Pertanika J. Sci. & Technol. 25 (S): 303 - 314 (2017)

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

A Pre-shared Diffie-Hellman Key Exchange Scheme for a Secure TFTP Protocol

Nur Nabila Mohamed*, Mohd Anuar Mat Isa, Yusnani Mohd Yussoff and Habibah Hashim

Faculty of Electrical Engineering, Universiti Teknologi MARA (UiTM), 40450 Shah Alam, Selangor, Malaysia

ABSTRACT

Implementation of Trivial File Transfer Protocol (TFTP) in embedded system is significant because of the speed and simplicity. However, no security service in TFTP marks its major limitations. In this work, a pre-shared Diffie Hellman Key Exchange (DHKE) technique was proposed for mutual authentication to achieve the same secret key in TFTP communication protocol. We also integrated the system with feasible compression and encryption process to significantly improve the TFTP communication performance. The DHKE proof of concept is discussed briefly to show the feasibility of the pre-shared technique on the protocol. Also, the experiment was performed on constrained embedded devices to analyse the performance of compression/encryption scheme in TFTP. From the results obtained, the combined encryption and compression process is able to reduce the time by about 30% compared to the original file transmission time. Thus, the proposed work presents both advantages to reduce file size and provide security for the data. This is a preliminary work to provide a secure TFTP communication which has benefits to be implemented in embedded system field and IoT services.

Keywords: Advanced encryption standard, Diffie Hellman Key Exchange (DHKE), Huffman coding, Trivial File Transfer Protocol (TFTP)

INTRODUCTION

TFTP is a multi-purpose protocol with simple functions used in many applications such as uploading/downloading images, upgrading system, booting nodes, etc. Nevertheless, its implementation in certain situations exposes the system to risks of cyber-attacks as it provides no security mechanism. For instance, during software updates of the Access Point (AP) remote host using Trivial

ARTICLE INFO

Article history:
Received: 25 October 2016
Accepted: 17 March 2017

E-mail addresses:
nurnabilamohamed@gmail.com (Nur Nabila Mohamed),
anuarls@hotmail.com (Mohd Anuar Mat Isa),
yusna233@salam.uitm.edu.my (Yusnani Mohd Yussoff),
habib350@salam.uitm.edu.my (Habibah Hashim)
*Corresponding Author

ISSN: 0128-7680

File Transfer Protocol (TFTP) (Isa et al., 2012), an unauthorised user can disturb the system by installing malicious codes such as a backdoor in the boot loader, kernel, or application to hack the private information linked with the hijacked AP. Furthermore, nowadays, TFTP has been used in embedded field (IoT, embedded multi-agent system), which presents the need to solve the security problem in this protocol. Thus, in this work, we proposed integrating TFTP with a key exchange protocol for secret key exchange and implementing combined encryption and compression scheme for data protection. We believe that this proposed protocol can be a significant enhancement in pervasive computing and IoT applications for managing and upgrading embedded infrastructure.

The remainder of this paper is structured into several sections: Section 2 discusses the related work; Section 3 explains the proposed pre-shared DHKE scheme; Section 4 addresses the proposed methodology, while Section 5 presents the results and discussions of the implementation. The last section concludes the work and briefly discusses several future enhancements of secure TFTP protocol.

Related Works

Trivial File Transfer Protocol. Due to insufficient storage and complexity issues in File Transfer Protocol (FTP), a faster and lightweight version (TFTP) was introduced in 1970s for configuring files and updating devices quickly and easily. However, its implementation is limited to allowing only conventional operations such as reading and writing to files. It has been noticed that in the current implementations of TFTP such as in IoT, embedded multi-agent system, etc., it might be possible for any machine to interrupt the system as TFTP does not require any authentication and directory permission checks (Bollins, 1992). In RFC 3617 (Lear, 2003), it is clearly declared that TFTP provides no mechanism for authentication within the protocol, and this protocol is vulnerable to Man-In-The-Middle (MITM) attack. Furthermore, recent research works have addressed the potential usage of this lightweight protocol for Radio Frequency (RF) (Kao et al., 2010) and remote attestation for Trusted Computing (Schiffman et al., 2011) such as Trusted Platform Modules (TPM) which have revealed the importance of security enhancements for TFTP. Hence, we believe that security integration to this protocol will bring huge advantages in the field of embedded systems.

