International Journal of Security and Its Applications
Vol.8, No.2 (2014), pp.353-362
http://dx.doi.org/10.14257/ijsia.2014.8.2.36

Design of the New Efficient Decripting Machine, that can be Applied to the Wireless Sensor Network, Based on the GF(2^n) Field Theory and the Modified RSA Algorithm

Hyeong-Keon An*

**School of Engineering, Tong Myung University, Pusan, Korea.*
hkan@tu.ac.kr

Abstract

Here we show the new modified RSA algorithm that can be applied to the WSN. Plaintext is converted into ciphertext by means of an encryption engine (again, generally a computer program) whose operation is fixed and determinate (the encryption method) but which functions in practice in a way dependent on a piece of information (the encryption key) which has a major effect on the output of the encryption process. In this process the encryption key and the decryption key may or may not be the same. When they are the cryptosystem is called a "symmetric key" system; when they are not it is called an "asymmetric key" system. The most widely-known instance of a symmetric cryptosystem is DES (the so-called "Data Encryption Standard"). In this paper, we apply the famous Reed Solomon code to the implementation of the asymmetrical cryptosystem based on the modified RAS algorithm which can be applied to the WSN.

Key Words*: Reed Solomon code, Symmetrical, Unsymmetrical, RSA (Rivest, Adi Shamir, Leonard Adleman), WSN (Wireless Sensor Network)*

1. Introduction

Encryption methods can be *SYMMETRIC* in which encryption and decryption keys are the same, or *ASYMMETRIC* (aka 'Public Key') in which encryption and decryption keys differ. 'Public Key' methods must be asymmetric, to the extent that the decryption key CANNOT be easily derived from the encryption key. Symmetric keys, however, usually encrypt more efficiently, so they lend themselves to encrypting large amounts of data. Asymmetric encryption is often limited to ONLY encrypting a symmetric key and other information that is needed in order to decrypt a data stream, and the remainder of the encrypted data uses the symmetric key method for performance reasons [1]. This does not in any way diminish the security nor the ability to use a public key to encrypt the data, since the symmetric key method is likely to be even MORE secure than the asymmetric method.. Fortunately, the simplest of all of the symmetric key 'stream cipher' methods is the TRANSLATION TABLE (or 'S table'), which should easily meet the performance requirements of even the most performance-intensive application that requires data to be encrypted. In a translation table, each 'chunk' of data (usually 1 byte) is used as an offset within one or more arrays, and the resulting 'translated' value is then written into the output stream. Similar to using a translation table, 'data repositioning' lends itself to use by a computer, but takes considerably more time to accomplish [2].

Public key encryption uses two keys - one to encrypt, and one to decrypt. The sender asks the receiver for the encryption key, encrypts the message, and sends the encrypted message to the receiver. Only the receiver can then decrypt the message - even the sender cannot read the encrypted message. When Bob wants to share a secret with Alice using public key encryption, he first asks Alice for her public key. Next, Bob uses Alice's public key to encrypt the message. In public key encryption, only Alice's private key can unlock the message encrypted with her public key. Bob sends his message to Alice. Alice uses

ISSN: 1738-9976 IJSIA
Copyright ⓒ 2014 SERSC

her private key to decrypt Bob's message. The things that make public key encryption work is that Alice very closely guards her private key and freely distributes her public key. She knows that it will unlock any message encrypted with her public key [3].

In this paper we introduce new efficient cryptosystem based on the Reed Solomon coding and modified RSA algorithm. Under this scenario, we formulate the symmetrical and unsymmetrical implementation of endecripting machine and give the optimal design using the Galois subfield theory. A simulated example is presented to show how to apply the optimal design. Finally,the paper is concluded with some remarks and future study.

