보안기능을 지원하는 TFTP 프로토콜의 설계 및 구현

윤승욱* · 권현경** · 옥성진*** · 강정하**** · 김은기*****

*한밭대학교 정보통신공학과

Design and Implementation of TFTP Protocol Supporting Network Security Functionalities

Seoung-uk Yuen* · Hyun-kyung Kwon** · Sung-Jin Ok*** · Jung-Ha Kang**** · Eun-Gi Kim*****

*Hanbat National University

E-mail : gusrud1832@nate.com

요 약

TFTP(Trivial File Transfer Protocol)는 UDP(User Datagram Protocol) 기반의 파일 전송 프로토콜이다. TFTP는 프로토콜 구조가 단순하여 작은 크기의 데이터를 빠른 속도로 전송할 때 사용된다. 하지만 TFTP는 보안 기능을 지원하지 않기 때문에 데이터 노출의 위험이 있다. 본 논문에서는 Diffie-Hellman 키 교환 방식과 AES-CBC(Advanced Encryption Standard-Cipher Block Chaining) 암호화 방식을 이용하여 TFTP 프로토콜에 보안 기능을 추가하였다. Diffie-Hellman 키 교환 방식을 이용하여 두 사용자 간에 비밀 키를 공유하도록 하였고, AES-CBC 암호화를 지원하여 기밀성을 제공하도록 하였다. 수신된 데이터는 암호화 과정의 역으로 복호화를 수행하였다. WireShark 프로그램을 통하여 암호화된 데이터가 전송 되는 것을 확인하였다.

키워드

TFTP (Trivial File Transfer Protocol), Diffie-Hellman, AES-CBC (Advanced Encryption Standard-Cipher Block Chaining), AES, Openssl

Ⅰ. 서 론

TFTP는 단순한 구조로 되어 있으며 저용량의 데이터를 실시간으로 송수신하는데 적합한 파일 전송 프로토콜이다[1]. 한 예로 라우터가 부팅될 때, Bootstrap과 구성 파일을 다운로드 하는데 사용된다. TFTP 단점은 패킷 전송 시에 공격자가 데이터를 가로챌 때 데이터의 정보가 노출될 수 있다는 보안상의 취약점이 있다.

본 논문에서는 이러한 취약점을 보완하기 위해 Diffie-Hellman 키 교환 방식과 AES-CBC 암호화를 이용하여 기밀성을 제공하는 TFTP모델을 제안한다.

본 논문의 구성은 다음과 같다. 2장에서는 Diffie-Hellman 알고리즘, AES 알고리즘, CBC mode에 대하여 기술하고, 3장에서는 암호화된 TFTP 설계 및 구현에 관한 내용을 기술한다. 마지막으로 4장에서는 암호화된 TFTP의 결과 및 기대 효과에 대하여 기술한다.

Ⅱ. 관련 연구

2. 1 Diffe-Hellman 알고리즘

Diffie-Hellman 키 교환은 암호 키를 교환하는 방법으로, 두 사람이 암호화되지 않은 통신망을 통해 공통의 비밀 키를 공유할 수 있도록 한다. 간단하게 원리를 설명하면 q와 α는 두 사용자에게 공유되어 있는 값이다. q는 큰 소수이고, α는 q의 원시근이다. 사용자 A와 B는 q보다 작은 임의의 값 즉, 개인키를 생성한다. q, α, 개인키를 이용하여 자신의 공개키를 생성한다. 이후, 사용자 A의 공개키는 B에게 전송하고 사용자 B의 공개키는 A에게 전송한다. 상대방의 공개키와 자신의 개인키, q를 이용하여 사용자 A와 B간의 비밀키를 생성한다. 아래에 (그림 1)는 Diffie-Hellman 키 교환을 나타낸다[2][3].

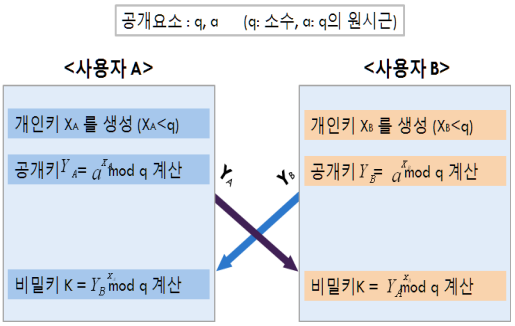

그림 1. Diffie-Hellman 키 교환

(그림 1)에서 K 값은 두 사용자간의 비밀키로, AES 알고리즘의 key로 사용된다.

