전자상거래 보안을 위한 YK2 암호 알고리즘의 설계

강 영 구† · 류 성 열††

요 약

전자상거래는 인터넷이라는 가상 공간을 통해 이루어지므로, 시간과 장소에 구애받지 않는다는 장점이 있는 반면, 누구나 쉽게 접근할 수 있는 개방형 네트워크의 특성에 따라 보안상의 문제점이 내포될 수 있다는 약점을 동시에 갖고 있다. 따라서, 안전하고 효율적인 전자상거래 구현을 위해서는 전자상거래 보안상의 문제를 해결하기 위한 장치가 필요한데, 이러한 장치 중에 하나가 견고한 암호 알고리즘의 구축이라 할 수 있겠다.

본 논문에서 제안한 YK2(Young Ku Kang) 암호 알고리즘은 이러한 전자상거래 보안을 충족시켜주기에 적합한 시스템으로서, 기본적으로 입·출력의 키와 암호화에 사용되는 키의 크기가 각각 128비트로 구성되고 32회전으로 전개됨으로서 기존의 64비트를 사용했던 블록 암호 알고리즘의 한계를 극복할 수 있다. 또한, 암호화에 민감한 영향을 미치는 요소 중에 하나인 키 스케쥴링을 보다 복잡하게 설계함으로써 계산 복잡도의 증가와 확률 계산의 증가를 도모하였다.

Design of YK2 Cipher Algorithm for Electronic Commerce Security

Young-Ku Kang† · Sung-Yul Rhew††

ABSTRACT

EC(Electronic Commerce), which is done the virtual space through Internet, has the advantage of time and space. On the contrary, it also has weak point like security problem because anybody can easily access to the system due to open network attribute of Internet. Therefore, we need the solutions that protect the EC security problem for safe and useful EC activity. One of these solutions is the implementation of a strong cipher algorithm.

YK2(Young Ku Kang) cipher algorithm proposed in this paper is advantage for the EC security and it overcomes the limit of the current 64 bits block cipher algorithm using 128 bits key length for input, output, encryption key and 32 rounds. Moreover, it is designed for the increase of time complexity and probability calculation by adapting more complex design for key scheduling regarded as one of the important element effected to encryption.

1. 서 론

최근 들어, 통신과 컴퓨터 기술이 비약적으로 발전하면서 일상적으로 행하던 은행 업무, 쇼핑 등의 일부 거래가 컴퓨터 네트워크를 이용, 전자 공간에서 이루어지고 있다. 특히, 인터넷의 급속한 확산에 의하여 많은 수의 인터넷 쇼핑몰들이 개장되면서 전자상거래도 역시 급속히 확산되고 있는 실정이다. 한편, 전자상거래를 보다 활성화하기 위해서는 전자상거래상의 안전성과 신뢰성을 확보할 수 있는 보안 기술이 필수 선결 요소라고 하겠다[7].

그러나, 전자상거래 상에서의 보안 기술 문제는 그리 쉬운 문제가 아니다. 왜냐하면, 인터넷의 개방 지향적인 특성 때문에 누구나가 쉽게 접속이 가능하고 해킹에 의한 정보 접근이 용이하기 때문이다. 따라서, 강

※ 이 논문은 서울산업대학교 교내 학술연구비 지원에 의하여 연구되었음
† 정 회 원 : 서울산업대학교 전자계산학과 교수
†† 종신회원 : 숭실대학교 컴퓨터학부 교수
논문접수 : 2000년 8월 12일, 심사완료 : 2000년 10월 17일

보의 암호화는 이러한 보안상의 문제점을 해결하기 위한 한 가지 대안으로 제시될 수 있다.

이에 본 논문에서는 효율적인 암호화 알고리즘인 YK2 암호 알고리즘을 제안함으로써 정보 보안상의 문제점을 해결하여 전자상거래의 거래 정보나 거래 내용들의 안전성과 신뢰성을 도모하도록 한다.

본 논문에서 제안한 YK2 암호 알고리즘은 하나의 데이터 블록이 다른 3개의 데이터 블록을 변환시키는 Type-3 Feistel 구조를 적용하여 설계하였다[4, 14]. 이를 바탕으로 암호문의 생성 속도가 빠르게 진행되고, 외부 공격에도 대처할 수 있도록 설계하였다.

2. 전자상거래 보안

2.1 전자상거래 상에서의 보안 문제

전자상거래라 함은 통합적으로 자동화된 정보체계 환경 하에서 기업과 기업간, 기업과 정부간, 기업과 개인간 거래관계의 모든 측면에 걸쳐 생산, 구매, 대금 지불, 수송, 행정, 서비스 등의 제반 비즈니스를 네트워크를 통해 전자적으로 행하는 것으로 정의할 수 있다[2].

최근 들어, 네트워크의 발달에 따른 개방형 통신망(open network)에 있어서 각 개인이나 기업 또는 정부의 각종 정보들이 인터넷을 통해 손쉽게 상대방에게 전달되고 있다. 바꾸어 말하면, 이러한 다수의 정보들을 다른 사람이 인터넷을 통하여 손쉽게 접근 할 수 있다는 것을 의미한다. 더욱이 웹 기술의 발달에 따라 일반 사용자들도 이러한 정보들에 손쉽게 접근이 가능해짐으로써 인증이나 개인의 사생활 및 개인정보 등의 전자상거래 보안 문제는 더욱 더 중요한 과제로 대두되고 있다.

따라서, 전자상거래에서도 인터넷 보안 문제를 반드시 고려하여야 하며, 인터넷상의 정보보호 문제를 포함한 보안 문제를 해결하기 위한 제반 장치가 마련되지 않는다면 전자상거래의 안전성과 신뢰성을 기대할 수가 없다. 이러한 정보보호 문제를 해결하기 위한 근본적인 방법 중의 하나가 네트워크 상에서 송·수신 메시지나 거래 내용에 관한 정보를 암·복호화하여 사용하는 암호 기술이다.

