암호와 복호가 동일한 변형 AES

(Modified AES having same structure in encryption and decryption)

조 경 연*, 송 홍 복**
(Gyeong-Yeon Cho and Hong-Bok Song)

요 약 블록 암호는 Feistel 구조와 SPN 구조로 나눌 수 있다. Feistel 구조는 암호 및 복호 알고리즘이 같은 구조이고, SPN 구조는 암호 및 복호 알고리즘이 다르다. 본 논문에서는 암호와 복호 과정이 동일한 SPN 구조 블록 암호 알고리즘을 제안한다. 즉 SPN 구조 전체를 짝수인 N 라운드로 구성하고 1 라운드부터 N/2 라운드까지는 정함수를 적용하고, (N/2)+1 라운드부터 N 라운드까지는 역함수를 적용한다. 또한 정함수단과 역함수단 사이에 대칭 블록을 구성하는 대칭단을 삽입한다. 본 논문에서 정함수로는 AES의 암호 알고리즘을, 역함수로는 AES의 복호 알고리즘을 사용하고, 대칭단은 간단한 행렬식과 라운드 키 합산으로 구성한다. 본 논문에서 제안한 암호와 복호가 동일한 변형 AES는 하드웨어 구성이 간단한 장점을 가지므로 제한적 하드웨어 및 소프트웨어 환경인 스마트카드와 전자 칩이 내장된 태그와 같은 RFID 환경에서 안전하고 효율적인 암호 시스템을 구성할 수 있다.

핵심주제어 : AES, SPN, 암호, 복호. 대칭단.

Abstract Feistel and SPN are the two main structures in a block cipher. Feistel is a symmetric structure which has the same structure in encryption and decryption, but SPN is not a symmetric structure. In this paper, we propose a SPN which has a symmetric structure in encryption and decryption. The whole operations of proposed algorithm are composed of the even numbers of N rounds where the first half of them, 1 to N/2 round, applies a right function and the last half of them, (N+1)/2 to N round, employs an inverse function. And a symmetry layer is located in between the right function layer and the inverse function layer. In this paper, AES encryption and decryption function are selected for the right function and the inverse function, respectively. The symmetric layer is composed with simple matrix and round key addition. Due to the simplicity of the symmetric SPN structure in hardware implementation, the proposed modified AES is believed to construct a safe and efficient cipher in Smart Card and RFID environments where electronic chips are built in.

Key Words : AES, SPN(Substitution Permutation Network), encryption, decryption, symmetric layer.

1. 서 론

비밀키 블록 암호는 암호 알고리즘 E에 비밀키 K를 적용하여 평문 P로부터 암호문 $C = E_K(P)$를 생성하며, 복호는 복호 알고리즘 D에 동일한 비밀키 K를 적용하여 암호문 C로부터 평문 $P = D_K(C)$를 얻는다. 이러한 비밀키 블록 암호의 암호 및 복호 알고리즘은 암호 및 복호 함수를 여러 라운드 반복 수행

* 부경대학교 공과대학 IT융합응용공학과 교수, 제1저자
** 동의대학교 공과대학 전자공학과 교수, 교신저자

하는 구조이다. 비밀키 블록 암호는 라운드 함수의 구조에 따라서 Feistel 구조[1]와 SPN(Substitution Permutation Network) 구조로 나눌 수 있다.

Feistel 구조의 라운드 함수는 입력을 둘로 나누고, 나누어진 반쪽에 라운드 키를 적용하여 비선형변환을 하고, 이를 나머지 반쪽과 배타적 논리합 즉 XOR 연산을 수행한 후에 서로 치환한다. 이러한 Feistel 구조는 암호 알고리즘과 복호 알고리즘이 동일하며, 적용하는 라운드 키의 순서만이 상호 역순이 된다. 각 라운드 함수에 적용하는 키를 라운드 키라고 하며, 이는 마스터 키로부터 생성한다. Feistel 구조는 암호와 복호 알고리즘이 동일하므로 하드웨어 및 소프트웨어 구현이 쉬운 장점이 있는 반면에 한 라운드에서 입력의 반만이 비선형변환되므로 충분한 안전성을 얻기 위해서는 라운드 수가 증가되며, 따라서 하드웨어 구현시 동작 속도가 느려지는 단점이 있다.

SPN 구조는 C. E Shannon의 혼돈(Confusion)과 확산(Diffusion)[2] 이론을 바탕으로 하였다. SPN 구조에서의 암호 라운드 함수는 키합산층(Key addition layer)과 혼돈을 수행하는 치환층(Substitution layer) S 및 확산층(Permutation layer) P의 세 단계로 구성된다. 복호 라운드 함수는 역확산층(Inverse Permutation layer) P^{-1}과 역치환층(Inverse Substitution layer) S^{-1} 및 키합산층(Key addition layer)의 세 단계로 구성한다. SPN 구조는 암호와 복호 알고리즘이 다르므로 구현이 복잡해지지만 각 라운드에서 입력이 전부 비선형변환되므로 라운드의 수가 적어진다. 따라서 하드웨어의 동작 속도가 빠른 장점을 가진다.

