Journal of Korea Multimedia Society Vol. 11, No. 6, JUNE 2008(pp. 852-859)

A Dynamic Keyed Block Encryption Algorithm

Wei Jiang[†], Sungje Kim[††], Kyooseok Park[†††]

ABSTRACT

In this paper, we propose a dynamic keyed block encryption algorithm. Most existing encryption algorithms are designed such that the key is not changed. Therefore, they have a disadvantage that plaintext could be easily exposed by differential and linear cryptanalysis.

In the proposed algorithm, several key generators are designed, and a key generator is attached to the encryption procedure. After performing the encryption procedure, ciphertext and the initial key generating values are transferred to the receiver's key generator for decryption.

Through simulation, the proposed algorithm is verified to satisfy the requirements of real-time processing and proved to have a high strength. It can be applied to practical use.

Key words: Block encryption, Dynamic key, Round function

1. INTRODUCTION

With the rapid development of modern society, the security problem is becoming increasingly serious. In the fields of military, science, finance, and business, security is one of the most important aspects and many security algorithms were designed and implemented to protect these fields.

Typical encryption technologies have been focused on the security of data. However, due to the confusion and diffusion theory, various types of research are focused on authentication, data integrity and access control.

In the proposed algorithm, several key generators are designed, and a key generator is attached to the encryption procedure. When performing the encryption procedure, the initial values are transferred to the receiver's key generator for decryption.

The remainder of this paper is as follows. Section 2 presents the related studies; then, in section 3 our proposed block encryption algorithm and some backgrounds are presented, while section 4 shows the performance of the algorithm, and conclusions are presented in section5.

※ Corresponding Author : Kyooseok Park, Adress : (631-701) 449 Wolyong-Dong, Masan, Gyungnam, Korea, TEL : +82-55-249-2650, FAX : +82-55-248-2554, E-mail : kspark@kyungnam.ac.kr

Receipt date : Nov. 20, 2007, Approval date : Apr. 4, 2008

† Div. of Computer Engineering, Kyungnam University (E-mail : greatjiang0924@hanmail.net)

†† Div. of Computer Engineering, Kyungnam University (E-mail : sung0702@kyungnam.ac.kr)

††† Div. of Computer Engineering, Kyungnam University

※ This work was supported by Kyungnam University Foundation Grant, 2007

2. RELATED STUDIES

2.1 Basic concept of cryptography

Nowadays, in the field of cryptography, there are two types of encryption systems: one is the symmetric algorithm and the other is the asymmetric algorithm.

A Symmetric algorithm is also called as a secret -key algorithm or a single-key algorithm in which the encryption key and the decryption key are same. The algorithm requires that the sender and receiver agree on a key before they can securely communicate. The security of a symmetric algorithm rests in the key; divulging the key means that anyone could encrypt and decrypt messages.

Fig. 1 shows the Symmetric-key algorithm.

A Public-key algorithms is also called as an

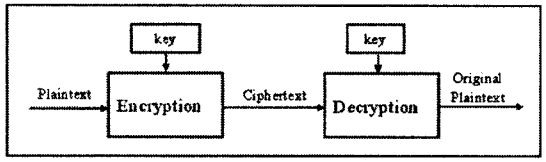


Fig. 1. Symmetric-key algorithm

asymmetric-key algorithm, it is designed such that the key used for encryption is different from that used for decryption. The decryption key cannot be calculated from the encryption key, and the encryption key can be made public. That is, a stranger can use the encryption key to encrypt a message; however, only a specific person with the corresponding decryption key can decrypt the message[1].

Fig. 2 shows the Public-key algorithm.

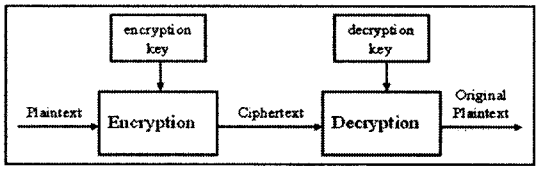


Fig. 2. Public-key algorithm

2.2 Block encryption algorithm

This kind of algorithm can be divided into two categories: one is the stream cipher, which operates on the plaintext one bit at a time; the other is the block cipher, which is a kind of private key system and plaintext is encrypted one block at a time.