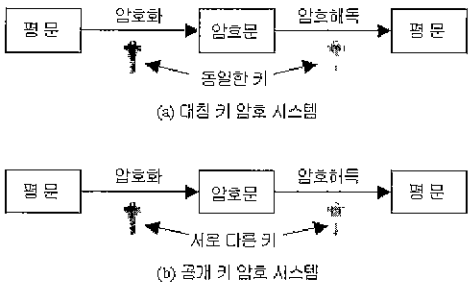
2.2 암호 기술

암호화란 원래 내용을 숨겨 원본과 달라진 전달문 혹은 비밀문을 만드는 과정으로서, 한 문자를 다른 문자로 대체하는 일련의 규칙들의 집합이며, 이 규칙들은 XOR나 어떤 수학적인 기법을 통해 만들어진다.

이러한 암호화를 기반으로 한 암호 기술은 암호키의 운용에 따라 대칭키 암호 시스템과 공개키 암호 시스템으로 크게 구별된다. 대칭키 암호 시스템(대칭형)에서는 송·수신자가 동일한 비밀키를 공유해야 한다.

이에 비해 공개키 암호 시스템(비대칭형)은 키 분배 문제(또는 키 공유 문제)에 근거하여 공개키 분배 알고리즘과 공개키 관리 알고리즘으로 나누어질 수 있다. 이것은 암·복호화 키가 서로 다르며, 어느 한 키를 공개하더라도 대응되는 다른 키를 유도해 내는 것은 계산상 불가능하도록 설계되어진다[9].

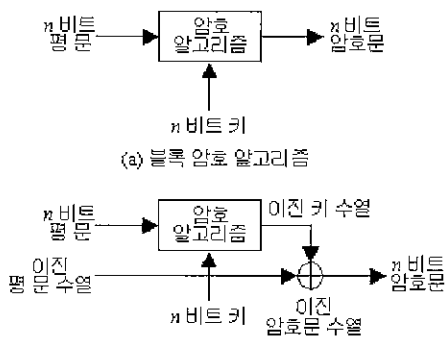
이상에서 설명한 암호시스템을 그림으로 비교해 보면 다음의 (그림 1)과 같다.



(그림 1) 대칭키 암호 시스템과 공개키 암호 시스템의 비교

2.3 대칭키 암호 시스템

대칭키 암호 시스템은 메시지 처리 형식에 따라 다시 블록 암호 알고리즘과 스트림 암호 알고리즘으로 나누어진다.



(그림 2) 대칭키 암호 시스템의 형태 비교

블록 암호 알고리즘은 (그림 2a)와 같이 고정된 크기의 입력 블록이 암호화에 사용된 임의의 키에 의해 고정된 크기의 출력 블록으로 변형되는 암호 알고리즘으로써, 출력 비트의 각 비트는 입력 블록과 암호화에 사용된 키의 모든 비트에 영향을 받아 결정된다[8].

이에 비해, 스트림 암호 알고리즘은 선형 쉬프트 레지스터(linear shift register)를 이용한 이진 수열 발생기를 사용하는 암호 알고리즘의 형태로서, (그림 2b)와 같이 평문을 이진 수열로 부호화 한 뒤, 이를 이진 수열 발생기에서 생성된 이진 수열과 XOR하여 이진 수열로 된 암호문을 생성하는 방식이다.

3. YK2 암호 알고리즘

3.1 YK2 암호 알고리즘의 소개

본 논문에서 제안한 YK2 암호 알고리즘은 대칭키 암호 알고리즘에 바탕을 둔 128 비트 블록 암호 알고리즘이다[11]. 또한, 암호화 키는 128 비트의 블록 크기와 128~400 비트 범위의 가변적인 키 크기이고, 높은 보안성과 빠른 속도 구현의 융통성을 제공하기 위해 다양한 평령을 사용함으로써 사실상 3-DES(triple-Data Encryption Standard) 보다도 더 견고한 안정성을 제공한다[12].

이러한 YK2 암호 알고리즘의 전체 구조를 보면 다음과 같은 특성으로 구성되어 있다.

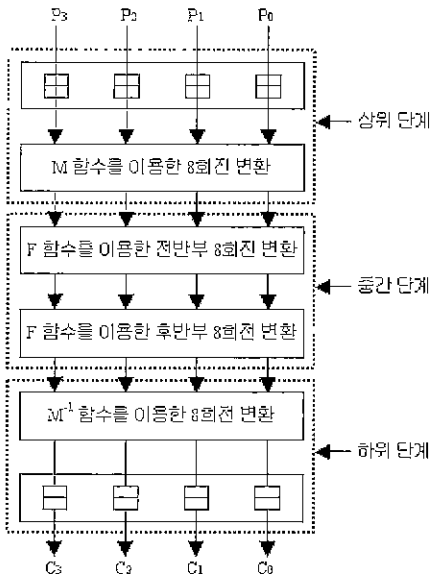
- 성 격 : 128 비트 블록 암호 알고리즘
- 구 조 Type-3 Feistel 구조
- 입/출력문의 크기 128비트
- 암호키의 크기 128~400비트
- 키 확장 프로시져 K[0…39]
- 회전 수 : 32회전
- 내부 함수 : M, F, M^{-1} 함수
- S-Box : 2개(S^1, S^2)

3.2 YK2 암호 알고리즘의 설계

3.2.1 전체 구성

YK2 암호 알고리즘은 128비트 평문을 4개의 32비트 블록 P_0, P_1, P_2, P_3로 분할하여 입력하고, 이를 로대로 키를 부가하여 입력 내용을 변환시키는 8회전의 상위 단계와 실제적인 암호화를 수행하는 16회전의 중간 단계, 그리고 상위 단계를 역으로 수행한 후 키를 감소시키는 8회전 하위 단계를 거친 후에 4개의 32비트 블록 C_0 C_1, C_2, C_3를 산출한 후, 최종적으로 이것들을 조합하여 128비트 암호문을 생성하는 구조로 설계되었다.

