International Journal of Electrical and Computer Engineering (IJECE)
Vol. 7, No. 2, April 2017, pp. 1032~1041
ISSN: 2088-8708, DOI: 10.11591/ijece.v7i2.pp1032-1041

DARE Algorithm: A New Security Protocol by Integration of Different Cryptographic Techniques

John Mark B. Espalmado, Edwin R. Arboleda
Department of Computer and Electronics Engineering, College of Engineering and Information Technology,
Cavite State University – Indang, Cavite, Philippines

Article Info

Article history:

Received Nov 14, 2016
Revised Mar 8, 2017
Accepted Mar 25, 2017

Keyword:

AES algorithm
DES algorithm
Digital signature algorithm
Hybrid cryptography
RSA encryption

ABSTRACT

Exchange of information between computer networks requires a secure communications channel to prevent and monitor unauthorized access, modification and denial of the computer network. To address this growing problem, security experts sought ways to advance the integrity of data transmission. Security Attacks compromises the security and hence hybrid cryptographic algorithms have been proposed to achieve safe service in the proper manner, such as user authentication and data confidentiality. Data security and authenticity are achieved using these algorithms. Moreover, to improve the strength and cover each algorithm's weaknesses, a new security algorithm can be designed using the combination of different cryptographic techniques. This design uses Digital Signature Algorithm (DSA) for authentic key generation, Data Encryption Standard (DES) for key scheduling, and Advanced Encryption Standard (AES) and Rivest–Schamir–Adleman Algorithm (RSA) in encrypting data. This new security algorithm has been proposed for improved security and integrity by integration of these cryptographic techniques.

Corresponding Author:

Edwin R. Arboleda,
Department of Computer and Electronics Engineering,
College of Engineering and Information Technology,
Cavite State University,
Main, Indang, Cavite, Philippines.
Email: edwin.r.arboleda@cvsu.edu.ph

1. INTRODUCTION

The science of encrypting and decrypting data using mathematics is called cryptography. It enables users to store private information and send it across mediums susceptible to attacks of hackers thereby reducing the compromise in data security. This makes it hard for any unauthorized interference to garble with sensitive data.

Cryptography works with the use of algorithms. A cipher or cryptographic algorithm is a mathematical function used in the process of encryption and decryption process. To hide confidential information, data or message can be encrypted using keys generated from a word, a number, or a phrase. This plaintext is encrypted to different ciphertexts using different keys. The strength of the cryptographic algorithm and the secrecy of the key greatly affects the security of encrypted data [1].

However, there are still persisting threats due to the familiarity of these cryptographic techniques added with its simplicity to potential attackers. Present problems include Brute-force attacks, low encryption strength, and insufficient randomness in key generation [2]. Moreover, data privacy is challenged since passwords can be also be stolen especially when logged in an unsecure network. The system's vulnerability to attacks must be less probable as a function of the security parameter. In this paper, we address this matters by proposing a mixed encryption algorithm by combining the presented cryptography techniques and utilize their strengths and as much as possible, reduce the weakness of one technique with that of another.

2. RESEARCH METHOD
2.1. Basic Principles involved in the Proposed Scheme
2.1.1. DSA (Digital Signature Algorithm)

Authentication and authenticity are ensued using Digital Signature Algorithm [3]. A parameter required is a secret key x such that it satisfies $0 < x < q$ in computing the private and public keys for a single user. Get the public key using the formula: $y=g^x \bmod p$. To solve for the modular exponentiations $h^{(p-1)}/q \bmod p$ and $g^x \bmod p$, exponentiation by squaring can be applied [4].

2.1.2. DES Key Scheduling

The key scheduling of the DES separates the 56-bit key is into two 28-bit halves; wherein each half is treated separately. These halves are rotated left by one or two bits per each successive round as specified in Table 1. Then, 48 subkey bits are selected by Permuted Choice 2 (PC-2) — 24 bits from the left half, and 24 from the right. There are varying sets of bits in each subkey because of the rounds; each bit is used in about 14 out of the 16 subkeys [5].