(1) Data Encryption Standard (DES): It operates on 64-bit block of plaintext. After an initial permutation, the block is divided into two halves: a right half and a left half, each 32-bit long. Then there are 16 rounds of identical operations, called function F, in which the data are combined with the key. After the 16 rounds, the left and right halves are joined, and a final permutation ends the algorithm[1-8].

(2) Triple DES: In order to the resolve some vulnerabilities of DES, and to design one more powerful algorithm, triple-DES is proposed. This algorithm is designed to reinforce the security of DES. It is designed to use triple encryption with DES and combined with two keys or three keys. Triple DES with two keys is a relatively popular alternative to DES, but sometimes it may be more powerful to use three keys than two keys in the Triple-DES encryption algorithm[1,2,7].

(3) AES: It was found to be a well-adjusted SPN (substitution permutation network) structure that can be made up for the existing ones. AES was a renowned system among the well-known different types of cryptosystems. It is deemed to be a fast and simple algorithm on various platforms. The input to the encryption and decryption algorithm is a single 128-bit block, and the key length can be independently specified to be 128, 192, or 256 bits. The 128-bit block is copied into a state array, which is modified at each stage of encryption or decryption. After the final stage, the state is copied to an output matrix[1-8].

(4) SEED: This is a 128-bit block cipher designed by KISA (Korea Information Security Agency). The algorithm is an industrial association standard of Korea. It has the Feistel structure with 16 rounds and is strong against differential cryptanalysis and linear cryptanalysis.

(5) Camellia: This is an encryption algorithm related to the Feistel structure. It has been applied favorably by several organizations. This algorithm uses an 18-round Feistel structure for 128-bit keys, and a 24-round Feistel structure for 192 and 256-bit keys, with additional input/output whitenings and logical functions referred to as the FL function and the FL^{-1} function inserted into every 6 rounds.

2.3 Strength

The performance of a block encryption algorithm is measured by the degree of stability.

The simplest attack is exhaustive cryptanalysis. In this attack, every possible key will be processed.

With the block ciphertexts encrypted using an n-bit key, and we can decrypt it by $2^{n-1}/2$ evaluations on an average.

In a block cipher algorithm, there are two types of most important attack methods, one is the differential cryptanalysis (DC) [9], and the other one is the Linear Cryptanalysis(LC) [10].

The basic idea of DC is to find two plaintexts x and x'; if they satisfy $x'=x\oplus a$, the result will be $E(x')=E(x)\oplus b$. That is, if we find the event $E(x')=E(x)\oplus b$, we will obtain the value of the corresponding ciphertext from the randomly chosen plaintext $x'=x\oplus a$, it is a kind of chosen-plaintext attack.

LC is designed to make use of the probability of $P_r[E(x)\in H_2 \mid x\in H_1]$ under two hyperplane conditions H_1 and H_2. That is, LC is a kind of known-plaintext attack that is used to evaluate the correlation between the event $x\in H_1$ and the event $E(x)\in H_2$ [10].

3. THE PROPOSED ALGORITHM

The proposed algorithm in this paper is focused on the symmetric key system as mentioned previously.

3.1 Backgrounds of algorithm

1) F-function: It is one of the important elements in the block encryption algorithm. It is also designed to reduce the relationship between the input and the output.

2) S-Box: An S-Box is actually a basic component of a symmetric key algorithm. The strength of various Feistel structure algorithms and specifically their resistance to differential and linear cryptanalysis depend directly on the S-Boxes.

S-Boxes a set of nonlinear lookup tables in encryption and decryption procedure; they are used to obscure the relationship between plaintext and ciphertext. An S-Box is a mapping of m-bit inputs to n-bit outputs, which is referred to as an m*n-bit S-Box.

3.2 Encryption algorithm

The proposed dynamic keyed encryption algorithm has several random key generators, and a key generator is attached to the encryption procedure. After performing the encryption procedure, ciphertext and the initial key generating values are transferred to receiver's key generator for decryption.

Fig. 3 shows structure of the proposed algorithm.

