TLS와 OTP를 이용한 보안 FTP 기법

김의정 · 서태호

공주대학교

Secure FTP Technique with TLS and OTP

Eui-jeong Kim · Tae-ho Seo

Kongju National University

E-mail : ejkim@kongju.ac.kr / syudal6712@smail.kongju.ac.kr

요 약

FTP(File Transfer Protocol)는 컴퓨터간의 파일을 전송하는 프로토콜로 암호화된 연결인 TLS를 이용한 FTPS, 이와 비슷한 SSH에 기반한 SFTP가 있다. ID와 Password 도용사례를 해결하기 위해 1회용 비밀번호인 OTP(One-Time Password)를 적용하려는 기조가 확산되었다. 본 논문에서는 클라이언트의 변화 없이 TLS Level에서 데이터 전송과 OTP를 이용한 신원 인증 모두 가능함을 확인 하였다.

ABSTRACT

File Transfer Protocol (FTP) is a protocol for transferring files between computers, FTPS with TLS, and SFTP based on similar SSH. The keynote to apply one-time password (OTP) has been diffused to resolve identity and password theft cases. In this paper, it has been confirmed that both data transmission and identity authentication using OTP at TLS level are possible without significant change in user client.

키워드

TLS FTP OTP X.509

Ⅰ. 서 론

클라우드 컴퓨팅이 발전함에 따라 인터넷 어딘가의 저장 공간에 정보를 저장해두고, 인터넷에 연결이 되어 있다면 필요한 순간에 언제 어디서든 정보에 접근 할 수 있게 되었다.

정보를 전송하는 방법에는 여러 프로토콜이 존재하지만, 정보를 전송하기 위해 고안된 FTP(File Transfer Protocol)는 1971년에 최초 고안되었으며 통신 규약으로 현재도 사용하고 있는 프로토콜이다. FTP의 안정성은 입증되었지만, 로그인에 필요한 ID와 Password, 파일 데이터가 모두 평문으로 전송되어 보안성이 매우 떨어진다는 단점이 존재했다[1]. 암호화된 통신을 지원하는 TLS를 이용한 FTPS와 SSH를 이용한 SFTP가 있으나 SFTP는 FTP와는 관련이 없는 새로운 프로토콜이다[2].

로그인에 필요한 ID와 Password를 사람이 관리하기에 이를 여러 가지 수법을 이용하여 탈취·도용하여 다수의 해킹 사례가 보고되고 있다[3]. 따라서 탈취·도용을 이용한 정보 유출을 막기 위한 기술이 필요하다.

본 논문에서는 공개키 암호화 방식인 TLS와 로그인을 할 때 마다 Password가 바뀌는 기술인 OTP를 이용하여 MITM(Man in the middle Attack, Bounce Attack)[4]과 로그인 정보 탈취·도용을 방지하지만, 클라이언트의 변화 없는 새로운 보안 FTP기법을 제안한다.

Ⅱ. 관련 연구

2.1. TLS(Transport Layer Security)에 관한 연구

TLS의 사용은 웹을 포함하여 이메일, VoIP과 같은 인터넷 통신 전반에 사용되고 있다.

TLS는 SSL(Secure Socket Layer)에 기반한 기술로 서로를 신뢰 할 수 있는지 확인하기 위해 전자서명에 기반한 인증서를 사용하며 제3자에 의한 도청을 방지하기 위해서는 인증서의 공개키를 이용하여 통신 데이터를 암호화한다[5]. 전자서명에는 소유자 정보와 발행자가 포함되어있으므로 사용자는 자신이 접속하려는 서버가 맞는지 확인 할 수 있어 MITM(Man in the middle attack, Bounce Attack)을 방지 할 수 있다. 아래 그림1은 TLS가 서로의 신원을 확인하기 위해 Handshake하는 과정을 보여주는 그림이다.

