different lock and key pair is issued (for security reasons), depending on the situation. This is quite a bit of task! Also, who is T, after all? T must be highly trustworthy and accessible to everybody. This is because each communicating pair has to approach T to obtain the lock-and-key pair. This is quite a tedious and time-consuming process!

2.6.2 Diffie–Hellman Key Exchange/Agreement Algorithm

Introduction Whitefield Diffie and Martin Hellman devised an amazing solution to the problem of key agreement or key exchange in 1976. This solution is called as the **Diffie–Hellman Key Exchange/Agreement Algorithm**. The beauty of this scheme is that the two parties, who want to communicate securely, can agree on a symmetric key using this technique. This key can then be used for encryption/decryption. However, we must note that Diffie–Hellman key exchange algorithm can be used only for key agreement, but not for encryption or decryption of messages. Once both the parties agree on the key to be used, they need to use other symmetric key encryption algorithms (we shall discuss some of those subsequently) for actual encryption or decryption of messages.

Although the Diffie-Hellman key exchange algorithm is based on mathematical principles, it is quite simple to understand. We shall first describe the steps in the algorithm, then illustrate its use with a simple example and then discuss the mathematical basis for it.

Description of the Algorithm Let us assume that Alice and Bob want to agree upon a key to be used for encrypting/decrypting messages that would be exchanged between them. Then, the Diffie-Hellman key exchange algorithm works as shown in Fig. 2.49.

1. Firstly, Alice and Bob agree on two large prime numbers, n and g. These two integers need not be kept secret. Alice and Bob can use an insecure channel to agree on them.
2. Alice chooses another large random number x, and calculates A such that:
   $A = g^x \bmod n$
3. Alice sends the number A to Bob.
4. Bob independently chooses another large random integer y and calculates B such that:
   $B = g^y \bmod n$
5. Bob sends the number B to Alice.
6. A now computes the secret key K1 as follows:
   $K1 = B^x \bmod n$
7. B now computes the secret key K2 as follows:
   $K2 = A^y \bmod n$

Fig. 2.49 *Diffie–Hellman key exchange algorithm*

This is shown diagrammatically in Fig. 2.50.

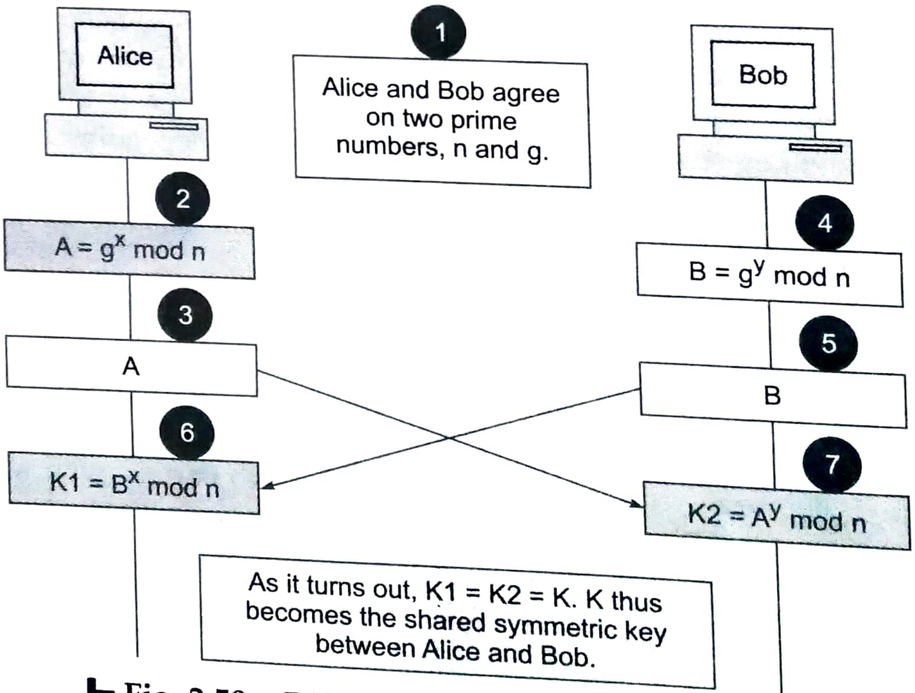
Alice
1
Alice and Bob agree on two prime numbers, n and g.
Bob
2
A = g^x mod n
4
B = g^y mod n
3
A
5
B
6
K1 = B^x mod n
7
K2 = A^y mod n
As it turns out, K1 = K2 = K. K thus becomes the shared symmetric key between Alice and Bob.

