Variations on Diffie-Hellman Key Exchange to Optimize the Resistivity

A Mini Project Report Submitted by

Harshit Sunil Shirsat (4NM17CS068)
Gagandeep Bekal (4NM17CS062)

UNDER THE GUIDANCE OF

Mr. RADHAKRISHNA DODMANE

Associate Professor

Department of Computer Science and Engineering

in partial fulfilment of the requirements for the award of the Degree of

Bachelor of Engineering in Computer Science & Engineering

from

Visvesvaraya Technological University, Belgaum

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

N.M.A.M. INSTITUTE OF TECHNOLOGY

(An Autonomous Institution under VTU, Belgaum) (AICTE approved, NBA Accredited, ISO 9001:2008 Certified) NITTE -574 110, Udupi District, KARNATAKA.

April 2020

NITTE EDUCATION TRUST | N.M.A.M. INSTITUTE OF TECHNOLOGY
(An Autonomous Institution affiliated to Visvesvaraya Technological University, Belagavi)
Nitte – 574 110, Karnataka, India
(ISO 9001:2015 Certified), Accredited with 'A' Grade by NAAC
☎: 08258 - 281039 – 281263, Fax: 08258 – 281265
Department of Computer Science and Engineering

B.E. CSE Program Accredited by NBA, New Delhi from 1-7-2018 to 30-6-2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Variations on Diffie-Hellman Key Exchange to Optimize the Resistivity

is a bonafide work carried out by

Harshit Sunil Shirsat (4NM17CS068)
Gagandeep Bekal (4NM17CS062)

in partial fulfilment of the requirements for the award of
Bachelor of Engineering Degree in Computer Science and Engineering
prescribed by Visvesvaraya Technological University,
Belgaum during the year 2019-2020.

It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report.

The Mini project report has been approved as it satisfies the academic requirements in respect of the project work prescribed for the Bachelor of Engineering Degree.

Signature of Guide Signature of HOD

ACKNOWLEDGMENT

We believe that our project will be complete only after we thank the people who have contributed to make this project successful.

First and foremost, our sincere thanks to our beloved principal, **Dr. Niranjan N. Chiplunkar** for giving us an opportunity to carry out our project work at our college and providing us with all the needed facilities.

We express our deep sense of gratitude and indebtedness to our guide **Mr. Radhakrishna Dodmane**, Associate Professor, Department of Computer Science and Engineering, for his inspiring guidance, constant encouragement, support and suggestions for improvement during the course of our project.

We sincerely thank **Dr. K.R. Udaya Kumar Reddy**, Head of Department of Computer Science and Engineering, Nitte Mahalinga Adyantaya Memorial Institute of Technology, Nitte.

We also thank all those who have supported us throughout the entire duration of our project.

Finally, we thank the staff members of the Department of Computer Science and Engineering and all our friends for their honest opinions and suggestions throughout the course of our project.

Harshit Sunil Shirsat (4NM17CS068)
Gagandeep Bekal (4NM17CS062)

ABSTRACT

A cryptographic algorithm requires a shared key between the sender and the receiver. The sender uses the key to encrypt the message, while the receiver needs the key to decrypt it. Without the shared key, they wouldn't be able to understand each other's messages.

Diffie-Hellman key exchange method can be used to securely exchange keys over an insecure network. This allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher. It works using the concepts of prime numbers and modulo operations.

In this project we have designed our own encryption and decryption technique by making use of Substitution Cipher and the key generated by making a variation to the Diffie-Hellman key exchange method.

By making use of Substitution Cipher we have first generated a set of ciphertext values. On this set of ciphertext values we have performed arithmetic operations using the shared key to make it less immune to attacks. This enhanced ciphertext will then be sent to the receiver on the other end. The receiver's task will be to undo the same set of arithmetic operation done on the sender's side and map the obtained numbers to its corresponding plain text.

INDEX

Chapter 1: INTRODUCTION

Asymmetric Encryption of data requires transfer of cryptographic private key. The most challenging part in this type of encryption is the transfer of the encryption key from sender to receiver without anyone intercepting this key in between. This transfer or rather generation on same cryptographic keys at both sides secretively was made possible by the Diffie-Hellman algorithm.

The Diffie-Hellman algorithm was developed by Whitfield Diffie and Martin Hellman in 1976. This algorithm was designed not to encrypt the data but to generate same private cryptographic key at both ends so that there is no need to transfer this key from one communication end to another. Though this algorithm is a bit slow but it is the sheer power of this algorithm that makes it so popular in encryption key generation. It is a method of securely exchanging cryptographic keys over a public channel and was one of the first public-key protocols.

