情報保護學會論文誌
第18卷 第4號, 2008. 8

스트림 암호 Grain-v1에 대한 연관키 공격*

이 유 섭[1†], 정 기 태[1], 성 재 철[2], 홍 석 희[1‡]
[1]고려대학교 정보보호기술 연구센터, [2]서울시립대 수학과

The Related-Key Attack on Grain-v1*

Yuseop Lee[1†], Kitae Jung[1], Jaechul Sung[2], Seokhie Hong[1‡]
[1]Center for Information Security Technologies, Korea University,
[2]Department of Mathematics, University of Seoul

요 약

Kücük은 eSTREAM 프로젝트에 제안된 스트림 암호인 Grain-v1에 대하여, 공격자에게 키스트림 수열이 주어지면 이를 1-비트 좌측 이동시킨 키스트림 수열을 생성할 수 있음을 보였다[5]. 본 논문에서는 [5]에서 제안된 공격을 확장하여 연관키 선택 *IV* 공격을 제안한다. 본 공격은 Grain-v1의 초기화 과정의 취약점을 이용하여 비밀키의 부분 정보를 획득하여 $2^{25.02}$개의 *IV*와 2^{56}의 시간 복잡도로 Grain-v1의 비밀키를 복구한다. 본 공격은 Grain-v1에 대한 첫 번째 키 복구 공격이다.

ABSTRACT

The slide resynchronization attack on Grain-v1 was proposed in [5]. Given the keystream sequence, this attack can generate the 1-bit shifted keystream sequence generated by Grain-v1. In this paper, extending the attack proposed in [5], we propose the key recovery attack on Grain-v1 using the related-key. Using the weakness of the initialization procedure of Grain-v1, this attack recover the master key with $2^{25.02}$ *IV*s and 2^{56} time complexity. This attack is the first known key recovery attack on Grain-v1.

Keywords : Stream cipher, Grain-v1, Related-key attack, Cryptanalysis

Ⅰ. 서 론

Grain[2]은 유럽에서 진행 중인 eSTREAM 프로젝트에 제안된 하드웨어 기반 스트림 암호로 LFSR과 NFSR 그리고 부울 함수 h로 구성된 단순한 구조로 설계되었다. Grain의 초기 버전에 대해서는 구별 공격[4]과 키 복구 공격[1]이 각각 제안되어 알고리즘의 취약점이 노출되었다. 이 후 Grain의 설계자들은 이런 취약점을 보완한 버전인 Grain-v1[3]을 제안하였다. 현재 Grain-v1은 Phase 3에 선정되어 안전성을 평가 받고 있으며 단순한 구조로 설계되었으면서도 충분한 안전성을 가지는 것으로 드러나 Phase 3에 선정된 다른 하드웨어 기반 스트림 암호들보다 많은 주목을 받고 있다. Grain-v1에 대한 분석 결과로는 Kücük이 제안한 Slide Resynchronization 공격[5]이 유일하다.

[5]에서 제안된 공격은 Resynchronization 공격의 일종으로 공격자에게 키스트림 수열이 주어지면, 이를 1-

접수일 : 2008년 1월 22일; 수정일 : 2008년 4월 20일;
채택일 : 2008년 5월 22일
* 본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음 (IITA-2008-(C1090-0801-0025))
† 주저자, yusubi@cist.korea.ac.kr
‡ 교신저자, hsh@cist.korea.ac.kr

비트 좌측 이동시킨 키스트림 수열을 생성할 수 있음을 보였다. 본 논문에서는 이 공격을 확장하여 Grain-v1에 대한 연관키 선택 *IV* 공격을 제안한다. 본 공격은 Grain-v1에 대한 첫 번째 키 복구 공격으로 2개의 좌측 순환 이동된 연관키를 사용할 때 발생하는 다음과 같은 취약점을 이용한다. 첫째, Grain-v1의 초기화 과정에서 a-비트 좌측 순환 이동된 연관키에 대해 생성되는 내부 상태값이 동일해지는 연관된 *IV* 쌍이 존재하고, 이 때 만족해야 하는 내부 상태값에 대한 식이 존재한다. 또한, 초기화 과정에서 비밀키와 *IV*가 각각의 레지스터에 직접 입력되기 때문에 내부 상태값에 대한 식은 비밀키와 *IV*에 대한 식으로 변형된다. 둘째, 5-변수 부울 함수 h는 2개의 입력값을 고정하여 2-변수 선형 함수로 변형되어 1개의 변수를 제외시킬 수 있다. 이런 취약점을 이용하여 a-비트 비밀키를 복구하고, *IV*와 비밀키에 대한 a개의 선형 방정식을 구성하여 a-비트 비밀키 정보를 추가로 얻는다. 이를 이용하여 키 전수조사의 시간 복잡도를 2^{80-2a}로 낮춘다($a=1,2,\cdots,12$). 따라서 최소 2^{56}의 시간 복잡도로 비밀키를 복구한다.

본 논문은 다음과 같이 구성되어 있다. 2장에서 Grain-v1과 [5]에서 제안된 공격을 간략히 소개한다. 3장에서 Grain-v1이 가지는 특성을 분석하고 이를 이용한 키 복구 공격을 제안한다. 마지막으로 4장에서 결론을 맺는다.

