Journal of Korea Multimedia Society Vol. 11, No. 8, August 2008(pp. 1093-1100)

대칭구조 SPN 블록 암호 알고리즘

김길호[†], 박창수[††], 조경연[†††]

요 약

블록 암호를 설계하는 방식으로 크게 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 사용하고, (N/2)+1 라운드부터 N 라운드까지는 역함수를 사용한다. 또한 정함수 단과 역함수 단 사이에 대칭 블록을 구성하는 대칭 단을 삽입한다. 본 논문에서 정함수로는 안전성이 증명된 AES의 암호 알고리즘을, 역함수로는 AES의 복호 알고리즘을 사용하고, 대칭 단은 32 비트 회전과 간단한 논리연산을 사용하여 비선형성을 증가시켜 바이트 또는 워드 단위의 공격에 강하게 한다. 본 논문에서 제안한 암호와 복호가 동일한 대칭 구조 SPN 알고리즘은 하드웨어 구성이 간단한 장점을 가지므로 제한적 하드웨어 및 소프트웨어 환경인 스마트카드와 전자 칩이 내장된 태그와 같은 RFID 환경에서 안전하고 효율적인 암호를 구성할 수 있다.

Symmetry structured SPN block cipher algorithm

Gil-Ho Kim[†], Chang-Soo Park[††], Gyeong-Yeon Cho[†††]

ABSTRACT

Feistel and SPN are the two main structures in designing a block cipher algorithm. Unlike Feistel, an SPN has an asymmetric structure in encryption and decryption. In this paper we propose an SPN algorithm which has a symmetric structure in encryption and decryption. The whole operations in our SPN algorithm are composed of the even numbers of N rounds where the first half of them, 1 to N/2, applies function and the last half of them, (N+1)/2 to N, employs inverse function. Symmetry layer is executed to create a symmetry block in between function layer and inverse function layer. AES encryption and decryption algorithm, whose safety is already proved, are exploited for function and inverse function, respectively. In order to be secure enough against the byte or word unit-based attacks, 32bit rotation and simple logical operations are performed in symmetry layer. Due to the simplicity of the proposed encryption and decryption algorithm in hardware configuration, the proposed algorithm is believed to construct a safe and efficient cipher in Smart Card and RFID environments where electronic chips are built in.

Key words: AES(AES), SPN(Substitution Permutation Network), Symmetry layer(대칭 단)

1. 서 론

통신 기술의 발전과 사회의 전반적인 활동이 무선 통신망과 인터넷과 같은 범용 통신망을 이용한 지식 기반 정보화 사회로 빠르게 진행함에 따라 정보보호에 대한 인식이 점차 높아지고 있으며, 정보보호를

※ 교신저자(Corresponding Author) : 조경연, 주소 : 부산광역시 남구 대연3동(608-737), 전화 : 051)629-6252, FAX : 051)629-6210, E-mail : gycho@pknu.ac.kr
접수일 : 2008년 2월 13일, 완료일 : 2008년 5월 26일
[†] 정회원, 부경대학교 컴퓨터공학과 박사수료
(E-mail : vnlqpcdd@hanmail.net)
[††] 준회원, 부경대학교 컴퓨터공학 박사
(E-mail : buddapcs@mail1.pknu.ac.kr)
[†††] 정회원, 부경대학교 컴퓨터공학과 교수

위한 기술적 대응 조치로 암호화 기술이 일반적이다. 그리고 그동안 표준 암호로 널리 사용되어 온 DES(Data Encryption Standard)[1]의 안전성에 문제가 있어 미국을 비롯한 유럽 및 선진국들은 자국의 표준 블록 암호 개발에 주력하고 있다. 특히 미국의 경우 AES(Advanced Encryption Standard)[2] 프로젝트를 통한 Rijndael[3] 알고리즘을 표준 블록 암호 알고리즘으로 선정하였으며, 우리나라의 경우 자체 개발한 SEED[4]와 ARIA[5]를 국가 표준으로 제정하였다.

대표적인 SPN(Substitution Permutation Network) 구조의 블록 암호 알고리즘인 Rijndael은 새로운 표준 블록 암호 알고리즘을 선정하는 프로젝트인 AES 공모를 통해 채택된 표준 블록 암호 알고리즘이다. 이후 본 논문에서는 Rijndael을 AES라고 표기한다.

AES 알고리즘은 잘 알려진 블록 암호 공격법 중 E. Biham, A. Shamir에 의해 소개된 차분분석(Differential Cryptanalysis)[6]과 M. Matsui에 의해 소개된 선형분석(Linear Cryptanalysis)[7] 그리고 J. Deamen, L. Knudsan, V. Rijmen에 의해 소개된 Square attack[8]에도 안전하다는 것이 검증되었다.

