Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.32, NO.5, Oct. 2022 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2022.32.5.955

IoT 환경에서의 네트워크 보안 프로토콜 성능 분석*

강 동 희,[1†] 임 재 덕[2‡]
[1]아주대학교 (학생), [2]한국전자통신연구원 (연구원)

Network Security Protocol Performance Analysis in IoT Environment*

Dong-hee Kang,[1†] Jae-Deok Lim[2‡]
[1]Ajou University (Student),
[2]Electronics and Telecommunications Research Institute (Researcher)

요 약

사물인터넷(Internet of Things, IoT)은 다양한 기술과 결합하여 일상생활의 전반으로 빠르게 자리 잡았다. 빠른 속도로 사회 전반에 자리 잡은 데 반해 보안에 대한 고려는 상대적으로 미흡하여 사이버 공격의 주요 대상이 되고 있다. IoT 환경의 모든 기기는 인터넷에 연결되어 일상생활과 밀접하게 활용되고 있기에 사이버 공격으로 인한 피해도 심각하다. 따라서 보다 안전한 IoT 환경에서의 서비스를 위해 네트워크 보안 프로토콜을 이용한 암호화 통신이 반드시 고려되어야 한다. 대표적인 네트워크 보안 프로토콜에는 IETF에서 정의한 TLS(Transport Layer Protocol)가 있다. 본 논문은 제한된 자원 특성을 갖는 IoT 기기에서 대표적인 네트워크 보안 프로토콜인 TLS의 부하를 예측하기 위하여 IoT 기기 오픈 플랫폼 환경에서 TLS 버전 1.2와 버전 1.3에 대한 성능 측정 결과를 분석한다. 또한 버전 1.3에서 지원하는 주요 암호화 알고리즘의 성능을 분석하여 IoT 기기 사양에 따라 적합한 네트워크 보안 프로토콜 속성을 설정할 수 있는 기준을 제시하고자 한다.

ABSTRACT

The Internet of Things (IoT), combined with various technologies, is rapidly becoming an integral part of our daily life. While it is rapidly taking root in society, security considerations are relatively insufficient, making it a major target for cyber attacks. Since all devices in the IoT environment are connected to the Internet and are closely used in daily life, the damage caused by cyber attacks is also serious. Therefore, encryption communication using a network security protocol must be considered for a service in a more secure IoT environment. A representative network security protocol includes TLS (Transport Layer Protocol) defined by the IETF. This paper analyzes the performance measurement results for TLS version 1.2 and version 1.3 in an IoT device open platform environment to predict the load of TLS, a representative network security protocol, in IoT devices with limited resource characteristics. In addition, by analyzing the performance of each major cryptographic algorithm in version 1.3, we intend to present a standard for setting appropriate network security protocol properties according to IoT device specifications.

Keywords: IoT security, TLS 1.2, TLS 1.3, performance analysis, system overhead estimation

Received(08. 19. 2022), Modified(09. 27. 2022), Accepted(09. 27. 2022)

* 이 논문은 2022년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임. (No. 2022-0-00103, IoT/IIoT 디바이스 안전성 보장을 위한 취약점 보안검증 기술 개발)

† 주저자, pinkbanana4@ajou.ac.kr
‡ 교신저자, jdscol92@etri.re.kr(Corresponding author)

I. 서 론

IoT(Internet of Things)는 AI(Artificial Intelligence), ML(Machine Learning), VR(Virtual Reality), 블록체인 등 다양한 기술과 결합하여 가정, 교통, 의료 등 일상생활의 전반에 일부분으로 자리 잡았다. 한국 ICD에 의하면 IoT의 시장 규모는 꾸준히 상승세를 유지하고 있으며 2025년 국내 IoT 시장은 38조 1870억 원을 기록할 것으로 예상된다[1]. 공격자의 관점에서 IoT 시장 규모의 증가는 공격할 수 있는 접점의 증가를 나타냄으로 기기 설계부터 IoT 환경에 맞는 보안을 고려해야 한다. 국내 IoT 시장은 2015년에서 2019년까지 연평균 23.7%로 가장 빠르게 성장한 데[2] 반해 미흡한 보안으로 많은 보안 사고가 발생했다. IP 카메라를 해킹해 사용자의 영상 정보에 접근하거나 스마트 도어록을 해킹해 공격자가 자유롭게 현관문을 여는 등 IoT 보안 사고의 피해는 일상생활과 직접적으로 연관되어 있다. 따라서 IoT 보안 위협의 증가에 대한 대책으로 안전한 보안 통신의 적용이 요구된다.