본 논문에서 제안한 YK2 암호 알고리즘의 전체 구조는 다음의 (그림 3)과 같다.



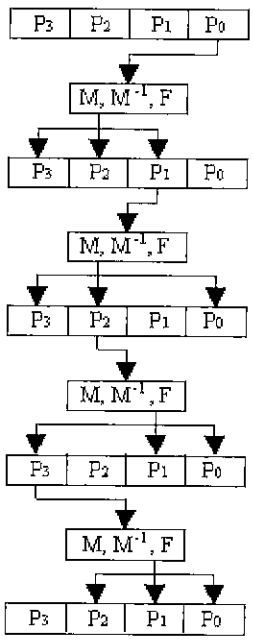
(그림 3) YK2 암호 알고리즘의 전체 구조

위의 그림에서, 상위 단계는 선택 평문 공격을 어렵게 하고 중간 단계에서 생성될 암호문에 대한 외부 공격을 어렵게 만들기 위한 과정으로서 M함수를 이용한 8회전으로 수행된다.

중간 단계는 실제로 암호화가 이루어지는 핵심 과정으로서 암호화와 복호화가 동일한 강도로 수행될 수 있도록 F함수를 이용한 전반부 8회전과 후반부 8회전으로 구분하여 총 16회전으로 수행된다.

그리고, 하위 단계는 상위 단계에서 수행된 것을 역순으로 치리하는 과정으로서 M^{-1} 함수를 이용한 8회전으로 수행되는데, 이것은 계산 복잡도를 증가시키고 선택 암호문 공격에 안전하게 대처하기 위한 단계이다.

그리고, (그림 4)는 사용되는 각 함수의 수행 과정, 즉 Type-3 Feistel 연산을 도식화 한 것이다. 여기서, M^{-1}는 M 함수를 역으로 수행하는 함수를 의미한다.

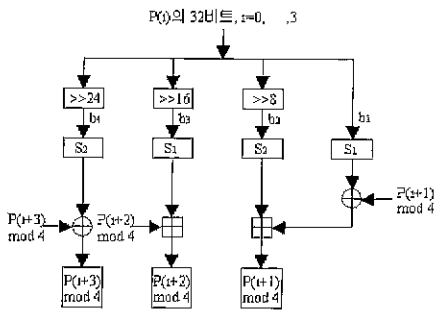


(그림 4) 각 함수의 변환 단계 구조

이제 각 단계별로 수행되는 과정을 살펴보도록 하자

(1) 상위 단계

이 단계에서는 4개의 각 평문 블록들에 키를 더한 후 8회전의 Type-3 Feistel 연산을 수행한다. 각 회전에서는 하나의 블록(초기 블록)을 사용하여 다른 3개의 블록(목표 블록)을 변경시킨다. 즉, (그림 5)와 같이 M 함수를 이용하여 초기 블록의 32비트를 우측으로 8비트씩 이동하면서 그 때마다 최하위 8비트를 2개의 S-Box(Sub- stitution-Box) S_1과 S_2에 대한 인덱스로 사용하여 이에 대응되는 S-Box의 내용들을 다른 3개의 목표 블록들과 더하거나 XOR 시킨다.



(그림 5) M 함수의 구조

이를 좀 더 자세히 설명하면, 예를 들어 처음 초기 블록은 32비트의 P(0)이고 목표 블록 역시 각 32비트의 P(1), P(2), P(3) 등 3개 블록이라고 하자. 초기 블록 P(0)의 32비트를 b_1, b_2, b_3, b_4라 할 때(여기서 b_1은 최하위 8비트이고, b_4는 최상위 8비트이다), b_1과 b_3는 S-Box S_1의 인덱스로 사용하고 b_2와 b_4는 S-Box S_2의 인덱스로 사용하게 된다. 먼저, 첫 번째 목표 블록 P(1)에 $S_1[b_1]$을 XOR 하고 다시 그 목표 블록에 $S_2[b_2]$를 더한다. 또한, 두 번째 목표 블록 P(2)에는 $S_1[b_3]$를 더하고, 세 번째 목표 블록 P(3)에는 $S_2[b_4]$를 XOR 한다.

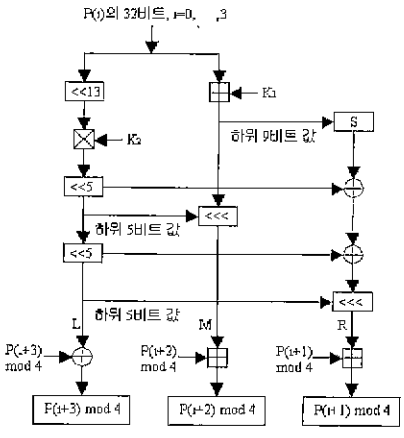
결국 이와 같은 연산을 수행하기 위해서는 (그림 5)에서 보는 바와 같이 초기 블록은 우측으로 24비트 회전이동 하게 된다.

그런 후에, 다음 회전을 위하여 4개의 평문 블록을 회전이동 시킨다. 즉, 현재의 첫 번째 목표 블록은 다음의 초기 블록이 되고 현재의 두 번째 목표 블록은 다음의 첫 번째 목표 블록, 현재의 세 번째 목표 블록은 다음의 두 번째 목표 블록, 그리고 현재의 초기 블록은 다음의 세 번째 목표 블록으로 각각 이동시킨다.

위와 같은 순서를 두 번 반복해서 8회전 수행한 후 그 결과를 중간 단계로 이동시킨다.

(2) 중간 단계

YK2 암호 알고리즘의 핵심 단계로서 16회전 Type-3 Feistel 연산을 수행한다. 각 회전에서는 F함수를 이용하여 상위 단계에서 산출된 평문 블록을 암호문으로 변환시킨다. 여기에서, F함수는 하나의 초기 블록을 입력으로 하여 3개의 출력 블록을 반환하도록 하는 함수이며, 그 구조는 (그림 6)과 같다



(그림 6) F 함수의 구조

이제 먼저 F함수에 대하여 자세히 설명하기로 한다. 우선, 편의상 3개의 임시 변수 L, M, R을 설정한다. 초기 L값은 초기 블록을 좌측으로 13 자리 회전이동 시킨 비트 값으로, M값은 초기 블록과 키(K_1)를 더한 값으로, R값은 M의 하위 9비트를 S-Box 배열의 인덱스로 사용하여 이에 대응되는 S-Box의 비트 값으로 정한다.