Notations of the operation are defined as follows:

kw : Sub-keys for whitening

kr : Sub-keys used in round functions

⊕ : Bitwise exclusive-OR operation

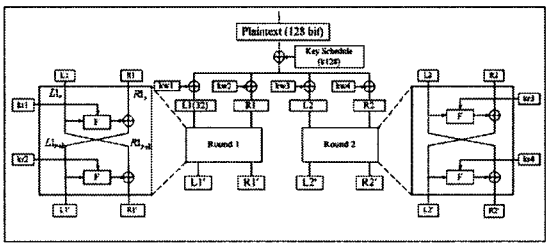


Fig. 3. Structure of the proposed algorithm

The entire process of the proposed encryption algorithm can be divided into three phases. In the beginning of the encryption, a 128-bit plaintext block operates XOR with 128-bit key (Fig. 3). In the second step, the XORed block is divided into four 32-bit blocks: then these blocks are XORed with the subkeys. In the third step, every two 32-bit blocks are attached to a round function, and the subkeys (kr1, kr2, kr3, kr4) used in the round functions are also generated from the key generator.

The operations in round functions are as follows: the divided 32-bit blocks L1 and R1 are attached to the left round function as $L1_r$ and $R1_r$, respectively. In the first step, the left part block $L1_r$ and the key kr_1 are attached the F-function for

operation, then, the result of this operation is XOR with the right part of the 32-bit block Rl_r. The second step is the same as the first step. The operations in the left round functions are performed as follows:

Fig. 4 shows The Round function.

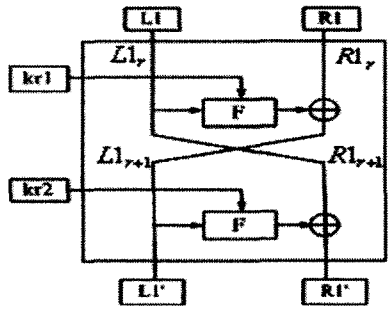


Fig. 4. Round function

3.3 Key schedule

Key generators are used to generate subkeys dynamically for each round. The subkeys, S-Box and F function are operated to enhance the difficulty in decryption.

1) Sub-key generator : The generated subkeys are changed dynamically to reduce the relationship between the input blocks and the output blocks.

First, a 128-bit key is divided into four 32-bit blocks. We embed two random number generators R1 and R2 for the key schedule, then these blocks are XORed with two 32-bit blocks respectively.

The random number generators are as follows:

R1= ((input value × a) +c) mod m
R2= ((input value × b) +c) mod m

The 128-bit key is showed in Fig. 5, it is divided into four 32-bit blocks X1, X2, X3, and X4. In the first round, X1 and X3 are attached to the random number generators R1 and R2 respectively; then, the two output values are XORed with X2 and X4, respectively. After the first round, they are permutated in the intermediate step; with the result of this operation, the key schedule steps into the second round to make the same operation as in the first step. In order to make the random key more confidential and steady, we add four F-functions into the key schedule. All these operations are operated as follows:

a1 = X1;
a2 = R1(X1) ⊕ X2;
a3 = X3;
a4 = R2(X3) ⊕ X4;
X1 ′ = F(a2);
X2 ′ = F(R1(a2) ⊕ a3);
X3 ′ = F(a4);
X4 ′ = F(R2(a4) ⊕ a1);

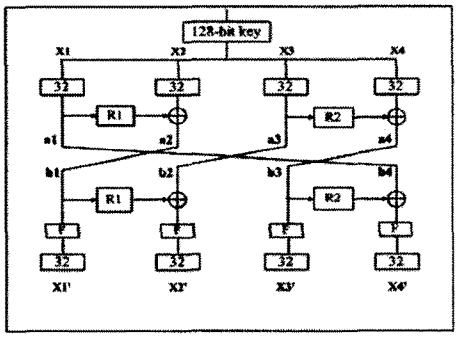


Fig. 5. Details of key schedule

2) Round key : As showed in Fig. 3, three 128-bit round keys are generated by the key schedule. The first 128-bit round key is transferred to the execution procedure, and it is transferred to the key schedule again as a feedback key to generate the second 128-bit round key; the third 128-bit round key is also generated by the same method as the second step.

The first 128 bit round key is XORed with the plaintext; the second 128-bit round key is supplied for sub-keys kw1, kw2, kw3, and kw4 to encrypt with previous divided four parts. The third 128-bit round key is also divided into four parts (kr1, kr2, kr3, kr4), which will be used in the round functions.