2. Symmetrical Cryptosystem based on the Reed Solomon Coding

2.1. Introduction

Symmetric-key algorithms[1] are a class of algorithms for cryptography that use the same cryptographic keys for both encryption of plaintext and decryption of ciphertext. The keys may be identical or there may be a simple transformation to go between the two keys. The keys, in practice, represent a shared secret between two or more parties that can be used to maintain a private information link [2]. This requirement that both parties have access to the secret key is one of the main drawbacks of symmetric key encryption, in comparison to public-key encryption [3].

2.2. Encription and Decription

In a symmetriccryptosystem, you've got two parties who want to communicate, and they've got a *shared secret*. That shared secret is the key to the cryptosystem. Put in pseudo-mathematical syntax, the symmetric cryptosystem is a pair of functions, En and De, such that given a secret key S:

- Ciphertext = En(S, Plaintext)
- Plaintext = De(S, Ciphertext)

We call it symmetric because both the sender and the receiver need the *same* information. Both of them can encrypt messages using the key; both can decrypt.

If we use, α and $\alpha^2 \in GF(2^n)$ as the secret Key for our symmetrical cryptosystem, we just add (EXOR) α to the odd rows, and add (EXOR) α^2 to the even rows of the pain text matrix to encrypt it at the transmitter side and at the receiver side, we also add (EXOR) α to the odd rows, and add (EXOR) α^2 to the even rows of the cipher text matrix to decript it [4].

Figure 1. Symmetrical Crypto system, P, S, C stand for Plain text, Secret Key, Cypher Text Respectively

Copyright ⓒ 2014 SERSC

Let's apply Galois Field Reed Solomon coding to the crypto system. IF

$$S = \begin{pmatrix} \alpha \\ \alpha^2 \end{pmatrix}$$

$$P = \begin{pmatrix} \alpha^2 & 1 & \alpha \cdots \\ \alpha & \alpha^3 & 0 \cdots \\ \alpha^4 & 1 & \alpha \cdots \end{pmatrix}$$

(1)

Then Ciphertext = En(S, Plaintext) = P ⊕S, so

$$C = \begin{pmatrix} \alpha^2+\alpha & 1+\alpha & \alpha+\alpha \cdots \\ \alpha+\alpha^2 & \alpha^3+\alpha^2 & 0+\alpha^2 \cdots \\ \alpha^4+\alpha & 1+\alpha & \alpha+\alpha \cdots \end{pmatrix}$$

⋯(2)

Also Plaintext = De(S, Ciphertext) = C ⊕S , so

$$P = \begin{pmatrix} \alpha^2+\alpha+\alpha & 1+\alpha+\alpha & \alpha+\alpha+\alpha \cdots \\ \alpha+\alpha^2+\alpha^2 & \alpha^3+\alpha^2+\alpha^2 & 0+\alpha^2+\alpha^2 \cdots \\ \alpha^4+\alpha+\alpha & 1+\alpha+\alpha & \alpha+\alpha+\alpha \cdots \end{pmatrix}$$

…(3)

So Plaintext is recovered by the same secret Key used for the encryption [4].

3. Asymmetrical Cryptosystem based on the Reed Solomon Coding and Modified RSA Algorithm

3.1. Asymmetrical Cryptosystem

Public-key cryptography refers to a cryptographic system requiring two separate keys, one of which is secret and one of which is public. Although different, the two parts of the key pair are mathematically linked. One key locks or encrypts the plaintext, and the other unlocks or decrypts the ciphertext. Neither key can perform both functions by itself. The public key may be published without compromising security, while the private key must not be revealed to anyone not authorized to read the messages.

Public-key cryptography uses asymmetric key algorithms and can also be referred to by the more generic term "asymmetric key cryptography"[5].

3.2. Asymmetrical Encription based on the Modified RSA Algorithm

The security comes from the computational difficulty of factoring large numbers. To be secure, very large numbers must be used for p and q.

① Key generation Generate two large prime numbers, p and q. To make the example easy to follow I am going to use small numbers, but this is not secure. To find random primes, we start at a random number and go up ascending odd numbers

Copyright ⓒ 2014 SERSC

until we find a prime. Lets have: *p = 7*, *q = 19*.