그런 후에, 키(K2)를 L과 곱하여 이를 5자리 좌측으로 회전이동 시킨다. 그리고, L을 R에 XOR 시키고, 또한 L의 하위 5비트 값을 회전이동 양으로 하여 M을 좌측으로 회전이동 시킨다. 다음으로 L을 5자리 좌측으로 더 회전이동 시키고 다시 R과 XOR 시킨다. 최종적으로 다시 L의 하위 5비트를 회전이동 양으로 하여 R을 좌측으로 회전이동 시킨다. 이러한 과정을 통하여 생성된 첫 번째 출력 블록을 L, 두 번째 출력 블록을 M, 그리고 세 번째 출력 블록을 R이라 한다.

중간 단계에서는 앞에서 언급했듯이 F 함수의 출력 블록과 목표 블록들의 연산을 두 번 반복하여 전반부 8회전과 후반부 8회전을 역순으로 적용한다. F함수의 첫 번째 출력 블록 L과 세 번째 출력 블록을 XOR하고, 두 번째 출력 블록 M과 세 번째 출력 블록 R을 두 번째 목표 블록과 첫 번째 목표 블록에 각각 더한다. 또한, 후반부 8회전의 각 회전에서는 첫 번째 출력 블록 L과 첫 번째 목표 블록을 XOR하고, 두 번째 출력 블록과 세 번째 출력 블록은 두 번째 목표 블록과 세 번째 목표 블록에 각각 더한다.

추가로, 매 회전이 끝난 후에는 다음 회전을 위하여 초기 블록을 13자리 좌측으로 회전이동 시킨 후 세 번째 목표 블록으로 하고 첫 번째 목표 블록은 초기 블록으로, 두 번째 목표 블록을 첫 번째 목표 블록으로, 세 번째 목표 블록을 두 번째 목표 블록으로 하여 F 함수를 이용하여 16회전을 반복 수행한다.

(3) 하위 단계

이 단계에서는 초기 블록이 상이한 순서로 처리되는 것을 제외하고는 상위 단계에서의 과정을 역순으로 처리한다. 즉, M^{-1} 함수를 이용하여 상위 단계의 출력 블록을 하위 단계의 입력 블록으로 처리하는 것을 의미한다. 하위 단계에서도 상위 단계과 동일하게 각 회전에서는 하나의 초기 블록을 이용하여 다른 3개의 목표 블록들을 변경시킨다. 다시 말해서, 초기 블록의 32비트를 b_1, b_2, b_3, b_4라 할 때 b_1과 b_3는 S-Box S_2의 인덱스로 사용하고 b_2와 b_4는 S-Box S_1의 인덱스로 사용하여, 첫 번째 목표 블록에서 $S_2[b_1]$을 XOR 시키고, 두 번째 목표 블록에서 $S_1[b_4]$를 빼고, 세 번째 목표 블록에서 $S_2[b_3]$를 뺀 후 여기에 $S_1[b_2]$를 XOR 한다. 결과적으로 초기 블록은 좌측으로 24비트 회전이동 시킨 결과가 된다.

그런 후에, 다음 회전을 위하여 4개의 평문 블록을 회전이동 시킨다. 즉, 현재의 첫 번째 목표 블록은 다음의 초기 블록이 되고 현재의 두 번째 목표 블록은 다음의 첫 번째 목표 블록, 현재의 세 번째 목표 블록은 다음의 두 번째 목표 블록, 그리고 현재의 초기 블록은 다음의 세 번째 목표 블록으로 각각 이동시킨다.

위와 같은 순서를 반복적으로 8회전 수행한 후 그 결과를 최종적으로 암호문으로 산출하게 된다.

3.2.2 키 스케줄링

키 스케줄링은 실제 암호화를 수행하는 중간 단계인 F함수 내부에서 사용되는 K_1, K_2의 키 스케줄링으로써, 임의로 구성된 각 32비트의 n개($4 \le n \le 14$) 키를 40개의 키로 확장하는 프로시저를 의미한다. 그런데, 키 확장 프로시저는 F함수 내의 곱셈에서 사용되는 키들이 다음과 같은 특성들을 갖도록 보장하는 어떤 알고리즘이 있어야 한다.

- 곱셈에 사용된 키에서 2개의 최하위 비트는 1이다.
- 어떤 키도 10개 비트가 연속적으로 0이거나 혹은 1을 포함해서는 안된다.

이러한 특성을 유지하기 위한 절차는 다음과 같은 단계를 통해 구성될 수 있다.

1. 먼저 15개의 임시 데이블 T[]에 초기 키들을 n만큼 복사하고 나머지는 0으로 한다.

$$T[0 \cdots n-1] = k[0 \cdots n-1].$$
$$T[n] = n,$$
$$T[n+1 \cdots 14] = 0$$

2. 그런 다음, 아래와 같은 과정을 4번 반복하여 매 반복 과정마다 다음 10개의 확장키를 계산한다.
   - 배열 T[]에 다음과 같은 선형 공식 을 적용하여 변환시킨다.

$$T[i] = ((T[i-7 \bmod 15] \oplus T[i-2 \bmod 15]) << 3) \oplus (4i-j)$$

여기서 i는 0…14이며, j는 반복횟수 로 $j=0$일 때 1회전, $j=1$일 때 2회전 임을 의미한다.

