경량 암호 알고리즘 HIGHT와 PRESENT의 저전력 매체(Arduino)에서의 성능 비교

김나영*, 신동*, 김병만*
*금오공과대학교 컴퓨터소프트웨어공학과
e-mail: amorea12@naver.com, dong_s@me.com, bmkim@kumoh.ac.kr

Benchmark of Lightweoght Block Ciphers (HIGHT & PRESENT) for Arduino

NaYoung Kim*, Dong Shin*, ByeongMan Kim*
*Dept of Computer Software Engineering,
Kumoh National Institute of Technology

요 약

IoT 환경의 센서 네트워크와 RFID 태그 등에서의 AES나 SEED에 대응 할 수 있는 새로운 저전력 경량화 암호 알고리즘이 필요 해짐에 따라 본 논문에서는 2006년 국내에서 제안된 HIGHT와 2007년 CHES에서 제안된 PRESENT 알고리즘을 Arduino에 적용하여 성능을 비교분석 하였다. 그 결과 HIGHT 알고리즘이 PRESENT알고리즘에 비하여 더 짧고, 적은 수행시간과 프로그램 메모리 사용량을 보였으며, 더 많은 동적 메모리 사용량을 보였다.

1. 서론

기존의 유선네트워크기반 컴퓨팅 환경은 서버-클라이언트 위주로서 폐쇄된 공간에서 합법적인 권리를 가진 사람만이 접근 가능하며, 전용망을 이용하여 데이터를 주고받아 정보의 누출이 어려웠다. 또한 해당 환경에서의 보안 수단만 구현하면 됐었다. 그러나 요즘의 현대사회는 서버-클라이언트 위주의 유선네트워크기반 컴퓨팅 환경이 아닌 노트북, 스마트 폰과 같은 다양한 유.무선 하드웨어들의 이용환경으로 바뀌었다. 심지어 최근 들어서는 사물들이 통신기능을 가지고 사람 또는 사물간에 상호 작용하는 환경이 되었다.

이러한 IoT 환경의 도입으로 사용자들의 편리함이 증가하였지만, 예전의 폐쇄적인 환경이 아닌, 사물로의 쉬운접근 때문에 개인의 사생활 정보 또한 쉽게 탈취 및 복제가 가능해져 사용자들의 보안위협 또한 증가하게 되었고, 이러한 상황에 맞춘 보안 도입의 필요가 대두되었다. 그러나 예전과 달리 현재 사용되고 있는 다양한 하드웨어에 대하여 보안이 이루어지기 위해서는, 적은 소비전력 뿐만이 아니라, 한정된 자원, 적은 면적 등의 요구조건을 충족시키면서도 적절한 속도를 낼 수 있어야한다. 그러한 요구조건을 충족하기 위해서 저전력 경량화 알고리즘의 도입이 필요하게 되었고 다양한 저전력 경량화 알고리즘들이 제안됐다.

위와 같이 저전력 경량화 알고리즘들이 다양하게 등장하면서, 저전력 경량화 알고리즘에 대하여 다양한 선택이 가능하게 되어졌다. 그러나 이러한 선택을 하게 될 때 각각R의 저전력 경량화 알고리즘들에 대하여 어느 부분에 중점을 두어야할지 고민의 폭 또한 넓어지게 됐다. 이에, 본 논문에서는 저전력 경량화 알고리즘들 중에서 두 가지의 알고리즘(HIGHT, PRESENT)을 선택하여 Arduiono에서의 성능을 살펴보았다.

2. 관련 연구

한국인터넷 진흥원 (KISA)에서 진행한 연구에서는, 모듈을 구현할 때 0.35㎛ 공정의 라이브러리와 100㎒ 클록 속도를 기준으로 Synopsys Design Complier를 통해 합성하여 면적을 측정하였으며, 평균 소비전력은 Synopsys Design Complier를 통하여 gate-level netlist로 변환된 데이터를 가지고 power analysis & optimization tool인 Power Compiler를 이용하여 측정하였다. 그 결과로 아래 <표 1>과 같은 성능 분석 결과가 나타났다.