② Let *n = pq then* n = 7 * 19 = 133

③Let m = (p - 1)(q - 1)

So m = (7 - 1)(19 - 1) = 6 * 18 = 108

④choose a small number e coprime to *m*. Here e coprime to *m* means that the largest number that can exactly divide both *e* and *m* (their greatest common divisor, or GCD) is 1. Euclid's algorithm is used to find the GCD of two numbers, but the details are omitted here.

e = 2 => GCD(e, 108) = 2 (no)
e = 3 => GCD(e, 108) = 3 (no)
e = 4 => GCD(e, 108) = 4 (no)
e = 5 => GCD(e, 108) = 1 (yes!)

⑤ up to now everythings are same as before, but from this tep, the RSA algorithm is modified as the equation in (4).

$$\textbf{Ciphertext} = \textbf{Plaintext} \bullet e^{k} = \alpha^{i}\,\alpha^{Ek} = \alpha^{i+kE} = \textbf{Enc}(P,e),$$
$$\textbf{Where } n = \alpha^{k},\quad e = \alpha^{E},\quad \alpha \in GF(2^{N}) \quad \cdots (4)$$

Where the message is Plaintext = α^{i}.

For the example, lets use the message "7".

$$\textbf{Plaintext} = 11100000 = \alpha^{198},\ \alpha \in GF(2^{8}) \cdots (5)$$

Here in equation (5), we see i= 198. Now use the equation (4) in the step ⑤ to get Cipher text as in the equation (6) .

$$\textbf{Ciphertext} = \alpha^{i+kE} \quad \ldots(6)$$

Here in equation (6), e= 5(decimal)= α^{50} =10100000(binary) as in the step ④, also n= 133 (decimal) = 10100001(binary)= α^{128}. So k value and E value of the equation (6) are 128 and 50. Therefore equation (6) gives rise to the result as in the equation (7).

$$\textbf{Ciphertext} = \alpha^{223} \in GF(2^{8}) \quad (\because \alpha^{255}=1)$$
$$= 10010000 \quad \ldots(7)$$

3.3. Asymmetrical Decription based on the Modified RSA Algorithm

From the Ciphertext,, we get the Plaintext as shown in the equation (8):

Plaintext = Ciphertext /(dn) = Decrip (C,d) , where n is α^{k} and comes from the step ② in section 3.2. ⋯ (8)

Now d is calculated by the step ⑥.

 Copyright ⓒ 2014 SERSC

⑥ d is calculated using the equation (9)

D = Minimum positive interger of E(k-1)+J(2^N-1) , where $\alpha \in GF(2^N)$, J is any integer and d= α^D … (9)

We continue our example to decript it to get the plaintext. From Ciphertext= α^{197} , the plaintext is found as shown in the equation (11). From the equation (9), D is calculated as shown in the equation (10).

D = **Minimum (+integer) of k(E-1)+(2^N-1)J**
= Minimum (+integer) of 6272-255J (J is any integer)
= 152 (J= 24). …(10)

So
d= α^{152} ,
Therefore
Plaintext = **Ciphertext /(dn) = α^{223} /(α^{k+E}) = $\alpha^{223}/\alpha^{280}$ = α^{223}/α^{25}**
=α^{198} $\in$GF (2^8) …(11)
This is correct plaintext transmitted from the sending terminal [6].

