https://doi.org/10.33778/kcsa.2023.23.1.045

유·무선 환경에 적용 가능한 효율적인 Wi-Fi Easy Connect 프로토콜 개선방안 연구★

유 호 제*, 김 찬 희**, 임 성 식**, 김 서 연**, 김 동 우**, 오 수 현***

요 약

최근 사물인터넷의 발전과 함께 UI가 없는 장치도 간편하게 네트워크에 연결할 수 있는 프로토콜에 관한 연구가 꾸준하게 이루어지고 있다. 이를 위해 Wi-Fi Alliance에서는 QR 코드를 사용하여 네트워크에 연결할 수 있는 Wi-Fi Easy Connect를 발표하였다. 하지만, Wi-Fi Easy Connect는 안전성을 위해 많은 연산량을 요구하고 있어 저전력·소형화된 사물인터넷 장치에 적용하기 어렵다는 문제가 있다. 또한, 확장성을 고려한 Wi-Fi Easy Connect는 유선 환경에서도 동작할 수 있도록 설계되었지만, TLS와 같이 보안 환경을 고려하지 않아 중복 암호화 등의 문제가 발생하고 있다. 따라서 본 논문에서는 Wi-Fi Easy Connect 프로토콜을 분석하고, TLS 환경에서도 효율적으로 동작할 수 있는 프로토콜을 제안한다. 제안하는 프로토콜은 기존 보안 요구사항을 만족함과 동시에 연산량이 큰 ECC scalar multiplication 연산이 약 67% 감소하는 것을 확인할 수 있었다.

A Research on Effective Wi-Fi Easy Connect Protocol Improvement Method Applicable to Wired and Wireless Environments

Ho-jei Yu*, Chan-hee Kim**, Sung-sik Im**, Seo-yeon Kim**, Dong-woo Kim**, Soo-hyun Oh***

ABSTRACT

Recently, with the development of the Internet of Things, research on protocols that can easily connect devices without a UI to the network has been steadily conducted. To this end, the Wi-Fi Alliance announced Wi-Fi Easy Connect, which can connect to a network using a QR code. However, since Wi-Fi Easy Connect requires a large amount of computation for safety, it is difficult to apply to low-power and miniaturized IoT devices. In addition, Wi-Fi Easy Connect considering scalability is designed to operate in a wired environment, but problems such as duplicate encryption occur because it does not consider a security environment like TLS. Therefore, in this paper, we analyze the Wi-Fi Easy Connect protocol and propose a protocol that can operate efficiently in the TLS environment. It was confirmed that the proposed protocol satisfies the existing security requirements and at the same time reduces about 67% of ECC scalar multiplication operations with a large amount of computation.

Key words : Wi-Fi Easy Connect, Device Provisioning Protocol, IoT, TLS

접수일(2023년 02월 22일), 수정일(2023년 03월 08일), 게재확정일(2023년 03월 23일)

★ 이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2021R1F1A1060897l2).

★ 본 논문은 유호제의 2022년도 석사 학위 논문을 발췌·정리하였음.[1]

* 호서대학교 정보보호학과 석사과정(주저자)

** 호서대학교 정보보호학과 석사과정(공동저자)

*** 호서대학교 컴퓨터공학부 교수(교신저자)

1. 서 론

인터넷과 연결할 수 있는 네트워크 기술은 유선 네트워크부터 무선 네트워크, 이동통신까지 각 분야에 맞게 발전하고 있다. 최근에는 IoT(Internet of Things) 장치의 발전과 더불어 국내 무선 인프라 시장이 매우 크게 성장하고 있으며, 'ICT R&D 기술로드맵 2025'에 따르면 2025년 국내 네트워크 장비시장 중 사업자용 장비의 무선 인프라의 규모는 약 6천억에 이를 것으로 전망하고 있다.[2]

이러한 성장 추세는 일반 사용자뿐만 아니라 산업 분야에서도 크게 성장하고 있으며[3], 다양한 환경을 고려하기 위한 무선 네트워크 프로토콜 또한 개발되고 있다.[4]

2018년 Wi-Fi Alliance에서는 UI(User Interface)가 없는 장치를 Wi-Fi에 연결하기 위한 Wi-Fi Easy Connect 표준[5]을 공개하였으며, 넓은 범위에 Wi-Fi 네트워크를 구성할 수 있는 Wi-Fi Mash 표준 등[6]을 공개하였다.

이때, Wi-Fi Easy Connect 표준은 기존 Wi-Fi Protected Setup[7] 프로토콜의 취약점을 개선함과 동시에 보안성을 높인 프로토콜로 무선 환경과 유선 환경을 고려하여 설계되었다. 하지만, 유선 환경에서 TLS(Transport Secure Layer)[8]를 사용하는 환경을 고려하지 않아 연산량이 많아지는 등의 문제가 있어 유·무선 환경을 고려한 Wi-Fi Easy Connect 프로토콜에 대한 연구가 필요하다.

따라서 본 논문에서는 Wi-Fi Easy Connect 프로토콜이 TLS를 사용하는 유선 환경에서 동작할 때 기존 보안 요구사항을 만족함과 동시에 연산량을 줄인 효율적인 인증 방법에 대해 기술한다.