2.2 AES 알고리즘

AES는 암호화와 복호화 과정에서 동일한 키를 사용하는 대칭키 알고리즘이다. AES는 암호와, 복호화를 4x4 행렬로 표현하며 연산을 수행한다. (그림 2)는 AES 암호화 과정이다[4].

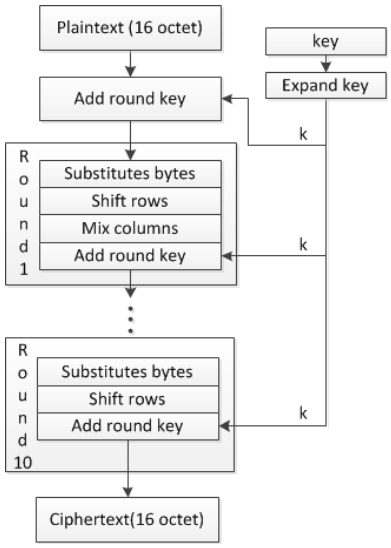

그림 2. AES 암호화 알고리즘

AES 암호화 방식은 128, 192, 256bit의 키의 길이에 따라 필요한 라운드 수는 10, 12, 14이다.

본 논문에서는 128bit 길이의 키를 사용하므로 10라운드를 수행한다. 각 라운드마다 Subbytes, Shift rows, Mix columns, Add round key 4단계를 수행한다.

- Subbytes: S-box를 이용한 치환연산
- Shift rows: 단순 자리바꿈
- Mix columns: 각 열마다 치환연산
- Addroundkey: 각 라운드 키와 XOR연산

이러한 과정을 통하여 16byte의 암호문을 얻을 수 있다.

2.3 CBC mode

CBC 모드는 한 단계 앞에서 수행된 결과로 출력된 암호문 블록에 평문 블록을 XOR한 후 암호화하는 과정을 반복 수행하는 방법이다. 최초의 평문 블록을 암호화할 때는 초기화 벡터(IV)를 이용한다. 복호화는 암호화의 반대 과정으로 수행한다. CBC mode의 암호화 과정은 (그림 3)과 같다.

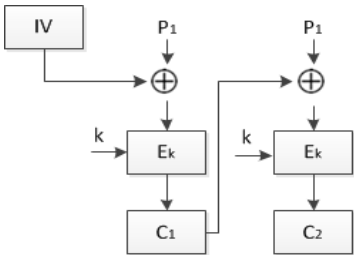

그림 3. CBC mode

마지막 블록이 16byte의 길이보다 작을 경우 패딩처리를 한다[4].

Ⅲ. 설계 및 구현

3.1 암호화된 TFTP 설계

TFTP 프로토콜은 UDP 기반의 파일을 전송하는 프로토콜이다. (그림 4)는 본 논문에서 제안한 보안기능이 추가된 TFTP의 서버와 클라이언트 간 동작을 나타낸다.

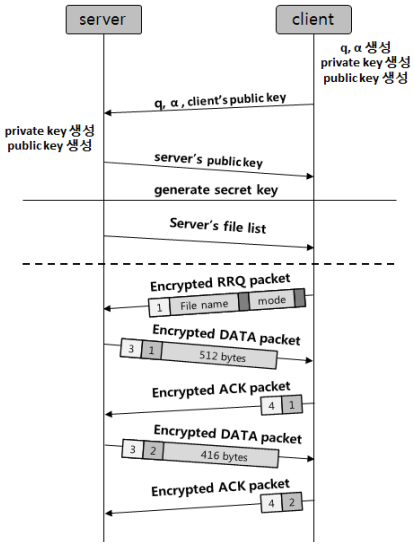

그림 4. 암호화된 TFTP 동작 흐름도

서버와 클라이언트간에 비밀키 생성을 위하여 클라이언트는 q와 α를 생성한 후 공개키와 개인키를 생성 한다. 그리고 클라이언트의 q, α, 공개키를 서버로 전송한다. 서버는 전송받은 q와 α를 이용하여 자신의 개인키와 공개키를 생성

한 후 클라이언트에게 공개키를 전송 한다. 서버와 클라이언트는 q, α, 자신의 개인키, 서로의 공개키를 이용하여 비밀 키를 생성한다.

