공 학 석 사 학 위 논 문

스프레드 랜덤 인터리버 방법에 기반을 둔 효율적인 DES 알고리즘

2012년 8월

부 경 대 학 교 대 학 원

정 보 통 신 공 학 과

황 재 영

공 학 석 사 학 위 논 문

스프레드 랜덤 인터리버 방법에 기반을 둔 효율적인 DES 알고리즘

지도교수 정 연 호

이 논문을 공학석사 학위 논문으로 제출함.

2012년 8월

부 경 대 학 교 대 학 원

정 보 통 신 공 학 과

황 재 영

황재영의 공학석사 학위논문을 인준함.

2012년 8월 24일

주 심 공학박사 김 성 운 (인)

위 원 공학박사 류 지 열 (인)

위 원 공학박사 정 연 호 (인)

목 차

그림 목차

표 목차

Efficient DES Algorithm based on Spread Random Interleaver Method

Jae-young Hwang

Department of Information and Communications Engineering,
The Graduate School, Pukyong National University

Abstract

Due to the fact that the initial and final permutation tables are opened to users, the conventional DES algorithm has some difficulty in its application in the personal cloud computing environment to provide security. In this thesis, we propose an advanced DES algorithm(I-DES algorithm) that is based on the spread random interleaver method.

The main idea of the proposed algorithm is to employ a permutation table generated by a spread random interleaver instead of the fixed permutation table to improve privacy further. Our simulation results reveal that the proposed I-DES algorithm is superior to the 3DES and legacy DES algorithms in the aspects of encryption speed and security guarantee. The proposed algorithm is also applicable to variable-length plain data block adapting the CBC(Cipher Block Chaining) method in the environment of the personal cloud computing.

제1장 서 론

최근 인터넷의 활용의 증가와 응용 프로그램 관련 콘텐츠 및 데이터 대용량화에 따라 정보의 저장과 처리에 있어 응용 서비스와 관계된 인터넷 데이터 센터(IDC : Internet Data Center)의 중요성이 점차 증대되고 있다. 그래서 이들을 활용하는 일반 사용자들은 다양한 컴퓨터 자원을 효율적이고 또한 경제적으로 사용하기를 원하는데, 이러한 것의 대안으로 개인기반 클라우드 컴퓨팅(Personal Cloud Computing) 기술이 주목을 받고 있다[1].

일반적으로 IDC 센터를 활용하는 개인기반 클라우드 컴퓨팅 환경에서 원격의 컴퓨터 자원을 이용하더라도 이들의 전달과정에서 여러 가지 위협으로부터 보호하기 위한 정보보안 문제가 크게 대두되고 있다.

해당 정보나 데이터의 전달과정에서 발생하는 보안에 대한 위협을 막기 위해 전통적으로 해당 전달 메시지에 암호화 및 복호화 알고리즘으로 DES(Data Encryption Standard)가 활용되었다[2]. 그리고 DES 알고리즘을 보안하기위해 3DES(Triple DES) 및 AES(Advanced Encryption Standard) 알고리즘 등이 개발되어 사용되어왔다[3].

현대 암호 알고리즘 중 대표적인 대칭 암호화 알고리즘 중의 하나인 DES 알고리즘의 경우 암호화키 크기가 56비트(패리티 비트 제외)로 전수 조사를 할 경우 2^{56}개 키를 조사함으로써 구할 수 있게 된다. 그러나 현재의 기술로 병렬처리가 가능한 컴퓨터를 이용할 경우 단시간에 모든 경우의 수를 검색할 수 있는 단점을 가진다. 다른 한편 이를 보완하는 3DES와 AES 알고리즘들을 개인기반 클라우드 컴퓨팅에서 활용에 있어서 자체의 복잡성에 기인하여 계산에 있어 다소 시간이 많이 걸리는 단

점을 가지고 있다[4][5].

본 논문에서는 개인기반 클라우드 컴퓨팅 환경에서 적용하기위해 다소 복잡하고 시간이 많이 소요되는 3DES 혹은 AES 알고리즘을 대신하여 기존의 DES 암호화 알고리즘을 개선하여 적용하는 방법을 제안한다. 제안된 방법의 핵심은 DES 알고리즘에 있어 안전성 측면에서 의미를 가지지 못하는 초기치환 및 최종치환을 위한 고정된 치환 테이블(substitution table) 활용을 스프레드 랜덤 인터리버(spread random interleaver) 방법[6]을 사용한 동적인 치환박스 활용으로 보안성을 높이는 개념이다. 결과적으로 이를 활용한 최소한의 변경으로 전달 데이터에 대한 기밀성과 안정성을 확보하고 빠른 속도로 암호화 및 복호화를 달성한다.

또한 개인기반 클라우드 컴퓨팅 환경에서 다양한 크기의 전달 메시지를 수용하기위해 블록암호 운영모드 중 하나인 CBC(Cipher Block Chaining) 블록 암호화 운영모드[7]을 적용하여 암호화되는 텍스트 길이가 가변적인 경우에도 적용 가능하도록 한다.

본 논문에서 제안하는 동적인 치환박스 생성에 활용된 스프레드 랜덤 인터리버 방법의 경우, 원래 무선통신 환경에서 적용되는 방법으로 잡음에 강하고 높은 기밀성을 제공하는 특징을 가지는데, 난수 생성을 위해 BBS(Blum Blum Shub) 의사난수 생성 방식을 제안하였다[8].

본 논문의 2장에서는 DES 및 3DES 알고리즘의 개념과 동작방법 및 문제점에 대해서 요약하고 이를 개인기반 클라우드 컴퓨팅 환경에서 적용할 때 발생하는 문제점을 기술한다. 그리고 3장에서는 스프레드 랜덤 인터리버를 활용한 개선된 스프레드 랜덤 인터리버 기반의 DES (I-DES : Interleaver-DES)를 제안하고, 4장에서 제안한 알고리즘의 안

전성 평가를 위해 먼저 보안성 제공에 대한 수치적인 분석과 인터넷 환경에서 테스트베드 구성 후 암호화 및 복호화 과정에서 요구되는 처리시간 분석도 수행 한다. 마지막으로 본 논문의 5장에서는 위에 제안된 여러 가지 기술들에 대한 평가와 활용성 및 향후 연구방향 등에 대해서 결론으로 정리한다.

제2장 DES 및 3DES 알고리즘

2.1 DES 알고리즘

1973년 미국 국립기술표준원(NIST : National Institute of Standards and Technology)[9]에서는 안전한 정보 전송을 위해 암호화를 위한 대칭키 기반 암호화 알고리즘을 공모하였는데, 결과적으로 IBM에서 제안한 DES 알고리즘이 채택되었다. 이 알고리즘은 블록 암호화 방식을 사용하는데 전송 정보의 해당 64비트 평문 블록에 적용되어 동일한 길이의 암호문을 생성한다. 이때 동일한 길이의 암호화키가 암호화와 복호화에 동일하게 사용된다[10].

DES 알고리즘의 암호화 과정은 두 개의 치환(P-박스 : permutation box)과 16개의 Feistel[11] 라운드(Round)로 구성된다[12]. 두 개의 치환 박스는 라운드 함수가 시작되기 전과 후에 사용되며 각각 초기치환 및 최종치환이라고 한다. 이 방법에서 각 라운드는 암호화키를 이용하여 키 생성기에 의해 생성된 48비트 길이를 가지는 라운드 키를 사용한다.

그림 2.1에서는 DES 알고리즘의 구조를 나타내고 있다. 먼저 64비트 평문(64-bit plaintext)이 입력되고 56비트 크기의 암호화키(56-bit cipher key)가 사용된다. 여기서 암호화키는 최초에 64비트가 입력이 되나 패리티 비트(parity bit) 제거과정을 거쳐 실제로는 56비트가 사용된다. 그리고 입력된 암호화키는 라운드 키 생성기(Round-key generator)를 통하여 각 라운드에 사용될 라운드 키를 생성하며, 입력된 평문은 초기치환(Initial permutation)을 거쳐 라운드 함수를 수행하고 최종치환(Final permutation)을 통하여 암호화된 64비트 크기의 암호문(64-bit ciphertext)이 생성된다.

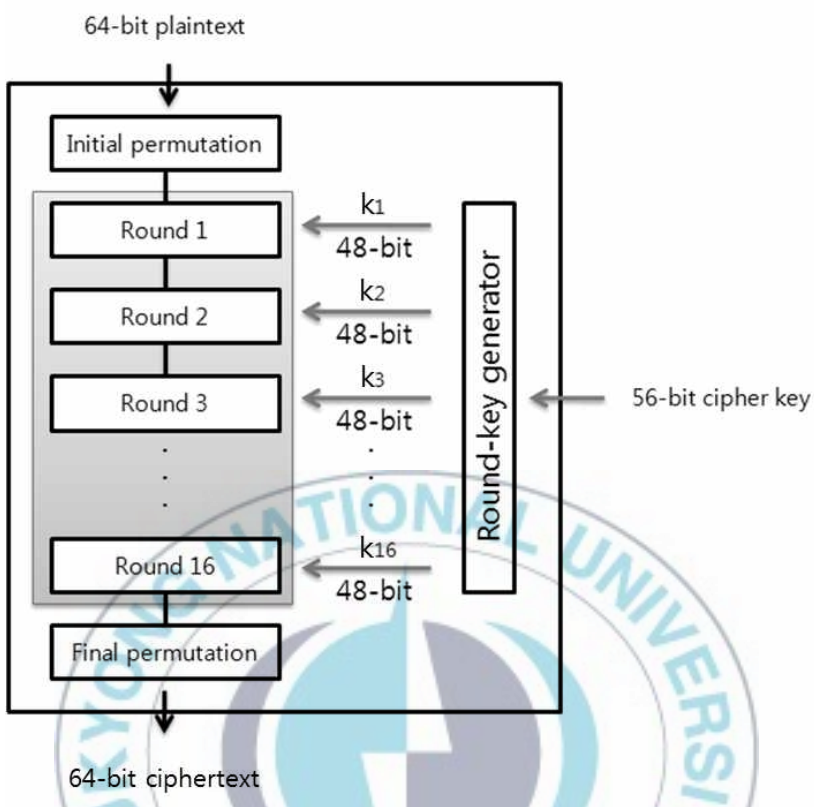

그림 2.1 DES 알고리즘 구조
Fig. 2.1 DES algorithm structure.

