通信情報保護學會誌
第1卷, 第2號, 1991. 8

DES의 복잡도 분석과 확장방안에 대한 고찰

이상번* · 남길현**

1. 서 론

컴퓨터를 이용하여 각종 정보를 대량으로 처리하는데 있어서 반드시 이들 정보를 안전하게 보호하기 위한 자료보안 문제는 우리나라가 정보사회로 진입하는 시점에서 선결되어야 할 중요한 문제로 인식되고 있다.

일반적으로 높은 수준의 자료보안을 위해서는 자료를 암호화하는 방법이 효율적이라고 생각되며, 암호화 방법중 가장 보편적으로 사용되고 있는 암호알고리즘은 DES(Data Encription Standard)[3]라고 볼수 있다. 70년대에 미국 정부 NBS(National Bureau of Standards)는 공개모집을 통하여 IBM에서 LUCIFER를 근간으로 하여 개발한 DES를 채택하고, 정부의 비밀문건이 아닌 문서를 송수신하는데 사용하고 일반 사용자도 누구나 활용할 수 있도록 하였다.

이 DES는 특히 70년대 말과 80년대 초, 컴퓨터 통신망의 확대와 더불어 급증한 보안사고로 인하여 자료를 필히 암호화하여 사용하여야 한다는 요구와 간단하게 하드웨어나 소프트웨어를 사용하여 구현할 수 있다는 장점으로 미 정부 및 국가표준으로 채택되어 그 사용폭을 급속히 넓혀 왔으며, 국제표준으로도 인정되어 사용하고 있다[2].

그러나 DES는 공표된 이래 암호강도에 대한 많은 연구와 공개비판이 있었다. 이 대부분은 DES 알고리즘에서 사용하고 있는 키(key) 길이가 요구되는 암호강도에 비해 너무 작다는 것과 암호결과를 쉽게 풀수 있도록 하는 어떤 비밀방안(trapdoor)이 포함되지 않았느냐 하는 것이다[2,5,7]. 특히 DES 설계에 미친 NSA(National Security Agency)의 영향력에 대한 의심으로써 본래 IBM에서 제안한 112비트 키 길이를 56비트로 줄이고 특정 S(Substitution)-box를 설계하도록 한 것은 DES를 쉽게 해독할 수 있도록 어떤 조작을 DES내에 포함시키지 않았겠느냐 하는 추측이다.

그러나 많은 DES에 대한 연구 및 공개비판도 DES의 암호강도를 현저히 낮추어 풀수 있는 방법이나 DES의 그 어떤 Trapdoor을 찾아서 발표한 것은 없으나 NSA는 1985년 후반에 1988년 12월 이후부터는 더 이상 DES를 지원하지 않겠다고 발표하였다[1].

일반적으로 암호시스템(Cryptosystem)의 신뢰성을 판단하는 척도로서는 비도(Crypto-complexity)를 사용한다. 비도란 비인가자가 시스템에서 사용하는 키를 찾아내거나 암호문에 해당하는 평문을 찾아내는데 소요되는 노력(시간 및 비용)의 정도를 말한다.

본고에서는 DES의 비도를 복잡도(complexity)의

* 학생회원, 국방대학원
** 종신회원, 국방대학원

표현수단인 시간(time)과 메모리(memory or space) 측면에서 지금까지 발표된 논문들을 중심으로 살펴보고, 비도를 높히는 방안으로서 기본적으로 현재의 DES 알고리즘을 그대로 활용하면서 DES보다 긴 키를 사용할 수 있는 방안을 살펴보고자 한다.

2. DES 알고리즘과 복잡도

가. DES 알고리즘의 특성

1) 기본 알고리즘

DES 알고리즘은 기본적으로 순열(Permutation or transposition), 치환(Substitution) 및 Modulo 2(exclusive OR) 연산을 사용하는 복잡한 암호시스템이면서 평문(plaintext)을 한번에 64비트씩 암호화하는 블럭 암호시스템이다.

알고리즘의 기본구성은 최초 64비트 평문의 각 비트 순서를 바꾸기 위한 초기순열(initial permutation), 순열과 치환으로 이루어진 암호화 과정을 16회 반복하여 수행하는 부분 및 각 비트순서를 다시 초기순열의 역배열로 바꾸는 최종 순열(final permutation) 부분으로 이루어진다.

16회 반복되는 암호화 과정은 반복횟수가 증가할 때마다 암호강도는 높아진다고 볼수 있으며, 암호화 과정은 64비트의 평문을 좌우 두 부분으로 나누어 전 라운드(round)의 오른쪽 부분이 새로운 라운드의 왼쪽부분이 되고, 전 라운드의 오른쪽 부분에 암호함수를 적용하여 전 라운드 왼쪽부분과 XOR(exclusive OR) 연산을 해서 새로운 오른쪽 부분이 된다.

암호화 과정을 수식으로 표현하면 다음과 같다.

$$L_i = R_{i-1} \quad \cdots\cdots (1)$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i) \quad \cdots\cdots (2)$$

(단, i는 라운드 수)

(1), (2)식을 간단히 변형하면 다음과 같다.

