Authentication of Diffie-Hellman Protocol Against Man-in-the-Middle Attack Using Cryptographically Secure CRC

Nazmun Naher, Asaduzzaman and Md. Mokammel Haque

Abstract Diffie-Hellman key exchange (DHKE) protocol, which is also known as exponential key exchange protocol, is one of the practical ways of generating a common secret key between two communicating parties. But this protocol itself is a non-authenticated protocol; hence, the protocol is exposed to man-in-the-middle (MITM) attack. An attacker can easily hijack sender's public value. Attacker calculates his own public value and sends this value to the receiver instead of sending the original value. Attacker does the same thing when receiver replies back to the sender. After this exchange, attacker can decrypt any messages sent by both of the communicating parties. In this paper, a simple authentication mechanism is developed based on the cryptographically secure version of well-known cyclic redundancy check (CRC). A cryptographically secure CRC is capable of detecting both random and malicious errors where the CRC divisor polynomial is randomly generated and secret. A common CRC divisor polynomial is generated for both of the communicating parties. The system is capable of generating cryptographically secure random numbers which are different in every session. Here the length of the divisor polynomial for CRC must be large. In our proposed system, cryptographically secure CRC is combined with the Diffie-Hellman algorithm for checking whether the public value of the sender is changed by an attacker. MITM attack is detected successfully by using only one securely and randomly generated secret nonzero divisor polynomial of cryptographically secure CRC. The length of public keys to be sent in the Diffie-Hellman protocol and modified system are also compared to show the overhead is negligible.

N. Naher (✉) · Asaduzzaman · Md. M. Haque
Department of Computer Science and Engineering, Chittagong University of Engineering and Technology, Chittagong 4349, Bangladesh
e-mail: nazmunsonia403@gmail.com

Asaduzzaman
e-mail: asad@cuet.ac.bd

Md. M. Haque
e-mail: mokammel@cuet.ac.bd

M. Chakraborty et al. (eds.), *Proceedings of International Ethical Hacking Conference 2018*, Advances in Intelligent Systems and Computing 811,
https://doi.org/10.1007/978-981-13-1544-2_12

Keywords Man-in-the-middle attack · Message authentication · Diffie-Hellman key exchange protocol · Cryptographically secure cyclic redundancy check (CRC)

1 Introduction

In the present era of high technology, environment network security is an important aspect of system admiration. Without having to be physically present, the network allows people to access geographically distant resources remotely. Hence, there can happen many unpredictable incidents, i.e., hacking, information loss, unauthorized access, and misuse. Cryptology is the study of designing a system which will ensure to keep the four aspects of modern cryptology such as confidentiality, integrity, authentication, and non-repudiation of the information in an organized and systematic way. It is mainly used to protect information that is sent over a channel which is insecure. Cryptography can be divided into two sections depending on the nature of key used in the symmetric key cryptography and asymmetric key cryptography. In a symmetric key cryptosystem, same key is used by both sender and receiver for encryption and decryption, respectively. On the other hand, in an asymmetric key cryptosystem, two different keys are used for encryption and decryption mechanism. Symmetric key cryptography is generally very fast and ideal for encrypting a large number of data. Security of the symmetric key encryption depends on the key exchange protocol. In 1976, two researchers at Stanford University, Diffie and Hellman presented a key exchange protocol. This protocol can proceed over public communication channels [1]. Diffie-Hellman (DH) protocol is still extensively used in the present days. But one of the major problems of this protocol is that it is not self-authenticated, so man-in-the-middle attack can be possible in DH protocol. In this paper, some currently available solutions are described in Sect. 2 with some of their shortcomings. But we proposed a system totally in a different approach to provide authentication in a simple manner but effectively which will prevent MITM attack on DH protocol.

Cyclic redundancy check is extensively used as a safeguard of data in transmission channels. It prevents errors occurred randomly in the communication channel [2]. The basis of our proposed system is the use of division modulo a random and secret polynomial over GF (2) for the purpose of authentication of the protocol. Cyclic redundancy check (CRC), which is most of the time used as a random error detection mechanism in the communication channel, is cryptographically varied here. A satisfactory level of security without losing reliability can be guaranteed, if the conventional CRC is made cryptographically secure. The main concept is to make the generator polynomial of CRC variable and secret [3, 4]. It is also computationally very simple and secure than other hash function when the simple CRC is converted to cryptographically secure CRC.

