정 보 보 호 학 회 지
제25권 제2호, 2015. 4

사물인터넷을 위한 경량 암호 알고리즘 구현

서 화 정*, 김 호 원**

요 약

모든 사물들이 인터넷에 연결되어 정보의 생산과 교환이 실시간으로 이루어지는 사물인터넷 기술은 사용자에게 개인 맞춤형 서비스를 제공함으로써 삶의 질을 향상시키고 있다. 하지만 사용자가 제공한 민감한 개인 정보가 적법한 서비스가 아닌 악의적인 목적으로 사용된다면 사물인터넷 기술의 발전은 인류에게 큰 재앙으로 다가올 것이다. 따라서 이를 방지하기 위한 방안으로 사물들 간에 교환되는 정보에 대하여 암호화 연산을 수행하게 된다. 이는 교환되는 정보에 대한 기밀성을 제공한다는 장점을 가지지만 암호화 연산을 수행하기 위해 추가적인 계산이 필요한 단점을 가진다. 이러한 연산 복잡도는 사물인터넷과 같은 임베디드 환경에서는 큰 부하로 다가온다. 따라서 이러한 문제점을 해결하기 위한 다양한 경량 암호 알고리즘이 암호학자에 의해 정의되고 이는 다시 암호 엔지니어에 의해 효율적인 구현 방안이 제시되고 있다. 본 논문에서는 지금까지 연구되어온 경량 암호 알고리즘의 실용성을 사물인터넷 환경에서 소프트웨어와 하드웨어 관점에서 살펴보도록 한다.

Ⅰ. 서 론

가트너가 선정한 대표적인 IT 기술로써 요즘 가장 큰 각광을 받고 있는 사물인터넷 기술은 기존에는 불가능했던 개인 맞춤형 서비스를 가능하게 하고 있다. 이는 사물들 간에 교환되는 다량의 센싱 데이터를 수집하고 이를 다시 머신러닝 기법을 적용하여 유용한 정보를 이끌어 내는 형식으로 서비스를 제공하고 있다. 하지만 다량의 센싱 데이터는 개인의 민감한 정보를 포함하고 있어 이를 암호화하지 않고 평문 형식으로 교환하게 될 경우 악의적인 공격자에 의해 도청 및 변형이 이루어질 수 있는 문제점을 가진다. 이러한 문제점을 해결하기 위해 사물들이 생성해 내는 모든 데이터는 암호화된 형식으로 다른 사물들과 통신이 이루어져야 한다. 이를 위해 암호 알고리즘을 이용하여 교환되는 패킷을 암호화함으로써 패킷에 대한 기밀성을 확보할 수 있다. 하지만 기존의 퍼스널 컴퓨터와 비교하여 사물인터넷의 임베디드 프로세서는 매우 제한적인 연산 속도와 저장공간을 가진다. 따라서 사물인터넷 환경을 고려한 경량 암호화 기법에 대한 연구가 높은 가용성 확보를 위해 중요하다. 사물인터넷 상에서의 암호화는 소프트웨어와 하드웨어로 구현이 가능하다. 소프트웨어는 대부분의 임베디드 보드에 소스코드를 올림으로써 유지보수가 편한 장점을 가진다. 하지만 속도가 느려서 가용성 확보에 어려움을 겪을 수 있다. 반면에 하드웨어의 경우 빠른 연산 속도를 자랑하지만 한번 칩을 생산하고 난 이후에는 이를 수정하는 것이 불가능한 단점을 가진다. 또한 추가적인 칩을 메인보드에 연결해야 하는 부수적인 작업이 필요하다. 따라서 가용성과 개발환경에 따라 차별화된 접근이 요구되어 진다. 본 논문에서는 사물인터넷 서비스 상에서 운영가능한 다양한 경량 암호화 기법에 대해 살펴보며 이를 실제로 구현 시의 가용성에 대해 살펴 봄으로써 각각의 사물인터넷 서비스에 유용한 기술에 대해 확인해 보도록 한다.

