HIGHT 블록암호 알고리즘 사양 및 세부 명세서

2009.07



1. 개요

HIGHT(HIGh security and light weigHT)는 RFID, USN 등과 같이 저전력·경량화를 요구하는 컴퓨팅 환경에서 기밀성을 제공하기 위해 2005년 KISA, 구) 국가보안연구소 및 고려대가 공동으로 개발한 64비트 블록암호 알고리즘이다.

HIGHT는 초경량 블록암호 알고리즘으로 128비트 마스터키, 64비트 평문으로 부터 64비트 암호문을 출력한다. 제한적 자원을 갖는 환경에서 구현될 수 있도록 8비트 단위의 기본적인 산술 연산들인 XOR, 덧셈, 순환이동만으로 SEED, AES 등 기타 알고리즘보다 간단한 알고리즘 구조로 설계되었다. 안전성과 효율성을 동시에 고려하는 정교한 설계 논리에 기반하고 있다.

2. 용어정의

- 가. 평문 : 암호화되지 않은 원래의 정보
- 나. 암호문 : 평문을 합당하게 푸는 방법을 모르는 사람에게는 알 수 없는 형태로 변환한 데이터
- 다. 암호화 : 평문을 암호문으로 변환하는 과정
- 라. 복호화 : 암호문을 평문으로 변환하는 과정
- 마. 대칭키 암호 : 암·복호화 키가 같은 암호
- 바. 블록암호 : 고정된 길이를 갖는 평문(암호문) 블록을 고정된 길이를 갖는 암호문(또는 평문) 블록으로 변환하는 함수
- 사. 반복 블록암호 : 라운드 함수를 순차적으로 반복하여 암·복호화를 수행 하는 블록암호
- 아. Feistel 구조 : 각각 t비트인 L_0, R_0 블록으로 이루어진 2t비트 평문 블록 (L_0, R_0) 이 r라운드 $(r \ge 1)$ 를 거쳐 암호문 (L_r, R_r) 으로 변환되는 반복 구조
- 자. 라운드 함수 : 반복 블록암호의 각 라운드에서 사용되는 함수
- 차. 서브키(라운드키) : 라운드 함수에서 사용되는 키
- 카. 암호키(마스터키): 평문 또는 암호문의 암·복호화에 사용되는 비밀정보
- 타. 화이트닝키 : 반복 구조의 블록암호에서 안전성을 높이기 위해 알고리즘의초기변환 또는 최종 변환에 적용되는 라운드키

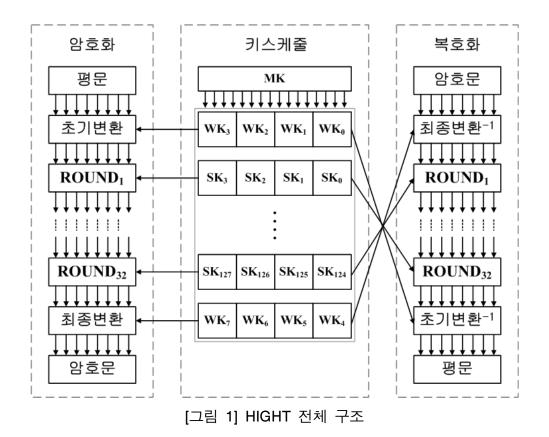
3. 기호와 표기

- ① 64비트 평문과 암호문은 각각 8개의 바이트로 다음과 같다.
 - 평문 : $P = P_7 \| P_6 \| \cdots \| P_0$
 - 암호문 : $C = C_7 \parallel C_6 \parallel \cdots \parallel C_0$
- ② 64비트 라운드함수 입·출력은 8개의 바이트로 다음과 같다.
 - $X_i = X_{i,7} \parallel X_{i,6} \parallel \cdots \parallel X_{i,0}$, $i = 0, \dots, 32$
- ③ 128비트 키(마스터키)는 16개의 바이트로 다음과 같다.
 - $MK = MK_{15} \parallel MK_{14} \parallel \cdots \parallel MK_0$
- ④ 라운드함수에 적용되는 라운드키는 서브키 SK_i 와 화이트닝키 WK_i 로 다음 과 같다.
 - SK_i , $i = 0, \dots, 127$
 - WK_i , $i = 0, \dots, 7$
- ⑤ 산술 연산 기호
 - 田 : 법 2⁸ 덧셈
 - □ : 법 2⁸ 뺄셈
 - ⊕ : XOR (배타적 논리합)
 - *A*^{≪s} : 8비트 값 *A*에 대한 *s*비트 좌측 순환 이동

4. 64비트 블록암호 HIGHT

4.1. 알고리즘 전체 구조

HIGHT의 전체 구조는 일반화된 Feistel 변형 구조로 이루어져 있으며, 64비트의 평문과 128비트 키로부터 생성된 8개의 8비트 화이트닝 키와 128개의 8비트 서브키를 입력으로 사용하여 총 32라운드를 거쳐 64비트 암호문을 출력한다. 다음 [그림 1]은 HIGHT 알고리즘의 전체 구조를 나타낸다.