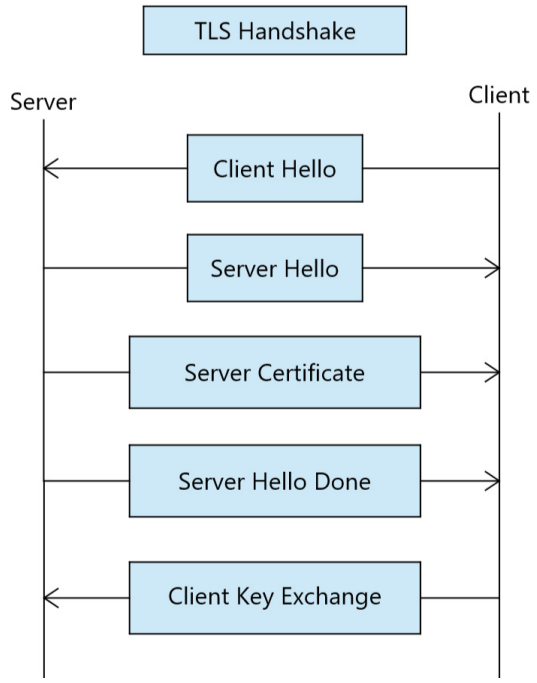

그림 1. TLS Handshake 순서도

2.2 OTP(One-Time Password)에 관한 연구

OTP는 SEED값에 기반한 1회용 비밀번호 시스템이다. 1회용 비밀번호이므로 노출되어도 재사용이 불가능하며 인터넷에 연결 하지 않고 SEED와 시간에만 기반하여 비밀번호를 생성하므로 인터넷에 연결되어 있음으로서 발생하는 패킷 감청·유출에서 안전하다[6,7].

계정도용방지를 위해 쓰는 보안메커니즘 중 보안성이 가장 높은 수단은 OTP다[8]. 키로거를 사용하여 키 스니핑 공격을 할 경우, 입력된 값이 유출될 수 있다는 문제가 있지만, 실제로 OTP를 인증할 수 있는 시간은 매우 짧기 때문에 공격이 성공할 확률은 매우 낮다[9].

Ⅲ. 본 론

3.1 FTPS(Securing FTP with TLS) 구현

클라이언트가 서버에 연결을 요청하면 서버가 클라이언트에 X.509 방식으로 생성된 인증서를 전송해야 한다[10]. X.509 방식의 인증서는 Oracle社의 keytool 유틸리티를 사용하여 인증서를 생성 할 수 있다[11]. 그림 2는 keytool을 이용하여 인증서를 생성하는 명령어 그림이다.

```
keytool -genkey -alias ftpkeya -keysize 1024
        -dname "cn=domain.com,o=domain,ou=domain,l=seoul,s=KR,c=KR"
        -keystore mykeystore.jks -keypass ftpkey -storepass ftpkey
        -keyalg RSA
keytool -list -keystore mykeystore.jks -storepass ftpkey -v
keytool -export -alias ftpkeya -keystore mykeystore.jks
        -storepass ftpkey -file serverCertificate.arm -rfc
```

그림 2. keytool을 이용하여 인증서 생성

서버의 인증서 전송이 끝나면 TLS Handshake를 이용하여 키를 교환한 이후에 이루어지는 모든 통신에 대하여 패킷 암호화를 진행한다. 그림 3은 WireShark를 이용하여 응답코드와 데이터가 모두 암호화되어 전송되는 것을 캡쳐한 그림이다.

```
0000  88 36 6c 9e f3 98 e0 d5  5e 49 77 1e 08 00 45 00   ·61····· ^Iw···E·
0010  00 52 1e 05 40 00 80 06  00 00 c0 a8 00 02 dd 9e   ·R··@··· ········
0020  ce 99 28 9d 03 de f1 85  cb fd 29 2e 08 d1 50 18   ··(····· ··)···P·
0030  08 00 6d 27 00 00 17 03  03 00 25 00 00 00 00 00   ··m'···· ··%·····
0040  00 00 01 3a b9 3f f5 37  c6 bd 88 6d f8 4b a0 18   ···:·?·7 ···m·K··
0050  86 a1 3e 3a 05 73 95 01  53 97 ac 5c 94 b8 aa a7   ··>:·s·· S··\····
```