- 4회전의 Type-1 Feistel 연산을 적 용하여 배열 T[]를 혼합시킨다

$$T[i] = (T[i]+S[T[i-1] \bmod 15]의 \text{ 하위 } 9비트]) << 9$$

- 그런 다음, T[]에서 10개를 취하여 확장키 배열 K[]에 재배치시킨다.

$$K[10j+i] = T[4i \bmod 15]$$

3 최종적으로, 암호를 위한 곱셈(K_2 키)에 사용되는 16개를 K[5], K[7], …, K[35]로 선정하여 이것들이 앞에서 언 급한 두 가지 특성을 반족하도록 아래와 같이 조정한나.

- j=K[i]∧3으로 하여 K[i]의 최하위 2비트를 j값으로 하고 ω = K[i]∨3으로 하면, ω의 최하위 2비트는 1이 된다.
- 10개 또는 그 이상의 연속적인 0이 나 1을 가지고 있는 ω에 대한 메스 크 M을 만든다. 즉, M은 ω_l이 10개 이상이 연속적인 0이나 1로 되어 있다면 M_l = 1로 하고, 0이나 1이 연속적으로 10개 이상이 아닐 경우에는 M_l = 0으로 한다 그리고, ω에서 0이나 1의 양쪽 마지막에 대응되는 M 에서의 1을 0으로 하고, M의 최하위 2비트와 최상위 비트를 0으로 한다 즉, $i<2$, $i=31$이거나 또는 ω의 I번째 비트가 (i+1)번째 비트나 (i-1)번째 비트와 다르면 M의 i번째 비트를 0으로 한다.

  예를 들어, $\omega=0^3 1^{13} 0^{12} 101 1$ 이라면, 처음에 M은 $M=0^3 1^{25} 0^4$로 된니. 그런 후에, 4, 15, 16, 28 자리의 1을 다시 0으로 하면 M은 $M=0^4 1^{11} 001^{10} 0^5$이 된다
- 다음으로 ω를 수정하기 위해 32비트 테이블 B를 이용하는데 이때, B데이블의 요소들은 7개의 연속적인 0이나 1을 포함하지 않도록 선택한다. 특히, B[]={a4a8d57b, 5b5d193b, c8a8309b, 73f9a978}을 사용하는데, 이것들은 S-Box내 S[265]~S[268]의 값들이다.

  이러한 요소들을 선택한 이유는 이 요소들에서는 2번씩 나타나는 유형이 단지 14개의 8비트 유형만 있고, 또한 어떤 유형도 2번 이상은 니타나지 않기 때문이다.

  그리고, K[i]를 이용하여 K[i-1]의 하위 5비트 값만큼 B[j]를 좌측으로 이동한 값을 p로 한다. 즉, 다음 수식과 같다.

$$p = B[j] << K[i-1]의 \text{ 하위 } 5비트 \text{ 값}$$

- 최종적으로, p를 메스크 M과 AND 연산을 시킨 후 XOR하여 그 결과를 K[i]로 한다. 결국 M의 최하위 2비 트가 0이므로 K[i]의 최하위 2비트는 1이 될 것이고 배열 B[]의 값들 때문에 K[i]는 10개의 연속적인 0이니 1을 갖지 않는다는 것이 보장된다.

  이런 키 확장 스케쥴링은 K[5], K[7], …, K[35]가 위에서 언급한 여러 가지 특성을 갖고 있다는 것을 보장할 뿐만 아니라 무작위 특성도 갖게 된다.

  이와 같은 키 확장 스케쥴링 알고리즘은 다음과 같다.

[알고리즘 1] 키 확장 스케쥴링

```
/* B[]의 초기화 */
B[] = {a4a8d57b, 5b5d193b, c8a8309b, 73f9a978}
/* 임의로 설정된 키로 T[] 초기화 */
T[0…n-1]=k[0…n-1], T[n]=n, T[n+1…14]=0

/* 4 회전 반복, 매번 반복마다 K[]의 10개키 계산 */
for j=0 to 3 do
 for i=0 to 14 do
  T[i]=((T[i-7 mod 15]⊕T[i-2 mod 15]) << 3) ⊕ (4i-j)
 repeat 4 times
  for i=0 to 14 do
   T[i]=(T[i]-S[low 9 bits of T[i-1 mod 15]]) << 9
 end repeat
 for i=0 to 9 do
  K[10j+i]=T[4i mod 15]
 end for

/* 곱하기 사용 키 변환 */
for i=5,7,…35 do
  j=K[i]의 최하위 2비트∧3
  ω=K[i]의 최하위(2비트∨3) 비트

/* ω의 비트 메스크 키 생성 */
 r=K[i-1]의 하위 5비트 값
 p=B[j] << r
 K[i]= ω⊕(p∧M)
end for
```

3.2.3 S-Box

S-Box는 대개의 블록 암호 알고리즘에서 사용되는 비선형 대입 연산을 수행하는 비선형 함수로서, 입력 크기와 출력 크기를 변경시킬 수 있으며, 임의적으로 생성시킬 수 있다. S-Box는 Lucifer 알고리즘에서 최초로 사용됐으며[4], 그 후 대개의 암호 알고리즘에서 널리 사용되고 있다. 이러한 S-Box는 암호화에 민감한 영향을 미치므로 S-Box의 구성을 어떻게 하느냐에 따라 견고한 암호 알고리즘을 구축할 수 있다.

암호화에 있어서 S-Box의 특성은 암호문 생성에 매우 중요한 요소라 할 수 있는데, 본 논문에서 설계한 S-Box는 의사난수(pseudo-random) 형태로 생성되어 만들어졌다. S-Box의 요소들은 $i = 0 \cdots 102$, $j = 0 \cdots 4$, $S[5i+j] = SHA\text{-}1(5i|c_1|c_2|c_3)j$로 하여 생성하였다. 여기서, SHA-1$(\cdot)j$는 SHA-1의 출력에서 j번째 비트이고 i는 32비트 정수를 나타내며, c_1, c_2, c_3는 고정 상수 값이다[9]

YK2 알고리즘에서 S-Box S_1, S_2는 다음과 같은 특성들을 만족하게끔 설계하였다.