1970년대에 개발되어 산업계 표준으로 사용되던 DES(Data Encryption Standard)[3]는 Feistel 구조인데 안전성에 문제가 있어 미국을 비롯한 유럽 및 선진국들은 자국의 표준 블록 암호 개발에 주력하고 있다. 미국은 AES(Advanced Encryption Standard) 프로젝트[4]를 수행하여 SPN 구조인 Rijndael 알고리즘[5]을 표준 블록 암호 알고리즘으로 선정하였으며, 우리나라도 자체 개발한 SEED[6]와 ARIA[7]를 사용하고 있다.

SEED는 Feistel 구조이며, ARIA는 암호 및 복호 알고리즘이 동일한 SPN 구조이다. ARIA는 치환층과 역치환층을 네 가지 S-박스, $S1$, $S2$, $S1^{-1}$, $S2^{-1}$를 적절하게 조합하여 동일하게 구성하였으며, 치환층을 대칭변환함수(involution function)로 구성하여 치환층과 역치환층이 동일하다. ARIA의 치환층은 바이트 단위의 16 X 16 행렬식으로 구성되어 있으므로 소프트웨어로 구현하면 동작 속도가 AES에 비하여 크게 느려지는 문제점이 있다.

본 논문에서는 암호와 복호 과정이 상이한 SPN 구조 블록 암호 알고리즘을 암복호 과정이 동일한 SPN 구조로 변형시키는 방식을 제안한다. 즉, 변형한 블록 암호 알고리즘은 짝수 N 라운드로 구성하고, 1 라운드부터 N/2 라운드까지의 전반부 라운드 함수는 키합산층(Key addition layer)과 치환층(Substitution layer) S 및 확산층(Permutation layer) P의 세 단계로 구성하고 이를 정함수단이라 가칭한다. 후반부 (N/2) + 1 라운드부터 N 라운드까지는 역확산층(Inverse Permutation layer) P^{-1}과 역치환층(Inverse Substitution layer) S^{-1} 및 키합산층(Key addition layer)의 세 단계로 구성하고 이를 역함수단이라 가칭한다.

또한 정함수단과 역함수단 사이에 규칙적인 라운드의 반복을 피하기 위해 간단한 행렬식과 라운드 키의 배타적 논리합으로 구성한 대칭단(Symmetry layer)을 삽입한다.

즉, 정함수단과 대칭단 및 역함수단으로 암호 알고리즘을 구성하고, 암호와 복호시에 사용하는 라운드 키를 역순으로 적용하면 암복호 과정이 동일한 SPN 블록 암호 알고리즘을 구성할 수 있다.

본 논문에서는 정함수단에는 안전성이 증명된 AES의 암호 알고리즘을, 역함수단에는 AES의 복호 알고리즘을 사용하며, 이를 변형 AES라 가칭한다.

본 논문에서 제안하는 변형 AES는 암호와 복호 알고리즘이 동일하며, 적용하는 라운드 키는 상호 역순으로 되어있다. 또한 기존의 AES 하드웨어에 대칭단만을 삽입하여 구성할 수 있으므로 기존 AES와 호환하면서 변형 AES 기능을 가지는 하드웨어를 용이하게 구현할 수 있다.

본 논문의 구성은 2장에서 정함수단을 기술하고, 3장에서 대칭단과 4장에서 역함수단을 각각 기술한다. 5장에서는 제안한 변형 AES의 안전성을 검증하고, 6장에서 결론을 맺는다.

2. 정함수단

본 논문에서는 AES의 표준인 FIPS-197[8]에서 정의한 기호를 사용한다. AES는 128 비트 블록이며 키의 길이에 의해 가변적인 라운드를 적용한다.
변형 AES의 정함수단의 의사 코드를 그림-1에 보인다. 그림-1에서 각 함수의 정의는 다음과 같다.

- ByteSub() 함수는 8비트 S-박스를 이용한 비선형 바이트 치환 함수이다.
- ShiftRow() 함수는 행 단위 왼쪽 회전 함수로 첫 번째 행은 회전하지 않으며, 두 번째 행은 1 바이트 회전, 세 번째 행은 2 바이트 회전, 네 번째 행은 3 바이트 회전을 적용하다.
- MixColumn() 함수는 열 단위로 혼합을 수행하는 4 8-비트 선형 변환 함수이다.
- AddRoundKey() 함수는 라운드 키 덧셈 함수이다.

```
Right_Function(byte in[16], byte out[16],
byte rk[NROUND+1][16])
  begin
    byte state[16]

    if encrypt {round=0, step=1}
     else {round=NROUND, step=-1}

    state = in ;

    AddRoundKey(state, rk[round])
    round = round + step

    for i = 1 to NROUND/2-1
      SubBytes(state)
      ShiftRows(state)
      MixColumn(state)
      AddRoundKey(state, rk[round])
      round = round + step
    end for

    SubBytes(state)
    ShiftRows(state)

    out = state ;
  end
```