In this paper, Sect. 2 discusses the related works on this problem. Section 3 represents the proposed system, and Sect. 4 explains the experimental results and their explanation. This section also includes secrecy analysis of the proposed system and comparison of the existing and modified proposed system. And finally, Sect. 5 represents the conclusion and future works.

2 Related Works

In 1976, Diffie and Hellman proposed their algorithm for symmetric key encryption system to exchange shared key between two communicating parties. But their system is not self-authenticated, so it is possible to attack both of the communicating parties by an attacker [1]. Here we will discuss some existing solution to prevent this attack. In the study [5], they proposed a secure system against MITM attack based on Geffe's generation of binary sequences and server to handle the communication. But server has to handle all user tables with a large number of entities, and both of the communicating parties have to send their private key to the server. In this study [5], they also discussed some problem of other authentication systems.

In another study [6], a biometric-based sender authentication system was developed using speech. But speech can be changed by acoustic surrounding and transduction device such as microphone.

2.1 Diffie-Hellman Key Exchange Protocol

Diffie-Hellman key exchange (DHKE) is one of the primitive concepts of public key cryptosystem. This protocol allows two communicating parties to share a common secret key over insecure communication mediums without meeting in advance. The process of this protocol supposes that Alice and Bob have different private keys, and they have to agree upon two relatively prime numbers p, g, and then each of them uses the obtained information to calculate the public keys. As a result, both of Alice and Bob obtained the shared key without sending their private keys through the channel [7]. Steps in Diffie-Hellman key exchange protocol are described below:

- Alice and Bob have to agree on two large numbers p and g, where p is a prime number, and p and g are public.
- Alice picks a large number a and keeps it secret; similarly Bob picks his secret key b.
- Alice sends a message to Bob containing (p, g, g^a mod p).
- Bob sends a message to Alice containing (p, g, g^b mod p).
- After receiving Bob's message, Alice performs the following computation:

  (g^b mod p)^a mod p, which yields g^{ab} mod p.

- After receiving Alice's message, Bob performs the same computation:

  (g^a mod p)^b mod p, which yields g^{ab} mod p.

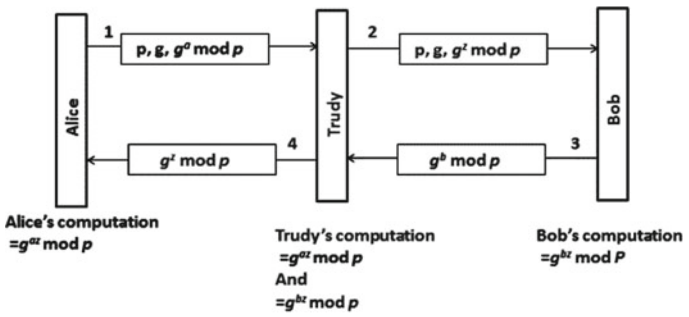

Fig. 1 Man-in-the-middle attack scenario [8]

Hence, the secret key of Alice and Bob is g^{ab} mod p, using which they will communicate securely [8].

The Diffie-Hellman key exchange protocol is exposed to man-in-the-middle (MITM) attack. An attacker can interpose Alice's public value and send his own calculated public value to Bob. When Bob replies back his public value, attacker again replaces it with his own public value and sends it to Alice. Attacker and Alice thus agree on one shared secret key. Attacker also agrees on another shared key with Bob. At the end of the session, any messages sent by Alice or Bob can be seen and modified by the attacker before encrypting them again with the key and sending them to the receiver. This attack is possible because any of the communicating parties are not authenticated in DH protocol. A man-in-the-middle attack can succeed only if a middleman or attacker can intercept each of the parties to their satisfaction as expected from the legitimate parties (see Fig. 1).