대다수의 IoT 통신 프로토콜은 IoT 환경에 적합한 통신을 위해 경량화, 유연성, 확장성 등의 특징을 지니고 있지만 제한된 메모리와 성능으로 인해 보안 기능이 적용되어 있지 않거나 보안에 대한 고려가 부족하다. 예로, IoT 환경에서 보편적으로 사용되는 메시지 프로토콜인 MQTT는 표준으로 정해진 보안 사항이 없으며 No TCP/IP와 TCP/IP가 혼용된 로컬 네트워크에서는 네트워크 구간을 신뢰할 수 없다[3,4]. 또 다른 메시지 프로토콜인 CoAP는 DTLS를 통해 TLS와 같은 보안성을 제공하지만 IoT 환경에서는 무거운 보안 기법이라는 단점이 있다[3]. 따라서 안전한 보안 통신 환경을 구축하기 위해서는 IoT 기기의 제한된 특성에 맞는 경량화된 암호화 프로토콜이 필요하다. 현재 IoT 환경에서 동작 가능한 다양한 경량형 암호화 프로토콜이 개발되고 있지만 신규 프로토콜은 기존의 검증된 기술과 달리 잠재적인 취약점을 포함할 수 있다[5]. 그러므로 IoT 통신 프로토콜과 검증된 암호화 통신 및 인증을 제공하는 TLS의 접목이 필요하며 TLS 버전 1.2보다 암호화 및 성능 문제를 해결한 TLS 버전 1.3의 사용을 고려해야 한다.

본 논문에서는 IoT 기기 사양에 적합한 TLS 버전 및 암호화 알고리즘을 선택할 수 있도록 TLS 버전 1.2와 버전 1.3의 성능 및 암호화 알고리즘의 성능을 측정 및 분석한 결과를 제시하여 네트워크 보안 프로토콜 적용에 참고할 수 있는 기준을 제시하고자 한다. 1장 서론에 이어 2장에서는 IoT 사이버 공격 동향을 조사한다. 3장에서는 TLS에서 발생할 수 있는 보안 위협을 분석하고 4장에서는 TLS 버전 1.3과 버전 1.2의 특징을 분석한다. 5장에서는 4장에서 분석한 내용을 바탕으로 자원의 제한성을 지닌 IoT 환경에서 TLS 버전 1.2와 버전 1.3의 성능을 비교·분석하고 IoT 환경에서 적합한 경량화 알고리즘을 선택할 수 있도록 TLS 버전 1.3에서 지원하는 다양한 암호화 알고리즘 조합의 성능을 측정한다. 6장에서는 5장의 분석 내용을 바탕으로 결론을 기술하며 논문을 마무리한다.

II. IoT 사이버 공격 위협

스마트 전등, 인공지능 스피커, 스마트 도어록, 월패드, CCTV, IP 카메라 등 다양한 종류의 IoT 기기가 출시되며 우리 생활의 전반이 인터넷에 연결되었다. 이는 편의성을 제공함과 동시에 공격자에게 다양한 공격 범위 및 접점을 제공한다. IoT 기기는 DDoS 공격의 봇넷처럼 하나의 공격 수단으로 활용될 수 있고 월패드 해킹 사건처럼 IoT 기기 자체가 직접적인 공격 대상이 될 수도 있다. 다양한 사이버 공격에 효과적으로 대응하기 위해서는 IoT 기기 보안에 대한 높은 수준의 신뢰성 보장 방안이 필요하다.

2.1 IoT 사이버 공격 동향

IoT 사이버 공격은 IoT 시장 규모와 비례하게 증가하는 추세다. 과학기술정보통신부는 2021년 주된 사이버 위협으로 월패드 해킹을 뽑았고, 2022년 예상되는 주된 사이버 위협으로는 IoT 기기 대상 사이버 위협을 전망했다[6]. IoT 사이버 공격의 주요 사례는 Table 1과 같다[7]. 각 공격 사례는 실생활과 밀접한 서비스로 사생활 침해가 크며 네트워크를 통해 공격이 이루어진다는 공통점을 갖고 있다. 2022년 초 발생한 자이스마트홈 월패드 네트워크 취약점을 활용하면 공격자가 자이스마트홈 앱의 사용자 인증 과정을 우회하여 세대주 권한을 탈취할 수 있다. 이는 2021년 11월 월패드 해킹 사건처럼 집 내부 영상이 다크웹을 통해 판매되는 사건으로 이어질 수 있다. 이처럼 IoT 보안 위협은 일상생활과 직접적으

Table 1. Key examples of IoT cyber attacks[7]

year	examples
2015.12	Unlock the door after hacking the apartment door lock
2016.02	Dissemination of unspecified number of CCTV images
2017.02	Smart toy hacking
2017.11	1,600 IP cameras hacked
2018.06	Baby monitor hacking
2018.11	Pet camera hacking
2019.01	Digital door lock vulnerability discovery
2021.11	Apartment wallpad hacking video Dark Web spreads

로 연관된 만큼 네트워크 보안 및 사용자 인증이 중요하다.