<표 1> 각 모듈별 성능 분석 결과

	Block size	Key size	Area (#GE)	Speed (kbps@100kHz)	Mean Power (μW)
CLEFIA	128	128	12170.7	355.56	95.57
DESXL	64	192	2757.0	43.84	16.88
mCRYPTON	64	128	4593.7	376.47	48.79
KASUMI	64	128	5823.8	640.00	44.19
PRESENT	64	80	3093.1	193.94	28.82
PRESENT	64	80	2285.4	11.41	14.91
XTEA	64	128	3366.9	96.97	25.45
SEA	96	96	3488.2	51.34	34.88
HIGHT	64	128	3138.6	188.23	13.84
AES	128	128	2874.0	11.90	3.00

위 연구에서는 면적측면에서 AES, HIGHT, PRESENT, DESXL, SEA, XTEA가 다른 모듈에 비해 적은 값을 나타냈다. S-BOX와 같은 substitution 연산을 활용하면 암호화강도는 향상될 수 있지만, 일반적인 연산에 비하여 상대적으로 많은 면적을 차지하게 되어, 이들 모듈의 경우에는 한 라운드에 여러 개의 S-BOX를 동시에 사용하지 않고, 하나의 S-BOX를 여러 라운드에 걸쳐서 반복 사용하여 경량화 하였다.

연산 속도에서는 KASUMI, mCRYPTON, CLEFIA가 좋은 성능을 보였다. KASUMI와 mCRYPTON은 다른 모듈과 비교하여 매우 적은 라운드로 암호화 및 복호화를 수행하기 때문에 강점을 보였다. CLEFIA는 암호화를 수행하는데 걸리는 라운드의 수가 mCRYPTON에 비해 약 2배 많으나 블록크기가 2배 크기 때문에 두 암호모듈은 비슷한 속도를 보였다.

반면, AES, DESXL, SEA, XTEA는 속도에서 많이 뒤처지는 모습을 보여주었다. 이는 S-BOX 등의 연산을 여러 라운드로 나눔으로써 면적에서 이득을 보는 반면, 연산에 필요한 라운드 수는 증가하기 때문이다. 평균소비전력에서는 HIGHT, AES, 면적을 줄인 PRESENT, DESXL이 좋은 성능을 보여주었다. 이들 모듈은 총 라운드의 수는 늘어났지만 라운드마다 수행하는 연산이 단순하고 적은 연산 횟수를 가지기 때문이다.

평균 소비전력의 경우에는 기존 연구에서는, 높은 암호화 강도를 가지는 CLEFIA가 다른 경량화 알고리즘에 비해 상대적으로 복잡한 연산과 많은 연산 횟수를 필요로 하기 때문에, 가장 넓은 면적과 가장 많은 소비전력을 보였으며 KASUMI와 mCRYPTON도 같은 이유로 비교적 평균소비전력이 컸다.

