경량 대칭키 암호 구현 기술

권혁동*, 임지환*, 우재민*, 안규황*, 서화정*

*한성대학교 IT융합공학부

hdgwon@naver.com, jhim000@naver.com, vlxksla123@naver.com, tigerk9212@gmail.com, hwajeong@hansung.ac.kr

Implementation Technology of Lightweight Symmetric Key Cryptography

Jihwan Lim*, Jae-Min Woo*, Hyeokdong Kwan*, Kyuhwang An*, Hwajeong Seo*

*Division of IT engineering, Hansung University.

요 약

정보의 기밀성을 확보할 수 있는 가장 효과적인 방법은 대칭키 암호화를 이용하여 정보를 안전하게 암호화하는 것이다. 지금까지 많은 대칭키 암호화 기술이 소개되었으며 이를 활용하여 사이버 공간 상에서의 사용자 비밀정보는 안전하게 보호되어 왔다. 최근에는 저전력 사물인터넷 플랫폼 상에서의 암호화 연산이 용이하도록 경량 대칭키 암호가 제안되고 있다. 경량 대칭키 암호 연산은 구현이 용이함은 물론 기존 대칭키 암호화와 동일한 보안강도를 제공하고 있다. 본 논문에서는 특히 경량 대칭키 암호인 PRESENT와 LEA에 대한 최신 구현을 32-비트 ARM Cortex-M3 프로세서 상에서 수행한 결과에 대해 확인해 보도록 한다.

I. 서론

우리 주위에 존재하는 모든 사물이 인터넷에 연결되는 사물인터넷 시대가 도래함에 따라 다량의 정보가 인터넷을 통해 유통되고 있다. 해당 정보는 공적인 정보를 포함하여 매우 민감한 정보를 포함할 수 있기 때문에 통신 중간에 적절한 인증을 받지 않은 사용자에 의한 불법 접근을 차단하는 것이 그 무엇보다 중요하다. 네트워크 상에서의 정보의 유출을 방지할 수 있는 가장 근본적인 기술로는 모든 데이터 패킷을 대칭키 암호화 기술을 통해 암호화 하는 것이다. 하지만 기존의 대칭키 암호화 기술은 저전력 사물인터넷 프로세서 상에서는 높은 오버헤드를 발생시키는 만큼 이를 대체하기 위한 경량 암호화 기술이 최근에는 활발히 연구 및 배포되고 있다. 본 논문에서는 경량 대칭키 암호인 PRESENT와 LEA에 대한 최신 구현을 32-비트 ARM Cortex-M3 프로세서 상에서 수행한 결과에 대해 확인해 보도록 한다.

본 논문의 구성은 다음과 같다. 2장에서는 PRESENT와 LEA 블록암호화 그리고 타겟프로세서인 ARM Cortex-M3 프로세서에 대해 확인해 보도록 한다. 3장에서는 최신 암호화 구현 기법에 대해 확인해 보도록 한다. 마지막으로 4장에서는 본 논문에 대한 결론을 내린다.

I. 관련 연구

2.1 PRESENT

CHES'07에서 발표된 경량 블록 암호화 PRESENT는 SPN 구조를 가진다 [1]. 블록 크기는 64-비트이며 키크기는 80-비트와 128-비트 두 가지 모드를 지원한다. S-box의 크기는 4-비트 기반으로 설계되어 초경량 디바이스 상에서 매우 효율적인 구현이 가능하다. Permutation 연산의 경우 하드웨어 상에서는 회로 연결을 통해 추가적인 연산없이 구현 가능하다.

2.2 LEA

NSR에 의해 개발되어 WISA'13에서 발표된 LEA 블록암호화는 ARX 구조에 기반을 둔 경량 암호화 기법이다 [2]. 지원하는 암호화 비트는 128, 192, 그리고 256-비트이며 블록의 크기는 128-비트이다. 특히 ARX 연산자는 32-비트 단위로 동작한다. 경량화된 연산자는 8-비트 AVR 프로세서와 32-비트 ARM 프로세서 상에서 효율적이며 특히 최신 병렬화 엔진인 SSE, AVX, 그리고 NEON를 활용하면 높은 성능을 기대할 수 있다.