(1)식에서 $R_{i-1} = L_i$ ····· (3)

(2)식에서 exclusive OR의 특성을 이용하면

$$L_{i-1} = R_i \oplus f(R_{i-1}, K_i)$$

$$= R_i \oplus f(L_i, K_i) \quad \cdots\cdots (4)$$

따라서 (3), (4)식을 살펴보면 i 라운드의 결과로서 i−1 라운드의 결과를 유도할 수 있다. 따라서 복호화를 위해서는 암호화시 사용된 키 $K_1, K_2, \cdots, K_{16}$를 역순으로 $K_{16}, \cdots, K_2, K_1$으로 적용하여야 함을 알수 있고, 암복호화시 16라운드 후에는 좌우측을 반드시 교환해 주어야 동일한 순서로 올바른 암복호화가 가능함을 알수 있다.

일반적으로 암호 알고리즘의 암호강도는 암호함수 f에 의해서 얻어진다. DES의 암호함수 f에서는 암호강도를 높이기 위해서 전통적인 암호화 기법인 순열법과 치환법을 반복 사용하게 되는데 이는 입력되는 32비트의 일부 비트를 반복 사용하여 48비트로 확장하고 이를 48비트 키와 XOR 연산을 한 다음 6비트씩 S−box를 통해 각각 4비트 값으로 대치시키고 이를 다시 재배열시키는 작업을 하게 된다.

DES 알고리즘에서는 키는 총 48비트의 키가 각 라운드별로 16회 사용된다. 이러한 키는 키 스케쥴(schedule)에 의해서 생성되는데, 키 스케쥴은 로테이션(rotation)과 재배열 기능을 포함함으로써 키 비트값을 암호문 전체에 확산하는 효과를 지니게 된다. 키 스케쥴은 최초 사용자가 제공하는 64비트의 키에서 8, 16, 24, 32, 40, 48, 56 및 64번째 비트를 제거하고 각 키의 위치를 바꾸어 재배열한다. 제거된 8비트는 패리티 비트로 활용된다. 재배열된 56비트의 키를 28비트씩 양분하고 이를 각 라운드별로 1−2 비트씩 left shift시킨 다음, 양분된 각 28비트에서 24비트를 선택하여 다시 재배열한다. 이와같이 결정된 48비트가 암호함수에 사용된다.

2) 알고리즘 특성

DES 알고리즘은 알고리즘 구조상 complementation의 성질을 지니게 되는데, 이는 평문과 키의 보수(complement)를 취하여 구한 암호문은 보수를 취하지 않고 구한 암호문의 보수가 되는 특성이다. 즉 $C = E_k(M)$이면 $\overline{C} = E_{\overline{k}}(\overline{M})$이다. 이러한 complementation은 DES 알고리즘에 쓰이는 두 XOR 연

산에 기인한다. 즉 암호함수에서 평문의 보수와 키의 보수가 입력되면, 두 값의 XOR 연산은 보수를 하기 전의 XOR 연산결과와 동일하게 되고, 따라서 S-box의 입력에는 변함이 없이 동일하므로 S-box 출력에도 변함이 없으며, S-box 출력이 평문보수와 XOR 연산할 때 보수가 되어 버리기 때문에 전체 결과 값은 보수가 되버리는 것이다.

이러한 complementation의 성질은 암호공격시 Chosen-plaintext 공격하에서는 전체 비도를 반으로 줄이는 효과를 가져온다[5].

또 하나의 특별한 성질은 키의 생성에서 weak key와 semi-weak key를 갖게 되는 것이다. weak key는 키가 모두 0이나 1로 이루어져 키 스케쥴에서 left shift나 right shift, permutation를 하더라도 16라운드의 키가 $k(1)=k(2)=\cdots=k(16)$되어 결국 암복호화가 똑같게 된다. 따라서 이 weak key를 사용하여 암호화하고, 또 똑같은 키를 사용 재 암호화하면 평문이 구해지는 단점을 지니게 된다.

semi-weak key는 $K_i^*=K_{17-i}(0<i<17)$로서 표현할 수 있다. 즉 semi-weak key는 쌍으로 존재하게 되는데 쌍중에서 하나의 키가 생성하는 16개 키의 각각은 다른 키가 생성하는 16개 키의 역순과 일치하여 하나의 키로 암호화하고 다른 키로 암호화하면 평문이 바로 구해진다. 이러한 semi-weak key는 키 비트중에서 일부 키가 서로 동일할 때 일어나며, 16개 키의 각각은 단지 두종류의 키로만 구성되는 특징이 있다.

또 DES의 키 중에서는 weak key나 semi-weak key 외에도 16개 키가 네 종류의 키로서 구성되는 특성도 있다.

DES 알고리즘 구성부분중 초기 및 최종 순열은 암호강도에는 아무런 도움을 주지 못하며, 오히려 DES를 소프트웨어로 구현시 수행속도만 20% 정도 증가시키는 단점을 지니고 있기 때문에 보통 비도를 분석평가시는 이들 부분을 배제하고 고려한다[5,9].