Fig. 2.50 *Diffie-Hellman key exchange illustrated*

It might come as a surprise, but K1 is actually equal to K2! This means that K1 = K2 = K is the symmetric key, which Alice and Bob must keep secret and can henceforth use for encrypting/decrypting their messages with. The mathematics behind this is quite interesting. We shall first prove it and then examine it.

Example of the Algorithm Let us take a small example to prove that the Diffie–Hellman works in practical situations. Of course, we shall use very small values for ease of understanding. In real life, these values are very large. The process of key agreement is shown in Fig. 2.51.

1. Firstly, Alice and Bob agree on two large prime numbers, n and g. These two integers need not be kept secret. Alice and Bob can use an insecure channel to agree on them.

   Let $n = 11, g = 7$.

2. Alice chooses another large random number x, and calculates A such that:
   $A = g^x \text{ mod } n$

   Let $x = 3$. Then, we have, $A = 7^3 \text{ mod } 11 = 343 \text{ mod } 11 = 2$.

3. Alice sends the number A to Bob.

   Alice sends 2 to Bob.

4. Bob independently chooses another large random integer y and calculates B such that:
   $B = g^y \text{ mod } n$

   Let $y = 6$. Then, we have, $B = 7^6 \text{ mod } 11 = 117649 \text{ mod } 11 = 4$.

5. Bob sends the number B to Alice.

   Bob sends 4 to Alice.

6. A now computes the secret key K1 as follows:
   $K1 = B^x \text{ mod } n$

   We have, $K1 = 4^3 \text{ mod } 11 = 64 \text{ mod } 11 = 9$.

7. B now computes the secret key K2 as follows:
   $K2 = A^y \text{ mod } n$

   We have, $K2 = 2^6 \text{ mod } 11 = 64 \text{ mod } 11 = 9$.

Fig. 2.51 *Example of Diffie-Hellman key exchange*

Having taken a look at the actual proof of Diffie–Hellman key exchange algorithm, let us now think about the mathematical theory behind it.

Mathematical Theory Behind the Algorithm Let us first take a look at the technical (and quite complicated) description of the complexity of the algorithm:

> *Diffie–Hellman key exchange algorithm gets it security from the difficulty of calculating discrete logarithms in a finite field, as compared with the ease of calculating exponentiation in the same field.*

Let us try to understand what this actually means, in simple terms.

(a) Firstly, take a look at what Alice does in Step 6. Here, Alice computes:
$K1 = B^x \text{ mod } n$.
What is B? From Step 4, we have:
$B = g^y \text{ mod } n$.

Therefore, if we substitute this value of B in Step 6, we will have the following equation:

$K1 = (g^y)^x \bmod n = g^{yx} \bmod n$

(b) Now, take a look at what Bob does in Step 7. Here, Bob computes:

$K2 = A^y \bmod n$.

What is A? From Step 2, we have:

$A = g^x \bmod n$.

Therefore, if we substitute this value of A in Step 7, we will have the following equation:

$K2 = (g^x)^y \bmod n = g^{xy} \bmod n$

Now, basic mathematics says that:

$K^{yx} = K^{xy}$

Therefore, in this case, we have: K1 = K2 = K. Hence the proof.

