PIPO 경량 블록암호 최적 구현 기술 동향

이민우[1], 김동현[2], 윤세영[2], 서화정[3]
[1]한성대학교 융합보안학과 석사과정
[2]한성대학교 IT융합공학부 학부생
[3]한성대학교 융합보안학과 부교수
minunejip@gmail.com, donghyun6469@gmail.com, sebbang99@gmail.com,
hwajeong84@gmail.com

PIPO block cipher optimal implementation technology trend

Min-Woo Lee[1], Dong-Hyun Kim[2], Se-Young Yoon[2], Hwa-Jeong Seo[3]
[1]Dept. of Convergence Security, Hansung University
[2]Dept. of IT convergence Engineering, Hansung University
[3]Dept. of Convergence Security, Hansung University

요 약

본 논문은 PIPO 알고리즘의 최적 구현 기술들에 대한 연구 동향을 살핀다. PIPO는 선형, 차분 공격에 안전한 S-box를 사용하는 SPN 구조의 경량 블록 암호 알고리즘이다. 블록 크기는 64비트이고 비밀키 크기에 따라 PIPO-128과 PIPO-256으로 나뉜다. PIPO 알고리즘의 S-Layer, R-Layer, Addroundkey의 3가지 내부 동작과정과 각 라운드에서 사용되는 연산들에 대한 자세한 설명이 제공된다. 본 논문에서는 RISC-V 및 ARM 프로세서, CUDA GPGPU에서 PIPO 알고리즘을 최적화 구현하는 방법을 다룬다. 해당 연구들에선 최적 구현 기술을 적용하여 PIPO 암호를 적용하는 IoT 장치에서도 안전하고 빠른 암,복호화를 수행할 수 있음을 보였고, 기존 연구와의 비교를 통해 성능 향상이 이루어짐을 확인할 수 있다.

1. 서론

최근 사회 전반적으로 IoT 기기의 사용량이 증가하고 있다. IoT 장치 및 센서는 개인 정보가 포함된 민감한 데이터를 수집하여 전송할 수 있으므로 이를 안전하게 지킬 보안 수단의 필요성이 높아졌다. IoT 기기들은 데스크탑에 비해 낮은 cpu 성능, 적은 메모리 등 제한된 리소스를 지니고 있기 때문에 경량 암호를 통한 보안이 요구된다. PIPO는 경량화 및 저전력을 요구하는 환경에서 기밀성을 제공하기 위해 제안된 암호 기법이다. 리소스가 제한된 IoT 장치 상에서 구현하기 용이하게 경량화 되어있다. 본 논문에서는 국산 경량 블록암호인 PIPO 알고리즘에 대한 최적 구현 기술들을 알아본다.

2. PIPO 암호 알고리즘

ISISC 2020에 제안된 PIPO는 SPN 구조의 경량 블록 암호 알고리즘이다. PIPO 알고리즘은 선형, 차분 공격에 안전한 S-box를 설계했으며 Unbalanced bridge 구조를 이용했다. S-box 구조는 테이블 참조 구현인 TLU 방식 외에 비트슬라이스(BitSlice) 방식의 구현이 가능하다. 한 개 블록 단위로 비트슬라이스 구현이 가능하며, 효율성과 성능이 높은 소프트웨어 구현이 가능하다. PIPO 알고리즘의 블록 크기는 64 비트이고, 비밀키의 크기에 따라서 PIPO-128과 PIPO-256으로 나뉘며, 각각 13라운드와 17라운드를 사용한다. PIPO 라운드의 각 연산 중간값은 64비트이다. PIPO 알고리즘의 규격은 <표1>과 같다[1].

<표 1> PIPO 알고리즘 규격

Type	Block Size	Key Size	Round
64/128	64-bit	128-bit	13
64/256	64-bit	256-bit	17

PIPO는 S-Layer, R-Layer, Addroundkey의 3가지 내부 동작과정으로 구성된다. S-Layer는 8비트 입력과 8비트 출력 연산 과정을 거친다. PIPO의 S-Layer는 5비트 입/출력 S-box 2개와 3비트 입/출력 S-box 1개의 총 3개 S-box를 사용한다.

S-Layer는 사전 계산된 TLU 방식이나 비트 슬라이스 과정으로 처리한다. R-Layer는 암호화 과정 내부에서 Rotation 연산 처리를 거치는 과정이다. PIPO 암호의 내부 블록 크기는 64비트이며, R-Layer를 처리하기 위해 8비트 단위로 내부 상태를 Rotation 연산 처리한다.