4. New Cyphering Algorithm based on the Modified RSA and Reed Solomon coding and Hardware Machine to implement the Algorithm

A. Algorithm

This is our algorithm. This will encrypt the plaintext at the transmitter and will decript the Cyphertext to get the plaintext at the receiver.

a. Plain text: **Plaintext = α^i , $\alpha \in GF(2^N)$**
b. Find the n and e values using the RSA(Rivest, Shamir and Adleman) algorithm and they are: **n = α^k , e = α^E**
c. Now encrypt the plain text: **Ciphertext =Enc(p,e) = p e^k = $\alpha^{(i+Ek)}$**
d. Find d using the modified RSA(Rivest, Shamir and Adleman) algorithm: **d= α^D, D= Minimum (+integer) of k(E-1)+(2^N-1)J (J is any interger).**
e. At the receiver Decription is processed: **Plaintext = Decrip (Cyphertext,d) =(p e^k /nd) = $\alpha^{(i+Ek-k-D)}$**

B. Hardware to Implement the Algorithm

For this algorithm, we need 3 kind of Hardware, Multiplier, Divider, Squarer to get the exponent term $\mathbf{e^k}$. In Figure 2 we shows the block diagram of the Hardware cct. (Processor system), which is consisted by 3 main execurion units, Multiplier, Divider, Squarer circuit.

Copyright ⓒ 2014 SERSC

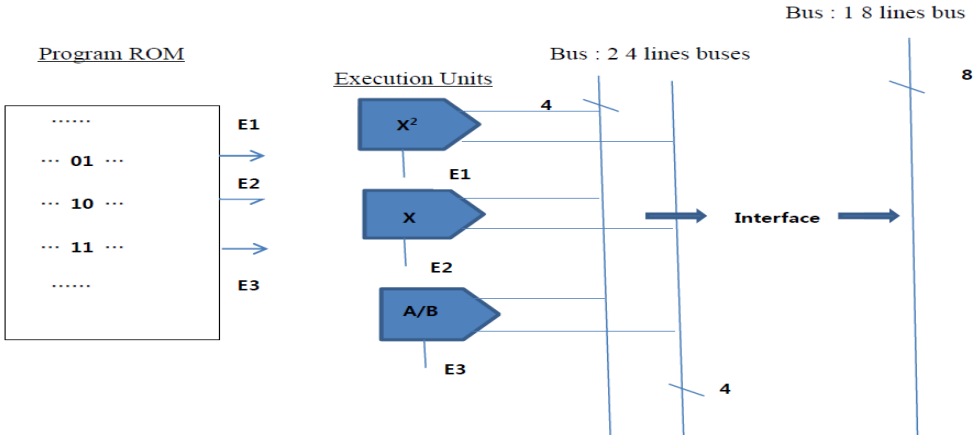

Figure 2. Processor Structure to Implement the New Crypto System based on GF(2^4) and GF(2^8)

In Figure 2 Multiplier, Divider is already described in [4]. Also the Squarer circuit is used for the computation of $\mathbf{e}^{\mathbf{k}}$, exponential calculation. For example to calculate α^5, we suuare the α, then square again then the result becomes α^4. The result is multiplied by α to get the final out α^5. In Figure 3 we shows the GF(2^8) square circuit using the subfield GF(2^4) multiplier. Equations 12 and 13 are the C0, C1 logical expressions of the multiplied results.

$$C = A^2 = (a0 + \beta\, a1)^2$$
$$= c0 + \beta\, c1 \quad , A, C \in GF(2^8) \qquad \cdots(12)$$
$$C0 = a0^2 + a1^2\,\gamma$$
$$C1 = a1^2,\ a1^2 = (z0+z2+z3,\ z3,\ z1+z3,\ z2+z3),\ a1 = (z0,z1,z2,z3)$$
Where $a0,a1,c0,c1 \in GF(2^4)$ and $z0,z1,z2,z3 \in GF(2)$
$$\cdots(13)$$