2.2 *Cryptographically Secure CRC*

Authentication can be done simply by using cryptographically secure CRC. The concept of cryptographically secure CRC is to use secret and random CRC generator polynomial. Steps involved in the sender side of cryptographically secure CRCs are as follows:

- Firstly, choose a message M(x) of m bit.
- Randomly generate the polynomial q(x) from a set. This set consists of all possible polynomials which has degree n over GF(2), where q(x) must be a nonzero polynomial, i.e., n > 1.
- Then M(x) is multiplied by x^n. Here n is the degree of generator polynomial q(x). Then the checksum is calculated and appended to the message (i.e., CRC (M)) for sending to the other end.

The CRC decoding or the receiver side check steps are as follows:

- Get the secure random polynomial q(x) form the server which is same as the sender's polynomial.
- Divide the message CRC(M) modulo q(x), where CRC(M) is the received message and q(x) is generator polynomial.
- Finally, the coefficients of the resulting remainder and the received CRC check bits are compared.

Any mismatch in the receiving side points out the occurrence of an error [9].

3 Proposed System

The fundamental target of the system proposed is to authenticate the sender of Diffie-Hellman protocol, so that the attacker's interception will be failed. Attacker could not be able to change the public value of any parties without being detected.

Here the server performs as a trusted third party. Alice and Bob are two communicating parties. The main concept of the server is hired from trusted third party (TTP). In the concept of TTP, both of the communicating parties use this trust to make a secure interaction between them [10]. TTP are common in most of the commercial transactions and in cryptographic transactions as well as cryptographic protocols. Steps in the modified system are described as follows:

- A trusted server and two clients are implemented who are capable of generating secure random number.
- Alice requests a session key from the server. The request message involves two information: Who is the sender and who is the receiver. They have identified the sender's and receiver's ID which is autogenerated after connecting to the server.
- Server then runs its generation algorithm to get the secure random and variable length nonzero generator polynomial. Server then replies to Alice with n-bit generator polynomial.
- Server also sends a message to Bob which consists of two parts: A token and n-bit generator polynomial. Token has also two parts: the sender id and the receiver id.
- Then Alice will run her generation algorithm gen() which will generate a large prime number p and another prime number g (relative prime of p). g is also chosen randomly. This generation algorithm also generates a random number a, which is the secret key for Alice. Generated p, g and a by the generation algorithm p, ganda are random and different in every session. Then Alice calculates the following things:

$$A = g^a mod\ p \tag{1}$$

$$r = A \cdot x^n \tag{2}$$

$$z = r\ mod\ q \tag{3}$$

$$CRC(A) = r \oplus z \tag{4}$$

- Alice sends a message to Bob for start communication which is in the format (p, g, CRC(A)) where p and g are the generated prime and relative prime, respectively. And CRC(A) is the calculated CRC value of the public key A.
- Now, Bob runs his generation algorithm which generates a random large secret key b, which less than p.
- Bob calculates the following things:

$$B = g^b mod\ p \tag{5}$$

$$r = B \cdot x^n \tag{6}$$

$$z = B\ mod\ q(x) \tag{7}$$

$$CRC(A) = r \oplus z \tag{8}$$

- Bob replies back to Alice with a message in format CRC(B). Here CRC(B) is the CRC of public message B calculated by Bob.
- Alice runs reverse CRC to get the public key of Bob. She divides CRC(B) modulo q(x) and then compares the coefficients of the resulting remainder with the CRC check bits received from the sender. If the result matches, then she will calculate:

$$B = CRC^{-1}(B) \tag{9}$$

$$K = B^a mod\ p \tag{10}$$

- Bob also runs reverse CRC to get the public key received from Alice. He divides the received message CRC(A) modulo q(x) and compares the coefficients of the resulting remainder with the CRC check bits that are received from the sender. If the result matches, then he will calculate:

$$A = CRC^{-1}(A) \tag{11}$$

$$K = A^b mod\ p \tag{12}$$

At the end of the algorithm, both parties gain the same shared secret key without any interception of attacker which is actually as follows:

$$K = g^{ab} mod\ p \tag{13}$$

Using this shared secret key, both of the communicating parties are able to communicate with each other securely (see Fig. 2).