An obvious question now is, if Alice and Bob can both calculate K independently, so can an attacker! What prevents this? The fact is, Alice and Bob exchange n, g, A and B. Based on these values, x (a value known only to Alice) and y (a value known only to Bob) cannot be calculated easily. Mathematically, the calculations do find out x and y are extremely complicated, if they are sufficiently large numbers. Consequently, an attacker cannot calculate x and y and therefore, cannot derive K.

Why Diffie–Hellman Works? The idea behind Diffie-Hellman is quite simple but beautiful. Think about the final shared symmetric key between Alice and Bob to be made up of three parts: g, x and y, as shown in Fig. 2.52.

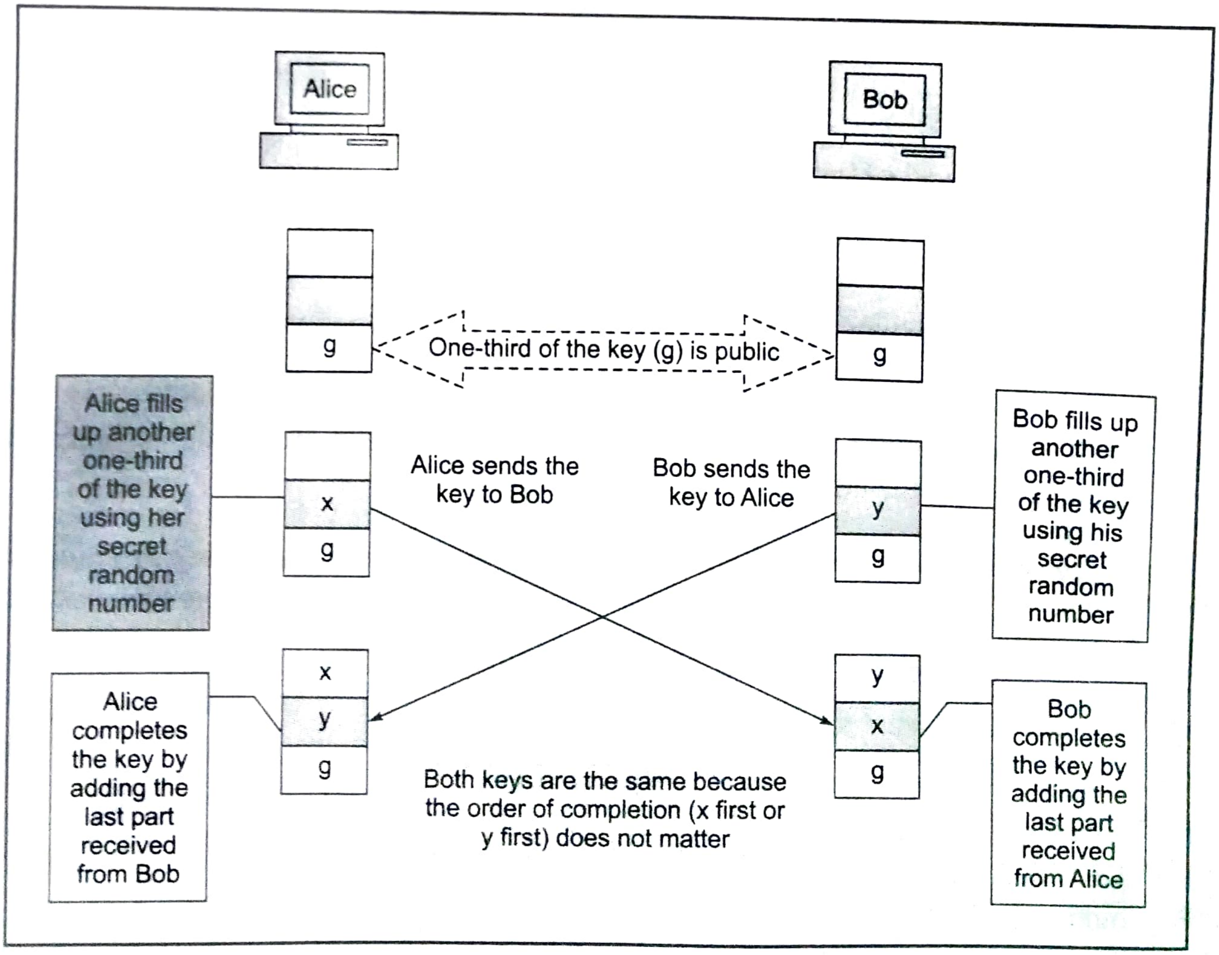
Alice
Bob
g
One-third of the key (g) is public
g
Alice fills up another one-third of the key using her secret random number
x
g
Alice sends the key to Bob
Bob sends the key to Alice
y
g
Bob fills up another one-third of the key using his secret random number
x
y
g
Alice completes the key by adding the last part received from Bob
y
x
g
Bob completes the key by adding the last part received from Alice
Both keys are the same because the order of completion (x first or y first) does not matter