- S-Box는 32비트가 모두 0이거나 또는 모두 1인 것은 포함하지 않는다.
- S-Box를 S_1과 S_2로 나눌 때, S_1과 S_2 에서 각각 두 개의 요소들은 32비트 중 적어도 3개가 달라야 한다.
- S-Box는 $S[i]=S[j]$, $S[i]=\neg S[j]$ 또는 $S[i]=-S[j]$가 되는 두 가지의 요소 $S[i]$, $S[j]$를 포함하지 않는다. 이때, i와 j 는 $i \neq j$ 이다
- S에서 두 개 원소를 XOR하면 $\binom{512}{2}$개, 뺄셈을 하면 $2\times\binom{512}{2}$개의 다른 요소가 된다.
- S의 모든 두 요소는 적어도 4비트가 다르다.

3.2.4 설계 배경

DES와 같은 반복적인 블록 암호를 해독하는 가장 강력한 방법은 각 회전에서 사용된 암호키를 찾아내는 공격 방법이다. 이러한 공격 방법의 예로는 차분 공격(differential attack)과 선형 공격(linear attack)이 있는데, 본 YK2 암호와 같이 자료의존회전(data dependent rotation)을 이용한 암호에서는 선형 공격에는 강하지만 차분 공격에는 약하다는 것이 증명되었다[10]. 그러나, 본 YK2 암호에서는 이러한 약점을 보완하기 위하여 회전량에서 어떤 차이가 발생하도록 설계하였다. 회전량에서 어떤 차이가 발생되면 그 회전 후의 출력 차이는 기본적으로 무작위가 됨으로써 그 결과의 특성이나 차분 형태를 차분 공격에 유용하게 사용될 수 없다.

앞에서 설명했듯이, F함수에서 사용되는 키의 특징은 키 스케줄링 알고리즘을 통하여 최하위 2비트는 모두 1이고 인접적으로 10개의 0이나 1을 포함하지 않도록 설계하였다. 키가 이러한 조건을 만족하여야 할 이유로는 <표 1>의 곱하기 연산에서 입력의 단일 비트 차이는 항상 출력에서 상위 10비트에 이런 차이를 보증하기 때문이다. 즉, F함수에서 두 개의 상이한 자료 워드와 키 곱하기 연산을 수행할 때 그 결과의 상위 10비트 값이 일치하지 않을 확률은 적어도 $1-2^{-8}$이기 때문에 거의 확실히 상이한 값들이 되는 의미이고 이 값을 회전량으로 이용케 하는 것이다.

3.2.5 YK2 암호 알고리즘의 차분 특성

YK2의 자료의존회전은 RC5 암호 알고리즘에서 동기를 얻었다. RC5의 가장 두드러진 특징은 자료의존회전과 간결한 설계를 들 수 있다. YK2는 RC5에서 사용된 자료의존회전에 곱하기 연산을 결합하여 사용함으로써 보다 더 강하게 하였다. 즉, YK2에서는 아래와 같이 F 함수에서 2번의 자료의존회전과 한 번의 곱하기 연산을 사용하였다.

〈표 1〉 YK2의 자료의존회전 특성

```
YK2의 F 함수
입력 : 32비트 크기의 1개 블록 in
       32비트 크기의 2개 키 K1, K2
출력 : 32비트 크기의 3개 블록 L,M,R
알고리즘
    1. M = in + K1
    2. L = (in <<< 13) * K2
    3. I = M의 하위 9비트
    4. R = S[I]
    5. L = L <<< 5
    6. M = M <<< L(자료의존회전)
    7. R = R⊕L
    8. L = L <<< 5
    9. R = L⊕R
   10. R = R <<< L(자료의존회전)
```

<표 1>에서 보는바와 같이 $L = (in<<<13)*K_2$에서 한 번의 곱하기가 수행되고, 첫 번째 자료의존회전은 $M = M<<<L$에서 회전량은 $r_1 = L[31..27]$이고, 두 번째 자료의존회전은 $R = R<<<L$에서 회전량은 $r_2 = L[26..22]$가 된다. 곱하기를 이용하는 이유는 곱하기의 분산 특징 때문에 곱하기 출력의 상위 비트를 회전량으로 사

용한다면 어떠한 자료의 입력시에도 높은 확률로 상이한 회전량을 유도할 수 있다는 것이다.

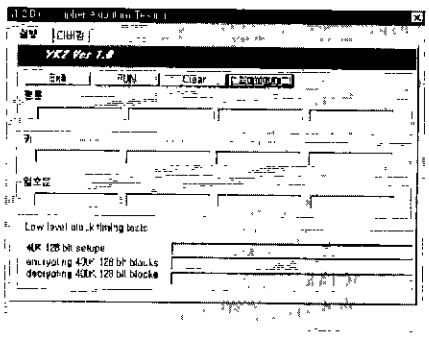
이러한 자료의존회전의 차분 특성 결과에 기초하여 YK2 암호 알고리즘은 차분 공격에 효과적으로 대응하게 된다.

3.3 YK2 암호 알고리즘의 구현

3.3.1 구현 환경

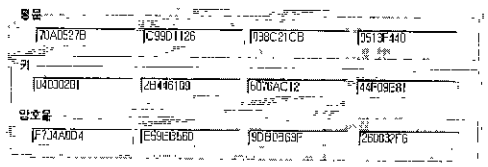
YK2 암호 알고리즘 구현에 사용된 환경과 초기화면 구성은 다음과 같다.

- 시스템 : Pentium II 400MHz
- 시스템 소프트웨어 : Windows 98
- 컴파일러 : MS Visual C++ 6.0
- 메모리 : 128MB

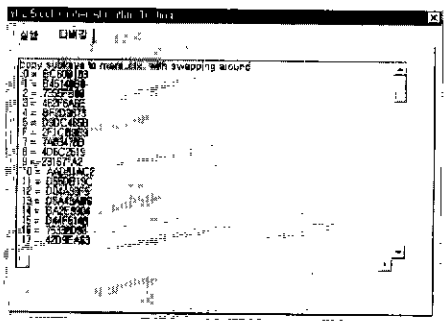


3.3.2 암·복호화 과정

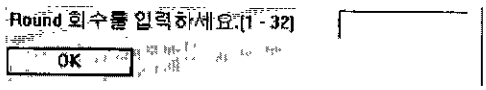
YK2 암호 알고리즘에서의 암호화 과정은 평문과 암호문, 그리고 키의 크기가 각각 128비트이다. 이것은 임의의 128비트 평문을 입력한 후 "RUN" 탭을 클릭하여 암호화에 사용된 키와 암호문을 산출하는 구조로 설계되었다.