Diffie-Hellman is one of the earliest practical examples of public key exchange implemented within the field of cryptography. Traditionally, secure encrypted communication between two parties required that they first exchange keys by some secure physical means, such as paper key lists transported by a trusted courier. The Diffie-Hellman key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher.

But this algorithm is no longer strong, since the key can be easily identified by discrete logarithmic approach.

Chapter 2: LITERATURE SURVEY

Monoalphabetic Substitution Cipher:

Substitution ciphers are probably the most common form of cipher. They work by replacing each letter of the plaintext with another letter.

A monoalphabetic substitution cipher, also known as a simple substitution cipher, relies on a fixed replacement structure. That is, the substitution is fixed for each letter of the alphabet. Thus, if "a" is encrypted to "R", then every time we see the letter "a" in the plaintext, we replace it with the letter "R" in the ciphertext.

In general, when performing a simple substitution manually, it is easiest to generate the ciphertext alphabet first, and encrypt by comparing this to the plaintext alphabet. The table below shows how one might choose to substitute this example.

Plaintext Alphabet	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Ciphertext Alphabet	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A

Table 1) Example of a monoalphabetic substitution cipher

Symmetric Key Cryptography:

Symmetric key cryptography, also known as private key cryptography, utilizes a single key for both encryption of the plaintext and decryption of the ciphertext. Symmetric encryption schemes rely on a single key that is shared between two or more users. The same key is used to encrypt and decrypt the so-called plaintext (which represents the message or piece of data that is being encoded).

If the encryption scheme is strong enough, the only way for a person to read or access the information contained in the ciphertext is by using the corresponding key to decrypt it. A 128-bit key, for example, would take billions of years to guess using common computer hardware. The longer the encryption key is, the harder it becomes to crack it. The key itself must be shared between the sender and the receiver, and this process, known as key exchange, constitutes an entire subtopic of cryptography.

Diffie-Hellman:

Diffie-Hellman key exchange, also called exponential key exchange, is a method of digital encryption that uses numbers raised to specific powers to produce decryption keys on the basis of components that are never directly transmitted, making the task of a would-be code breaker mathematically overwhelming.

Steps in Diffie-Hellman Key Exchange:

The simplest and the original implementation of the protocol uses the multiplicative group of integers modulo p, where p is prime, and g is a primitive root modulo p.

1. Alice and Bob agree on a prime number p and a base g.
2. Alice chooses a secret integer a, then sends Bob

$$A = g^a \ mod \ p$$

3. Bob chooses a secret integer b, then sends Alice

$$B = g^b \ mod \ p$$

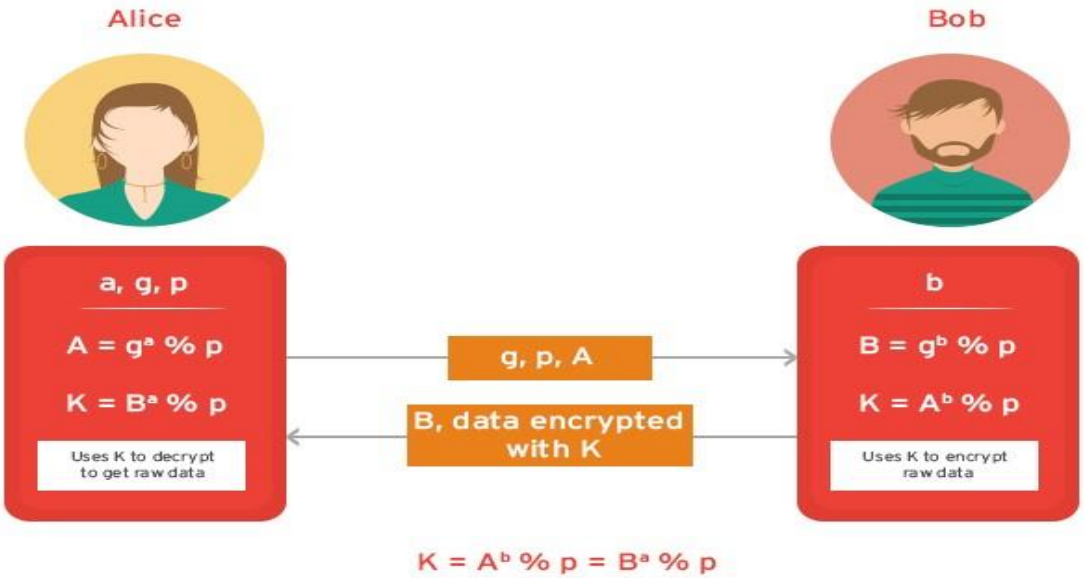
4. Alice computes

$$K2 = B^a \ mod \ p$$

5. Bob computes

$$K2 = A^b \ mod \ p$$

6. Alice and Bob now share a secret key such that *K1=K2* ie. both Bob and Alice can use this number as their key.



Chapter 3: PROBLEM DEFINITION

The main aim of our project is to design our own cipher system. We intend to do so by making use of monoalphabetic substitution and generate a cipher test by using the shared key generated by Diffie-Hellman key exchange in order to increase the resistivity. We also intend on making a variation on the traditional Diffie-Hellman algorithm to overcome the shortcomings of the algorithm.