Table 1. Shifts for each round in DES Key Scheduling

Round	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Shift	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

2.1.3. AES (Advanced Encryption Standard) Algorithm

AES algorithm is a symmetric block cipher, which offers better security and efficiency than DES [3] in message encryption, is a widely-used algorithm primarily executed using a software. It has low memory requirements making it appropriate for fast usage in some constrained environment. AES encryption process is operated in a 4 x Nb matrix (also known as state) where Nb is equivalent to the quotient of the data block length and 32 [6]. Encryption comprises the following steps [7], [8]:

a. AddRoundKey

Each byte in the state matrix is XORed with the Roundkey value is XORed.

b. SubBytes

In this stage, each byte is substituted with its equivalent byte as defined from a look-up table. Refer to Table 2(a) for the look-up table during encryption and Table 2(b) during decryption.

Table 2. AES SubByte Transformation Table (a) & Inverse Subbyte Transformation (b) [3]

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	db	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

(a)

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

(b)

c. ShiftRows

With an incrementing number of rows, a cyclic shift to the left of each byte is operated in each column of the matrix.

d. Mixed Columns

Yielding a four-term polynomial by treating the matrix column by column and multiplying with another polynomial as stated in the standard of over GF (2^8) comprises this step.

2.1.4. RSA

RSA algorithm is a public key encryption algorithm, utilizes a public key and a private key. The keys appear as pairs, and the corresponding key must be utilized for encryption and decryption operation [4], [9].

a) Confidential prime numbers p and q are chosen in the same order of magnitude.
b) Compute for $n = p\phi q$, $\phi(n) = (p\text{-}1)\ (q\text{-}1)$, where $\phi(n)$ is the Euler function value of n.
c) Pick an integer e to satisfy $1 < e < \phi(n)$, $\phi(n)$
d) Generate a decryption key d as follows: $(e \times d) \bmod \phi(n) = 1$

Encryption $c = m^e \bmod n$

Decryption $\mathrm{m} = c^d \bmod n$

The encryption keys are *e* and *n while* d and n are the decryption keys. The ciphertext is *m* while *c* is the decrypted ciphertext. The public keys are (e, n) while (d, n) constitute the private key, prime numbers *p* and *q* should be discarded.

2.2. Proposed Hybrid Algorithm Architecture

It is desired to communicate data with high security. At present, various types of cryptographic algorithms provide high security to information on controlled networks. These algorithms are required to provide data security and users authenticity. This new security protocol has been designed for better security using a combination of DSA key generation and DES key scheduling with AES subByte Transformation and RSA encryption.

