IoT 환경에서 활용되는 TLS의 키 생성 과정 및 암호화 라이브러리 연구

김도원*, 최은영*, 박해룡*, 이주현**, 홍득조***
*한국인터넷진흥원
**전북대학교 컴퓨터공학과
***전북대학교 IT정보공학과
e-mail:kdw900920@gmail.com

A study of TLS key generation process and its library in IoT Environment

Do-won Kim*, Hae-ryong Park*, Juhyun Lee**, Deukjo Hong***
*Korea Internet & Security Agency
**Dept of Computer Engineering, Chonbuk National University
***Department of Information Technology, Chonbuk National University

요 약

암호 기술은 보안에 있어 필수적인 요소이며, 암호 키는 암호기술의 안전성에 있어 가장 핵심적인 요소이다. 시스템의 보안을 위해서는, 암호 키를 안전하게 생성하고 관리하는 것이 중요하다.

생체정보 같은 민감한 데이터를 주고받는 다양한 경량 디바이스나 센서들로 구성된 IoT 환경에서도, 암호 키는 중요한 보안 요소이다. IoT 기술이 발전하면서, 일반 PC나 스마트폰이 아닌 경량 디바이스에서도 일반적인 인터넷 환경에서 활용되는 통신프로토콜 및 보안기술들이 활용된다. 특히, 가장 많이 이용되는 TLS 프로토콜은 IoT 환경에서도 활발히 이용되고 있다.

본 논문에서는 보안프로토콜 TLS의 키 생성 과정을 분석하고 IoT 환경에서 자주 활용되는 라이브러리를 조사하였다.

1. 서론

상용에서 사용되는 대부분의 암호 기술은 안전성을 검증받은 후에 사용된다. 다양한 연구를 기반으로 안전한 암호 알고리즘들이 개발되고, 안전한 알고리즘을 적절하게 이용해야 안전한 보안 체계가 갖춰질 수 있다.

안전한 보안체계에서 가장 중요한 것 중 하나는 암호 키이다. 암호 키가 노출되면 어떤 알고리즘을 어떻게 사용하더라도 기밀성이나 무결성 등이 손상되고, 그렇게 되면 그 시스템은 안전성을 보장할 수 없게 된다. 그렇기 때문에 안전한 보안체계가 갖춰지기 위해서는 암호 키를 안전하게 생성하고 관리하는 방법이 필수적이다.

하지만, 안전하게 암호 키를 생성하는 것은 많은 비용을 발생시킨다. 이상적으로 암호 키를 생성하는 것은 적절한 방법으로 잡음원을 수집하고 난수발생기를 이용하여 난수성이 보장되는 충분한 길이의 난수를 생성해서 암호 키로 활용해야 한다. 이 과정에서 발생하는 비용은 연산 능력이 부족한 소형 디바이스에서는 다소 부담스러울 수 있다. 하지만, IoT 기술이 발전하면서 디바이스의 성능은 향상되고, 일반 인터넷 환경에서 이용되는 다양한 보안기술들이 IoT 환경에서도 이용되고 있다.

TLS프로토콜은 인터넷 환경에서 가장 널리 이용되는 보안 프로토콜이다. 특히 HTTP에 적용되어(HTTPS), 웹에서는 거의 모든 브라우저 및 서버들에서 필수적으로 사용하고 있다. IoT 환경에서도 익숙한 HTTP와 함께 사용하는 것이 용이하여 널리 사용되고 있으며, IoT 오픈플랫폼인 oneM2M이나 OCF 등에서도 TLS를 규격에 포함하고 있다.

본 논문에서는 TLS의 키 생성 과정을 분석하고 IoT에서 널리 사용되는 라이브러리를 조사하여 키 생성과 관련된 기능들을 조사하였다. 2장에서는 TLS프로토콜과 핸드쉐이크 과정을 살펴보고, 3장에서는 TLS프로토콜의 키 생성 과정을 자세하게 분석한다. 4장에서는 TLS 오픈소스 라이브러리인 openSSL과 mbedTLS의 키 생성 관련 기능을 살펴보고, 5장에서는 결론으로 마친다.