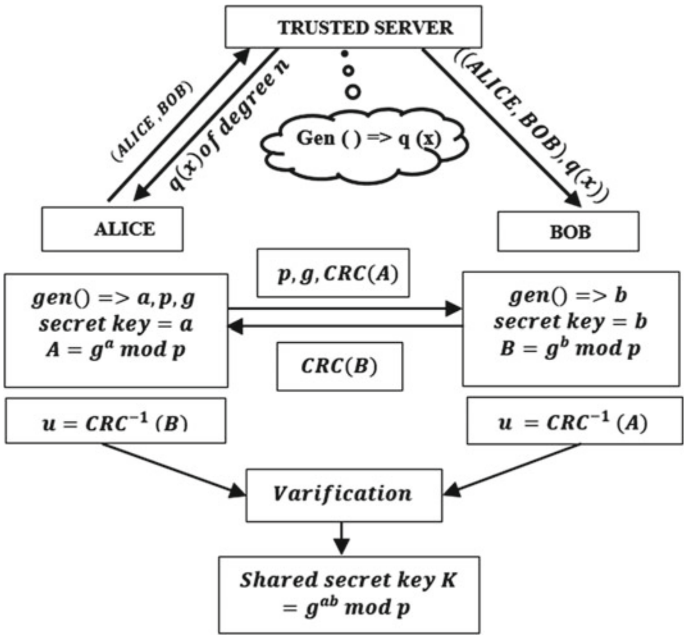

Fig. 2 System architecture

A simple example of the authentication can be represented as follows for simplification:

1. Sender (Client1) → Sends request to the server to communicate with Client2.
2. Server randomly generates 10011001 and sends the number to both clients (Client1 and Client2).
3. Client1 → Generates random secret key, a = 00100000, p = 11101111 and g = 00010111. He also calculates g^a mod p = 0 1 0 0 10 1 1 (7 bits), the CRC value of 01001011 using 10011001 is 0010010110001111 (14 bits) and sends (11101111, 00010111, 0010010110001111) to Client2.
4. Receiver (Client2) → Checks the validity of the received message by CRC check bits. After that, he calculates reverse CRC using the same secret random polynomial 10011001 and retrieve the main public value 0 1 0 0 1 0 1 1.
5. Client2 → Generates random secret key, b = 00011000. He calculates g^b mod p = 0 1 1 0 0 1 0 1 (7 bits), the CRC value of 01100101 using 10011001 is 0011001011111110 (14 bits) and sends 0011001011111110 to Client1.
6. Client1 do the same thing as step 4.

The algorithm of authenticated DH protocol which is proposed has obviously some difference from the non-authenticated version of this protocol. There has been created some variation in our system from Diffie-Hellman algorithm which obviously will

Table 1 A short comparison between DH and proposed scheme

Diffie-Hellman algorithm			Modified system			Impact
Alice	Attacker	Bob	Alice	Attacker	Bob	
a	–	–	a	–	–	1. Man-in-the-middle attack can be prevented 2. Attacker will know nothing about the calculated public key and any of the secret key generated by the server and both of the communicating parties 3. Public key length will be increased
–	–	b	–	–	b	
p	P	p	p	p	p	
g	g	g	g	g	g	
–	–	–	q	–	q	
A	A	A	A, CRC(A)	CRC(A)	A, CRC(A)	
B	B	B	B, CRC(B)	CRC(B)	B, CRC(B)	

Table 2 A short comparison between the length of data to be sent in the DH protocol and modified version (bits)

Generated key length (bits)					Diffie-Hellman public key length (bits)		Modified Diffie-Hellman hashed public key length (bits)	
A	b	p	g	q	Alice	Bob	Alice	Bob
5	3	6	4	6	6	4	11	8
4	5	6	2	8	5	6	12	13
3	3	8	5	8	4	6	11	13
4	2	9	4	7	7	8	13	14
6	5	8	5	8	7	7	14	14
5	4	8	6	4	7	6	10	9
5	5	7	4	8	6	6	13	13
5	4	8	6	7	6	6	12	14

create some impacts and extra overheads on the existing protocol. These variations, impacts, and overheads are shown theoretically in Table 1.