Ⅱ. 기본 이론

본 장에서는 Grain-v1과 [5]에서 제안된 공격을 소개한다. 본 논문에서는 다음과 같은 표기를 사용한다.

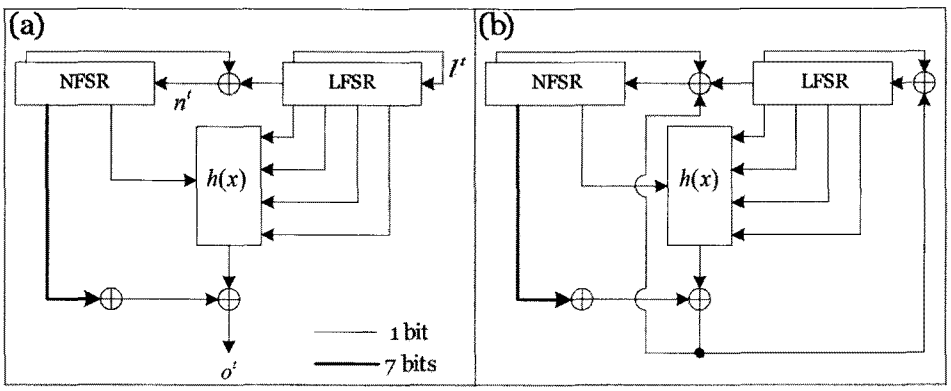
- $S^t=(S_0^t,S_1^t,\cdots,S_{160}^t)$: 시간 t일 때, 전체 내부 상태값.
- $N^t=(N_0^t,N_1^t,\cdots,N_{79}^t)$: 시간 t일 때, NFSR의 내부 상태값.
- $L^t=(L_0^t,L_1^t,\cdots,L_{79}^t)$: 시간 t일 때, LFSR의 내부 상태값.
- n^t : 시간 t일 때, NFSR의 갱신 비트.
- l^t : 시간 t일 때, LFSR의 갱신 비트.
- o^t : 시간 t일 때, Grain-v1의 출력 비트.
- $z=(z_0,z_1,\cdots)$: 키스트림 수열 ($z_t=o^{t+160}$).
- $z[a]$: 키스트림 수열 z를 a 비트 좌측 이동시킨 수열. 즉, $z[a]=(z_a,z_{a+1},\cdots)$.

2.1 Grain-v1

Grain-v1은 80-비트 비밀키 $K=(k_0,\cdots,k_{79})$와 64-비트 $IV=(iv_0,\cdots,iv_{63})$를 사용하는 스트림 암호로서 전체 구조는 [그림 1]과 같이 LFSR, NFSR, 부울 함수 $h(x)$로 구성된다. Grain-v1에 사용되는 NFSR과 LFSR은 각각 80-비트 내부 상태값을 갖는다. LFSR과 NFSR의 갱신 비트 l^t,n^t는 각각 다음과 같다.

$$\begin{aligned}n^t = {} & L_0^t \oplus N_{62}^t \oplus N_{60}^t \oplus N_{52}^t \oplus N_{45}^t \oplus N_{37}^t \oplus N_{33}^t \\ & \oplus N_{28}^t \oplus N_{21}^t \oplus N_{14}^t \oplus N_9^t \oplus N_0^t \oplus N_{63}^t N_{60}^t \\ & \oplus N_{37}^t N_{33}^t \oplus N_{15}^t N_9^t \oplus N_{60}^t N_{52}^t N_{45}^t \oplus N_{33}^t N_{28}^t N_{21}^t \\ & \oplus N_{63}^t N_{45}^t N_{28}^t N_9^t \oplus N_{60}^t N_{52}^t N_{37}^t N_{33}^t \\ & \oplus N_{63}^t N_{60}^t N_{21}^t N_{15}^t \oplus N_{63}^t N_{60}^t N_{52}^t N_{45}^t N_{37}^t \\ & \oplus N_{33}^t N_{28}^t N_{21}^t N_{15}^t N_9^t \oplus N_{52}^t N_{45}^t N_{37}^t N_{33}^t N_{28}^t N_{21}^t .\end{aligned}$$

$$l^t = L_{62}^t \oplus L_{51}^t \oplus L_{38}^t \oplus L_{23}^t \oplus L_{13}^t \oplus L_0^t .$$



[그림 1] (a) Grain-v1 키스트림 생성 과정 (b) Grain-v1 초기화 과정

부울 함수 $h(x)$는 LFSR의 4-비트 내부 상태값과 NFSR의 1-비트 내부 상태값을 입력 받는 함수로 다음과 같다.

$$h(L_3^t, L_{25}^t, L_{46}^t, L_{64}^t, N_{63}^t) = L_{25}^t \oplus N_{63}^t \oplus L_3^t L_{64}^t \oplus L_{46}^t L_{64}^t \oplus L_{64}^t N_{63}^t \oplus L_3^t L_{25}^t L_{46}^t \oplus L_3^t L_{46}^t L_{64}^t \oplus L_3^t L_{64}^t N_{63}^t \oplus L_{25}^t L_{46}^t N_{63}^t \oplus L_{46}^t L_{64}^t N_{63}^t.$$

출력 비트 o^t는 다음과 같이 생성된다. 여기서 $A = \{1,2,4,10,31,43,56\}$이다.

$$o^t = \sum_{k \in A} N_k^t \oplus h(L_3^t, L_{25}^t, L_{46}^t, L_{64}^t, N_{63}^t).$$

Grain-v1의 초기화 과정은 다음과 같다 ([그림 1-(b)] 참조).

