라즈베리파이에서 MIRACL을 활용한 암호함수 연산시간 분석

류종석, 권덕규, 손승환, 이준영, 박영호

경북대학교

jongseok@knu.ac.kr, kdk145@knu.ac.kr, sonshawn@knu.ac.kr, harry250@knu.ac.kr, parkyh@knu.ac.kr

Analysis of Cryptographic Function Operation Time using MIRACL in Raspberry PI

Ryu Jong Seok, Kwon Deok Kyu, Son Seung Hwan, Lee Joon Young, Park Young Ho*

Kyungpook National Univ.

요 약

모바일 디바이스의 보급 확산으로 IoT(Internet of Things) 환경은 스마트 홈, 스마트 시티 및 헬스 케어 등 다양한 분야에 사용되고 있으며 제한된 환경에서 사용자에게 안전한 서비스를 제공하기 위한 효율적인 보안 프로토콜 연구가 활발히 진행되고 있다. 본 논문에서는 프로토콜 성능 분석을 위해 MIRACL(Multiprecision Integer and Rational Arithmetic Cryptography Library)을 활용하여 Sutrala 등과 Das 등이 제안한 프로토콜에서 사용된 암호함수 연산시간을 분석한다.

Ⅰ. 서 론

최근 모바일 디바이스의 보급이 확산됨에 따라 IoT 환경은 스마트 홈[1], 스마트 시티[2] 및 헬스 케어[3] 등 다양한 분야에 활용되고 있다. 그러나 이러한 서비스들은 공개 채널을 통하여 제공되기 때문에 다양한 보안 공격에 노출될 수 있다. 따라서 임베디드, 모바일 및 차량 등 제한된 환경에 안전하고 효율적인 프로토콜 설계가 필요하며 프로토콜 수행 간 사용된 암호함수들의 연산시간을 측정할 수 있는 라이브러리가 요구된다. 이러한 프로토콜 성능 분석을 위해 PBC(Pairing-Based Cryptography)와 MIRACL[4] 라이브러리와 같은 소프트웨어 개발 도구들이 사용되고 있다.

본 논문에서는 라즈베리파이 환경에서 MIRACL을 이용하여 Sutrala 등[5]과 Das 등[6]이 제안한 프로토콜에 사용된 암호함수와 연산시간을 비교하고 분석한다.

Ⅱ. MIRACL을 활용한 암호함수 분석

본 논문에서는 다양한 암호함수 구현 및 제한된 환경에서 보안을 구축할 수 있는 MIRACL에 대해 살펴보고 Sutrala 등과 Das 등이 제안한 보안 프로토콜에서 사용된 암호함수와 연산시간을 분석한다. 또한 사용자의 모바일 장치와 같은 제한된 환경에 대한 연산시간 측정을 위해 라즈베리파이 환경에서 MIRACL을 활용한다. 본 논문에서 사용된 라즈베리파이의 사양은 Model: Raspberry PI 3B, OS: Ubuntu 20.04.2 LTS 64-bit, Memory: 1GB이다.

1. MIRACL

MIRACL은 C/C++ 언어를 기반으로 해시함수 및 ECC(Elliptic Curve Cryptography) 등 다양한 암호함수 구현을 지원하며 임베디드, 모바일 및 차량 등 제한된 환경에 보안 체계를 구축할 수 있도록 최적화된 소프트웨어 개발 도구이다. 본 논문에서는 MIRACL을 활용하여 Sutrala 등과 Das 등이 논문에서 측정한 암호함수를 살펴보고 성능을 분석한다. 라즈베리파이 환경에서 MIRACL을 구현한 화면은 그림 1과 같고 해시함수 연산시간을 측정한 예시는 그림 2와 같다.

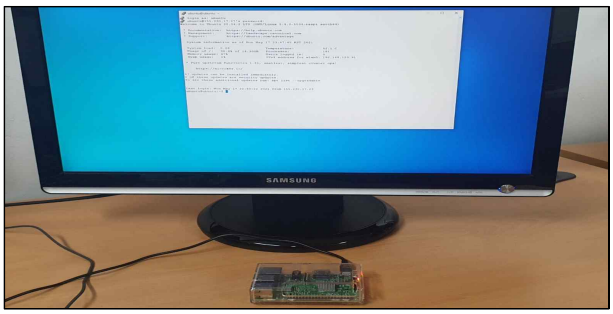

그림 1. 라즈베리파이 환경에서 MIRACL 구현.

```
ubuntu@ubuntu: ~/miracl2
ubuntu@ubuntu:~/miracl2$ ./average_hash.out
Input byte : 32

max time : 0.036 ms
min time : 0.018 ms
average time : 0.019 ms
```

그림 2. 해시함수 연산시간 측정 예시.

2. Sutrala 등과 Das 등이 사용한 암호함수 연산시간 분석

2.1 Sutrala 등이 측정한 암호함수 연산시간

Sutrala 등의 환경은 라즈베리파이 Model: Raspberry PI 3B+, OS: Ubuntu 20.04 LTS 64-bit, Memory: 1GB이다. Sutrala 등이 사용한 암호함수를 100회 수행하여 측정한 평균 연산시간은 표 1과 같다.

표 1. Sutrala 등이 측정한 암호함수 평균 연산시간 결과

표기	암호함수	평균 연산시간(ms)
T_{ecm}	Elliptic curve point multiplication	2.288
T_{eca}	Elliptic curve point addition	0.016
T_h	Hashing algorithm	0.309
T_e	Modular exponentiation	0.315
T_{senc}	Symmetric encryption	0.018
T_{sdnc}	Symmetric decryption	0.014

2.1.1 라즈베리파이를 활용하여 측정한 연산시간

본 논문의 라즈베리파이 환경에서 Sutrala 등이 사용한 암호함수를 100회 측정한 평균 연산시간은 표 2와 같다.