2.3 ARM Cortex-M3 프로세서

ARM Cortex-M3 프로세서는 산업체에서 많이 사용되는 32 비트 프로세서이다. 저전력으로 동작할 뿐 아니라 가격이 저가이며 실시간 임베디드 응용 프로그램에 적합한 특성을 가진다. 프로세서는 ARMv7-M 구조를 따르며 3단계 파이프 라인을 지원한다. 또한 Thumb-1과 Thumb-2 명령어 셋을 모두 지원한다. Thumb-1의 경우 기존 32-bit 크기의 ARM 명령어와는 달리 16-bit로 명령어를 정의하는 방안으로써 코드크기를 획기적으로 줄임과 동시에 기존 ARM의 성능을 유지하는 장점을 가진다. Thumb-2의 경우에는 ARM과 Thumb-1 명령어를 혼용해서 사용함으로써 성능과 코드 크기의 적절한 절충점을 찾은 명령어 셋으로 볼 수 있다. 이를 이용하면 ARM의 성능을 Thumb-1의 코드 크기로 달성하는 것이 가능하다.

II. 최신 구현 기법

2.1 PRESENT

SPN 구조를 가지는 PRESENT의 경우에는 경량 프로세서 상에서의 구현보다는 고성능 SIMD 프로세서 상에서 bitslicing기법을 활용하여 효율적인 연산이 가능함을 증명하였다. bitslicing 기법은 기존의 워드 단위의 연산 체계가 아니라 워드를 비트 단위로 나눈 다음 해당 비트 단위로 연산을 수행함으로써 연산 성능을 향상시키는 기법이다. 특히 PRESENT 블록암호화의 경우 permutation 연산이 비트단위로 수행되는 특징으로 인해 bitslicing 기법이 활발히 연구되고 있다. 하지만 bitslicing 기법이 효과적으로 적용되기 위해서는 한 번에 많은 암호화 연산을 수행해야 하는 경우를 선정해야 한다. 혹은 하드웨어와 같이 비트단위로 연산을 수행할 수 있어야 한다.

하지만 마이크로 프로세서는 매우 한정적인 레지스터 저장 공간을 가지고 있기 때문에 많은 연산을 모두 레지스터에 저장하는 것이 불가능할 뿐 아니라 비트 단위 연산자를 기본적으로 제공하지 않는 문제점을 가지고 있다. 이를 효과적으로 개선하기 위해 최근에 CHES'17에서 발표된 논문에서는 Permutation과 S-box 연산의 순서를 교환하여 암호화 연산을 적용하는 기술이 제안되었다 [3]. 해당 기법은 32-비트 ARM 프로세서 상에서 구현 시 16-비트 단위 연산에 필요한 두 개의 인자를 하나의 32-비트 워드에 넣어 구현하는 방법을 취했다. 그리고 S-box 연산을 기존 Look-Up Table (LUT)가 아닌 연산의 조합으로 풀어서 구현하는 방법을 취했다. Permutation을 먼저 수행하게 되는 경우 워드 상의 값들이 S-box를 효율적으로 수행할 수 있는 형태로 변경되는 특징을 가진다. 이 형태라고 하는 것은 4-비트 S-box가 주어진다면 각각의 비트 위치가 일치하도록 데이터 정렬이 용이함을 의미한다. 따라서 제안되는 기법에서와 같이 S-box를 Permutation과 순서를 달리 하여 구현하게 된다면 S-box를 bitslicing으로 효과적으로 연산이 가능하다는 장점을 가진다.