4.2. 키 스케줄

HIGHT의 라운드키는 화이트닝키와 LFSR을 사용하여 생성한 서브키들로 이루어진다.

4.2.1. 화이트닝키 생성

화이트닝키는 마스터키를 사용하여 다음과 같은 방법으로 생성한다.

$$WK_{i} = \begin{cases} MK_{i+12} &, \ 0 \leq i \leq 3 \\ MK_{i-4} &, \ 4 \leq i \leq 7 \end{cases}$$

4.2.2. LFSR(Left Feedback Shift Register) h

LFSR h의 연결 다항식은 $x^7 + x^3 + 1$ 이다. 이 다항식은 $F_2[x]$ 에서 원시다항식이기 때문에 h는 $2^7 - 1 = 127$ 의 주기를 갖는다. h의 초기 내부 상태 값은

 $\delta_0 = (s_6, s_5, s_4, s_3, s_2, s_1, s_0) = (1, 0, 1, 1, 0, 1, 0)_2$ 으로 정해진다. 그러면 $i=1,\cdots,127$ 에 대하여 δ_i 는 다음과 같이 생성된다.

$$\begin{split} s_{i+6} &= s_{i+2} \oplus s_{i-1}, \\ \delta_i &= (s_{i+6}, \, s_{i+5}, \, s_{i+4}, \, s_{i+3}, \, s_{i+2}, \, s_{i+1}, \, s_i), \qquad 1 \leq i \leq 127 \end{split}$$

h의 주기가 127이기 때문에 $\delta_0 = \delta_{127}$ 이다.

4.2.3. 서브키 생성

서브키를 생성하는 알고리즘은 다음과 같이 구성된다.

For
$$i = 0$$
 to 7
For $j = 0$ to 7
 $SK_{16.i+j} \leftarrow MK_{j-i \mod 8} \boxplus \delta_{16.i+j};$
For $j = 0$ to 7
 $SK_{16.i+j+8} \leftarrow MK_{(j-i \mod 8)+8} \boxplus \delta_{16.i+j+8};$

 δ_i $(0 \le i \le 127)$ 는 LFSR h의 내부 상태의 값들이다.

4.2.4. 암호화키 생성

HIGHT의 암호화 알고리즘이 수행되기 위하여 라운드키 생성과정은 바이트 단위의 8개의 화이트닝키 WK_0 , ..., WK_7 와 128개의 서브키 SK_0 , ..., SK_{127} 를 생성한다.

4.2.5. 복호화키 생성

HIGHT의 복호화 알고리즘은 암호화 알고리즘과 같은 화이트닝키와 서브키바이트 들을 이용하지만 순서가 다르다. 복호화 알고리즘에서는 $WK_{4,}WK_{5,}WK_{6,}WK_{7}$ 이 초기변환에, $WK_{0,}WK_{1,}WK_{2,}WK_{3}$ 이 최종 변환에 사용된다. 복호화 알고리즘에서 사용되는 서브키 바이트들은 SK_{i}^{\prime} 로 표시되며, 다음과 같이 정의된다.

$$SK'_{i} = SK_{127-i}, i = 0, \dots, 127$$

4.3. 암호화

4.3.1. 암호화 초기변환

HIGHT 암호화의 초기변환은 네 개의 화이트닝키 바이트 WK_0, WK_1, WK_2, WK_3 를 이용하여 평문 P를 첫 번째 라운드 함수의 입력인 $X_0 = X_{0,7} \parallel X_{0,6} \parallel \cdots \parallel X_{0,0}$ 로 변환한다.

$$\begin{split} &X_{0,i} = P_i \,, \quad i = 1, \, 3, \, 5, \, 7 \\ &X_{0,0} = P_0 \boxplus \text{WK}_0 \\ &X_{0,2} = P_2 \oplus \text{WK}_1 \\ &X_{0,4} = P_4 \boxplus \text{WK}_2 \\ &X_{0,6} = P_6 \oplus \text{WK}_3 \end{split}$$

4.3.2. 암호화 라운드 함수

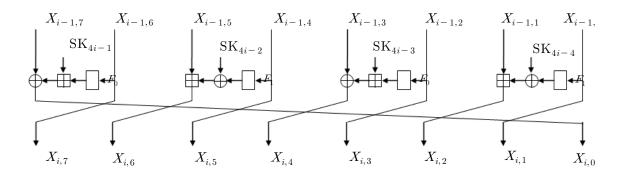
HIGHT의 라운드 함수는 다음과 같은 두개의 보조 함수를 갖는다.

$$\begin{split} F_0(X) &= X^{\ll \, 1} \oplus X^{\ll \, 2} \oplus X^{\ll \, 7} \\ F_1(X) &= X^{\ll \, 3} \oplus X^{\ll \, 4} \oplus X^{\ll \, 6} \end{split}$$

라운드 함수는 Round,부터 Round,32까지 32회 반복한다.