본 논문의 구성으로 2장에서는 간편한 연결을 지원하는 Wi-Fi Protected Setup 프로토콜과 Wi-Fi Easy Connect 프로토콜에 대해 정리한다.

3장에서는 앞서 분석한 Wi-Fi Easy Connect 프로토콜을 기준으로 유·무선 환경을 고려한 프로토콜을 제안한다.

4장에서는 제안하는 프로토콜을 Wi-Fi Easy Connect 유선 환경에서의 프로토콜과 비교 및 검증하며, 마지막으로 5장에서는 결론을 기술한다.

2. 관련연구

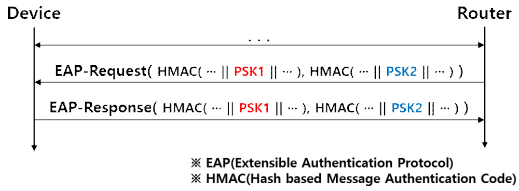
UI가 없는 장치에서도 무선 네트워크에 연결할 수 있도록 지원하는 DPP(Device Provisioning Protocol)는 Wi-Fi뿐만 아니라 ZigBee[9]와 Z-Wave[10] 등 다양한 프로토콜도 지원한다. 이때, Wi-Fi 프로토콜은 무선 네트워크에서 가장 많이 사용하는 프로토콜로 Wi-Fi Protected Setup과 Wi-Fi Easy Connect가 존재한다.

2.1 Wi-Fi Protected Setup 프로토콜

WPS(Wi-Fi Protected Setup) 프로토콜은 UI가 없는 장치에서 버튼을 누르거나, 장치에 고정된 PIN 번호를 라우터에 입력하여 네트워크에 연결하게 된다. 이때, PIN 번호는 (그림 1)과 같이 8자리를 사용하고 있으며, 앞 4자리와 뒤 3자리, 체크섬 1자리로 구성되어있다.

(그림 1) WPS PIN 구성

이러한 PIN 번호 구성은 (그림 2)와 같이 연결과정에서 사용되지만, 앞 4자리의 범위인 0부터 9,999까지 10,000개의 조합과 뒤 3자리 1,000개의 조합으로 구성되어 11,000개의 조합으로 무차별 대입 공격이 가능하다는 한계점이 존재한다.[11, 12]



(그림 2) WPS 연결과정에서 사용되는 PIN

또한, WPS는 제조사에서 0으로 구성된 취약한 난수를 사용하는 경우도 존재하여 Wi-Fi Easy Connect의 공개와 동시에 사용하지 않게 되었다.

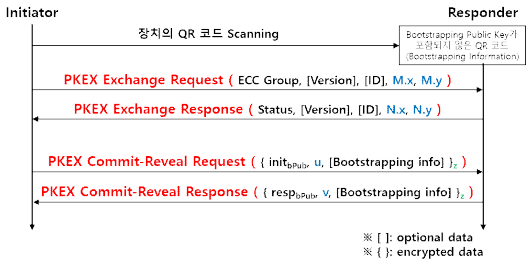
2.2 Wi-Fi Easy Connect 프로토콜

Wi-Fi Alliance에서는 WPS에서 발생한 취약점을 해결하기 위해 QR 코드를 사용하여 간편하게 네트워크에 연결하고, 강력한 보안을 제공할 수 있는 Wi-Fi Easy Connect를 공개하였다.

Wi-Fi Easy Connect의 동작 과정은 PKEX(Public Key Exchange) 과정, Bootstrapping 과정, Authentication 과정, Configuration 과정, Network Access 과정으로 구분된다. 또한, 사용하는 역할은 시작하는 장치(QR 코드를 인식하는 장치)인 Initiator와 요청에 응답하는 Responder로 구분되며, Authentication 과정 이후 네트워크에 참여하는 Enrollee와 네트워크를 구성하는 Configurator로 변경된다.

2.2.1 PKEX(Public Key Exchange) 과정

Wi-Fi Easy Connect의 PKEX 과정은 Bootstrapping 공개키를 안전하게 전달함과 동시에 ID와 패스워드를 기반으로 장치를 식별하는 과정이다. 따라서 PKEX 과정은 (그림 3)과 같이 PKEX Exchange 과정을 통해 장치를 식별하고, Commit-Reveal 과정을 통해 각자의 공개키를 주고받는다.