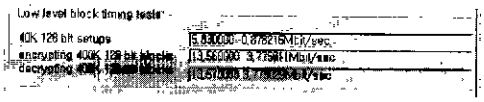
내부적으로 키는 키 스케줄링 프로시저에 의해 키를 40개(K[0]~K[39])로 확장하여 암호문을 생성하는데 이용하였다. 이와 같은 내용을 단계별로 확인하기 위해서는 메인 화면에서, "디버깅" 탭을 이용한다.



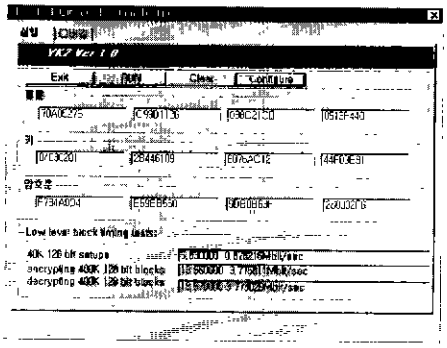
또한, 수행되는 회전 수를 조정할 수 있게 설계함으로서 프로그램에 의해 산출된 128비트의 암호문을 해독하는 과정에 있어서 전수조사 공격(exhaustive attack)이나 차분 공격 등으로 암호해독을 수행할 때, 수작업으로 3 회전 정도는 확인해 볼 수 있으므로, 화면 구성에서 "Round 회수를 입력하세요(1-32)"에 관한 메시지박스에서 회전수를 입력하여 위의 공격법에 의한 산출된 결과와 비교해 볼 수 있다.



그리고, YK2 암호 알고리즘에서 키 셋팅과 암·복호화 처리 속도를 산출하기 위해 "Low level block timing tests" 항목을 설계하였다



최종적으로, 본 논문에서 제안한 "YK2 Block Cipher Algorithm Testing" 프로그램에 의해 생성된 최종화면은 다음과 같다.



본 논문에서는 복호화 과정에 대해서는 생략하는데, 복호화의 과정 역시 암호화의 과정에서 산출된 암호문과 동일한 키를 사용하여 "RUN" 탭을 클릭하면 평문을 산출할 수 있다.

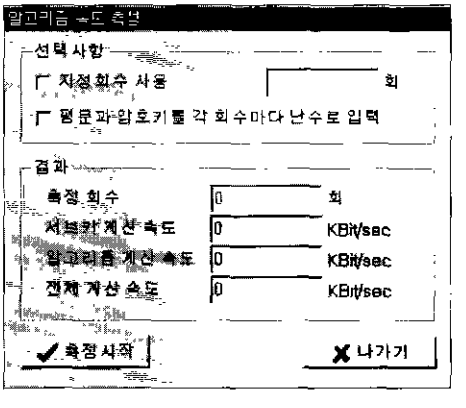
3.4 YK2 암호 알고리즘의 비교 분석

본 논문에서는 AES(Advanced Encryption Standard) 계열의 블록 암호 알고리즘들에서 사용된 키 스케줄링 알고리즘에 근거한 키 처리 속도와 암·복호화 생성 속도를 각각 측정하여 비교 분석하였다[13].

테스팅을 위한 실험 환경은 YK2 암호 알고리즘 구현에 사용된 환경과 동일하며, 실험 범위는 다음과 같이 설정하였다.

- 키 처리 속도는 임의의 키와 고정 키 를 사용하는 암호 알고리즘을 구별하 여 동일한 방식으로 처리하였으며, 키 에 대한 키 스케줄링 알고리즘을 반복 적으로 수행하여 이를 평균으로 산출한다.
- 암·복호회 생성 속도는 임의의 단위 블록을 반복적으로 임호화하는 과정과 이것을 반복적으로 복호화하는 과정을 수행하여 이를 평균으로 산출한다.
- 각 알고리즘에 대한 키 처리 속도와 암·복호화 생성 속도를 측정하기 위해 동일하게 10초간 수행하여 그 속도를 측정한다.

실험은 다음과 같이 개발한 알고리즘 속도 측정틀에 의해 수행하였다.



위의 툴을 이용하여 다음과 같은 내용들을 실험하였다.

- 지정회수 사용 : 실험할 회수 지정
- 평문과 암호키를 각 회수마다 난수로 입력 : 평문과 암호키 값을 난수와 고 정 값으로 구분하여 선택
- 측정 회수 : 실제 실험에 측정된 회수 출력
- 서브키 계산 속도 : 키 생성 알고리즘 에 의한 서브키 생성 속도 출력
- 알고리즘 계산 속도 : 서브키를 제외 한 알고리즘 자체 속도 출력
- 전체 계산 속도 : 전체 암·복호화 수 행 속도 출력

다음의 <표 2>는 위의 실험 환경과 실험 범위를 바탕으로 AES 블록 암호 알고리즘과 YK2 블록 암호 알고리즘을 중심으로 성능을 비교한 것이다

〈표 2〉 블록 암호 알고리즘의 성능 비교

알고리즘명	키 처리 속도	암·복호화 생성 속도
MARS	9.099 μsec	7.443 Mbps
RC6	5.058 μsec	11.304 Mbps
Rijndael	7.173 μsec	9.702 Mbps
Serpent	7.227 μsec	7.389 Mbps
Twofish	4.194 μsec	10.638 Mbps
YK2	3.872 μsec	6.867 Mbps