2.2 네트워크 보안 위협

2020 Unit 42 IoT 위협 보고서에 따르면 IoT 기기 트래픽의 98%는 암호화되지 않아 공격자가 암호화되지 않은 네트워크 트래픽을 통해 개인 또는 기밀 정보를 수집하여 해당 데이터를 악용할 수 있으며 주요 IoT 위협은 IoT 기기의 취약점을 이용하여 네트워크의 다른 시스템을 공격하는 것이다[8].

국내외에서 발생한 IoT 사이버 공격은 IoT 환경에서 네트워크 보안 위협의 심각성을 보여준다. TRENDnet의 아기 모니터링 카메라 해킹 사건은 사용자 로그인 자격 증명을 암호화하지 않은 채 평문으로 전송함으로써 발생하였다[9]. 또한 자이스마트홈 월패드 해킹 사건의 경우 앱 접속 시 서버에서 사용자의 인증을 확인하지 않음으로써 발생하였고, 이는 같은 망을 사용하는 다른 세대로 피해가 확산될 수 있는 위험을 갖고 있다. 이처럼 IoT 환경은 다양한 네트워크 보안 위협을 갖고 있고 적절한 보안 조치를 하지 않을 경우 심각한 피해를 초래할 수 있다.

따라서, 안전한 보안 통신 환경을 위하여 IoT 기기의 망 분리뿐만 아니라 데이터 암호화, 사용자 인증 등의 보안도 함께 적용되어야 한다.

2.3 IoT 보안과 기존 보안과의 차이점

IoT는 PC나 모바일의 고전력, 고성능 환경에서의 보안과 달리 저전력, 저성능의 자원으로 보안 기능을 구현해야 한다. 따라서 기존 보안을 그대로 적용하는 것이 아닌 IoT 환경 및 자원에 적합한 보안을 적용할 필요가 있다.

IoT는 제한된 자원에서 계산량이 적고 경량화된 보안 알고리즘을 사용해야 하므로[4] 본 논문의 5장에서 성능 시험 및 분석을 통해 IoT 환경에서 보다 적합한 TLS 버전 및 암호화 알고리즘을 선택할 수 있는 지표를 제공하고자 한다.

III. TLS 보안 위협

네트워크 보안을 위해서는 TLS 통신이 필요하지만 MITM 공격 또는 잘못된 구현 및 운영으로 Table 2와 같은 위협이 발생할 수 있다. IoT 환경에서 주로 사용되는 TLS 버전 1.2에서도 POODLE, DROWN 등 다양한 위협이 존재한다. 대표적인 TLS 보안 위협은 다음과 같다.

3.1 TLS Renegotiation Attack

TLS Renegotiation attack은 TLS 버전 1.2 이하의 재협상 과정을 이용한 공격이다[10]. MITM 공격으로 공격자는 TLS 재협상 과정에서 cipher spec 등을 공격자가 원하는 cipher spec으로 설정할 수 있다.

3.2 FREAK Attack

FREAK(Factoring Attack on RSA-EXPORT keys) attack은 MITM 공격을 통해 클라이언트가 RSA-EXPORT key를 요청하는 것으로 변조시킨다[11,12]. RSA-EXPORT key는 미국에서 전 세계 트래픽 감시를 목표로 제작한 512bit 암호화 키를 말하며 공격자는 강제로 암

