Ascon128과 AES-GCM AEAD 암호 알고리즘의 Arduino-Uno에서의 성능 비교

윤성우[1], 이석준[2]
[1]가천대학교 컴퓨터공학과 학부생
[2]가천대학교 컴퓨터공학부(스마트보안전공) 교수
borok2311@gachon.ac.kr, junny@gachon.ac.kr

Performance Comparison of Ascon-128 and AES-GCM AEAD Cryptographic Algorithm in Arduino-Uno

Sung-Woo Yun[1], Sokjoon Lee[2]
[1]Dept. of Computer Engineering, Gachon University
[2]Dept. of Smart Security, Gachon University

요 약

2018년 NIST에서 AEAD 형태의 암호를 가진 경량 암호 표준화 공모를 진행하였다. 이후 2라운드를 통해 최종적으로 10개의 경량 암호가 남게 되었고, 이후 2023년 2월에 ASCON 암호가 NIST 경량 암호 표준으로 지정되었다. 경량 암호의 표준이 된 만큼, 기존에 사용되던 AEAD 암호와는 속도나 메모리 사용량 등, 저사양 기기에 특화된 차이점이 존재할 것이다. 본 논문에서는 저사양 기기의 환경에서, 기존 AEAD 암호에 해당하는 AES-GCM과 이번 표준으로 지정된 ASCON 암호를 Arduino-Uno에서 직접 실행함으로써 성능 면에서 어떤 차이점이 있는지 보인다.

1. 서론

2018년 8월, NIST에서 경량 암호 표준화 후보 알고리즘을 공모를 시작했으며, 2번의 라운드와 파이널 리스트를 거쳐서 2023년 2월 ASCON 암호를 최종 표준 알고리즘으로 선정하였다. 이번 공모에서는 이전과 다르게 AEAD(Authenticated Encryption with Associated Data) 방식을 모집하였다[1].

AEAD 암호는 암복호화 기능과 무결성을 인증할 수 있는 태그(tag)를 같이 제공하며, 평문 메시지 이외 논스(nonce, number used once)값과 연관 데이터(associated data)를 입력으로 받는 암호화 방식으로, 기밀성, 무결성 및 인증을 동시에 제공할 수 있는 암호화 방식이다.

현재 NIST에서 승인한, AEAD 암호화 방식을 이용한 가장 효율적인 암호로는 AES-GCM이 있는데[2], 여기서 GCM(Galois/Counter Mode)은 CTR 모드(Counter Mode)를 기반으로 데이터 무결성과 인증을 보장하는 AES 암호의 운영모드 중 하나이며, 현재 TLS/SSL 프로토콜에서도 사용하고 있다. AES-GCM 암호와 ASCON128 암호의 경우 동일한 AEAD 방식을 이용하여 만들어졌다는 공통점이 있지만, ASCON128 암호의 경우 저사양 기기를 대상으로 만들어졌다는 차이점을 가지고 있다.

본 논문에서는 저사양 기기를 대상으로 만든 ASCON128 암호와 기존에 널리 사용되던 AES-GCM 암호에 대하여, 저사양 기기 상에서 실험하여 성능을 비교한다. 이를 위해 Arduino-Uno 환경에서 특정 길이의 평문을 암·복호화하는 시간과 이때 사용되는 메모리 용량을 분석함으로써, 저사양 기기에서 ASCON128 암호가 AES-GCM 암호와 비교하여 어떤 성능을 보이는지 직접 확인한다.

2. 실험 환경 및 과정

<표 1> 실험에 사용한 Arduino Uno 사양

구분		설명
동작 주파수		16MHz
보드 동작 전압		DC 5V
CPU	IC	ATmega328P
	Clock Speed	16 ~ 20MHz
메모리	Flash Memory	32KB
	SRAM	2KB
	EEPROM	1KB

본 논문의 실험에서는 ATmega328P 마이크로컨트롤러가 탑재된 가장 대표적인 저전력·저사양 기기 중 하나인 Arduino Uno를 사용하였으며, 이 기기는

<표 1>과 같이 32KB 플래시 메모리, 2KB SRAM 등으로 구성되어 있다.

실험에 과정에서 Arduino Cryptography Library [3]에 수록되어 있는 AES와 Ascon 예제 코드를 활용하였다. 테스트 케이스의 키, 연관데이터, 태그, 논스값은 모두 16비트로 설정하였으며, 평문의 길이는 각각 32, 64, 112, 160, 240비트로 설정하여 평문의 길이에 따른 변화를 확인하고자 했다.