3. 연구 동향

3.1 32-bit RISC-V 상에서의 PIPO 최적화 구현

RISC-V는 UC 버클리 대학에서 2010년부터 개발중인 RISC(ReducedInstruction Set Computer) 기반의 컴퓨터 아키텍처이다. RISC-V는 RV32I, RV64I의 2가지 모델이 있고, 각각 32비트와 64비트 레지스터를 사용한다. 해당 논문[2]에서는 32비트를 지원하는 RV32I를 사용하며 ECB, CBC, CTR 3가지 운용모드의 단일 블록에 대한 최적화 구현과 병렬 최적 구현을 진행한다. 단일 블록 구현에서는 32비트 레지스터 상에서의 효율적인 8비트 단위 R-Layer 함수 구현을 제안한다. 병렬 구현에서는 레지스터 내부 정렬을 진행하고, 서로 다른 4개 블록이 하나의 레지스터 상에서 R-Layer 함수 연산을 진행하는 방법을 설명한다. CBC 운용모드의 병렬 구현에선 암호화 과정의 병렬 구현 기법 적용이 어렵다. 이에 해당 논문에서는 CTR 운용모드의 병렬 구현에서 확장된 초기화 벡터를 사용하며 레지스터 내부 정렬을 생략하는 기법을 제안한다. 또한 병렬 구현 기법이 여러 블록암호 운용 모드에 적용 가능함을 보여준다.

해당 논문에서는 단일 블록 구현에서 상위 24비트를 무시하여 연산을 진행하고, 상위 24비트에 영향을 받는 R-Layer 함수에서 상위 24비트를 0으로 바꾸는 작업을 추가하여 최적화 구현을 진행하였다. 병렬 구현에서는 운용모드 별로 최적화 구현을 진행하였다. 이 중 CBC 운용 모드에서는 하나의 데이터를 병렬 구현하지 않고, 서로 다른 데이터를 병렬구현 하고 복호화 과정에서의 병렬 구현을 제안했다. 해당 논문에서 제안한 기법을 적용해 성능을 측정한 결과, 키 스케줄을 암호화 과정에 포함해 구현한 기존 연구 대비 단일 구현에서는 1.7배의 성능 향상을 확인하였으며, 병렬 구현에서는 1.89배 성능 향상을 확인했다. 또한 병렬 구현 기법을 ECB 운용 모드 외에 CTR, CBC 운용 모드에도 큰 성능의 차이 없이 적용할 수 있음을 보였다.

3.2 CUDA GPGPU 상에서의 PIPO 최적 구현

GPU 최적화는 GPU가 이론적으로 얻은 암호를 분석하거나, 축소된 버전을 합리적인 시간 내에 검증하는데 사용될 수 있다. 해당 논문[3]에서는 다양한 환경에서 구현되고 있는 PIPO 경량 블록암호를 대상으로 GPU 상에서 구현하였다. 또한 PIPO에 대한 무차별 대입 공격을 고려하여 최적 구현하였다. 특히 비트슬라이스 기법을 적용한 최적화 구현과 GPU 요소를 최대한 사용했다. 비트슬라이스 기법은 여러 블록을 암호화하므로, GPU 환경과 같이 높은 데이터 처리량을 요구하는 환경에 적합하다. GPU 코어의 32비트 연산 단위에 따라 평문 32블록에 대한 병렬 연산을 구현하였다. RTX 3060 상에서의 Visual Studio 환경에서 구현하였으며, 그리드 당 블록 수와 블록 당 스레드 수를 조절하며 성능을 측정하였다. 카운터 기반 키 탐색 기법을 적용하지 않았을 때 RTX 3060 상에서 블록 당 스레드 수 64, 그리드 당 블록 수 1024에서 초당 약 3.6177억회 암호 연산을 수행했다. 블록 당 스레드 수가 많을수록 더 높은 성능을 보였다. 이전 연구인 [4]의 PIPO 병렬 암호화 구현에서 RTX 3070에서 초당 약 1.59억회 암호연산을 처리한 결과보다 높은 처리량을 보였다. 결과적으로 제안된 기법의 구현은 RTX 3060 환경에서 초당 약 195억의 처리량을 보였으며, 이전 연구보다 약 122배 높은 처리량을 달성했다.