2. TLS 프로토콜 및 핸드쉐이크

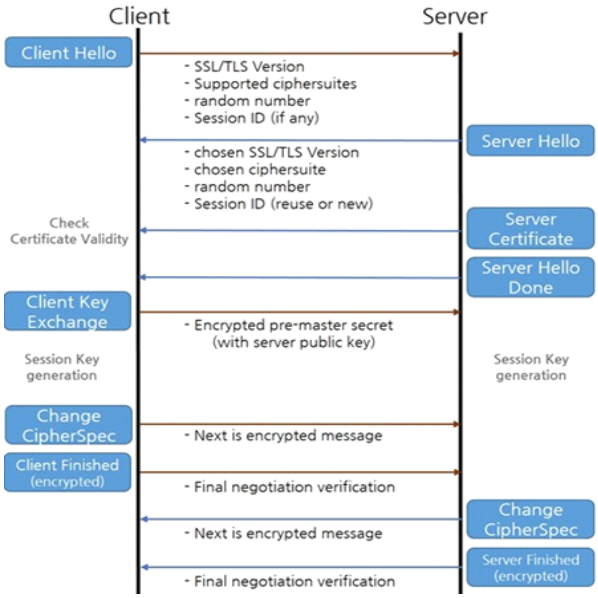
TLS(Transport Layer Security)는 전송 계층에 적용되는 보안 프로토콜로써, 클라이언트/서버 프로그램이 통신하는 과정에서 도청, 간섭, 위조를 방지하기 위해 설계되었다. 주로 웹브라우저와 서버 사이의 HTTP에 적용되어 HTTPS라는 이름으로 사용되며, 그 외에 FTP나 SMTP같은 프로토콜에도 사용될 수 있다[1].

TLS 프로토콜의 보안 규격은 핸드쉐이크(handshake) 과정에서 대부분 결정된다. 서버와 클라이언트가 암호화된

※ 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.2017-0-00267)

세션을 생성하기 전에, 핸드쉐이크 과정에서 상호인증을 통해 암호 알고리즘을 결정하고 그에 필요한 암호 키를 교환한다.

TLS 프토토콜 버전은 1.3까지 있으며[2], (그림 1)은 TLS1.2의 핸드쉐이크 과정이다[3]. 서버와 클라이언트가 필요한 암호 알고리즘(Cipher suite)을 선택하고, 각자 생성한 난수를 주고 받는다. 세션을 암호화하는 세션 키는 여기서 주고받는 난수를 이용하여 생성한다.



(그림 1) TLS1.2 핸드쉐이크

TLS1.3의 핸드 쉐이크 과정은 TLS1.2보다 가볍고 빠르게 설계되었지만, 세션 키를 생성하는 과정은 유사하다.

3. TLS 키 생성 과정

클라이언트는 48byte의 난수인 'pre-master secret'을 생성하여, 서버의 공개키를 이용하여 안전한 방법으로 전송한다. 서버와 클라이언트는 서로 주고받은 난수와 이 pre-master secret을 이용하여, (그림 2)와 같이 master secret을 생성한다.

```
master_secret = PRF(pre_master_secret, "master secret",
                    ClientHello.random + ServerHello.random)
                    [0..47];
```

(그림 2) master_secret 생성 방법

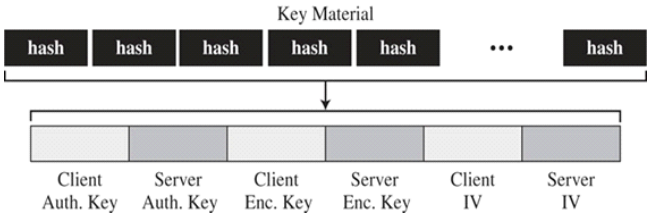
여기서 사용되는 PRF(Pseudo Random Function)는 (그림 3)과 같다. (그림 3)에서 A값은 이전 HMAC값을 이용하며, 첫 A값은 seed값이다. seed는 PRF의 pre_master_secret 외의 두 변수의 연접이다.

```
P_hash(secret, seed) = HMAC_hash(secret, A(1) + seed) +
                       HMAC_hash(secret, A(2) + seed) +
                       HMAC_hash(secret, A(3) + seed) + ...
```

(그림 3) PRF 함수

예를 들어, SHA-256을 기반으로 80바이트의 데이터를 싶은 경우, HMAC 함수를 3번 사용하여 96바이트(32*3)를 생성하고, 뒤의 16바이트를 버려서 80바이트를 생성한다.