가. 초기치환 및 최종치환

초기치환과 최종치환이라 부르는 P-박스는 64비트 평문을 입력받아 그림 2.2에 나타낸 치환 테이블에서 정의된 위치로 재배열 과정을 수행한다. 즉 평문의 첫 번째 비트는 초기치환에서 58번째 비트 자리에 위치하게 된다. 여기서 DES 알고리즘의 문제점이 발생하는데, 즉 초기치환과 최종치환에서 사용되는 테이블 값은 키의 값과 관련 없는 고정된 하나의 함수 값이기 때문에 암호학적으로 별도의 기능 없이 섞어주는 역할로, 외부로부터의 공격에 의해 쉽게 해독할 수 있는 빌미를 제공한다.

Initial permutation								Final permutation							
58	50	42	34	26	18	10	02	40	08	48	16	56	24	64	32
60	52	44	36	28	20	12	04	39	07	47	15	55	23	63	31
62	54	46	38	30	22	14	06	38	06	46	14	54	22	62	30
64	56	48	40	32	24	16	08	37	05	45	13	53	21	61	29
57	49	41	33	25	17	09	01	36	04	44	12	52	20	60	28
59	51	43	35	27	19	11	03	35	03	43	11	51	19	59	27
61	53	45	37	29	21	13	05	34	02	42	10	50	18	58	26
63	55	47	39	31	23	15	07	33	01	41	09	49	17	57	25

그림 2.2 초기치환과 최종치환 테이블
Fig. 2.2 Tables for initial permutation and final permutation.

나. 라운드 함수

DES 알고리즘은 16번의 라운드 함수를 수행한다. 각 라운드에서는 각각의 라운드 함수의 이전 라운드 함수(첫 라운드에서는 초기치환 박스)의 출력 값 L_{I-1}과 R_{I-1}을 입력으로 받아, 다음 라운드(마지막 라운드에서는 최종치환 박스)에 입력으로 적용될 L_I와 R_I를 생성한다. 해당 알고리즘에서는 각 라운드 별 혼합기(Mixer)와 스와퍼(Swapper)를 사용하는데, 먼저 혼합기는 다음 소절에서 설명되는데 배타적 논리합 연산이기 때문에 역연산이 가능하다. 그리고 스와퍼는 단순히 암호화 과정에 있는 32비트 블록을 오른쪽과 왼쪽의 위치를 바꾸기 때문에 또한 역연산이 가능하다. 여기서 $f(R_{I-1}, k_I)$ 함수가 DES 알고리즘의 가장 중요한 부분인데 다음 절에서 상세히 기술한다. 그림 2.3은 위에서 설명한 라운드 함수의 구성을 보여준다.

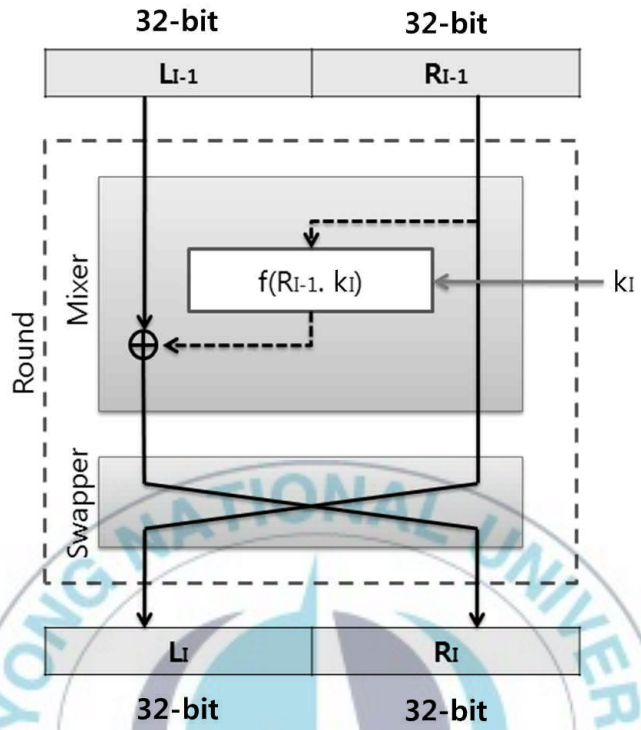

그림 2.3 DES 라운드 함수
Fig. 2.3 DES round function.

다. $f(R_{I-1}, k_I)$ 함수

$f(R_{I-1}, k_I)$ 함수는 DES 알고리즘에서 핵심이 되는 부분으로, 이전 라운드에서 받은 비트들 중 가장 오른쪽 32비트 블록과 라운드 키 값을 이용하여 연산을 수행한다. 그림 2.4와 같이 확장 P-박스(Expansion P-box), 배타적 논리합 연산(XOR operation), 8개의 S-박스(S-boxes) 그리고 단순 P-박스(Straight P-box)의 4개 부분으로 구성된다.

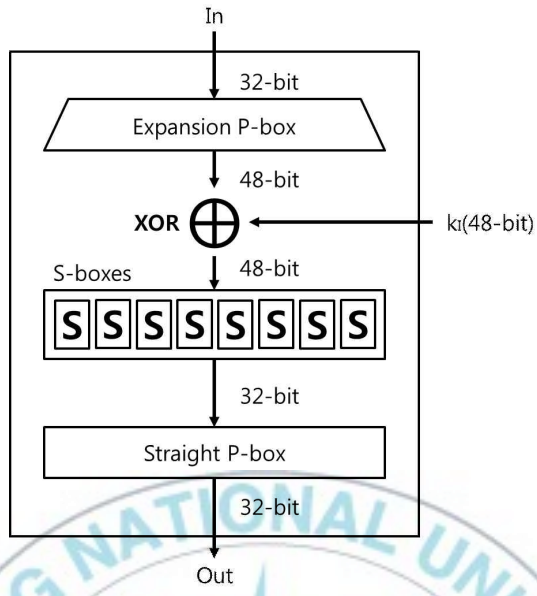

그림 2.4 $f(R_{I-1}, k_I)$ 함수
Fig. 2.4 $f(R_{I-1}, k_I)$ function.

(1) 확장 P-박스 기능

$f(R_{I-1}, k_I)$ 함수에 입력된 블록의 길이는 32비트이므로 라운드 키 48비트와 배타적 논리합 연산을 위하여 48비트로 확장이 필요하게 된다. 이를 위해 그림 2.5의 확장 P-박스 테이블에 의해 확장과정을 수행한다. 즉 그림의 음영부분은 새로이 더 첨가되는 비트들을 나타내고 각 수들은 입력된 해단 번째 비트를 재활용함을 의미한다. 나머지 입력된 32비트는 음영 부분의 사이에 해당 번호대로 위치시켜 48비트로 확장을 수행한다.

32	01	02	03	04	05
04	05	06	07	08	09
08	09	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	01

그림 2.5 확장 P-박스 테이블
Fig. 2.5 Table for expansion p-box.

(2) 배타적 논리합 연산 기능

확장 P-박스 확장을 수행한 48비트 블록을 라운드 키와 배타적 논리합 연산을 수행한다.

(3) S-박스 기능

S-박스는 배타적 논리합 연산을 마친 48비트 블록을 S-박스 연산을 통하여 데이터를 섞어주고, $f(R_{I-1}, k_I)$ 함수 입력 시의 길이와 같은 32비트로 축소시키는 과정을 수행한다.

아래 그림 2.6은 S-박스 기능을 설명한 것으로 먼저 입력으로 라운드 키와 배타적 논리합 연산을 수행한 48비트 블록을 8개의 6비트 단위의 블록들로 나누어 각각을 S-박스에 입력하여 연산을 수행한다.

그림에서처럼 각각의 S-박스 연산은 나눠진 블록의 6비트 중 맨 첫 번째 비트와 마지막 비트의 두 비트 값은 행을 나타내는 2진수 값을 의미하여 행을 결정하는데 사용되고, 그리고 가운데의 나머지 4비트들은 열의 2진수 값으로 활용되어 열을 결정하는데 쓰인다. 해당되는 행과 열의 S-박스에는 4비트 길이의 임의의 값을 가지고 있다.

즉 입력되는 값 6비트가 101010라고 한다면 양끝의 비트 1과 0은 10진수로 2행을, 가운데 4비트 0101은 10진수로 5열을 의미하게 된다. 그리고 테이블 2행 5열의 값이 7이라면, 출력되는 값은 7의 이진수 형태인 0111로 6비트 길이의 블록이 4비트 길이의 블록으로 축소된다.