<그림 1> 변형 AES의 정함수단

<그림 1>에서 NROUND는 128비트, 192비트 및 256비트 키에서 각각 10, 12 및 14로 AES의 라운드 수이다. 마지막 라운드 이외의 라운드는 모든 함수를 적용하며, 마지막 라운드는 ByteSub()과 ShiftRow()만을 수행한다. 정함수단의 라운드 키는 암호시에는 0, 1, 2.. 라운드 키를 순서적으로 적용하며, 복호시에는 암호시 라운드의 키의 역순으로 적용한다.

그림-1의 정함수단을 효과적으로 구현하기 위해서 256 4-바이트 테이블 T0-T3를 다음과 같이 정의한다.

$$T0[a] = \begin{bmatrix} S[a]*2 \\ S[a] \\ S[a] \\ S[a]*3 \end{bmatrix} \qquad T1[a] = \begin{bmatrix} S[a]*3 \\ S[a]*2 \\ S[a] \\ S[a] \end{bmatrix}$$

$$T2[a] = \begin{bmatrix} S[a] \\ S[a]*3 \\ S[a]*2 \\ S[a] \end{bmatrix} \qquad T3[a] = \begin{bmatrix} S[a] \\ S[a] \\ S[a]*3 \\ S[a]*2 \end{bmatrix}$$

T0-T3 테이블을 사용하여 SubByte(), ShiftRows(), MixColumn()을 4번의 테이블 참조와 3번의 XOR 연산으로 구현할 수 있다. 마지막 라운드는 바이트 단위로 S-박스 치환만을 수행한다.

3. 대칭단 구조

본 논문에서 제안하는 대칭단 구조의 의사코드를 <그림 2>에 보인다.

```
Symmetric_Stage(word a[4], word b[4],
word rkw[NROUND/2][4])
  begin
    word t[2], k[4] ;

    if encrypt
      { k[0] = rkw[NROUND/2][0]
        k[1] = rkw[NROUND/2][1]
        k[2] = rkw[NROUND/2][2]
        k[3] = rkw[NROUND/2][3] }
      else
      { k[0] = rkw[NROUND/2][2]
```

```
        k[1] = rkw[NROUND/2][3]
        k[2] = rkw[NROUND/2][0]
        k[3] = rkw[NROUND/2][1] }

    t[0] = a[0] ⊕ k[0]
    t[1] = a[3] ⊕ k[1]

    b[0] = a[1] ⊕ a[2] ⊕ t[1] ⊕ k[2]
    b[1] = t[0] ⊕ a[2] ⊕ t[1]
    b[2] = t[0] ⊕ a[1] ⊕ t[1]
    b[3] = t[0] ⊕ a[1] ⊕ a[2] ⊕ k[3]
end
```

<그림 2> 변형 AES의 대칭단

<그림 2>에서 a[4]는 4개의 32-비트 워드 대칭단 입력이고, b[4]는 4개의 32-비트 워드 대칭단 출력, 그리고 k[4]는 4개의 32-비트 워드 라운드 키이다. 대칭단은 32-비트 워드 단위로 연산을 수행하므로 CPU의 엔디안(endianness)에 무관하게 구현할 수 있다.

대칭단에서 라운드 키 연산 부분을 제외하면 다음의 행렬식으로 표현된다.

$$\begin{vmatrix} b[0] \\ b[1] \\ b[2] \\ b[3] \end{vmatrix} = \begin{vmatrix} 0\,1\,1\,1 \\ 1\,0\,1\,1 \\ 1\,1\,0\,1 \\ 1\,1\,1\,0 \end{vmatrix} \begin{vmatrix} a[0] \\ a[1] \\ a[2] \\ a[3] \end{vmatrix}$$

위 행렬식으로부터 '$B = MA$'이며 또한 '$A = MB$'가 된다.

암호 라운드 키와 복호 라운드 키는 교환해서 적용한다. 즉, 암호시에에 k[0]와 k[1]은 복호시에 각각 k[2]와 k[3]가 되며, 암호시에 k[2]와 k[3]는 복호시에 각각 k[0]와 k[1]이 된다. 이와 같이 구성하므로써 암호시에는 A를 B로 변환하고, 복호시에는 B를 A로 변환한다. 즉,

$$B = SymmetricFunction(A, K0, K1)$$
$$A = SymmetricFunction(B, K1, K0)$$

이 성립한다.

4. 역함수단

변형 AES의 역함수단의 의사 코드를 <그림 3>에 보인다. <그림 3>에서 각 함수의 정의는 다음과 같다.