본 논문의 구성은 다음과 같다. 2장에서는 경량암호화 중 SPN과 ARX 구조에 대해 살펴본다. 3장에서는 소프트웨어와 하드웨어 관점에서의 구현 고려사항을 확인해 본다. 4장에서는 소프트웨어와 하드웨어 상에서의 암호화 알고리즘 성능을 평가한다. 마지막으로 5장에서는 본 논문의 결론을 내린다.

본 연구는 미래창조과학부 및 정보통신기술진흥센터의 대학ICT연구센터육성 지원사업의 연구결과로 수행되었음 (IITP-2015-H8501-15-1017)

* 부산대학교 전기컴퓨터공학부 (hwajeong84@gmail.com howonkim@pusan.ac.kr)

Ⅱ. 경량암호화 기법

2.1. SPN 구조

SPN 기반 블록 암호화는 substitute과 permutate 과정을 혼합하여 암호화를 수행하는 것을 의미한다. substitute는 입력으로 받은 값을 1대1로 매칭되는 다른 값으로 출력하는 연산을 의미하며 permutate는 입력으로 받은 값들의 배열을 변경하여 복잡도를 높이는 연산을 의미한다. 해당 연산을 통해 평문은 암호화된 형식으로 출력되게 된다.

2.1.1. AES

AES는 현재 표준으로써 가장 널리 쓰이는 블록암호화 기술로써 AES conference'98에서 제안된 기술이다. 해당 기술은 128-bit의 블록 사이즈를 가지며 128, 192, 256-bit와 같이 다양한 키를 통해 암호화되는 특징을 가진다[14]. 예전의 컴퓨터가 높은 성능을 가지지 않은 이유로 8-bit word를 기준으로 설계되어 현재 사물인터넷의 마이크로 프로세서에 적합한 특징을 가진다. 32-bit의 경우 T-table을 통해 성능을 향상 시킬수 있다. 이는 모든 프로세서에서 높은 성능을 제시할 수 있다는 장점을 가진다. AES는 라운드 함수를 반복해서 수행하는 구조를 가지며 총 4개의 단계를 반복하게 된다. 첫 번째는 subbyte 연산으로써 8-bit의 입력을 받으면 이를 다시 8-bit의 출력으로 나오게 하는 일종의 substitute 역할을 하게 된다. 그다음에는 shiftrows를 통해 입력값의 배열을 섞어 주게 되며 mixcolumns를 통해 특정한 값을 곱한 값을 출력하게 된다. 마지막으로 round 킷값을 현재 평문에 더해주는 형식으로 연산을 수행하게 된다.

2.1.2. PRESENT

PRESENT는 CHES'07에서 제안된 알고리즘으로써 64-bit의 블록사이즈를 가지며 80-bit와 128-bit의 키크기를 가진다[13]. 해당 논문에서는 AES에 비해 2.5배 작은 하드웨어 설계가 가능하다고 주장하고 있을 정도로 경량의 구현이 가능한 장점을 가진다. 31 Round로 구성되며 저전력 높은 하드웨어 효율성을 목표로 만들어졌다. 현재 ISO, IEC표준으로 정의되어 있다.

2.1.3. CLEFIA

CLEFIA는 SONY 사에 의해 FSE'07에서 제안된 알고리즘이다[15]. 128-bit의 블록사이즈를 가지며 128, 192, 256-bit의 키 크기를 가진다. Feistel 구조를 가지며 18, 22, 26round를 가진다. 해당 알고리즘은 DRM system 적용을 위해 개발되었다. 일본 정부의 추천하는 cryptographic techniques이며 CRYPTREC2013에 명시되었다. s-box를 통해 substitute 연산을 제공하며 linear combination을 통해 diffusion 특성을 제공한다. Bit-wise exclusive-or연산과 GF(2^8) 상에서의 곱셈연산으로 구성된다.

2.1.4. PRINCE

PRINCE는 ASIACRYPT'12에서 제안된 알고리즘으로써 64-bit 블록 사이즈를 가지며 128-bit의 키가 제공된다[16]. SPN구조로 12 round를 수행하며 AES와 같이 4-by-4 행렬로 내부 구조를 정의하고 있다. 핵심연산으로는 ShiftRows, MixColumns과 4-bit S-box로 구성된다. 하드웨어 구현 시 AES-128에 비해 1/15 크기를 가지는 특징을 가진다.

