선택적인 암호화를 지원하는 TCP에 관한 연구

성정기* · 서혜인* · 김은기*

*한밭대학교

A Study on the TCP Supporting Optional Encryption

Jeong-gi Seong* · Hye-in Seo* · Eun-gi Kim*

*Hanbat National University

E-mail : taro1714@naver.com

요 약

SSH, SSL/TLS 등의 보안 프로토콜들은 TCP 상에서 동작하며 응용 계층이 전송하는 모든 데이터를 암호화한다. 하지만 모든 데이터를 암호화하는 것은 기밀성이 요구되지 않는 데이터도 암호화하므로 불필요한 성능저하를 발생시킨다. 본 논문에서는 응용 사용자에 의해 기밀성이 요구되는 데이터만 선택적으로 암호화하는 TCP OENC(Optional Encryption)를 제안한다. TCP OENC는 기본적인 TCP 표준 동작을 따르며, 응용 사용자가 데이터 암호화를 요구할 때만 동작한다. TCP OENC는 처음 데이터 암호화 이전에 키 협의를 수행하며, 이후 키 협의를 통해 공유된 키를 사용하여 응용 사용자가 원하는 데이터를 암호화하여 전송한다.

ABSTRACT

The security protocols such as SSH and SSL/TLS operate over TCP and encrypt all data from the application layer. However, this method has unnecessary performance degradation because it encrypts even data which does not require confidentiality. In this paper, we propose TCP OENC(TCP Optional Encryption) which optionally encrypts only confidential data by the application user. The proposed TCP OENC is in accordance with TCP standard operation, and it operates if application user demand on encrypting data. Before the TCP OENC sends first encrypted data, performs the key agreement, and then encrypts and sends data which application user is desired by using shared key obtained from the key agreement.

키워드

TCP, 보안, 암호화, 전송 계층, 네트워크

Ⅰ. 서 론

최근 지속되는 사이버 공격의 증가와 개인정보 보호에 대한 인식 강화로 인해 많은 인터넷 서비스들은 전송하는 데이터를 암호화한다. 인터넷 웹, 파일 전송, 영상 스트리밍과 같은 응용 서비스들은 데이터를 암호화하기 위해 SSH[1], SSL/TLS[2]와 같은 보안 프로토콜을 주로 사용한다. 대부분의 보안 프로토콜은 TCP 상에서 동작하며 응용 계층이 전송하는 모든 데이터를 암호화한다. 하지만 모든 데이터를 암호화하는 것은 기밀성이 요구되지 않는 데이터까지 암호화하므로 불필요한 성능저하를 발생시킨다. 따라서 본 논문에서는 선택적으로 암호화를 수행하는 TCP 옵션을 제안하여 이와 같은 문제를 해결하고자 한다. 제안하는 TCP 옵션은 응용 사용자가 별도

의 보안 프로토콜 없이 선택적으로 데이터를 암호화하는 것을 지원한다. 기본적으로 TCP 표준 동작을 따르고, 응용 사용자에 의해 데이터가 기밀성이 요구될 경우에만 동작한다.

II. TCP Optional Encryption

본 논문에서 제안하는 선택적인 암호화를 지원하는 TCP는 TCP OENC(Optional Encryption)로 기술한다. TCP OENC는 하나의 TCP 연결에서 사용자의 요구에 따라 데이터를 암호화하도록 지원한다. 그림 1은 TCP OENC의 동작을 나타낸다.

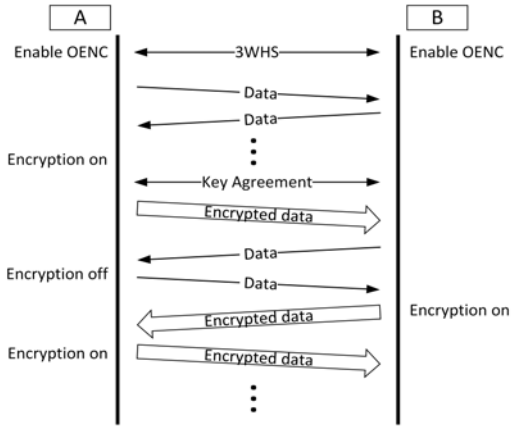

그림 1. Overview of TCP OENC

TCP OENC는 3WHS (3-Way Handshaking)을 통해 OENC 기능을 활성화하고 연결을 설정한다. TCP 연결 설정 이후 기존의 TCP와 동일하게 동작하다가 호스트가 데이터 암호화를 원하는 경우, 데이터를 암호화하여 전송한다. 처음 암호화를 수행한다면 두 호스트는 ECDH(Elliptic Curve Deffie-Hellman) 알고리즘[3]을 사용한 키 협의를 수행한다. 키 협의 이후 두 호스트는 공유하는 키를 생성하고, 공유된 키를 사용하여 자신이 원할 때 데이터를 암호화하여 전송한다.