Thereby we aim to design a simpler and more secure cipher system by using monoalphabetic substitution cipher and a modified Diffie-Hellman key exchange.

Chapter 4: METHODOLOGY

Working Principle:

We first generate a shared key by making use of a modified variant of Diffie-Hellman key exchange.

Modified Diffie-Hellman key exchange

Our proposed implementation of the protocol uses integer p, where p is prime, and g is a primitive root of p.

1. Alice and Bob agree on a prime number p and a primitive root g.
2. Alice chooses a secret integer a, then sends Bob

$$A = (g * a) * p$$

3. Bob chooses a secret integer b, then sends Alice

$$B = (g * b) * p$$

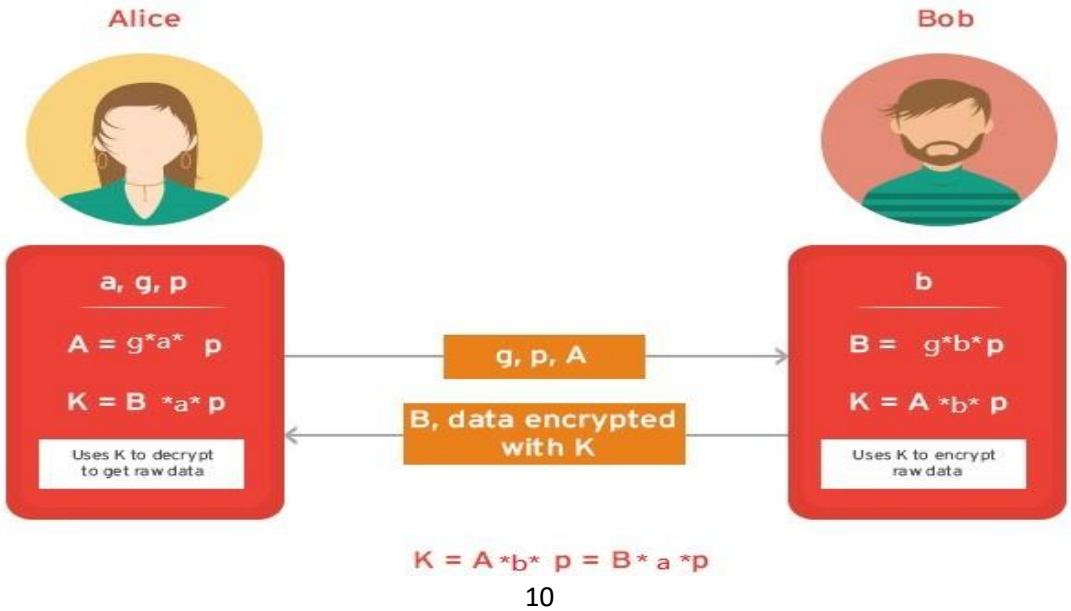
4. Alice computes

$$K2 = (B * a) * p$$

5. Bob computes

$$K2 = (A * b) * p$$

6. Alice and Bob now share a secret key such that *K1=K2* i.e both Bob and Alice can use this number as their key.



On the Sender's side:

Modified monoalphabetic substitution cipher

We have substituted the plaintext characters below in the increasing order of prime numbers as given below,

Plaintext	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	g	h	i	j
Ciphertext	2	3	5	7	11	13	17	19	23	31	37	41	43	47	53	59	61	67	71	73

k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	A
79	83	89	97	101	103	107	109	113	127	131	131	137	139	149	151	157

B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
163	167	173	179	181	191	193	197	199	211	223	227	229	233	239	241

R	S	T	U	V	W	X	Y	Z
251	257	263	269	271	277	281	283	293

For example,
If the message from the sender is 'Hello' then CT for the message is,

H → 193 e → 53 l → 83 l → 83 o → 101

Updated Ciphertext for transmission

On the above monoalphabetic substitution cipher we have performed the below operation by making use of the shared secret key. This operation performed has been agreed upon by the sender and receiver beforehand privately. This adds an extra layer of security to the cipher system.