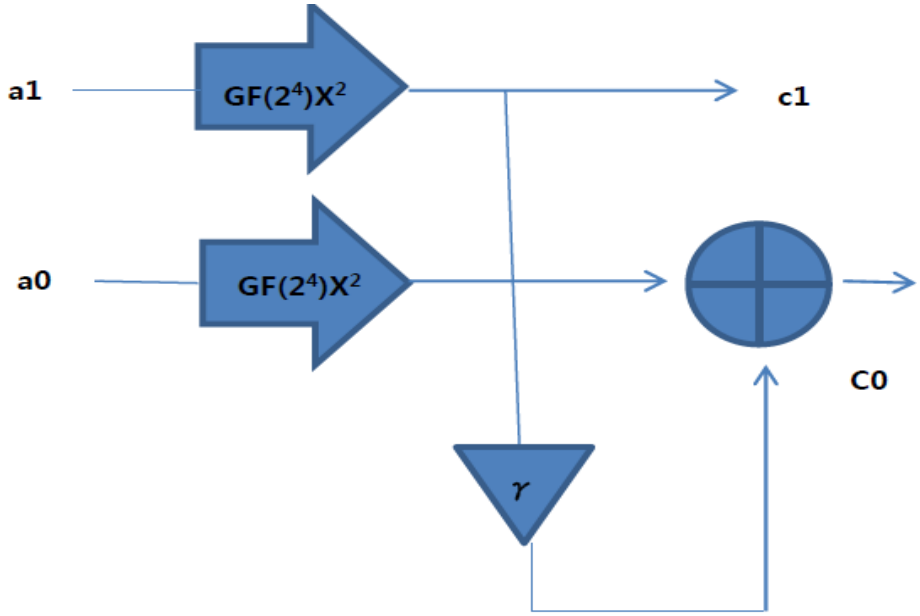

Figure 3. GF(2^8) Squarer Circuit based on GF(2^4) Squaring Circuit and γ Multiplier

Copyright ⓒ 2014 SERSC

5. Simulation of the Algorithm based on Modified RSA and Reed Solomon Coding

We now simulate the GF(2^8) squaring circuit and our new Cyphering machine using Altera VHDL software.

5.1. GF(2^8) Squaring Circuit Simulation

All the simulations are based on the equations (12) and (13) and the results are shown in the Figure 4.

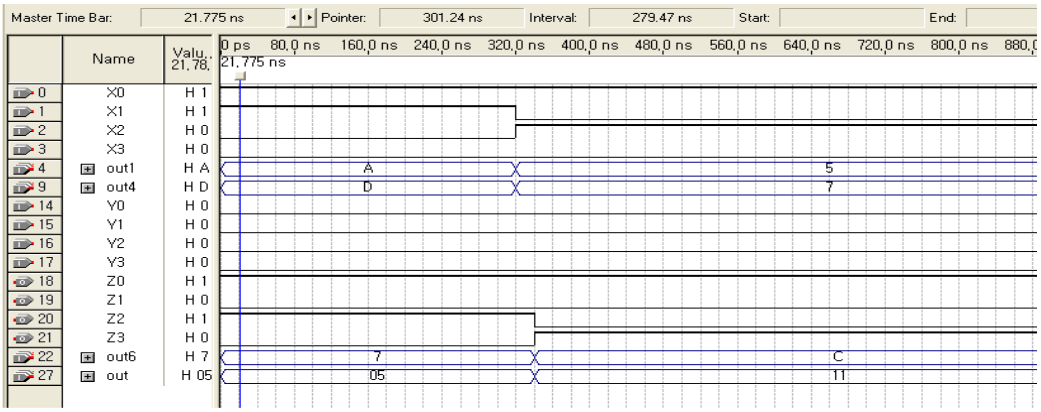

Figure 4. Squaring Circuit Input/Output Waveform

In Figure 4, we see inputs of α^{25}(03H), α^{50} (05H)$\in$GF(2^8) gives rise to ($\alpha^{10,}$, α^{11}), -> (A,D) and (α^9, α^7) $\in$GF(2^4) -> (5,7) also their squared outs are α^{50}(05H), α^{100}(11H) $\in$GF(2^8), and (α^9, α^7)-> (5,7) and , (α^4, α^{14}) -> (9,C) $\in$GF(2^4) .

5.2. Encripting and Decripting Simulation of our New Method

This is an extremely simple another example using numbers you can work out on a pocket calculator.