① NFSR의 초기 상태값에 비밀키 K를 로딩한다: $N_i^0 = k_i (0 \le i \le 79)$.

② LFSR의 초기 상태값에 IV를 로딩한다:
$$\begin{cases} L_i^0 = iv_i & (0 \le i \le 63) \\ L_i^0 = 1 & (64 \le i \le 79) \end{cases}.$$

③ 160번의 공회전을 수행한다. 공회전을 수행하는 동안의 출력 비트 o^i는 키스트림 비트로 출력되지 않고, LFSR과 NFSR의 갱신비트에 각각 XOR된다.

2.2 Kücük의 Slide Resynchronization 공격

Kücük의 Resynchronization 공격은 초기화 과정의 특성을 이용하여 (K,IV)에 의해 생성된 키스트림 수열을 1-비트 좌측 이동시킨 키스트림 수열을 생성하는 $(\tilde{K},\tilde{IV}) = (\tilde{k}_0,\cdots,\tilde{k}_{79},\tilde{iv}_0,\cdots,\tilde{iv}_{63})$쌍이 2^{-2}의 확률로 존재함을 보인다. 이 공격은 다음 두 가지 사실을 이용한다.

① 1 클럭에 갱신되는 비트 수는 2이다.

② 초기화 과정은 키스트림 생성 과정과 매우 유사하다.

시간 t에서 (K,IV), $(\tilde{K},\tilde{IV})$에 의해 생성되는 내부 상태값을 각각 S^t, $\tilde{S}^t$라 할 때, $S^1 = \tilde{S}^0$을 만족하면 $S^{t+1} = \tilde{S}^t (t = 0,\cdots,159)$이 성립한다. 이 후 S_{161}은 키 생성 과정에서의 갱신 함수에 의해 갱신되는 반면, $\tilde{S}^{160}$은 초기화 과정에서의 갱신 함수를 이용하여 갱신된다. 따라서 S_{160}에서 생성된 키스트림 비트 $z_0 = 0$이면, $S^{161} = \tilde{S}^{160}$이고 $z_{t+1} = \tilde{z}_t (t = 0,1,\cdots)$이다. 한편, $S^1 = \tilde{S}^0$을 만족하는 $\tilde{S}^0$이 존재하기 위해서는 $l^0 = 1$을 만족해야 한다. 그러므로 $l^0 = 1$과 $z_0 = 0$을 만족하는 IV가 존재하는 확률이 이 공격의 성공 확률이다. 여기서 l^0와 z_0가 랜덤한 분포를 따른다고 가정하면, (K,IV)에 의해 생성된 키스트림 수열을 한 비트 좌측 이동시킨 키스트림 수열을 생성하는 $(\tilde{K},\tilde{IV})$가 존재할 확률은 2^{-2}이다.

Ⅲ. Grain-v1에 대한 키 복구 공격

본 장에서는 [5]에서 제안된 공격을 확장하여 Grain-v1에 대한 연관키 선택 IV 공격을 제안한다. 이 공격은 2개의 연관키를 이용하여 최소 2^{56}의 시간 복잡도로 64-비트 비밀키를 복구할 수 있다.

3.1 Grain-v1의 초기화 과정 분석

Kücük은 1-비트 좌측 순환 이동시킨 연관키에 대해 초기화 과정에서 $S^1 = \tilde{S}^0$을 만족하여 1-비트 좌측 이동된 키스트림 수열이 생성되는 IV와 $\tilde{IV}$가 높은 확률로 존재함을 보였다. 본 절에서는 이를 확장하여 a-비트 좌측 순환된 연관키를 사용할 때 키스트림 수열의 처음 16 비트에 발생하는 특성을 분석한다. 본 논문에서 사용하는 연관키 $\tilde{K}$와 연관 $\tilde{IV}$ 는 다음과 같다. 여기서, a는 1이상이고 12이하인 정수이다.

$$K = (k_0,k_1,\cdots,k_{79}) \Rightarrow \tilde{K} = (k_a,k_{a+1},\cdots,k_{a-1}).$$
$$IV = (iv_0,iv_1,\cdots,iv_{63}) \Rightarrow \tilde{IV} = (iv_a,iv_{a+1},\cdots,iv_{63},1,\cdots,1).$$

(K,IV)와 $(\tilde{K},\tilde{IV})$에 의한 내부 상태값을 각각 S와 $\tilde{S}$로 표기하고, 이 때 생성되는 키스트림 수열을 각각 z와 $\tilde{z}$으로 표기한다. 두 내부 상태값 S와 $\tilde{S}$는 다음과 같은 특성을 가진다. 전개상의 편의를 위하여 $(iv_{64},\cdots,iv_{79}) = (1,\cdots,1)$을 가정한다.

