암호/복호를 동일하게 개선한 RC6 알고리즘

김길호* · 김종남* · 조경연*

*부경대학교 컴퓨터공학

Encryption/Decryption the same improved RC6 algorithm

Gil-Ho Kim* · Jong-Nam Kim* · Gyeong-Yeon Cho*

*Dept of Computer Engineering PuKyong National Univ.

E-mail : vnlqpcdd@hanmail.net

요 약

암호/복호 알고리즘이 서로 다른 RC6을 간단한 논리 연산과 회전 연산을 사용한 대칭단의 삽입으로 암호/복호를 같게 구현했다. 즉 RC6의 전체 라운드의 반은 RC6의 암호 알고리즘을 나머지 반은 RC6의 복호 알고리즘을 사용하고 암호와 복호 알고리즘 중간에 대칭 단을 넣어 암호/복호가 같은 개선된 RC6을 구현했다. 제안한 RC6 알고리즘은 기존의 RC6 알고리즘과 수행 속도에서는 거의 차이가 없고, 안전성은 대칭단의 삽입으로 차분 및 선형 분석에 필요한 높은 확률의 패스를 단절시켜 효과적인 분석을 어렵게 하고 있다. 제안한 알고리즘은 암호/복호가 다른 블록 암호 알고리즘에 간단히 적용하여 암호/복호가 같게 만들 수 있으며, 새로운 블록 암호 알고리즘의 설계에도 좋은 아이디어로 사용할 수 있다.

ABSTRACT

RC6 which has different algorithm of encryption and decryption has been implemented to have the same algorithm between encryption and decryption though inserting symmetry layer using simple rotate and logical operation. That means the half of whole RC6 round uses encryption algorithm and the rest of it uses decryption one and symmetry layer has been put into the middle of encryption and decryption. The proposed RC6 algorithm has no difference with the original one in the speed of process. However it is quite safe because by inserting symmetry layer the path of high probability which is needed for differential and linear analysis is cut off so that it is hard to be analyzed. The proposed algorithm can be easily applied to the algorithm which has different encryption and decryption and make it same, and it can be good idea to be used to design a new block cipher algorithm.

키워드

RC6, Feistel, Symmetry layer

Ⅰ. 서 론

간단하고, 빠르며, 안전한 블록 암호 알고리즘인 RC6은 미국의 AES(Advanced Encryption Standard)[1] 선정 프로젝트, 유럽의 NESSIE(New European Schemes for Signatures. Integrity. and Encryption)[2] 프로젝트, 일본의 CRYPTREC[3]의 128bit 및 가변길이 블록 암호 선정의 최종 후보 암호 알고리즘이었다.

RC6은 변형된 Feistel 구조로 암호/복호 알고리즘이 다른 단점을 가지고 있다. 이러한 단점은 암호 알고리즘을 하드웨어로 구현 시 암호/복호 알고리즘이 같은 것보다 면적이 약 2배 정도 증가하게 된다.

본 본문에서는 암호/복호가 다른 RC6을 간단한 논리 연산만으로 구성된 대칭단을 삽입하여 RC6의 암호/복호 알고리즘을 같게 개선하였다. 다시 말해서 RC6의 전체 20 라운드 중 10 라운드는 RC6의 암호 알고리즘을, 나머지 10 라운드는 복호 알고리즘을 사용하고 중간에 대칭단을 삽입하여 RC6을 암호/복호를 같게 구성했다.

대칭단이 적용된 제안한 RC6 알고리즘은 기존의 RC6 알고리즘과의 수행속도 비교에서 거의 같은 수행속도를 보이고 있으며, 안전성 측면에서도 대칭단의 적용이 암호 분석을 어렵게 하고 있다.

II. 관 련 연 구

RC6은 변형된 Feistel 구조의 암호 알고리즘으로 RC6-w/r/b로 표기한다. 여기서 w는 워드의 크기로 32비트이며, r은 라운드 수로 블록의 크기가 128비트인 경우 20 라운드 이고, b는 암호화 키의 바이트 수로 16바이트 이다. 본 논문에서는 128비트 블록으로 설명한다.

RC6에서 사용된 연산자는 다음과 같다.