DES가 비록 공개적으로 발표되었다 하더라도 알고리즘 자체에 쓰인 테이블(table)의 선택에 있어서는 전혀 IBM이나 NSA에서 설명이 없고 비밀로 유지하기 때문에 의심을 가지게 한다. 즉 DES에 사용된 테이블을 간단히 수정만 하더라도 어떤 trapdoor를 내포시킬 수 있기 때문이다. 이러한 면도 DES의 비도가 많은 논란의 대상이 되는 단점이 되는 것이다.

나. DES 알고리즘의 복잡도 분석

1) 가용자료에 의한 암호공격 방법의 분류

일반적으로 암호분석을 통해 암호문을 해독하기 위한 기본적인 공격방법은 가용한 자료의 종류에 따라 세가지로 분류할 수 있다.

첫째는 Ciphertext-only 공격으로서 암호문을 만들어낸 기본적인 요건들을 알고 있더라도 단지 얻어진 암호문만을 가지고 키를 찾아내는 공격방법이다.

둘째는 Known-plaintext 공격으로서 필요한 만큼의 암호문과 대응하는 평문의 쌍들을 알고서 비밀키를 찾아내는 공격방법이다. 평문에 해당하는 암호문이 얻어질 때까지 가능한 모든 키를 적용해 보는 가장 간단한 방법도 이 공격법에 속하게 된다.

셋째는 Chosen-plaintext 공격으로서 암호공격자가 특정한 평문을 자유롭게 선택할 수 있고 거기에 대응하는 암호문을 획득할 수가 있는 상황하에서 비밀키를 찾아내는 공격법이다. 따라서 암호공격자에게는 가장 좋은 공격방법이 된다.

위와 같은 공격방법을 기반으로 DES를 공격하기 위한 다양한 연구들이 DES의 약점을 파헤치려는 노력과 병행하여 지속적으로 진행되어 왔다.

2) 기존의 공격방법 고찰

- exhaustive 공격

일반적으로 exhaustive search는 키와 메세지에 대한 exhaustive search로서 구분할 수 있다.

미 스탠포드 대학의 Hellman과 Diffie는 DES의 56비트 키로서는 exhaustive search(혹은 brute force 기법)에 너무 약하다고 주장하였다[5,7,9]. 즉 동일한 키를 사용한 평문과 암호문의 쌍들을 갖고 있으면 Known-plaintext 공격방법을 적용하여 암호화에 가능한 키 2^{56}개를 전부 적용해 봄으로써 해당 평문과 암호문의 일치여부로 사용된 암호키를 쉽게 찾을 수가 있다.

즉 2^{56}=약 10^{17}개의 키로서 한개의 키를 암호화 하는데 1us가 소요된다고 할때 약 10^{6}일이 소요되고, DES는 한개의 칩으로 만들수 있기 때문에 많은 칩을 병렬로 사용하여 처리하면 exhaustive search가 가능하게 되며, 백만개의 DES 칩을 연결하여 공격시는 하루가 소요되고 평균 12시간내에 해결할 수가 있다는 결론이 나온다. 따라서 기술 발전속도를 고려할때 10년내에 DES는 생명력을 잃을 것이며, Known-plaintext 공격에 강해지기 위해서는 키 길이를 늘리는 것이 최선의 방법이라고 주장하였다.

그들은 또 키 길이가 112비트만 되더라도 충분한 비도를 지닐수 있는데, 키 길이의 증가는 NSA가 반대하며 이는 풀수 없는 암호시스템을 원하지 않는 다분히 정치적인 목적이라고 주장하였다.

● time-memory tradeoff 탐색방법

Hellman은 그후 발표된 논문에서 컴퓨터의 기억용량이 더 많이 요구되고, 사전 계산시간(pre-computing time)이 충분히 필요하지만 exhaustive search보다 키 탐색시간이 덜 걸리는 time-memory tradeoff 방법을 주장하였다[6,7].

일반적으로 time-memory tradeoff 탐색방법은 해결가능한 N개의 문제는 N=TM(T : time, M : words of memory)으로 정의할 수가 있고, 이 time과 memory는 exhaustive search와 테이블 lookup의 탐색방법으로 해결되는 것이다.

테이블 lookup 방법은 사전에 선택한 평문에 가능한 키 전부를 적용하여 암호화시킨 암호문을 메모리에 저장해 놓고 어떠한 암호문의 키를 구하고자 할때는 저장된 암호문을 탐색하여 이에 해당하는 해당 키를 찾아내는 방법이다. 따라서 테이블 lookup은 키의 가능한 갯수만큼 메모리와 사전 계산시간 및 탐색시간이 필요하게 되며, 결국 time-memory tradeoff도 이 요소들이 필요하게 된다.

Hellman이 주장한 time-memory tradeoff 방법은 키가 이루는 함수를 $f(K)=R(E_k(M_0))$ (여기서 M_0 : 특정 평문, R : 64비트 블럭을 56비트 블럭으로 변환하는 reduction 함수)라고 정의할 때, K를 랜덤하고 균일하게 선택하여 f(K)에 대한 값을 테이블에 저장해 놓고 선택한 암호문 C_0($C_0=E_k(M_0)$)에 대한 $R(C_0)$값과 일치하는 f(K)를 테이블에서 찾아 K를 구하는 방법으로 요약된다.