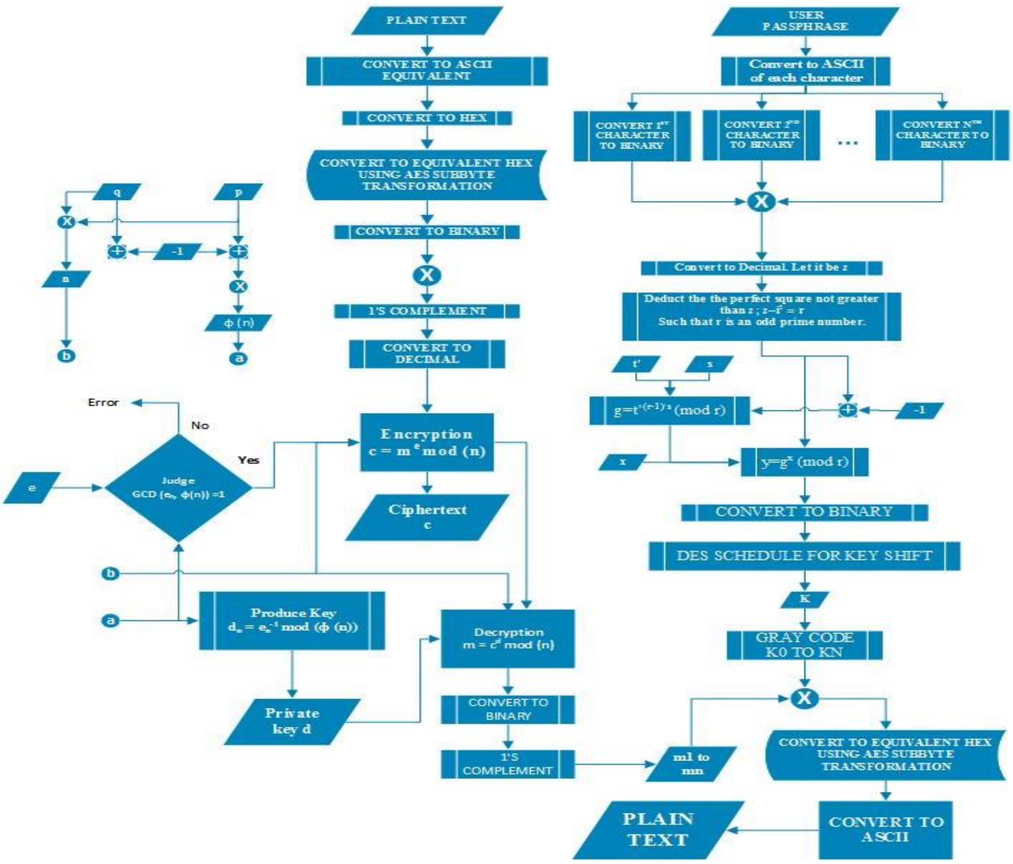

Figure 1. Block Diagram of Proposed Algorithm

As shown in Figure 1, the proposed algorithm requires a passphrase shared by both the sender and receiver to securely authenticate data exchange. The passphrase is encrypted using the first phase of Digital Signature Algorithm (DSA) in which different users in the system share the same algorithm parameters. In the encryption stage, the passphrase is encrypted to ensure that hackers may not intrude and interfere with the transmission. The encrypted public key (y) is then converted into binary and n keys are derived using DES key Scheduling. Those keys are XORed with the plain text, concealed using AES subByte Transformation, to securely hide the original plain text. The data is then complemented, converted to decimal and encrypted with RSA to get the ciphertext for more confidentiality. During the decryption stage, same steps are applied to the receiver in which they must also input passphrase, encrypted with DSA, key generate with DES to allow access to data. The keys will then be XORed to the decrypted ciphertext with RSA to gain the AES subByte equivalent. Plaintext will be recovered using the table for subByte Transformation.

This proposal, DARE (DSA/DES – AES – RSA Encryption) algorithm is composed of three components: Key generation which uses DSA and DES; data encryption and decryption which uses AES and RSA.

2.3. Key Generation

The Key generator works as follows:

a. Alice inputs a passphrase of up to n characters (l_1 l_2 l_3 … l_n) which is converted to its binary equivalent, bitwise XORed, gray coded, and converted to decimal form
b. Alice picks a perfect square not greater than the decimal and deducts it from the number such that its difference is an odd prime number. Let this difference be *p1*.
c. Alice chooses a number q1 such that it is a prime factor of *p1-1* and a number *h'* such that it is less than *p1-1*.
d. Alice computes *g=h'(p1-1)/q1 (mod p)*
e. Alice chooses a private key *x*.
f. Alice computes $y = g^x \ (mod\ p1)$.
g. Alice converts *y* to binary uses the table for round shifts in DES key scheduling which results to k_0 up to k_n.

2.4. Encryption

The encryption algorithm works as follows: to encrypt a message *m* to Bob under Alice' public key (k_0 up to k_n) Alice picks a prime number p2 such that p2 > (largest ASCII equivalent of the message) and a random number q.

a. Alice calculates n= p2 * q2 and ϕ(n) = (p2-1) * (q2-1).
b. Alice chooses e such that (0 <e< ϕ(n)).
c. Alice computes for d - the inverse modulo of e. using Euclidian Algorithm e * d mod ϕ(n))= 1
d. Alice publishes her public key (e, n) and (d, n)
e. Alice converts the message into its HEX equivalent.
f. Alice uses the table for SubByte Transformation in AES to hide the message and converts them into binary.
g. Alice uses her keys to bitwise XOR it with her binary message.
h. Alice computes its 1's complement and converts it to decimal form. Let it be m.
i. Alice uses the c = me (mod n) to encrypt m.

2.5. Decryption

The decryption algorithm works as follows: to decrypt a ciphertext {c_1, c_2, c_3 … c_n} with keys public keys (k_0 - k_n, e, d, n)

a. Bob inputs the same shared passphrase for authentication which allows him to access the ciphertext.
b. Using the public key *d*, Bob calculates the message: $m = c^d \mod n$
c. Bob converts the m_1 to m_n from decimal to binary form and get the 1's complement of each.
d. Bob uses the keys k_0 to k_n in Bitwise XORing each of the 1's complemented m.
e. Bob then uses the inverse SubByte Transformation table in AES to recover the hex equivalent of the message.
f. The hexes are then converted to ASCII equivalent to decrypt the plain text.