각 i번째 라운드 함수 $\mathrm{Round}_{i},\ i=1,\cdots,31$ 는 $X_{i-1}=X_{i-1,7}\|\cdots\|X_{i-1,0}$ 을 $X_{i}=X_{i,7}\|\cdots\|X_{i,0}$ 로 다음과 같이 변환하며 [그림 2]와 같다.

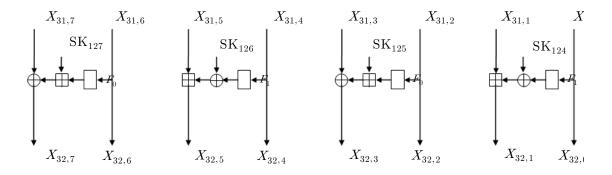
$$\begin{split} &X_{i,j} = X_{i-1,j-1} \,, \ j=1, \, 3, \, 5, \, 7 \\ &X_{i,0} = X_{i-1,7} \oplus (F_0(X_{i-1,6}) \boxplus \mathrm{SK}_{4i-1}) \\ &X_{i,2} = X_{i-1,1} \boxplus (F_0(X_{i-1,0}) \oplus \mathrm{SK}_{4i-4}) \\ &X_{i,4} = X_{i-1,3} \oplus (F_0(X_{i-1,2}) \boxplus \mathrm{SK}_{4i-3}) \\ &X_{i,6} = X_{i-1,5} \boxplus (F_0(X_{i-1,4}) \oplus \mathrm{SK}_{4i-2}) \end{split}$$



[그림 2] HIGHT의 i번째 라운드 함수 $\mathrm{Round}_i,\ i=1,\cdots,31$

마지막 라운드 함수인 Round_{32} 에서는 바이트 들을 섞지 않는다. 따라서 입력값 $X_{31}=X_{31,7}\parallel\cdots\parallel X_{31,0}$ 으로부터 출력값 $X_{32}=X_{32,7}\parallel\cdots\parallel X_{32,0}$ 은 다음과 같이 계산되며 [그림 3]과 같다.

$$\begin{split} &X_{32,i} = X_{31,i} \,, \ i = 0, \, 2, \, 4, \, 6 \\ &X_{32,1} = X_{31,1} \boxplus (F_1(X_{31,0}) \oplus \mathrm{SK}_{124}) \\ &X_{32,3} = X_{31,3} \oplus (F_0(X_{31,2}) \boxplus \mathrm{SK}_{125}) \\ &X_{32,5} = X_{31,5} \boxplus (F_1(X_{31,4}) \oplus \mathrm{SK}_{126}) \\ &X_{32,7} = X_{31,7} \oplus (F_0(X_{31,6}) \boxplus \mathrm{SK}_{127}) \end{split}$$



[그림 3] HIGHT의 32번째 라운드 함수

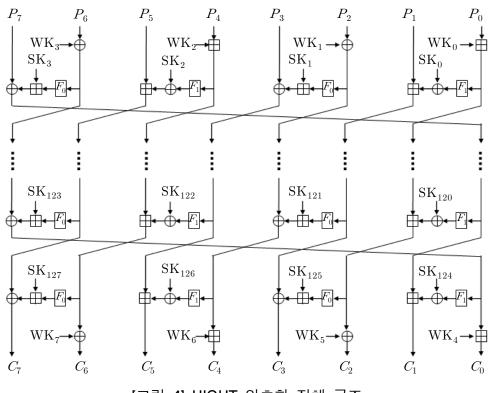
4.3.3. 암호화 최종변환

HIGHT의 최종변환은 $\operatorname{Round}_{32}$ 의 출력인 $X_{32}=X_{32,7}\parallel\cdots\parallel X_{32,0}$ 을 4개의 화이트닝키 WK_4 , WK_5 , WK_6 , WK_7 을 이용하여 변환한다.

$$\begin{split} &C_i = X_{32,i}\,,\ i = 1,\,3,\,5,7\\ &C_0 = X_{32,\,0} \boxplus \mathrm{WK}_4\\ &C_2 = X_{32,\,2} \oplus \mathrm{WK}_5\\ &C_4 = X_{32,\,4} \boxplus \mathrm{WK}_6\\ &C_6 = X_{32,\,6} \oplus \mathrm{WK}_7 \end{split}$$

4.3.4. 암호화 전체 구조

암호화는 키 스케줄을 통해 생성된 서브키와 평문 블록을 초기 변환, 라운드 함수 32회 및 최종 변환으로 구성되어 있다.