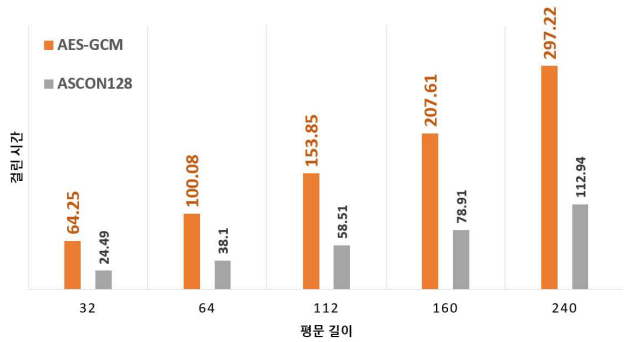
실험은 키 설정, 암호화, 연관 데이터 추가, 복호화, 태그값 비교로 총 5단계로 나누어 진행하였으며, 각각의 단계를 5000회 반복하여 초 단위로 시간을 측정함으로써 총 소요 시간을 계산하였다.

3. 실험 결과

<표 2> AES-GCM과 ASCON128 메모리 사용량 비교

	저장 공간 사용량	동적 메모리 사용량
AES-GCM	13,972 Byte (43%)	1509 Bytes (73%)
ASCON128	13,264 Byte (41%)	1500 Bytes (73%)

Arduino Uno에서 AES-GCM과 ASCON128을 구현·실행하는 과정에서 사용하는 메모리 사용량에 대한 비교를 <표 2>에서 보인다. 총 저장 공간(32,256 Bytes) 관점에서, ASCON128은 AES-GCM에 비해 약 2%(708 Bytes) 적은 저장 공간을 차지하였으며, 총 동적 메모리 2048 Bytes 중에서는 ASCON128이 AES-GCM보다 9 Byte 적은 메모리를 차지하긴 했지만 두 알고리즘 모두 전체 SRAM 크기 대비 73%에 해당하는 동적 메모리 사용량을 보이는 등 차이가 크지 않았다.



(그림 1) AES-GCM과 ASCON128 전체 연산 시간 (5000회 반복) 비교 (단위: 초)

반면, Arduino Uno에서 두 알고리즘에 대한 실행 속도는 (그림 1) 및 <표 3>에서 볼 수 있듯, ASCON128을 사용한 경우에 대해 AES-GCM과 비교하여 각각 전체 연산(키 설정, 암호화, 연관 데이터 추가, 복호화, 태그값 비교 등 5단계) 속도는 약 2.62배, 이 중 암호화 속도는 약 2.65배 빠르다는 것을 확인할 수 있었다.

<표 3> AES-GCM과 ASCON128 전체 연산 및 암호화 1회당 평균 시간 비교 (단위: ms)

평문 길이	AES-GCM		ASCON128		성능비	
	전체	암호화	전체	암호화	전체	암호화
32	12.85	3.61	4.90	1.36	2.624	2.650
64	20.02	7.20	7.62	2.72	2.627	2.647
112	30.77	12.59	11.70	4.76	2.629	2.646
160	41.52	17.98	15.78	6.79	2.631	2.646
240	59.44	26.96	22.59	10.19	2.632	2.645

4. 결론

본 논문에서는 저사양 환경인 Arduino Uno 환경에서, AEAD 암호화 방식을 따르는 두 암호 AES-GCM 와 ASCON128 암호를 비교해보았다. 실험 결과, 저장 공간면에서는 ASCON128의 근소한 우위, 그리고 동적 메모리 사용량은 거의 차이가 없었다. 실행 시간 면에서 ASCON128 암호는 AES - GCM 암호와 비교하여 전체 연산 기준 약 2.62배, 암호 연산 기준 2.65배 가량 빠른 성능을 확인하였으며, 이는 저사양 기기에서 ASCON128의 암호화 성능이 매우 뛰어남을 의미한다.

다음 연구에서는, 32/64비트 CPU를 탑재한 고성능 기기에서 어떤 성능을 보이는지, 또한 LEA를 비롯한, 기존 경량암호와 비교하여 성능면에서 어떤 알고리즘이 우위를 보이는지 등을 연구고자 한다.

5. Acknowledgement

이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. NRF-2022R1F1A1073211).

참고문헌

[1] NIST, "Submission Requirements and Evaluation Criteria for the Lightweight Cryptography Standardization Process", 2018.08

[2] Chad Boutin, “NIST Selects ‘Lightweight Cryptography’ Algorithms to Protect Small Devices”, NIST, https://www.nist.gov/news-events/news/2023/02/nist-selects-lightweight-cryptography-algorithms-protect-small-devices, 2023.02

[3] Arduino Cryptography Library, https://github.com/rweather/arduinolibs