3. RESULTS AND ANALYSIS

In this section, we calculated the ciphertext produced when a user passphrase "dare" is used to encrypt a message "Waltz, nymph, for quick jigs vex Bud." to prove that this proposal is achievable.

3.1. Key generation

1. With this proposal, a user can now input a Passphrase for more secure key generation:
   Example: *dare*
   ASCII Equivalent
   d = 100; a = 97: r = 114; e = 101
2. XOR the binary equivalent of the ASCII passphrase and it will yield
   0001 0010
3. Gray Code the XORed passphrase.
   Answer = 0 0 0 1 1 1 0 1
4. Convert it to decimal form.
   Answer = 27
5. Get the difference between the decimal with the perfect square not greater than the decimal as long as its difference is an odd number
   Answer = 27 – 16 = 11 In this case; 16 will be deducted from 27.
6. Using **DSA** key pair generation, let:
   p = 11 prime number between 512 to 1024 bits long
   p-1 = 10
   q = 5
   h' = 7 such that h' < p-1
   $g=h'^{(p-1)/q} \pmod p$
   $= 7^{10/5} \bmod 13$
   = 5
   x=3 private key
   $y = g^x \pmod p$
   $= 5^3 \pmod{13}$
   = 4
7. Convert y to 8 bit binary
   0 0 0 0 1 0 0 0
8. Using **DES** schedule for key shift in DES algorithm, find k1 to kn.
   K1 = K17 = K33 = 0001 0000
   K2 = K18 = K34 = 0010 0000
   K3 = K19 = K35 = 1000 0000
   K4 = K20 = K36 = 0000 0010
   K5 = K21 = K37 = 0000 1000
   K6 = K22 = K38 = 0010 0000
   K7 = K23 = 1000 0000
   K8 = K24 = 0000 0010
   K9 = K25 = 0000 0100
   K10 = K26 = 0001 0000
   K11 = K27 = 0100 0000
   K12 = K28 = 0000 0001
   K13 = K29 = 0000 0100
   K14 = K30 = 0001 0000
   K15 = K31 = 0100 0000
   K16 = K32 = 1000 0000
9. **RSA** key generation
   p= 127; such that p>m
   q= 5 any prime number
   n= p * q = 127 * 5 = 635
   $\phi(n) = (p-1) * (q-1) = 126 * 5 = 504$
   Let e = 11 $(0 < e < \phi(n))$
   Let d= inverse modulo of e

To compute for d, using Euclidean Algorithm:
e * d mod ϕ(n))= 1
5 * d mod 504 = 1
504 +11y = 1
504 = 45(11) + 9 [eq.1]
11 = 1(9) + 2 [eq.2]
9 = 4(2) +1 [eq.3]

Extended Euclidean Algorithm (Back Substitution)
1 = 9 – 4(2) eq. (4): substitute eq.3
1 = 9 –4[11-1(9)] eq. (5): substitute eq. 2
1 = 9 – 4(11) + 4(9) eq. (6): extending eq. 5
1 = 5(9) – 4(11) eq. (7): combining similar terms in eq. 6
1 = 5[504-45(11)]-4(11) eq. (8)
1 = 5(504) – 5(45)(11) – 4(11) eq. (9)
1=5(504) – 49(11) eq. (10)
d= -49 mod 504
d= 504 – 49
d=455
Public keys (e,n) = (11 , 635)
Private keys (d,n) = (383, 635)

Encryption

1. Convert the message into its ASCII equivalent

Letter	ASCII	Letter	ASCII
W	87	i	105
a	97	c	99
l	108	k	107
t	116	j	106
z	122	g	103
n	110	s	115
y	121	v	118
m	109	e	101
p	112	x	120
h	104	B	66
f	102	u	117
o	111	d	100
r	114	,	44
q	113	.	46
u	117	(space)	32

2. Convert it to its HEX equivalent

Letter	ASCII	Hex	Letter	ASCII	Hex
W	87	57	i	105	69
a	97	61	c	99	63
l	108	6C	k	107	6B
t	116	74	j	106	6A
z	122	7A	g	103	67
n	110	6E	s	115	73
y	121	79	v	118	76
m	109	6D	e	101	65
p	112	70	x	120	78
h	104	68	B	66	42
f	102	66	u	117	75
o	111	6E	d	100	64
r	114	72	,	44	2C
q	113	71	.	46	2E
u	117	75	(space)	32	20