(그림 3) Public Key Exchange 전체 동작 과정

또한, Initiator와 Responder는 식(1, 2)와 같이 인증값을 계산하고, Bootstrapping 공개키와 Bootstrapping 정보(QR 코드 정보)를 암호화하여 전송 및 검증한다.

$$J.x, J.y = resp_{tPub} * init_{bPriv} \quad (1)$$
$$u = HMAC(J.x, init_{bPub} \| resp_{tPub} \| init_{tPub})$$

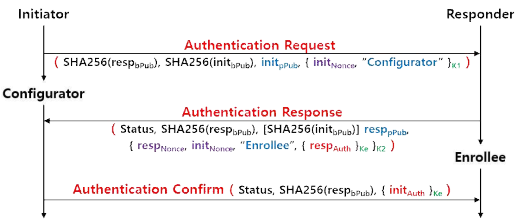
$$L.x, L.y = init_{tPub} * resp_{bPriv} \quad (2)$$
$$v = HMAC(L.x, resp_{bPub} \| init_{tPub} \| resp_{tPub})$$

2.2.2 Bootstrapping 과정

Bootstrapping 과정은 Bootstrapping 공개키를 확인하는 과정으로 Responder는 “chirp” 문자열과 자신의 Bootstrapping 공개키를 해시한 값을 Presence Announcement 패킷을 통해 Broadcast로 전송하고 있으며, Initiator는 해당 패킷을 통해 연결 대상을 확인할 수 있다.

2.2.3 Authentication 과정

Authentication 전체 과정은 (그림 4)와 같이 Initiator와 Responder의 역할인 Configurator와 Enrollee를 정하며 Protocol 키를 생성하고 Configuration 단계에서 사용할 암·복호화 키를 협상한다.



(그림 4) Authentication 전체 동작 과정

먼저 Initiator는 Protocol 키와 난수를 생성하고, 식(3)과 같이 Responder의 Bootstrapping 공개키와 자신의 Protocol 개인키로 암·복호화 키 K1을 계산한다. 이후 난수와 자신의 역할(Configurator)을 암호화하여 전송한다.

$$M.x, M.y = resp_{bPub} * init_{pPriv} \quad (3)$$
$$K_1 = HKDF(<>, \text{'first intermediate key'}, M.x)$$

Request를 받은 Responder는 식(4)와 같이 암·복호화 키 K2를 계산하고, 식(5)와 같이 이후 단계에서 사용할 암·복호화 키 Ke를 계산한다.

$$N.x, N.y = init_{pPub} * resp_{pPriv} \quad (4)$$
$$K_2 = HKDF(<>, \text{'second intermediate key'}, N.x)$$

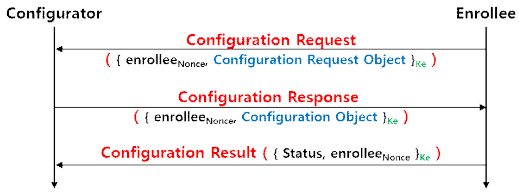
$$b_k = HKDF_{ext}(init_{nonce} \| resp_{nonce}, M.x \| N.x) \quad (5)$$
$$K_e = HKDF_{exp}(b_k, \text{'DPP Key'})$$

또한, Responder는 Authentication 단계에서 주고받은 키와 난수를 확인하기 위해 식(6)과 같이 인증값을 생성하여 Configurator에게 전송한다.

$$resp_{auth} = SHA256(init_{nonce} \| resp_{nocne} \| init_{pPub.x} \| resp_{pPub.x} \| resp_{bPub.x} \| 0) \quad (6)$$

2.2.4 Configuration 과정

Wi-Fi Easy Connect의 Configuration 과정은 Enrollee가 AP(Access Point)에 접근할 때 필요한 정보를 Configurator가 생성 및 서명하고 전달하는 과정이며 (그림 5)와 같이 3단계로 나눠진다.



(그림 5) Configuration 전체 동작 과정

먼저 Enrollee는 네트워크에서 사용할 이름과 역할 정보 등을 포함하여 JSON 형태의 Configuration Request Object를 생성 및 전송한다. 이를 받은 Configurator는 Enrollee가 생성한 Protocol 공개키로 네트워크에 접근할 수 있도록 <표 1>과 같은 데이터를 포함하여 JWS 형식[13]의 DPP Connector를 생성한다.

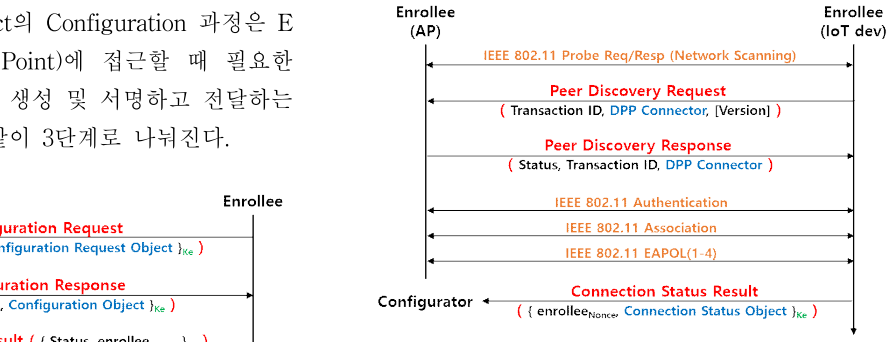
<표 1> DPP Connector의 JWS Payload 정보

JSON Key	설명
groups	네트워크에서 구분되는 그룹
netAccessKey	Responder의 Protocol 공개키
expiry	DPP Connector의 만료 시간

이후 Configurator는 JWS 형식의 DPP Connector를 포함한 Configuration Object를 Enrollee에게 전송한다. 이때, Enrollee가 AP인 경우 Configurator는 서명용 키를 생성하게 되고, AP가 아닌 경우에는 AP에서 생성한 서명용 키를 사용하게 된다. 따라서 Enrollee의 DPP Connector를 받은 AP는 Configurator의 서명을 검증할 수 있다.