그림 3. TLS로 암호화 되어 전송되는 패킷

3.2 OTP 구현

시간 기반 일회용 비밀번호 알고리즘(TOTP)과 HMAC 기반 일회용 비밀번호 알고리즘(HOTP)를 사용하여 OTP를 구현한다[12,13]. 그림 4는 범용적으로 사용하는 Google社의 Google OTP에서 사용하는 OTP Code 생성 Pseudocode이다[14].

```
function GoogleAuthenticatorCode(string secret)
    key := 5B5E7MMX344QRHYO
    message := floor(current Unix time / 30)
    hash := HMAC-SHA1(key, message)
    offset := last nibble of hash
    truncatedHash := hash[offset..offset+3]
    Set the first bit of truncatedHash to zero
    code := truncatedHash mod 1000000
    pad code with 0 from the left until length of code is 6
    return code
```

그림 4. OTP Code 생성 Pseudocode

기존의 FTP의 로그인은 ID와 Password을 이용하여 로그인 하지만, OTP를 이용한 로그인에서는 ID와 OTP Code(Password)를 입력한다. 그림 5는 OTP를 이용하여 사용자 인증을 보여주는 그림이다.

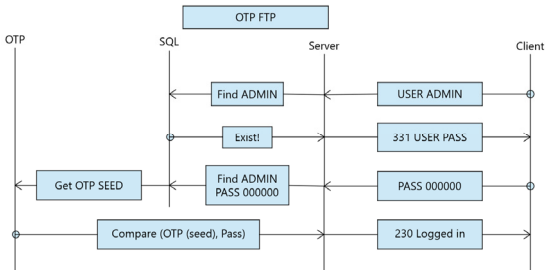

그림 5. OTP를 이용한 FTP 사용자 인증

Ⅳ. 실 험

FTP, FTPS의 성능을 비교하기 위하여 2대의 PC를 사용하였으며 표 1은 PC 각각의 사양을 보여준다.

표 1. PC 사양

	서버	클라이언트
운영체제	Ubuntu 18.04	WINDOWS 10
CPU	i5-8400	E8400
메모리	12GB	4GB
연결 속도	500Mbps	

위와 같은 사양으로 FTP, FTPS의 500MB의 파일을 다운로드 받을 때의 평균 시간을 측정하였으며, 클라이언트에선 범용 FTP 클라이언트인 FileZilla社의 FileZilla Ftp Client 3.36.0을 사용하였다. 그림 6은 클라이언트가 데이터를 다운로드 하는데 소요된 평균시간을 나타낸 것이다.

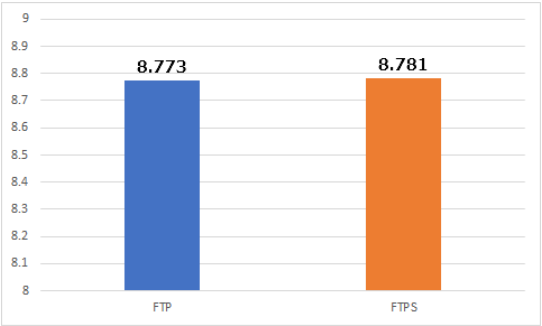

그림 6. FTP와 FTPS의 평균 소요시간(초)

FTPS에선 암호화를 진행해야 하므로 평문으로 전송되는 FTP보다 느린 것은 사실이지만, 유의미한 차이는 발생하지 않았다.

OTP는 FTP에서의 정적 비밀번호 대신에 사용되었으며 그림 7은 Google社의 Google OTP와 연동되어 OTP코드가 생성된 그림이다.

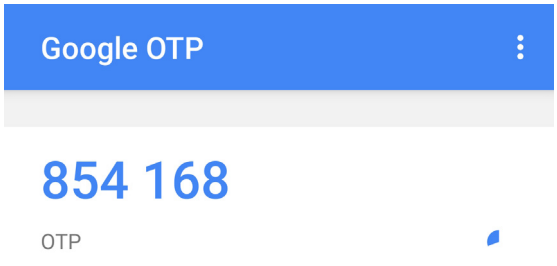

그림 7. Google社의 Google OTP

그림 8은 생성된 OTP를 범용 FTP 클라이언트인 FileZilla社의 FileZilla Ftp Client 3.36.0를 이용하여 서버에 접속한 그림이다.