3. Using AES using SubByte transformation, take the first bit of the hex as the row and second (x) and the second bit as the column (y).
W=17 = m1
a=ef = m2
l=50 = m3
t=92 = m4
z=da = m5
,=71 = m6 = m13
space=b7 = m7 = m14 = m18 = m24 = m29 = m33
n=9f = m8
y=b6 = m9
m=3c = m10
p=51 = m11
h=45 = m12
f=33 = m15
o=9f = m16
r=40 = m17
q=a3 = m19
u=9d = m20 = m35
i=f9 = m21
c=fb = m22
k=7b = m23
j=02 = m25
g=85 = m27
s=8f = m28
v=38 = m30
e=4d = m31
x=bc = m32
B=2c = m34
u=9d = m35 = m20
d=43 = m36
.=31 = m37

The message is now in the form:
17 EF 50 92 DA 71 B7 9F B6 3C 51 45 71 B7 33 9F 40 B7 A3 9D F9 FB 7B B7 02 F9 85 8F B7 38 4D BC B7 2C 9D 43 31

4. Convert each it into binary form

M1	0001 0111	M20	1001 1101
M2	1110 1111	M21	1111 1001
M3	0101 0000	M22	1111 1011
M4	1001 0010	M23	0111 1011
M5	1101 1010	M24	1011 0111
M6	0111 0001	M25	0000 0010
M7	1011 0111	M26	1111 1001
M8	1001 1111	M27	1000 0101
M9	1011 0110	M28	1000 1111
M10	0011 1100	M29	1011 0111
M11	0101 0001	M30	0011 1000
M12	0100 0101	M31	0100 1101
M13	0111 0001	M32	1011 1100
M14	1011 0111	M33	1011 0111
M15	0011 0011	M34	0010 1100
M16	1001 1111	M35	1001 1101
M17	0100 0000	M36	0100 0011
M18	1011 0111	M37	0011 0001
M19	1010 0011		

5. The message is now XORed with corresponding keys K1 to KN. It will now become:

0000 0111	1100 1111	1101 0000	1001 0000	1101 0010	0101 0001
1111 0111	1001 1101	1011 0010	0010 1100	0001 0001	0100 0100
0111 0101	1010 0111	0111 0011	0001 1111	0101 0000	1001 0111
0010 0011	1001 1111	1111 0001	1101 1011	1111 1011	1011 0101

0000 0110	1110 1001	1100 0101	1000 1110	1011 0011	0010 1000
0000 1101	0011 1100	1010 0111	0000 1100	0001 1101	0100 0001
0011 1001					

6. Get the 1's complement

1111 1000	0011 0000	0010 1111	0110 1111	0010 1101	1010 1110
0000 1000	0110 0010	0100 1101	1101 0011	1110 1110	1011 1011
1000 1010	0101 1000	1000 1100	1110 0000	1010 1111	0110 1000
1101 1100	0110 0000	0000 1110	0010 0100	0000 0100	0100 1010
1111 1001	0001 0110	0011 1010	0111 0001	0100 1100	1101 0111
1111 0010	1100 0011	0101 1000	1111 0011	1110 0010	1011 1110
1100 0110					

7. Convert it to decimal

248 = m1	30 = m2	47 =m3	111 = m4	45 = m5
174 = m6	8 = m7	98 = m8	77 = m9	211 = m10
238 = m11	187 = m12	138 = m13	88 = m14	140 = m15
224 = m16	175 = m17	104 = m18	220 = m19	48 = m20
14 = m21	36 = m22	4 = m23	74 = m24	249 = m25
22 = m26	58 = m27	113 = m28	76 = m29	215 = m30
242 = m31	195 = m32	88 = m33	243 = m34	226 = m35
190 = m36	198 = m37			