On any given ciphertext, before transmission we update the ciphertext by the below given formula

$$Updated\ CT = (CT * K) + K^2$$

where, CT is the Ciphertext from monoalphabetic substitution

K is the shared key from modified DHE

For example,
If the ciphertext generated from monoalphabetic substitution cipher for the letter 'a' is 31 then updated CT for 'a' is, (assume K=2)

$$Updated\ CT = (37 * 2) + 2^2$$

$$= 78$$

On the Receiver's side:

Encryption of the Updated Ciphertext

The receiver now has to undo the operation done on the updated ciphertext to retrieve the original cipher text. The receiver can perform this without anyone else knowing as both the sender and receiver have decided upon the operation to be performed privately beforehand.

On the received ciphertext, the receiver undoes the operation by the below given formula

$$CT = \frac{(Updated\ CT - K^2)}{K}$$

where, CT is the Ciphertext from monoalphabetic substitution

K is the shared key from modified DHE

For example,
If the updated ciphertext received is 78 then CT for 78 is, (assume K=2)

$$CT = \frac{(74 - 2^2)}{2}$$

$$= 37$$

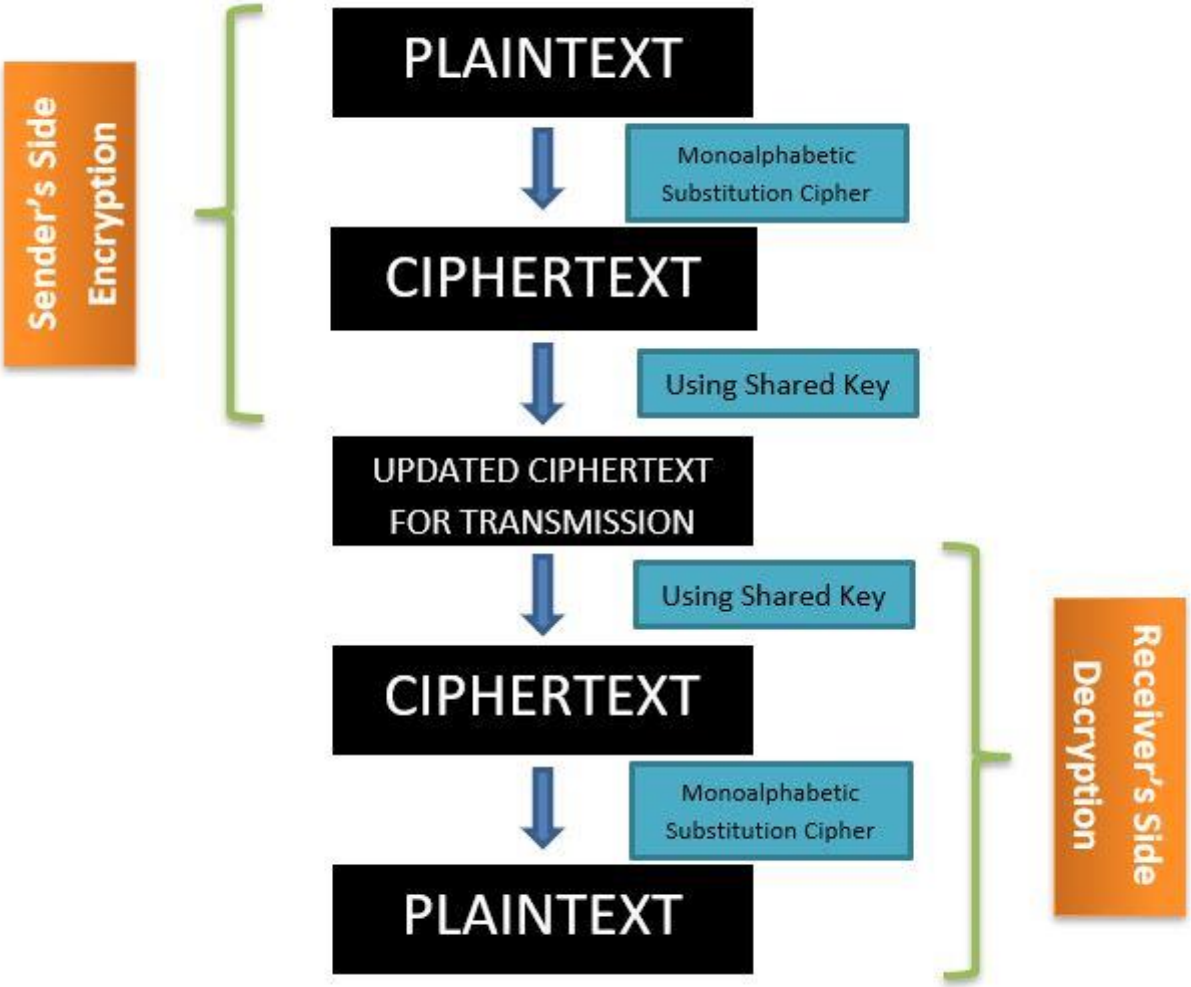
Mapping the ciphertext back to plaintext

This is done by looking at the modified monoalphabetic substitution table. And on performing this me retrieve the original message sent by the sender.