2.1.5. SEA

SEA는 SCRAA'06에서 제안된 알고리즘으로써 96-bit 블록 사이즈를 가지며 96-bit의 키를 사용한다[17]. bitwise xor, 3x3 S-box, rotation, addition으로 구성된다. 총 93 round의 연산을 수행하며 feistel 구조를 가진다.

2.1.6. KLEIN

KLEIN은 SaP'12에서 제안된 알고리즘으로써 64-bit 블록 사이즈를 가지며 64, 80, 96-bit의 키를 사용한다[18]. 12, 16, 20 round 연산이 feistel 구조를 통해 수행된다. 4x4 S-box를 통해 substitute 연산이 수행되며 16개의 4-bit 값들이 묶여서 rotation되는 구조를 가진다. 전체 8byte를 두 개의 4byte로 나누어 GF((2^8)^4)곱셈연산이 수행된다. 해당 연산은 AES의 MixColumn 연산과 유사한 구조를 가진다.

2.1.7. mCrypton

mCrypton은 ISA'06에서 제안된 알고리즘으로써 64-bit 블록 사이즈를 가지며 64, 96, 128-bit의 키를 사용한다[19]. CRYPTON 암호화 기법을 경량화한 기법이다. AES와 같이 4x4 matrix 형식으로 데이터가 표기되며 연산을 수행한다. 한 번의 round는 s-box, bit permutation, xor연산으로 구성된다. 총 4개의 s-box를 가지며 이는 GF(2^4)의 inverse 연산으로 구성된다. 암복호화가 키 스캐쥴링을 제외하고는 동일한 구조를 가진다.

2.1.8. LED

LED는 CHES'11에서 발표된 암호화 기법으로써 64비트 블록 단위 암호화가 가능하며 64/80/96/128비트의 키를 사용한다 [12]. 64비트의 경우 8회 그 외의 경우에는 12회의 round 연산을 반복 수행하게 된다. 개발 시 최소 면적과 이를 위한 serialized 구조를 염두에 두고 개발되었다. AES 암호의 구조를 사용하지만 키 스캐쥴을 하지 않는 것이 특징이다. 4-bit를 이용한 4x4 배열을 가지며 4-bit 단위로 연산이 수행된다. addconstants, subcell, shiftrows, mixcolumnserail으로 구성된다.

2.1.9. DESLX

DESLX는 FSE'07에서 제안된 경량암호와 기법이다. 기존 DES 알고리즘의 짧은 키 길이를 개선함과 동시에 DES를 최소화하여 구현하였다[26]. 64비트 블록 암호화와 184비트의 키를 사용한다. 총 16 라운드를 반복하며 8개의 S-box를 하나의 s-box를 축소한 장점을 가진다.

2.1.10. HummingBird-2

HummingBird-2의 경우 revere security 사에서 2011년에 개발한 암호화 알고리즘으로써 16비트 블록 단위 암호화와 128비트의 키를 사용한다 [20]. Enigma rotor machine을 기반으로 하며 4-bit S-box 및 16-bit 덧셈, XOR등으로 구성된다. 16비트 블록을 사용함에도 128비트의 내부 상태를 사용한다.

2.2. ARX 구조

ARX 기반 암호화는 기존의 SPN 구조와는 달리 간단히 사칙연산을 가지고 블록 알고리즘을 구성하는 기법이다. 이는 보편적으로 덧셈, 회전 그리고 XOR연산의 조합을 가진다. 해당 기법의 주된 관심사는 워드 크기가 되며 이에 따라서 해당 사칙연산에 대한 구현 접근 방안이 결정되게 된다.

2.2.1. HIGHT

HIGHT는 CHES'06에서 제안된 알고리즘으로써 64-bit 블록 사이즈를 가지며 128-bit의 키를 사용한다 [21]. Feistel 구조를 가지며 whitening 연산이 수행된다. GF(2^8) 상에서의 덧셈과 뺄셈 그리고 xor, rotation 연산으로 구성된다. F0와 F1는 xor과 3개의 다른 rotation으로 구성된다. whitening 키는 마스터 키에서 8 byte를 취한다.