Diffie Hellman Key Exchange Protocol (DHKE). DHKE algorithm was introduced by Diffie and Hellman in 1976 to securely exchange secret parameters over insecure channels. Several studies regarding the DHKE implementation method have proven that this protocol enables the communicating users to jointly establish a shared secret over the public channel (Li, 2010; Carts, 2001). However, one of its limitations is the vulnerability to MITM attack, where the unauthorised party attempts to intercept and obtain the information passing through insecure channel. For example, Alice and Bob want to share their shared secret key in a public medium.

Eve, the Man-In-The-Middle, comes to intercept by performing her own public parameters to alter their communication. The important parameters in DHKE protocol are listed below:

A	Alice's public value, known to Alice, Bob and Eve
A'	Eve's public value sent to Bob
a	Alice's secret value, known only to Alice
B	Bob's public value, known to Alice, Bob and Eve
B'	Eve's public value sent to Alice
b	Bob's secret value, known only to Bob
g	Primitive root modulo p, known to public
p	Large prime numbers, known to public

MITM attack is explained based on the following steps:

- Alice and Bob agree on p=97 and g=5 values, in which p is a big prime number and g is the generator of p.
- Alice chooses her private value, a=36 and computes her public value, A=50. She then sends A to Bob which is intercepted by Eve.
- Eve obtains the values 97(p), 5(g), and A=50 through improper means, she computes a new public value, A'=13 using her secret 25. She sends A'=13 to Bob.
- On the other side, Bob chooses his private value, b=58, and computes B = 44. He then sends B to Alice, which is also intercepted by Eve.
- Eve creates a new secret, 30, computes B'=79, and then sends to Alice.
- Alice believes that B' is received from Bob, so she proceeds to compute the secret key *'K'* as $K = B^a \bmod p = 22$.
- Bob understands that A = 13 is received from Alice, and thus computes the secret key, 'K' as $K = A^b \bmod p = 53$.
- Eve who has intercepted and modified the messages will compute two keys, one of which is to be shared with Alice, and the other one with Bob.
- For this, she computes the key to be shared with Alice as $K = 5^{30} \bmod 97 = 22$, and the key to be shared with Bob as $K = 44^{25} \bmod 97 = 53$.

Once the attack is done, Eve can decrypt the key she has exchanged between Alice and Bob and compromise the entire communication. Thus, the original DHKE protocol has no means to mitigate MITM attacks.

Therefore, we believe that the proposed pre-shared concept in DHKE is one of the simplest security solutions for securing the communication from MITM attack in the TFTP communication protocol. The pre-shared concept enables both parties to have knowledge of several parameters, and only one side needs to send the public value to the other side. The pre-shared method will be explained in Section 3. Meanwhile, the next subsection will address the need for compression and encryption in secure communication. A secure key exchange technique ensures that associated encryption and decryption keys are safely exchanged between the communicating parties.

Compression and Encryption Approach. In the situation when dealing with a large amount of data, compression scheme is required to reduce the size; nevertheless, the scheme alone is not sufficient as it is still accessible and open to public. Therefore, an employment of joint compression and encryption process enable faster and secured data transmission. Several related works have addressed the necessity to combine the compression and encryption approach for efficient data transmission. For instance, Singh and Manimegalai (2012) found that symmetric encryption is an efficient encryption mechanism to protect the compressed data. Also, the related work by Razzaque (2012) revealed that the execution of compression technique, followed by the encryption technique, is much more preferable as an intruder has less cleave to obtain data information when using this sequence. Meanwhile, Chaudhari (2013) proposed performing the compression technique using two types of lossless compression algorithms, followed by encrypting data using symmetric encryption (DES). Nevertheless, the author did not elaborate on the key exchange process in the system. According to the related works cited above, we believe that symmetric encryption algorithms such as AES or DES are compatible to be integrated with compression coding; thus, we integrate the proposed idea with minimal compression and encryption process to significantly reduce the computational requirements in TFTP communication.