Fig. 2.52 *Why Diffie–Hellman works?*

The first part of the key (g) is a public number, known to everyone. The other two parts of the key (i.e. x and y) must be made available by Alice and Bob. Alice adds the second part (x), while Bob adds the third (y). When Alice receives the two-thirds completed key from Bob, she adds the one-third remaining part (i.e. x). This completes Alice's key. Similarly, when Bob receives the two-thirds completed key from Alice, he adds the one-third remaining part (i.e. y). This completes Bob's key.

Note that although Alice's key is made up using a sequence of g-y-x and Bob's key is made up using a sequence of g-x-y, the two keys are the same because $g^{xy} = g^{yx}$.

Importantly, although the eventual two keys are the same, Alice cannot find Bob's part (i.e. y) because the computation is done using modulus n. Similarly, Bob cannot derive Alice's part (i.e. x).

Problems with the Algorithm Can we now consider that the Diffie-Hellman key exchange algorithm solve all our problems associated with key exchange? Unfortunately, not quite!

Diffie–Hellman key exchange algorithm can fall pray to the **man-in-the-middle attack** (or to be politically correct, *woman-in-the-middle attack*), also called as **bucket brigade attack**. The name *bucket brigade attack* comes from the way fire fighters of yesteryears formed a line between fire and water source and passed full buckets toward the fire and the empty buckets back. The way this happens is as follows.

1. Alice wants to communicate with Bob securely and therefore, she first wants to do a Diffie-Hellman key exchange with him. For this purpose, she sends the values of n and g to Bob, as usual. Let $n = 11$ and $g = 7$. (As usual, these values will form the basis of Alice's A and Bob's B, which will be used to calculate the symmetric key K1 = K2 = K.)
2. Alice does not realize that the attacker Tom is listening quietly to the conversation between her and Bob. Tom simply picks up the values of n and g and also forwards them to Bob as they originally were (i.e. $n = 11$ and $g = 7$). This is shown in Fig. 2.53.

Alice	Tom	Bob
$n = 11, g = 7$	$n = 11, g = 7$	$n = 11, g = 7$

Fig. 2.53 *Man-in-the-middle attack - Part I*

3. Now, let us assume that Alice, Tom and Bob select random numbers x and y as shown in Fig. 2.54.

Alice	Tom	Bob
$x = 3$	$x = 8, y = 6$	$y = 9$

Fig. 2.54 *Man-in-the-middle attack - Part II*

4. One question at this stage could be: why does Tom selects both x and y? We shall answer that shortly. Now, based on these values, all the three persons calculate the values of A and B as shown in Fig. 2.55. Note that Alice and Bob calculate only A and B, respectively. However, Tom calculates both A and B. We shall revisit this shortly.