- A+B 정수덧셈 mod 2^w
- A-B 정수뺄셈 mod 2^w
- A⊕B word 단위 bit별 xor 연산
- A × B 정수곱셈 mod 2^w
- A≪B B의 log w만큼 A를 왼쪽 회전연산
- A≫B B의 log w만큼 A를 오른쪽 회전연산
- (A,B,C,D)=(B,C,D,A) word 단위 할당연산

RC6-w/r/b의 암호화 알고리즘

```
B = B + S[0];
D = D + S[1];
for(i=1; i<r; i++)
{
  t = (B × (2B + 1)) ≪ log w;
  u = (D × (2D + 1)) ≪ log w;
  A = ((A ⊕ t) ≪ u) + S[2i];
  C = ((C ⊕ u) ≪ t) + S[2i+1];
  (A, B, C, D) = (B, C, D, A);
}
A = A + S[2r+2];
C = C + S[2r+3];
```

RC6-w/r/b의 복호화 알고리즘

```
C = C + S[2r+3];
A = A + S[2r+2];
for(i=r; i>=1; i--)
{
  (A, B, C, D) = (D, A, B, C);
  u = (D × (2D + 1)) ≪ log w;
  t = (B × (2B + 1)) ≪ log w;
  C = ((C - S[2i+1]) ≫ t) ⊕ u;
  A = ((A - S[2i]) ≫ u) ⊕ t;
}
D = D - S[1];
B = B - S[0];
```

RC6의 암호/복호는 32비트 워드 단위로 A, B, C, D 4개의 저장 장소에 평문/암호문을 가지고 20 라운드를 반복 수행 후 암호문/평문을 만들어 낸다. A, B, C, D 4개의 워드는 라운드 함수 수행 후 모두 병렬로 워드 단위 왼쪽/오른쪽 회전 연산이 이루어지며, 라운드 함수 수행 전과 후에 화이트닝(whitening) 단계로 라운드 키와 덧셈/뺄셈을 수행한다. 라운드 함수 내의 핵심적인 안전성은 데이터 의존 회전 연산이고, 이 회전 연산 양은 f(x) = x(2x+1)의 2차 함수에 고정된 5비트 왼쪽 회전 연산으로 만들어진다.

RC6은 암호에 적용된 라운드 키는 복호할 때는 암호의 역순으로 적용한다. 그리고 RC6은 변형 휘스탈(Feistel) 구조로 암호/복호 알고리즘이 서로 다르다.

III. 제 안 사 항

3.1 대칭단(Symmetry layer)구조

본 논문에서 제안하는 대칭단은 바이트 단위 논리 연산과 고정된 회전 연산으로 구성되어 소프트웨어 및 하드웨어 수행 속도가 빠르다. RC6 전체 진행 라운드의 반은 암호 알고리즘을 수행하고, 나머지 반은 복호 알고리즘을 수행하면서 중간에 대칭단을 삽입하여 암호/복호가 다른 RC6을 암호/복호 알고리즘이 같은 개선된 RC6을 만든다.

대칭단의 역할은 크게 2가지로 볼 수 있으며, 첫째는 암호/복호가 동일한 RC6을 만드는 것이고, 둘째는 동일한 라운드의 적용에서 대칭단의 삽입으로 불규칙성을 통해서 RC6의 안전성을 향상시키는 것이다.

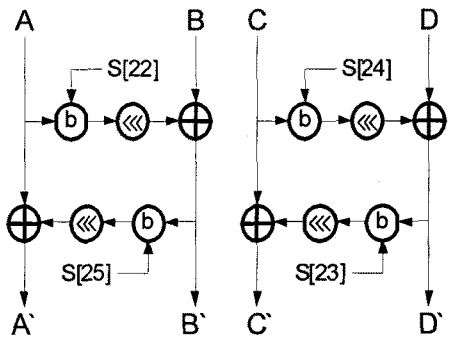

그림 1. 대칭단

그림 1은 대칭단을 그림으로 표현한 것으로 전체 128비트를 32비트 A, B, C, D로 나누고 A, B 블록과 C, D 블록으로 나누어 진행한다. 먼저 32비트 A와 라운드 키 S[22]를 가지고 b 함수를 수행한다. b 함수는 32비트 블록을 바이트 단위로 다시 나누어 and 와 or 연산을 번갈아 수행한다.

b 함수 수행 후 12비트 왼쪽 회전 연산을 수행하고 32비트 B와 xor 연산을 수행 후 출력 B'로 보낸다. 그리고 B'와 라운드 키 S[25]를 b 함수 실행 후 29비트 왼쪽 회전 연산을 수행한 다음 A와 xor 연산을 적용해서 출력 A'를 만든다.

같은 방법으로 블록 C, D를 수행한 후 출력 C', D'를 만든다.

3.2 RC6에 대칭단 구현

대칭단을 RC6 알고리즘에 적용할 때 기존의 라운드 함수 내의 연산은 변경 없이 그대로 사용한다. 그러나 전체 진행 라운드의 반은 암호 알고리즘을 나머지 반은 복호 알고리즘을 적용하고, 중간에 대칭단을 삽입한다.