Related Works in Securing TFTP. Referring to some previous works on TFTP (see Kao et al., 2010; Schiffman et al., 2011), we realised the importance of providing security mechanism in the protocol to prevent various attacks that will be used in low computational power devices. Nevertheless, there are very few studies in the field of secure TFTP protocol. So far, we have studied two recent works elaborating on the importance of securing TFTP. Horvat, Žagar and Martinović (2013) improved the security in TFTP protocol by introducing a security extension to the general TFTP using Digest Access Authentication (DAA), which was accompanied by hash function SHA1 for authentication credibility. The authors proposed two data ciphers to establish data confidentiality, XTEA and AES, because these schemes are considered secure for all known attacks. The work, however, does not explain the key exchange protocol for secret key generation, and the results on generating nonce values for TFTP protocol have also not been discussed.

In our previous works (Mohamed et al., 2013; Mohamed, Hashim & Yussoff, 2014), the framework of the proposed secure TFTP was presented, and a preliminary experiment on securing TFTP packet was done by analysing the execution and transmission time of encrypting and decrypting data. The security framework in TFTP introduced by Isa et al. (2012) provides security negotiation such as attestation before file transmission process. The author proposed a new packet header in the existing TFTP option extension using two cryptographic principles, symmetric (AES algorithm) and asymmetric encryption (DHKE concept). This work is considered as a preliminary step on securing TFTP packet but its implementation and integrity properties still have not been discussed to ensure the encrypted data payload is protected from adversary.

Based on two major related works (Isa et al., 2012; Horvat, 2013) on securing the TFTP protocol, we believe that the proposed scheme can provide considerable security requirement in TFTP communication.

Pre-shared DHKE Scheme. The MITM attack usually happens because DHKE does not authenticate the communicating participants. It enables the adversary to interrupt the communication by replacing his/her own public value to both communicating parties. When the client sends his public value to server, the Man-In-The-Middle intercepts it by sending its own value to either client, or server. Once they agree on the shared parameters exchange, the intruder can use the key to decrypt any messages sent out by the client, or server, and he/she is able to modify the information before transmitting it to the other side. Thus, the implementation of DHKE pre-shared public parameters has been considered as one of the reliable techniques to mitigate MITM attacks and it can be implemented in the TFTP file transmission protocol as only the server side shares the public value.

In this work, a network packet analyser called Wireshark was utilised to capture packet sent between client and server. This was done to show that if the Man-In-The-Middle tried to interrupt the communication between client and server, he/she could only get or modify the public value B, but even this would not affect the security of the whole communication system. Figure 1 shows that the intruder may read and change the server's public value, which is B, but this will not compromise the communication since both communicating parties already have knowledge of one of the public values (A). Hence, we believe that the proposed pre-shared technique is considerably secure from MITM because the adversary would have been unable to obtain the true secret value and this has become the strength in this protocol as well. Furthermore, in TFTP, the client sends the RRQ together with the server's IP address. Even after the intruder executes some modifications, the client still knows that the value received is wrong because after computing the secret key, it would fail to decrypt the data. Thus, the client would need to send another request to the server.