[특성 1] $a \le i \le 160$인 i에 대하여 $S^a = \tilde{S}^0$이면, $S^i = \tilde{S}^{i-a}$이다. 다시 말해 o^i와 $\tilde{o}^{i-a}$가 동일하다.

증명) Grain-v1의 초기화 과정에서 160번의 공회전을 수행하므로 이 특성이 성립한다.

비밀키 K에 대해 $S^a = \tilde{S}^0$를 만족하는 IV를 유효한 IV로 정의한다. 유효한 IV는 다음과 같은 특성을 가진다.

[특성 2] 유효한 IV는 다음 식 (1)과 식 (2)를 만족한다.

$$n^i \oplus o^i = \tilde{S}^0_{80-a+i} \Leftrightarrow n^i \oplus o^i = \tilde{k}_{80-a+i} = k_i \ (0 \le i < a) \quad (1)$$

$$l^i \oplus o^i = 1. \quad (2)$$

[특성 2]는 Grain-v1의 초기화 과정에 갱신되는 비트 n^i, l^i, o^i에 대한 식에 의해 알 수 있다. 여기서 식 (2)는 비밀키에 관한 선형식이기 때문에 유효한 IV가 주어지면 이를 식 (2)에 대입하여 a 개의 비밀키에 관한 선형식을 구성할 수 있다. 다음 [정리 1]은 유효한 IV를 키스트림 수열을 비교하여 추측할 수 있음을 보인다.

[정리 1] z의 처음 b 비트가 0일 때, $S^a = \tilde{S}^0$이면 $z_{a+i} = \tilde{z}_i (0 \le i < 16-a+b)$이다. 그리고 두 키스트림 수열 z와 $\tilde{z}$이 $z_{a+i} = \tilde{z}_i (0 \le i < 16-a+b)$을 만족하는 경우를 $z \sim_a \tilde{z}$로 표시할 때, $z \sim_a \tilde{z}$이지만 유효한 IV가 아닐 확률은 다음과 같이 $2^{-(16-a+b)}$이고, 평균 $2^{-(18-a)}$이다.

증명) $S^a = \tilde{S}^0$이면 [특성 1]에 의해 $S^{160} = \tilde{S}^{160-a}$이다. $z_t = 0 \ (0 \le t < b)$이면 키스트림 생성 과정과 초기화 과정의 갱신 방법이 동일하므로 $S^{160+b} = \tilde{S}^{160-a+b}$ 이다. 이 후 $(a-b)$번 클럭된 두 내부 상태값 S^{160+a}와 $\tilde{S}^{160}$은 각각의 LFSR과 NFSR에서 마지막으로 갱신된 $(a-b)$ 비트를 제외한 나머지 부분은 동일하다. 즉, $N_i^{160+a} = \tilde{N}_i^{160}$, $L_i^{160+a} = \tilde{L}_i^{160}$ $(0 \le i < 80-a+b)$이다. 서로 다른 $2(a-b)$ 비트가 키스트림 생성 과정에서 사용되기 전까지 동일한 $(16-a+b)$-비트 키스트림 수열이 생성된다. z와 $\tilde{z}$의 키스트림 비트가 통계적으로 랜덤하다고 가정하면, $z \sim_a \tilde{z}$이지만 $S^a \neq \tilde{S}^0$일 확률은 랜덤한 $(16-a+b)$ 비트가 같을 확률이므로 $2^{-(16-a+b)}$이다. 여기서 b의 기대값은 $2(=1 \cdot \frac{1}{2} 2 \cdot \frac{1}{2^2} + 3 \cdot \frac{1}{2^3} + \cdots)$ 이 되고 이 확률의 평균은 $2^{-(18-a)}$가 된다.

IV에 대해 $z \sim_a \tilde{z}$를 검사하는 것을 *D-Test*로 정의한다. [정리 2]는 유효한 IV의 짝수 개의 비트에 보수를 취하여 다른 유효한 IV를 구할 수 있음을 보인다.

[정리 2] 유효한 IV의 짝수 개의 비트를 보수로 변경하여 유효한 $IV^* = (iv_0^*, \cdots, iv_{63}^*)$를 생성할 수 있다. 또한, 원래의 가 유효하지 않으면 동일한 방법으로 변경한 IV*도 유효하지 않다.

증명) 식 (2)에서 l^i는 다음과 같이 표현된다.

$$l^i = iv_i \oplus iv_{i+13} \oplus iv_{i+23} \oplus iv_{i+38} \oplus iv_{i+51} \oplus iv_{i+62}.$$

위의 식은 IV에 관한 선형식이기 때문에 짝수 개의 비트에 보수를 취해도 동일한 값을 가지게 된다. 이렇게 변형된 비트가 식 (1)과 식 (2)의 다른 부분에 영향을 주지 않으면 식 (1)과 식 (2)는 모두 성립한다. 그러므로 이런 IV*은 유효하다. 그리고 동일하게 IV가 유효하지 않은 경우 IV*도 유효하지 않다.

[표 1]은 [정리 2]가 만족하도록 IV를 변경하는 비트의 위치를 나타낸다. [따름정리 1]은 키스트림을 비교하여 유효한 IV를 옳게 추측할 수 있을 확률을 1에 근사할 수 있음을 보인다.