그림 2는 제안한 알고리즘의 전체 진행과정을 그림으로 표현한 것으로 먼저 암호화 과정은 라운드 함수 진행 전에 화이트닝 단계로 라운드 키와 덧셈 연산을 수행한 후 10 라운드 암호화 라운드를 수행한다. 각 라운드 연산에서 32비트 라운드 키를 2개씩 덧셈 연산에 사용한다. 다음으로 대칭 단의 적용은 3.1 대칭단 구조에서 설명한 대로 실행하며, 32비트 라운드 키를 4개 사용한다. 나머지 10 라운드는 RC6의 복호 알고리즘을 적용하고 각 라운드마다 32비트 라운드 키 2개씩 뺄셈 연산을 수행 후 최종적으로 마지막 화이트닝 과정을 거친 후 128비트 암호문을 생성한다.

제안한 알고리즘의 복호는 그림 2의 과정을 그대로 수행하며, 라운드 키의 적용을 암호화 과정의 역순으로 적용한다. 그리고 화이트닝 단계에서 수행한 덧셈 연산을 뺄셈 연산으로 전환하고, 대칭단의 수행과정을 암호의 역순으로 적용한다.

본 논문에서 제안한 알고리즘의 키 스케줄링은 RC6의 키 스케줄링을 그대로 사용하며, 단지 대칭 단에서 사용된 32비트 4개의 키를 더 생성해서 총 32비트 48개의 키를 사용한다.

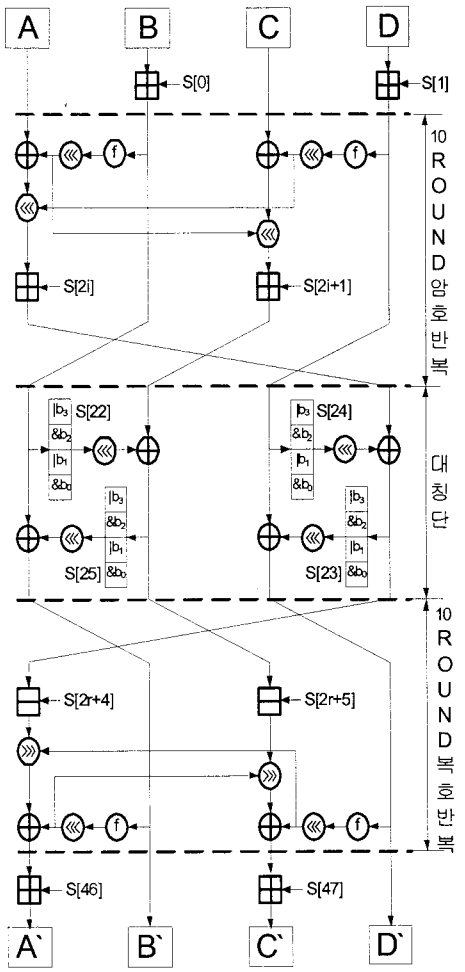

그림 2. 제안한 알고리즘 진행도

IV. 연 구 결 과 및 분 석

4.1 수행 테스트 결과

본 논문에서 제안한 대칭단 구조를 적용한 RC6 알고리즘은 암호 운영모드로 CBC(Cipher Block Chaining)모드를 적용하여 Visual Studio 2005 C 컴파일러를 사용하여 암호/복호가 정상적으로 수행되는 것을 확인했으며, 약 30MB 정도의 그림, 표, 특수문자 등이 있는 일반적인 한글 문서파일로 Windows XP, 셀러론 2,8Ghz, 700M RAM의 환경에서 기존의 RC6 알고리즘과 제안한 알고리즘의 수행 시간을 테스트했다. 결과는 표 1과 같으며, 제안한 알고리즘의 수행 시간이 약 0.32초 정도 증가하는 것으로 나타났다. 이는 대칭 단에서 적용한 간단한 논리 연산이 전체적인 암호/복호 알고리즘 수행에 거의 영향을 미치는 않는 것으로 판단된다.

표 1. 수행시간 테스트 결과(단위: 초)

시간 / 알고리즘	암호	복호
RC6	10.171	10.156
제안한 알고리즘	10.531	10.438

4.2 안전성 검증

RC6의 안전성 평가는 [4]에 잘 나타나 있으며 본 논문에서는 대칭 단의 안전성을 평가해서 제안한 알고리즘의 안전성을 평가한다. 대칭단 내에는 라운드 키와 and 와 or 연산, 그리고 고정된 길이의 왼쪽 회전 연산이 있다. 라운드 키와 and 와 or 연산의 차분[5] 및 선형[6] 분석의 영향을 표 2, 표 3에 나타냈다.