2.2.2. KATAN

KATAN은 CHES'09에서 제안된 알고리즘으로써 32, 48, 64-bit 블록 사이즈를 가지며 80-bit의 키를 사용한다[22]. 254 번의 round함수가 수행된다. non-linear boolean, shift, xor연산으로 구성되는 특성을 가진다. 매 round마다 fa, fb연산이 수행되며 결과값은 shift되어 저장되게 된다. irregular 값은 round마다 정의된 값을 사용하게 된다.

2.2.3. IDEA

IDEA는 1991년에 제안된 알고리즘으로써 64-bit의 블록 크기를 가지며 128-bit의 키크기를 가지게 된다 [23]. Lai-Massey scheme 구조를 가지며 8.5 round를 가지는 특성을 가진다. IDEA에 대한 특허는 2012년에 소멸되었으며 Pretty Good Privacy(PGP) v2.0에 사용되며 OpenPGP 표준에 선택적으로 사용가능하다. Bitwise exclusive-or연산과 2^16 상에서의 덧셈과 2^16+1 상에서의 곱셈이 수행된다. 8 번의 round 연산 수행 후 0.5 round 연산을 통해 암호문을 생성하게 된다.

2.2.4. TEA/XTEA

TEA/XTEA는 cambridge computer laboratory에 의해 1994년에 제안되었다 [25]. 64-bit 블록 크기를 가지며 128-bit 키를 사용한다. Feistel 구조를 가지며 64 round를 추천한다. Addition, Bit-wise exclusive-or, rotation연산을 수행한다. TEA는 hash function으로 사용 시 보안 취약점이 발견되었으며 이는 Microsoft의 Xbox game console에서 취약성을 보였다. TEA의 안전성이 보강된 모델인 XTEA가 제안되었다.

2.2.5. SIMON/SPECK

SIMON/SPECK은 NSA에 의해 2013년에 제안된 ARX 기반 암호화 기법이다[1,2]. Feistel 구조를 따른다. SIMON의 경우 rotation과 and 그리고 xor 연산으로 구성된다. SPECK의 경우 rotation과 addition 그리고 xor연산으로 구성된다. 기존의 암호화 기법들과는 달리 매우 다양한 블록 크기와 키 크기를 제공한다. 블록 크기의 경우 32, 48, 64, 96, 128-bit가 가능하며 키 크기의 경우 64, 72, 96, 128, 144, 192, 256-bit가 제공 가능한 특징을 가진다.

2.2.6. LEA

LEA는 NSRI에 의해 2013년에 제안된 알고리즘으로써 ARX 기반의 대칭키 암호화이다[11]. 32-bit 컴퓨터를 타겟으로 설계되어 기본적인 word크기는 32를 가진다. 암호화의 경우 3번의 덧셈과 6번의 xor 그리고 3번의 rotation이 반복되는 간단한 구조를 가진다. 128, 192, 256-bit 암호화 연산을 위해서는 동일한 round 함수를 24, 28, 32회 반복하는 형식으로 수행이 가능하다. 따라서 하나의 구조를 통해 다양한 암호화 강도에 대한 구현이 가능한 장점을 가진다.

2.2.7. Salsa

Salsa는 Daniel J. Bernstein에 의해 제안된 알고리즘으로써 eStream project에서 선택되었다[24]. 키 크기는 256-bit이며 블록 크기는 512-bit이다. ARX 연산으로 구성되며 steam cipher를 위한 암호화 기법이다.