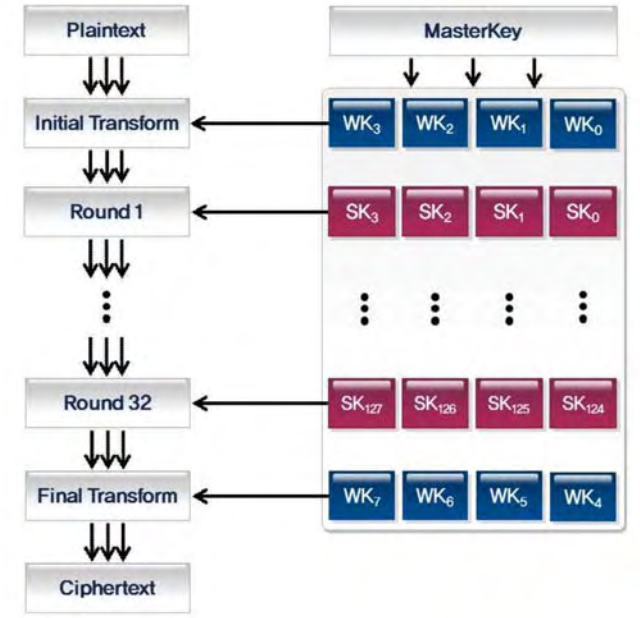
3. 경량 암호 알고리즘

본 논문에서는 비슷한 시기에 제안 되어 다른 구조를 가진 저전력 경량화 알고리즘을 비교하기로 결정하여, 국산 알고리즘 HIGHT와 PRESENT 알고리즘을 선택, 비교하였다.

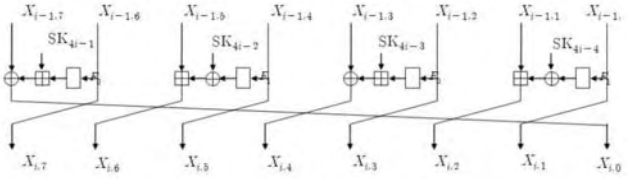
3.1 HIGHT

HIGHT(HIGh sercurity and light weigHT)알고리즘은 RFID, USN등과 같이 저전력, 경량화를 요구하는 환경에서 기밀성을 제공하기 위해 개발된 64비트 블록암호 알고리즘이다. HIGHT는 128비트의 마스터키, 64비트의 평문으로부터 64비트의 암호문을 출력한다. 제한적 자원을 갖는 환경에서 구현될 수 있도록 8비트 단위의 기본적인 산술 연산만으로 구성되어 있으며 SEED, AES등 기타 알고리즘보다 간단한 알고리즘 구조로 설계되었다.

HIGHT의 전체구조는 일반화된 Feistel 변형 구조로 이루어져 있으며 64비트의 평문과 128비트의 마스터키로부터 생성된 8개의 8비트 화이트닝키와 128개의 8비트 서브키를 입력으로 사용하여 총 32라운드를 거쳐 64비트 암호문을 출력한다.



(그림 1-1) HIGHT 전체 구조



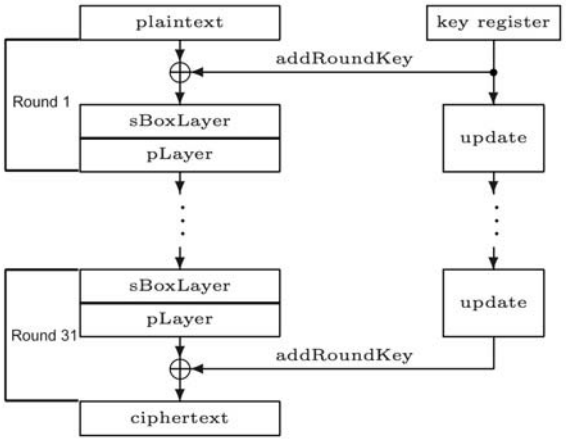
(그림 1-2) HIGHT 각 라운드의 구조

각 라운드는 64비트 Feistel 변형구조로 이루어져 있다. 각각의 연산법은 2^8덧셈과 xor, F함수로 이루어진 간단한 형태이다. F함수도 기존 암호 알고리즘에서 사용되는 복잡한 형태가 아닌 Shift와 xor로 이루어진 간단 한 연산이다.

3.2 PRESENT

PRESENT 알고리즘 또한 저전력, 경량화를 요구하는 환경에서 기밀성을 제공하기 위해 개발된 64비트 블록암호 알고리즘이다. PRESENT는 80/128의 비밀키를 사용하며, SPN구조를 가지는 대칭키 암호로서 총 31라운드를 가진다. 본 논문에서는 PRESENT-80을 선택하였다.