8. The decimals will be encrypted using RSA

RSA: Encryption

1. **Using formula C = m^e (mod n)**

C1	248^{11} * (mod 635) = 542	C2	30^{11} * (mod 635) = 450
C3	47^{11} * (mod 635) = 38	C4	111^{11} * (mod 635) = 631
C5	45^{11} * (mod 635) = 605	C6	174^{11} * (mod 635) = 419
C7	8^{11} * (mod 635) = 32	C8	98^{11} * (mod 635) = 412
C9	77^{11} * (mod 635) = 588	C10	211^{11} * (mod 635) = 396
C11	238^{11} * (mod 635) = 377	C12	187^{11} * (mod 635) = 88
C13	138^{11} * (mod 635) = 62	C14	88^{11} * (mod 635) = 587
C15	140^{11} * (mod 635) = 590	C16	224^{11} * (mod 635) = 435
C17	175^{11} * (mod 635) = 55	C18	104^{11} * (mod 635) = 84
C19	220^{11} * (mod 635) = 600	C20	48^{11} * (mod 635) = 182
C21	14^{11} * (mod 635) = 459	C22	36^{11} * (mod 635) = 521
C23	4^{11} * (mod 635) = 129	C24	74^{11} * (mod 635) = 314
C25	249^{11} * (mod 635) = 454	C26	22^{11} * (mod 635) = 103
C27	58^{11} * (mod 635) = 12	C28	113^{11} * (mod 635) = 557
C29	76^{11} * (mod 635) = 341	C30	215^{11} * (mod 635) = 460
C31	242^{11} * (mod 635) = 163	C32	195^{11} * (mod 635) = 560
C33	88^{11} * (mod 635) = 587	C34	243^{11} * (mod 635) = 192
C35	226^{11} * (mod 635) = 276	C36	190^{11} * (mod 635) = 500
C37	198^{11} * (mod 635) = 352		

Decryption

1. Use RSA Decryption to get the value of **m** using the formula: **m = c^d mod n**

248	30	47	111	45
174	8	98	77	211
238	187	138	88	140
224	175	104	220	48
14	36	4	74	249
22	58	113	76	215
242	195	88	243	226
190	198			

2. Convert each decrypted decimal into binary form and get its 1's complement.

1111 1000	0011 0000	0010 1111	0110 1111	0010 1101	1010 1110
0000 1000	0110 0010	0100 1101	1101 0011	1110 1110	1011 1011
1000 1010	0101 1000	1000 1100	1110 0000	1010 1111	0110 1000
1101 1100	0110 0000	0000 1110	0010 0100	0000 0100	0100 1010
1111 1001	0001 0110	0011 1010	0111 0001	0100 1100	1101 0111
1111 0010	1100 0011	0101 1000	1111 0011	1110 0010	1011 1110

3. XOR the 1's complement with corresponding keys K1 to Kn and convert it to hex.

0000 0111	1100 1111	1101 0000	1001 0000	1101 0010	0101 0001
1111 0111	1001 1101	1011 0010	0010 1100	0001 0001	0100 0100
0111 0101	1010 0111	0111 0011	0001 1111	0101 0000	1001 0111
0010 0011	1001 1111	1111 0001	1101 1011	1111 1011	1011 0101
0000 0110	1110 1001	1100 0101	1000 1110	1011 0011	0010 1000
0000 1101	0011 1100	1010 0111	0000 1100	0001 1101	0100 0001
0011 1001					

4. Get the 1's complement of the resulting bits in number 3.

1111 1000	0011 0000	0010 1111	0110 1111	0010 1101	1010 1110
0000 1000	0110 0010	0100 1101	1101 0011	1110 1110	1011 1011
1000 1010	0101 1000	1000 1100	1110 0000	1010 1111	0110 1000
1101 1100	0110 0000	0000 1110	0010 0100	0000 0100	0100 1010
1111 1001	0001 0110	0011 1010	0111 0001	0100 1100	1101 0111
1111 0010	1100 0011	0101 1000	1111 0011	1110 0010	1011 1110
1100 0110					

5. Convert it to HEX
   17 EF 50 92 DA 71 B7 9F B6 3C 51 45 71 B7 33 9F 40 B7 A3 9D F9 FB 7B B7 02 F9 85 8F B7 38 4D BC B7 2C 9D 43 31
6. From the inverse S box, locate the two-bit HEX equivalent of the message and convert it back to ASCII.
   57 61 6c 74 7a 2c 20 6e 79 6d 70 68 2c 2e 66 6e 72 20 71 75 69 63 6b 20 6a 69 67 73 20 76 65 78 20 42 75 64 2e
7. Convert it to ASCII equivalent
   87 97 108 116 122 44 32 110 121 109 122 104 44 32 102 111 114 32 113 117 105 99 107 32 106 105 103 115 32 118 101 120 32 66 117 100 46
8. Convert it to equivalent message.
   The original plaintext message is recovered to be:
   Waltz, nymph, for quick jigs vex Bud.