[그림 4] HIGHT 암호화 전체 구조

4.4. 복호화

HIGHT의 복호화 알고리즘은 암호화 알고리즘과 마찬가지로 라운드키 생성과정, 초기변환(암호화 최종변환의 역변환), 라운드 함수 32회 반복, 최종변환(암호화 초기변환의 역변환)으로 구성되어 있다. 복호화 알고리즘에서의 중간값은 $X'_{i} = X'_{i,7} \parallel \cdots \parallel X'_{i,0}, i = 0, \cdots, 32$ 로 표시된다.

4.4.1. 복호화 초기변환

복호화 알고리즘의 초기 변환은 암호화 알고리즘의 최종 변환에 대한 역변환이다. 이 함수는 4개의 화이트닝키 $WK_{4,}WK_{5,}WK_{6,}WK_{7}$ 를 이용하여 암호문 C를 복호화알고리즘의 첫 번째 라운드 함수의 입력값 $X'_{0}=X'_{0,7}\parallel\cdots\parallel X'_{0,0}$ 으로 다음과 같이변환하다.

$$X'_{0, i} = C_i, i = 1, 3, 5, 7$$

$$X'_{0, 0} = C_0 \boxminus WK_4$$

$$X'_{0, 2} = C_2 \oplus WK_5$$

$$X'_{0, 4} = C_4 \boxminus WK_6$$

$$X'_{0, 6} = C_6 \oplus WK_7$$

5.4.2. 복호화 라운드 함수

복호화 알고리즘의 i번째 라운드 함수를 Round' $_i,\ i=1,\cdots,32$ 로 표시한다. i번째 라운드 함수 Round' $_i$ 는 $X'_{i-1}=X'_{i-1,7}\parallel\cdots\parallel X'_{i-1,0}$ 를 $X'_{i}=X'_{i,7}\parallel\cdots\parallel X'_{i,0}$ 로 다음과 같이 변환한다.

$$\begin{split} &X'_{i,j} = X'_{i-1,j+1 \bmod 8}, \ j=1, 3, 5, 7 \\ &X'_{i,0} = X'_{i-1,1} \boxminus (F_1(X'_{i-1,0}) \oplus \text{SK}'_{4i-1}) \\ &X'_{i,2} = X'_{i-1,3} \oplus (F_0(X'_{i-1,2}) \boxplus \text{SK}'_{4i-2}) \\ &X'_{i,4} = X'_{i-1,5} \boxminus (F_1(X'_{i-1,4}) \oplus \text{SK}'_{4i-3}) \\ &X'_{i,6} = X'_{i-1,7} \oplus (F_0(X'_{i-1,6}) \boxplus \text{SK}'_{4i-4}) \end{split}$$

암호화 알고리즘에서와 마찬가지로, 마지막 라운드 함수인 Round'32에서는 바이트들을 섞지 않는다. 따라서 출력이 다음과 같이 연산된다.

$$\begin{split} &X'_{32,i} = X'_{31,i} \;,\; i = 0,\, 2,\, 4,\, 6 \\ &X'_{32,1} = X'_{31,1} \boxminus (F_1(X'_{31,0}) \oplus \text{SK}'_{127}) \\ &X'_{32,3} = X'_{31,3} \oplus (F_0(X'_{31,2}) \boxplus \text{SK}'_{126}) \\ &X'_{32,5} = X'_{31,5} \boxminus (F_1(X'_{31,4}) \oplus \text{SK}'_{125}) \\ &X'_{32,7} = X'_{31,7} \oplus (F_0(X'_{31,6}) \boxplus \text{SK}'_{124}) \end{split}$$

4.4.3. 복호화 최종 변환

복호화 알고리즘의 최종 변환은 암호화 알고리즘에서의 초기 변환의 역 변환에 해당한다. 이 함수는 네 개의 화이트닝키 WK_0, WK_1, WK_2, WK_3 를 이용하여 마지막 라운드 함수 $Round'_{32}$ 의 출력값인 $X'_{32} = X'_{32,7} \parallel \cdots \parallel X'_{32,0}$ 을 평문 P로 변환한다.

$$\begin{split} P_i &= {X'}_{32,\,i}\,,\ i=1,\,3,\,5,7 \\ P_0 &= {X'}_{32,\,0} \boxminus \text{WK}_0 \\ P_2 &= {X'}_{32,\,2} \oplus \text{WK}_1 \\ P_4 &= {X'}_{32,\,4} \boxminus \text{WK}_2 \\ P_6 &= {X'}_{32,\,6} \oplus \text{WK}_3 \end{split}$$