〔표 1〕 IV^* 생성시 IV 를 변경하는 비트의 위치

	변경하는 비트의 위치
IV_1^*	iv_{15}, iv_{40}
IV_2^*	iv_{16}, iv_{41}
IV_3^*	iv_{17}, iv_{42}
IV_4^*	iv_{18}, iv_{43}
IV_5^*	$iv_{15}, iv_{40}, iv_{16}, iv_{41}$
IV_6^*	$iv_{15}, iv_{40}, iv_{17}, iv_{42}$
IV_7^*	$iv_{15}, iv_{40}, iv_{18}, iv_{43}$
IV_8^*	$iv_{16}, iv_{41}, iv_{17}, iv_{42}$

[따름정리 1] IV와 [정리 2]를 통해 변형한 $(c-1)$개의 IV*이 모두 *D-Test*를 통과할 때, IV가 유효하지 않을 확률은 $2^{-c(18-a)}$이다.

증명) IV가 유효하지 않은 경우 모든 IV*가 모두 유효하지 않으므로 각각의 IV*가 *D-Test*를 통과하는 것은 독립이다. 그러므로 *D-Test*를 통과하지만 IV가 유효하지 않을 확률이 $2^{c(18-a)}$이므로 IV와 $(c-1)$개의 IV*가 모두 *D-Test*를 통과하지만 IV가 유효하지 않을 확률은 $2^{-c(18-a)}$이다.

3.2 부울 함수 h의 분석

h는 10개의 항을 가지는 3차 5-변수 부울 함수이다. 하지만 입력값의 일부가 고정되면 [표 2]와 같은 단순한 형태로 변형된다. 본 절에서는 h함수의 취약점을 이용하여 유효한 IV가 주어지면 a 비트 부분키를 복구할 수 있음을 보인다.

〔표 2〕 입력값에 따른 h의 변형

변형된 h	고정되는 입력값	출력값
h_1	$L_{46}^t=0, L_{64}^t=1$	$L_3^t \oplus L_{25}^t$
h_2	$L_3^t=L_{25}^t, L_{46}^t=1, L_{64}^t=1$	$N_{63}^t \oplus 1$

[정리 3] $2a$-비트 $iv_{12+1},\cdots,iv_{12+a},iv_{24+1},\cdots,iv_{24+a}$는 모든 값을 선택하고 $iv_{45+1},\cdots,iv_{45+a}$는 0으로 선택하고 이외의 비트는 모두 같은 값을 가지도록 선택한 2^{2a}개의 IV의 집합에는 유효한 IV가 정확히 1개 존재한다.

증명) Grain-v1의 초기화 과정에서 IV의 값이 LFSR에 직접 입력되고, LFSR의 마지막 16 비트는 모두 1이므로 $L_{46}^i=0, L_{64}^i=1 (0\le i<a)$가 되어 h는 h_1의 형태가 되므로 식 (1)과 식 (2)는 다음과 같이 변형 된다.

$$iv_i \oplus iv_{i+3} \oplus iv_{i+25} = k_i \oplus (n^i \oplus iv_i) \oplus \sum_{j\in A} k_{i+j}. \quad (3)$$

$$iv_i \oplus iv_{i+13} \oplus iv_{i+23} \oplus iv_{i+38} \oplus iv_{i+51} \oplus iv_{i+62} \oplus iv_{i+3} \oplus iv_{i+25} = 1 \oplus \sum_{j\in A} k_{i+j}. \quad (4)$$

식 (3)의 우변은 비밀키에 대한 식이므로 선택한 2^{2a}개의 IV에 대해 고정된 값을 가진다. 그리고 iv_{i+3}은 2^{2a}개의 IV에 대해 같은 값을 가지고 iv_{i+25}는 모든 값을 가지므로 식 (5)를 만족하는 2^a개의 IV가 식 (3)을 만족한다.

$$iv_{i+25} = iv_{i+3} \oplus k_i \oplus n^i \oplus \sum_{j\in A} k_{i+j}. \quad (5)$$

식 (5)의 좌변의 값은 2^{2a}개의 IV 모두 고정된 값이므로 식 (5)를 만족하는 2^a개의 IV의 iv_{i+25}는 모두 같다. 그러므로 이 2^a개의 IV는 모두 iv_{i+13}이 모든 값을 가지고 나머지 비트들은 같은 값을 가지므로 식 (6)을 만족하는 IV가 정확히 1개 존재한다.

$$iv_{i+13} = 1 \oplus \sum_{j\in A} k_{i+j} \oplus iv_i \oplus iv_{i+23} \oplus iv_{i+38} \oplus iv_{i+51} \oplus iv_{i+62} \oplus iv_{i+3} \oplus iv_{i+25}. \quad (6)$$

IV가 식 (3)과 식 (4)를 만족하는 것과 식 (5)와 식 (6)을 만족하는 것이 동치이므로 선택한 2^{2a}개의 IV중에서 유효한 IV가 정확히 1개 존재한다.