블록 암호를 설계하는 방식에는 Feistel[9] 구조와 SPN 구조가 있으며, Feistel 구조는 암호와 복호 알고리즘이 동일한 것이 장점이고, SPN 구조는 암호와 복호 알고리즘이 서로 다르다는 것이 단점이다. AES는 대표적인 SPN 구조 암호 알고리즘으로 암호 및 복호 알고리즘이 서로 다르고 우리나라의 표준 블록 암호 알고리즘인 SEED는 Feistel 구조로 구현 되었으며, 최근에 새로이 표준으로 제정된 ARIA는 암호와 복호 알고리즘이 동일한 SPN 구조이다.

AES와 ARIA의 SPN 구조에서는 모든 평문을 치환(Substitution)하고 그 결과를 확산(Permutation)하는 방식인데, 특히 AES의 경우 치환은 8비트 S-박스를 사용하고 확산은 32비트 단위로 수행한다. 이를 복호하기 위해 8비트 역 S-박스와 32비트 역 확산 함수를 사용한다.

본 논문에서는 암호와 복호 과정이 동일한 SPN 구조 블록 암호 알고리즘을 제안한다. 즉, SPN 구조 알고리즘의 전체를 짝수인 N 라운드로 구성하고, 1 라운드부터 N/2 라운드까지는 정함수를 사용하고, (N/2) + 1 라운드부터 N 라운드까지는 역함수를 사용한다. 또한 정함수 단과 역함수 단 사이에 규칙적인 라운드의 반복을 피하기 위해 불규칙적인 대칭 블록을 하나의 독립적인 라운드 형태인 대칭 단(Symmetry layer)을 삽입한다.

본 논문에서는 정함수로는 안정성이 증명된 AES의 암호 알고리즘을, 역함수로는 AES의 복호 알고리즘을 사용하고, 대칭 단은 최근에 유럽의 표준 블록 암호 알고리즘 선정 프로젝트인 NESSIE(New European Schemes for Signatures. Integrity. and Encryption)[10]의 두 번째 라운드에서 선택된 Camellia[11]의 FL/FL^{-1} 함수를 참조하여 32비트 회전(Rotation)과 AND, OR 논리 연산을 사용하여 비선형을 증가시키고 바이트 또는 워드 단위 공격에 강하게 했다. 제안한 대칭 구조 SPN 블록 암호 알고리즘은 암호와 복호 알고리즘이 동일하며 본래 SPN의 안정성을 유지하면서 하드웨어 구현이 간단해진다는 장점을 가진다.

본 논문은 2007년 한국멀티미디어학회 추계 학술발표대회[12]에 발표했던 내용을 추가로 연구하여 완성했다. 본 논문의 구성은 2장에서 AES 알고리즘을 좀 더 상세히 설명하고, 3장에서는 대칭 단에 대한 자세한 설명과 4장에서 대칭 구조를 AES에 적용하고 이에 따른 안전성을 검증하고 5장에서 결론으로 끝맺는다.

2. AES 알고리즘 소개

AES는 SPN 구조의 블록 암호 알고리즘으로 SPN 구조는 C. E. Shannon의 혼돈(Confusion)과 확산(Diffusion)[13] 이론을 바탕으로 구성되었으며, 혼돈은 평문과 암호문과의 상관관계를 숨기는 것을 말하고 치환을 통해 이룰 수 있고, 확산은 평문의 통계적 특성을 암호문 전반에 확산시키는 것을 말하고 이와 같은 혼돈과 확산을 반복 적용하여 안전성을 높인 블록 암호 알고리즘이 AES이다.

AES는 128비트 블록이며 키의 길이에 의해 가변적인 라운드를 적용한다. 본 논문에서는 128비트 키를 적용한 10 라운드 AES 알고리즘으로 설명한다. AES는 암호 및 복호과정에서 생성되는 중간 결과값 스테이트(State)를 바이트 단위로 4×4의 2차원 행렬로 간단히 표현할 수 있고, 4×4의 2차원 행렬은 기존의 행 우선이 아닌 열 우선으로 행렬의 순서를 표시한다.

AES의 라운드 함수 내에 크게 4가지의 독립적인 함수가 있으며, 라운드 함수를 C 코드로 표기하면 다음과 같다.

```
Round(State, RoundKey)
{
    ByteSub(State);
    ShiftRow(State);
    MixColumn(State);
    AddRoundKey(State, RoundKey);
}

Final Round(State, RoundKey)
{
    ByteSub(State);
    ShiftRow(State);
    AddRoundKey(State, RoundKey);
}
```

라운드 함수의 입력으로 State는 암호화 및 복호화 과정의 중간 결과 값을 나타내며, RoundKey는 키 스케줄링 알고리즘에 의해 생성된 라운드 키이다.