For example,
If the ciphertext is as given below then corresponding plaintext for it is

Ciphertext	193	53	83	83	101
Plaintext	H	e	l	l	o

Flowchart of cipher system:



Implementation of the algorithm:

```
***CONFIDENTIAL - Diffie Hellman Key Exchange***

Enter the value of P which is agreed upon by the SENDER and RECEIVER - 53
The value of P : 53

Enter the value of G which is agreed upon by the SENDER and RECEIVER - 7
The value of G : 7

***SENDER***

Enter SENDER's private key - 12
The private key 'ka' for SENDER : 12

***RECEIVER***

Enter RECEIVER's private key - 15
The private key 'kb' for RECEIVER : 15

***SENDER***

SENDER's Shared Secret key : 13

***RECEIVER***

RECEIVER's Shared Secret key : 13

***KEY EXCHANGE SUCCESSFUL***
Enter any key to continue to message transmission page !!
```

```
***SENDER SIDE***

Enter the message to be transmitted - Corona
Message from the SENDER -  Corona

***ENCRYPTION AT SENDER SIDE***

SENDER's Shared Key : 13
SENDER's Message Encrypted As :
C       109
o       47
r       61
o       47
n       43
a       2

***TRANSMITTED MESSAGE VISIBLE TO MIDDLEMAN/ATTACKER OVER THE NETWORK***

Message transmitted as : 1586780962780728195

***DECRYPTION AT RECEIVER SIDE***

RECEIVER's Shared Key - 13
RECEIVER's Decrypts Message As :
1586    C
780     o
962     r
780     o
728     n
195     a

***RECEIVER SIDE***

Message Received by RECEIVER : Corona

***TRANSMISSION SUCCESSFUL***

Process returned 0 (0x0)   execution time : 48.337 s
Press any key to continue.
```

Chapter 5: ADVANTAGES, DISADVANTAGES AND APPLICATIONS

Advantages:

1. The sender and receiver don't need any prior knowledge of each other.
2. Once the keys are exchanged, the communication of data can be done through an insecure channel.
3. It is a lightweight two-pass protocol with only a public key transport from participant A to participant B and again from B to A.
4. Producing a new key pair is extremely fast.
5. The time complexity of our modified Diffie-Hellman algorithm is decreased after our variations on it.
6. It is more difficult for the cryptanalyst attacker to decipher the values after our modification in the algorithm.
7. The Diffie-Hellman key exchange relies on one-way functions as the basis for its security. These are calculations which are simple to do one way, but much more difficult to calculate in reverse.

Disadvantages:

1. The algorithm cannot be used for any asymmetric key exchange.
2. It cannot be used for signing digital signatures i.e there is no identity of the parties involved in the exchange.
3. It is easily susceptible to man-in-the-middle attacks.
4. The algorithm is computationally intensive. Each multiplication varies as the square of n, which must be very large. The number of multiplications required by the exponentiation increases with increasing values of the exponent, x or y in this case.
5. The computational nature of the algorithm could be used in a denial-of-service attack very easily.

Applications:

1. **Encryption**: Our program can be used to do encryption. One modern approach of it is called Integrated Encryption Scheme which provides security against chosen plaintext and chosen clipboard attacks.
2. **Password Authenticated Agreement:** When two parties share a password, a password-authenticated key agreement can be used to prevent the man in the middle attack. This key agreement can be in the form of Diffie-Hellman. Secure Remote Password Protocol is a good example that is based on this technique.
3. **Forward Secrecy:** Forward secrecy based protocols can generate new key pairs for each new session, and they can automatically discard them at the end when the session is finished too. In these forward Secrecy protocols, more often than not, key is exchanged.
4. **Secure Sockets Layer (SSL):** The SSL is the standard security technology developed by Netscape in 1994 to establish an encrypted link between a web server and a browser. This link ensures privacy and integrity of all data passed between the web server and browsers. SSL is used by millions of websites in the protection of their online transactions with their customers.
5. **IP Security (IPSec):** IPSec (IP Security) is an extension of the Internet Protocol (IP)—it is a suite of protocols introduced by the Internet Engineering Task Force (IETF) to aid in configuring a communications channel between multiple machines. Operating at the IP layer of the seven-layer model, it does its job by authenticating and encrypting IP packets.
6. **Secure Shell (SSH):** SSH is a network security protocol very common for secure remote login on the Internet. The secure shell has come to replace the unsecured Telnet on the network and FTP on the system, mostly because both Telnet and FTP do not encrypt data, and instead send them in plaintext. SSH, on the other hand, can automatically encrypt, authenticate and compress transmitted data.

