제33회 한국정보처리학회 춘계학술발표대회 논문집 제17권 제1호 (2010. 4)

산술 쉬프트 레지스트(ASR)를 이용한 고속 스트림 암호 구현

김길호*, 김종남*, 조경연*
*부경대학교 컴퓨터공학과
e-mail:vnlqpcdd@hanmail.net

Implementation of fast stream cipher using arithmetic shifter register

Gil-Do Kim*, Jong-Nam Kim*, Gyeong-Yeon Cho*
*Dept of Computer Engineering, Pu-Kyong University

요 약

본 논문은 산술 쉬프트 레지스트(ASR)를 이용한 32비트 출력의 고속 스트림 암호를 제안한다. 제안한 스트림 암호는 소프트웨어 및 하드웨어 구현이 쉽게 디자인되었으며, 무선 통신 장비와 같이 제한적인 환경에서 빠르게 수행할 수 있도록 개발되었다. 제안한 스트림 암호는 SSC2, Salsa20과 수행속도 비교에서 좋은 결과를 보여주고 있으며, 좋은 통계적 분산 특성과 안전성 또한 현대 스트림 암호 알고리즘이 필요로 하는 안전성을 만족하고 있다.

1. 서론

본 논문에서는 휴대폰과 같은 제한된 자원을 가지는 무선 통신장비에서 데이터 암호의 적용을 위한 스트림 암호를 소프트웨어로 빠르게 수행가능하고 안전성이 검증된 ASR 스트림 암호를 제안한다.

스트림 암호 ASR은 산술 쉬프트 레지스트[1]와 블록 암호 알고리즘인 AES[2]를 조합하여 32비트 키 스트림을 출력한다. 먼저 초기화 과정으로 AES를 비밀키와 IV(Initial Vector)를 이용하여 10라운드 암호화 과정을 수행한 후 AES의 8, 9라운드 값으로 산술 쉬프트 레지스트를 초기화 시키고 AES 10라운드를 계속해서 산술 쉬프트 레지스트를 이용하여 업 데이터 한 후 축소 함수를 적용하여 32비트 키 스트림을 계속해서 생성한다. AES의 10라운드 128비트는 32비트 단위로 확산을 수행한다. 확산 과정은 간단한 논리연산으로 구성된 비선형 함수를 적용하여 소프트웨어 및 하드웨어 구현 시 빠르게 수행할 수 있도록 디자인 하였다. 제안한 ASR 스트림 암호는 SSC2[3], Salsa20[4]과 수행속도 비교에서 좋은 결과를 보여주고 있으며, 안전성 또한 현대 암호 알고리즘이 필요로 하는 안전성을 만족하고 있고, 좋은 통계 분석적 특성을 가지고 있다.

2. 에이전트 개발도구의 요구사항

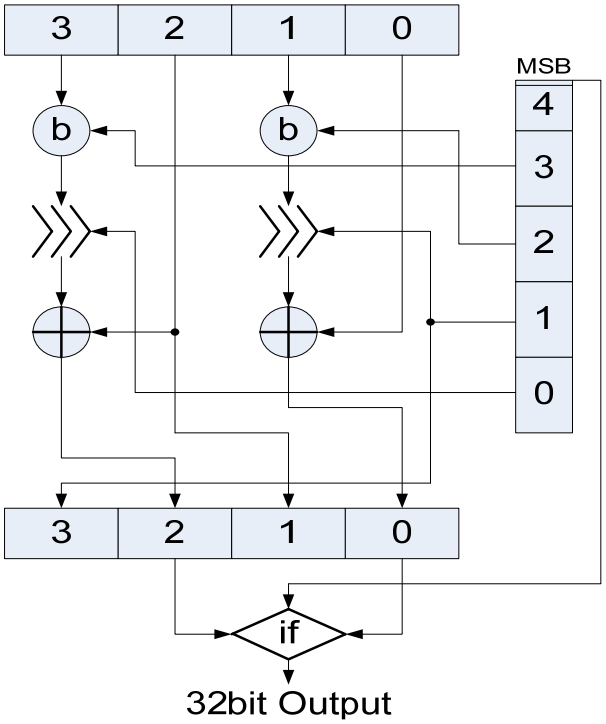
의사난수발생기로 사용할 수 있는 산술 쉬프트 레지스터는 $GF(2^n)$상에서 0이 아닌 초기 값에 0 또는 1이 아닌 임의의 수 D를 곱하는 수열로 정의한다. 그리고 ASR의 선형 복잡도(Linear Complexity)는 기존의 LFSR(Linear Feedback Shift Register)의 선형 복잡도 보다 높아서 안전도가 높다. 본 논문에서는 $GF(2^{151})$상에서 특성다항식은 '0x00800000 0x00000001 0x00000001 0x00000004 0x00000025', D = 2^{23}을 적용한다. ASR의 동작은 다음과 같다.

```
w0 = ASR[4]
ASR[4] = ASR[3]》9
ASR[3] = ((ASR[3]《23) | (ASR[2]》9)) ⊕ w0
ASR[2] = ((ASR[2]《23) | (ASR[2]》9)) ⊕ w0
ASR[1] = ((ASR[1]《23) | (ASR[0]》9)) ⊕ (w0《2)
ASR[0] = (ASR[0]《23)⊕ (w0《5)⊕ (w0《2) ⊕ w0
```

128비트의 확산과정은 (그림 1)과 같으며, (그림 1)에서 오른쪽 0~4까지 앞서 설명한 산술 쉬프트 레지스트를 저장하는 공간이다. 각각의 저장 공간은 32비트이며, 최상위 워드를 저장하는 4번은 23비트만 유효한 값이 된다.