Results show that the proposal is doable since the original message has been recovered from following the steps included in the section. In this proposal, both the sender and receiver must input a passphrase which undergoes encryption in DSA. The encrypted password is then shifted with rounds using DES key scheduling. Failure of inputting the correct passphrase would result in a different message since the keys used in encrypting and decrypting the message obtained from the passphrase. This functionality strengthens data authenticity of this proposal. Moreover, hiding data using the SubByte Transformation in AES reinforces the encryption strength of RSA.

4. CONCLUSION

In this paper, we proposed a robust protocol using DSA and DES for key generation and scheduling and AES and RSA for encryption and decryption of information. The combination of these different cryptography algorithms delivers a maximized efficiency, amending or compensating each other's deficiencies.

Moreover, it offers a more protected exchange of data since both ends have an encrypted passphrase required to decrypt the message. Therefore, it requires more effort to the hackers to discover the message itself because they have to decrypt the passphrase first. This new hybrid protocol yields a strong cryptosystem together with a secure key encryption management system ensuring all security goals.

ACKNOWLEDGEMENTS

The authors are grateful to Dr. Robles and Dr. Hosea Matel for the encouragement and financial support, especially in providing publication fees.

REFERENCES

[1] M. J. Dubal, *et al.*, "Design of New Security Algorithm Using Hybrid Cryptography Architecture," *IEEE*, vol/issue: 3(11), pp. 99-101, 2011.
[2] D. Lazar, *et al.*, "Why does cryptographic software fail? A case study and open problems," *ACM*, pp. 1-7, 2014.
[3] G. Mateescu and M. Vladescu, "A Hybrid Approach of System Security for Small and Medium Enterprises: combining different Cryptography techniques," *2013 Conference on Computer Science and Information Systems on IEEE*, pp. 659-662, 2013.
[4] M. Y. Ree, "Internet Security: cryptographic principles, algorithms, and protocols, PO19 8SQ," England, John Wiley & Sons Ltd, 2003.
[5] H. Yang, "Herong Yang," 2015. [Online]. Available: http://www.herongyang.com/Cryptography/DES-. [Accessed 1 November 2016].
[6] X. Mingyuan, "A Mixed Encryption Algorithm Used in Internet of Things Security Transmission System," *IEEE*, vol/issue: 6(15), pp. 62-65, 2015.
[7] A. M. Atteya and A. H. Madian, "A Hybrid Chaos-AES Encryption Algorithm and Its Implementation Based on FPGA," *IEEE*, vol/issue: 7(14), pp. 214-220, 2014.
[8] A. Nadjia and A. Mohamed, "AES IP for Hybrid Cryptosystem RSA-AES," *2015 12th International Multi-Conference on Systems, Signals & Devices on IEEE*, vol/issue: 7(15), pp. 1-6, 2015.
[9] X. H. Wu and X. J. Ming, "Research of the Database Encryption Technique Based on Hybrid Cryptography," *2010 International Symposium on Computational Intelligence and Design on IEEE*, vol/issue: 3(10), pp. 86-71, 2010.

BIOGRAPHIES OF AUTHORS

John Mark B. Espalmado was born in Nabua, Camarines Sur, Philippines in 1996. He is currently studying BS Electronics and Communications Engineering in Cavte State University, Indang, Cavite, Philippines. He is a grantee of the RA 7687 DOST-SEI academic scholarship since 2012. He is a Board-of-Director of the Jr. Institute of Electronics and Communications Engineers of the Philippines – Cavite chapter since July 2016. He is currently the president of the Cavite State University League of Adamant DOST (Department of Science and Technology) Scholars. His research interests include cryptosystem development, and control and monitoring systems.

Edwin R. Arboleda was born in Indang, Philippines in 1979. He received the B.S. degree in Electronics and Communications Engineering in Polytechnique University of the Philippines, Sta. Mesa, Manila, and Meng degree in De La Salle University, Manila in 1995 and 2008 respectively. Since 2001, he has been a member of the faculty of Department of Computer and Electronics Engineering in Cavite State University in Indang, Cavite, Philippines where he is currently an assistant professor. His research interests include near infrared spectroscopy, fuzzy logic, robotics, mechatronics, network security and artificial neural networks.