Table 1 shows that here only one overhead is: The length of the data to be transmitted during this protocol will be increased. But it is not a problem here, and actually, it creates no overhead at all because the application of DH protocol is in SSL/TLS (secure socket layer/transport layer security). In SSL/TLS data, unit is called record with maximum payload field or the maximum length of data 14 k or 16,384 bytes. Hence, generated overhead will be negligible here. Public key length of DH and modified DH is compared in Table 2.

The client–server communication will be unicast communication like DHCP (Dynamic Host Configuration Protocol) server. Every client IP will be stored in the server's table. So, there is no need to store any extra entity in the server table which saves both memory and time. And there is no need to share the secret key with the server. Server also does not need to store the generated polynomial by itself. Unicast communication is used for all network processes in which a private or unique resource is requested. The client unicast a request message to the server for generator, as the server IP is known to him. After receiving the request, server will unicast the keys to the client. There is no need to store any secret keys in the server table.

4 Performance Analysis

4.1 Security Analysis

Security of the proposed system depends on the attacker's ability to get the generator polynomial. If he could figure out the secret generator polynomial, then he will be able to change the data. But in our system, we showed that attacker will not be able to get the polynomial because the server will generate a different polynomial in every session. And for brute force search, the probability of finding the polynomial is 2^n, where n is the length of the polynomial (generating capability of the server). So, for secure communication n should be larger, i.e., 80 bit or more, so that the probability will be greater than or equal 2^{80}, which is non-negligible in secrecy analysis of any cryptographic methods.

Attacker fails to get the shared secret key (see Fig. 3). He requested to the server acting like he is Alice, but he is given a secret key which is $q'(x)$ or he randomly generates $q'(x)$ which is different from $q(x)$. Attacker gets CRC(A) from Alice, then he calculates $CRC'(A)$ using his own generated secret key a' and $q'(x)$. He sends $CRC'(A)$ to Bob instead of CRC(A). But Bob can recognize the change because he performs the authentication using $q(x)$, which is same as Alice's calculation. Attacker also replaces CRC(B) by $CRC'(B)$ using a' and $q'(x)$. But none of his changes are succeeded without being detected by using only one secure random polynomial.

The example given before in Sect. 3 can be extended for showing attacker's activity:

1. Attacker changes Client1's message 0010010110001111 by 0001000111101110 using his own secret key 00001010 and polynomial 01010101 because he doesn't know the secret generator polynomial.
2. After that, he sends (11101111, 00010111, and 0001000111101110) to client2.
3. After receiving the message, Client2 can recognize the change because the calculated checksum by 00010111 and 10011001 are not same.
4. Client2 sends an attack notification to Client1 and the server.

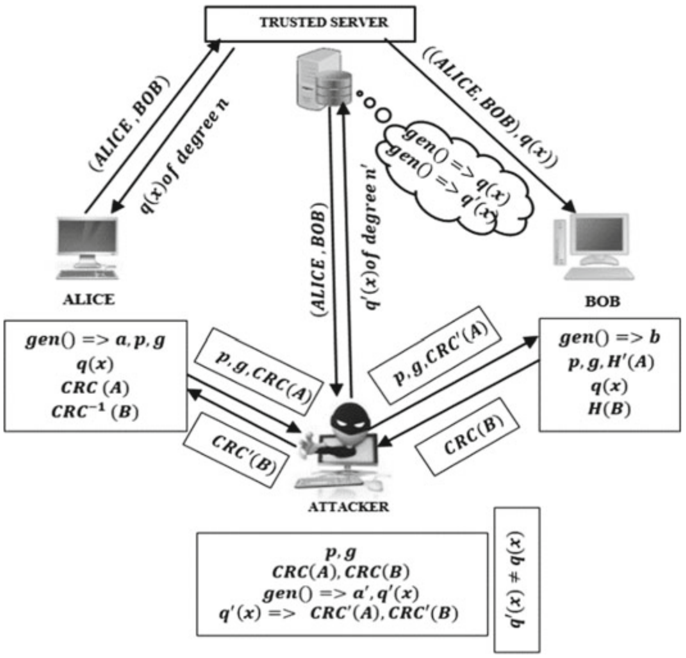

Fig. 3 Security against attack

So, attacker can't change the value of any keys without being caught. There is no option to steal any key from the server because the server need not store any secret or public key of any communicating parties. And it is not possible to get that how much bits are appended after the key because every time the length of the generator polynomial provided by the server is different.