2.2 LEA

LEA 블록 암호화는 32-비트 단위의 Addition-Rotation-XOR (ARX) 연산으로 구성되어 있다. 해당 연산은 32-비트 ARM 프로세서 상에서는 효율적으로 구현이 가능하다 [4]. 그 이유는 ARM 프로세서 자체적으로 제공하는 명령어 종류가 전부 32-비트 단위의 연산을 제공하기 때문이다. 덧셈과 XOR 연산의 경우 ARM 프로세서 상에서 제공하는 ADD 혹은

EOR 연산자를 활용하면 1 clock cycle 안에 구현이 가능하다. Rotation 연산의 경우에도 offset에 상관없이 사용자가 원하는 연산을 1 clock cycle안에 구현가능한 장점을 가진다. ARM 프로세서가 가지는 또 하나의 특징은 Barrel shifter이다. 해당 특징을 이용하면 두 번째 인자로 들어가는 연산자에 대한 회전 연산을 clock cycle의 소모없이 구현 가능한 장점을 가진다. 따라서 3 종류의 명령어 셋을 효과적으로 조합하게 될 경우 매우 효율적으로 LEA를 ARM 프로세서 상에서 구현하는 것이 가능하다. 두 최신 구현기법을 통해 구현한 결과에 대한 성능 평가를 확인해 보면 표 1과 같다. PRESENT와 LEA 블록암호화를 비교해 보면 LEA 암호화가 PRESENT 보다 나은 성능을 나타내게 된다. 그 이유는 LEA의 경우 기본적인 워드 연산이 ARM 프로세서의 워드와 동일하여 대부분의 연산자가 1 clock cycle안에 연산 가능한 특징을 가지기 때문이다. 반면에 PRESENT는 4-비트 S-box와 비트 단위의 permutation 연산의 경우 32-비트 ARM 프로세서 상에서는 매우 비효율적인 특징을 가진다. 따라서 이를 효과적으로 구현한 최신 bitslicing 기법의 경우에도 LEA에 비해서는 매우 느린 성능을 나타냄을 확인할 수 있다.

[표 1] Performance evaluation of block cipher on 32-bit ARM processors where (C/B) is cycle per byte.

Algorithm	Encryption (C/B)
PRESENT-128 [3]	264
LEA-128 [4]	31

III. 결론

최근 들어 경량 블록암호화에 대한 연구가 활발히 진행되고 있다. 이전과 달리 새로운 블록암호화 설계 시 가장 중요시 되는 부분은 해당 암호화가 과연 현존하는 컴퓨터 상에서 얼마만큼의 높은 성능을 보여줄 수 있는지이다. 이를 위해 암호 엔지니어들은 새로운 블록암호를 분석하여 이를 새로운 컴퓨터 구조에 적용하는 일에 많은 노력을 쏟고 있다. 본 논문에서 살펴본 새로운 구현 기법을 기반으로 저전력 사물인터넷 환경 상에서도 안전하고 고속으로 연산을 수행하는 것이 가능함을 확인할 수 있었다. 앞으로도 이에 대한 연구가 지속적으로 이루어져서 보다 나은 성능을 가지는 결과가 도출될 수 있으면 좋을 것으로 생각된다.

Acknowledgement

본 연구는 고려대 암호기술 특화연구센터(UD170109ED)를 통한 방위사업청과 국방과학연구소의 연구비 지원으로 수행되었습니다.

[참고문헌]

[1] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, C. Vikkelsoe, "PRESENT: An ultra-lightweight block cipher," Conference on Cryptographic Hardware and Embedded Systems (CHES'07), vol. 4727, pp. 450-466, 2007.

[2] D. Hong, J. K. Lee, D. C. Kim, D. Kwon, K. H. Ryu, D. G. Lee, "LEA: A 128-bit block cipher for fast encryption on common processors," In International Workshop on Information Security Applications (WISA'13), pp. 3-27, 2013.

[3] T. Reis, D. Aranha, J. López, " PRESENT Runs Fast: Efficient and Secure Implementation in Software," *Conference on Cryptographic Hardware and Embedded Systems (CHES'17)*, 2017.

[4] H. Seo, I, Jeong, J. Lee, W. H. Kim, "Compact Implementations of ARX Based Block Ciphers on IoT Processors," *ACM Transactions on Embedded Computing Systems (ACM-TECS)*, vol.17, no. 3, pp. 60:1-60:16, 2018.