Table 2. Summary of attacks on TLS

attack	explanation
TLS Renegotiation Attack[10]	MITM attack that utilizes weak encryption vulnerabilities in the TLS renegotiation process
FREAK Attack[11,12]	Downgrade to RSA using weak key length and MITM attack by exploiting the vulnerability that occurs in the ssl3_get_key exchange function of OpenSSL s3_clnt.c
Logjam Attack[13,14]	Downgrade the TLS connection usage key to a 512-bit export encryption key by taking advantage of the temporary Diffie-Hellman key exchange process vulnerability
DROWN Attack[15]	View encrypted communications using vulnerabilities in SSLv2 or SSLv3
BEAST Attack[16]	Using vulnerabilities in SSLv3 or TLS 1.0 to neutralize block-based encryption algorithms
CRIME Attack[17,18]	Stealing cookies by recovering encrypted HTTP packets using the vulnerability of DEFLATE, the compression algorithm of HTTP
BREACH Attack[19]	Using the vulnerability of the compression algorithm DEFLATE and the fact that the CSRF token is delivered as compressed content in the HTTP response
Padding Oracle Attack[20]	Using the vulnerability of the decryption method of CBC mode, which is one of the block type encryption
POODLE Attack[21,22]	Downgrade the SSL/TLS version by using the encryption communication vulnerability that occurs in TLS 1.2 or lower versions
RC4 Attack on TLS[23,24,25]	Using the vulnerability of the RC4 algorithm itself when using the RC4 encryption algorithm in TLS 1.1 or earlier
ALPACA Attack[26]	TLS connection with other services with the same domain in possible by using a certificate vulnerability that the TLS protocol only verifies the address of the server to connect to and does not verify the target service

호화 키의 길이를 512bit로 다운시켜 brute forcing 공격으로 RSA 키를 얻어 암호화 통신 내용을 복호화할 수 있다.

3.3 POODLE Attack

POODLE(Padding Oracle On Downgraded Legacy Encryption) attack은 TLS 버전 1.2 이하 버전에 영향을 주는 공격으로 MITM 공격을 통해 TLS의 최신 버전이 아닌 SSLv3 프로토콜을 사용하도록 유도한다[21,22]. 클라이언트와 서버가 SSLv3을 통한 통신을 시작하면 공격자는 Padding Oracle 공격을 통해 암호화된 통신 메시지를 복호화하여 열람할 수 있다.

TLS 버전 1.3은 재협상을 금지하여 TLS Renegotiation attack이 불가능하고, 취약한 RSA와 Diffie-Hellman을 cipher suite에서 제거하여 FREAK attack과 Logjam attack을 방지한다. 또한, 특정 길이의 패딩이 아닌 임의 길이의 패딩을 사용하여 POODLE attack으로부터 안전하다. 이처럼 TLS 버전 1.3은 버전 1.2의 프로토콜 취약점을 상당 부분 해소하였기에 버전 1.3의 사용이 권고된다. 현재 대다수의 웹 사이트는 TLS 버전 1.3을 지원하지만 IoT 환경은 약 17%를 제외하고 TLS 버전 1.3을 지원하지 않고 있다[27,28]. TLS 버전 1.3은 버전 1.2보다 안전성이 향상되었기에 IoT 환경에서도 TLS 버전 1.3의 적용을 고려할 필요가 있다.

IV. TLS 버전별 특징

RFC 8446에 따르면 TLS 버전 1.3은 1.2 이전 버전의 암호화 알고리즘의 안전성 문제 및 속도 문제를 해결하였으며 버전 1.3과 버전 1.2의 주요 차이점은 Table 3과 같다[29,30,31].

Table 3. Major Difference between TLS 1.2 and TLS 1.3

	TLS 1.2	TLS 1.3
key exchange	Static RSA, DHE, DH	ECDHE(base), EdDSA
handshake	2-RTT	1-RTT, 0-RTT
symmetric encryption	AEAD, CBC, RC4, 3DES	AEAD
cipher suite (authentication & key exchange)	merged	separate
elliptic curve algo	ECDSA(P-256,P-384)	EdDSA(Ed25519,Ed448)

4.1 Cipher Suite

TLS 버전 1.3은 지원하는 cipher suite 목록에서 Static RSA와 Diffie-Hellman을 제거해 모든 공개키 기반 key exchange 메커니즘은 forward secrecy를 제공하도록 하였으며 RC4, CBC, 3DES 같은 legacy 알고리즘의 지원을 중단하여 버전 1.3의 모든 대칭키 암호화 알고리즘은 AEAD 알고리즘으로 보안을 향상시켰다[28,29,30]. 또한, cipher suite의 협상 방법도 기존의 인증과 키 교환 알고리즘, 암호화 알고리즘을 조합해서 사용하는 방식에서 각각 독립적으로 선택하는 방법으로 변경해 향후 추가될 암호화 알고리즘의 확장성을 개선하였다.