3) Random number generator : We designed several random number generators that are defined

as follows:

$$X1 = ((a*x0) + c) \bmod m \quad (1)$$

$$X2 = ((x0*x0) + c) \bmod m \quad (2)$$

$$X3 = ((a*x0) + x0) \bmod m \quad (3)$$

$$\vdots \qquad \vdots$$

Fig. 6 shows the procedure of initial key generation.

```
public int SetRndNum(int X0, int num)
{
        int Rem = 256;
        int a = 175;
        int c = 11220;
        int Xn = 0;
        switch(num)
        {
                case 1:
                Xn = ((a*X0)+c) % Rem;
                break;
                case 2:
                Xn = ((X0*X0)+c) % Rem;
                break;
                case 3:
                Xn = ((a*X0)+X0) % Rem;
                break;
        }
        return Xn;
}
public byte[] KeyGenerate(int X, int num)
{
        int nRoof = 16;
        int Xn = X;
        byte[] Data = new byte[nRoof];
        for(int i=0; i<nRoof;i++)
        {
                Xn = SetRndNum(Xn, num);
                Data[i] = (byte)Xn;
        }
        return Data;
}
```

Fig. 6. Procedure of initial key generation

4) S-Box : Four S-Boxes are designed in an F-function to against the differential and linear cryptanalysis. The primitive polynomial $a = x^8 + x^6 + x^5 + 1$ [8] is used to generate the S-Box which is designed to have 8-bit input and 8-bit output.

Table 1 shows the S-Box1 embedded in an F-function.

Table 1. S-Box1

	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xA	0xB	0xC	0xD	0xE	0xF
0x0	A3	D7	09	83	F8	48	F6	F4	B3	21	15	78	99	B1	AF	F9
0x1	E7	2D	4D	8A	CE	4C	CA	2E	52	95	D9	1E	4E	38	44	28
0x2	0A	DF	02	A0	17	F1	60	68	12	B7	7A	C3	E9	FA	3D	53
0x3	96	84	6B	BA	F2	63	9A	19	7C	AE	E5	F5	F7	16	6A	A2
0x4	39	B6	7B	0F	C1	93	81	1B	EE	B4	1A	EA	D0	91	2F	B8
0x5	55	B9	DA	85	3F	41	BF	E0	5A	58	80	5F	66	0B	D8	90
0x6	35	D5	C0	A7	33	06	65	69	45	00	94	56	6D	98	9B	76
0x7	97	FC	B2	C2	B0	FE	DB	20	E1	EB	D6	E4	DD	47	4A	1D
0x8	42	ED	9E	6E	49	3C	CD	43	27	D2	07	D4	DE	C7	67	18
0x9	89	CB	30	1F	8D	C6	8F	AA	C8	74	DC	C9	5D	5C	31	A4
0xA	70	88	61	2C	9F	0D	2B	87	50	82	54	64	29	7D	03	40
0xB	34	4B	1C	73	D1	C4	FD	3B	CC	FB	7F	AB	E6	3E	5B	A5
0xC	AD	04	23	9C	14	51	22	F0	29	79	71	7E	FF	8C	0E	E2
0xD	0C	EF	BC	72	75	6F	37	A1	EC	D3	8E	62	8B	86	10	E8
0xE	08	77	11	BE	92	4F	24	C5	32	36	9D	EF	F3	A6	BB	AC
0xF	5E	6C	A9	13	57	25	B5	E3	BD	A8	3A	01	05	59	2A	46

5) F-function : Function F provides the elements of confusion in a Feistel cipher and is nonlinear. Its structure is showed in figure 7. X1, X2, X3, X4 are 8-bit values; kf1, kf2, kf3, and kf4 are four subkeys generated from the key schedule. Four S-Boxes are designed in an F-function to resist the differential and linear cryptanalysis.