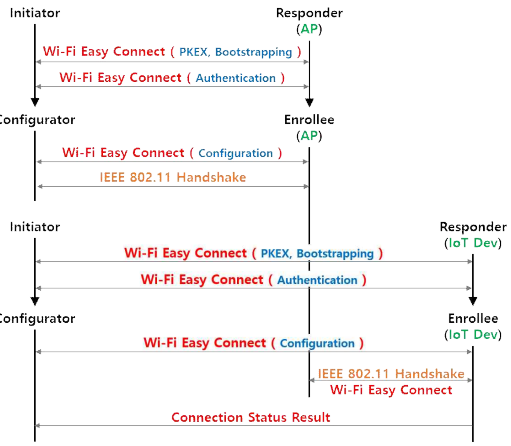
2.2.5 Network Access 과정

Wi-Fi Easy Connect의 Network Access 과정은 사전에 IoT 장치가 AP에 접근하는 과정이다. 따라서 (그림 6)과 같이 AP와 Enrollee는 자신의 DPP Connector를 주고받은 후 Configurator의 서명을 검증하고 연결을 진행한다. 또한, 해당 과정은 IEEE 802.11[14] 연결과정을 포함하고 있다.



(그림 6) Network Access 전체 동작 과정

IoT 장치가 AP에 연결이 성공적으로 마무리되면 result를 STATUS_OK로 설정하고 Configurator에게 전달하여 Wi-Fi Easy Connect를 통한 연결이 마무리된다. 결과적으로 1개의 AP와 IoT 장치가 Configurator에 의해 연결이 이루어지는 과정은 (그림 7)과 같이 동작한다.

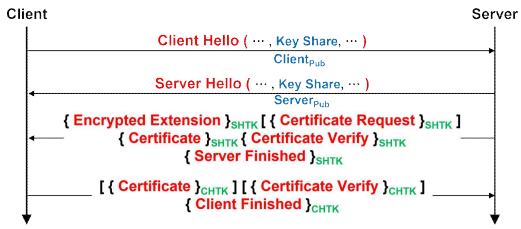


(그림 7) Wi-Fi Easy Connect를 이용한 AP 구성 및 IoT 장치 연결과정

2.3 유선 환경에서의 Wi-Fi Easy Connect

유선 환경에서는 TCP 3-way Handshake 이후부터 동작을 시작하며, TLS 환경의 경우 3-way Handshake 이후 TLS Handshake를 진행한다.

본 논문에서는 TLSv1.3을 기준으로 작성하였으며 TLSv1.3 Handshake 과정은 (그림 8)과 같이 ECC 공개키를 주고받은 후, HTK(Handshake Traffic Key)와 ATK(Application Traffic Key)를 도출하여 암·복호화키로 사용한다.



(그림 8) TLSv1.3 Handshake 과정

따라서 HTK와 ATK를 도출하기 위해 서로의 공개키를 주고받은 Server와 Client는 식(7)과 같이 Common Secret을 계산한다.

$$EarlySecret(ES) = HKDF_{ext}(0,0) \quad (7)$$
$$DerivedES(dES) = HKDF_{exp}(ES, 'derived', H(''))$$
$$CommonSecret(CS) = Server_{Pub} * Client_{Priv}$$

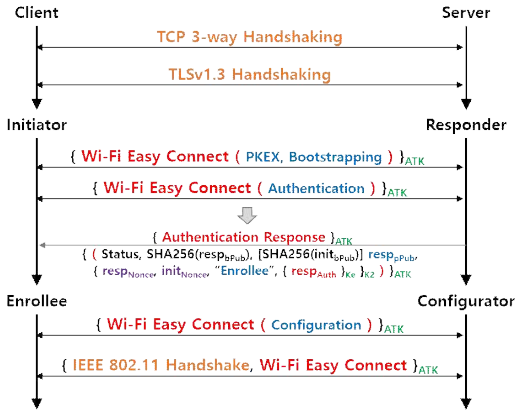
이후 Handshake 메시지를 암·복호화할 수 있는 HTK를 계산하고, Application 데이터(Wi-Fi Easy Connect 데이터)를 암·복호화할 수 있는 ATK를 식(8-10)과 같이 도출한다.