4.2 Handshake

Fig. 1은 TLS 버전 1.2와 버전 1.3의 TLS Handshake 전체 메시지 흐름이다[29,30,31]. TLS 버전 1.2의 Handshake는 Change Cipher Spec을 포함해 2-RTT(Round Trip Time)인 반면 1.3에서는 extension을 통해 Change Cipher Spec 같은 메시지 교환을 줄여 1-RTT로 단축하였다. 서버는 ClientHello 메시지의 extension 정보를 바탕으로 암호화 알고리즘 및 파라미터를 선택할 수 있기에 Fig. 1과 같이 ServerHello 메시지 이후의 모든 Handshake 메시지를 암호화할 수 있다.

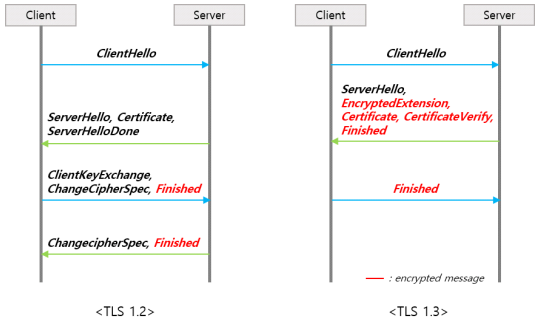

Fig. 1. Message flow for a full TLS Handshake for new session

RFC 8446 규격 분석을 통해 TLS 버전 1.3 버전의 Handshake 속도 향상을 예상할 수 있으며, 지원하는 암호화 알고리즘 목록 개선 등을 통해 보안이 향상됨을 알 수 있다. 실제 IoT 환경에서 TLS 버전 1.3의 성능 개선 정도를 5장의 성능 시험을 통해 확인하였다.

V. 성능 시험 및 분석

IoT 환경에서 TLS 버전 1.2와 버전 1.3의 성능을 분석하기 위해 4개의 core로 구성된 라즈베리파이 4로 IoT 기기를 구성하였다. 또한, IoT의 자원 제약성을 고려하여 임베디드 시스템을 대상으로 경량화한 SSL/TLS 오픈소스 라이브러리인 wolfSSL(v.5.5.0, released 2022.08)을 사용해 TLS 통신을 구현하였다.

분석에 사용한 라즈베리파이 4의 기기 사양은 다음과 같다.

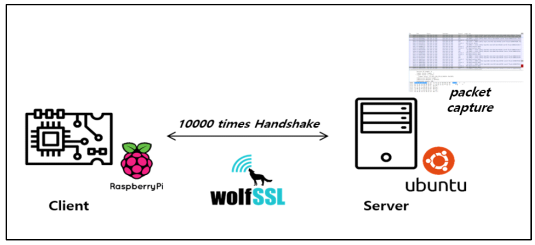

Fig. 2. TLS communication environment

Table 4. Technical specification of devices

Device	CPU	RAM	OS
Raspberry pi 4 model B Rev 1.2	quad-core Cortex-A 72, 1.5GHz	4GB	Debian GNU/Linux 11 (kernel: Linux 5.15.32-v8+)

5.1 TLS 1.2와 TLS 1.3의 성능 비교 분석

TLS 버전 1.3의 Handshake 처리 속도 향상률 및 안정성을 분석하기 위해 총 1만 번의 Handshake 과정을 실행하여 평균 세션 연결 속도와 표준편차를 측정하였다. Handshake의 Round Trip Time을 기준으로 2-RTT인 TLS 버전 1.2와 1-RTT인 TLS 버전 1.3의 Handshake, 0-RTT인 TLS 버전 1.3의 early data를 통한 resumption의 경우의 Handshake 처리 속도를 분석하였다. 이때 서버는 클라이언트의 Certificate를 요구하지 않고 Alert나 HelloRetryRequest 메시지 없이 새로운 세션과 early data에 대한 full Handshake만을 가정하였다. Handshake 평균 처리 속도를 측정하기 위하여 클라이언트에서 서버와 SSL/TLS Handshake를 시작하는 wolfSSL_connect 함수의 전후 시간 차이를 이용하였으며 Table 5는 각 경우의 평균 세션 연결 속도와 표준편차를 측정한 결과이다.