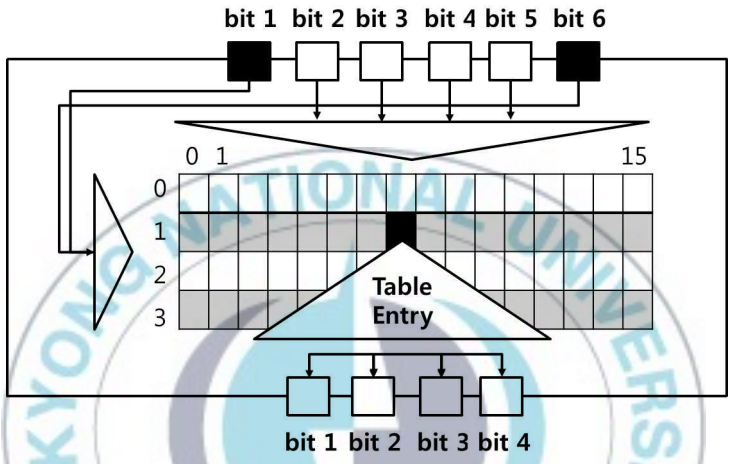

그림 2.6 S-박스 연산
Fig. 2.6 S-box operation.

(4) 단순 P-박스 기능

단순 P-박스는 S-박스 연산을 수행한 32비트 블록을 그림 2.7에 나타낸 테이블에서 정의된 위치로 재배열 과정을 수행한다.

16	07	20	21	29	12	28	17
01	15	23	26	05	18	31	10
02	08	24	14	32	27	03	09
19	13	30	06	22	11	04	25

그림 2.7 단순 P-박스 테이블
Fig. 2.7 Table for straight P-box.

라. DES 알고리즘의 암호화 및 복호화 과정

DES 알고리즘의 암호화 및 복호화 과정은 순서적으로 유사하게 이루어지나, 각 라운드에 사용되는 라운드 키의 적용 순서가 반대로 입력된다. 아래 그림 2.8은 같은 암호화키를 사용하는 암호화와 복호화과정을 설명한다.

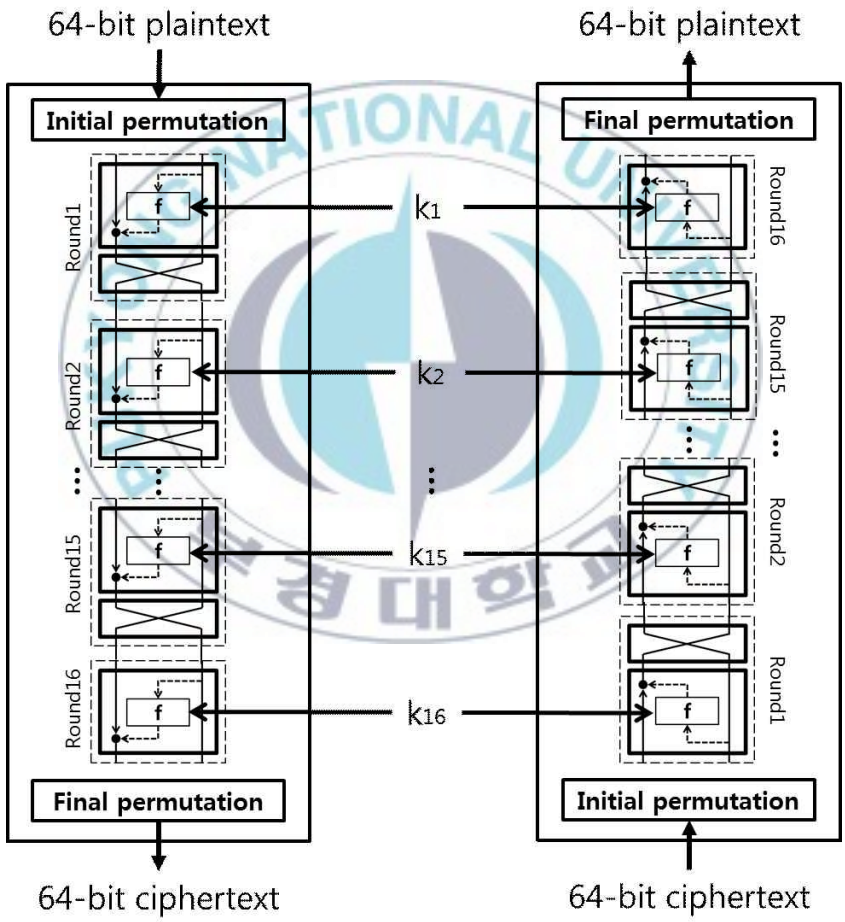

그림 2.8 DES 알고리즘의 암호화 및 복호화 과정
Fig. 2.8 Encryption and decryption procedure of DES algorithm.

2.2 3DES 알고리즘

DES 알고리즘은 키 길이가 56비트로 짧아 쉽게 해독할 수 있는 문제점을 가지고 있어 이를 보완하기위해 암호화와 복호화 과정에서 DES 알고리즘을 세 번 적용하는 3DES 알고리즘이 제안되었다[13].

3DES 알고리즘은 암호화키로 K_1, K_2 두 개를 사용한다. 먼저 K_1은 첫 번째와 세 번째 암호화 및 복호화 과정에서 사용되고, K_2는 두 번째 암호화 및 복호화 과정에서 사용된다.

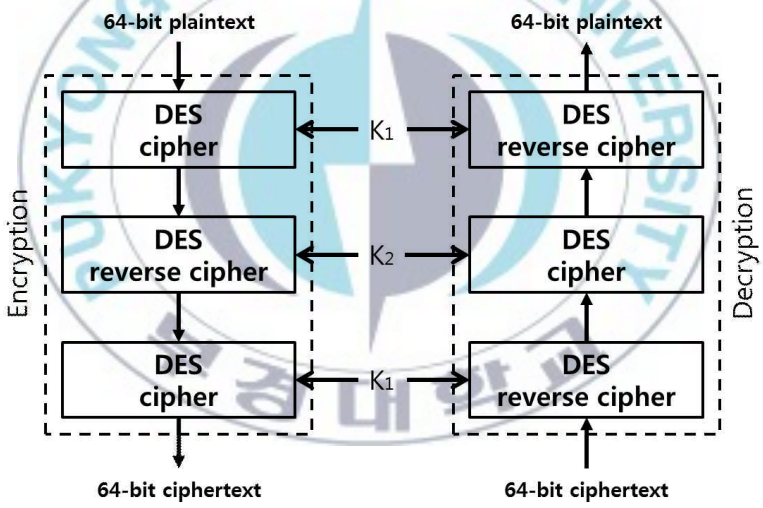

그림 2.9 두 개의 키를 갖는 3DES 알고리즘
Fig. 2.9 3DES algorithm with two keys.

그림 2.9에 설명된 것처럼 암호화 단계(Encryption)에서는 첫 번째 및 세 번째 과정(DES cipher)에서 DES 알고리즘 암호화 과정을 적용하고, 중간 과정(DES revserse cipher)은 복호화 과정을 적용한다.

또한 복호화 단계(Decryption)에서는 중간 과정에서 암호화 과정을 적

용하고, 첫 번째 및 세 번째 단계에서 복호화 과정을 적용한다.

결과적으로 두 개의 키를 사용하게 되므로 전체 키 길이는 112비트로 기존의 DES 알고리즘보다 두 배의 길이를 가져 보다 더 강력한 보안성을 제공한다.

다른 한편으로는 두 개의 키를 활용하는 3DES의 경우 키의 길이가 두 배로 늘어남으로써 전수조사 공격에 의해서는 강력하나 알려진 평문 공격(known plaintext attack)[14]에서는 취약한 점을 보여, 이를 보완하기 위해 세 개의 키를 암호화 및 복호화 과정에서 적용하여 보안성을 높이는 개선된 3DES 알고리즘이 제안되었다[4].

2.3 기존 알고리즘의 문제점

일반적으로 DES 알고리즘 적용에 있어 고정된 초기치환 및 최종치환 테이블의 사용은 보안성 측면에서 문제가 있고 암호화 및 복호화 연산 속도 면에서 20%정도 지연시키는 단점으로 지적되고 있다[15]. 또한 암호화 및 복호화 과정에서 사용되는 키의 크기가 56비트로 짧아, 전수조사를 할 경우, 병렬처리가 가능한 컴퓨터를 활용할 경우 단시간 내에 모든 경우의 수를 검색할 수 있어 키의 보안이 노출되는 단점을 가진다.

위에서 서술한 단점을 보완은 그 이후 표준화된 3DES와 AES 알고리즘을 적용할 수 있다. 그러나 3DES 알고리즘은 DES 알고리즘을 세 번 수행하여 더 많은 시간이 소요 되며, AES 알고리즘은 강력한 보안성을 제공하는 대신에 복잡한 수학적인 연산을 포함하므로 특정한 응용(예를 들어 개인기반 클라우드 컴퓨팅 환경과 같은)에서는 경우에 따라 적용에 있어 연산속도가 느리다는 단점이 있다[5].

3DES 알고리즘 이외에 기존 DES 알고리즘의 문제점을 해결하기 위해서 디지털 홀로그램[16]을 이용한 방법 등이 있다. 이 방법에서는 디지털 홀로그램을 사용하여 암호화 과정을 거친 후 그 결과를 대상으로 기존의 DES 알고리즘을 적용한다. 아래 그림 2.10은 해당 알고리즘에 대한 전체 과정을 설명한 것인데, 이 방법도 속도 측면에서 본 논문에서 적용하려는 응용 대상에 대해서는 효과적인 방법은 아니다.