표 2. and 와 or 의 차분분석

and	S[i]	△x[i], △y[i]	영향
	0	△x[i]=1 → △y[i]=0	yes
	1	△x[i]=1 → △y[i]=1	no
or	S[i]	△x[i], △y[i]	영향
	0	△x[i]=1 → △y[i]=1	no
	1	△x[i]=1 → △y[i]=0	yes

표 3. and 와 or 의 선형분석

and	S[i]	x[i], y[i] 의 값	영향
	0	x[i]와 독립으로 y[i]=0	yes
	1	y[i] = x[i]	no
or	S[i]	x[i], y[i] 의 값	영향
	0	y[i] = x[i]	no
	1	x[i]와 독립으로 y[i]=1	yes

표 2, 표 3에서 x[i]는 입력으로 32비트 x의 i 번 째 비트를 나타내며, 출력 y[i], 라운드 키 S[i] 도 x[i]와 같이 표기한다. 그리고 △x는 입력 차분이고, △y는 출력 차분이다.

표 2에서 S[i] = 0 일 경우 and 연산으로 입력 차분 △x[i] = 1 은 출력 차분 △y[i] = 0 이 되므로 차분분석에 유용하며, S[i] = 1 일 경우 or 연산으로 입력 차분 △x[i] = 1 은 출력 차분 △y[i] = 0 이 되므로 차분분석을 할 수 있다. 표 3은 S[i] = 0 일 때 입력 x[i] 와 and 연산은 출력 y[i] 는 무조건 0 이 되므로 선형분석에 영향이 있으며, S[i] = 1 일 때 입력 x[i] 와 or 연산은 출력 y[i] 는 무조건 1 이 되므로 선형분석을 할 수 있다.

대칭단에서 사용된 라운드 키 S[i] 는 1/2 의 확률을 가진다. 입력차분 △x[i] 나 입력 값 x[i] 는 S[i] 와 and, or 연산을 통해 출력차분 △y[i] 나 출력 값 y[i] 를 예측할 수 없는 방해가 일어난다. 특히 △x의 Hamming weight가 h일 경우 △y = 0 일 때 적용된 라운드 키의 확률은 2^{-h} 이다. 예를 들어 32비트 길이에서 △x ≠ 0 이며, △y = 0 일 때 사용된 라운드 키를 얻을 수 있을 확률은 $\frac{1}{2^{32}}\sum_{i=1}^{32}\binom{32}{i}2^{-i}$ 이다.

제안한 알고리즘의 차분, 선형분석은 RC6 암호 알고리즘을 적용한 10 라운드에서 확률이 높은 차분, 선형 패스가가 대칭단에서 라운드 키와의 논리 연산의 적용 후 변화 또는 단절이 일어나 대칭단 이후 유효한 차분, 선형 패스의 구성을 어렵게 하고 있다. 그리고 Square[7] 공격과 같은 바이트 패턴이 각각의 라운드 사이에서 전파되는 특성을 이용한 공격도 대칭단의 논리 연산과 회전 연산을 통해 바이트 단절이 일어나 Square 공격에도 내성이 있다.

V. 결 론

본 논문에서 RC6의 암호와 복호 알고리즘이 다른 것을 간단한 논리 연산과 고정된 회전 연산만으로 이루어진 대칭단을 삽입하여 RC6의 암호와 복호를 같게 구현했다. 즉 RC6의 10 라운드는 암호 알고리즘을, 나머지 10 라운드는 복호 알고리즘을 사용하고 중간에 대칭단을 삽입해서 암호와 복호를 같게 만들었다.

제안한 알고리즘을 기존의 RC6과 수행 속도를 비교해서 별 차이가 없었으며, 안전성에서도 대칭단의 적용이 암호 알고리즘을 분석하는데 유효한 차분 및 선형 패스를 단절 또는 변화를 통해 방해하여 어렵게 하고 있다.

제안한 알고리즘은 제한적인 하드웨어 환경인 스마트 카드나 전자 칩이 내장된 RFID의 태그 등에 구현 시 하드웨어의 면적을 1/2 정도로 줄일 수 있다. 그리고 기존의 블록 암호 알고리즘 중 암호와 복호가 다른 알고리즘도 간단한 대칭단의 삽입으로 암호와 복호를 같게 할 수 있으며, 새로운 블록 암호 알고리즘의 설계에도 좋은 아이디어로 사용할 수 있다.

감 사 의 글

본 연구는 중소기업청의 산학연공동기술개발지원사업(선도형), 한국산업기술재단의 지역혁신인력양상사업의 지원으로 수행되었음.

참 고 문 헌

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

[2] "New European Schemes for Signatures. Integrity. and Encryption(NESSIE)." http://cryptonessie.org/.

[3] "Cryptography Research and Evaluation Committees(CRYPTREC)." http://www.cryptrec.go.jp/.

[4] S. Contini, R.L. Rivest, M.J.B. Robshaw, and Y.L. Yin, "The security of RC6." http://www.rsasecurity.com/rsalabs/aes.

[5] E. Biham and A. Shamir, "Differential cryptanalysis for DES-like cryptosystem." Journal of Cryptlogy 4(1): 3-17, 1991.

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

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