3.3 64-bit ARM 프로세서 상에서의 PIPO 병렬 최적 구현

해당 논문[4]에서는 ARM 프로세서를 대상으로 PIPO 블록 암호의 병렬 최적 구현을 제안한다. 제안하는 구현물은 8평문과 16평문의 병렬 암호화가 가능하다. 구현에는 레지스터 내부 정렬, 최적의 명령어 활용, 로테이션 연산 최적화 기법을 사용했다. 구현은 A10x fusion 프로세서를 대상으로 한다. 해당 프로세서 상에서 기존 PIPO 레퍼런스 코드는 64/128, 64/256 규격에서 각각 34.6 cpb, 44.7 cpb의 성능을 보였고, 제안하는 기법은 8평문 64/128, 64/256 규격에서 각각 12.0 cpb, 15.6 cpb의 성능을 보였다. 16평문 64/128, 64/256 규격에서 각각 6.3cpb, 8.1 cpb의 성능을 보였다. 성능 향상의 가장 큰 원인은 병렬 구현에 있다. 기존 구현은 병렬 연산이 불가능해 1개의 평문만을 암호화 할 수 있다. 그러나 제안된 구현물은 8개 혹은 16개의 평문을 동시에 암호화 할 수 있다. 즉 단위 시간 당 처리 할 수 있는 바

이트의 수가 늘어났기에 기존 대비 향상된 성능을 보인다. 또한 어셈블리 명령어를 활용한 최적 구현을 통해 성능 향상을 보였다. 특히 R-Layer의 로테이션 연산을 두 개의 명령어만 사용하여 구현하여 동작 속도를 크게 개선하였다. 결론적으로, 기존 대비 각 규격 별 8평문 병렬 구현물은 약 65.3%, 66,4% 향상된 성능을 보였으며, 16평문 병렬 구현은 약 81.8%, 82.1% 더 좋은 성능을 보였다.

4. 결론

본 논문에서는 PIPO 경량 블록암호의 최적화 구현 연구를 소개하였다. 32비트 RISC-V와 64비트 ARM 프로세서 상에서의 최적 구현 연구가 진행되었고, CUDA GPGPU 상에서의 최적 구현 연구가 수행되었다. 해당 연구들에선 최적 구현 기술을 적용하면 PIPO 암호를 적용한 IoT 장치에서도 안전하고 빠른 암,복호화를 수행할 수 있음을 검증하였다. PIPO 알고리즘은 향후 스마트 제어 설비나 스마트 홈과 같은 IoT 환경 등에 사용될 수 있는 초경량 암호 알고리즘이기에, 최적 구현을 통해 성능 향상을 이룬 연구는 PIPO를 적용하여 효율적으로 사용하는데 도움을 줄 수 있다. 하지만 현재 PIPO 최적 구현이 이루어진 환경은 많지 않으며, 다양한 IoT 환경에 PIPO 알고리즘이 탑재될 수 있기에 추후 더 많은 환경에서의 최적화 및 구현 기법을 통한 성능 향상 연구가 필요하다.

5. Acknowledgements

This work was partly supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2018-0-00264, Research on Blockchain Security Technology for IoT Services, 50%) and this work was partly supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2022-0-00627, Development of Lightweight BIoT technology for Highly Constrained Devices, 50%).

참고문헌

[1] Kim, Hangi, et al. "PIPO: A lightweight block cipher with efficient higher-order masking software implementations." Information Security and Cryptology–ICISC 2020: 23rd International Conference, Seoul, South Korea, December 2–4, 2020, Proceedings 23. Springer International Publishing, 2021.

[2] 엄시우, et al. "32-bit RISC-V 상에서의 PIPO 경량 블록암호 최적화 구현." 정보처리학회논문지. 컴퓨터 및 통신시스템 11.6 (2022): 167-174.

[3] 김현준, 엄시우, and 서화정. "CUDA GPGPU 상에서 경량 블록 암호 PIPO 의 최적 구현." 정보보호학회논문지 32.6 (2022): 1035-1043.

[4] S.W. Eum et al., “Optimized Implementation of Block Cipher PIPO in Parallel-Way on 64-bit ARM Processors,” KIPS Transactions on Computer and Communication Systems, vol. 10, no. 8,pp. 223–230, Aug.