[정리 3]에 해당하는 2^{2a}개의 IV의 집합을 선택한 후, [따름정리 1]을 이용하여 유효한 IV를 결정한다. 그리고 이 IV를 식 (4)에 대입하면 비밀키에 관한 a개의 선형식을 구성할 수 있다. [정리 4]는 [표 2]의 h_2와 유효한 IV를 이용하여 a-비트 비밀키 $k_{63},\cdots,k_{62+a}$를 복구할 수 있음을 보인다.

[정리 4] [정리 3]에 의해 결정되는 유효한 IV에서 [표 3]에서 나타내는 비트를 변경한 IV를 $IV_t^{\triangle}$ $(0\le t<a)$라 표기한다. $IV_t^{\triangle}$가 유효하면 $k_{63+t}=iv_{3+t}\oplus iv_{25+t}\oplus 1$이고, 그렇지 않으면 $k_{63+t}=iv_{3+t}\oplus iv_{25+t}$이다. [표 3]에서 $IV_t^{\triangle}$를 생성할 때, iv_j^c는 $iv_{3+t}\neq iv_{25+t}$인 경우만 변경한다.

증명) 식 (3)과 식 (4)는 각각 a개의 식으로 구성되

〔표 3〕 $IV_i^{\triangle}$ 생성시 IV를 변경하는 비트의 위치

	변경하는 비트 위치
$IV_0^{\triangle}$	$iv_{46}, iv_{59}, iv_{25}^c, iv_{15}^c$
$IV_1^{\triangle}$	$iv_{47}, iv_{60}, iv_{26}^c, iv_{16}^c$
$IV_2^{\triangle}$	$iv_{48}, iv_{61}, iv_{27}^c, iv_{17}^c$
$IV_3^{\triangle}$	$iv_{49}, iv_{62}, iv_{38}, iv_{28}^c, iv_{18}^c$
$IV_4^{\triangle}$	$iv_{50}, iv_{29}^c, iv_{19}^c$
$IV_5^{\triangle}$	$iv_{51}, iv_{38}, iv_{30}^c, iv_{20}^c$
$IV_6^{\triangle}$	$iv_{52}, iv_{39}, iv_{31}^c, iv_{21}^c$
$IV_7^{\triangle}$	$iv_{53}, iv_{40}, iv_{32}^c, iv_{22}^c$
$IV_8^{\triangle}$	$iv_{54}, iv_{41}, iv_{33}^c, iv_{37}^c, iv_{61}^c$
$IV_9^{\triangle}$	$iv_{55}, iv_{42}, iv_{34}^c, iv_{38}^c, iv_{62}^c$
$IV_{10}^{\triangle}$	$iv_{56}, iv_{43}, iv_{35}^c$
$IV_{11}^{\triangle}$	$iv_{57}, iv_{44}, iv_{36}^c$

어 있다. $i=t$일 때의 식을 각각 식 (3)$-t$와 식 (4)$-t$라 할 때, IV의 iv_{46+t}에 보수를 취하고 $iv_{3+t} \neq iv_{25+t}$인 경우에 iv_{25+t}에 보수를 취하면 $L_3^t = L_{25}^t, L_{46}^t = 1, L_{64}^t = 1$이 되어 식 (3)$-t$와 식 (4)$-t$에서 h는 h_2의 형태로 변형되어 식 (3)$-t$와 식 (4)$-t$는 다음과 같이 변형된다.

$$iv_t \oplus (k_{t+63} \oplus 1) = k_t \oplus (n^t \oplus iv_t) \oplus \sum_{j \in A} k_{t+j}.$$

$$iv_t \oplus iv_{t+13} \oplus iv_{t+23} \oplus iv_{t+38} \oplus iv_{t+51} \oplus iv_{t+62} \oplus (k_{t+63} \oplus 1) = 1 \oplus \sum_{j \in A} k_{t+j}.$$

이 때, 보수를 취한 iv_{46+t}와 iv_{25+t}가 식 (3)$-t$와 식 (4)$-t$를 제외한 다른 (2a-2)개의 식의 좌변의 값을 바꾸는 경우, 같은 식에 위치하는 IV의 비트에 보수를 취하여 좌변의 값이 동일한 값을 가지도록 변경할 수 있다. 그리고 이 때 변경된 비트가 또 다른 식에 영향을 주는 경우, 다시 같은 식에 위치하는 IV의 비트에 보수를 취하여 좌변의 값이 동일하도록 변경하는 작업을 반복하여 식 (3)$-t$와 식 (4)$-t$를 제외한 (2a-2)개의 식의 좌변의 값은 동일해지도록 $IV_t^{\triangle}$를 생성할 수 있다. [표 3]의 변경하는 비트들은 이를 만족한다. 그러므로 IV가 유효하면 $IV_t^{\triangle}$는 식 (3)$-t$와 식 (4)$-t$를 제외한 (2a-2)개의 식을 반드시 만족하고, 식 (3)$-t$와 식 (4)$-t$는 k_{63+t}의 값에 따라 만족하는지가 결정된다. 다시 말해서 $IV_t^{\triangle}$가 유효하면 식 (3)$-t$와 식 (4)$-t$을 만족하므로 $k_{63+t} = iv_{3+t} \oplus iv_{25+t} \oplus 1$이고, $IV_s^{\triangle}$가 유효하지 않으면 식 (3)$-t$와 식 (4)$-t$가 만족하지 않으므로 $k_{63+t} = iv_{3+t} \oplus iv_{25+t}$이다.