1. Select primes p=11, q=3.
2. n = pq = 11●3 = 33
   phi = (p-1)(q-1) = 10.2 = 20
3. Choose e=3
   Check gcd(e, p-1) = gcd(3, 10) = 1 (i.e. 3 and 10 have no common factors except 1),
   and check gcd(e, q-1) = gcd(3, 2) = 1
   therefore gcd(e, phi) = gcd(e, (p-1)(q-1)) = gcd(3, 20) = 1
4. D= Minimum (+integer) of E(k-1)+(2^N-1)J (J is any interger) . here D is calculated as follows

Using our new algorithm, Public key (n,e) = (10000100, 11000000)= (α^{138}, α^{25}), and Let's assume the Plaintext= 5 = (10100000)= α^{50}. From our new algorithm d= (D=252) using the equation 10.So from the equation (8) or (11), we get the plaintext= $\mathbf{p\ e^k\ /nd}$ = α^{50} α^{3450} / (α^{138} α^{252})= α^{50}. This is correct. Now the Fig.5 shows the simulation

Copyright ⓒ 2014 SERSC

result using the VHDL software. From the top waveforms represent the e(α^{25}), n(α^{138}), d (α^{252}), Encrip(Plaintext)= α^{185} , Decripted plaintext : p(α^{50}) sequentially[4].

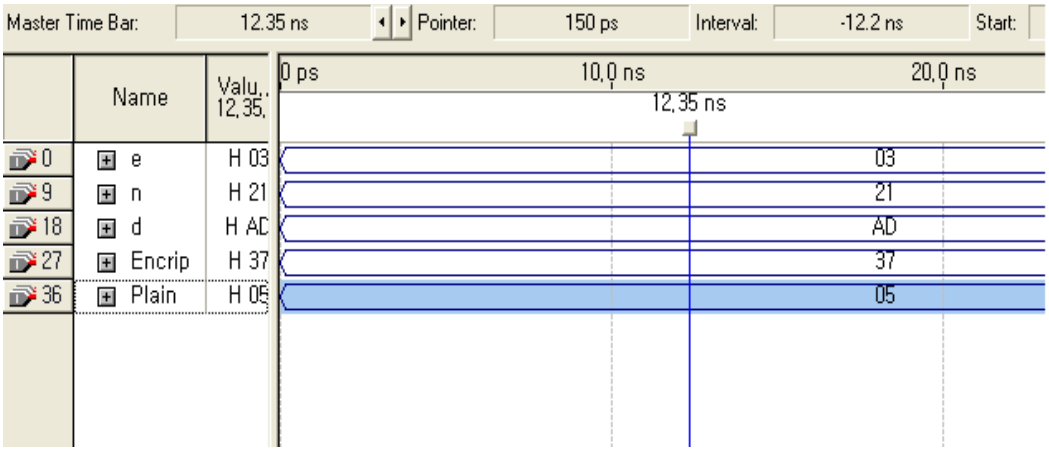

Figure 5. Outputwaveforms of the New Cryptosystem

6. Application of the Modified RSA Algorithm by the Digital Signature to the Wireless Sensor Network

6.1. Digital Signature

We here briefly summarize the Cryptosystem and the Digital signature system. In the Cryptosystem: Cyphertext C is made from encrypting the message M using the public key e as in the equation (14). In the receiver side M is recovered from C using the private key d as in equation (15).

$$C = Enc(M,e) \quad \cdots(14)$$
$$M = Dec(C,d) \quad \cdots(15)$$

In the digital signature system, Cyphertext C is made from signing the message M using the private key d as in the equation (16). In the receiver side M is verified and recovered from C using the public key e as in equation (17).

$$C = Dec(M, d) \quad \cdots(16)$$
$$M = Enc(C, e) \quad \cdots(17)$$

In this way, the message and the signature are sent to the receiver. The receiver receives the message and the signature and applies the verifying algorithm to the combination. If the result is true, the message is accepted; otherwise, it is rejected.