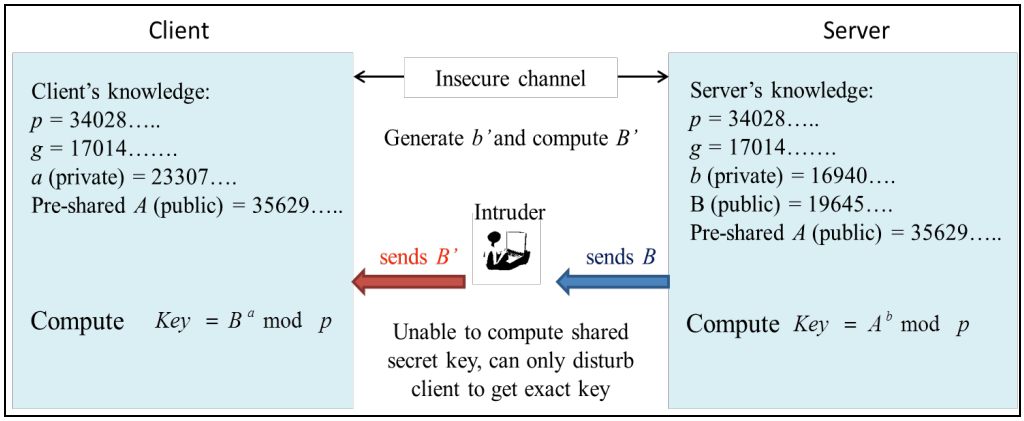

Figure 1. DHKE Pre-shared public parameters

METHOD

The Proposed Method

The experiment was set up using two microprocessors (Raspberry Pi) acting as both the client and server. The devices were placed in a distance about a meter connecting via wireless using Wi Pi dongle. Besides, additional software, OS Linux Raspbian 6 (Wheezy), had also been installed on each workstation. The public parameters pre-shared between the communicating devices (p, g and A) were established during the initial communication. The proposed key exchange concept to share the public parameters in wireless TFTP communication is illustrated in Figure 2 below. It starts with the client sending the RRQ packet to get server's public value, B. The RRQ packet sent contains opcode 1 and the filename field contains RRQ of Public Key, B. The server acknowledges the request by sending the DATA packet containing a 3 in the opcode field, the block# starts with 1 and DATA field contains the value B. The client approves by sending acknowledgment ACK packet containing opcode 4 and block# 1. The file containing payload less than 512 bytes signs the end of data transfer. In this work, the prime value for p is large and hard enough to be guessed in order to be secured against attacks. Besides, it is also important to ensure that the public value B should always be different in every transmission to prevent the adversary from computing the key easily.

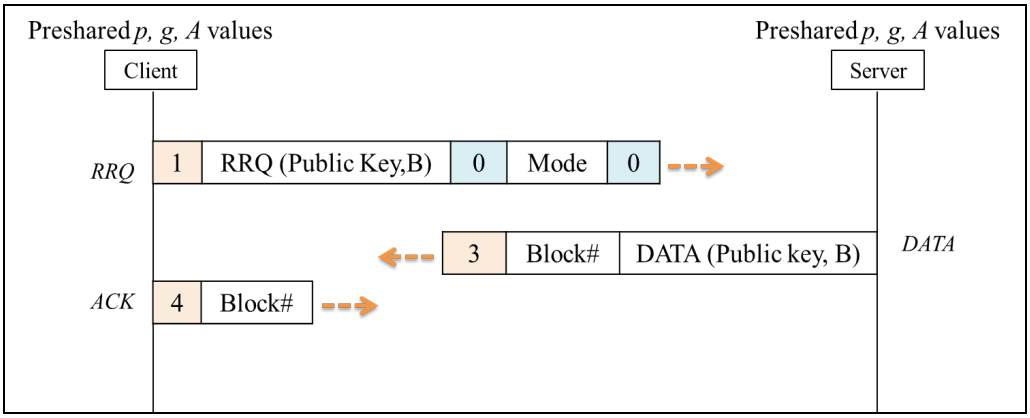

Figure 2. The proposed work in TFTP

In order to achieve time minimisation and data confidentiality in this work, Huffman coding compression algorithm and AES encryption algorithm were selected based on the related works by several researchers. A sample original file of various sizes was processed in three ways. First, it was simply encrypted using AES encryption scheme; secondly it was simply processed using the Huffman coding. Finally, the file was compressed and then encrypted, respectively. Both the results and discussions are given in the next section based on the performance of the proposed technique in TFTP communication.

RESULTS AND DISCUSSION

Experimental Results for Variation in File Size after Processing

Once the exchange was done and the secret key to be used in symmetric encryption and decryption process was obtained, the compression/security scheme technique was performed. The following equation was then used to compute the file size variation in kiloBytes (kB) after the compressed and encrypted process.

$$Variation\ of\ File\ Size = \frac{Size\ after\ Compression/Encryption}{Size\ before\ Compression/Encryption} \quad [1]$$

Table 1 shows the different variations in the file size of four file types (original file, compressed file, encrypted file, and combined compressed-encrypted file). Based on the results given below, the percentage average of compressed file is about 28.8%, which has reduced about 2/3 of the original file size. In contrast, an increase in the file size was recorded for the encrypted file due to the insertion of extra bytes of padding during encryption process. However, it can be seen that the average variation for combined scheme is about 29.3%, which is almost similar to the result obtained with independent compression thus providing good compression ratio, as the difference between them is only 1.5%. This result shows the significance of implementing the combined scheme for the proposed secure TFTP communication protocol.