비밀 키 생성이 완료된 후, 서버는 클라이언트에게 자신의 비밀 키를 이용하여 암호화된 파일 목록을 전송한다. 클라이언트는 서버에게 명령어를 전송한다. 전송되는 명령어는 TFTP 메시지 구조로 전송된다. (그림 5)는 TFTP 메시지 구조를 나타낸다[1][5].

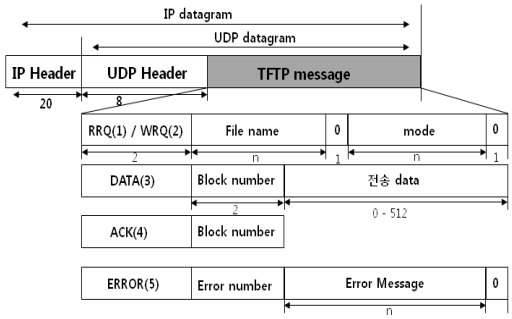

그림 5. TFTP 메시지 구조

클라이언트는 명령어 데이터를 AES-CBC 모듈을 이용하여 암호화한 후 서버에 전송한다. 패킷을 수신한 서버는 AES-CBC 모듈을 이용하여 복호화한 후 클라이언트가 요청한 파일의 데이터를 TFTP 메시지 구조에 맞게 생성 한 후 암호화하여 클라이언트로 전송한다. 클라이언트는 수신한 패킷을 복호화한 후 데이터를 저장하고, ACK 메시지를 만들어 암호화하여 서버에 전송한다. 이와 같은 과정을 파일 전송이 완료될 때까지 반복한다. 클라이언트가 수신한 데이터의 크기가 512byte보다 작거나 0이라면 ACK 메시지를 전송 후 종료한다.

3.2 암호화된 TFTP 서버 클라이언트 구현

(그림 6)은 구현된 TFTP 서버의 동작 알고리즘을 나타낸다.

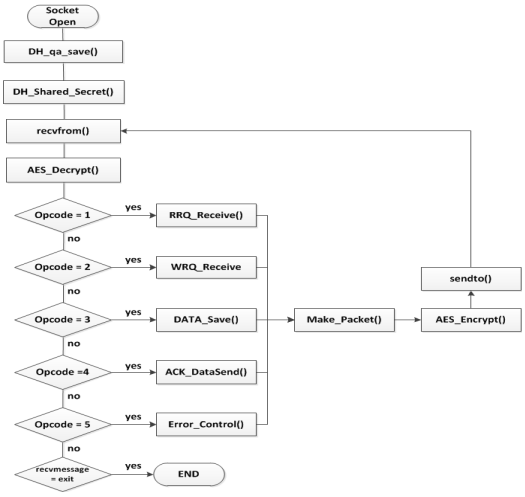

그림 6. 구현된 TFTP 서버 알고리즘

TFTP 서버에서 보안 기능을 위하여 구현된 함수는 다음과 같은 기능을 수행한다.

- DH_qa_save(): q와 α값 수신 후 저장
- DH_Shared_Secret(): 비밀키 생성
- AES_Encrypt(): 패킷 암호화
- AES_Decrypt(): 패킷 복호화
- RRQ_Receive(): RRQ 패킷 파싱
- WRQ_Receive(): WRQ 패킷 파싱
- DATA_Save(): DATA 패킷 파싱 후 데이터 저장
- ACK_Send(): ACK 패킷 파싱
- Error_Control(): Error 처리
- Make_Packet(): 메시지 구조 생성

(그림 7)은 구현된 TFTP 클라이언트의 동작 알고리즘이다.

