32-bit RISC-V 상에서의 LEA-CTR 최적화 구현

엄시우*, 권혁동*, 김현지*, 양유진**, 서화정***

*한성대학교 (대학원생), **한성대학교 (대학생), ***한성대학교 (교수)

Optimized Implementation of LEA-CTR on 32-bit RISC-V

Si-Woo Eum*, Hyeok-Dong Kwon*, Hyun-Ji Kim*, Yu-Jin Yang**, Hwa-Jeong Seo***

*Hansung University(Graduate student)

**Hansung University(Undergraduate student)

***Hansung University(Professor)

요 약

사물인터넷의 발전으로 제한된 환경에서 효율적이며 안전한 경량 암호의 사용이 증가되고 있다. 본 논문에서는 32-bit RISC-V 상에서의 국산 경량 블록암호 LEA의 CTR 운용 모드 최적화 구현을 제안한다. CTR 운용 모드의 Nonce 값이 고정되는 특성을 활용하여 사전 연산을 통한 최적화 기법과 암호화 과정에서의 값의 이동을 생략하는 최적화 기법을 제안한다. 제안 기법을 통해 암호화 과정에서 6번의 라운드키 XOR 연산, 3번의 Addition 연산, 3번의 Rotation 연산이 생략 가능하다. 또한 값의 이동을 생략하여 $X_i[0]$의 값이 $X_{i+1}[3]$ 으로 이동하는 연산을 24번 생략 가능하다. 결과적으로 제안 기법 적용을 통하여 기존 연구 LEA 최적화 구현 대비 2%의 성능 향상을 확인하였다.

I. 서론

사물인터넷의 발전으로 다양한 기기에서 암호가 사용되고 있다. 다양한 기기들은 각각의 다양한 컴퓨팅 환경을 가지며, 이 중 제한된 컴퓨팅 환경에서 동작하는 기기에서 사용하기 위한 암호가 개발되고 있다. 2015년부터 NIST(National Institute of Standards and Technology)에서도 이런한 제한된 컴퓨팅 환경에서 사용할 경량 암호 공모전을 개최하였으며 여러 경량 암호 알고리즘이 발표되고 있다. 국내에서도 LEA, HIGHT, PIPO등과 같은 경량 블록암호가 개발되어 있으며, 최적화 연구 또한 활발하게 진행되고 있다.

본 논문에서는 국내 경량 블록암호인 LEA의 CTR 운용모드 최적 구현을 제안한다. 현재까지 LEA-CTR 최적 구현 연구는 8-bit AVR 마이크로컨트롤러와 ARMv8 마이크로컨트롤러 상에서의 최적화 구현 연구가 제안되었다[1][2]. 본 논문에서는 기존 연구의 최적화 기법을 참고하여 32-bit RISC-V 플랫폼 상에서의 최적화 구현을 진행한다.

본 논문의 구성은 다음과 같다. 2장에서는 경량 블록암호 LEA, CTR 운용 모드, 그리고 RISC-V에 관하여 설명한다. 3장에서는 제안 기법을 설명한다. 4장에서는 제안 기법의 성능을 평가한다. 마지막으로 5장에서는 본 논문의 결론을 내린다.

II. 관련 연구

2.1 LEA 경량 블록암호

LEA는 빅데이터, 클라우드등 고속 환경뿐만 아니라 IoT기기, 모바일기기 등 경량 환경에서도 기밀성을 제공하기 위해 2013년 국내에서 개발된 경량 블록암호이다. 블록 길이는 128-bit, 키 길이는 128, 192, 256-bit 세 종류를 지원한다[3]. 자세한 매개 변수는 [표 1]과 같다. 가장 널리 사용되는 AES 대비 약 1.5배 빠른 암호화가

가능하며, 2019년 경량 블록 암호 분야 표준(ISO/IEC 29192-2:2019)으로 제정되었다[4].

Cipher	n	k	rk	r
128/128	128	128	192	24
128/192	128	192	192	28
128/256	128	256	192	32

Table 1. Parameters of LEA; n:block size, k:Key size, rk:round key size, r:number of rounds

알고리즘 구조는 ARX(Addition, Rotation, eXclusive-or)로 이루어져 있으며, 입력받은 블록은 32-bit State로 나뉘어져 암호화가 진행된다. 전체적인 알고리즘 구조는 [그림 1]과 같다.