Hellman은 이러한 time-memory tradeoff 방법으로 메모리의 크기와 탐색시간이 $N^{2/3}$ 필요한 탐색방법을 제시함으로서, exhaustive search나 테이블 lookup보다 효과적인 공격방법이 될수 있다고 주장하였다. 단 이러한 방법을 위해서는 테이블을 형성하기 위한 수년간의 사전 계산시간이 요구되는 점과, 또한 Known-plaintext 공격에서는 chaining 기법을 사용한 암호문에는 적용 불가하다는 단점도 있어서 실제 적용상에는 문제점이 있다고 볼수 있다.

더욱이 time-memory tradeoff의 사전 계산시간을 배제하고 생각하더라도 키 길이가 64 비트 이상이면 거의 실용화가 불가능하게 된다.

● differential cryptanalysis

Hellman 이후 다양한 공격방법이 여러 연구가들에 의해 발표되었지만 DES를 아주 쉽게 short cut하여 공격하는 방법이 발표된 바는 없는데, CRYPTO '90에서 이스라엘의 E. Biham과 A. Shamir는 새로운 DES-like 암호시스템의 공격법을 제시하였다[12,13].

이 새로운 공격방법은 DES의 형태와 같이 암호화 기본구조를 반복하는 반복 암호시스템(iterated cryptosystem)에 적용되는 공격형태로서 PC를 사용하여 6라운드의 DES는 0.3초내에, 8라운드의 DES는 2분내에 풀수 있으며, 15라운드의 DES도 2^{52}의 연산이 소요되어 exhaustive search보다 더욱 빨리 암호문을 해독할 수 있다고 주장하였다.

이러한 differential cryptanalysis는 Chosen-plaintext 공격을 사용한다. 사용되는 기법은 살펴보면 DES와 같이 암호화 과정이 반복 수행되는 반복 암호시스템에서 i-1 라운드에 특정한 차이 A가 나는 두 평문이 적용되었을때 이들의 결과인 i 라운드의 암호문은 특정한 차이 B가 나는 효과를 이용하여 이 (A, B)의 쌍으로서 i 라운드를 정의하고, 다음과 같은 절차를 이용하여 키를 구한다. DES에서 (A, B)는 결국 S-box의 입력이 특정 출력이 발생될 수 있는 높은 확률(high probability)로서 정의할 수가 있기 때문이다.

1) i−1 라운드에서 높은 확률을 갖는 difference (A, B)를 구한다.

2) 평문 X를 랜덤 균일하게 선택하여 X와 A의 difference가 있는 X*를 계산한다.

X와 X*와 암호화한 결과인 Y(i)와 Y*(i)로부터 B만큼의 difference가 계산될 수 있는 모든 가능한 키인 K(i)를 구한다. 여러 K(i)가 구해질 수가 있는데 동일한 K(i)가 구해질 때마다 횟수를 count 한다.

3) 위의 과정을 반복하여 가장 자주 구해지는 키를 선택하여 이것이 실제의 K(i)인가를 확인한다.

그러나 이 공격방법도 16라운드의 DES는 오히려 exhaustive search 방법보다 증가한 2^{58}개의 연산을 필요로 한다.

3. DES를 이용하여 보안성을 높이는 방안

가. 일반적인 고려사항

DES를 이용하여 보안성을 더욱 높이도록 하는 방안은 여러가지로 고려될 수 있으며, 일반적으로 현 DES를 가장 안전한 방법으로 사용하는 방법과 DES 알고리즘 일부를 수정해서 암호공격에 더욱 강하게 하여 사용하는 방법을 생각할 수가 있다[9].

일반적인 암호시스템에서와 마찬가지로 DES를 안전하게 사용하기 위해서는 키 선택이 아주 중요하다.

키 선택시 weak key나 semi−weak key를 배제해야 되고, 랜덤하게 선택한 키를 사용해야 한다. 또한 키를 대문자 알파벳만을 이용하는 등 제한된 범위내에서 선택한다면 이는 효과적인 키 길이를 줄이는 결과를 초래하게 되므로 키 선택은 확실하고 안전성 있는 키 생성 알고리즘을 이용하여 이루어져야 할 것이다.

키 선택뿐만 아니라 키의 분배면에서도 안전한 시스템을 사용하여야 한다. 키 분배에서 가장 좋은 방법은 공개키(public key) 암호시스템을 이용하는 것인데 다만 공개키 암호시스템의 특성인 지수연산으로 인하여 속도가 느리다는 단점을 배제하기 위해서는 미리 사용될 prime number를 계산하여 작성한 테이블을 놓고 랜덤하게 선택해서 사용할 수가 있다. 또한 동일한 키를 장시간 사용하기 보다는 one−time session 키를 사용하는 편이나 auto-key(running key)를 적용하여 매 암호화마다 다른 키를 생성하는 것도 좋을 것이다.

블럭사이퍼(block cipher)인 DES는 평문을 암호화하면 64비트 즉, 8바이트마다 평문의 특성이 암호문에 그대로 드러나게 되므로 미리 평문을 압축하여 불필요한 redundancy를 제거하여 암호화하거나 안전한 운용 모드를 이용하여 암호화하는 것이 바람직스럽다.