Table 1
Variation in file sizes after processing

Original file size (kB)	After compression		After encryption		After compression/ encryption		
	kiloByte (kB)	Percentage (%)	kiloByte (kB)	Percentage (%)	kiloByte (kB)	Percentage (%)	
9.22	2.68	0.291	9.53	1.034	2.98	0.323	
51.10	14.74	0.288	51.94	1.016	15.05	0.295	
102.21	29.45	0.288	102.50	1.003	29.75	0.291	
511.05	147.15	0.288	511.35	1.001	147.45	0.289	
1024.00	294.82	0.288	1024.29	1.000	295.13	0.288	
2047.45	289.46	0.288	2047.75	1.000	589.77	0.288	
5119.39	1473.83	0.288	5119.69	1.000	1474.13	0.288	
10239.80	2947.93	0.288	10240.10	1.000	2948.23	0.288	
20479.61	5895.83	0.288	20479.91	1.000	5896.13	0.288	
30719.41	8843.73	0.288	30719.72	1.000	8844.04	0.288	
		28.8		100.5		29.3	Average percentage (%)

Experimental Results for File Transmission Time

The file transmission time was taken three times to ensure that the value is accurate. The transmission time for every scheme was taken to show the comparison of time difference when transmitting the original file and processed files. Based on the data given in Table 2, the transmission time increased when the file size increased. Meanwhile, transmission of the encrypted file took the longest time as the file size increased a few bytes compared to the original. The transmission was done via wireless network configuration. Therefore, it can be seen that the execution time for sending large file takes much more time. The reason is due to several connection problems such as low connection, connection lost, etc. The results were then plotted in the graph shown in Figure 3 for comparison purposes.

Based on Figure 3, the transmission time for the compressed file is the fastest compared to the other schemes. However, it can be seen it is approximately similar with the transmission time when using the combined scheme (compressed-encrypted file). It is shown that the proposed combined process is able to reduce the time by about 30% compared to the original file transmission time. Thus, this can be considered as a significant result along with the fact that it particularly presents both advantages to reduce file size and provide security for the data.

Table 2
File transmission time

Original file transmission										
File size (kB)	9.2	51.1	102.2	511.1	1024.0	2047.5	5119.4	10239.8	20479.6	30719.4
Transmission time (s)	0.6	3.3	6.6	32.5	67.3	141.6	336.9	670.8	1017.6	1642.9
Encrypted file transmission										
File size (kB)	9.5	51.9	102.5	511.4	1024.3	2047.8	5119.7	10240.1	20479.9	30719.7
Transmission time (s)	0.7	3.4	6.8	33.1	68.1	147.1	348.5	698.8	1259.7	1820.6
Compressed file transmission										
File size (kB)	2.7	14.7	29.5	147.2	294.8	589.5	1473.8	2947.9	5895.8	8843.7
Transmission time (s)	0.24	1.6	2.7	10.5	28.7	47.5	155.1	298.9	659.3	1019.3
Compressed/encrypted file transmission										
File size (kB)	3.0	15.1	29.8	147.5	295.1	589.8	1474.1	2948.2	5896.1	8844.0
Transmission time (s)	0.2	1.9	3.1	15.8	29.3	65.9	166.3	371.0	735.4	1169.9