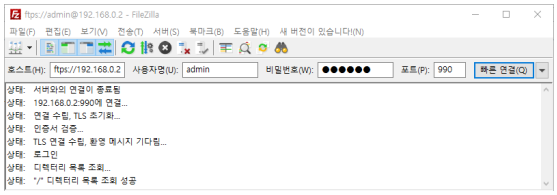

그림 8. TLS와 OTP를 이용하여 서버 접속

OTP Application과 FTP 클라이언트 모두 범용적으로 사용하는 것이므로 사용자가 새로운 것에 적용할 필요 없이도 사용 할 수 있다는 장점이 있다.

Ⅴ. 결 론

본 논문은 FTP가 평문으로 전달되어 일어날 수 있는 MITM(Bounce Attack), 로그인 정보 탈취·도용과 같은 보안문제를 보완하고자 하였다. TLS만 사용하면 발생할 수 있는 Brute-Force, Password 유출과 같은 문제와 OTP만 사용하면 발생할 수 있는 MITM(Bounce Attack), Sniffing과 같은 문제점들을 TLS와 OTP를 동시에 사용함으로써 상쇄할 수 있다. 가장 큰 장점은 흔히 사용되는 사용자 클라이언트를 사용하여 사용자의 변화 없이도 많은 보안 문제들을 해결 할 수 있다는 점이다.

References

[1] M. Horowitz, Cygnus Solutions, S. Lunt, Bellcore, “FTP Security Extensions”, RFC 2228, IETF, Oct. 1997.

[2] Mitcheal stahnke, “Pro OpenSSH”, Apress, p.84, 2006.

[3] AntiPhishingGroup, Phsing Activity Trends Report [Internet]. Available : http://antiphishing.org

[4] Carnegie Mellon University, *2002 Tech Tip,* Massachusetts, MA: CERT Division, 2002.

[5] P. Ford-Hutchinson, IBM UK Ltd, "Securing FTP with TLS", RFC 4217, IETF, Oct. 2005.

[6] K. Y. Kim, "A Study on the Authentication System Based on the OTP", *Korea Institute Of Information Security And Cryptology*, Vol. 17, No. 3, pp. 26-31, Jun. 2007.

[7] M. Allman, NASA Glenn/Sterling Software, S. Ostermann, Ohio University, "FTP Security Considerations", RFC 2577, IETF, May. 1999.

[8] B. T. Kang, "A Study on the Vulnerability of OTP Application Environment Using MITM Attack", Korea, Korea University Library, 2016

[9] Wei Chi KU, Hao Chuan TSHI, Maw Jinn TSUAR, "Stolen verifier attack on an efficient smartcard based one time password authentication scheme", IEICE Transactions on Communication, vol.E87-B, pp 2734-2376, 2005.

[10] P. Hallam-Baker, Comodo Group Inc., "X.509v3 Tansport Layer Security (TLS) Feature Extension", RFC 7633, IETF, Oct. 2015.

[11] Oracle, Sun Java System Application Server 9.1 Management Manual [Internet]. Available : https://docs.oracle.com/cd/E19159-01/820-4605

[12] D. M'Raihi, VeriSign, M. Bellare, UCSD, F. Hoornaert, Vasco, D. Naccache, Gemplus, O. Ranen, Aladdin, "HOTP: An HMAC-Based One ‐Time Password Algorithm", RFC4226, IETF, Dec. 2005.

[13] D. M'Raihi, Verisign Inc., S. Machani, Diversinet Corp., M. Pei, Symantec, J. Rydell, Protwise Inc., "TOTP: Time-Based One-Time Password Algorithm", RFC 6238, IETF, May. 2011.

[14] Wikipedia, Google Authenticator [Internet]. Available : https://en.wikipedia.org/wiki/Google_Authenticator