Chapter 6: RESULT AND DISCUSSION

Time complexity analysis of Diffie-Hellman and modified Diffie-Helman key exchange:

Here, P – Prime number
G – Primitive root of P
a – Private key for sender
b – Private key for receiver

The time complexity of the general algorithm is shown in the table 2.

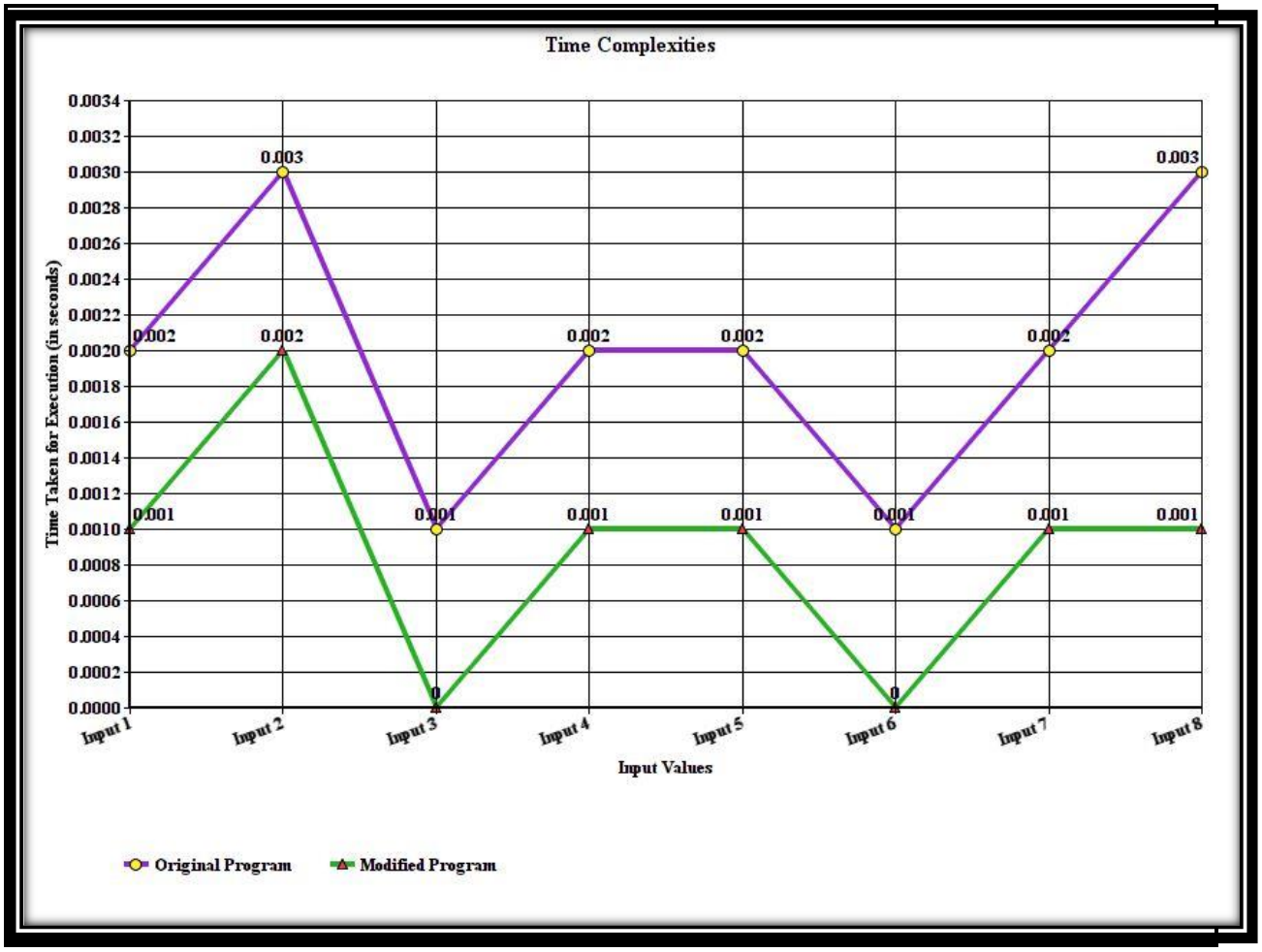
P	G	a	b	Time in (s)
53	7	12	15	0.002000
218	11	5	6	0.003000
17	4	3	6	0.001000
941	627	347	781	0.002000
353	3	97	223	0.002000
23	9	4	3	0.001000
7	11	3	9	0.002000
41	11	31	41	0.003000

Table 2) Time complexity of the general DHE algorithm

The time complexity of the modified algorithm is shown in table 3.