Ⅲ. 구현 관점

3.1. 소프트웨어 관점

코드의 크기를 최소화하기 위해서는 for 문으로 묶어서 나타낼 수 있는 형태로 암호화 구조가 설계되면 for 문의 반복 횟수만큼 코드의 크기를 줄일 수 있다. 또한 해당 for문 안에도 공유되는 연산이 존재하는 경우 이를 묶어서 연산 효율화를 이끌 수 있는 장점을 가진다. 또한 S-box 혹은 T-table과 같은 사전 연산값은 사용하지 않아 이에 대한 저장공간의 소비를 피할 수 있도록 한다. 시간 최소화를 위해서는 목표로 하는 프로세서의 word와 목표로 하는 암호화 알고리즘의 word의 크기가 동일해야 한다. 만약 해당 word 크기가 상이한 경우에는 이를 conversion하는 연산이 추가적으로 필요한 단점을 가진다. 또한 사전값을 미리 계산하여 이에 대한 연산 복잡도를 최소화하는 것이 효과적으로 구현이 가능하게 하는 기법이다.

3.2. 하드웨어

면적을 최소화하기 위해서는 키사이즈와 블록 크기를 작게 설정하여 이를 저장하기 위한 레지스터 및 ALU의 크기를 줄이는 것이 중요하다. 또한 내부 상태를 유지하기 위한 저장 공간이 사용되는 경우 이에 따른 추가적인 부하가 요구되는 문제점을 가진다. 또한 복잡하지 않은 연산 구조를 활용하면 보다 효율적으로 연산이 가능하다. 적은 비트의 S-Box를 사용하거나 키 스캐줄 과정을 생략하게 되면 면적이 최소화되는 특성을 가진다. 워드 단위로 동일한 연산을 하는 경우 serialize를 통해 공유하는 연산이 많아져서 ALU를 적게 사용하고도 연산이 가능한 장점을 가진다.

시간 성능을 향상시키기 위해서는 면적을 최소화했던 특성들을 반대로 수행하는 과정이 요구된다. 만약 지나치게 공유하는 자원이 많아지는 경우 이를 수행하기 위해서는 이전 연산이 해당 ALU를 수행하고 난 다음에 해당 ALU를 사용해야 함으로 보다 많은 시간이 요구된다. 또한 bit 단위의 연산이 수행되는 경우 해당 bit를 처리하기 위해 보다 많은 시간이 byte 단위 연산에 비해 많이 요구되는 단점을 가진다.

Ⅳ. 성능 평가

4.1. 소프트웨어 성능 평가

사물인터넷 디바이스로 설정한 ATmega는 대표적인 8비트 마이크로 프로세서로써 8Mhz 이상의 클락으로 동작하며 128Kbyte 이상 의 EEPROM과 4Kbyte 이상의 RAM을 제공한다. 총 32개의 레지스터를 제공하며 사용자는 효율적인 연산 수행을 위해 메모리에 비해 소모되는 연산 복잡도가 낮은 레지스터를 잘 활용해야 한다. 초기 ATmega 모델에서는 가장 기본적인 대칭키 암호화인 DES를 제공했다. 이는 레지스터 주소에 암호화하고자 하는 값 정보를 넣으면 일정한 클락이 지난 이후에 값이 지정된 주소 레지스터에 저장되는 방식으로 수행된다.

8-bit word를 기반으로 설계된 AES가 8-bit 프로세서 상에서 가장 좋은 성능 보인다. 이는 연산들을 다중으로 수행해야 하는 큰 word 연산이 필요하지 않은 장점을 가진다. LEA의 경우 8 bit AVR에선 보통의 성능을 가진다. 그 이유는 LEA는 32-bit word를 기반으로 설계되어 Intel x86 혹은 ARMv7과 같이 32-bit 기기에서 좋은 성능을 나타내도록 하였기 때문이다. 최근에 제안된 Speck/Simon 중 소프트웨어를 목표로 설계된 Speck이 보다 높은 성능을 보임을 확인할 수 있다. Salsa의 경우 암호화 연산만 고려 시 LEA와 비슷한 성능을 나타낸다. 그 이유는 두 개의 암호화 모두 동일한 ARX 연산을 기반으로 설계되어 있기 때문이다. CLEFIA는 GF 상에서의 곱셈과 같은 높은 부하 연산이 필요한 단점을 가진다. 따라서 속도가 느리게 나타난다. TEA와 XTEA의 경우 단순한 연산을 64라운드를 연속 수행함으로써 높은 복잡도가 생기게 되는 알고리즘이다, 두 암호화는 비슷한 연산을 수행함으로써 유사한 복잡도를 나타내게 된다. HIGHT 암호화도 AES와 유사하게 8-bit word 단위로 설계되어 높은 성능을 나타낸다. PRESENT와 KATAN의 경우 구조는 간단하지만 Round의 수가 많아 수행 속도는 매우 느려지는 단점을 가진다. 이처럼 경량 암호화의 경우 간단한 연산을 반복하는 과정이 필요하기 때문에 구현 시 성능이 좋지 않은 결과를 보인다. 각각의 암호 알고리즘에 대한 성능은 [표 1]에 상세히 기술되어 있다.