- ByteSub() 함수는 8비트 S-박스를 이용한 비선형 바이트 치환 함수이다.
- ShiftRow() 함수는 행 단위 왼쪽 회전 함수로 첫 번째 행은 변환하지 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 32비트 선형 변환 함수이다.
- AddRoundKey() 함수는 라운드 키 덧셈 함수이다.

AES의 암호화 과정에서 라운드 함수를 적용하기 전에 화이트닝 단계로서 평문과 라운드 키 덧셈을 먼저 적용하고 라운드 함수를 수행한다. 그리고 마지막 라운드는 MixColumn() 함수를 제외하고 라운드 함수를 수행한다. 복호화 과정은 암호화 과정에서 수행한 4개의 개별 함수들의 역 변환 함수가 존재하고 그 역 변환 함수와 암호화 과정의 라운드 키의 역순으로 적용한다.

3. 대칭 단 구조

본 논문에서 제안하는 대칭 단 구조는 간단한 논리 연산과 자리바꿈으로 구성한다. 전체 진행 알고리즘에서 전반부 반은 AES 암호 알고리즘을 그리고 후반부 반은 AES 복호 알고리즘을 그대로 적용하면 바이트 단위로 움직이는 ShiftRow() 함수의 확산은 제자리로 돌아오게 되고 이것은 특정 바이트의 평문과 암호문의 위치가 동일하게 되어 어떤 공격의 단서를 제공할 가능성이 있다. 그래서 이것을 막기 위해 암호 알고리즘에서 복호 알고리즘으로 바뀌는 중간 단계에서 불규칙 라운드인 대칭 단을 삽입한다. 그림 1은 대칭 단 구조를 그림으로 표현한 것이다.

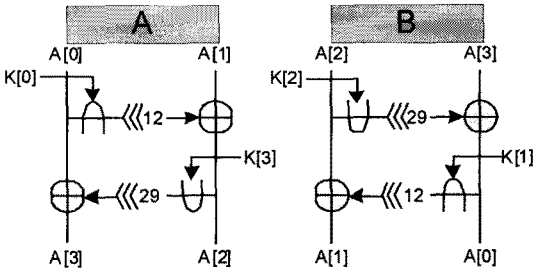


그림 1. 대칭 단 구조

전체 128비트 블록을 64비트 A, B로 나누고 각각의 A, B블록은 32비트로 나누어 2 라운드 Feistel 구조를 이루고 있다. 그림 1에서 보는 바와 같이 대칭 단의 입력/출력은 32비트 워드 단위 역순위로 대칭이 이루어진다. 먼저 대칭 단의 A블록 Feistel 구조 1 라운드에서는 A[0]과 K[0]이 AND연산 후 12비트 왼쪽 회전 연산을 수행하고 A[1]과 XOR연산을 수행 후 출력 A[2]에 저장한 후 다음 라운드로 넘어간다. 2 라운드에서는 1 라운드의 결과 값과 K[3]이 OR연산을 수행 후 29비트 왼쪽 회전 연산을 수행 후 A[0]과 마지막 XOR연산을 수행한 다음 출력으로 A[3]에 보낸다. 유사한 방법으로 B블록도 똑같이 수행한다.

```
void Symmetry(union sbuf *a, int rkey, int c0)
{
word32 temp0, temp1;
int c1;
c1=c0^1;
temp0=(*a).word[1]^ROTL((*a).word[0]
        & wkey.word[rkey][c0], 12);
temp1=(*a).word[3]^ROTL((*a).word[2]
        | wkey.word[rkey][c0+2], 29);

(*a).word[1]=(*a).word[2]^ROTL(temp1
        & wkey.word[rkey][c1], 12);
```

```
(*a).word[3]=(*a).word[0]^ROTL(temp0
        | wkey.word[rkey][c1+2], 29);

(*a).word[0]=temp1;
(*a).word[2]=temp0;
}
```

그림 1의 대칭 단 구조를 C 언어로 구현했으며, 구현한 함수는 Symmetry() 함수이다. Symmetry() 함수의 입력으로 받는 인자는 a, rkey, c0 가 있다. a는 암호와 복호 과정에서 생성된 중간 결과 값인 스테이트의 포인터 변수이고, rkey는 키 스케줄링 알고리즘에 의해 생성된 라운드 키의 인덱스 번호이며, c0은 암호일 경우는 '0' 이고 복호일 경우는 '1' 로 셋팅 된다. 32비트 또는 8비트 단위로 처리를 위해 union 구조를 하고 있다. Symmetry() 함수내의 지역변수로 word32 temp0, temp1은 논리연산을 수행 후 자리바꿈을 위한 32비트 임시 변수이다. ROTL() 함수는 왼쪽 회전 연산을 수행하는 매크로 함수이다.