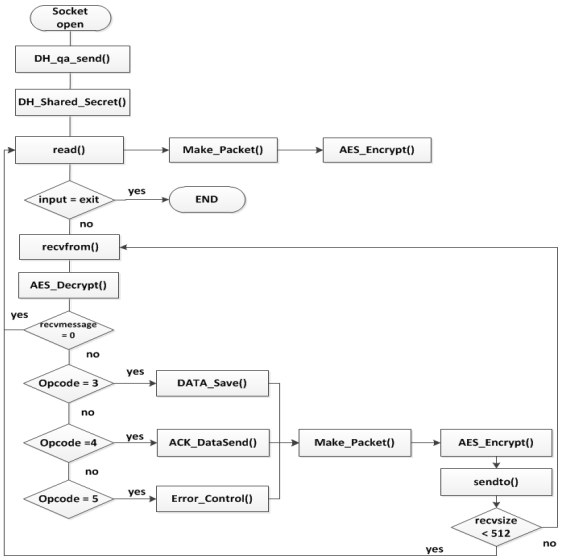

그림 7. 구현된 TFTP 클라이언트 알고리즘

- DH_qa_send(): q와 α값 생성 후 송신
- DH_Shared_Secret(): 비밀키 생성
- AES_Encrypt(): 패킷 암호화
- AES_Decrypt(): 패킷 복호화
- DATA_Save(): DATA 패킷 파싱 후 데이터 저장
- ACK_Send(): ACK 패킷 파싱
- Error_Control(): Error 처리
- Make_Packet(): 메시지 구조 생성

Ⅳ. TFTP의 결과 및 기대 효과

4.1 암호화된 TFTP 실행 결과

TFTP는 송수신 간에 데이터가 노출된다. (그림 8)은 암호화가 적용되지 않은 TFTP의 데이터 전송을 wireshark 프로그램으로 캡쳐한 화면 이다.

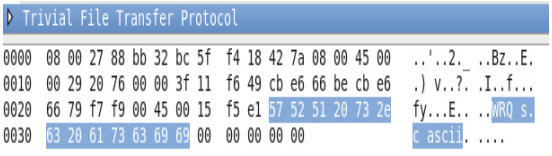

그림 8. 기존 TFTP 데이터 전송

(그림 9)는 클라이언트에서 서버로 전송할 명령어가 암호화 되어 생성되는 것을 보여준다. (그림 10)은 wireshark를 통해 암호화된 데이터가 전송되어지는 것을 보여준다.

```
++++++++++Enter the command++++++++++
[TFTP]# WRQ s.c ascii
send encrypte message
[65] [a0] [13] [f1] [6a] [f5] [f3] [8a]
[9b] [b1] [28] [7d] [54] [1e] [d6] [f]
```

그림 9. 암호화된 TFTP 데이터

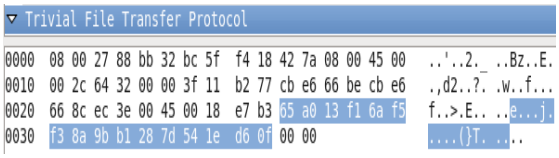

그림 10. 암호화된 TFTP 데이터 전송

클라이언트에서 입력한 명령어를 암호화하여 hex값으로 출력한 결과와 서버로 데이터를 전송하는 과정에서 wireshark로 패킷을 확인한 결과 데이터의 내용이 암호화 되어 기밀성이 보장되는 것을 알 수 있다.

4.2 기대 효과

본 논문은 Diffie-Hellman 키교환 방식과 AES-CBC 암호화 방식을 이용하여 TFTP 송수신 간에 데이터 노출에 대해 보완하였다. Diffie-Hellman 키교환 방식을 이용하여 두 사용자 외에 다른 사용자의 개입을 방지 하였고 AES-CBC 암호화 방식을 이용하여 데이터의 기밀성을 추구 하였다. 두 사용자 외에는 비밀 키를 알 수 없기 때문에 암호화된 데이터를 복호화 할 수 없다. 따라서 안전한 TFTP의 데이터 송수신이 가능해진다.

감사의 글

본 연구는 교육부와 한국연구재단의 지역혁신 인력양성사업으로 수행된 연구결과임 (No. 20130 1590001).

참고문헌

[1] Forouzan, TCP/IP Protocol Suite(4th ed.), Mcgraw-Hill, pp.643-651, 2012.

[2] John Viega, Network Security with OpenSSL, O'Reilly, Chap. 8.2, 2002.

[3] James F.Kurose, Keith W.Ross, Computer Networking, Addison Wesley, pp.706-711, 2007.

[4] Messier, John Viega, Secure Programming Cookbook for C and C++, O'Reilly, Chap. 5.6.3, Chap. 5.17.3, 2003.

[5] Fred Halsall, Computer Networking and the Internet(5th ed.), Addison Wesley, pp.538-541, 2005.