3.3 키 복구 공격 알고리즘

본 절에서는 $a=12$일 때의 공격 알고리즘을 소개한다. a가 12미만인 경우에도 유사한 방법으로 공격이 가능하다. 본 공격 알고리즘은 IV의 유효성을 검사하는 IsValidIV 알고리즘과 유효한 IV를 이용하여 a-비트 부분키를 복구하는 키 결정 알고리즘으로 구성된다. IsValidIV 알고리즘은 [표 4]와 같다. $a=12$이므로 1개의 유효한 IV만을 결정하기 위해서 $2^{24-c\times(18-12)} \ll 1$이 되도록 하는 $c=5$를 사용한다.

〔표 4〕 IsValidIV 검사 알고리즘 (a=12, c=5)

1. 입력된 IV가 *D-Test*를 통과하면 다음을 수행 하고 그렇지 않으면 유효하지 않은 IV임을 반환.
2. $i=1$로 설정하고 $i<5(=c)$이면 다음을 수행.
   2-1. IV_i^*를 생성하여 *D-Test*를 통과하지 않으면 유효하지 않은 IV임을 반환 ([표 1] 참조).
   2-2. i에 1을 더한다.
3. 유효한 IV임을 반환한다.

유효한 IV를 입력받아 $k_{63}, \cdots k_{74}$를 복구하는 키 결정 알고리즘은 [표 5]와 같다.

〔표 5〕 키 결정 알고리즘 (a=12, c=5)

1. $i=0$으로 설정한다.
2. $i<12$이면 다음을 수행한다.
   2-1. $IV_i^{\triangle}$를 생성한다 ([표 3] 참조).
   2-2. IsValisIV를 이용하여 $IV_i^{\triangle}$가 유효한지 검사.
   2-3. $IV_i^{\triangle}$가 유효하면 $k_{63+i} = iv_{3+i} \oplus iv_{25+i} \oplus 1$ 로 결정하고, 유효하지 않으면 $k_{63+i} = iv_{3+i} \oplus iv_{25+i}$로 결정한다.
   2-4. i에 1을 더한다.

본 논문에서 제안하는 키 복구 공격 알고리즘은 [표 6]과 같다.

〔표 6〕 키 복구 공격 알고리즘

1. 공격자는 $iv_{13} \sim iv_{36}$의 모든 값을 선택하고 나머지 비트는 0으로 고정한 2^{24}개의 IV를 선택한다.
2. IsValidIV 알고리즘을 이용하여 유효한 IV를 추측한다.
3. 단계 2에서 추측한 유효한 IV를 키 결정 알고리즘을 입력하여 $k_{63} \sim k_{74}$를 복구한다.
4. 식 (4)에 단계 2에서 추측한 IV를 대입하여 12개의 비밀키에 관한 선형식을 구성한다.
5. 단계 3에서 복구한 비밀키와 단계 4에서 구성한 선형식을 이용하여 비밀키를 전수조사 한다.

본 공격에는 $2\left(\sum_{i=0}^{c-1} 2^{2a-i(18-a)}\right) + ac$개의 IV가 필요하

〔표 7〕 a와 c에 따른 공격 복잡도

a	c	성공 확률	계산 복잡도	필요한 IV 개수
1	1	99.99%	2^{78}	$2^{3.17}$
	2	99.99%	2^{78}	$2^{3.32}$
4	1	99.99%	2^{72}	$2^{9.01}$
	2	99.99%	2^{72}	$2^{9.02}$
8	2	99.99%	2^{64}	$2^{17.00}$
	3	99.99%	2^{64}	$2^{17.00}$
12	5	99.99%	2^{56}	$2^{25.02}$
	7	99.99%	2^{56}	$2^{25.02}$

〔표 8〕 구현 결과

a	c	공격 시간(초)	성공 확률 (성공 회수/시도 회수)
1	2	$\ll 1$	99.99%(100/100)
8	3	23.24	99.99%(100/100)
12	7	47.72	99.99%(100/100)

다. 계산 복잡도는 [표 6]의 단계 5의 계산 복잡도와 같다. 그러므로 $a=12$인 경우 $2^{25.02}$개의 IV를 이용하여 2^{56}의 계산 복잡도로 비밀키를 복구한다. 2^{2a}개의 IV 중에서 유효한 IV는 1개만 존재한다. 하지만 유효하지 않은 IV가 단계 2를 통과할 확률이 평균 $2^{-c\times(18-a)}$이므로 유효하지 않은 IV들이 단계 2에서 모두 걸러질 확률은 $(1-2^{-c(18-a)})^{2^{2a}-1}$이 된다. c의 값을 조절하면 이 확률을 1에 근사 할 수 있다. $a=12$, $c=5$이면 성공 확률은 98.45%이다. 하지만 c의 값을 증가시켜도 공격에 필요한 IV의 수는 거의 변화가 없기 때문에 $c=7$로 설정한 경우, 성공확률은 99.99%이다. a와 c에 따른 공격 복잡도는 [표 7]과 같다. [표 8]은 공격 알고리즘의 단계 4까지를 구현한 결과이다. 복구한 부분키와 비밀키에 대한 선형식이 정확한 결과인지 실험하였다. 본 구현은 Pentium-4, 2.4G CPU, Windows XP Pro Sp 2에서 실행되었고, $a=12$, $c=7$ 인 경우 평균 50초 이내에 선형식을 구성하고 비밀키를 복구하였다.