P	G	a	b	Time in (s)
53	7	12	15	0.001000
218	11	5	6	0.002000
17	4	3	6	0.000001
941	627	347	781	0.001000
353	3	97	223	0.001000
23	9	4	3	0.000001
7	11	3	9	0.001000
41	11	31	41	0.001000

Table 3) Time complexity of the modified DHE algorithm



Graph 1) Line graph comparing time complexity of the two algorithms

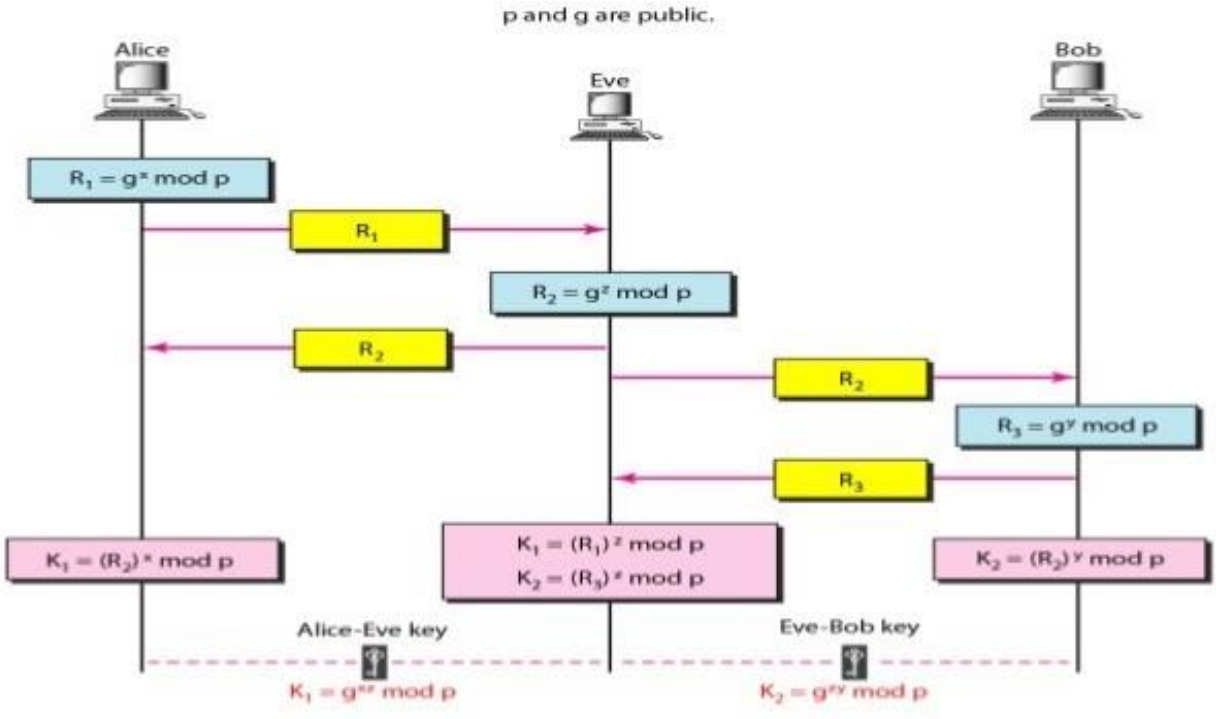
Issues and Security:

Diffie-Hellman key exchange is vulnerable to 2 kind of attacks

1) Man-in-the-middle Attack

   Diffie-Hellman key exchange, especially in its early forms, has a noteworthy weakness as man-in-the-middle vulnerability. In this attack, a malicious third party, usually referred to as "Eve" (for "eavesdropper") recovers Alice's public key and sends her own public key to Bob. At the point when Bob transmits his public key, Eve interrupts on and substitutes the value with her own public key and after that sends it to Alice. At this point, Alice would be making an agreement on a common secret key with Eve rather than Bob. This exchange should be possible in opposite, and it is feasible for Eve to decrypt any messages conveyed by Alice or Bob, and after that read and perhaps alter them before the re-encryption with the suitable key and transmitting them to alternate party. This

weakness is available on the grounds that Diffie- Hellman key exchange does not authenticate the members.



2) <u>Impersonation Attack</u>

The impersonation attack proceeds as follows:

step1: A → C(B): α^{Xa} mod G

step2: C(B) → A: α^{Xc} mod G

User A sends his public key *Y* in a message addressed to user B, but the message is intercepted by enemy C. C saves A's public key and sends a message to A back which includes B's User ID but C's public key *Yc*. In this attack, B is simply not involved in implementation of the agreement. The attack result is C and A shared key (Yc^{Xa} mod G), while the A thinks the key is t shared with B. From the analysis of Diffie-Hellman protocol we can see the fundamental reason that the protocol subject to the above attacks is that there is no authentication between the two parties

Authentication methods:

The above attacks can be avoided by bringing in various authentication methods. Usually there are three different authentication methods used with Diffie-Hellman key exchange algorithm.