6.2. Applying the Digital Signature Process of the New Method to the WSN

We now apply the new method to the wireless sensor network by digitally signing the transmitted data. The wireless sensor network is shown in the Figure 6a [9]. The encriptor and decriptor are positioned as shown in the Figure 6b.

The purpose of the endecripting machine is to recognize which nodes, from the many sensing nodes at the transmitter, actually sent the received data. So if the decripting public key is e_i , and it certainly decripts the received data successfully, then we know that the decripted data was sent by the ith sensor from the many sensing nodes at the transmitted side. Otherwise we should use other public key to decript the data and definitely the data

 Copyright ⓒ 2014 SERSC

was sent by the other sensor node. Therefore our endecripting machine is functioning as the digital signature machine. The back ground algorithms are as follows. Here we use the public key for decripting key, and the private key for encrypting key so the process is the reverse of the normal asymmetrical cryptosystem which uses the private key for the decription and the public key for the encryption.

Encription is done using the private key "d", and if the plain text is "P", and the cypher text is "C". Enc ( ), Decrip( ) operator are as in the equations (4), (8):

$$C = \text{Decrip}(d,P) = \frac{P}{nd} \quad \cdots(17)$$

Now decription is done using the public key "e".

$$\text{Encrip}(e,C) = C \bullet (e)^k = \frac{P}{nd} e^k = P \quad \cdots(18)$$

So plain text is successfully recovered.

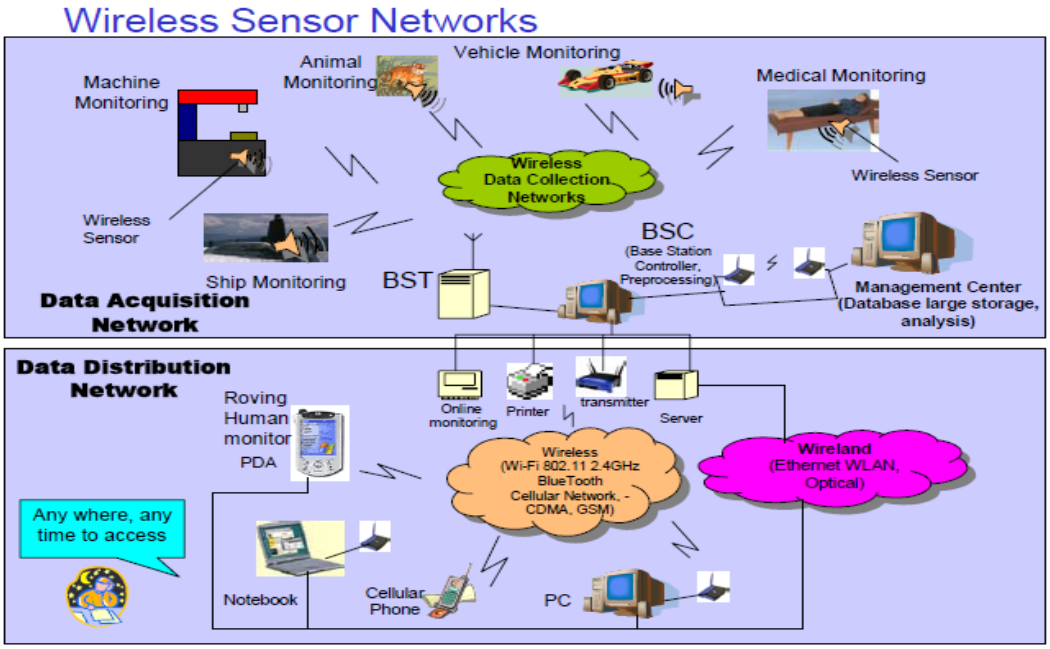

Figure 6. (a) Wireless Sensor Network [9]