$$HandshakeSecret(HS) = HKDF_{ext}(dES, CS.x) \quad (8)$$
$$DerivedHS(dHS) = HKDF_{exp}(HS, 'derived', H(''))$$
$$MasterSecret(MS) = HKDF_{ext}(dHS, 0)$$

$$ClientATS(CATS) = HKDF_{exp}(MS, 'c\ ap\ traffic', H(ClientHello) |...| H(ServerFinished)) \quad (9)$$

$$ClientATK(CATK) = HKDF_{exp}(CATS, 'key', H('')) \quad (10)$$

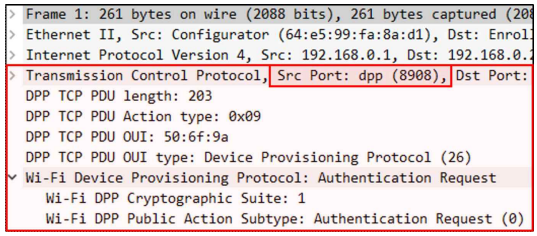
이후 유선 환경에서는 시작하는 주체가 IoT 장치이므로 Initiator에서 Enrollee로 역할이 변경되는 것을 확인할 수 있다. 또한, (그림 9)와 같이 Responder Authentication 값을 3번 암호화하여 전송하는 것을 확인할 수 있다.



(그림 9) TLS 환경에서의 Wi-Fi Easy Connect

3. 제안하는 효율적인 Wi-Fi Easy Connect 프로토콜 개선방안

Wi-Fi Easy Connect는 무선 환경뿐만 아니라 유선 환경에서도 사용할 수 있도록 설계되어 있고 (그림 10)과 같이 TCP 패킷에 캡슐화되어 전송되며 8908번 포트를 사용하고 있다.



(그림 10) 캡슐화된 Wi-Fi Easy Connect 패킷

위와 같은 유선 환경에서는 TLS와 같은 보안 메커니즘을 사용할 수 있고, 구성 환경의 보안 요구사항에 따라 TLS를 필수로 사용하는 환경이 존재한다. 이때, TLS와 Wi-Fi Easy Connect를 동시에 사용하는 경우 2중 암호화와 성능저하 등의 문제가 발생할 수 있다.

따라서 제안하는 프로토콜은 TLS에서 제공할 수 있는 보안 메커니즘과 Wi-Fi Easy Connect에서 제공하는 보안 메커니즘이 중복되는 것을 최소화하고, 저전력·소형화를 지향하는 IoT 장치에서도 활용할 수 있도록 설계하였다.

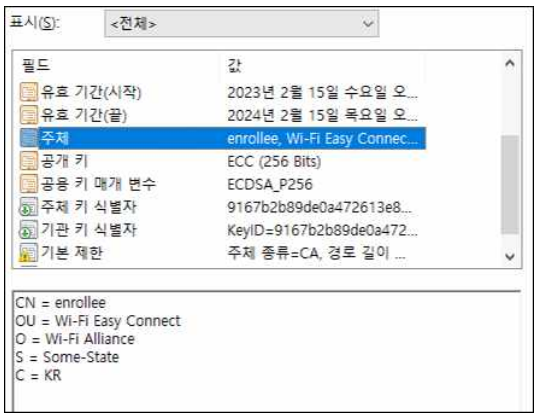
3.1 TLS Handshake 과정

TLS Handshake 과정에서 사용하는 인증서는 X.509 형식의 인증서[15]를 사용한다. 하지만, 저전력·소형화를 지향하는 IoT 장치가 사용하기에는 많은 연산량과 저장 공간을 요구하므로 최근 연구되고 있는 경량화된 인증서[16, 17] 등으로 X.509 인증서를 대체할 수 있다.

3.2 PKEX, Bootstrapping 과정

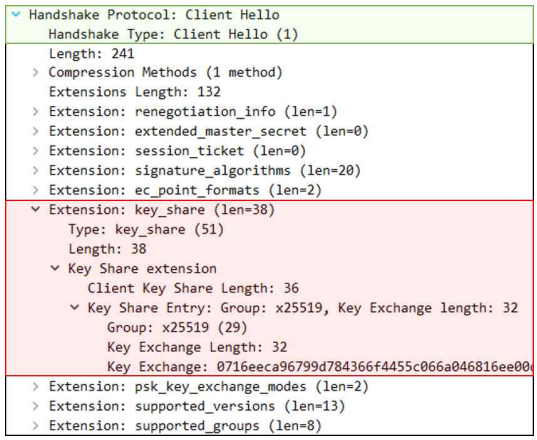
Wi-Fi Easy Connect의 PKEX 과정은 ID와 비밀번호를 통해 정상 사용자(장치)를 식별하고 Bootstrapping 공개키를 전달하는 과정이다.

해당 과정은 TLSv1.3 Handshake 동작 과정에서 (그림 11)과 같은 Self-signed 인증서와 Certificate, Certificate Verify를 통해 Responder(Configurator)와 Initiator(Enrollee)를 식별할 수 있다..



(그림 11) Enrollee의 Self-signed 인증서 예시

또한, (그림 12)와 같이 TLSv1.3의 Client Hello와 Server Hello 단계에서 key_share 필드를 활용하여 Bootstrapping 공개키를 전달할 수 있어 PKEX 과정을 대체할 수 있다.