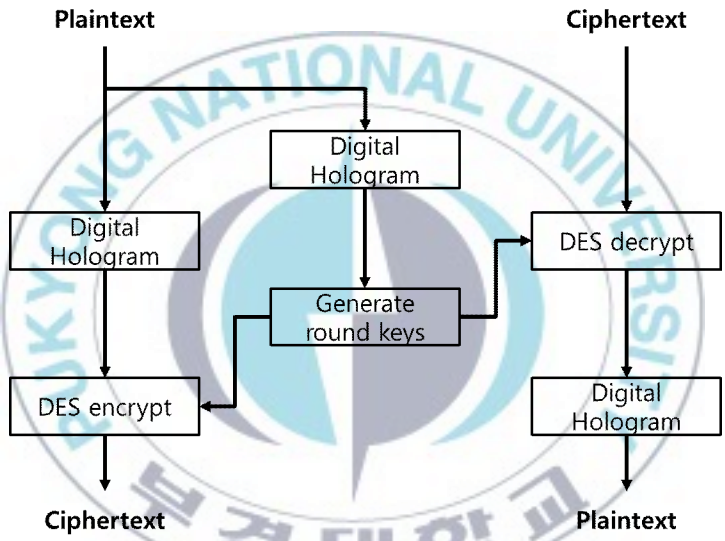

그림 2.10 디지털 홀로그램 방식의 DES 알고리즘의 암호화 및 복호화 과정

Fig. 2.10 Encryption and decryption procedure of DES algorithm based on digital hologram.

본 논문에서는 개인기반 클라우드 컴퓨팅 환경에서 적용하기위해 다소 복잡하고 시간이 많이 소요되는 3DES 혹은 AES 알고리즘을 대신하여 기존의 DES 암호화 알고리즘을 개선하여 적용하는 방법을 제안한다. 제안된 방법의 핵심은 DES 알고리즘에 있어 안전성 측면에서 의미를 가지지 못하는 초기치환 및 최종치환을 위한 고정된 치환 테이블 활용을

스프레드 랜덤 인터리버 방법을 사용한 동적인 치환박스 활용으로 보안성을 높이는 개념이다. 결과적으로 이를 활용한 최소한의 변경으로 전달 데이터에 대한 기밀성과 안정성을 확보하고 빠른 속도로 암호화 및 복호화를 달성한다.

제3장 스프레드 랜덤 인터리버 기반의 DES 알고리즘

3.1 스프레드 랜덤 인터리버

인터리버는 무선통신 환경에서 전송되는 데이터를 변조과정에서 시간이나 주파수 영역에서 데이터를 분산시키거나 데이터의 위치를 섞어 복조 이후에 집단적으로 발생하는 연집오류(burst error)를 산발오류(random error) 발생 형태로 개선하는 기능으로 오류정정을 용이하게 하는 방법이다[17].

본 논문에서는 이러한 인터리버 방식을 DES 암호화 알고리즘의 초기치환 및 최종치환 단계에서 적용하여 고정된 테이블 사용에 의한 보안성 문제점을 해결한다. 여기서 적용된 방식은 스프레드 랜덤 인터리버 방법으로 치환용 테이블 구성 단계에서 암호화 대상 데이터에 인터리빙을 수행할 때 인터리빙 된 비트들의 위치가 거리 S이상 떨어지도록 위치하도록 인터리버를 구성함으로써 무작위적인 방법에 대해 상대적으로 더 나은 성능을 확보하는 방법이다. 이 방법에서 사용되는 위치 값들의 생성은 다음 3.2소절에서 설명한 BBS 의사난수 생성기를 활용한다[18][19][20].

아래 그림 3.1은 제안한 스프레드 랜덤 인터리버의 동작과정이다. 먼저 암호화 대상 데이터의 변경된 위치 값을 생성(Generates position value)하는 과정에서 새로 생성된 비트를 위한 위치 값과 S개 이전동안 생성된 비트들의 위치 값들과 비교(Checks condition)하여 거리가 ±S 안에 있으면 현재의 위치 값을 버리고 다시 위치 값을 생성하고, 거리가 ±S이

상 떨어져 있다면 위치 값을 저장(Saves position value)한다. 위의 과정을 입력데이터의 비트수인 N개가 저장될 때까지 반복한다. 위에서 설명한 과정을 통해서 생성된 N개의 위치 값들을 이용하여 비트들을 재배열(Rearrange bits)하여 인터리빙 과정을 완료한다.

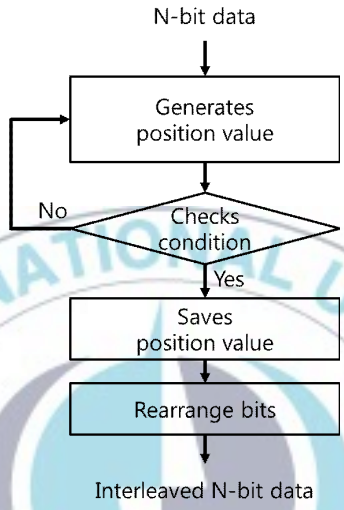

그림 3.1 스프레드 랜덤 인터리버 동작과정
Fig. 3.1 Operating procedure of spread random interleaver.

예를 들어 S가 5이고, 첫 번째 생성된 위치 값을 20이라 가정하고, 두 번째 생성된 위치 값이 16에서 24사이의 값일 경우 새로운 위치 값을 생성하고 그렇지 않을 경우 생성된 새로운 값을 저장하고 이 과정을 입력되는 비트 수 만큼 반복한다.

위치 값들의 간격 S가 증가하면 모든 N개의 위치 값을 얻는데 걸리는 시간이 증가하며, 반대로 S가 감소하면 비트들이 연집성이 높아져 성능이 떨어진다. 그러므로 S값의 결정은 일반적으로 다음 식 (1)을 만족하는 가장 큰 정수로 한다[18].

$$S < \sqrt{N/2} \qquad (1)$$

3.2 BBS 의사난수 생성기

본 논문에서 적용된 스프레드 랜덤 인터리버 방법에서는 무작위적 위치 값 생성을 위하여 BBS 의사난수 생성기를 사용하였다.

BBS 알고리즘은 그림 3.2에서와 같이 동작하며 0과 1로 된 비트열을 생성하며 구체적인 동작과정은 다음과 같다.

먼저 이방법의 시작을 위해 임의의 정수인 k에 대해 4k+3 형태를 갖는 두 개의 매우 큰 소수 p와 q를 정한다. 그런 후 아래 그림 3.2에 표시된 n값은 p와 q의 곱으로 구한다. 그다음 단계로 n과 서로 소(relative prime)인 임의의 수 r을 선택한다. 이 값들을 기초로 최초의 종자 값(seed)으로 쓰일 x_0의 값은 r^2을 n으로 나눈 나머지(r^2 mod n)값을 계산하여 구한다. 그 다음의 종자 값(x_{i+1})은 현재의 종자 값(x_i)의 제곱을 n으로 나눈 나머지(x_i^2 mod n)를 이용하여 구한다. 위와 같은 과정을 통하여 생성된 난수 정수열의 최하위 비트(Least Significant Bit)를 추출하여 그것을 의사난수 비트로 사용하게 된다. 본 논문에서는 무작위 적으로 선택된 새로운 위치 계산을 위해 8비트를 사용했는데 이는 아래 그림 3.2를 8번 시행한 비트열의 값이다. 결과적으로 해당 비트열의 값이 DES 입력문의 64개 비트의 새로운 위치와 매핑 되어 결정된다.

만약 p와 q값이 노출되어 있다면 가능한 x_0값을 이용하여 i번째 비트 값을 알아낼 수 있다. 따라서 생성기의 복잡 도는 n을 인수분해 하는 것을 의미한다. 그러나 n이 큰 수라면 이 수열은 예측 불가능하여 안전한 수열이 된다. 즉 n의 값이 크다면 이전의 모든 비트를 알고 있다고 하더라고 다음 비트 값을 추출할 수 없으며 각 비트가 0이 되거나 1이 될 확률은 약 50%로 n의 크기 조정에 따라 원하는 우수한 성능을 가질 수 있다[21].

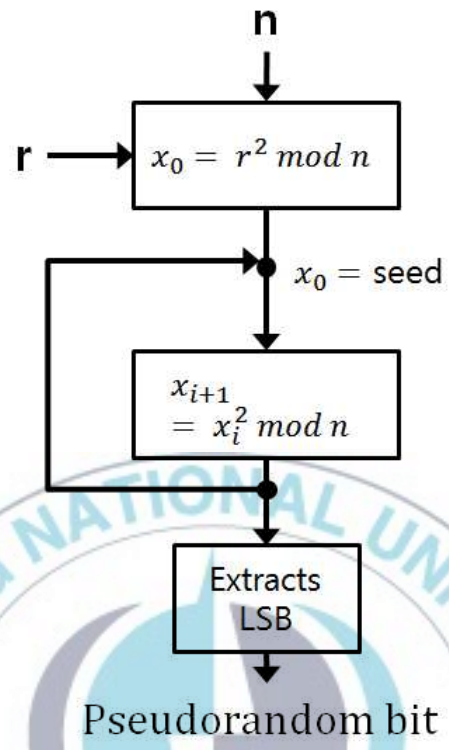

그림 3.2 BBS 의사난수 생성기 동작과정
Fig. 3.2 Operating procedure of BBS pseudorandom number generator.

아래 그림 3.3은 BBS 의사난수 생성기를 이용하여 생성된 8비트 길이의 값 10,000개의 분포도를 보여주고 있다. 히스토그램에서 보면 값들이 고르게 분포되어 있음을 확인할 수 있고, 그림에서는 분별할 수 없지만 시뮬레이션 값에 의하면 1과 0의 비율이 1:0.998로 약 1:1임을 확인하였다.

