferences, workshops, and faculty development programmes.

Dr. Akash Sanghi is currently working as an Associate Professor in the Computer Science and Engineering Department at Invertis University in Bareilly, Uttar Pradesh, India. He is Ph. D. (CSE), M.Tech (SE) and B.Tech (CS). He has 19 years of teaching experience. He has over 20 research papers published in various National and International journals. He has participated in numerous conferences, workshops, and faculty development programmes. He has mentored over ten M.Tech students and is currently mentoring six Ph..D. research scholars. Biometrics, MANETs, VANETs, and network security are among his research interests. He is a member of numerous professional bodies, including IFERP, IAE, and IRED.

Professor Y.D.S. Arya is the Vice-Chancellor of Invertis University, Bareilly, India. He is Ph. D. in Computer Science and Engineering. He has over 55 National and International publications to his credit. His area of interest include computer programming and database systems. He is the Chairman of the visiting team for NBA accreditation of undergraduate/postgraduate engineering programs at Colleges/ Institutes/Universities. He is a Mentor/Auditor for a few Colleges/Institutions/Universities in the MHRD's World Bank-sponsored TEQIP III project. He is also the Head of the AICTE Team in UGC expert committees for University evaluation.

Dr. Gaurav Agarwal is currently working as HoD and Associate Professor in the Computer Science and Engineering Department at Invertis University in Bareilly, Uttar Pradesh, India. He is Ph. D. (CSE), M.Tech (IT) and B.Tech (CSE). He has 18 years of teaching experience. He has over 60 research papers published in various National and International journals. He has supervised three Ph.D. students and is currently mentoring six Ph.D. research scholars. Network security, Agile Technology and Cloud Computing are among his research interests. He is editorial board member in many National and International Journals.