〔표 1〕 AVR 프로세서 상에서의 블록암호화 성능 비교

Method	키생성+암호화 (Cycle / byte)	타겟	기본 연산
Speck 128-bit[2]	139(키생성 제외)	SW	ARX
AES 128-bit[3]	171	SW	SPN
Salsa 20 / 12[1]	177(키생성 제외)	SW	ARX
Simon 128-bit[2]	333(키생성 제외)	HW	ARX
IDEA 128-bit[9]	337	SW	ARX
HIGHT 128-bit[6]	370.5	HW	ARX
LEA 128-bit[11]	190(키생성 제외)	SW	ARX
PRINCE 80-bit[8]	406	HW	SPN
KLEIN 80-bit[9]	761	SW/ HW	SPN
SEA 96-bit[6]	804	SW	SPN
TEA 128-bit[6]	926	SW	ARX
XTEA 128-bit[4]	961	SW	ARX
Present 128-bit[6]	1417	HW	SPN
mCRYPTON 96-bit[10]	2057	HW	SPN
CLEFIA 128-bit[7]	2336	SW/ HW	SPN
KATAN 80-bit[9]	9007	HW	ARX

4.2. 하드웨어 성능 평가

하드웨어 구현은 크게 칩의 크기와 속도로 나누어 생각해 볼 수 있다. 칩의 크기는 목표로 하는 연산을 구현하기 위해 사용할 게이트의 수를 의미하며 속도는 몇 번의 clock이 지난 후에 원하는 결과값을 얻을 수 있는지를 의미한다. 두 성능은 상호간의 연관성이 매우 높아서 나누어 생각할 수 없다.

높은 성능을 나타내는 암호화 알고리즘은 mCrypton으로써 13 clock cycle안에 수행이 가능한 특징을 가진다. LED 암호화의 경우 칩의 크기가 다른 알고리즘에 비해 현저히 작음을 확인할 수 있다. LEA의 경우 빠른 속도로 암호화가 가능함을 확인할 수 있다. 하지만 워드의 크기가 32-bit를 기본으로 하기 때문에 이를 연산하기 위한 32-bit ALU와 register가 요구되기 때문에 칩의 크기가 증가하는 단점을 가진다. 다만 한번의 round를 한 clock 안에 수행하게 하는 기법을 적용할 경우 매우 효과적으로 연산이 가능한 장점을 가진다.

〔표 2〕 하드웨어 상에서의 블록암호화 성능 비교

Design	Area (GE)	Cycle	Throughput /Area
mCrypton-64[19]	2,420	13	0.19
mCrypton-96[19]	2,681	13	0.17
mCrypton-128[19]	2,949	13	0.16
PRESENT-80[13]	1,570 /1,075	32/547	0.12/0.01
PRESENT-128[13]	1,884/1,391	32/559	0.11/0.01
LED-64[12]	966	1,248	0.005
LED-96[12]	1,116	1,872	0.003
LED-128[12]	1,265	1,872	0.002
HIGHT[21]	2,766 /2,418	32/136	0.07/0.02
DESXL[26]	2,168	144	0.02
CLEFIA[15]	2,893	176	0.03
KATAN-32[22]	802	254	0.02
KATAN-64[22]	1,054	254	0.02
HB2-4C[20]	3,220	4	0.12
HB2-20C[20]	2,159	20	0.04
LEA-128[11]	4,549 /3,760	24/168	0.11/0.02
LEA-192[11]	5,505 /4,704	28/168	0.08/0.02
LEA-256[11]	5,015 /5,838	32/288	0.08/0.01