- InvByteSub() 함수는 8비트 S^{-1}-박스를 이용한 비선형 바이트 치환 함수이다.
- InvShiftRow() 함수는 행 단위 오른쪽 회전 함수로 첫 번째 행은 회전하지 않으며, 두 번째 행은 1 바이트 회전, 세 번째 행은 2 바이트 회전, 네 번째 행은 3 바이트 회전을 적용하다.
- InvMixColumn() 함수는 열 단위로 혼합을 수행하는 4 8-비트 선형 변환 함수이다.

역함수단에서 InvMixColumn()은 선형 변환 함수이므로 '$M(A \oplus B) = MA \oplus MB$'가 성립한다. 따라서 라운드 키에 InvMixColumn()를 적용하여

```
InvShiftRows(state)
InvSubBytes(state)
InvMixColumn(state)
```

```
Inverse_Function(byte in[16], byte out[16],
byte rk[NROUND+1][16])
  begin
    byte state[16]

    state = in ;

    if encrypt
        {round=NROUND/2+1, step=1}
     else {round=NROUND/2-1, step=-1}

    for i = 1 to NROUND/2-1
      InvShiftRows(state)
      InvSubBytes(state)
      AddRoundKey(state, rk[round])
      InvMixColumn(state)
      round = round + step
    end for

    InvShiftRows(state)
    InvSubBytes(state)

    AddRoundKey(state, rk[round])

    out = state ;
  end
```

<그림 3> 변형 AES의 역함수단

AddRoundKey
(state, InvMixColumn(wk[round]))

로 변환할 수 있다.

역함수단을 효과적으로 구현하기 위해서 256개의 4-바이트 테이블 T4-T7을 다음과 같이 정의한다.

$$T4[a] = \begin{bmatrix} S^{-1}[a]*0xe \\ S^{-1}[a]*0x9 \\ S^{-1}[a]*0xd \\ S^{-1}[a]*0xb \end{bmatrix} \quad T5[a] = \begin{bmatrix} S^{-1}[a]*0xb \\ S^{-1}[a]*0xe \\ S^{-1}[a]*0x9 \\ S^{-1}[a]*0xd \end{bmatrix}$$

$$T6[a] = \begin{bmatrix} S^{-1}[a]*0xd \\ S^{-1}[a]*0xb \\ S^{-1}[a]*0xe \\ S^{-1}[a]*0x9 \end{bmatrix} \quad T7[a] = \begin{bmatrix} S^{-1}[a]*0x9 \\ S^{-1}[a]*0xd \\ S^{-1}[a]*0xb \\ S^{-1}[a]*0xe \end{bmatrix}$$

T4-T7 테이블을 사용하여 InvSubByte(), InvShiftRows(), InvMixColumn()을 4번의 테이블 참조와 3번의 XOR 연산으로 구현할 수 있다.

5. 변형 AES의 성능 및 안전성

본 논문에서 10 라운드 변형 AES를 소프트웨어로 구현 시에 소요되는 주요 연산을 <표 1>에 보인다.

<표 1> 변형 AES의 주요 연산자의 수

	Load Word	Store Word	XOR	Load Byte	Store Byte
Right Function	88	20	68	96	16
Symmetric Function	18	6	10	0	0
Inverse Function	88	20	68	96	16
Total	194	42	146	192	32

<표 1>로부터 대칭단에서 소요되는 시간은 5% 정도임을 알 수 있다. 동일한 방식으로 AES 구현시에 소요되는 연산을 비교한 표를 <표 2>에 보인다.