위의 실험 결과에서, YK2 암호 알고리즘의 암·복호화 생성 속도가 6.867 Mbps로 다른 AES 암호 알고리즘보다 빠르게 처리되며, 키 처리 속도가 3.872 μsec로 산출되므로 MAC(Message Authentication Code)이나 해쉬 함수의 블록 구축시 상당히 효율적으로 응용될 수 있다

4. 결 론

인터넷의 급속한 확산에 따라 인터넷을 상거래에 이용하는 전자상거래가 활발히 이루어지고 있다. 또한, 인터넷의 영향력이 날로 증대되고 인터넷 사용자 수가 점점 많아지면서 전자상거래는 향후 일반적인 기례 형태로 성착될 전망이다. 따라서, 이러한 전자상거래를 안전하게 실현하기 위해서는 암호 기술이 필수이며, 보안의 중요성이 점점 크게 부각되고 있다

본 논문에서 제안한 YK2 암호 알고리즘은 이러한 암호 기술에 적합하도록 안전성과 효율성을 고려하여 설계하였으며, 또한 일반적으로 암호화에 있어서 영향을 미치는 S-Box를 생성하기 위해 특성에 맞게 설계된 두 개의 S-Box S_1과 S_2를 사용하였고, 외부 공격에 의해 암호화에 사용된 키가 쉽게 발견되지 않도록 키 스케줄링 프로시져를 설계하였다.

제안한 YK2 암호 알고리즘은 민간분야의 전자상거래 상에서의 안전성과 신뢰성을 보장하기 위한 목적으로 개발했으며, 개인 및 기업의 컴퓨터 내 중요 정보 보호니 전자우편 시스템에서의 메시지 암호화, 그리고 인터넷을 이용한 전자상거래, 특히 전자화폐나 전자지불 시스템에 적합하게 사용될 수 있다. 또한, 금융 네트워크에서의 정보보호나 유료 수업 내용의 보호가 필요한 가상교육 시스템, 위성방송사업과 연계되어 유료 방송 내용에 관한 암호화를 필요로 하는 CAS(Conditional Access System) 등에 유용하게 사용될 수 있을 것이라 예측된다.

향후, YK2 암호 알고리즘의 보안 수준을 좀 더 향상시키기 위한 방법의 하나로 암호화에 영향을 미치는 키 스케쥴링을 견고하게 생성하기 위해 키 확장 알고리즘을 좀 더 보완하고, 동일한 구조로 설계된 다른 형태의 S-Box를 적용시켜 봄으로서 그것을 분석하고, 내부 함수 F 내부에 또 다른 함수를 하나 더 추가하여 계산 복잡도를 증가시켜 업그레이드된 버전의 더욱 견고한 암호 알고리즘을 구축할 계획이다.

참 고 문 헌

[1] N. Heintze, D. Tygar, "Model Checking Electronic Commerce Protocol," ARPA contract F33615-93-1-1330, Nov. 1995.

[2] A. Froomkin, "The Essential Role of Trusted Third Parties in Electronic Commerce," Ver.1.02 Oct. 1996.

[3] M. Green, "Role of Certificate Authrity in Internet Commerce," 1997.

[4] A. Biryukov, E Kushilevitz, "Improved cryptanalysis of RC5," Advances in Cryptology EUROCRYPT '98, Lecture Notes in Computer Science, Vol.1403, K. Nyberg ed., Springer-Verlag, p 85-99, 1998.

[5] H.Heys, S.Tavares, "Substitution-permutation networks resistant to differential and linear cryptanalysis," J. Cryptology, 9(1), 1996.

[6] M. Naor, O.Reingold, "On the construction of pseudorandom permutations : Luby-Rackoff Revisited," Proceeding of the 29th ACM Symposium on Theory of Computing, 1997.

[7] H. Sun, "Computer and Network Security," Lecture by Rivest of MIT, http://theory.lcs.mit.edu/~rosario/6.915/ lecture6.

[8] J. Daemen, L. Knudsen, V Rijmen, "The Block Cipher Square," Fast Software Encrytion, 4th International Workshop Proceedings, Springer-Verlag, 1997

[9] W. Madryga, "A High Performance Encryption Algorithm," Computer Security : A Global Challenge, Elsevier Science Publishers, 1984.

[10] K. Aoki, K Ohta, "Strict Evaluation of the Maximum Average of Differential Probability and the Maximum Average of Linear Probability," IEICE Transactions Fundamentals of Elections, Communications and Computer Sciences, Vol E80-A, No.1, pp.2-8, 1997.

[11] NIST, "Announcing Development of a Federal Information Standard for Advanced Encryption Standard," Federal Register, Vol.62, No.1, Jan 1997.

[12] E. Biham, A. Shamir, "Differential cryptanalysis of the data encryption standard," Springer-Verlag, 1993.

[13] NIST, 2nd AES Conference : AES Round2 Information, 1999 http://csrc.nist.gov/encryption/aes

[14] IBM Corp., "MARS-a candidate cipher for AES," August, 20 1999.

강 영 구

e-mail : ykkang@duck.snut.ac.kr

1971년 서울대학교 천문기상학과 (이학사)

1980년 Florida Institute of Technology 전산과 (공학석사)

1991년 숭실대학교 대학원 컴퓨터학과 박사과정 수료

1971년~1985년 총무처 전산처리관

1985년~현재 서울산업대학교 전자계산학과 교수

2000년~현재 서울산업대학교 전자계산소장

관심분야 : 소프트웨어 개발론, 알고리즘, 암호학

류 성 열

e-mail : sytlew@computing.soongsil.ac.kr

1996년 아주대학교 컴퓨터학부 (공학박사)

1997년~1998년 George Mason Univ. 교환교수

1981년~현재 숭실대학교 컴퓨터학부 교수

1998년~현재 숭실대학교 정보과학대학원 원장

1998년~현재 숭실대학교 전자계산원 원장

관심분야 : 리엔지니어링, 분산 객체 컴퓨팅, 소프트웨어 재사용, 전자상거래