4.2 Comparison of Existing and Modified System

Authentication can be done simply by using cryptographically secure CRC. From the Table 2, we can see that in the modified algorithm public key size is increased depending on the length of CRC polynomial and the public key length in the existing Diffie-Hellman protocol. So it becomes tough to guess by the attacker that what the value of the public key is in the modified algorithm. And here we can see that the appended value after the key is not fixed, it is changed in every session.

So for man-in-the-middle attack, the attacker must have to gain the generator polynomial or guess it. But guessing is not so when the length of the polynomial is larger. And the probability that the attacker can find the polynomial is 2^n where n is the length of the generator polynomial. But the increasing length will not create any overhead in the channel because TLS/SSL where DH protocol is used has a larger unit of data called record which has 16,384 bytes or 14 kB payload field. The length is handled here in bits or some bytes but needs not to be more than 1 kB.

5 Conclusion

In the present era, Diffie-Hellman key exchange protocol is playing a very vital role in information technology and security. It is the most widely used protocol. The Diffie-Hellman algorithm can be efficiently authenticated using cryptographically secure CRC and a trusted server. Implementation of this algorithm is very easy to understand because the implementation of cryptographically secure CRC is most likely as the conventional CRC except the generator polynomial is programmable (random) and secret. And the responsibilities of the server are reduced. Here server's job is to provide only a secure random divisor polynomial of CRC which is different in every session but not to store them. There is no need to send any other secret or public key values to the server by the communicating parties. All the private or secret key values should be random, large, and different in every session. Hence, man-in-the-middle attack can be detected simply.

As a future work, our proposed system can be altered in order to contribute a better secure system and also can be used in the particular fields where it is applicable. We are also very much interested to study the comparative analysis of our proposed system with other existing systems in order to enhance the performance and security of the system.

References

1. Diffie, W., Hellman, M.: New directions in cryptography. IEEE Trans. Inf. Theory **22**, 644–654 (1976)
2. Baylis, J.: Cyclic codes. In: Error-correcting Codes, pp. 141–159 (1998)
3. Dubrova, E., Näslund, M., Selander, G., Lindqvist, F.: Message authentication based on cryptographically secure CRC without polynomial irreducibility test. Cryptogr. Commun. **10**, 383–399 (2017)
4. Krawczyk, H.: LFSR-based hashing and authentication. In: Advances in Cryptology—CRYPTO, pp. 129–139 (1994)
5. Khader, A., Lai, D.: Preventing man-in-the-middle attack in Diffie-Hellman key exchange protocol. In: 2015 22nd International Conference on Telecommunications (ICT) (2015)
6. Laser, J., Jain, V.: Enhanced security mechanism in public key cryptosystems using biometric person authentication. In: 2016 International Conference on Computation of Power, Energy Information and Communication (ICCPEIC) (2016)
7. Kumar, C.K., Jose, G.J.A., Sajeev, Suyambulingom, C.: Safety measures against man-in-the-middle attack in key exchange. Asia Research Publishing Network (ARPN) J. Eng. Appl. Sci. **7**, 243–246 (2006)
8. Tanenbaum, A., Tanenbaum, A.: Computer Networks, 4th edn. Prentice Hall PTR, Upper Saddle River, NJ (2003)

9. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Advances in Cryptology: Proceedings of CRYPTO 84. Lecture Notes in Computer Science, vol. 7, pp. 47—53 (1984)
10. Trusted third party. https://en.wikipedia.org/wiki/Trusted_third_party, Accessed 11 Aug 2017