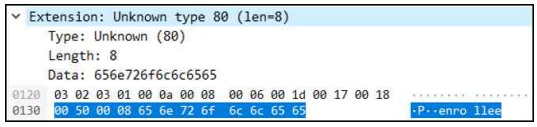


(그림 12) TLSv1.3 Client Hello 예시

3.3 Authentication 과정

제안하는 Authentication 과정은 TLS Handshake 과정에서 도출된 ATK를 사용하여 이후 과정의 데이터를 암·복호화할 수 있으므로 Authentication 과정에서 수행하는 Ke를 계산할 필요가 없다.

또한, (그림 11, 13)과 같이 TLS에서 사용하는 인증서나 TLSv1.3 Handshake 확장 필드에 역할을 추가하여 전달할 수 있다. 따라서 Wi-Fi Easy Connect에 제안하는 TLS를 적용하면 Authentication 과정을 생략할 수 있다.

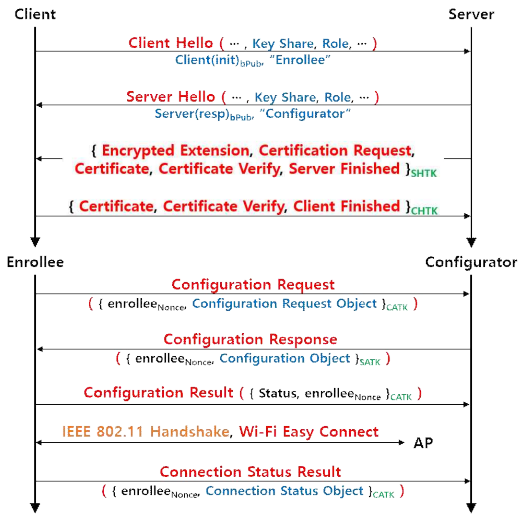


(그림 13) TLS 확장 필드를 활용한 예시

3.4 Configuration 과정

제안하는 Configuration 과정은 Authentication 과정이 생략되므로 TLSv1.3 Handshake 과정에서 도출한 ATK를 사용하여 암·복호화한다.

따라서 제안하는 프로토콜은 (그림 14)와 같이 TLSv1.3 Handshake 과정 이후 PKEX 과정과 Bootstrapping 과정, Authentication 과정이 생략되고, Configuration 과정에서 ATK를 사용하여 암·복호화를 수행하는 것을 확인할 수 있다.



(그림 14) 제안하는 프로토콜을 적용한 Wi-Fi Easy Connect 전체 연결과정

4. 제안하는 프로토콜의 안전성 및 효율성 분석

본 논문에서는 제안하는 프로토콜을 기존 Wi-Fi Easy Connect 프로토콜과 비교하기 위해 Specification[5]에 명시된 보안 위협을 바탕으로 분석하였으며, 각 과정에서 수행하는 연산 과정을 비교한다.

4.1 프로토콜의 안전성 분석

Wi-Fi Alliance는 <표 2>와 같이 각 연결 단계에서의 보안 위협과 보안 요구사항을 명시하였다. 이때, Wi-Fi Easy Connect에서 사용하는 모든 키는 ECC를 기반으로 두고 있으며 최소 NIST(National Institute of Standards and Technology)의 Prime256v1을 사용하도록 권장하고 있다.

또한, Authentication 과정과 Configuration 과정에서 재전송 공격을 방지하기 위해서 Nonce를 사용하고 있다.

본 논문에서 제안하는 프로토콜은 TLSv1.3을 기준으로 하고 있으며 TLSv1.3 Handshake 과정에서 전송하는 임시 ECC 키 쌍을 Bootstrapping 공개키로 사용하고 있다. 따라서 Wi-Fi Easy Connect에서 사용하는 Prime256v1을 사용할 수 있으며 동일한 암호 강도를 제공할 수 있다.

Authentication 과정과 Configuration 과정의 Nonce도 TLSv1.3 Handshake 과정에서도 Client Hello와 Server Hello에서 각자의 Nonce를 생성하여 전송하므로 재전송 공격을 최소화할 수 있다.

결과적으로 Wi-Fi Easy Connect가 각 과정에서 사용한 변수는 <표 3>과 같이 변경하여도 기존 보안 요구사항을 만족시킬 수 있으며, ECC 보안 강도를 쉽게 변경할 수 있을 것으로 본다.

<표 2> Wi-Fi Easy Connect의 보안 위협 및 보안 요구사항

단계	보안 위협 시나리오	보안 요구사항
PKEX, Bootstrapping	QR 코드를 변조하여 공격자의 Bootstrapping 공개키를 사용하는 경우	PKEX 과정을 수행하여 상호 식별 및 인증을 제공하고, PKEX 과정에서 사용하는 Bootstrapping Key는 주기적으로 변경
	QR 코드 이외의 방법으로 Bootstrapping 공개키를 전송 시 공격자가 먼저 공개키를 전송하는 경우	
	Bootstrapping 공개키에 대응되는 개인키가 유출·분석되는 경우	타원 곡선 암호(ECC)를 기반으로 높은 암호 강도를 유지
	PKEX 비밀번호가 유출·분석되는 경우	PKEX 비밀번호를 주기적으로 변경
Authentication	Initiator의 Bootstrapping 공개키를 얻지 못하여 악의적인 Initiator와 연결을 시도하는 경우	PKEX 과정을 수행하여 상호 식별 및 인증을 제공
	Responder의 Bootstrapping 공개키 Hash 값을 얻어 DoS(Denial of Service) 공격을 수행하는 경우	Initiator/Responder의 Nonce를 비교
Configuration	Configurator의 서명용 공개키에 대응되는 개인키가 유출·분석되는 경우	타원 곡선 암호(ECC)를 기반으로 높은 암호 강도를 유지
Network Access	DPP Connector를 변조하는 경우	
	DPP Connector의 Network Access Key에 대응되는 개인키가 유출·분석되는 경우	