1) Digital signatures: The exchange is authenticated by signing a mutually obtainable hash; each party encrypts the hash with its private key. The hash is generated over important parameters, such as user ID and nonce.
2) Public-key encryption: The exchange is authenticated by encrypting parameters such an ID and nonce with the sender's private key.
3) Symmetric-key encryption: A key derived by some out-of-band mechanism can be used to authenticate the exchange by symmetric encryption of exchange parameters.

Improving security of Diffie-Hellman key exchange:

Parameters for number selection

If a real-world implementation of the Diffie-Hellman key exchange used numbers as small as those in our example, it would make the exchange process trivial for an attacker to crack. But it's not just the size of the numbers that matter, the numbers also need to be sufficiently random. If a random number generator produces a predictable output, it can completely undermine the security of the Diffie-Hellman key exchange. The number *p* should be 2048 bits long to ensure security. The base, *g*, can be a relatively small number like 2, but it needs to come from an order of *G* that has a large prime factor.

Result:

Our modified Diffie Hellman Key can be used with RSA, DES Public key and AES public key to encrypt and decrypt the message. It also decreases the time complexity hence making it more time efficient.

We also stated the possible attacks on the Diffie-Hellman key exchange algorithm and made modifications to the traditional algorithm to make it a more efficient key exchange protocol.

We also performed analysis on the time complexities of the traditional Diffie-Hellman key exchange algorithm and our modified algorithm by using a line graph. The results proved that our modification made the algorithm better.

The Diffie-Hellman key exchange algorithm has turned out to be a standout amongst the most fascinating key distribution schemes being used today. Nonetheless, one must know about the way that in spite of the algorithm is safe against passive eavesdropping, it is not necessarily protected from active attacks. Diffie-Hellman algorithm should be complemented with an authentication mechanism.

Chapter 7: REFERENCES

1. Manoj Ranjan Mishra, Jayaprakash Kar, A STUDY ON DIFFI-HELLMAN KEY EXCHANGE PROTOCOLS, International Journal of Pure and Applied Mathematics, Volume 114 No.2 2017, 179-189.
2. Nan Li, Research on Diffie-Hellman Key Exchange Protocol, The People's Armed Police Force Academy of China, International Conference on Computer Engineering and Technology, Volume 4, 2010.
3. Sivanagaswathi Kallam, Diffie-Hellman: Key Exchange and Public Key Cryptosystems, Indiana State University, Math and Computer Science Dept, 2015.
4. Mr. Randhir Kumar, Dr. Ravindranath C, Analysis of Diffie Hellman Key Exchange, International Journal of Emerging Trends and Technology in Computer Science(IJETICS), ISSN 2278-6856.
5. Stinson, D. R. (2005). Cryptography: Theory and Practice (3 ed.), Volume 36 of Discrete Mathematics and Its Applications. University of Waterloo, Ontario, Canada: CRC. Press Online.
6. Vasco, M. I. G., M. N¨aslund, and I. Shparlinski (2004). New results on the hardness of Diffie-Hellman bits. In Proc. Intern. Workshop on Public Key Cryptography, Volume 2947 of Lect. Notes in Comp. Sci., Singapore, pp. 159–172. Springer-Verlag
7. Y. Amir, Y.Kim, C. Nita-Rotaru, " Secure communication using contributory key agreement", IEEE Transactions on Parallel and Distributed systems, pp. 468-480,2009
8. Ram Ratan Ahirwal, Manoj Ahke, Elliptic Curve Diffie-Hellman Key Exchange Algorithm for Securing Hypertext Information on Wide Area Network, IJCSIT
9. A. M. Fiskiran and R. B. Lee. "Workload characterization of elliptic curve cryptography and other network security algorithms for constrained environments". IEEE International Workshop on WWC-5, 2009
10. D. Hakerson, A. Menezes, and S. Vanston, "Guide to Elliptic Curve Cryptography," Springer-Verlag, NY (2004
11. Ch. Suneetha, D. Sravana Kumar and A. Chandrasekhar, "Secure key transport in symmetric cryptographic protocols using elliptic curves over finite fields," International Journal of Computer Applications, Vol. 36,1 November 2011.
12. Mohsen Machhout eat.al., "coupled FPGA/ASIC Implementation of elliptic curve crypto-processor," International Journal of Network Security & its Applications Vol. 2 No. 2 April 2010