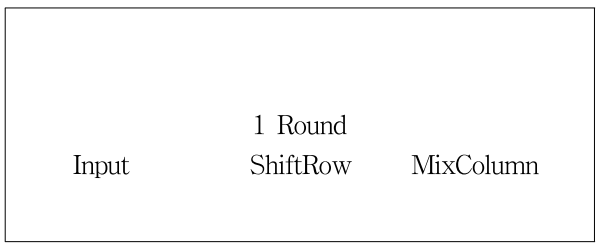
<표 2> 변형 AES와 AES의 주요 연산자의 수

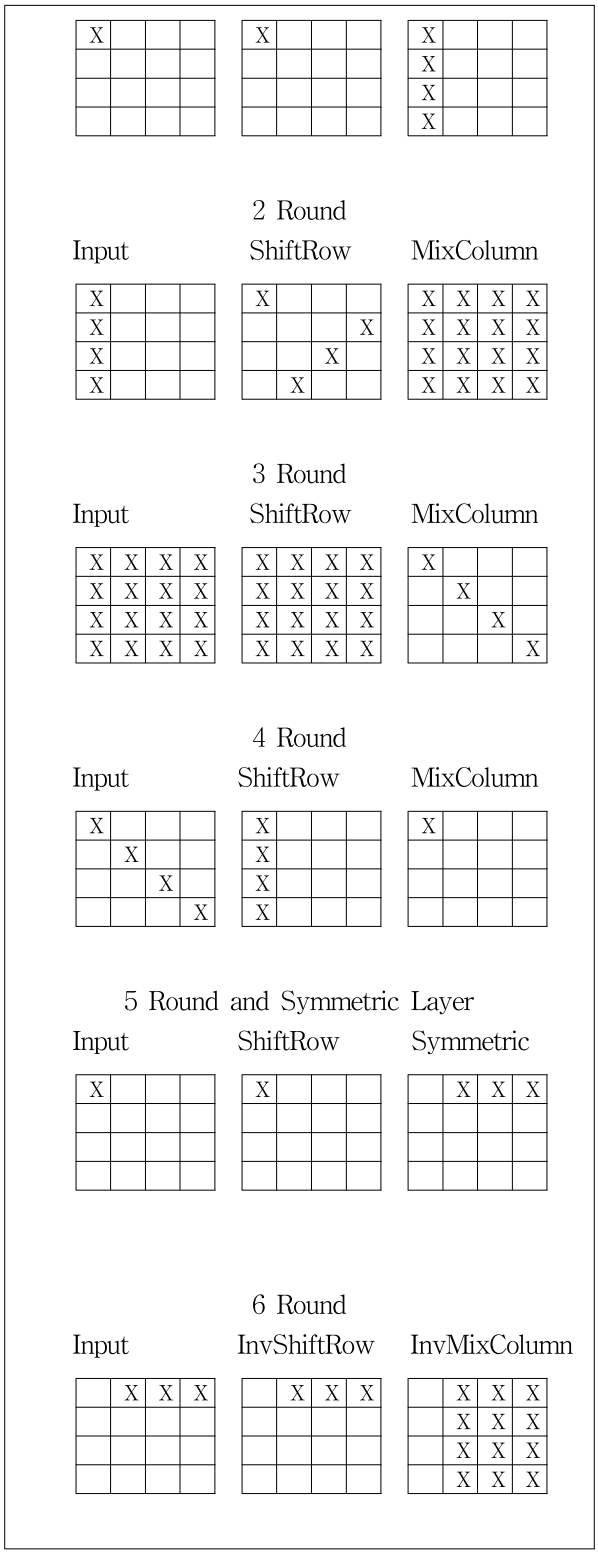
	Load Word	Store Word	XOR	Load Byte	Store Byte
Modified AES	194	42	146	192	32
AES	196	44	152	176	16

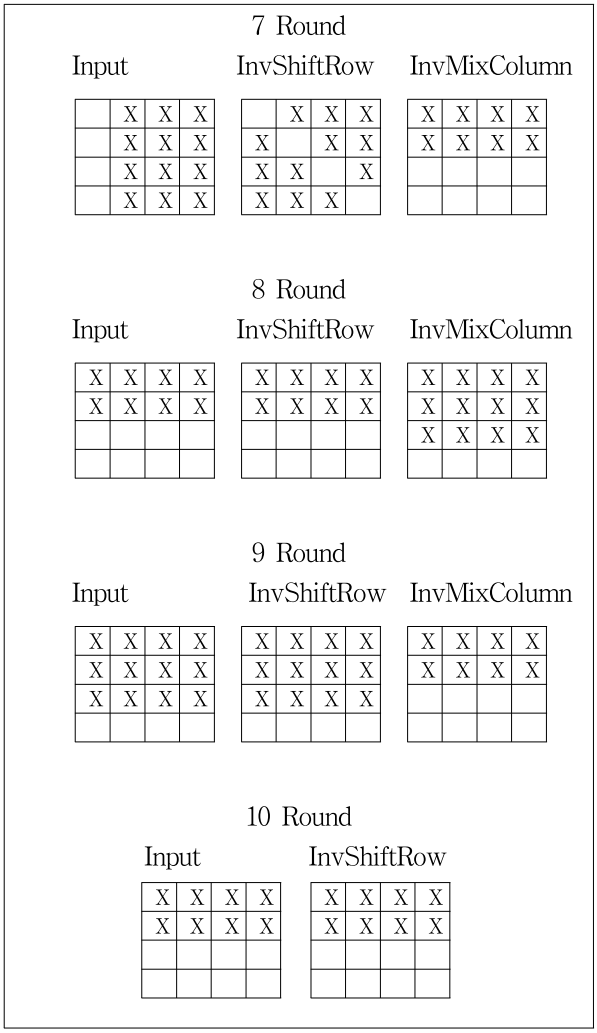
<표 2>에서 바이트 단위 로드/스토어가 변형 AES에서 많은 이유는 정함수단과 역함수단의 마지막 라운드에 기인한다. 32 비트 CPU에서는 대부분 연산자의 수행 속도가 같다. 표-2로부터 변형 AES는 606개 연산, AES는 584개 연산이 소요되어, 변형 AES가 AES에 비하여 3.7% 정도 수행 속도가 느리다.