결과적으로 IoT 환경에서 TLS 버전 1.3은 버전 1.2에 비해 Handshake 연결 속도가 약 39.94% 향상되었으며 early data를 사용할 경우 약 116.20% 향상된 것을 확인할 수 있다. 표준편차 역시 TLS 버전 1.3이 1.2 보다 작은 값으로 TLS 버전 1.3을 적용할 경우 연결 속도뿐만 아니라 속도의 안전성도 향상됨을 알 수 있다. TLS 버전 1.3에서 Handshake 연결 속도가 향상된 원인은 분석 환경과 동일한 가정하에 TLS 버전 1.2는 총 10번의 Handshake 메시지를 교환하는데 반에 버전 1.3은 총 7번의 Handshake 메시지를 교환하기 때문이다. 0-RTT를 통한 resumption을 진행할 경우 서버의 Certificate 및 CertificateVerify 메시지를 전송할 필요가 없기 때문에 메시지 교환 횟수가 5번으로 단축되어 Handshake 속도가 향상된다.

Table 5. Time performance of TLS 1.2 and TLS 1.3

	TLS 1.2	TLS 1.3	TLS 1.3 (early data)
Avg (ms)	9.649	6.895	4.463
Std	1.764	1.258	0.994

Handshake 연결 속도의 향상은 보안 채널 연결을 위한 상호 인증 및 암호 키 교환 시간의 단축을 의미한다. 시간이 단축될수록 사용자에게 보안 기능의 제공으로 인한 서비스 지연을 줄여 보안에 대한 불편함을 줄일 수 있다. 또한, IoT 시장 규모가 커지고 있는 상황에서 늘어나는 IoT 기기에 대응하기 위해서는 처리 속도 향상이 필요하므로[3] 연결 속도 및 안정성이 향상된 TLS 버전 1.3의 적용을 고려해야 한다.

5.2 Cipher Suite 성능 분석

IoT의 제한된 자원으로 가장 적합한 알고리즘을 선택할 수 있도록 TLS 버전 1.3에서만 지원하는 5개의 암호화 알고리즘 조합을 대상으로 성능 분석을 진행하였다. 암호화 및 복호화의 성능 분석을 위하여 세션 연결 이후 클라이언트에서 14byte 크기의 메시지를 암호화 및 복호화하는 평균 시간과 표준편차를 측정하였다. 각 알고리즘의 시스템 부하를 알아보기 위한 성능 메트릭으로는 1만 번의 Handshake 과정에서 CPU 및 메모리의 사용량을 측정하였다. 측정 도구는 리눅스의 성능 측정 도구 패키지인 sysstat의 sar(System Activity Reporter) 명령어를 이용하였다. 해당 프로그램은 리눅스 시스템의 CPU, 메모리, 네트워크 등의 지표 정보를 수집하여 실시간으로 지표를 확인할 수 있게 한다. 이 중 분석 환경인 4 core의 전체 CPU 사용률과 메모리 사용률을 모니터링하여 평균값을 측정하였다.

Table 6은 각 알고리즘의 암호화, 복호화 시간 및 메모리와 CPU의 사용량을 측정한 결과이다. 암호화 속도는 TLS_AES_128_CCM_8_SHA256 알고리즘이 0.035ms로 가장 빨랐으며 TLS_AES_256_GCM_SHA256 알고리즘이 가장 느렸다. 반면 복호화 속도는 TLS_CHACHA20_POLY1305_SHA256과 TLS_AES_128_GCM_SHA256이 빨랐다. 알고리즘 간의 전체 CPU와 메모리 사용량의 차이가 미미하기 때문에 본 논문의 분석 환경인 4

Table 6. performance of cipher suites in Raspberry pi 4

Algo	enc time(ms) (Avg/Std)	dec time(ms) (Avg/Std)	CPU (%)	MEM (%)
TLS_AES_128_GCM_SHA256	0.040/0.019	0.345/0.109	21.04	6.73
TLS_AES_256_GCM_SHA384	0.041/0.019	0.357/0.173	21.54	6.68
TLS_CHACHA20_POLY1305_SHA256	0.037/0.013	0.344/0.082	20.83	6.69
TLS_AES_128_CCM_SHA256	0.037/0.013	0.350/0.082	20.25	6.68
TLS_AES_128_CCM_8_SHA256	0.035/0.011	0.358/0.077	20.23	6.69

core의 라즈베리파이와 유사한 환경이라면 보안 강도가 높은 암호화 알고리즘을 선택하여도 무방하다.