Ⅴ. 결 론

본 논문에서는 사물인터넷 서비스의 활성화를 위해 선행되어야 하는 경량 암호 알고리즘과 구현에 대해 확인해 보았다. 경량 암호 알고리즘은 최소한의 연산 복잡도와 코드/칩 크기를 제공하는 것이 가장 중요한 요소이다. 하지만 연산 복잡도와 코드/칩 크기는 언제나 상호연관성을 가지는 특성으로써 하나만을 목표로 선택할 수는 없다. 따라서 본 논문에서 살펴본 여러 알고리즘 중 어떤 기법이 가장 효율적이라고 단언하기에는 아직 이른감이 있다. 그 대신 적합한 경량 암호화 선택시 고려해야 할 부분은 소프트웨어/하드웨어 그리고 보안 강도로 생각해 볼 수 있으며 현재 표준과의 상호 호환성이다. 따라서 자신이 목표로하고 있는 서비스의 특징과 원하는 정도의 서비스 가용성을 고려하여 적합한 암호 알고리즘을 채택하는 것이 보다 현실적인 접근 방안이라고 생각할 수 있다.

참 고 문 헌

[1] Beaulieu, Ray, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. "The SIMON and SPECK Block Ciphers on AVR 8-bit Microcontrollers.", IACR ePrint.

[2] Beaulieu, Ray, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. "The SIMON and SPECK Families of Lightweight Block Ciphers." IACR Cryptology ePrint Archive 2013 (2013): 404.

[3] Bos, Joppe W., Dag Arne Osvik, and Deian Stefan. "Fast Implementations of AES on Various Platforms." IACR Cryptology ePrint Archive 2009 (2009): 501.

[4] AVR-Crypto-Lib, Available at https://www.das-labor.org/wiki/AVR-Crypto-Lib

[5] Papagiannopoulos, Konstantinos, and Aram Verstegen. "Speed and size-optimized implementations of the PRESENT cipher for tiny AVR devices." In Radio Frequency Identification, pp. 161-175. Springer Berlin Heidelberg, 2013.

[6] Eisenbarth, Thomas, Sandeep Kumar, Christof Paar, Axel Poschmann, and Leif Uhsadel. "A survey of lightweight-cryptography implementations." IEEE Design & Test of Computers 24, no. 6 (2007): 522-533.

[7] Rembrand van Lakwijk, "Comparing Performance of Software CLEFIA to Established Block Ciphers on 8-bit Devices," 15th Twente Student Conference on IT.

[8] Aria Shahverdi, Cong Chen, and Thomas Eisenbarth, "AVRprince - An Efficient Implementation of PRINCE for 8-bit Microprocessors"

[9] Eisenbarth, Thomas, Zheng Gong, Tim Güneysu, Stefan Heyse, Sebastiaan Indesteege, Stéphanie Kerckhof, François Koeune et al. "Compact im-

plementation and performance evaluation of block ciphers in ATtiny devices." In Progress in Cryptology-AFRICACRYPT 2012, pp. 172-187. Springer Berlin Heidelberg, 2012.

[10] mCrypton 결과, Available at http://perso.uclouvain.be/fstandae/lightweight_ciphers/

[11] Hong, Deukjo, Jung-Keun Lee, Dong-Chan Kim, Daesung Kwon, Kwon Ho Ryu, and Dong-Geon Lee. "LEA: A 128-bit block cipher for fast encryption on common processors." In Information Security Applications, pp. 3-27. Springer International Publishing, 2014.

[12] Rivest, Ronald L., M. J. B. Robshaw, Ray Sidney, and Yiqun Lisa Yin. "The RC6TM block cipher." In First Advanced Encryption Standard (AES) Conference. 1998.

[13] Bogdanov, Andrey, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew JB Robshaw, Yannick Seurin, and Charlotte Vikkelsoe. PRESENT: An ultra-lightweight block cipher. Springer Berlin Heidelberg, 2007.

[14] Daemen, Joan, and Vincent Rijmen. "AES proposal: Rijndael." (1999): 21.