2.1 TCP OENC 옵션

TCP OENC는 기존 TCP와의 하위 호환성을 가지기 위해 TCP 옵션으로 동작한다. TCP OENC는 1바이트 크기의 서브 옵션을 사용하여 동작한다. 표 1은 TCP OENC에서 사용하는 서브 옵션을 나타낸다.

표 1. TCP OENC Options

Sub Option	Description
PROBE	Probe OENC
PERMIT	Permit OENC
INIT	Use for key agreement
ENC_ON	Indicate starting encryption
ENC_OFF	Indicate stopping encryption

2.2 OENC 옵션 확인 및 보안 알고리즘 협의

TCP OENC는 3WHS를 통해 서로의 OENC 지원여부를 확인하고 보안 알고리즘을 협의한다. TCP OENC는 기본적으로 키 협의 알고리즘으로 ECDH, 암호화 알고리즘으로 AES(Advanced Encryption Standard)[4]를 지원한다.

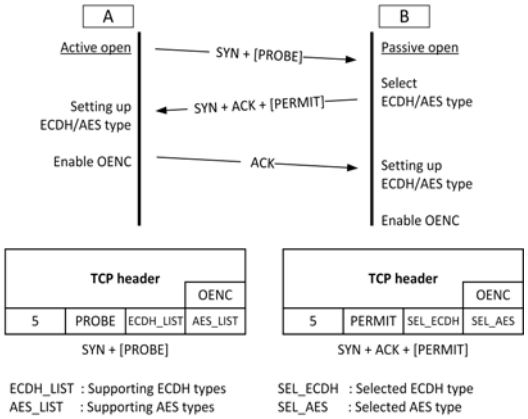

그림 2. TCP 3-Way Handshaking with OENC

그림 2는 TCP에서 OENC를 사용하는 3WHS 과정을 나타낸다. 3WHS 과정에서 A는 PROBE 옵션을 포함한 SYN를 전송하여 TCP 연결을 요청한다. PROBE 옵션은 상대 호스트의 OENC 지원 여부를 확인함과 동시에 자신이 사용 가능한 ECDH/AES 알고리즘 타입들을 명시한다. ECDH_LIST와 AES_LIST는 각 1바이트의 크기를 가지며, 각 비트는 알고리즘의 타입을 명시하고 있다. 표 2는 ECDH 타입, 표 3은 AES의 타입을 나타낸다.

표 2. ECDH types for TCP OENC

Bit No.	ECDH Type
0	ECDHE-NIST-P192-SHA512
1	ECDHE-NIST-P256-SHA512
2-7	Undefined

표 3. AES types for TCP OENC

Bit No.	AES Type
0	AES-128-CTR
1	AES-192-CTR
2	AES-256-CTR
3-7	Undefined

B는 PROBE가 포함된 SYN를 수신하면 ECDH와 AES의 타입을 선택한다. 선택된 ECDH 타입은 SEL_ECDH, AES 타입은 SEL_AES이다. 이후 PERMIT 옵션을 SYN+ACK와 함께 전송한다.

A는 PERMIT를 수신하면 선택된 ECDH와 AES 타입으로 보안 알고리즘을 설정하고 OENC를 활성화한 후 ACK를 전송하여 연결 설정을 완

료한다. B는 A의 ACK를 수신하면 OENC를 활성화한 후 연결 설정을 완료한다. 만약 B가 OENC를 지원하지 않는다면 PROBE 옵션은 무시되고 기존의 TCP 동작을 수행한다[5].

2.3 키 협의

TCP OENC가 활성화되면 두 호스트는 현재 연결을 유지하는 동안 자신이 원할 때 데이터를 암호화하여 전송할 수 있다. 하지만 데이터 암호화를 수행하기 위해 두 호스트는 동일한 키를 가져야만 한다. 예를 들면, 두 호스트 중 A가 처음 암호화를 수행한다면 데이터를 암호화하기 전에 키 협의를 통해 공유 키를 생성한다. 그림 3은 TCP OENC의 키 협의 과정을 나타낸다.

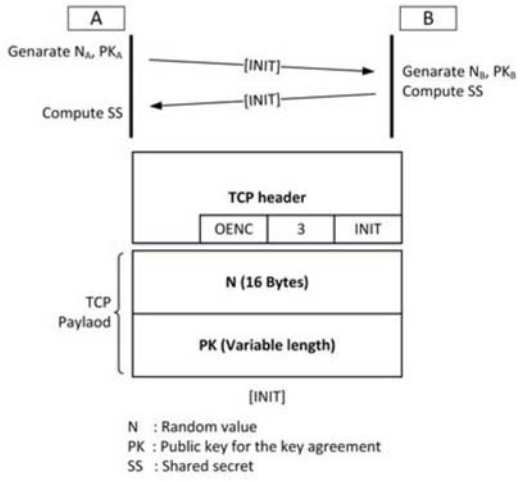

그림 3. TCP OENC Key Agreement

A는 먼저 키 협의를 위한 16바이트 크기의 랜덤 값 N_A과 ECDH 공개키 PK_A를 생성한다. N_A와 PK_A가 생성되면 INIT 옵션을 통해 B에게 전송한다. 그림 3에서 N_A와 PK_A가 TCP 페이로드로 전송되는데, 여기서 N_A와 PK_A가 TCP 옵션의 최대 크기 40바이트를 초과하기 때문에 TCP 페이로드를 사용하여 전송한다. 따라서 INIT 옵션이 포함된 세그먼트는 응용 데이터를 포함할 수 없으며, 응용 계층으로 전달되지 않는다.