VI. 결 론

IoT의 시장 규모가 커지고 기기 종류가 다양해짐에 따라 보안 위협 역시 증가하고 있다. IoT의 보안 위협은 일상생활과 직접적으로 연관된 만큼 안전한 보안 통신 환경을 구축하기 위해 TLS를 비롯한 네트워크 보안 프로토콜이 필수적으로 적용되어야 한다. TLS 버전 1.3은 TLS 버전 1.2에 비해 보안과 속도 향상이 이루어졌다는 장점을 갖고 있다. TLS 버전 1.3의 장점으로 대다수의 웹 사이트는 TLS 버전 1.3을 지원하는 데에 반해 대다수의 IoT 기기는 TLS 버전 1.3을 지원하지 않는다. TLS 버전 1.3은 TLS 버전 1.2의 프로토콜 취약점을 대다수 해결하였기에 IoT 보안이 중요해진 만큼 IoT 기기 역시 TLS 버전 1.3의 사용이 필요하다. 또한 IoT는 PC나 모바일 환경과 달리 제한된 자원으로 보안 통신을 구축해야 하므로 임베디드 시스템에 맞게 경량화된 보안 프로토콜을 사용하는 것이 적합하다.

본 논문에서 IoT 환경에서 TLS 버전 1.3을 사용했을 때 Handshake 과정에서 버전 1.2에 비해 어느 정도의 안정성을 가진 속도 향상이 이뤄지는지와 TLS 버전 1.3의 주요 암호화 알고리즘 조합의 성능 측정 자료를 제공한다. 결과적으로 TLS 버전 1.3은 TLS 버전 1.2에 비해 Handshake 처리 속도가 약 40% 향상되며 표준편차는 약 1.3으로 TLS 버전 1.2보다 안정적이다. 따라서 IoT 환경에서 TLS 버전 1.3을 사용할 경우 IoT 기기 사용자의 서비스 지연 시간을 단축할 수 있으며 다수의 IoT 기기를 처리해야 하는 경우 TLS 버전 1.3의 사용으로 처리량을 증가시킬 수 있다. 4 core의 라즈베리파이 4 환경에서는 암호화 알고리즘별 전체 CPU와 메모리 사용량의 차이가 미미하기 때문에 분석 환경과 유사한 환경에서는 보안 강도가 높은 암호화 알고리즘을 선택하는 것이 좋다.

본 논문은 대중적인 오픈 IoT 기기 플랫폼 환경에서 대표적인 네트워크 보안 프로토콜인 TLS에 대한 성능을 버전별, 암호화 알고리즘 조합별로 분석함으로써 TLS를 활용한 보안 통신을 구축할 때 IoT 기기에 적합한 버전 및 암호화 알고리즘을 고려할 수 있는 하나의 지표를 제시하였다. 하지만 본 논문에서의 성능 지표 및 분석 범위가 제한적임을 감안하면 보다 다양한 조건에서 활용할 지표로 충분하지 않을 것이다.

향후 연구에서는 다양한 IoT 기기 환경을 고려하여 더욱 실질적인 보안 프로토콜 선정 기준을 제시하기 위해 성능 분석 대상을 다양한 아키텍처 구조를 가진 로엔드 IoT 플랫폼과 mbedTLS를 포함한 보안 프로토콜 솔루션을 확장하고, 네트워크 대역폭, 에너지 소비 등을 성능 분석 지표를 세분화하여 성능 분석 연구를 확장 및 고도화할 계획이다.

References

[1] IDC Korea, “Wordwide Semiannual Internet of Things Spending Guide,” Nov. 2021.

[2] KIAT, “Domestic and international IoT industry trends,” Apr. 2020.

[3] Young-hwan Jang, Jae-sung Shim and Seok-cheon Park, “Anaysis Standardized of IoT-based Low-power ·Light-weight Protocol,” Journal of the Korea Institute Of Information and Communication Engineering, 20(10), pp. 1895-1902, Oct. 2016.