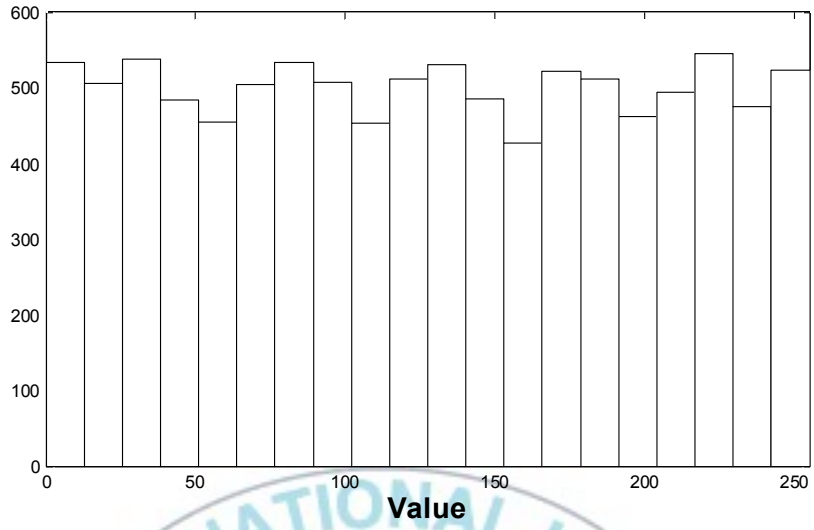

그림 3.3 BBS 의사난수 생성기 결과 값들의 히스토그램
Fig. 3.3 Histogram of BBS pseudorandom number.

3.3 스프레드 랜덤 인터리버 기반의 DES 알고리즘 (I-DES)

본 논문에서는 기존의 DES 알고리즘의 초기치환 및 최종치환에 사용되는 고정된 치환 테이블을 스프레드 랜덤 인터리버를 이용한 동적인 치환 테이블 구성 방법을 제시하였고, 이를 통하여 기존의 DES 알고리즘보다 높은 보안성을 확보할 수 있다. 결과적으로 개인기반 클라우드 컴퓨팅 환경에서 요구되는 빠른 수행시간에 부합되도록 특정 응용에 3DES 알고리즘이나 AES 알고리즘 보다 빠른 암호화 및 복호화 속도를 보일 수 있다.

기존의 DES 알고리즘의 경우 치환 테이블이 고정되어 있어 같은 평문에서 초기치환 과정을 수행할 경우 항상 같은 치환결과가 도출되어 DES 알고리즘에서 안정성 측면에서 의미를 부여하지 못한다. 그러나 제안한 암호화 알고리즘은 의사난수 생성기를 이용하여 초기치환과 최종치

환 테이블을 생성하므로 같은 평문을 치환할 경우 2^{64}가지의 경우의 수가 발생되고 이를 초기치환, 16개의 라운드 함수 그리고 최종치환을 거쳐 암호화 및 복호화가 진행이 된다.

공개된 치환 테이블의 경우 엔트로피(entropy)는 0비트이나, 식 (2)에서 보듯이 난수 생성기에 의해 생성된 테이블의 경우 6비트의 엔트로피를 가진다[22].

$$H(S) = -\log_2 \frac{1}{64} = 6\,(bits) \qquad (2)$$

그림 3.4는 제안한 I-DES의 암호화 및 복호화 과정을 나타낸다. 그림에서 보면 암호화가 시작되면 각 라운드에 사용될 라운드 키들을 암호화 키를 이용하여 생성(Generate round-keys)한다. 그 후 BBS 의사난수 생성기에 의해 생성된 난수를 이용한 스프레드 랜덤 인터리버를 이용하여 치환 테이블을 생성한다. 생성된 치환 테이블에 따라 64비트 길이의 평문(64-bit plaintext)을 초기치환(Initial permutation) 과정을 수행하고, 이후 16번의 라운드(Round)를 수행한다. 이를 다시 원래의 배열 순서로 치환하기 위하여 최종치환(Final permutation) 과정을 수행하게 되면, 최종적으로 암호화된 64비트 길이의 암호문(64-bit ciphertext)이 완성된다.

복호화 과정은 암호화 과정과 유사하나 각 라운드에서 사용되는 라운드 키가 역순으로 사용된다.

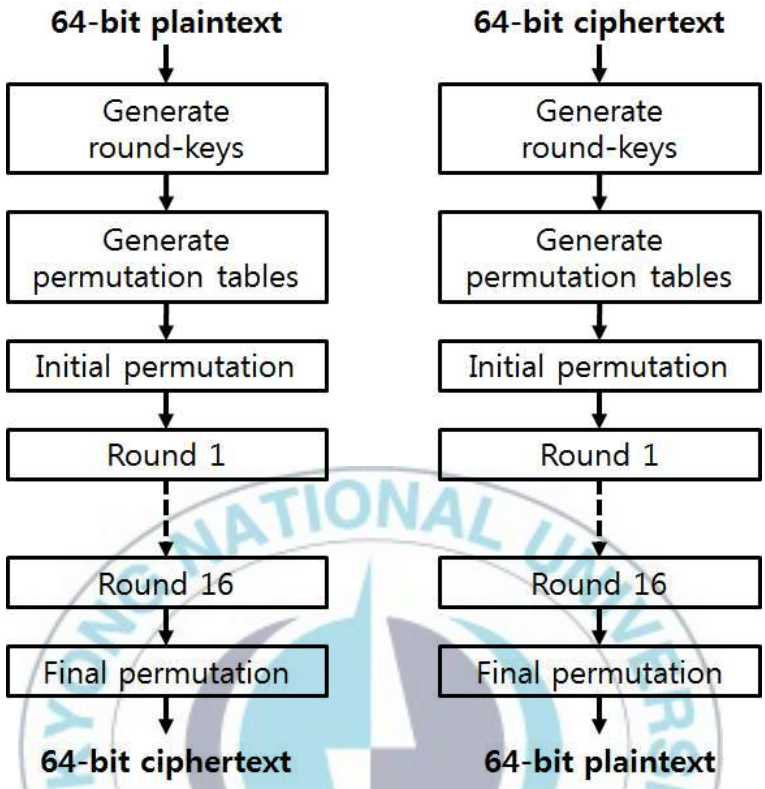

그림 3.4 I-DES 알고리즘의 암호화 및 복호화 과정
Fig. 3.4 Encryption and decryption procedure of I-DES algorithm.

일반적으로 대칭키 암호화 기법은 평문을 블록단위로 암호화하는 개념에 기반을 둔다. 본 논문에서는 DES 알고리즘의 경우 64비트 길이의 고정된 블록에 제한되어 사용되는 문제점을 해결하기 위해 블록 확장 암호화 방식의 하나인 CBC 알고리즘을 적용하여 가변길이를 가지는 데이터에 대해서도 암호화가 가능하도록 하였다[23]. 이를 나타낸 순서도는 그림 3.5와 같다[24].

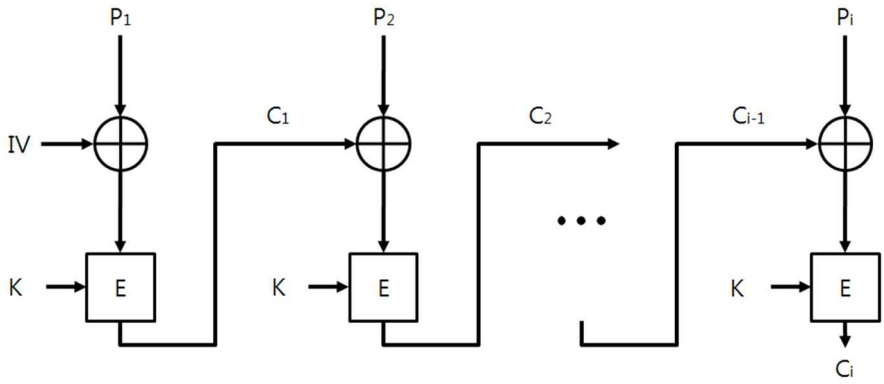

그림 3.5 CBC 알고리즘 블록 다이어그램
Fig. 3.5 Block diagram of CBC algorithm.

위의 다이어그램에서 사용된 P, IV, K, C의 정의는 식 (2)과 같다.

$$\begin{aligned} P &= \{P_1, \ldots, P_i\} \\ K &= \{K\} \\ C &= \{C_1, \ldots, C_i\} \\ IV &= \{IV\} \end{aligned} \quad (2)$$

P는 암호화될 평문 블록이며 64비트 단위로 블록화 되어있다. K는 송신자 및 수신자가 알고 있는 64비트 길이의 암호화키이며, C는 최종 암호화된 블록이고 각 블록에서 암호화 과정을 거쳐 생성된 암호블록들의 합이다. IV(Initial Vector)는 초기화 벡터이며, E는 암호화를 의미한다. 즉 64비트 단위의 여러 개 블록으로 구성된 평문들은 블록들로 나누어 P_1부터 P_i까지의 과정을 연계하여 수행한다.

CBC 개념을 적용한 I-DES 알고리즘은 각각의 블록을 암호화 하기위해 이전블록의 암호화된 블록을 이용한다. 그러나 최초의 평문 블록은 이전블록의 암호화된 블록이 존재하지 않기 때문에 암호화를 위해 IV라는 초기 값을 사용하게 된다. 본 연구에서 사용된 IV는 일반적인 정의에서 사용되는 '0' 값을 가지는 8바이트 길이의 고정 IV를 사용하였다.