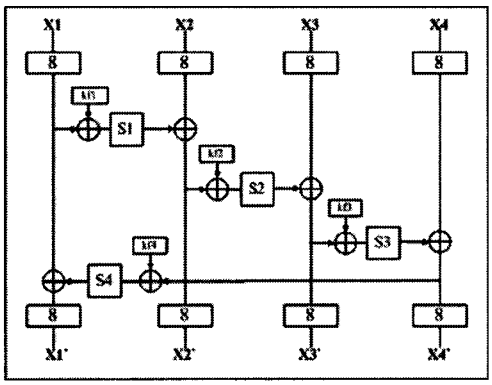


Fig. 7. Structure of the F function

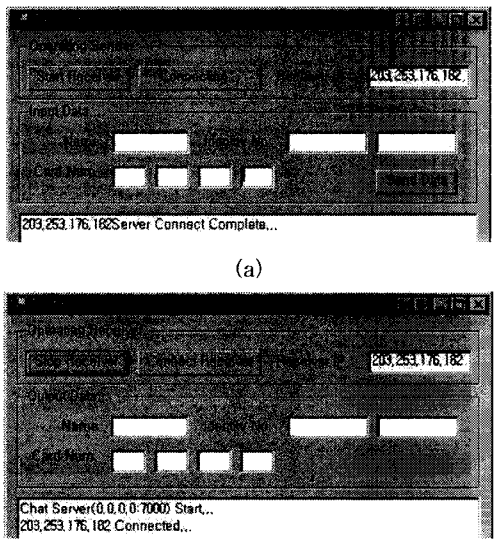
3.4 Decryption

In the proposed algorithm, decryption can be operated in the same way as the encryption by reversing the order of processing. The initial values used for encryption are attached to receiver's key generator, and the subkeys generated from the key schedule are applied for decryption.

4. SIMULATION AND EVALUATION

4.1 Simulation

Fig. 8(a, b) shows the connecting of the sender and receiver.



(a)

(b)

Fig. 8. Sender and receiver

Fig. 9(a, b) shows the encrypted messages, decrypted messages and the average processing time. The system condition is as follows:

The data for encryption

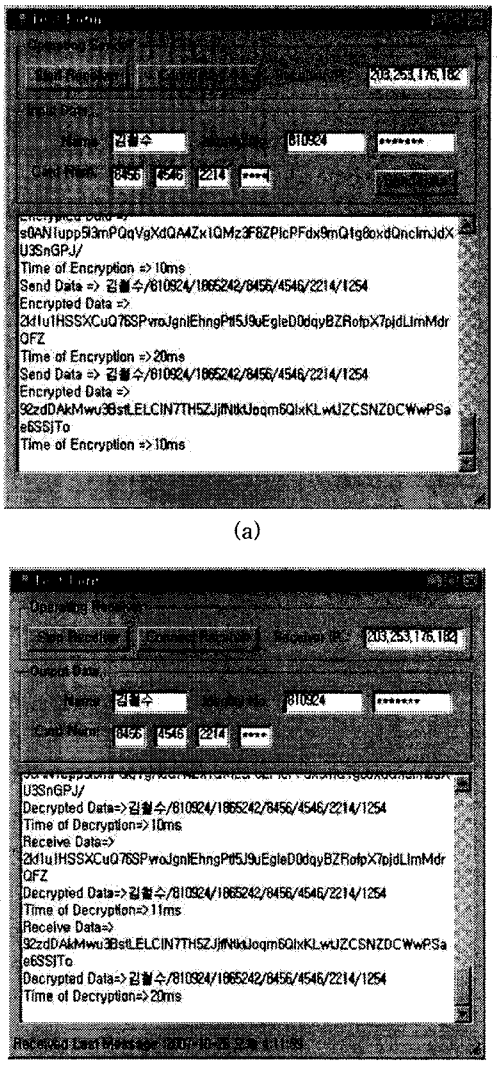
- name: 6 characters;
- card No: 16 characters;
- ID: 13 characters

Total 35 characters are operated as plaintext, so the total length of the plaintext is 280 bit.

However, in the proposed algorithm, since the length of the plaintext is limited to 128 bits; the proposed algorithm can encrypt the input messages in a 128-bit block respectively, then the encrypted block is got in a buffer and thus achieve the encryption of 280 bits.

4.2 Evaluation

- *Complexity of the proposed algorithm*



(a)

(b)

Fig. 9. Operations of the client and server

(1) 128-bit plaintext are XORed with 128-bit keys, the complexity degree is n.

(2) The four divided n/4-bit are XORed with kw1~kw4, the complexity degree is n/4.

(3) In the key schedule, four divided blocks (n/4-bit) are XORed four times, so the complexity degree is: 4(n/4) = n. And in formula R1 and R2, the complexity degree in each of them is n/2 respectively. Therefore, the complexity degree of the key schedule is: n + 4(n/2) = 3n.

(4) In an F-function, every n/16-bit is XORed 8 times, the complexity degree is: 8(n/16)=n/2.

Hence, the complexity of the proposed algorithm T(n) is:

$$T(n) = n + n/2 + n/4 + 3n = 4.75n$$

$T(n) < O(5n)$.