[4] Se-Ra Oh and Young-Gab Kim, "Security Analysis of MQTT and CoAP protocols in the IoT Environment," CISC-W'16, pp. 297-299, Apr. 2016.
[5] Jong-mo Hwang, "Internet of Things (IoT) trends and future prospects in the financial sector," Financial Security Institute, Jul. 2016.
[6] Ministry of Science and ICT, "Cyber Threat Analysis for 21 Years and Analysis for 22 Years," Dec. 2021.
[7] boannews and securityword, "2021 Domestic and overseas security market forecast report," Feb. 2021.
[8] Unit 42, "2020 Unit 42 IoT Threat Report," Mar. 2020.
[9] iotforall, "The 5 Worst Examples of IoT Hacking and Vulnerabilities in Recorded History" https://www.iotforall.com/5-worst-iot-hacking-vulnerabilities, Last Accessed 19 Sep. 2022.
[10] Thierry Zoller, "TLS/SSLv3 renegotiation vulnerability explained," G-SEC, Apr. 2011.
[11] HITACHI, "HIRT-PUB15003: [tutorial] SSL/TLS implementations 'FREAK' issue" https://www.hitachi.com/hirt/publications/hirt-pub15003/index.html, Last Accessed 12 Aug. 2022.
[12] c0D3M, "FREAK Attack Explained" https://medium.com/@c0D3M/freak-attack-explained-3048ab9d3f30, Last Accessed 12 Aug. 2022.
[13] David Adrian et al., "Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice." CCS '15, pp 5-17, Oct. 2015.
[14] Red Hat, "Logjam: TLS Vulnerability (CVE-2015-4000)" https://access.redhat.com/ko/articles/1480443, Last Accessed 12 Aug. 2022.
[15] THE DROWN Attack, "drownattack" https://drownattack.com/, Last Accessed 12 Aug. 2022.
[16] IInvicti, "How the BEAST Attack Works" https://www.netsparker.com/blog/web-security/how-the-beast-attack-works/, Last Accessed 12 Aug. 2022.
[17] Ivan Ristic, "CRIME: Information Leakage Attack against SSL/TLS" https://blog.qualys.com/product-tech/2012/09/14/crime-information-leakage-attack-against-ssltls, Last Accessed 12 Aug. 2022.
[18] Amrita Mitra, "What is the CRIME Attack?" https://www.thesecuritybuddy.com/vulnerabilities/what-is-crime-attack/, Last Accessed 12 Aug. 2022.
[19] BREACH, "breachattack" https://breachattack.com/, Last Accessed 12 Aug. 2022.
[20] Team Sesame, "Padding Oracle attacks" https://tlseminar.github.io/padding-oracle/, Last Accessed 12 Aug. 2022.
[21] Tomasz Andrzej Nidecki, "What Is the POODLE Attack?" https://www.acunetix.com/blog/web-security-zone/what-is-poodle-attack/, Last Accessed 12 Aug. 2022.
[22] Bodo Möller, Thai Duong and Krzysztof Kotowicz, "This POODLE Bits: Exploiting The SSL 3.0 Fallback" https://www.openssl.org/~bodo/ssl-poodle.pdf, Sep. 2014.
[23] Nadhem J. AlFardan et al., "On the security of RC4 in TLS," SEC'13, pp 305-320, Aug. 2013.
[24] Min Se-ah, "What is the security quality of HTTPS?" https://www.boannews.com/media/view.asp?idx=51115, Last Accessed 17 Jul. 2022.
[25] RC4 NOMORE, "RC4 attack" https://www.rc4nomore.com/, Last Accessed 12 Aug. 2022.
[26] ALPACA Attack, "alpaca attack"

https://alpaca-attack.com/, Last Accessed 12 Aug. 2022.
[27] Daniel J. Dubois and David Choffnes, "IoTLS: Understanding TLS Usage in Consumer IoT Devices," IMC '21, pp 165-178, Nov. 2021.
[28] Ralph Holz, Jens Hiller, Johanna Amann, Abbas Razaghpanah, Thomas Jost, Narseo Vallina-Rodriguez, and Oliver Hohlfeld, "Tracking the deployment of TLS 1.3 on the Web: A story of experimentation and centralization," ACM SIGCOMM Computer Communication Review 2020, vol. 50, no. 3, pp. 3-15, Jul. 2020.
[29] E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.3," RFC 8446, Aug. 2018.
[30] Patrick Nohe, "TLS 1.3: Everything you need to know" https://www.thesslstore.com/blog/tls-1-3-everything-possibly-needed-know/, Last Accessed 13 Jul. 2022.
[31] Tech School, "A complete overview of SSL/TLS and its cryptographic system" https://dev.to/techschoolguru/a-complete-overview-of-ssl-tls-and-its-cryptographic-system-36pd, Last Accessed 13 Jul. 2022.

〈저자소개〉

강 동 희 (Dong-hee Kang) 정회원
2022년 7월~8월: 한국전자통신연구원 연구연수생
2018년 3월~현재: 아주대학교 사이버보안학과 학부과정
〈관심분야〉 IoT 보안, 정보보호 등

임 재 덕 (Jae-Deok Lim) 종신회원
1999년 2월: 경북대학교 전자공학과 졸업
2001년 2월: 경북대학교 전자공학과 석사
2013년 8월: 충남대학교 컴퓨터공학과 박사
2000년 12월~현재: 한국전자통신연구원 정보보호연구본부 책임연구원
〈관심분야〉 IoT 보안, 운영체제 보안, 네트워크 보안, 접근제어 등