CBC 개념을 적용한 I-DES에서는 입력되는 평문의 길이가 블록길이인 64의 배수가 아닐 경우에는 영채우기(zero-padding)를 통하여 64의 배수로 만든 후 DES 알고리즘의 암호화 과정을 수행하도록 하였다. 이 과정을 그림 3.6에서 보여준다.

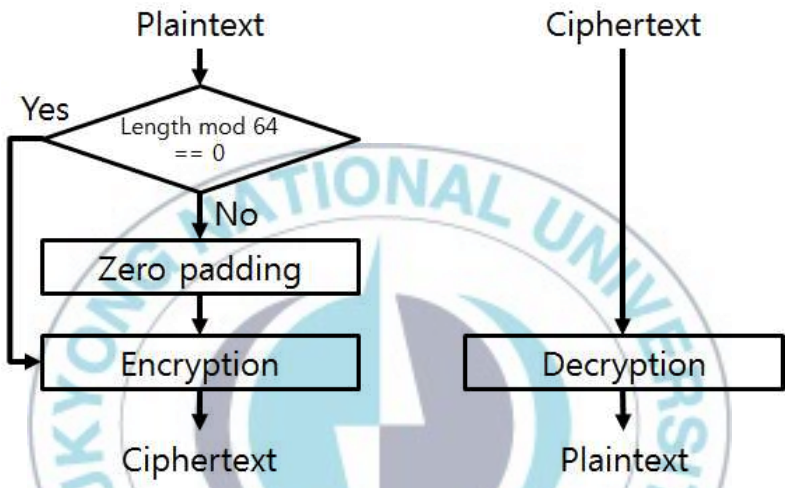

그림 3.6 CBC 알고리즘
Fig. 3.6 CBC algorithm.

제4장 성능 분석

4.1 테스트베드 구현

본 논문에서 제안한 CBC 개념을 활용한 I-DES 알고리즘을 적용하여 평문을 암호화 과정을 수행하고, 이를 다시 복호 화하여 원래의 평문을 얻을 수 있음을 확인하기위해 인터넷 환경에서 테스트베드를 구현하여 성능 분석을 수행하였다.

테스트베드는 아래 그림 4.1과 같이 서버와 클라이언트 환경으로 구성하였으며 그림에서 테스트베드용 PC가 두 가지 역할을 수행 한다.

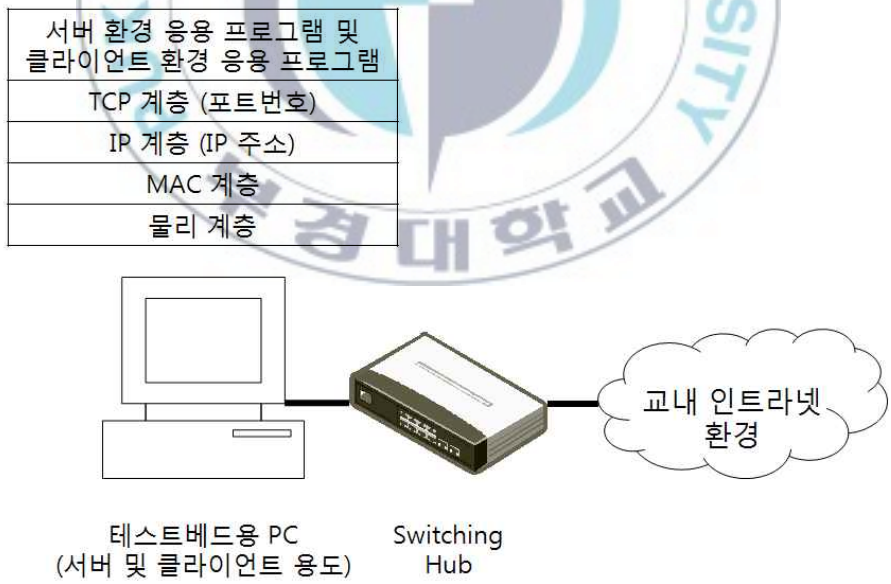

그림 4.1 테스트베드 구성도
Fig. 4.1 Test-bed configuration.

아래 그림 4.2는 전체 테스트베드 동작 과정을 나타낸 것이다. 먼저 서버 역할로 암호화 및 복호화 모듈 테스트를 위한 서버 응용 프로그램 시작용 소켓을 생성하고 IP와 포트번호를 바인딩 과정을 수행하여 클라이언트가 접속을 요청할 때까지 서버 환경 응용 프로그램이 기다리는 상태에 들어간다. 다음으로 클라이언트 환경의 해당 응용 프로그램이 서버에 접속 요청을 하기위해 연결 요청용 소켓을 생성하고 서버의 IP와 포트번호를 바인딩 한 후 서버 쪽으로 서버 접속 요청용 소켓을 전송한다. 이에 대해 서버 환경 응용 프로그램은 접속을 승인한 후 양쪽 서버와 클라이언트 응용 프로그램이 연결됨을 알리는 연결 요청 응답 소켓을 전송한다. 이 과정에서 사용된 IP주소는 루프백 기능을 사용하여 서버 및 클라이언트 PC가 같은 컴퓨터로 그 역할을 담당하게 된다. 그 다음으로 성능 분석에 사용될 암호화된 응용 메시지 송신과 복호화 및 재 암호화된 응용 메시지 송신으로 성능 분석용 메시지가 송수신된다.

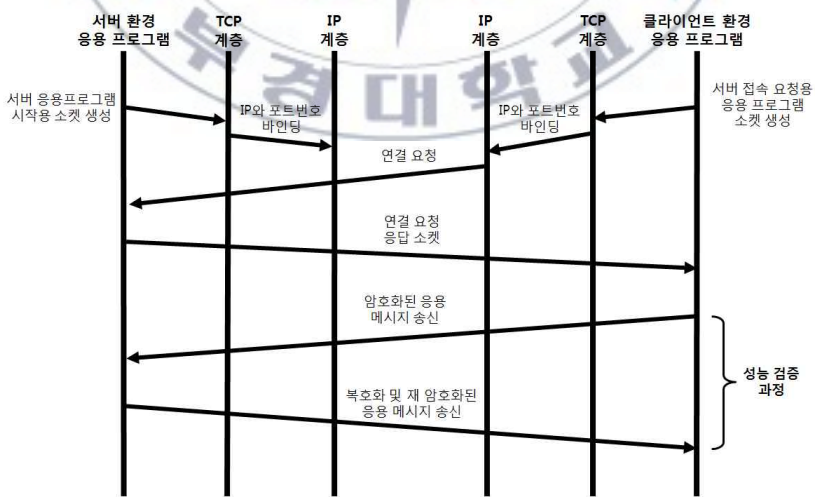

그림 4.2 성능분석을 위한 응용 소켓 송수신 과정
Fig. 4.2 Transmission and reception procedure of application socket for performance analysis.

아래 표 4.1은 그림 4.1의 테스트베드 구성도상에서 그림 4.2의 성능분석을 위한 응용 소켓 송수신 과정에 대한 테스트베드에서 사용되는 하드웨어 및 소프트웨어 환경을 요약하였다.

표 4.1 테스트베드 하드웨어 및 소프트웨어 환경
Table 4.1 Test-bed environments of H/W and S/W.

테스트베드용 PC 환경	H/W : Intel i5-2400 CPU 3.10GHz DDR 8GB Memory
	OS : Windows 7 Home Premium K (64-bit)
응용 프로그램 구성 환경	구현 S/W : Microsoft Visual Studio 2010 .Net C# Framework 4.0
	구현 통신 S/W : TCP/IP
I-DES 알고리즘에 적용할 파라미터	BBS 의사 난수 생성기 입력 p, q, r값
	스프레드 랜덤 인터리버에 의한 무작위 적으로 선택된 치환 테이블에서의 새로운 위치 계산

표 4.1의 테스트베드용 PC 환경과 응용 프로그램 구성 환경은 표의 내용대로 수행되나, 세 번째 항목인 I-DES 알고리즘에 적용할 파라미터는 그림 4.1의 성능 검증에 사용되는 파라미터들로 먼저 BBS 의사난수 생성기에서 사용되는 임의의 p와 q값을 입력받고 임의의 정수인 k에 대해 4k+3 형태로 입력받은 수보다 큰 수들 중에서 소수조건을 만족하는 가장 작은 수로 최종적인 p와 q으로 설정하였다. 그리고 r의 값은 p와 q의 곱인 n보다 큰 서로 소인 관계를 가지는 값들 중에서 가장 작은 값으로 r값을 정하였다.

여기서 사용되는 p와 q의 곱인 n은 크기가 클수록 예측 불가능한 안전한 수열이 생성되나, 암호화를 위해 활용한 프로그램 언어인 C#에서 사용되는 데이터 형인 부호 없는 정수형 데이터 표현 범위가 0에서

2^{32}-1까지 이므로 32비트 범위를 초과하지 않는 양의 정수로 설정하였다.

마지막으로 표 4.1의 세 번째 항목인 I-DES 알고리즘에 적용되는 마지막 파라미터(스프레드 랜덤 인터리버에 의한 무작위 적으로 선택된 치환 테이블에서의 새로운 위치 계산)를 위해 사용될 위치 값들은 C# 언어에서 가장 작은 데이터 단위인 바이트 단위(8 비트)로 생성하였다.