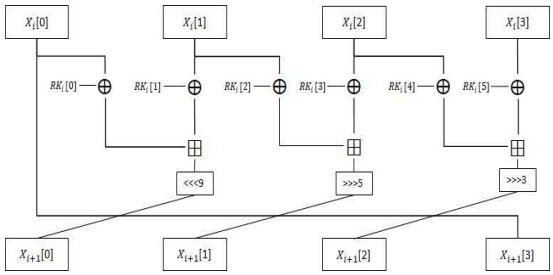

Fig 1. Algorithm Structure of LEA

2.2 CTR(Counter) 운용 모드

블록 암호 운용 방식에는 ECB, CBC, CTR 등 여러 운용 방식이 존재한다. 그 중 CTR 운용 모드는 고정된 상수를 사용하는 Nonce 값과 변수인 Counter 값이 결합한 값을 입력 값으로 사용하여 암호화를 진행하고 마지막에 평문과 XOR 하는 방식으로 암호화가 진행된다. 이때 Counter 값을 통해 현재 블록이 몇 번째 블록인지 알 수 있으며, 각 블록이 이전 블록에 의존하지 않기 때문에 병렬적으로 동작하는 것도 가능하다.

2.3 RISC-V

캘리포니아 대학교 버클리에서 2010년부터 개발중인 RISC(Reduced Instruction Set Computer) 기반의 컴퓨터 아키텍처이다[5]. RISC-V는 RV32I, RV64I의 두 가지 모델이 있으며, 각각 32-bit, 64-bit 레지스터를 사용한다. 본 논문에서 사용한 32-bit 구조의 RV32I는 32-bit 레지스터 32개를 제공하며, 각각의 레지스터 용도는 [표 2]와 같다[6].

Register	Description	Saver
zero(x0)	Zero register	
ra(x1)	return address	
sp(x2)	stack pointer	callee
gp(x3)	global pointer	
tp(x4)	thread pointer	
a0~a7	function arguments and return value	
s0~s11	saved registers	callee
t0~t6	temporal registers	

Table 2. Purpose of RISC-V Register

III. 제안 기법

본 논문에서 최적화를 위해 제안하는 기법은 두 가지이다. 첫 번째는 CTR 운용 모드의 특성을 활용한 사전 연산 최적화 기법이다. 두 번째는 레지스터간의 이동을 생략하고 고정된 레지스터로 연산을 진행하는 고정 레지스터 사용 최적화 기법이다.

3.1 사전 연산 최적화

CTR 운용 모드는 고정된 Nonce값과 변수인 Counter 값을 결합하여 입력 값으로 사용한다. 이때 고정된 Nonce 값으로 인하여 Counter 값이 바뀌어도 특정 라운드까지 암호화가 진행될 때 각 라운드에서 고정된 값이 존재하게 된다. 따라서 사전 연산을 통해 고정된 값을 얻고 암호화를 진행하게 되면 고정된 값을 얻을 때를 제외하고 다음 블록의 암호화가 진행될 때는 일정 부분의 연산을 생략할 수 있다.

라운드가 진행됨에 따라 확산이 이루어지는 과정과 고정되는 값은 [그림 2]와 같다.

Input	Counter(32-bit)	Nonce(32-bit)	Nonce(32-bit)	Nonce(32-bit)
ROUND 1	$X_1[0]$	$X_1[1]$	$X_1[2]$	$X_1[3]$
ROUND 2	$X_2[0]$	$X_2[1]$	$X_2[2]$	$X_2[3]$
ROUND 3	$X_3[0]$	$X_3[1]$	$X_3[2]$	$X_3[3]$

Fig 2. Counter value diffusion process

[그림 2]에 색이 채워진 부분은 Counter 값이 변하였을 때, 영향을 받는 부분이다. Counter 값이 변하였을 때 각 라운드가 진행됨에 따라서 영향을 받는 부분이 고정되어 있는 것을 확인할 수 있으며, 3라운드에서 전체 확산이 이루어지는 것을 확인 할 수 있다. 1라운드에서는 X_1 [1], X_1 [2], 2라운드에서는 X_2 [1]의 값이 Counter 값이 증가하여도 고정된 Nonce 값으로 인해 값이 고정된다. 따라서 해당 값을 매번 연산할 필요 없이 사전 연산을 통해 값을 얻고, 다음 블록을 연산할 때는 해당 값을 얻기 위한 연산을 생략할 수 있다.

또한 암호화 입력 값으로 사용되는 Nonce 값 96-bit 전부를 불러올 필요 없이 X_1 [0]을 얻기 위해 사용되는 32-bit Nonce 값만 필요하다. X_1 [3] 값의 경우 Counter 값이 그대로 옮겨지기 때문에, 결론적으로 1라운드에서의 연산은 X_1 [0]의 값만 연산하게 된다.

결과적으로 블록 한 개의 암호화 과정에서 6번의 라운드키 XOR 연산, 3번의 Addition 연산, 3번의 Rotation 연산이 생략 가능하다.

3.2 고정 레지스터 사용

LEA의 암호화 과정에서 각 State들은 다음 라운드로 넘어가면서 옆 State와 자리를 바꿔가며 암호화가 진행된다. 또한 $X_i[0]$ 값은 연산 과정 없이 $X_{i+1}[3]$으로 값이 이동하게 된다. 이러한 이동하는 과정을 생략하고 고정된 자리에서 연산을 진행하게 되면 값이 이동하는 연산을 생략할 수 있다. 즉 $X_i[0]$의 값이 다음 라운드에서도 $X_{i+1}[0]$ 으로 그대로 고정되게 된다.