4. 대칭 구조 SPN의 구현 및 검증

4.1 대칭 구조 SPN 암호 알고리즘 구현

본 논문에서 제안한 대칭 단 구조를 적용한 AES 알고리즘은 Visual Studio 6.0 C 컴파일러를 사용하여 암호/복호가 정상적으로 수행되는 것을 확인했으며, 약 5MB정도의 그림, 표, 특수문자 등이 있는 일반적인 문서파일로 Windows XP, 셀러론 2.8GHz, 700M RAM의 환경에서 기존의 AES 알고리즘과 제안한 알고리즘의 수행 시간 테스트 결과는 제안한 알고리즘이 약 1.24% 정도 증가하는 것으로 나타났다. 이것은 대칭 단에서 사용된 간단한 논리연산이 전체적인 암호/복호 알고리즘 수행에 미세한 영향을 미치지 않은 것으로 판단된다.

대칭 단 구조를 AES 알고리즘에 적용할 때 ByteSub(), ShiftRow(), MixColumn(), AddRoundKey() 함수를 변경 없이 그대로 사용한다. 그러나 전체 라운드에서 반은 암호 알고리즘을 사용하고 나머지 반은 복호 알고리즘을 사용해 암호에서 복호로 바뀌는 중간에 대칭 단 연산이 삽입된다.

AES의 암호 알고리즘 구성은 모든 라운드에 동일한 연산이 반복되면 수학식이 단순해져서 선형 공격에 취약하다. 그래서 1 라운드 시작 전에 AddRoundKey() 함수를 수행하고 마지막 라운드에서는 MixColumn() 함수를 제외하고 있다. 본 논문에서는 AES와는 다르게 전체 라운드에서 중간에 불규칙적인 라운드인 대칭 단 연산을 삽입해서 수학식이 단순해지는 점을 극복하므로 AES와 같은 1 라운드 시작 전에 AddRoundKey() 함수와 마지막 라운드에서 MixColumn() 함수를 제외하는 것이 필요하지 않다.

그림 2에서 보는 바와 같이 전반부는 AES 암호 알고리즘을 수행하는 부분으로 AES의 AddRoundKey(), ByteSub(), ShiftRow(), MixColumn() 함수 순서로 1-5 라운드까지 적용한다. AES 알고리즘과의 차이는 없고, 단지 초기 키 덧셈 부분이 제안한 알고리즘에서는 라운드 함수에 포함되어 진행하면서 5 라운드 이후 사용되는 라운드 키가 대칭 단에서 사용되는 키로 된다는 차이가 있다. 이와 같이 1-5 라운드까지 반복 수행 후 Symmetry() 함수를 수행한다. Symmetry() 함수 수행 후 AES 복호 알고리즘을 수행하고 암호 알고리즘에서 사용된 5개의 복호용 역 알고리즘을 사용한다. AES 복호 알고리즘의 적용 순서는 InvMixColumn(), InvByteSub(), InvShiftRow(), InvAddRoundKey() 함수 순으로 적용하고 6-10 라운드를 반복 수행하여 최종적으로 암호문을 만든다.

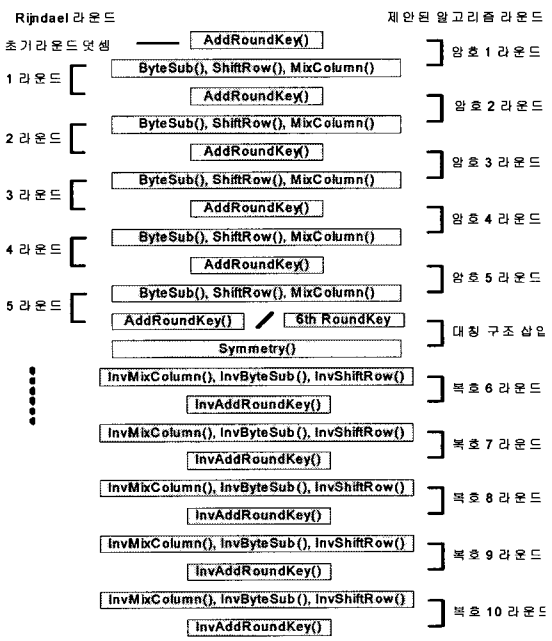


그림 2. 제안한 알고리즘의 전체 진행과정

복호과정은 그림 2의 순서를 그대로 적용하고, 단지 라운드 키의 입력이 암호화 과정의 역순으로 입력하면 된다. 본 논문에서 제안한 알고리즘의 키 스케줄링은 AES의 키 스케줄링을 그대로 사용한다.