Alice

$A = g^x \bmod n$
$= 7^3 \bmod 11$
$= 343 \bmod 11$
$= 2$

Tom

$A = g^x \bmod n$
$= 7^8 \bmod 11$
$= 5764801 \bmod 11$
$= 9$

$B = g^y \bmod n$
$= 7^6 \bmod 11$
$= 117649 \bmod 11$
$= 4$

Bob

$B = g^y \bmod n$
$= 7^9 \bmod 11$
$= 40353607 \bmod 11$
$= 8$

Fig. 2.55 *Man-in-the-middle attack - Part III*

5. Now, the real drama begins, as shown in Fig. 2.56.

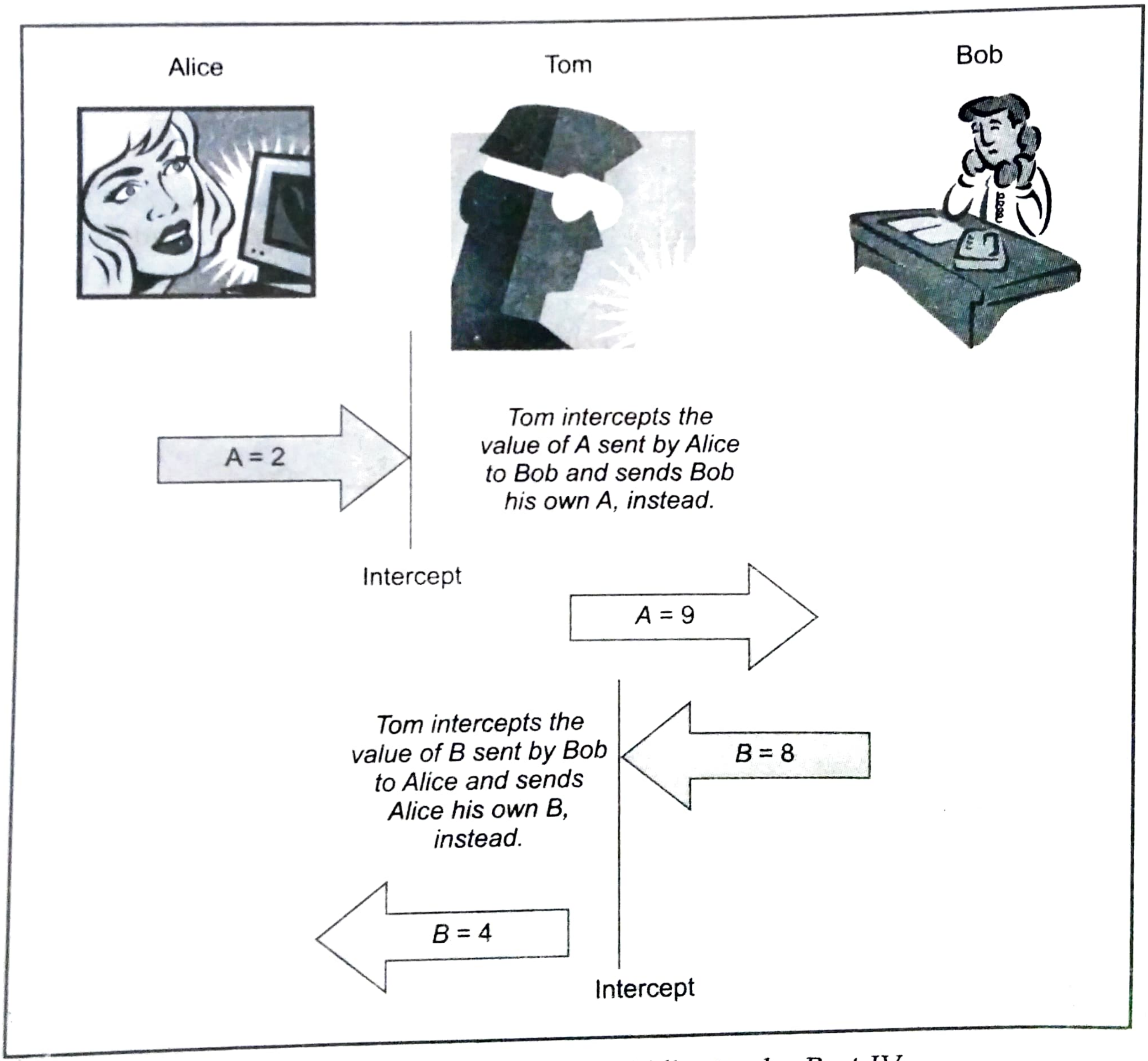
Alice
Tom
Bob
A = 2
Tom intercepts the value of A sent by Alice to Bob and sends Bob his own A, instead.
Intercept
A = 9
Tom intercepts the value of B sent by Bob to Alice and sends Alice his own B, instead.
B = 8
B = 4
Intercept