(그림 2)은 PRESENT의 전체구조이며, 4비트의 S-BOX와 XOR 및 비트 시프트로 연산되고 31 라운드로 구성되어 있는 것을 나타낸다. 암호화 강도는 AES에 비하여 떨어지지만 Smart card와 USN 과 같은 시스템에 사용하기 위해 면적과 소비 전력을 개선시킨 암호 알고리즘이다.



(그림 2) PRESENT 전체 구조

4. 경량 암호 Arduino에서의 모듈 성능 분석

HIGHT와 PRESENT의 비교를 위하여 운영모드는 CBC 모드로 통일하여 암,복호화 과정시의 성능을 비교하였다.

4.1 각 알고리즘의 Arduino상에서 용량

HIGHT 알고리즘은 Arduino-nano상에서 실행되었을 때 최대 32,256 바이트의 프로그램 저장 공간 중에서 2,908 바이트의 사용하여 총 9%의 메모리를, 전역 변수를 저장하는 동적 메모리는 최대 2,048바이트 중에서 352바이트를 사용하여 총 17%의 동적 메모리 사용률을 보였다.

PRESENT 알고리즘의 경우에는 최대 32,256바이트의 프로그램 저장 공간 중에서 3,878바이트를 차지하여 12%의 사용률을 보였으며, 변수를 저장하는 동적메모리의 경우에는 최대 2,048바이트 중에서 298바이트를 사용 하여 14%의 동적메모리 사용률을 보였다.

4.2 각 알고리즘의 Arduino상에서 실행시간

HIGHT 알고리즘의 경우에는 암호화의 경우에는 총 70번의 실행결과 평균 544.571μs의 실행시간을 보였으며, 복호화의 경우에는 총 70번의 실행결과 평균 569.6μs의 실행시간을 보였다.

PRSENT 알고리즘의 경우에는 암호화의 경우에는 총 70번의 실행결과 평균 12747.54μs의 실행 시간을 보였으며 복호화의 경우에는 70번의 실행결과 평균 13488.4μs의 실행시간을 보였다.

5. 결론

본 논문은 Arduino-nano에서 PRESENT와 HIGHT 알고리즘의 성능에 대하여 연구하였다. Arduino상에서의 성능 분석을 통하여 확인을 하여 본 결과, PRESENT와 HIGHT의 알고리즘 중 저장공간의 경우에는 HIGHT가 더욱 경량화 되어 있었으며, 동적 메모리의 경우에는 PRESENT가 더 경량화 되어 있는 모습을 보였다. 또한 HIGHT알고리즘이 암,복호화 시에 PRESENT 알고리즘에 비하여 더 빠른 실행 속도를 보였다<표 2>.

본 논문에서는 전력 소모량은 장비의 부족으로 측정하지 못하였다. 차후 전력측정 장비가 도입되면 HIGHT와 PRESENT의 Arduino-nano에서 전력측정을 할 것이며, 또한 이번 논문에서 실행한 HIGHT와 PRESENT 뿐만 아니라, CLEFIA, DEXSL을 포함한 다양한 저전력 경량화 알고리즘을 Arduino에서 비교 할 예정이다.

참고문헌

[1] Korea Internet & Security Agency "HIGHT 블록암호 알고리즘 사양 및 세부 명세서"

[2] Korea Internet & Security Agency "Hardware Implementation of Lightweight CryptoAlgorithms (HIGHT)"

[3] Center for Information Security Technologies, Korea University "Research on efficient HW/SW co-design method of light-weight cryptography using GEZEL"

[4] A. Bogdanov, L.R. Knudsen, G. Leander, C. Paar1, A. Poschmann, "PRESENT: An Ultra-Lightweight Block Cipher"

<표 2> HIGHT와 PRESENT 성능 분석 결과

Category	Block size (bits)	Key size (bits)	Rounds	Memory (byte)	Dynamic Memory (byte)	Encryption Operating time (μs)	Decryption Operating time (μs)
HIGHT	64	128	32	2,908	352	544.571	569.6
PRESENT	64	80	31	3,878	298	12747.54	13488.4