• *Processing speed*

We compared the processing speed of the proposed algorithm with several existing algorithms in the same environment.

The system environment is as follows:

- System: Pentium III 996MHz
- RAM: 512 MB
- OS: Windows 2003 Enterprise Server
- S/W: C# Visual Studio .Net 2003

• *Strength*

To test the strength of the proposed algorithm, we analyze a key in every nanosecond by using a high-performance computer. The average times required to find the correct key is $2^{127}/2$

The strength of the proposed algorithm is calculated as follows:

$$\frac{2^{127}/2 \times 1/10^9 \times 3}{3600 \times 24 \times 365} \approx 8.09E+21$$

Table 2 shows that proposed algorithm has shorter processing time than DES & SEED [8], and its complexity is not very high. The strength of the proposed algorithm is 8.09E+21, which is higher than DES & SEED.

Table 2. Comparison of the proposed algorithm With several earlier algorithms(8).

Index	Algorithm name	Key size	Encryption speed(sec)	Required time(year)
1	AES (Rijindeal)	128, 196, 256	0.012	3.2E+70
2	SEED	128	0.016	2.7E+21
3	DES	64(128)	0.016	2.50E+9
4	Proposed algorithm	128	0.015	8.09E+21

5. CONCLUSION

In the proposed algorithm, several random key generators are designed, and a key generator is attached randomly to the encryption procedure; the initial values are transferred to the receiver's key generator for decryption.

The simulation results indicate that, the proposed algorithm requires a shorter processing time than DES & SEED, and its complexity is not very high. The strength of the proposed algorithm is 8.09E+21, which is higher than DES & SEED, and it should be applied to various applications for data protection and security.

The proposed algorithm also has some disadvantages. In the future study, more research will be required to design a more efficient cryptosystem.

REFERENCES

[1] Bruce Schneier, *Applied Cryptography*, Wiley, USA, 1995.

[2] William Stalling, *Cryptography and Network Security*, Prentice Hall, New Jersey. USA, 2002.

[3] Douglas R. Stinson, *Cryptography: Theory and Practice*, CRC Press, Boca Raton, 1995.

[4] Niels Ferguson and Bruce Schneier, *Practical cryptography*, Wiley, Indianapolis, Indiana, 2003.

[5] H.X. Mel and Doris Baker, *Cryptography Decrypted*, Addison-Wesley, San Francisco, USA, 2001.

[6] Man Young Rhee, *Cryptography and secure communications*, MCGraw-Hill 1994.

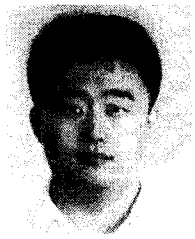
[7] Wenbo Mao, *Modern cryptography: theory and practice*, Prentice Hall 2003.

[8] Chang-Doo Lee and Kyoo-Seok Park, "Design and evaluation of a block encryption algorithm using dynamic-key mechanism," Future Generation Computer Systems, pp.

327-338, 2004.

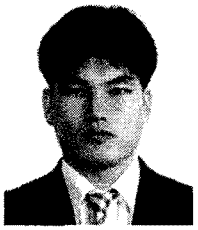
[9] E.Biham and A.Shamir, "Differential Crypt-analysis of DES-like cryptosystems," Advances in Cryptology - CRYPTO '90: Proceedings, pp. 2-21 1990.

[10] Matsui, "Linear Cryptanalysis Method for DES Cipher," EUROCRYPT'93 Vol.765 of Lecture Notes in Computer Science, Springer-verlag, pp. 386-397, 1993.



Wei Jiang

Wei Jiang is received the M.S. degree in the department of Computer Science at Kyungnam University in 2008. His research focuses on Distributed system, specifically applied to internet computing, Security system and Multimedia system.



SungJe Kim

SungJe Kim is a master course student in the department of Computer Science at Kyungnam University. His research focuses on Distributed system, specifically applied to internet computing, Security system and Multimedia system.

Park, Kyoo Seok

Park, Kyoo Seok(PH.D) is a professor in the department of Computer engineering at Kyungnam University, Korea. He is the honorary president of Korea Multimedia society now. His research focuses on Distributed system, specifically applied to internet computing, Security system and Multimedia system. MS and PhD in Computer science from the ChungAng University, Korea.