4.2 대칭 단 구조 SPN 암호 알고리즘의 검증

4.2.1 제안한 알고리즘의 확산과정

제안한 알고리즘은 AES 알고리즘과 똑같은 SPN 구조를 이루고 있으며, 각 라운드의 확산특성이 같다. 그림 3은 제안한 알고리즘의 확산 과정을 그림으로 설명한 것이다.

1 라운드 초기 상태에서 하나의 바이트만을 공격하고 다른 바이트들은 공격하지 않는 경우를 상정했다. 그림 3에서 변화가 발생하는 바이트를 'x'로 표시했다. 1 라운드에서는 1개의 S-박스가 활성화되고 MixColumn() 함수를 수행하면 하나의 열이 변화한다. 이는 Rijndael의 MixColumn()이 MDS (Maximum Distance Separated)[14] 행렬을 이루기 때문이다. 2 라운드에서는 4개의 S-박스가 활성화되고 확산과정을 수행한 후 모든 바이트가 변화한다. 3 라운드에서는 16개의 S-박스가 활성화되고 각 열에서 1 바이트만이 변화한다. 4 라운드에서는 4개의 S-박스가 활성화되고 초기 상태와 같이 하나의 바이트만이 변화한다.

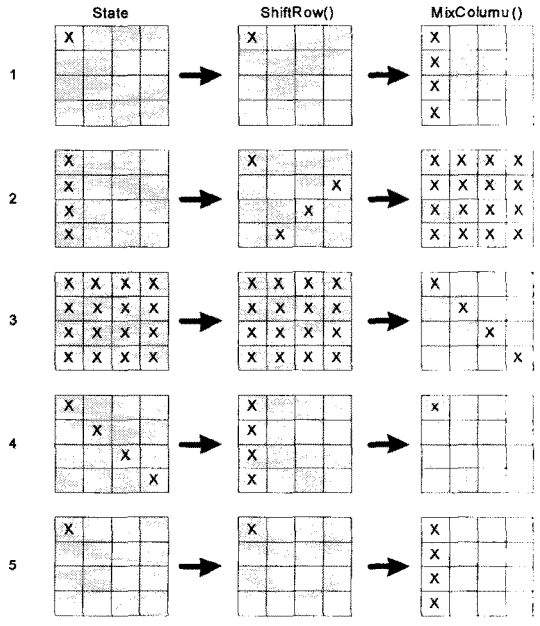


그림 3. 제안한 알고리즘의 확산 진행 과정

이러한 과정이 공격에 가장 약한 확산과정으로 보인다. 따라서 그림 3과 같은 확산과정은 제안한 알고리즘에서 가장 적은 활성화된 S-박스를 구할 수 있는 것으로 제안한 알고리즘의 안전성을 나타내는 기준이 된다.

본 논문에서 제안한 알고리즘은 그림 3에서 설명한 과정을 거친 후에 불규칙 라운드인 Symmetry() 함수를 수행하고 이어서 AES의 복호과정을 수행 한다. 복호 알고리즘의 확산 진행 과정도 그림 3의 진행과 유사하다.

4.2.2 제안한 알고리즘의 안전성 검증

SPN 구조 암호 알고리즘의 안전성 평가는 Hong [15]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 확률은 각각 2^{-6} 이다.

그리고 AES의 경우 암호용 S-박스와 복호용 역 S-박스는 일대일 대응되는 전단사 함수로 같은 차분/선형 확률 값을 가진다. 그리고 암호용 MixColumn() 함수 연산과 복호용 InvMixColumn() 함수 연산 역시 일대일 대응되는 전단사 함수다.

그림 3의 확산과정 설명에서 AES 알고리즘의 전체 10 라운드의 확산과정을 진행해보면 활동성을 가지는 S-박스의 수는 총 55개이므로 AES의 안전성은 $(2^{-6})^{55} = 2^{-330}$이라고 할 수 있다. AES 알고리즘의 암호와 복호가 일대일 대응으로 생기는 라운드간의 상쇄효과를 단절시키기 위해 불규칙 라운드인 대칭 단을 삽입했고, 대칭 단 함수인 Symmetry() 함수 내의 안전성 분석은 크게 라운드 키와 AND, OR연산[16] 수행과 바이트 단위의 자리바꿈인 확산으로 나누어 볼 수 있다.

첫 번째로 그림 1의 32비트 단위 라운드 키와 논리연산을 다음과 같이 표시 한다.

$$y = x \cap rkey, \quad \text{그리고} \quad y = x \cup rkey$$

$x[32]$는 32비트 입력을 표시하며, $x[i]$는 x의 i 번째 비트를 나타낸다. 출력 y역시 x와 같은 방법으로 표현하고, $rkey$는 라운드 키를 나타낸다.