각 State의 이동이 생략됨에 따라서 연산의 각 라운드에 맞춰 연산의 수정이 필요하게 된다. 4개의 State가 라운드당 한 칸씩 이동하며 암호화가 진행되기 때문에 4라운드가 진행되게 되면 처음의 위치로 다시 돌아오게 된다. 따라서 4라운드 단위로 반복된 연산을 진행하게 된다.

이로 인해 전체 암호화 과정에서 128-bit 키 길이 기준으로 보았을 때, 24라운드가 진행되기 때문에 $X_i[0]$의 값이 $X_{i+1}[3]$ 으로 이동하는 연산을 24번 생략 가능하다.

IV. 성능 평가

본 논문에서는 SiFive 사의 HiFive RevB 보드를 사용하여 측정을 진행하였으며, SiFive 사의 Freedom Studio를 사용한다.

32-bit RISC-V 상에서의 LEA-CTR 최적화 구현에 관한 기존 연구가 없다. 같은 32-bit RISC-V 플랫폼 상에서 LEA 최적화 구현 연구인 Kwak et al.[7]과 성능 비교를 진행하였다.

	Cycle
Kwak et al.[7]	775
Our Work	757

Table 3. Performance Result on 32-bit RISC-V

[표 3]은 제안 하는 기법을 적용한 LEA-CTR의 Cycle을 나타낸다. 128-bit 하나의 블록을 암호화 할때의 Cycle을 나타내며, LEA 최적화 구현을 진행한 기존 연구 대비 2%의 성능 향상을 확인하였다.

V. 결론

본 논문에서는 32-bit RISC-V 상에서 국산 경량 블록암호인 LEA의 CTR 모드 최적화 구현을 제안한다. 제안하는 기법은 CTR 운용 모드에서의 고정된 Nonce 값의 특성을 활용하여 사전 연산을 통한 연산 생략과 State 고정을 통해 State간 값의 이동을 생략한 최적화 구현을 제안한다. 제안 기법을 통해 암호화 과정에서 6번의 라운드키 XOR 연산, 3번의 Addition 연산, 3번의 Rotation 연산이 생략 가능하다. 또한 값의 이동을 생략하여 $X_i[0]$의 값이 $X_{i+1}[3]$ 으로 이동하는 연산을 24번 생략 가능하다. 결과적으로 기존 LEA 최적화 연구 대비 2%의 성능 향상을 확인하였다. 향후 과제로 여러 국산 블록암호의 여러 운용모드에서의 최적화 구현을 제안한다.

VI. Acknowledgment

이 논문은 부분적으로 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (<Q|Crypton>, No.2019-0-00033, 미래컴퓨팅 환경에 대비한 계산 복잡도 기반 암호 안전성 검증 기술개발, 25%) 그리고 부분적으로 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2018-0-00264, IoT 융합형 블록체인 플랫폼 보안 원천 기술 연구, 25%) 그리고 부분적으로 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2021-0-00540, GPU/ASIC 기반 암호 알고리즘 고속화 설계 및 구현 기술개발, 25%) 그리고 부분적으로 2021년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. NRF-2020R1F1A1048478, 25%).

[참고문헌]

[1] Y.B. Kim, H.D. Kwon, et al. “Efficient Implementation of ARX-Based Block Ciphers on 8-bit AVR Microcontrollers”. Mathematics 2020. 8(10). 1837.

[2] J.G. Song, S.C. Seo. “Efficient Parallel Implementation of CTR Mode of ARX-Based Block Ciphers on ARMv8 Microcontrollers”. Applied Sciences 2021. 11(6). 2548.

[3] D.J Hong, et al. “LEA: A 128-Bit Block Cipher for Fast Encryption on Common Processors”. International Workshop on Information Security Applications; WISA 2013; Information Security Applications ;pp. 3-27.

[4] ISO. ISO/IEC 29192-2: 2019: Information Security-Lightweight Cryptography-Part 2: Block Ciphers; International Organization for Standardization: Geneva, Switzerland, 2019.

[5] Waterman, Andrew, et al. "The risc-v instruction set manual, volume i: Base user-level isa." EECS Department, UCBerkeley, Tech. Rep. UCB/EECS-2011-62 116, 2011

[6] SiFive, Inc. “The RISC-V Instruction Set Manual Volume I: User-Level ISA Document Version 2.2”. May 7, 2017. https://riscv.org/wp-content/uploads/2017/05/riscv-spec-v2.2.pdf

[7] Y.J. Kwak, Y.B. Kim, S.C. Seo. "Benchmarking Korean Block Ciphers on 32-Bit RISC-V Processor", Journal of the Korea Institute of Information Security & Cryptology, vol. 31, Issue. 3, pp. 331-340, 2021.