4.2 성능 분석

그림 4.2 성능분석을 위한 응용 소켓 송수신 과정에서 성능 검증 과정은 실제 전달 응용메시지의 암호화 복호화를 통한 안전성 검증과 암호화 및 복호화의 속도 측면이 그 대상이 된다.

먼저 전달 응용메시지의 암호화 복호화를 통한 안전성 검증을 위해 아래 그림 4.3과 같은 성능 검증과정을 수행한다.

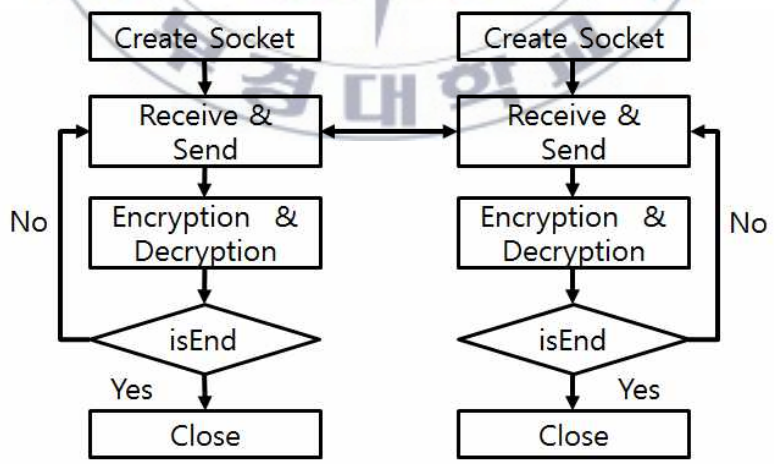

그림 4.3 성능 검증 과정
Fig. 4.3 Procedure of performance evaluation.

위의 성능 검정 과정 수행의 하나의 예로 아래 그림 4.4와 같이 송신 응용 메시지에 I-DES 암호화 과정을 거쳐 송신(클라이언트 응용 메시지 프로그램)하고 서버의 응용 프로그램에서 복호화를 수행한 후 다시 암호화하여 클라이언트 응용 프로그램으로 송신한 후 이를 또다시 복호화 하여 보낸 응용 메시지와 되돌려받은 응용 메시지가 일치하는 경우를 보여준다. 이 과정을 무작위 횟수만큼 수행한 결과에 따르면 100% 안전성 보장이 확인되었다.

그림 4.4 암호문의 송수신
Fig. 4.4 Transmission and reception of ciphertext.

둘째로 본 논문에서 제안한 I-DES 알고리즘의 성능 검정을 위해 DES와 I-DES의 치환 테이블 구성 속도 측면을 분석하였다. 이 과정에서 CBC 개념이 적용되었고 시뮬레이션 결과에 따르면 큰 차이가 나지 않

음을 확인하였다. 즉 맨 처음으로 스프레드 랜덤 인터리버 방법을 이용한 I-DES 알고리즘이 DES에 비해 보다 치환 테이블 구성에 많은 시간이 걸리나, 인터리빙 과정이 최초에 한번만 수행되어 치환 테이블을 생성하게 되므로 치환 속도에 영향이 크게 미치지 않아 유사한 수행속도를 보여준다.

셋째로 DES와 I-DES 알고리즘의 암호화 및 복호화 과정의 속도에 대한 성능 분석으로 그림 4.5 및 그림 4.6에 시뮬레이션 결과를 도식화하였다. 먼저 그림 4.5의 측정 결과를 보면 거의 암호화 속도가 비슷함을 알 수 있고 이는 치환 테이블을 한번만 생성되어 큰 블록의 대상 메시지에서는 치환 속도가 암호화에 영향이 없음을 보여준다. 다른 한편으로 그림 4.6의 복호화 과정도 거의 유사한 형태를 나타낸다.

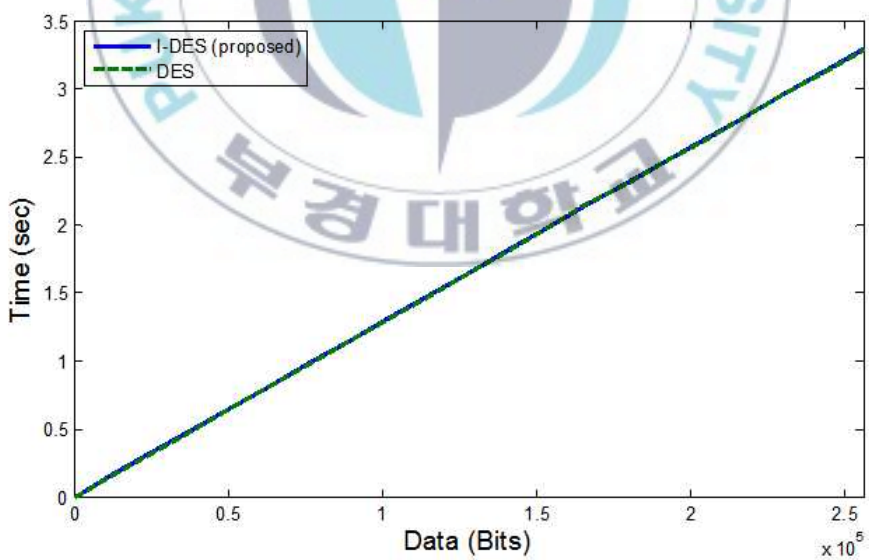

그림 4.5 CBC 알고리즘을 적용한 I-DES와 DES 알고리즘의 암호화 속도 비교

Fig. 4.5 Comparison of proposed CBC-based I-DES and legacy DES algorithms relative to encryption speed.

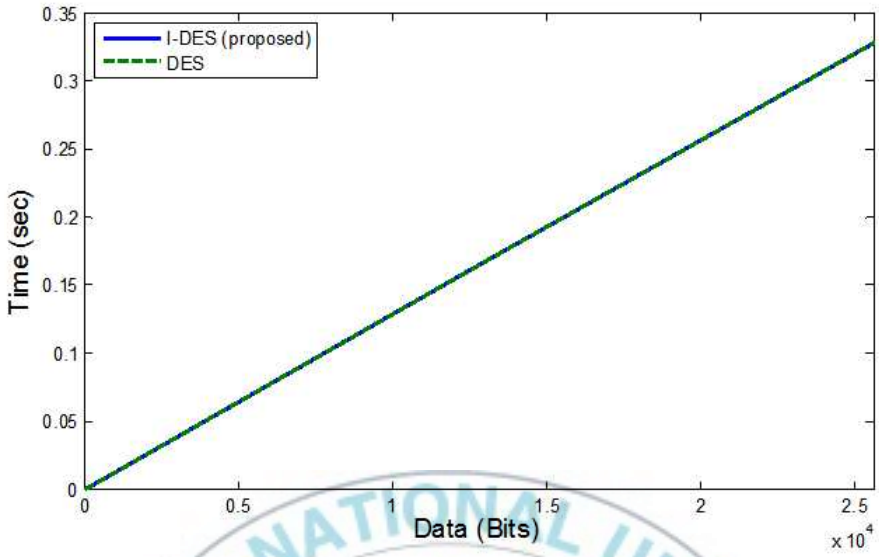

그림 4.6 CBC 알고리즘을 적용한 I-DES와 DES 알고리즘의 복호화 속도 비교

Fig. 4.6 Comparison of proposed CBC-based I-DES and legacy DES algorithms relative to decryption speed.

넷째로 평문의 비트수 변화에 대해 암호문의 비트수 변화된 정도를 통하여 확산성 검증을 수행하였다. 여기서 확산성의 문제는 같은 평문을 대상으로 1비트 또는 2비트의 내용을 바꾸었을 때 암호화된 암호문의 내용이 변화가 많아야 원래 평문을 예측하는데 어려움 정도가 높아지는 것에 대한 척도이다.

그림 4.7은 평문의 비트수 변화에 대한 암호문의 비트수 변화를 보이는 시뮬레이션 결과이다. 결과에 따르면 평문의 변화된 비트수에 무관하게 I-DES나 DES 알고리즘에 의한 암호문의 내용이 평균적으로 50.88%가 바뀌므로 확산성면에서 안전도가 높음을 알 수 있다.

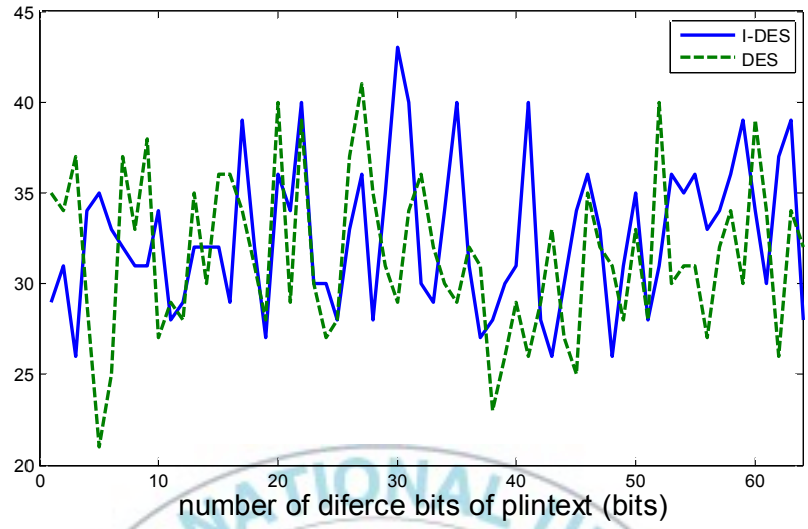

그림 4.7 I-DES와 DES 알고리즘의 쇄도효과
Fig. 4.7 Avalanche effect of I-DES and legacy DES algorithms.