라운드 키와 AND, OR연산이 차분분석에 어떤 영향을 미치는지 분석하면, $rkey[i]$의 값은 '0' 또는 '1'의 값을 가진다. 입력 차분 $\Delta x[i]$ = '1' 일 경우 $rkey[i]$가 '0'으로 AND연산을 수행하면 출력 차분 $\Delta y[i]$ = '0' 되며 차분분석에 영향을 미친다. 또한 $\Delta x[i]$ = '1' 일 경우 $rkey[i]$가 '1' 로 AND연산을 수행하면 출력 차분 $\Delta y[i]$ = '1' 이 되며 입력과 출력이 같아지므로 차분분석에는 영향을 미치지 않는다. 같은 방법으로 OR연산을 분석하면, 입력 차분 $\Delta x[i]$ = '1' 일 경우 $rkey[i]$가 '0'으로 OR연산을 수행하면 출력 차분 $\Delta y[i]$ = '1' 이 되며 차분분석에 영향을 미치지 않으며, 입력 차분 $\Delta x[i]$ = '1' 일 경우 $rkey[i]$가 '1' 로 OR연산을 수행하면 출력 차분 $\Delta y[i]$ = '0' 이 되며 차분분석에 영향을 미친다.

라운드 키와 AND, OR연산이 선형분석에 어떤 영향을 미치는지 분석하면, $rkey[i]$의 값이 '0' 일 경우 AND연산을 수행하면 $x[i]$의 값과는 상관없이 $y[i]$ = '0' 이 되어 선형분석에 영향을 미치고, $rkey[i]$의 값이 '1' 일 경우 $y[i] = x[i]$가 되어 선형분석에는 영향을 미치지 않는다. 같은 방법으로 OR연산은 분석하면, $rkey[i]$의 값이 '0' 일 경우 OR연산을 수행하면 $y[i] = x[i]$가 되어 선형분석에는 영향을 미치지 않는다. $rkey[i]$의 값이 '1' 일 경우 $x[i]$의 값과는 상관없이 $y[i]$ = '1' 이 되어 선형분석에 영향을 미친다.

대칭 단에 적용된 라운드 키 $rkey[i]$는 '0' 아니면 '1' 이므로 1/2의 확률을 가진다. 앞에서 설명한 바와 같이 입력차분 $\Delta x[i]$는 대응되는 라운드 키 $rkey[i]$와 논리연산을 통해 출력차분 $\Delta y[i]$가 예측할 수 없는 교란이 일어난다. 특히 Δx가 Hamming weight 가 h 일 경우 $\Delta y = 0$ 일 때 적용된 라운드 키의 확률은 2^{-h} 이다. 예를 들어 Δx가 n-비트 길이를 가지고, $\Delta x \neq 0$ 이며, $\Delta y = 0$ 일 때 적용된 라운드 키를 얻을 수 있는 확률은 $\frac{1}{2^n}\sum_{i=1}^{n} {}_nC_i \cdot 2^{-i}$ 이다.

앞서 설명한 경우는 부정차분분석(Truncated Differentials Cryptanalysis)에도 유사한 효과를 나타낸다. 부정차분분석[17]은 바이트 단위로 연산이 적용되는 암호에서 바이트 패턴이 서로 다른 경우 '1', 같은 경우 '0' 으로 정의된 차이값을 가지고 분석하는 것으로 입력차분과 출력차분의 전부를 고려해야하는 차분분석보다 쉽고 정확하게 확률을 계산할 수 있다. 본 논문에서 제안한 대칭 단은 AES의 암호/복호 알고리즘이 반반씩 적용되는 중간에 삽입이 되며, AES의 암호/복호 라운드와는 다른 독립적인 불규칙라운드이다. 대칭 단이 전체 라운드 중간에 삽입될 때 암호 전체의 안전성을 나타내주는 확률뿐만 아니라 차분 path도 변하게 된다. 이는 대칭 단에서 적용되는 라운드 키와 논리연산이 차분 path 분석을 더욱 어렵게 하고 있으며, 효과적인 차분 path를 찾기 위한 대칭 단에서 사용된 라운드 키를 추론하는 것은 그 라운드 키에 의존한다는 것을 의미한다. 결론적으로 대칭 단에서 사용하고 있는 2가지 논리연산이 위와 같은 효과로 인해 대칭 단 이후의 유용한 부정차분분석의 구성을 어렵게 하고 있다.