DES의 운영모드는 CFB(cipher feedback) 모드, CBC(cipher block chaining) 모드 및 OBFB(output−block feedback) 모드 등이 있다[4,11].

나. DES의 다중사용

1) double encipherment

이중 암호화방법(double encipherment)은 평문을 다른 키로 두번 중복 암호화하는 것으로서 키의 크기를 112비트로 늘리는 효과를 가져와 암호공격에 강하게 할수 있다고 생각될 수 있다[5,7].

그러나 이런 이중 암호화 방법은 closed 문제와 meet in the middle 공격으로 암호강도는 2^{112}에 미치지 못한다.

왜냐하면 closed로서 하나의 키를 사용하여 평문을 암호화하고 다른 키를 사용하여 다시 암호화하면 그 결과는 하나의 다른 어떤 특정키를 사용하여 암호화한 것과 동일하지 않느냐 하는 것이다.

즉 A를 암호화하여 F를 얻고 F를 다시 암호화하여 L을 얻었다면 결국 두번 암호화한 결과는 A를 어떤 다른 키를 사용하여 L로 직접 암호화 한것과 같다는 의미가 되기 때문이다.

또 meet in the middle 공격에서는 만약 첫번의 암호화 과정을 $M=S_{k1}(P)$이라고 할때 암호문 C는 $C=S_{k2}(M)$이 되고, 중간값 M은 $M=S_{k2}^{-1}(C)$가 된다. 따라서 P, C를 알고 있는 Known−plaintext 공격에서는 가능한 키 2^{56}개로서 P를 암호화하고 암호문 C를 복호화하여 이들 연산의 결과인 두개의

데이타 집합중에서 결과치가 동일한 것의 키는 바로 이중 암호화에서 사용한 키가 되는 것이다. 따라서 이러한 공격방법은 탐색시 binary search를 사용했을 때의 탐색시간인 NlogN과 2^{56}의 메모리 량 및 연산을 필요로 하게 됨으로 이중 암호화도 비도를 높이는 데는 별 효과가 없는 셈이 된다.

2) triple encipherment

이중 암호화보다 암호공격에 강한 다중 암호화 방식(multiple-encipherment)으로서는 이중 암호화와 같이 두개의 키를 이용하거나 또는 각기 틀린 세개의 키를 이용하여 세번 중복 암호화하는 방법을 생각할 수 있다[5,7,10].

두개의 키를 이용한 삼중 암호화 방법은 $C=S_{k1}(S_{k2}^{-1}(S_{k1}(P)))$로 표현할 수 있는데 이러한 암호화 방법은 이중 암호화에 대한 meet in the middle 공격을 피할수 있는 방법으로 생각되나 Chosen-plaintext 공격에서는 이중 암호화와 동일한 메모리 용량이나 연산정도를 필요로 하게 된다.

즉 평문 P와 암호문 C에서 유도되는 중간값은 M_1, M_2라고 할때

$$M_1=S_{k1}(P) \quad \cdots\cdots (1)$$

$$M_2=S_{k2}^{-1}(S_{k1}(P))=S_{k2}^{-1}(M_1)=S_{k1}^{-1}(C) \quad \cdots\cdots (2)$$

로 표시된다.

따라서 (1)에서 특정한 값 M_1을 선택하면 이 M_1이 유도되는 키 K_1은 exhaustive search로서 계산될 수 있고, 다시 테이블 lookup을 사용해서 (2)에서 K_2를 구할수 있기 때문에 (1)에서의 2^{56} 연산과 (2)에서 테이블 lookup의 2^{56} 메모리와 탐색시간만 주어지면 해당 키를 구하게 되므로, 결국 두개의 키를 이용한 삼중 암호화 방법도 이중 암호화와 비슷한 비도밖에 되지 않는다.

각기 틀린 세개의 키로 암호화하는 방법은 $C=S_{k1}(S_{k2}^{-1}(S_{k3}(P)))$로 표현할 수 있는데 이러한 삼중 암호화 방법은 meet in the middle 공격때문에 $(2^{56})^2=2^{112}$의 암호강도를 갖게 되는 것으로 판단된다.

다. DES의 부분 수정

DES의 소프트웨어를 일부 수정해서 암호공격에 강하도록 하는 개선방법도 다양하게 생각할 수 있다[9].

DES 내부에 숨어있을지도 모를 trapdoor를 방지하기 위하여 S-box나 키 스케쥴을 랜덤하게 할수 있다. 이러한 방법은 키에 종속적인 S-box나 키 스케쥴을 생각할 수 있는데 이는 S-box의 테이블을 고정시킬 것이 아니라 사용되는 키에 따라서 S-box 테이블을 새로 작성하여 사용하는 방안과 키 스케쥴에서 키를 단순히 1-2 비트씩 shift시키는 고정된 방법보다는 키에 의존한 랜덤 number를 발생하여 키 rotation으로서 사용할 수도 있다.