<표 3> 제안하는 프로토콜에서 변경한 변수

과정	Wi-Fi Easy Connect	제안하는 프로토콜
PKEX, Bootstrapping	PKEX ID	TLS Extension (패스워드는 해시)
	PKEX 비밀번호	
	Bootstrapping Key	TLS key share
	PKEX Temp Key	삭제
Authentication	Protocol Key	Configuration 단계에서 생성
	Nonce	TLS Extension
	Network Role	
	Authentication Tag	삭제
	암·복호화 키 K1, K2	삭제
	암·복호화 키 Ke	SATK, CATK
Configuration	Network Access Key	유지
	Nonce	
	Configuration Obj	

4.2 기존 프로토콜과의 성능 비교

Wi-Fi Easy Connect에서 수행하는 연산은 Initiator와 Responder, Enrollee의 Nonce를 생성하고 ECC 개인키를 생성할 때 사용하는 난수 생성 연산과 해시 연산, ECC scalar multiplication 연산, ECC Point Addition 연산, 암·복호화 연산으로 총 5개의 연산 과정이 존재한다.

본 논문에서는 IoT 장치인 Enrollee가 수행하는 5개의 연산 과정 중 <표 4, 5>와 같이 연산량이 가장 높은 ECC scalar multiplication 연산은 TCP 환경의 Wi-Fi Easy Connect와 성능을 비교하고, 암·복호화 연산은 TLS 환경의 Wi-Fi Easy Connect 프로토콜과 성능을 비교한다.

<표 5> 제안하는 프로토콜의 암·복호화 연산 횟수 비교

과정	Wi-Fi Easy Connect (TLS)	제안하는 프로토콜	증감
TLS Handshake	8회	8회	**0회**
PKEX	2회 + 4회	0회	**-6회**
Bootstrapping	0회 + 1회	0회	**-1회**
Authentication	4회 + 3회	0회	**-7회**
Configuration	3회 + 3회	3회	**-3회**
합계	**28회**	**11회**	**-17회**

먼저 ECC scalar multiplication 연산의 경우 PKEX 과정과 Bootstrapping 과정, Authentication 과정을 수행하지 않으므로 11번의 연산이 줄어든 것을 확인할 수 있다. 또한, TLS Handshake 과정을 추가하여 3회의 연산이 추가되었지만, 결과적으로 8회의 연산이 감소하는 것을 확인할 수 있다.

암·복호화 연산의 경우 TLS 환경에서 전송되는 모든 패킷을 다시 암·복호화하므로 패킷의 개수만큼 연산 횟수가 증가하는 것을 확인할 수 있다. 이를 해결하기 위해 제안하는 프로토콜을 적용하면 443번 포트에서 TLS handshake를 진행한 후 8908번 포트를 통해 Wi-Fi Easy Connect를 수행하며 불필요하게 2~3중으로 암·복호화되는 것을 최소화할 수 있다.

결과적으로 제안하는 프로토콜을 적용하면 기존 무선 환경뿐만 아니라 유선 환경에서도 간편하게 연결할 수 있으며, ECC scalar multiplication 연산의 경우 약 66.67% 감소하고, 암·복호화 연산의 경우 약 60.7% 감소함과 동시에 보안 요구사항을 충족할 수 있다는 장점이 있다.

<표 4> 제안하는 프로토콜의 ECC scalar multiplication 연산 횟수 비교

단계	Wi-Fi Easy Connect (TCP)		제안하는 프로토콜		
	연산 값	횟수	연산 값	횟수	증감
TLS handshake	-	-	Bootstrapping 개인키, CS, ECDSA 서명 검증	3회	**+3회**
PKEX	Bootstrapping 개인키, Temp 개인키, Qi, Qr, K, J, L	7회	-	0회	**-7회**
Bootstrapping	-	-	-	-	**-**
Authentication	Protocol 개인키, M, N, L	4회	-	0회	**-4회**
Configuration	ECDSA(Elliptic Curve Digital Signature Algorithm) 서명 검증	1회	Protocol 개인키	1회	**0회**
합계		**12회**		**4회**	**-8회**

5. 결 론

2018년에 공개된 Wi-Fi Alliance의 Wi-Fi Easy Connect는 안전성과 더불어 사용자 편의성을 제공하기 위해 개발되었다. 하지만, 저전력·소형화를 지향하는 IoT 장치에는 많은 연산량을 요구하고 있어 적용하기 어렵다는 한계점이 있고, 무선 환경뿐만 아니라 유선 환경에서 동작할 때 발생하는 문제를 고려하지 않았다. 따라서 본 논문에서는 Wi-Fi Easy Connect 프로토콜을 분석하고, TLS를 사용하는 유선 환경에서도 보안 요구사항을 만족하면서 효율성을 높인 프로토콜을 제안하였다.