GNU-C 컴파일러를 사용하여 동작을 검증하였으며, Windows XP, 셀러론 2.8GHz, 700M RAM 환경에서 1억 개 블록에 대하여 10 라운드 AES와 변형 AES의 암호/복호 수행 시간을 테스트한 결과 제안한 알고리즘이 약 3% 정도 수행 시간이 증가하는 것으로 나타났다.

변형 AES는 AES와 동일한 구조를 가지고 있으므로 안전성은 AES와 동일한 방식으로 검증할 수 있다. AES에 대한 안전성 분석은 차분 공격[9], 선형 공격[10], Square 공격[11], 부메랑 공격[12], 불능 차분 공격(Impossible Differentials Cryptanalysis)[13], 부정차분공격(Truncated Differentials Cryptanalysis)[14] 등이 수행되었다. 이들 공격은 공격 패스를 설정하고, 설정한 패스에서 활동성을 가지는 S-박스의 차분/선형 확률에 의하여 공격 복잡도를 계산한다. 그러므로 변형 AES의 안전성을 검증하기 위해서는 최적의 패스에서 활동성을 가지는 S-박스의 수를 산출하고 이를 AES와 비교하여 상대적인 안전성을 분석할 수 있다.







<그림 4> 변형 AES의 최소 활성 S-박스

<그림 4>에 10 라운드 변형 AES에서 최소로 활동성을 가지는 크리티칼 패스(critical path)를 보인다. 1 라운드 초기 상태에서 한 개의 S-박스만이 활동성을 가지도록 평문을 설정한다. 그림-4에서는 활동성을 가지는 S-박스의 위치를 'x'로 표시했다. 1 라운드에서는 ShiftRow()를 수행해도 여전히 한 개의 S-박스만이 활성화된다. 그러므로 1 라운드에서는 한 개의 S-박스만이 활성화된다. MixColumn() 함수를 수행하면 하나의 열에 속한 모든 S-박스가 활성화가 된다. 이는 AES의 MixColumn()이 MDS(Maximum Distance Separated) 행렬[15]을 이루기 때문이다.

2 라운드의 입력은 1 라운드의 MixColumn() 결과이다. 라운드 키 덧셈은 선형변환이므로 S-박스의 활

성 상태에 영향을 주지 않는다. ShiftRow()를 수행하면 활성 S-박스의 위치가 변화한다. 2 라운드에서는 네 개의 S-박스가 활성화된다. MixColumn() 함수를 수행하면 모든 S-박스가 활성화가 된다.

3 라운드에서는 16개의 S-박스가 활성화된다. MixColumn() 함수를 수행하면 각 열에서 오직 한 개의 S-박스만이 활성화되는 경우가 발생한다. 이는 MixColumn()가 MDS 행렬을 이루기 때문이다. 따라서 공격이 가장 용이한 경우로 대각선 위치에 있는 오직 한 개의 S-박스만이 활성화되는 경우를 가정하였다.

4 라운드의 입력에는 4개의 S-박스가 활성화되는데, ShiftRow()를 수행하면 이들이 하나의 열에 정렬된다. MixColumn() 함수를 수행하면 한 개의 S-박스만이 활성화된다. 이는 공격이 가장 용이한 경우를 가정한 것이다. 3 라운드와 4 라운드와 같은 경우가 실제로 발생하는 입력을 찾지 못했으나 공격이 가장 용이한 패스로 설정하였다.

5 라운드는 MixColumn()이 없으므로 S-박스의 활성화 상태가 변화하지 않는다. 대칭단에서는 하나의 열이 다른 세 개의 열에 영향을 주므로 그림에서와 같이 세 개의 S-박스가 활성화된다.

6 라운드부터 9 라운드까지는 InvMixColumn()이 MDS 행렬을 이루며, 최소한의 S-박스만이 활성화되는 패스를 설정하면 그림-4와 같이 된다. 10 라운드는 InvMixColumn()이 없다.

<표 3>에 <그림 4>에서 분석한 변형 AES의 최소 패스에서의 활성화 S-박스 수와 동일한 방법으로 분석한 AES의 최소 패스에서의 활성화 S-박스 수를 비교한다.

<표 3> 활성 S-박스의 비교

	1	2	3	4	5	6	7	8	9	10	계
Modified AES	1	4	16	4	1	3	12	8	12	8	69
AES	1	4	16	4	1	4	16	4	1	4	55

SPN 구조 암호 알고리즘의 안전성 평가는 Hong[16] 등에 의해서 제안된 바 있다. 차분/선형 분석법은 차분/선형 활동성을 가지는 S-박스의 수를 구하므로 해서 SPN 구조 암호 알고리즘의 안전성을 구할 수 있다. AES S-박스의 최대 차분/선형 공격 확률은 각각 2^{-6} 이다. AES의 경우 암호용 S-박스와 복호용 S^{-1}-박스는 일대일 대응되는 전단사 함수로 같은 차분/선형 확률 값을 가진다. 그리고 암호용 MixColumn() 함수 연산과 복호용 InvMixColumn() 함수 연산 역시 일대일 대응되는 전단사 함수이다.