Ⅳ. 결 론

본 논문에서는 eSTREAM에 제안된 스트림 암호 Grain-v1에 대한 연관키 선택 IV 공격을 제안하였다. 본 공격은 Grain-v1 최초의 키 복구 공격으로 다음과 같은 취약점을 이용하여 2개의 연관키와 $2^{25.02}$개의 IV를 이용하여 2^{56}의 시간 복잡도로 비밀키를 복구할 수 있다.

① a-비트 좌측 순환 이동된 연관키 K와 $\widetilde{K}$에 대해, $S^a=\widetilde{S}^0$를 만족하는 IV와 $\widetilde{IV}$가 존재한다.
② $S^a=\widetilde{S}^0$를 만족하면 *D-Test*를 통과한다.
③ 초기화 과정에서 IV의 일정 비트를 고정하면 h의 입력값이 고정되어 취약한 형태의 부울 함수가 된다.

첫 번째 취약점은 Grain-v1의 설계의 근본적인 문제점으로 쉽게 고칠 수 없어 보인다. 두 번째 취약점의 경우, 키스트림 생성 과정에서 사용되는 비트의 위치를 수정하는 방법으로 해결이 가능하다. 마지막으로 세 번째 취약점은 h 자체를 다른 함수로 변경하거나 입력되는 위치를 NFSR에 더 많은 비중을 주는 방법으로 해결할 수 있다. 위 취약점들은 Grain-128에도 유사하게 적용된다. 그러므로 Grain-128에 대한 키 복구 공격 역시 가능할 것으로 예상된다.

초기화 과정과 키스트림 생성 과정이 유사한 구조를 가지는 스트림 암호를 설계하는 경우, 이러한 공격 방법에 대한 안전성을 고려해야 할 것이다.

참고문헌

[1] C. Berbain, H. Gilbert and A. Maximov, "Cryptanalysis of Grain", FSE 2006, LNCS 4047, pp.15-29, SpringerVerlag, 2006.

[2] M. Hell, T. Johansson and W. Meier, "Grain - A Stream Cipher for Constra ined Environments," ECRYPT Stream Cipher Project Report 2005/010, 2005. Available at http://www.ecrypt.eu.org/stream/ciphers/grain/grain.pdf

[3] M. Hell, T. Johansson and W. Meier, "Grain - A Stream Cipher for Constrai ned Environments", 2007. Available at http://www.ecrypt.eu.org/stream/p3ciphers/grain/Grain_p3.pdf

[4] S. Khazaei, M. Hassanzadeh and M. Kiaei, "Distinguishing Attack on Grain", ECRYPT Stream Cipher Project Report 2005/71, 2005.

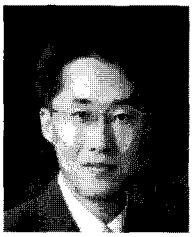
Available at
http://www.ecrypt.eu.org/stream/papersdir/071.pdf

[5] Ö. Kücük, "Slide Resynchronization Attack on the Initialization of Grain 1.0", ECRYPT Stream Cipher Project Report 2006/44, 2006. Available at
http://www.ecrypt.eu.org/stream/papersdir/2006/ 044.ps

〈著者紹介〉

이 유 섭 (Yuseop Lee) 학생회원
2007년 2월 : 서울시립대학교 수학과 학사
2007년 3월~현재 : 고려대학교 정보경영공학전문대학원 석박사 통합과정
<관심분야> 스트림 암호 및 해쉬 함수의 분석 및 설계

정 기 태 (Kitae Jeong) 학생회원
2004년 2월 : 고려대학교 수학과 학사
2006년 2월 : 고려대학교 정보보호대학원 석사
2006년 3월~현재 : 고려대학교 정보경영공학전문대학원 박사과정
<관심분야> 블록 암호, 스트림 암호 및 해쉬 함수의 분석 및 설계



성 재 철 (Jaechul Sung) 종신회원
1997년 8월 : 고려대학교 수학과 학사
1999년 8월 : 고려대학교 수학과 석사
2002년 8월 : 고려대학교 수학과 박사
2002년 8월~2004년 1월 : 한국정보보호진흥원 선임연구원
2004년 2월~현재 : 서울시립대학교 수학과 조교수
<관심분야> 암호 알고리즘 설계 및 분석

홍 석 희 (Seokhie Hong) 종신회원
1995년 2월 : 고려대학교 수학과 학사
1997년 2월 : 고려대학교 수학과 석사
2001년 2월 : 고려대학교 수학과 박사
1999년 8월~2004년 2월 : (주) 시큐리티 테크놀로지스 선임연구원
2004년 4월~2005년 2월 : K.U.Leuven 박사후연구원
2005년 3월~현재 : 고려대학교 정보경영공학전문대학원 조교수
<관심분야> 암호 알고리즘 설계 및 분석, 컴퓨터 포렌식