결론적으로 고찰하면 CBC 기반의 I-DES의 적용은 안전성 보장 측면에서 DES와 보다 더 높은 효과를 가질 수 있고, 또한 암호화 및 복호화 속도 측면에서도 별 차이가 없이 활용이 가능하며 확산성 문제도 별반 차이가 없음을 확인할 수 있다. 결과적으로 같은 조건에서 I-DES의 적용이 훨씬 높은 안전성을 제공한다.

제5장 결 론

기존의 DES 알고리즘은 보안성 측면에서 취약하여 개인기반 클라우드 컴퓨팅 환경에 적용하기에 어려움이 있다. 이러한 문제점을 개선한 3DES 알고리즘의 경우에 암호화키의 크기가 112비트로 커져 DES에서의 전수 조사 공격으로 인한 문제는 감소하였으나, DES 알고리즘의 암호화 및 복호화를 3번 수행해야 하므로 많은 시간이 요구된다.

본 논문에서는 기존의 DES 알고리즘의 정적인 초기치환 및 최종치환 테이블을 스프레드 랜덤 인터리버 방식을 이용한 동적인 치환 테이블로 대체하여 개선된 I-DES 알고리즘을 제안하였다.

테스트베드 구성을 통해 I-DES 알고리즘과 기존의 DES 알고리즘 적용에 대한 안전성 확보 정도와 암호화 및 복호화 수행 속도, 그리고 확산성 측면에서의 안전도 검증 측면으로 시뮬레이션을 수행하였다. 결과에 따르면 DES 알고리즘의 단순성을 유지하면서도 더 높은 안전성이 제공됨을 확인하였다.

향후 과제로는 인터리버 기능뿐만 아니라 DES 알고리즘의 라운드 함수를 병렬로 처리하여 치환 및 암호화 소요시간을 더욱 단축하고 기밀성 및 보안성이 강화된 방법을 찾아 개인기반 클라우드 컴퓨팅에 효과적으로 적용하는 것이다.

참 고 문 헌

[1] 김지연 외 3명, “클라우드 컴퓨팅 환경의 가상화 기술 취약점 분석연구”, 한국정보보호학회, 제 19권 제 4호, pp. 72-77, 2009

[2] National Bureau of Standards, “Data Encryption Standard”, National Bureau of Standards, U.S. Department of Commerce, NBS FIPS PUB 46 1977

[3] National Institute of Standard and Technology. “Advanced Encryption Standards”, NIST FIPS PUB 2001

[4] Behrouz A. Forouzan, “암호학과 네트워크 보안”, Mcgraw-Hill Korea, 2008

[5] 신승수, 한군희, “암호화된 데이터베이스에서 인덱스 검색 시스템 구현”, 한국산한기술학회논문지, 제11권 제5호, pp.1653-1660, 2010

[6] S. Dolinar and D. Divsalar, “Weight distributions for Turbo codes using random and nonrandom permutations”, JPL TDA Progress Report, pp.56-65, 1995

[7] William F. Ehrsam, Carl H. W. Meyer, John L. Smith, Walter L. Tuchman, "Message verification and transmission error detection by block chaining", US Patent 4074066, 1976

[8] L. Blum, M. Blum, M. Shub, “A Simple Unpredictable Pseudorandom Number Generator”, SIAM Journal on Computing, Vol. 15, No.2, pp.364-383, 1983

[9] NIST, http://www.nist.gov

[10] William M. Daley, Raymond G. Kammer, NIST, 1999

[11] H. Feistel, “Cryptography and Computer Privacy”, Scientific

American, pp.15-23, 1973

[12] 원동호, “현대 암호학”, 도서출판 그린, pp.3, 2003

[13] NIST, "Recommendation for the Triple Data Encryption Algorithm(TDEA) Block Cipher," NIST FIPS PUB 800-67, 2008

[14] 김광조, “DES의 선형 해독법에 관한 해설(I)”, 정보보호학회, 정보보호학회 논문지, 제3권 제3호, pp.7-22, 1993

[15] W.Diffie and M.E.Hellman, "Exhaustive Cryptanalysis of the NBS Data Encryption Standard," IEEE, Vol.10, No.6, pp.74-84, 1977

[16] 노창오, 조범주, 문인규, “디지털 홀로그램을 이용한 전수키 조사 공격에 강인한 DES 알고리즘 구현”, 한국멀티미디어학회, 춘계학술발표대회논문집, 제13권1호, pp.67-68, 2011

[17] Juha Heiskala, “OFDM 무선랜”, 브레인코리아, 2003

[18] 진익수, 노예철, 주유상, 강법주, “터보 부호의 인터리버 분석”, 정보통신산업진흥원, 주간기술동향 905호, 1999

[19] 황재영, 최동욱, 정연호, “모바일환경에서 클라우드 컴퓨팅 보안을 위한 효율적인 암호화기술”, 한국정보통신학회, 2011

[20] D. Divsalar, F. Pollara, “Multiple Turbo Codes for Deep-Space Communication”, The JPL TDA Progress Report, pp.42-121, 1995

[21] 신상호, 최장희, 유기영, “난수 및 의사난수생성기에 대한 평가도구의 분석”, 대한전자공학회 하계학술대회, 제33권 1호, pp. 1648-1651, 2010

[22] Thomas M. Cover, Joy A. Yhomas, “Elements of Information Theory”, Wiley, 2005

[23] 정계영, “DES와 AES의 비교분석“, 관동대학교 교육대학원, 2002

[24] Niels Ferguson, Bruce Schneier, “슈나이더의 크립토그라피”, Wiley, 2004

연구논문 발표실적

등재 논문

[1] 황재영, 정연호, Perrmalloy 를 이용한 효율적인 무선 전력송신 기술, 한국정보통신학회, 2009

[2] 황재영, 이주한, 이호진, 정연호, 효율적인 차량제어를 위한 모바일기반의 범용 통합 제어모듈, 한국정보통신학회, 2009

[3] 황재영, 최동욱, 정연호, CBC-MAC 방식을 적용한 보안 모바일기기 제어시스템, 한국정보통신학회, 2010

[4] 황재영, 정신일, 정연호, 스마트폰을 이용한 차량용 주행 모니터링 모듈 개발, 한국정보통신학회, 2010

[5] 황재영, 최동욱, 정연호, 모바일환경에서 클라우드 컴퓨팅 보안을 위한 효율적인 암호화기술, 한국신호처리시스템학회, 2011

[6] 최동욱, 황재영, 정연호, 차세대 이동통신 시스템에서 유동적 USIM 카드를 이용한 인증 시스템, 한국정보통신학회, 2011

참여 연구

[1] 선박기기용 문선충전 장치 개발, 2008

[2] 모바일 기기를 이용한 수송기계제어 통합모듈 개발, 2009

[3] 스마트폰을 이용한 차량용 주행모니터링 모듈 개발, 2010

[4] 모바일 환경에서 클라우드 컴퓨팅 보안을 위한 효율적인 암호화 기술, 2010

[5] 퍼스널 클라우드 컴퓨팅을 위한 보안시스템 개발, 2012

특허

[1] 무접점 전력송수신기, Contactless power transfer, 2011

[2] 모바일 기기를 이용한 수송기계 제어시스템 및 그 제어방법, Method for Transportation Machine Using Mobile Apparatus, 2011

감사의 글

대학에 온지 10년이 되었습니다. 그동안 많은 것을 배우고, 또 많은 사람들을 만날 수 있었습니다. 이제는 학교가 아닌 사회에서 또 다른 것을 배우고, 또 새로운 사람들을 만나려 합니다.

지도교수님이신 정연호 교수님께서는 항상 대학생활에서 배우는 모든 것들이 사회 나갈 무기가 될 것이라 하셨습니다. 그 말을 항상 생각하며 대학과 대학원 생활을 하였고, 목표로 하던 곳에 일을 할 수 있게 한 원동력이라고 생각합니다. 이 글을 통해서나마 감사의 말을 올립니다.

그리고 지도 교수님으로서, 논문 마감까지 바쁘신 와중에도 작성 방법부터 하나하나 도와주신 김성운 교수님과 류지열 교수님께도 감사의 말을 전하고자 합니다.

저의 인생선배이자 연구실 선배로서 항상 인생 멘토가 되어주신 최기영 선배님, 최동욱 선배님 항상 감사하는 마음을 지면을 통해서 전합니다. 그리고 4년 동안 MTS 연구실 생활을 하면서 같이 있어주었던 동기 동희, 그리고 나보다 먼저 사회에 나간 후배들 주한, 진현, 호진, 영석, 희근, 오래 있지는 못하였지만 준영, 세종에게도 앞으로 하는 일 잘되길 바라며, 또 부족한 선배에게 많은 힘이 되었던 후배들에게도 감사의 말을 전합니다.

핀란드에서 많은 도움 주신 정상중 선배, 석사 동기들 재식, 종욱, 상호, 순태, 그리고 USN 연구실에 재호, 성모에게도 고맙다는 말 남깁니다. 그리고 미쳐 지면에 싣지 못한 친구, 선후배 분들 그리고 저를 아는

모든 분들에게 죄송스럽게 생각하며 앞으로 하시는 일 모두 잘 되기를 바랍니다.

마지막으로 힘든 집안 사정에도 오랫동안 학교에 다니느라 미안한 마음뿐인 어머니에게도 고맙고 앞으로 받은 것 이상 보답하겠습니다.

2012년 6월

황 재 영 올림