단 이러한 수정이 DES의 암호강도를 급격히 낮출수도 있으므로 신중한 판단이 요구되며[12], 하드웨어의 구현이 힘들기 때문에 보다 효과적인 방법이 요구된다.

라. 새로운 알고리즘의 개발

NSA의 DES에 대한 불인정에 대한 공표에 따라 DES의 대체 알고리즘 개발이 필요하게 되었고, 다수의 새로운 알고리즘이 개발되었다.

이중 개발된 알고리즘의 하나가 FEAL(Fast Data Encipherment Algorithm)이다[15]. FEAL은 8비트 마이크로프로세서상에서도 효과적으로 구현할 수 있도록 설계되어 PC 수준에서도 암호화하기 쉽도록 되었는데, 기본적으로 64비트를 한번에 암호화 한다. FEAL은 DES와 유사하게 암호과정이 반복되는 기본구조를 가지면서 FEAL의 두개 version인 FEAL-4는 총 4라운드의 암호과정을 거치게 되며 FEAL-8은 총 8라운드의 암호과정을 수행하게 된다.

이때 새로운 암호함수와 키 스케쥴 알고리즘을 사용하고 있는데 암호함수는 비트 roration과 addition으로서 이중 addition 연산은 선형(linear)적인 결과를 가져오게 되므로 FEAL의 암호 비도는 비트 rotation에 기인한다.

이러한 암호 알고리즘은 differential cryptanaly-

sis와 같은 암호 공격방법에는 상당히 약한 것으로 평가되었다[12]. 즉 FEAL-4는 8개의 암호문과 1개의 평문으로 해결 가능하고 FEAL-8은 2000개 이하의 암호문만 있으면 풀수 있다고 하였다. 이에 대한 반발로 FEAL 확장방안인 FEAL-N과 128비트를 사용하는 FEAL-NX이 발표되었는데 이러한 FEAL의 확장도 exhaustive search의 암호비도보다도 약한 비도를 갖는다고 differential cryptanalysist인 E. Biham과 A. Shamir는 주장하였다.

EUROCRYPT '90에서 Lai와 Massay는 총 8라운드로서 수행되는 블럭사이퍼(block cipher)인 새로운 암호 알고리즘 PES(Proposed Encryption Standard)를 발표하였다[14].

이 PES는 기본적으로 세가지의 연산을 사용한다. 하나는 16비트의 서브키(subkey)와 Mudulo 2(exclusive OR) 연산이며, 또 하나는 16비트의 서브키와 addition Modulo 2^{16}이며 다른 하나는 모두 영(zero)으로 구성된 서브키와의 multiplication modulo $2^{16}+1$이다.

이러한 PES는 differential cryptanalysis에 강한 것으로 분석되었는데 2라운드의 PES는 10라운드의 DES만큼이나 강하고 4라운드의 PES는 differential cryptanalysis로서는 해결 불가능하다고 발표하였다[13].

4. DES의 알고리즘 확장방안

가. 고려사항

지금까지 살펴본 바와 같이 다양한 DES의 암호공격방법이나 DES의 확장방안에서 공통적인 사항은 56비트의 키 길이로서는 관용키 암호 알고리즘에서 충분한 암호비도를 제공하지 못하기 때문에 키 길이를 늘리는 것이 제반 암호공격에 좋은 방책이라는 것을 지적하고 있다.

왜냐하면 키 길이가 충분히 길다고 하는 것은 exhaustive search의 공격방법에 효과적으로 대응할 수 있고, time-memory tradeoff도 키 길이가 늘어난 만큼 시간과 메모리를 더 요구하기 때문에 효과적인 대응방법이 되는 것이다.

다만 키 길이가 길어지면 긴 키를 생성하고 분배하고 보관하기 위해서는 그만큼 시간과 공간을 필요로 하기 때문에 키 관리가 어려워지는 면도 고려해야 할 것이다. 또한 처리속도면도 고려해야 할 필수사항이 된다. 그러나 안전해야 한다는 암호 알고리즘의 기본 요구를 고려할 때, 암호공격에 대비한 충분한 키 길이의 제공은 필히 확보되어야 할 사항인 것이다.

본고에서는 암호공격에 충분히 대응하기 위해서 키 길이가 112비트가 되는 DES 암호 알고리즘의 변형방안을 고려해 보기로 한다. 단 이를 위해 DES의 기본적인 함수는 그대로 이용하면서 간단하게 구조를 변경하여 키 길이가 112비트가 요구되도록 암호 알고리즘을 변형하는 방안을 생각해 보기로 한다.

나. 키 길이 증가를 위한 DES 알고리즘 확장방안

일반적으로 좋은 암호 알고리즘을 개발하기 위해서는 첫째 암복호화를 위한 암호 알고리즘의 기본구성을 어떻게 하느냐하는 문제와 둘째 암복호화 알고리즘의 기본구성에서 사용될 암호함수 즉, S-box나 P-box 등을 어떻게 설계하느냐 하는 문제가 아주 중요하다.

즉 개발한 알고리즘의 기본구성과 선택되어 사용될 S-box나 P-box가 암호비도를 가장 높일수 있도록 설계되어야 하기 때문이다.