이렇게 생성된 master secret과 서버와 클라이언트가 주고받는 난수를 PRF함수에 다시 한 번 이용하여, 충분한 길이의 key material을 생성한다. 여기서 key material의 길이는 필요한 인증 키, 세션 키, IV 등을 모두 얻을 수 있는 길이이다(그림 4).



(그림 4) Key material

키 생성 과정에서는 pre-master secret과 PRF에서 사용하는 해시 함수가 중요하다. 클라이언트가 생성하는 유일한 비밀 값인 pre-master secret은 난수발생기 등을 이용하여 안전하게 생성해야 하고, MD5나 SHA-1같은 취약한 해시 함수를 PRF에서 사용하지 않아야 한다.

4. OpenSSL/mbedTLS 함수 분석

본 장에서는 TLS 오픈소스 중 가장 많이 활용되는 openSSL[4]과 mbedTLS[5]의 암호기능 관련 함수 중 키 생성에 관련된 함수를 조사하였다.

<표 1> openSSL 키 생성 관련 함수

openssl/ssl/tls13_enc.c	
1	tls13_hkdf_expand
	인자 값에 따라 key, finished key, iv 값 등을 만들어내는 함수
2	tls13_derive_key
	tls13_hkdf_expand함수를 호출하여 key를 생성하는 함수
3	tls13_derive_iv
	tls13_hkdf_expand함수를 호출하여 iv값을 생성하는 함수
4	tls13_derive_finishedkey
	tls13_hkdf_expand함수를 호출하여 finished key를 생성하는 함수
5	tls13_generate_secret
	인자 값에 따라 secret 값을 생성하는 알고리즘 함수
6	tls13_generate_handshake_secret

	tls13_generate_secret함수를 호출하여 handskae secret값을 생성
7	tls13_generate_master_secret
	tls13_generate_secret함수를 호출하여 master secret 값을 만드는 함수
8	tls13_generate_master_secret
	tls13_generate_secret함수를 호출하여 master secret 값을 만드는 함수
9	tls13_final_finish_mac
	handshake 과정에서 마지막 메시지의 mac값 계산 함수
10	tls13_setup_key_block
	암호와 해시 값 초기화에 사용되는 함수

<표 2> mbedTLS 키 생성 관련 함수

mbedtls/library/sssl_ciphersuites.c	
1	mbedtls_ssl_list_ciphersuites
	ciphersuites 목록을 반환하는 함수
mbedtls/library/ssl_cli.c	
1	ssl_generate_random
	TLS handshake에서 사용되는 클라이언트의 random 값을 만든다
2	ssl_write_encrypted_pms
	pre-master secret을 생성하고 서버의 RSA 키를 이용해 암호화
mbedtls/library/ssl_srv.c	
1	ssl_parse_client_key_exchange
	키 교환 단계에서 클라이언트의 키를 받아오는 함수
mbedtls/library/ssl_tls.c	
1	psa_status_t setup_psa_key_derivation
	키 생성에 사용되는 함수
2	ssl_compute_master
	master secret을 계산하는 함수
3	mbedtls_ssl_derive_keys
	random 값을 이용하여 secret 값 계산 함수를 호출하고 값을 저장하는 함수
4	mbedtls_ssl_psk_derive_premaster
	pre-master secret 값을 생성한다.

6. 결론

본 논문에서는 SSL/TLS 프로토콜의 키 생성 과정과 많이 활용되는 openSSL과 mbedTLS의 키 생성 함수에 관련된 함수를 살펴보았다.

TLS 프로토콜을 사용하더라도, 취약한 암호 키를 사용하게 되면 그 시스템은 안전성을 보장할 수 없다. 시스템을 구성하고 TLS 프로토콜을 적용할 때, 키 생성에 필요한 중요 요소들을 안전하게 세팅하여야 한다.

참고문헌

[1] 전송 계층 보안 - 위키백과, ko.wikipedia.org/wiki/전송_계층_보안

[2] RFC 8446 - The Transport Layer Security (TLS) Protocol Version 1.3, https://tools.ietf.org/html/rfc8446

[3] RFC 5246 - The Transport Layer Security (TLS) Protocol Version 1.2, https://tools.ietf.org/html/rfc5246

[4] OpenSSL, https://www.openssl.org/

[5] mbedTLS, https://tls.mbed.org/