불능차분분석(Impossible Differentials Cryptanalysis) [18]은 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. 앞서 설명한 부정차분분석과 같이 불능차분분석 역시 대칭 단의 적용이 전체 확률뿐만 아니라 분석 path도 바뀌게 한다. 이것은 대칭 단이 불능차분분석 path도 적용된 라운드 키에 의존한다는 것이다. 이는 불능차분 path를 구하기 위한 확률이 '0' 이며, 적용 가능한 불능차분 path가 아니라는 의미이다. 그러므로 대칭 단에서 사용된 라운드 키는 불능차분 path를 교란시키며, 불능차분분석을 할 수 없게 만들므로 대칭 단 이후의 불능차분분석은 유효하지 않다.

대칭 단에서의 선형분석은 적용된 논리연산이 라운드 키 $rkey[i]$에 의존적이며, 출력 $y[i]$는 입력 $x[i]$에 독립적이다. 이는 입력과 출력의 비트쌍$(x[i],y[i])$이 어떠한 선형근사식(Linear approximation)을 가질 수 없다는 것을 말한다. 다시 말해서 대칭 단에 적용된 라운드 키를 구하기 위해 입력과 출력의 비트쌍$(x[i],y[i])$의 평균 반이 선형근사식을 만들 수 없다. 이와 같은 이유로 대칭 단의 삽입은 높은 키 의존성에 의해 만들어진 선형근사식 및 선형근사식을 구성하는 방법의 한계를 보여준다. 유사한 방법으로 Linear hull 효과 또한 대칭 단의 AND 와 OR 연산으로 감소하고 Linear hull path도 교란시킨다. 결론적으로 대칭 단의 적용은 선형분석에 저항성이 있으며, 대칭 단 이후의 라운드 분석에 어려움이 많다.

Square 공격과 같은 바이트 패턴이 각각의 라운드 사이에서 전파되는 특성을 이용한 공격도 제안한 알

고리즘에서는 전체 라운드 중간에 있는 대칭 단에서 32비트 단위의 12비트, 29비트 왼쪽 순환 연산과 라운드 키 의존적인 논리연산을 통해 바이트 단절이 일어나 Square공격에도 대칭 단 이후 라운드에서는 적용이 어렵다.

두 번째로 대칭 단 내에서 바이트 단위 자리바꿈의 영향을 살펴보면 그림 4의 상위 3개의 4×4 행렬은 제안한 알고리즘의 대칭 단을 통과하면서 자리바꿈이 일어난 후 6 라운드 확산 진행을 보여 주는 것이고, 하위 2개의 4×4 행렬은 AES의 정상적인 확산 진행 상황이다.

이는 그림 4에서 보는 봐와 같이 활성화된 S-박스의 수는 한 개로 줄었지만 4×4 행렬의 바이트 순서는 추적할 수 없을 정도로 자리바꿈이 일어난 것을 볼 수 있다. 이후의 확산 진행 과정은 그림 3과 유사하며, 총 활성화된 S-박스의 개수는 표 1과 같다.

표1에서는 대칭 단의 확산과정을 정확하게 추론할 수 없으므로 최악의 경우를 상정하여 6 라운드에서 하나의 S-박스만이 활성화되는 것으로 가정했다. 이 경우에 52개의 S-박스가 활성화됨으로 AES의 55개 S-박스와 비교해서 유사한 안정성을 가지는 것으로 평가된다. 그러나 본 논문의 가정은 가장 보수적인 가정으로 대칭 단의 확산과정에 관한 연구는 추후

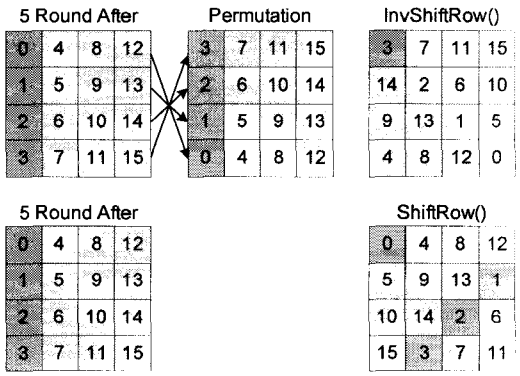


그림 4. 대칭 단 이후의 확산 진행과정 비교

표1 활성화 된 S-박스의 수 비교

라운드 / 알고리즘	1	2	3	4	5	S	6	7	8	9	10	총계
AES	1	4	16	4	1	-	4	16	4	1	4	55
제안한 알고리즘	1	4	16	4	1	-	1	4	16	4	1	52

진행되어야 한다.