표 2. 본 논문에서 측정한 암호함수 평균 연산시간 결과

표기	암호함수	평균 연산시간(ms)
T_{ecm}	Elliptic curve point multiplication	2.579
T_{eca}	Elliptic curve point addition	0.019
T_h	Hashing algorithm	0.352
T_e	Modular exponentiation	0.315
T_{senc}	Symmetric encryption	0.021
T_{sdnc}	Symmetric decryption	0.017

2.1.2 분석 결과

표 1과 표 2를 통해 Sutrala 등과 본 논문 환경에서 타원곡선 곱셈, 타원곡선 덧셈, SHA-256 해시함수, 모듈러 지수 연산, 대칭키 암호화 및 복호화에 대한 평균 연산시간 측정 결과를 확인할 수 있다. Sutrala 등과 본 논문의 라즈베리파이 환경에서 측정한 평균 연산시간을 비교한 결과 각 암호함수별로 측정치가 유사함을 도출할 수 있다.

2.2 Das 등이 측정한 암호함수 연산시간

Das 등의 환경은 Galaxy S5, OS: Google Android 4.4.2 Memory 2GB이다. Das 등이 사용한 암호함수를 10,000회 수행하여 측정한 평균 연산시간은 표 3과 같다.

표 3. Das 등이 측정한 암호함수 평균 연산시간 결과

표기	암호함수	평균 연산시간(ms)
T_{ecm}	Elliptic curve point multiplication	13.405
T_{eca}	Elliptic curve point addition	0.081
T_h	Hashing algorithm	0.056
T_{me}	Modular exponentiation	2.249

2.2.1 라즈베리파이를 활용하여 측정한 연산시간

본 논문의 라즈베리파이 환경에서 Das 등이 사용한 암호함수를 10,000회 측정한 평균 연산시간은 표 2와 같다.

표 4. 본 논문에서 측정한 암호함수 평균 연산시간 결과

표기	암호함수	평균 연산시간(ms)
T_{ecm}	Elliptic curve point multiplication	14.951
T_{eca}	Elliptic curve point addition	0.092
T_h	Hashing algorithm	0.064
T_{me}	Modular exponentiation	2.482

2.2.2 분석 결과

표 3과 표 4를 통해 Das 등과 본 논문 환경에서 타원곡선 곱셈, 타원곡선 덧셈, SHA-1 해시함수 및 모듈러 지수 연산에 대한 평균 연산시간 측정 결과를 확인할 수 있다. 측정치를 비교한 결과 Das 등의 환경과 본 논문의 환경에서 측정한 암호함수들의 평균 연산시간의 차이는 0.87~0.9배로 일정하다. 따라서 MIRACL은 모바일 및 라즈베리파이 등 환경이 제한된 디바이스에서 암호함수 연산시간 측정에 활용할 수 있다.

Ⅲ. 결론

최근 모바일 디바이스의 보급 확산에 따라 IoT 환경은 스마트 홈, 스마트 시티 및 헬스 케어 등 다양한 분야에 활용되고 있지만 이러한 서비스는 공개 채널을 통해 사용자에게 제공되므로 다양한 공격에 노출되기 쉽다. 따라서 제한된 환경에서 사용자에게 안전한 서비스 제공을 위한 효율적인 보안 프로토콜 설계가 필요하며 프로토콜 수행 간 사용된 암호함수들의 연산시간을 측정할 수 있는 라이브러리가 요구된다. 본 논문에서는 라즈베리파이 환경에서 MIRACL을 활용하여 Sutrala 등과 Das 등이 제안한 프로토콜에 사용된 암호함수에 대한 평균 연산시간을 비교하였으며 유사한 연산시간을 도출할 수 있었다. 따라서 MIRACL은 암호함수 구현 및 연산시간을 측정할 수 있는 신뢰성 높은 소프트웨어 개발 도구이며 보안 프로토콜 성능 측정에 사용할 수 있다.

참 고 문 헌

[1] Oh, J., Yu, S., Lee, J., Son, S., Kim, M., and Park, Y. “A Secure and Lightweight Authentication Protocol for IoT-Based Smart Homes,” Sensors, vol 21, no. 4, pp. 1-24, 2021.

[2] Yu, S., Lee, J., Park, K., Das A. K., and Park, Y. “IoV-SMAP: Secure and Efficient Message Authentication Protocol for IoV in Smart City Environment,” IEEE Access, vol 8, pp. 167875-167886, 2020.

[3] Park, K., Noh, S., Lee, H., Das A. K., Kim, M., Park, Y., and Wazid, M. “LAKS-NVT: Provably Secure And Lightweight Authentication and Key Agreement Scheme Without Verification Table in Medical Internet of Things,” IEEE Access, vol 8, pp. 119387-119404, 2020.

[4] MIRACL Cryptographic SDK (https://github.com/miracl/MIRACL)

[5] Sutrala A. K., Obaidat M. S., Saha, S., Das A. K., Alazab, M., and Park, Y. “Authenticated Key Agreement Scheme With User Anonymity and Untraceability for 5G-Enabled Softwarized Industrial Cyber-Physical Systems,” IEEE Transactions on Intelligent Transportation Systems, pp. 1-15, 2021.

[6] Das A. K., Wazid, M., Yannam A. R., Rodrigues J. J. P. C., and Park, Y. “Provably Secure ECC-Based Device Access Control and Key Agreement Protocol for IoT Environment,” IEEE Access, vol 7, pp. 55382-55397, 2019.