<표 3>에서 10 라운드 변형 AES에서 활성화 되는 S-박스의 수는 총 69개이다. 실제의 차분/선형 공격은 마지막 라운드의 라운드 키를 탐색하므로 마지막 라운드의 S-박스는 안정도에 영향을 주지 않는다. 따라서 10 라운드 변형 AES에서의 안전성은 $(2^{-6})^{61} = 2^{-366}$이다. 즉, 차분/선형 공격으로 변형 AES의 키를 찾기 위해서는 2^{366} 회의 공격이 필요하다. 이는 AES의 $(2^{-6})^{51} = 2^{-306}$보다 안전성이 높다. 이를 <표 4>에 보인다.

<표 4> 차분 및 선형 공격 비교

	Differential attack	Linear attack
Modified AES	2^{-366}	2^{-366}
AES	2^{-306}	2^{-306}

부정차분공격은 바이트 단위로 연산이 적용되는 암호에서 바이트 패턴이 서로 다른 경우 '1', 같은 경우 '0'으로 정의된 차이 값을 가지고 분석하는 것으로 입력차분과 출력차분의 전부를 고려해야하는 차분분석보다 용이하게 공격할 수 있다. 본 논문에서 제안한 변형 AES는 정함수단과 역함수단이 반씩 적용되며 중간에 대칭단이 삽입된다. 대칭단에서 차분 패스가 변하면서 활성화되는 S-박스의 수가 증가한다. 그러므로 대칭단 이후의 유용한 부정차분분석의 구성이 어렵게 된다.

불능차분공격은 차분확률이 '0'에 수렴하거나, 차분이 존재하지 않는 경우 평문쌍에 이와 같은 차분을 가지는 라운드 키는 제외한 후 남은 라운드 키를 가지고 틀린 키를 제외하는 분석방법으로 가능한 후보 서브 키에 한정해서 적용하는 방법이다. 불능차분분석 역시 대칭단의 적용으로 분석 패스가 바뀌므로 복잡성을 증가시킨다.

Square 공격과 같은 바이트 패턴이 각각의 라운드 사이에서 전파되는 특성을 이용한 공격도 대칭 단에서 바이트 패턴이 바뀌므로 공격을 어렵게 한다.

6. 결 론

SPN 비밀키 블록 암호 및 복호 알고리즘은 여러 라운드의 암호 또는 복호 함수를 반복 수행한다. 암호 라운드 함수는 키합산층(Key addition layer)과 치환층(Substitution layer) S 및 확산층(Permutation layer) P의 세 단계로 구성된다. 복호 라운드 함수는 역확산층(Inverse Permutation layer) P^{-1}과 역치환층(Inverse Substitution layer) S^{-1} 및 키합산층(Key addition layer)의 세 단계로 구성된다. 이러한 SPN 구조는 암호와 복호 알고리즘이 다르므로 구현이 복잡해지는 단점을 가진다.

본 논문에서는 암호와 복호 과정이 상이한 SPN 구조 블록 암호 알고리즘을 암복호 과정이 동일한 SPN 구조로 변형시키는 방식을 제안했다. 즉, 1 라운드부터 N/2 라운드까지의 전반부 라운드 함수는 키합산층(Key addition layer)과 치환층(Substitution layer) S 및 확산층(Permutation layer) P의 세 단계로 구성하고 이를 정함수단이라 칭했으며, 후반부 (N/2) + 1 라운드부터 N 라운드까지는 역확산층(Inverse Permutation layer) P^{-1}과 역치환층(Inverse Substitution layer) S^{-1} 및 키합산층(Key addition layer)의 세 단계로 구성하고 이를 역함수단이라 칭했다. 또한 정함수단과 역함수단 사이에 규칙적인 라운드의 반복을 피하기 위해 간단한 행렬식과 라운드의 배타적 논리합으로 구성한 대칭단(Symmetry layer)을 삽입했다.

본 논문에서는 정함수단에는 안전성이 증명된 AES의 암호 알고리즘을, 역함수단에는 AES의 복호 알고리즘을 사용하였으며, 이를 가칭 변형 AES라 칭했다. 제안한 변형 AES는 암호와 복호 알고리즘이 동일하며, 암호와 복호시의 라운드 키는 상호 역순이다.

변형 AES는 암호와 복호 알고리즘이 동일하므로 하드웨어 구현시 256X8 S-박스의 수가 기존 AES에 비하여 반으로 줄어드는 장점을 가진다. 소프트웨어 수행 속도는 변형 AES가 기존 AES에 비하여 3.7% 정도 느리므로 실제적으로 큰 차이를 보이지 않았다.