Fig. 2.56 *Man-in-the-middle attack - Part IV*

As shown in the figure, the following things happen:

(a) Alice sends her A (i.e. 2) to Bob. Tom intercepts it and instead, sends his A (i.e. 9) to Bob. Bob has no idea that Tom had hijacked Alice's A and has instead given his A to Bob.

(b) In return, Bob sends his B (i.e. 8) to Alice. As before, Tom intercepts it and instead, sends his B (i.e. 4) to Alice. Alice thinks that this B came from Bob. She has no idea that Tom had intercepted the transmission from Bob and changed B.

(c) Therefore, at this juncture, Alice, Tom and Bob have the values of A and B as shown in Fig. 2.57.

Alice	Tom	Bob
$A = 2, B = 4^*$	$A = 2, B = 8$	$A = 9^*, B = 8$

(Note: * indicates that these are the values after Tom hijacked and changed them.)

Fig. 2.57 *Man-in-the-middle attack - Part V*

6. Based on these values, all the three persons now calculate their keys as shown in Fig. 2.58. We will notice that Alice calculates only K1, Bob calculates only K2, whereas Tom calculates both K1 and K2. Why does Tom need to do this? We shall discuss that soon.

Alice		Tom		Bob	
K1	$= B^x \bmod n$	K1	$= B^x \bmod n$	K2	$= A^y \bmod n$
	$= 4^3 \bmod 11$		$= 8^8 \bmod 11$		$= 9^9 \bmod 11$
	= 64 mod 11		= 16777216 mod 11		= 387420489 mod 11
	= 9		= 5		= 5
		K2	$= A^y \bmod n$		
			$= 2^6 \bmod 11$		
			= 64 mod 11		
			= 9		

Fig. 2.58 *Man-in-the-middle attack - Part VI*

Let us now revisit the question as to why Tom needs two keys. This is because at one side, Tom wants to communicate with Alice securely using a shared symmetric key (9) and on the other hand, he wants to communicate with Bob securely using a *different* shared symmetric key (5). Only then can he receive messages from Alice, view/manipulate them and forward them to Bob and vice versa. Unfortunately for Alice and Bob, both will (incorrectly) believe that they are directly communicating with each other. That is, Alice will feel that the key 9 is shared between her and Bob, whereas Bob will feel that the key 5 is shared between him and Alice. Actually, what is happening is, Tom is sharing the key 5 with Alice and 5 with Bob!

This is also the reason why Tom needed both sets of the secret variables *x* and *y*, as well as later on, the non-secret variables A and B.

As we can see, the *man-in-the-middle attack* can work against the Diffie–Hellman key exchange algorithm, causing it to fail. This is plainly because the *man-in-the-middle* makes the actual communicators believe that they are talking to each other, whereas they are actually talking to the *man-in-the-middle*, who is talking to each of them!

This attack can be prevented if Alice and Bob authenticate each other before beginning to exchange information. This proves to Alice is Bob is indeed Bob and not someone else (e.g. Tom) posing as Bob. Similarly, Bob can also get convinced that Alice is genuine as well.