B가 INIT를 수신하면 N_B와 PK_B를 생성하고 A에게 INIT 옵션과 함께 전송한다. 이후 B는 A의 PK_A와 자신의 PK_B를 사용하여 공유 키 SS를 생성한다. 이후 SS와 N_A, N_B를 이용하여 AES 키와 IV(Initialization Vector)를 생성한다. A도 INIT을 수신하면 B와 동일한 동작을 수행하여 B와 동일한 AES 키와 IV를 생성한다.

2.4 암호화/복호화 제어

키 협의를 완료하면 두 호스트는 임의대로 데이터를 암호화하여 전송할 수 있다. 하나의 호스트가 암호화 동작을 수행하면 상대 호스트는 복호화 동작을 하므로 암호화가 수행 중임을 알아야 한다. TCP OENC에서 암호화 동작을 제어하기 위해 ENC_ON와 ENC_OFF 옵션을 사용한다. 그림 4는 TCP OENC에서 두 호스트가 데이터 암호화 시작과 중지를 처리하는 과정을 나타낸다.

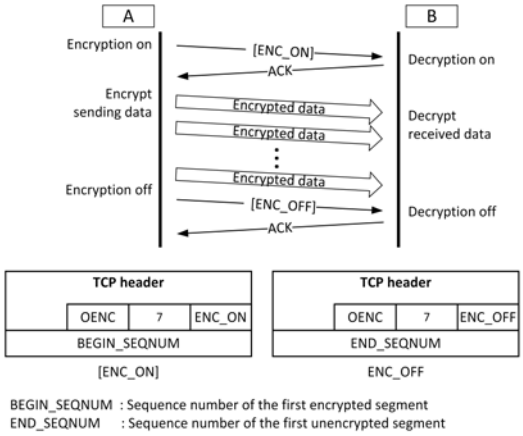

그림 4. Encryption and Decryption Control

그림 4와 같이 A는 데이터 암호화를 시작하기 이전에 ENC_ON을 전송한다. ENC_ON은 데이터 암호화 시작을 알리는 옵션으로, 처음 암호화된 세그먼트의 순서번호 BEGIN_SEQNUM을 포함한다. B는 ENC_ON을 수신하면 BEGIN_SEQNUM의 세그먼트부터 데이터를 복호화하여 응용 계층에 전달한다. 만약 A가 데이터 암호화를 중지할 경우, 암호화 중지를 알리는 ENC_OFF를 전송한다. ENC_OFF의 END_SEQNUM은 암호화되지 않은 첫 번째 세그먼트의 번호이다. ENC_OFF을 수신한 B는 END_SEQNUM을 가지는 세그먼트부터 데이터 복호화를 수행하지 않고 기존의 TCP 동작을 수행한다.

III. 결 론

본 논문에서는 선택적인 암호화를 지원하는 TCP OENC를 제안하였다. TCP OENC는 응용 사용자의 요구에 따라 데이터를 암호화하도록 설계되었고, 기존 TCP와의 하위 호환성을 가진다. TCP OENC는 기밀성이 요구되는 데이터만 암호화하여 전송하도록 지원하여 응용 사용자가 필요에 따라 별도의 보안 프로토콜 없이 데이터 암호화를 수행하도록 설계되었다. 결과적으로 응용 사용자는 기밀성이 필요한 데이터만 암호화하면 불필요한 성능 저하를 줄일 수 있을 것으로 기대된다.

참고문헌

[1] IETF Std. RFC 4251, The Secure Shell (SSH) Protocol Archi tecture, IETF, T. Ylonen. 2006.

[2] RFC 5246, The Transport Layer Security (TLS) Protocol Version 1.2, Retrieved 16, Feb. 2015.

[3] Seok-Ho Kim, “Comparison and analysis on efficiency of scalar multiplication for Elliptic Curve Cryptosystem,” M.S. dissertation, Korea Maritime and Ocean University, pp. 11-15, 2003.

[4] FIPS Std. FIPS PUB 197, Adveanced Encryption Standard (AES), FIPS, NIST, 2001.

[5] RFC 1122, Requirements for Internet Hosts -- Communication Layers, IETF, October. 1989.