[15] Shirai, Taizo, Kyoji Shibutani, Toru Akishita, Shiho Moriai, and Tetsu Iwata. "The 128-bit blockcipher CLEFIA." In Fast software encryption, pp. 181-195. Springer Berlin Heidelberg, 2007.

[16] Borghoff, Julia, Anne Canteaut, Tim Güneysu, Elif Bilge Kavun, Miroslav Knezevic, Lars R. Knudsen, Gregor Leander et al. "PRINCE–a low-latency block cipher for pervasive computing applications." In Advances in Cryptology–ASIACRYPT 2012, pp. 208-225. Springer Berlin Heidelberg, 2012.

[17] Standaert, François-Xavier, Gilles Piret, Neil Gershenfeld, and Jean-Jacques Quisquater. "SEA: A scalable encryption algorithm for small embedded applications." In Smart Card Research and Advanced Applications, pp. 222-236. Springer Berlin Heidelberg, 2006.

[18] Gong, Zheng, Svetla Nikova, and Yee Wei Law. "KLEIN: a new family of lightweight block ciphers." In RFID. Security and Privacy, pp. 1-18. Springer Berlin Heidelberg, 2012.

[19] Lim, Chae Hoon, and Tymur Korkishko. "mCrypton–a lightweight block cipher for security of low-cost RFID tags and sensors." In Information Security Applications, pp. 243-258. Springer Berlin Heidelberg, 2006.

[20] ENGELS, Daniel, et al. The Hummingbird-2 lightweight authenticated encryption algorithm. In: RFID. Security and Privacy. Springer Berlin Heidelberg, 2012. p. 19-31.

[21] HONG, Deukjo, et al. HIGHT: a new block cipher suitable for low-resource device. In: Cryptographic Hardware and Embedded Systems-CHES 2006. Springer Berlin Heidelberg, 2006. p. 46-59.

[22] DE CANNIERE, Christophe; DUNKELMAN, Orr; KNEŽEVIĆ, Miroslav. KATAN and KTANTAN—a family of small and efficient hardware-oriented block ciphers. In: Cryptographic Hardware and Embedded Systems-CHES 2009. Springer Berlin Heidelberg, 2009. p. 272-288.

[23] LAI, Xuejia; MASSEY, James L. A proposal for a new block encryption standard. In: Advances in Cryptology—EUROCRYPT'90. Springer Berlin Heidelberg, 1991. p. 389-404.

[24] Bernstein, Daniel J. "The Salsa20 family of stream ciphers." In New stream cipher designs, pp. 84-97. Springer Berlin Heidelberg, 2008.

[25] Wheeler, David J., and Roger M. Needham. "TEA, a tiny encryption algorithm." In Fast Software Encryption, pp. 363-366. Springer Berlin Heidelberg, 1995.

[26] Poschmann, Axel, Gregor Leander, Kai Schramm, and Christof Paar. "New Light-Weight DES Variants Suited for RFID Applications, proceedings of Fast Software Encryption 14." Lecture Notes in Computer Science, Springer (to appear) (2007).

〈저 자 소 개〉

서 화 정 (Hwajeong Seo)
종신회원

2010년 2월 : 부산대학교 컴퓨터공학과 학사 졸업
2012년 2월 : 부산대학교 컴퓨터공학과 석사 졸업
2012년 3월~현재 : 부산대학교 컴퓨터공학과 박사과정
관심분야 : 정보보호, 암호화 구현, IoT

김 호 원 (Howon Kim)
종신회원

1993년 2월 : 경북대학교 전자공학과 학사 졸업
1995년 2월 : 포항공과대학교 전자전기공학과 석사 졸업
1999년 2월 : 포항공과대학교 전자전기공학과 박사 졸업
2008년 2월 : 한국전자통신연구원 정보보호연구단 선임연구원/팀장
2008년 3월~현재 : 부산대학교 정보컴퓨터공학부 부교수
관심분야 : 스마트그리드 보안, RFID/USN 정보보호 기술, PKC 암호, VLSI 설계, embedded system 보안, IoT