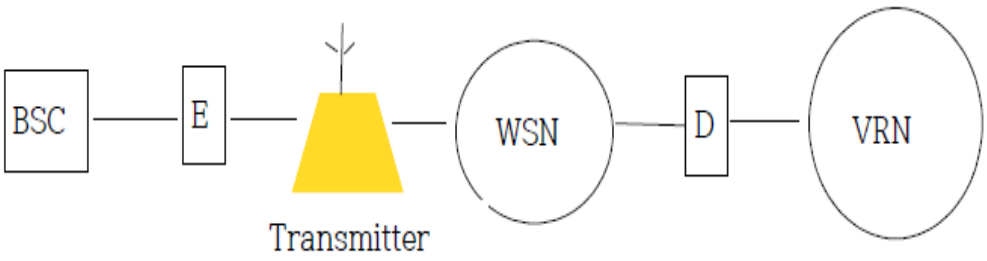

Figure 6. (b) E: Encriptor, D: Decriptor , WSN : Wireless Cellular Network, VRN : Various Receiving Nodes (PC, Cellular Phone, PDA etc)

Copyright ⓒ 2014 SERSC

7. Conclusion

When using symmetric algorithms, both parties share the same key for en- and decryption. To provide privacy, this key needs to be kept secret. Once somebody else gets to know the key, it is not safe anymore. Symmetric algorithms have the advantage of not consuming too much computing power. A few well-known examples are: DES, Triple-DES (3DES), IDEA, CAST5, BLOWFISH, TWOFISH. Asymmetric algorithms use pairs of keys. One is used for encryption and the other one for decryption. The decryption key is typically kept secretly, therefore called ``private key'' or ``secret key'', while the encryption key is spread to all who might want to send encrypted messages, therefore called ``public key''. Everybody having the public key is able to send encrypted messages to the owner of the secret key [6, 7].

In this paper we tried to construct the new symmetrical and asymmetrical cryptosystem based on the famous Reed Solomon coding and modified RSA algorithm. We find the new method is computationally more efficient and faster than the classical method. The new Machine can be applied to the Wireless Sensor network to protect the transmitted data. WSN has applications in a variety of fields such as environmental monitoring, military purposes and gathering sensing information in inhospitable locations [8, 9].

Acknowledgement

“This Research was supported by the Tongmyong University Research Grants 2012”.

References

[1] N. Ferguson and B. Schneier, “Practical Cryptography”, Wiley. ISBN 0-471-22357-3, **(2003)**.
[2] C. Paar and J. Pelzl, “Introduction to Public-Key Cryptography”, Springer, **(2009)**.
[3] J. Katz and Y. Lindell, “Introduction to Modern Cryptography”, CRC Press. ISBN 1-58488-551-3, **(2007)**.
[4] H.-K. An, “Fast and low cost GF(2^8) multiplier design based on double subfield transformation”, International Journal of Software Engineering and Its application(IJSEIA), **(2013)** July.
[5] A. O. EI-Rayis and X. Zhao, “Low power RS Codec Using Cell Based Reconfigurable Processor”, IEEE Transactions on Computers, **(2009)**, pp. 279-282.
[6] P. Pavel and G. Elyar, “Variable redundancy Reed-Solomon encoder”, USP 8176397, **(2012)** May.
[7] http://voices.yahoo.com/comparing-symmetric-asymmetric-key-encryption-6329400.html.
[8] Y. J. Zhao, R. Govindan and D. Estrin, “Residual energy scan for monitoring sensor networks”, Wireless Communications and Networking Conference, 2002 (WCNC2002), 2002 IEEE, vol. 1, **(2002)** March 17-21, pp. 356-362.
[9] F. L. Lewis, “Wireless Sensor Network”, Smart environments: Technologies, Protocols and Applications e.d. S. K. Das, John Wiley, New York, **(2004)**.

*Corresponding author: Dr. Hyeong-Keon An
Dept. of Information and Telecommunication Engineering
School of Engineering
Tong-Myung University,
Pusan, Korea.
E-mail: hkan@tu.ac.kr

 Copyright ⓒ 2014 SERSC