그리고 (그림 1)에서 b함수는 워드단위 입력을 2개 받아 바이트 단위로 나누어 AND, OR 논리연산을 교번 수행한 후 다시 결합한 결과를 워드로 만든다. 워드에서 바이트로 분할을 쉽게 구현하기 위해 union 구조체로 32비트 변수를 선언하였다. b함수의 수행과정은 다음과 같다.

```
AES[3].byte[0] = AES[3].byte[0] | ASR[3].byte[0]
AES[3].byte[1] = AES[3].byte[1] & ASR[3].byte[1]
AES[3].byte[2] = AES[3].byte[2] | ASR[3].byte[2]
AES[3].byte[3] = AES[3].byte[3] & ASR[3].byte[3]
```



[그림 1] ASR 32비트 스트림 암호 출력 흐름도

제안한 일고리즘의 소프트웨어 구현은 Visual Studio 2005 C 컴파일러를 사용하였고 실행환경은 Windows Vista, Intel Core(TM)2 Duo CPU 2.26Ghz, 2.27Ghz, 2GB RAM의 환경에서 알고리즘의 수행 시간을 테스트했다. 결과는 <표 1>과 같다. <표 1>에서 1.5GB는 각각의 알고리즘 수행 후 출력 결과의 양을 의미한다. 제안한 알고리즘은 SSC2보다 수행 속도는 늦지만 Salsa20보다는 빠르다. SSC2와 Salsa20은 개발자들이 제공한 소스를 가지고 소프트웨어로 구현한 후 수행 시간을 테스트 했다.

<표 1> 수행시간 테스트 결과

시간 / 알고리즘	수행 결과
ASR	1.5GB / 33초
SSC2	1.5GB / 30초
Salsa20	1.5GB / 48초

제안한 스트림 암호의 안전성은 먼저 AES 10라운드 후의 각각의 워드는 안전성이 검증된 랜덤한 값이므로 분석 확률은 2^{-32}이 된다. 그리고 b함수의 분석 확률은 바이트 단위로 AND, OR 연산을 수행하며 이와 같은 AND, OR 연산[5]의 확률은 각각 $2^{-3.25}$이고, 워드 단위의 분석 확률은 2^{-13}이 된다. 그리고 왼쪽 회전 연산의 분석[6] 확률은 32비트 단위의 회전 연산이므로 LSB 5비트만 유효한 값이 된다. 그래서 왼쪽 회전연산의 최종적인 분석 확률은 2^{-5}이 된다. b함수와 왼쪽 회전연산은 각각 독립적으로 수행하므로 두 연산 결과의 분석 확률은 $2^{18} = 2^{-13} * 2^{-5}$이다. 그리고 이 결과에 AES의 32비트 워드가 더해지므로 AES의 각각의 워드의 분석 확률은 $2^{-50} = 2^{-18} * 2^{-32}$이다. 마지막으로 2^{-50}의 AES 워드 중에서 0 또는 2번째 워드를 ASR의 MSB로 선택하므로 2^{-1}의 확률이 적용된다. 그래서 산술 쉬프트 레지스트를 이용한 스트림 암호의 최종적인 분석 확률은 2^{-51}이 된다.

3. 결론

본 논문에서는 블록 암호 AES와 산술 쉬프트 레지스트를 조합한 32비트 키 스트림 출력의 새로운 스트림 암호를 제안한다. 산술 쉬프트 레지스트 스트림 암호는 소프트웨어 구현이 쉽게 디자인된 암호 알고리즘이다. 특히 계산능력이 제한된 무선 통신장비에서 빠르게 수행할 수 있도록 개발되었다. 제안한 스트림 암호는 128비트 마스터 키(KEY)와 128비트 초기벡터(IV)를 가지고 AES 10라운드를 수행한 후 워드 단위로 확산연산을 수행하면서 최종적으로 32비트 출력을 생성한다. 제안한 알고리즘은 매우 간결한 구조를 가지고 있으며 산술 쉬프트 레지스트와 확산연산 과정에서 간단한 논리연산으로 구성된 비선형 변환함수를 적용한 결합 스트림 암호 알고리즘이다.

제안한 스트림 암호는 SSC2보다 수행 속도는 느리지만 Salsa20보다는 빠른 결과를 보여주고 있으며, 안전성 또한 현대 암호 알고리즘이 필요로 하는 안전성을 만족하고 있고, 좋은 통계적 분산을 보여주고 있다. 그래서 제안한 스트림 암호는 휴대폰과 같은 제한된 리소스를 가지고 있는 무선 통신 장비에서 사용할 암호 알고리즘으로 적당하다.

감사의 글

본 연구는 동남광역경제권선도산업지원단의 광역경제권선도산업육성사업, 중소기업청의 산학연공동기술개발지원사업(선도형)으로 수행된 연구 결과임.

참고문헌

[1] 박창수, 조경연 “갈로이 선형 궤환 레지스터의 일반화” 전자공학회논문지 제43권 C1편 제1호 2006. 1.

[2] Daemen, J., and Rijmen, V. “AES Proposal: Rijndael, Version2.,” Submission to NIST, March 1999.

[3] C. Carroll, A. Chan, and M. Zhang "The software-oriented stream cipher SSC-Ⅱ" FSE 2000 LNCS Vol.1978 p.p 39-56 2000.

[4] D. J. Bernstein, Synchronous Stream Cipher Salsa20, http://www.ecrypt.eu.org/stream/salsa20.html.

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

[6] S. Contini, R.L. Rivest, M.J.B. Robshaw, and Y.L. Yin, “The Security of the RC6 Block Cipher”, 1998.