따라서 본고에서 제시하고자 하는 키 길이 증가를 위한 알고리즘 확장방안이 DES의 기본함수를 그대로 적용한다 하더라도 구조의 변경이 오히려 암호비도를 약화시키지 않을까 하는 의심도 있지만 자세한 분석은 본고에서 고려하지 않고 단지 변형방법의 한 예로서 생각해 보기로 한다.

키 길이 증가를 위한 알고리즘의 확장방안의 기본구상은 두개의 DES 알고리즘을 합쳐 놓아 총 112비트의 키가 요구되도록 구조를 조정하는 것이다. 즉 128비트의 평문을 네개의 32비트로 나누어 좌우 두개를 DES 알고리즘에 적용하고 좌우측의 결과를 교환함으로써 키와 평문이 암호문 전체에 영향을 미치도록 하는 것이다.

간단하게 생각할 수 있는 방법은 좌우 두개의 64비트의 평문을 각각 DES 알고리즘에 적용하고, 일정 라운드가 끝나면 결과치를 서로 교환하여 또 다시 남은 라운드를 수행하는 것이다. 이때 암복호화가 동일한 알고리즘이 되도록 하기 위해서는 일정 라운드가 끝나면 좌우 두 부분은 먼저 각자의 두 32비트의 결과값을 교환한 후에 전체적으로 교환하면 된다.

더욱이 키와 평문이 암호문에 충분히 뒤섞이는 5라운드 후에 좌우측을 교환하는 것을 반복하면 결과의 암호문은 112비트의 키와 128비트의 평문이 완전히 뒤섞어는 결과를 가져올 수 있을 것이다.

그러나 이와 같은 방법을 한 라운드마다 적용하여 좌우를 교환시는 일부 평문이 암호문에 그대로 존재해 버리는 단점과 단순히 작은 라운드의 DES를 모두 합쳐 놓은 형태가 되기 때문에 암호공격에는 약한 단점을 지닌다.

또 이와 다른 방안은 그림 1과 같이 라운드마다 좌우 64비트중 암호함수를 적용한 32비트와 암호함수를 적용하지 않은 32비트를 XOR 연산을 수행하는 방법과 이들을 그림 2와 같이 라운드마다 좌우를 바꾸어 주는 방법도 생각할 수 있는데 이들

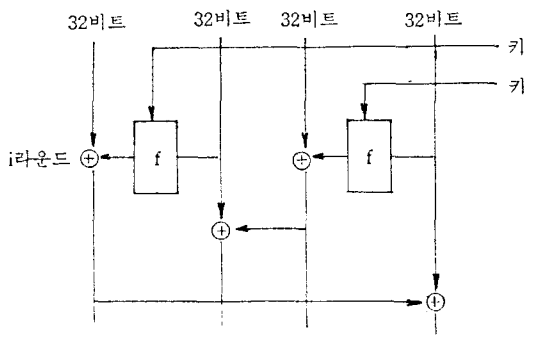


그림 1. 키길이 확장을 위한 변형방안 1

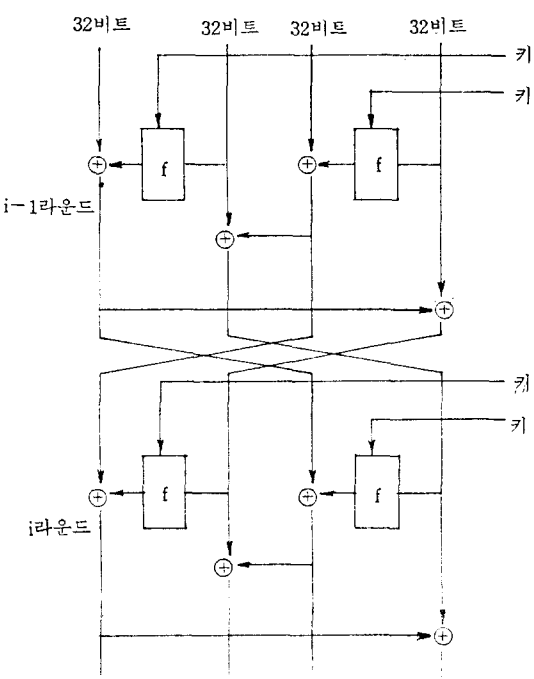


그림 2. 키길이 확장을 위한 변형방안 2

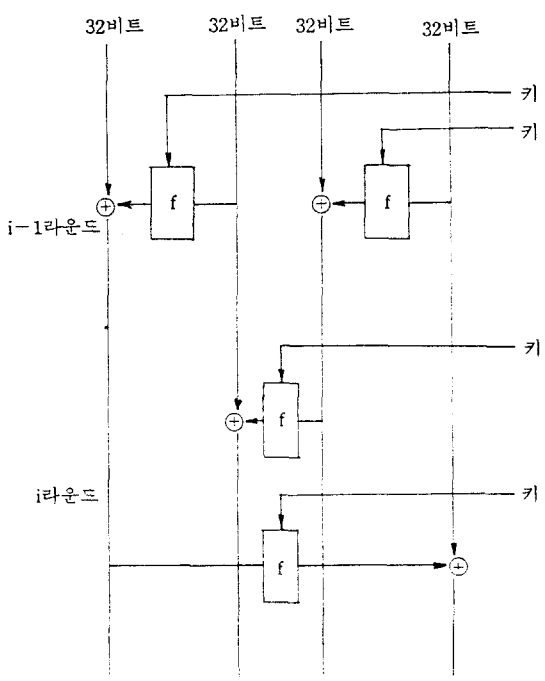


그림 3. 키길이 확장을 위한 변형방안 3

방법은 평문에 하나의 비트만 수정될 경우에 전체 암호문이 바뀌는 확산효과가 적다는 것이 단점이다.