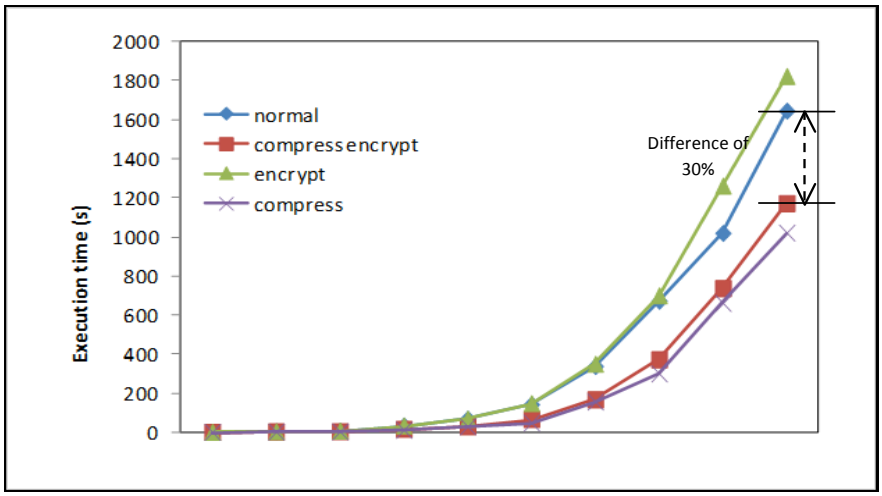

Figure 3. The TFTP file transmission time

CONCLUSION

In conclusion, this work has introduced the key exchange concept using the pre-shared technique to secure communication in the TFTP protocol. The idea of pre-shared technique was proposed to mitigate the MITM attack, which is one of the security vulnerabilities in the DHKE protocol. It has been explained in the result and discussion section that the pre-shared

DHKE provides increased security as one of the communicating parties has already known the public value of the other and could use it to compute the secret key using DHKE.

Besides, we also have considered the implementation of file compression coding and cryptographic security mechanism in TFTP using the Huffman compression coding and AES encryption algorithm for an efficient and secure file transmission. From the results, it can be seen that the time difference when using the proposed combined process is about 30% compared to the original file transmission time. Thus, this work has presented both advantages, which are reducing file size and providing security for the data. The proposed idea of pre-shared key exchange concepts in TFTP shows a significant contribution of this work, which can enhance security while transferring data in wireless AP or IoT system to prevent any MITM attack.

ACKNOWLEDGEMENTS

The authors would like to thank the Ministry of Higher Education (MOHE) for providing the NRGS grant 600-RMI/NRGS 5/3 (5/2013) in the project entitled, Algebraic-Structure-Based Lightweight Security Technique, Research Management Institute (RMI), UiTM, and also members of Computer Engineering research, Faculty of Electrical Engineering, UiTM Shah Alam, for supporting the research project reported in this paper.

REFERENCES

Bollins, K. (1992). The TFTP Protocol (Revision 2) RFC 1350. *Official Protocol Standard*, 1–11.

Carts, D. (2001). A Review of the Diffie-Hellman Algorithm and its Use in Secure Internet Protocols. *SANS Institute Infosec Reading Room*, 1-9.

Chaudhari, M. (2013). Fast and Secure Data Transmission using Symmetric Encryption and Lossless Compression. *International Journal of Computer Science and Information Technology, 2*(2), 58–63.

Horvat, G., Žagar, D., & Martinović, G. (2013). STFTP: Secure TFTP protocol for embedded multi-agent systems communication. *Advanced in Electrical and Computer Engineering, 13*(2), 23–32.

Isa, M. A. M., Mohamed, N. N., Hashim, H., S. Adnan, F. S., Manan, J. L. A., & Mahmod, R. (2012). A lightweight and secure TFTP protocol for smart environment. *IEEE Symposium on Computer Applications and Industrial Electronics*, 302–306.

Kao, K. F., Liao, I. E., & Lyu, J. S. (2010). An indoor location-based service using access points as signal strength data collectors. *International Conference on Indoor Positioning and Indoor Navigation*, 15–17.

Lear, E. (2003). Uniform Resource Identifier (URI) Scheme and Applicability Statement for the Trivial File Transfer Protocol (TFTP). *Request for Comment, 3617*, 1-7.

Li, N. (2010). Research on Diffie-Hellman key exchange protocol. *2nd International Conference of Computer Engineering Technology*, 634–637.

Mohamed, N. N., Hashim, H., & Yussoff, Y. M. (2014). Compression and Encryption Technique on Securing TFTP Packet. *IEEE Symposium on Computer Applications and Industrial Electronics (ISCAIE)*, 198–202.

Mohamed, N. N., Hashim, H., Yussoff, Y. M. & Isa, M. A. M. (2013). Securing TFTP Packet: A Preliminary Study. *IEEE Control and System Graduate Research Colloquium*, 158–161.

Razzaque, A. (2012). Image Compression and Encryption: An Overview. *International Journal of Engineering Research and Technology, 1*(5), 1–7.

Schiffman, J., Moyer, T., Jaeger, T., & McDaniel, P. (2011). Network-based root of trust for installation. *IEEE Security and Privacy, 9*, 40–48.

Singh, K. J., & Manimegalai, R. (2012). A Survey on Joint Compression and Encryption Techniques for Video Data School of Information Technology and Engineering. *Journal of Computer Science, 8*(5), 731–736.