5. 결 론

본 논문에서는 AES 공모 프로젝트에 의해 선택된 SPN 구조의 대표적인 블록 암호 알고리즘인 AES의 암호, 복호 알고리즘이 서로 다르다는 단점을 간단한 논리 연산과 자리바꿈으로 구성된 대칭 단을 적용하여 암호, 복호 알고리즘이 동일한 SPN 구조 블록 암호 알고리즘을 제안했다. 즉 AES의 전체 라운드 중 1 라운드에서 N/2 라운드까지는 AES의 암호 알고리즘을 적용하고, (N/2)+1 라운드부터 N 라운드까지는 AES의 복호 알고리즘을 적용하면서 암호 단과 복호 단 사이에 하나의 독립적인 라운드인 대칭 단을 적용했다.

제안한 알고리즘은 AES과 유사한 안전성을 보여주고 있으며 암호, 복호가 동일하기 때문에 하드웨어 구성이 간단해서 스마트카드 및 RFID 태그와 같은 제한된 하드웨어 및 소프트웨어 환경에서도 쉬게 구현 가능하다. 그리고 AES 이외의 다른 SPN 구조의 블록 암호 알고리즘들도 유사한 대칭 단 구조를 삽입해서 암호, 복호 알고리즘을 동일하게 구현할 수 있으므로 다른 SPN 구조의 블록 암호 설계 및 개발에도 유용한 아이디어로 사용될 것이다.

제안한 대칭 단이 알고리즘의 안정성에 미치는 해석은 향후 보다 깊은 연구가 요구된다.

참 고 문 헌

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

[2] "Report on the Development of the Advanced Encryption Standard(AES)," http://www.csrc.nist.gov/encryption/aes/round2/r2report.pdf.

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

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

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

[6] E. Biham, and A. Shamir, "Differential crypt-

analysis of DES-like cryptosystems," *Journal of Cryptology,* Vol.4, No.1, pp. 3-72, 1991.

[7] M. Matsui, "Linear cryptanalysis method for DES cipher," In Tor Helleseth, editor, Advances in *Cryptology-EUROCRYPT'93,* LNCS Vol.765, pp. 386-397, 1994.

[8] J. Daemen, L. Knudsan, and V. Rijmen, "The Block Cipher Square," *Proceeding of FSE'97,* LNCS Vol.1267, pp. 149-165, 1997.

[9] H. Feistel, "Cryptography and Computer Privacy," *Scientific American,* Vol.228, No.5, pp. 15-23, May 1973.

[10] NESSIE project, New European Schemes for Signatures Integrity and Encryption, http://cryptonessie.org/.

[11] K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita, "Specification of Camellia - a 128bit Block Cipher," http://info.isl.ntt.co.jp/camellia/, 2000.

[12] 김길호, 조경연, "대칭구조를 적용한 Rijndael 암호 알고리즘," 2007년 한국멀티미디어학회 추계학술발표대회, 10권, 2호, 2007년 11월 23-24일.

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

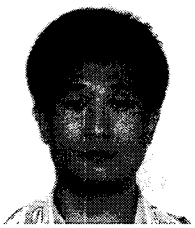
[14] A. M. Youssef, S. Mister, and S. E. Tavares, "On the Design of linear Transformation for Substitution and Permutation Encryption Networks," in the Workshop Record of the Workshop on Selected Areas in Cryptography *(SAC '97),* pp. 40-48, Aug. 1997.

[15] 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.

[16] Y. L. Yin, "A Note on the Block Cipher Camellia," a contribution for ISO/IEC JTC1/SC27, Aug. 2000.

[17] L. R. Knudsen, "Truncated and higher order differential," Fast Software Encryption-Second International Workshop, LNCS Vol.1008, pp. 196-211, 1995.

[18] E. Biham, A. Biryukov, and A. Shamir, "Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials," Advances in *Cryptology-EUROCRYPT'99,* LNCS Vol.1592, pp. 12-23, 1999.



김 길 호

2000년 2월 한국방송통신대학교 전자계산학과 학사
2002년 2월 부경대학교 컴퓨터공학과 석사
2007년 2월 부경대학교 컴퓨터공학과 박사수료

관심분야 : 암호 알고리즘, 무선통신 보안, 컴퓨터구조, 게임프로그래밍

박 창 수

1995년 2월 인제대학교 전자공학과 학사
2001년 2월 부경대학교 컴퓨터공학과 석사
2007년 8월 부경대학교 컴퓨터공학과 박사
2008년 3월~현재 부경대학교 누리사업단 계약교수

관심분야 : 반도체회로 설계, 암호 알고리즘, 컴퓨터 구조

조 경 연

1990년 2월 인하대학교 전자공학과 박사
1983년~1991년 삼보컴퓨터 기술연구소 책임연구원
1991년~2003년 삼보컴퓨터 기술연구소 기술고문
1998년~2003년 에이디칩스 기술고문
1991년~현재 부경대학교 공과대학 전자컴퓨터정보통신공학부 교수

관심분야 : 전산기구조, 반도체회로 설계, 암호 알고리즘