제안하는 프로토콜은 Wi-Fi Easy Connect에서 사용하는 암·복호화 키 Ke를 TLSv1.3 Handshake 과정에서 도출되는 ATK로 사용하였고, 중복되는 보안 메커니즘을 최소화하여 PKEX 과정과 Bootstrapping 과정, Authentication 과정을 생략하였다. 그 결과 연산량이 큰 ECC scalar multiplication 연산의 경우 약 67% 감소하는 것을 확인할 수 있어 제안하는 프로토콜이 Wi-Fi Easy Connect와 같은 Device Provisioning Protocol 발전에 이바지할 것으로 기대한다.

참고문헌

[1] 유호제, “IoT 환경에서 사용성과 편의성을 강화한 Device Provisioning Protocol 연구,” 호서대학교 대학원 석사학위논문, 2022.

[2] 최성호, “ITC R&D 기술로드맵 2025 - 통신·전파 분야,” 정보통신기획평가원(IITP), 2020.

[3] “글로벌 시장동향보고서 - 산업용 IoT 시장,” 연구개발특구진흥재단, 2021.

[4] “Matter Specification Version 1.0,” Connectivity Standards Alliance, 2022.

[5] “Wi-Fi Easy Connect Specification Version 2.0,” Wi-Fi Alliance, 2020.

[6] “Wi-Fi Easy Mash Specification Version 5.0,” Wi-Fi Alliance, 2022.

[7] “Wi-Fi Protected Setup Specification Version 2.0.8,” Wi-Fi Alliance, 2020.

[8] E.Rescorla, “The Transport Layer Security(TLS) Protocol Version 1.3.” RFC 8446, Internet Engineering Task Force(IETF), 2018.

[9] “ZigBee Specification,” ZigBee Alliance, 2015.

[10] “Z-Wave Command Class Control Specification,” Z-Wave Alliance, 2021.

[11] Stefan Viehböck, “Brute forcing Wi-Fi Protected Setup,” 2011.

[12] Mohtadi, Hamed and Alireza Rahimi, “New attacks on wi-fi protected setup,” ACSIJ Advances in Computer Science: an International Journal, Vol. 4, Issue 5, No. 17, pp. 127-132, 2015.

[13] M. Jones, J. Bradley and N. Sakimura, “JSON Web Signature(JWS),” RFC7515, Internet Engineering Task Force(IETF), May 2015.

[14] “ISO/IEC/IEEE - International Standard - Telecommunications and information exchange between systems - Specific requirements for local and metropolitan area networks - Part 11: Wireless LAN medium access control and physical layer specifications,” IEEE Standards Association, 2022.

[15] International Telecommunication Union, “Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks,” Recommendation ITU-T X.509, 2019.

[16] F. Forsby, et al. “Lightweight X.509 Digital Certificates for the Internet of Things,” International Conference on Interoperability in IoT, International Conference on Safety and Security in IoT: Interoperability Safety and Security in IoT, Lecture Notes of the Institute for Computer Sciences Social Informatics and Telecommunications Engineering, vol. 242, pp. 123-133, 2018.

[17] V. Siddhartha, et al. “A Lightweight Authentication Protocol using Implicit Certificates for Securing IoT Systems,” Procedia Computer Science, vol. 167, pp. 85-96, 2020.

〔저자소개〕

유 호 제 (Ho-jei Yu)
2021년 2월 호서대학교 정보보호학과 학사
2022년 8월 호서대학교 정보보호학과 석사
email : hojei1452@naver.com

김 동 우 (Dong-woo Kim)
2023년 2월 호서대학교 컴퓨터공학부 학사
2023년 3월 ~ 현재 호서대학교 정보보호학과 석사과정
email : penetrick0@gmail.com

김 찬 희 (Chan-hee Kim)
2021년 2월 호서대학교 정보보호학과 학사
2023년 2월 호서대학교 정보보호학과 석사
email : my960816@naver.com

오 수 현 (Soo-hyun Oh)
1998년 2월 성균관대학교 정보공학과 학사
2000년 2월 성균관대학교 전기전자 및 컴퓨터공학과 석사(공학석사)
2003년 8월 성균관대학교 전기전자 및 컴퓨터공학과 박사(공학박사)
2004년 3월~현재 호서대학교 컴퓨터공학부 교수
email : shoh@hoseo.edu

임 성 식 (Sung-sik Im)
2023년 2월 호서대학교 컴퓨터공학부 학사
2023년 3월 ~ 현재 호서대학교 정보보호학과 석사과정
email : sungsik9797@gmail.com

김 서 연 (Seo-yeon Kim)
2023년 2월 호서대학교 컴퓨터공학부 학사
2023년 3월 ~ 현재 호서대학교 정보보호학과 석사과정
email : komtti@naver.com