변형 AES는 AES와 동일한 구조를 가지고 있으므로 안전성은 AES와 동일한 방식으로 검증할 수 있다. 블록 암호의 안전성 분석은 차분 공격, 선형 공격, Square 공격, 부메랑 공격, 불능 차분 공격, 부정차분 공격 등이 있다. 이들 공격은 공격 패스를 설정하고, 설정한 패스에서 활동성을 가지는 S-박스의 차분/선형 확률에 의하여 공격 복잡도를 계산한다. 차분/선형 공격에서 최소로 활성화되는 S-박스의 수는 10 라운드 변형 AES에서 69개로 조사되었으며, 이는 기존 AES의 55개보다 많아서 변형 AES의 안전성이 높은 것으로 판단되었다.

제안한 알고리즘은 높은 안전성을 보여주고 있으며, 암호와 복호가 동일하기 때문에 하드웨어 구성이 간단해서 스마트카드 및 RFID 태그와 같은 제한된 하드웨어 및 소프트웨어 환경에서도 쉽게 구현 가능하다. 향후 변형 AES의 부정차분공격 등에 대한 보다 심도 깊은 연구가 요구된다.

참 고 문 헌

[1] H. Feistel, "Cryptography and Computer Privacy", Scientific American, Vol. 228, No. 5, page 15-23, 1973.

[2] C.E. Shannon, "Communication Theory of Secrecy System" Bell System Technical Journal, Vol. 28, No. 4, page 656-715, 1949.

[3] National Bureau of Standards, Data Encryption Standard, FIPS-Pub. 46. National Bureau of Standards, U.S. Department of Commerce, Washington D.C., January 1977

[4] "Report on the Development of the Advanced Encryption Standard(AES)", http://www.csrc.nist.gov/encryption/aes/.

[5] J. Daemen, and V. Rijmen, "AES Proposal: Rijndael," http://www.csrc.nist.gov/encryption/aes/rijndael/Rijndael.pdf, 1999.

[6] SEED, http://www.kisa.or.kr/seed/.

[7] ARIA, http://www.nsri.re.kr/ARIA/.

[8] Federal Information Processing Standards Publication 197, "Announcing the ADVANCED ENCRYPTION STANDARD(AES)," Nov. 2001,

csrc.nist.gov/publications/fips/fips197/fips-197.pdf
[9] E. Biham and A. Shamir, "Differential Cryptanalysis of the Full 16-Round DES", LNCS 537, page 2-21, 1990.
[10] M. Matsui, "Linear Cryptanalysis Method for DES", LNCS 765, page 386-397, 1994.
[11] J. Daemen, L. Knudsan, and V. Rijmen, "The Block Cipher Square," Proceeding of FSE'97, LNCS Vol.1267, pp. 149-165, 1997.
[12] A. Birynkov, "The Boomerang attack on 5 and 6-round reduced AES", LNCS 3373, page 42-57, 2005.
[13] J. Cheon, M. Kim, K. Kim, J. Lee and S. Kang, "Improved impossible differential cryptanalysis of Rijndael and Crypton", LNCS 2288, page 39-49, 2001.
[14] L. R. Knudsen, "Truncated and higher order differential," Fast Software Encryption-Second International Workshop, LNCS Vol.1008, pp. 196-211, 1995.
[15] A. M. Youssef, S. Mister, and S. E. Tavares, "On the Design of linear Transformation for Substitution and Permutation Encryption Networks," in the Workshop Record of the Workshop on Selected Areas in Cryptography (SAC '97), pp. 40-48, Aug. 1997.
[16] S. Hong, S. Lee, J. Lim, J. Sung, and D. Cheon, "Provable security against differential and linear cryptanalysis for the SPN structure," In Fast Software Encryption 2000, LNCS Vol.1978, pp. 273-283, 2001.

조 경 연 (Gyeong-Yeon Cho)

- 1990년 2월 : 인하대학교 전자공학과 박사
- 1983년~1991년 : 삼보컴퓨터 기술연구소 책임연구원
- 1991년~2003년 : 삼보컴퓨터 기술연구소 기술고문
- 1998년~2003년 : 에이디칩스 기술고문
- 1991년~현재 : 부경대학교 공과대학 IT융합응용공학과 교수
- 관심분야 : 전산기구조, 반도체회로 설계, 암호알고리즘

송 홍 복 (Hong-Bok Song)

- 1985-1990년 :동의과학대학 조교수
- 1990년 : 동아대학교 공학박사
- 1989-1990년 : 일본구주공대 객원연구원
- 1991년-현재 : 동의대학교 공과대학 전자공학과 교수
- 1994-1995년 : 일본 미야쟈키대학 전기. 전자공학부 (POST-DOC)
- 관심분야 : 다치논리이론 및 다치논리시스템 설계, VLSI설계, 마이크로프로세서 응용

논 문 접 수 일 : 2010년 01월 27일
1차수정완료일 : 2010년 06월 03일
2차수정완료일 : 2010년 06월 08일
게 재 확 정 일 : 2010년 06월 08일