이러한 단점을 보완하기 위해서는 그림 3과 같이 암호함수를 적용하는 방법을 생각할 수 있다. 이와같은 기본구조 변경은 단순한 exhaustive search상에서는 전체 키인 112비트를 알아야 하기 때문에 암호비도를 증가시킬 수 있다.

이와 같이 변형된 확장방안에 대한 다른 공격에 대해서는 아직 심도있는 분석이 이루어지지 않고 있으나 DES의 키 길이를 증가시키는 방안으로 고려할 수 있으리라 생각된다.

5. 결 론

보안목표에 따라서 다양한 보안시스템의 구축이 이루어지겠지만 데이타의 암호화 방법은 그 활용성에 따라 계속 증가하게 될 것이며, 따라서 DES와 같은 기존의 암호 알고리즘을 보다 철저히 연구분석하여 다양한 암호공격에 대응할 수 있는 암호 알고리즘의 개발이 이루어져야 할 것이다.

기존 DES를 이용하거나 새로운 암호 알고리즘을 개발하기 위해서는 기본적인 암호함수를 개발하기 위한 지식의 축적이 무엇보다도 중요하다 하겠다.

본고에서는 가장 보편화되어 있는 DES의 비도에 관하여 여러가지 분석방법들을 고찰해 보고 기존 알고리즘을 활용하여 더욱 긴 키를 이용할 수 있는 확장방안을 제안해 보았다. 제안된 확장방안에 대한 좀더 심도있는 분석이 이루어진다면 기존의 DES를 이용해서 높은 비도를 얻을수 있는 효율적인 방안이 도출될 수 있으리라 기대된다.

참 고 문 헌

1. "The DES Revisited," Computer & Security, No. 4, 1986.

2. T. Goeltz, "Why not DES ?," Computer & Security, No. 5, pp.24-27, 1986.

3. "Data Encryption Standard," FIPS Pub. 46, NSA, U.S. Dep. of Commerce, Washington, DC, Jan. 1977.

4. C.R. Abbruscato, "Data Encryption Equipment," IEEE Com. Magazine, Vol. 22, No. 9, Sep. 1984.

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

6. M.E. Hellman, "A Cryptanlytic Time-Memory Tradeoff," IEEE Trans. of Info. theory, Vol. IT-26, No. 4, pp.401-406, July, 1980.

7. M.E. Hellman, "DES Will be Totally Insecure Within Ten Years," IEEE Spectrum, Vol. 16, No. 7, pp.32-39, July, 1979.

8. R.C. Merkle and M.E. Hellman, "On the Security of Multiple Encryption," Comm. of ACM, Vol. 24, No. 7, pp.465-467, July, 1981.

9. J.M. Carroll, "Strategies for Extending the Useful Lifetime of DES," Computer & Security, No. 4, pp.300-313, 1987.

10. D. Coppersmith, "Cryptography," IBM J. Res. Develop., Vol. 31, No. 2, pp.244-248, Mar. 1987.

11. P.W. Sanders and V. Varadharajan, "Secure Communications between Microcomputer Systems," Security, Vol. 6, No. 5, pp.245-252, Oct. 1983.

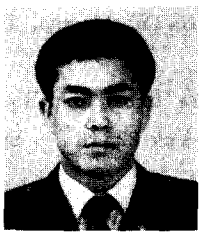
12. E. Biham and A. Shamir, "Differential Cryptanalysis of DES-like Cryptosystems, " Abstracts of CRYPTO '90.

13. X. Lai and J.L. Massey, "Markov Ciphers and Differential Cryptanalysis," Abstracts of EUROCRYPT '91.

14. X. Lai and J.L. Massey, "A Proposal for a New Block Encryption Standad," Proc. of EUROCEYPT '90.

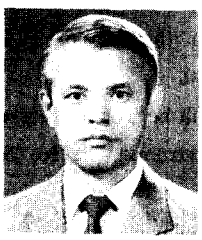
15. A. Shimizu and S. Miyaguchi, "Fast Data Encryption Algorithm," Abstracts of EUROCRYPT '87.

□ 著者紹介



이 상 번(正會員)

1981年 空軍士官學校 卒業
現 國防大學院 碩士課程 電算科 在學



남 길 현(正會員)

陸軍士官學校 卒
서울工大 土木科 卒
美海軍 大學院(電算學 碩士)
위스콘신(메디슨) 州立大(電算學 碩士)
루이지아나州立大(電算學 博士)
陸軍士官學校 教授部 勤務, 現在 國防大學院 副教授