UNIT 8 DIGITAL SIGNATURES

8.1 INTRODUCTION

In the earlier units, we have seen how cryptography protects the communication between Alice and Bob from the prying eyes of Eve. However, the methods and algorithms that we have seen so far doesn't protect Alice and Bob from each other in the following sense: Suppose Alice is a stock broker and Bob is a client who buys and sell stocks through Alice. Assume that they use some public key cryptosystem to communicate with each other. Suppose Bob asks Eve to sell some stocks and they go up in value afterwards. Bob can claim that he never sent the message and that it was forged by Alice since the encrytion exponent of Bob is in the public domain. The other way around, suppose Alice wants to offload some worthless stock on Bob. Since Bob's encryption algorithm is in the public domain, she can forge a message from Bob that supposedly instructs her to buy those worthless stocks and claim that she was acting on Bob's instructions when she purchased the stocks. Of course, these problems could arise in the case of symmetric key cryptosystems also because both Alice and Bob have the encryption an decryption keys. How to address this issue?

In a traditional commercial transaction, the physical signature of the writer on a paper document served to authenticate the document and for non-repudiation. So, the question arises, whether there is a digital analogue that can serve the same purpose and protects Alice and Bob from each other. We answer this question in this Unit. In Sec. 8.2, we explain two alogrithms for digital signatures which provide the essential functionalities of the physical signature.

Objectives

After studying this unit, you should be able to

- define digital signature;
- explain the functionalities provided digital signatures;
- explain direct and arbitrated signature schemes;
- explain how to sign a message using RSA algorithm and verify a signature created with RSA algorithm;
- explain how to sign a message using ElGamal algorithm and verify the signature created using ElGamal algorithm;and
- explain how to sign a message using Digital Signature Standard and verify the signature created using Digital Signature Standard.

8.2 DIGITAL SIGNATURE ALGORITHMS

In the third unit of the first block of this course, we discussed the goals of cryptography which is to achieve PAIN, i.e. namely Privacy/confidentiality, Authentication, Integrity of data and Non-repudiation. As we have seen already, the cryptosytems provide privacy. Symmetric key cryptosystems provide certain level of authentication also if we assume that no one other than the authorised persons have access to the keys. If Bob and Alice alone know a key and Bob receives a message encrypted with this key, Bob can be certain that the message was sent by Alice. This is not there in public key cryptosystems because anyone can send a message using the encyrption exponent available in the public domain.

Even in the case of private key cryptosystems the following situations can arise:

1) Bob can send a message to Alice and deny that he sent it at a latter date.

2) Alice can forge a message and claim that Bob sent it to her.

In the case of public key cryptosystems, a third person, say Eve, can fool Alice into thinking a message from Eve to be a message from Bob.

To avoid such situations, we need the digital analogues of the conventional signatures used in paper documents. Note that, in the case of paper documents, the signature in one document cannot be pasted on another without being detected. So, whatever the digital analogue of the conventional signature be, it should not be possible to copy the digital signature from one document to another document. In other words, the digital signature and the document it authenticates must be 'inseparable' from each other. How do we achieve this?

Before we discuss specific algorithms, let us first see what are the properties of a digital signature. With the help of digital signature, we should be able to:

1) Verify the author of signature.

2) Authenticate the contents of the message, i.e. check that the message has not been tampered with.

3) Resolve disputes with the help of a third party if necessary.

To be of use, the digital signature has to satisfy the following conditions:

1) The signature should be a bit pattern that should depend on the sender.

2) Creation of digital signature should involve a secret information that is known only to the sender.

3) It should be easy to compute the digital signature.

4) The computation for verifying the signature should be easy.

5) It should be computationally infeasible to construct a message for an existing digital signature. For example, suppose Eve intercepts a message $\mathscr{M}$ by Alice to Bob with signature s. She should not be able to create another messsage $\mathscr{M}'$ with the same signature s.

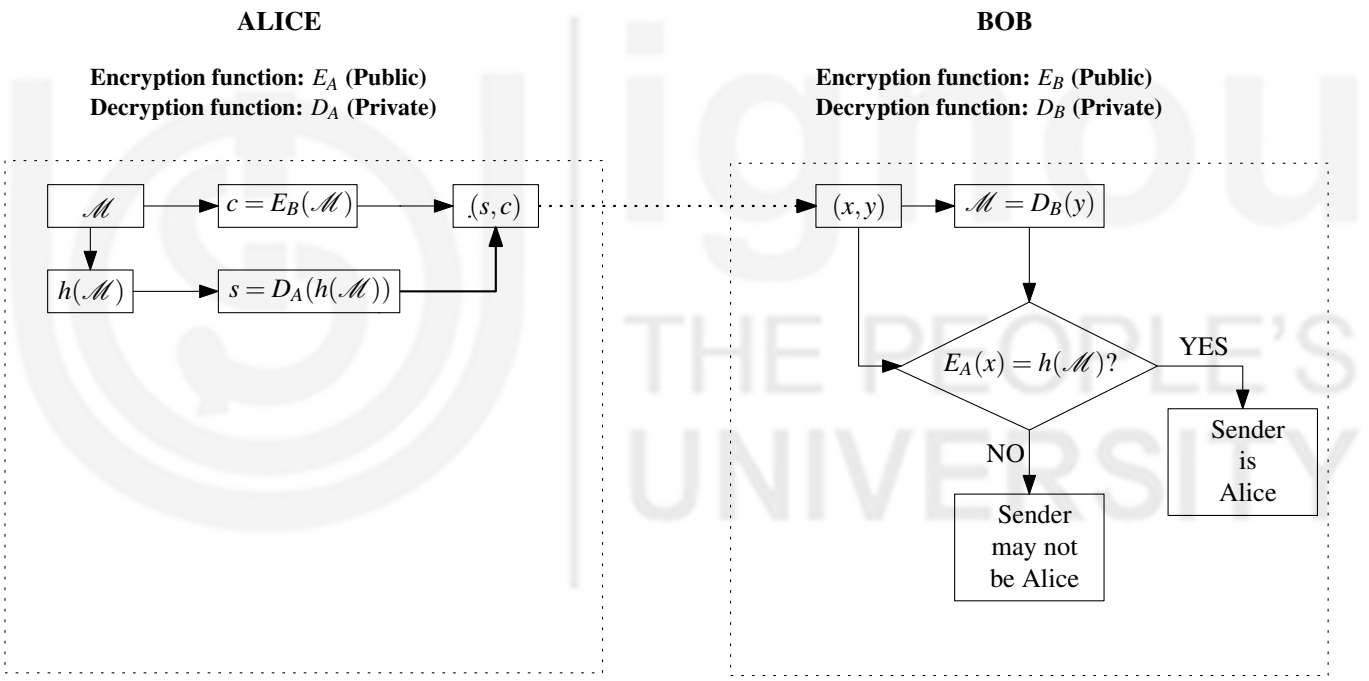
6) It should be computationally infeasible to construct a fraudulent signature for an existing message. For example, it should not be possible for Eve create a message $\mathscr{M}$ and create a signature s and make it appear that Alice has created and signed the message.

We begin by formally defining digital signature.(See [9].)

Definition 5: The digital signature for a data is the result of a cryptographic transformation of the data that, when properly implemented, provides a mechanism for verifying origin, authentication, data integrity and signatory non-repudiation.

The different approaches that have been suggested for digital signatures fall into two broad categories, namely **direct signatures** and **arbitrated signatures**.

Direct Signatures involve only Alice and Bob, the communicating parties. Suppose $(\mathrm{E_A}, \mathrm{D_A})$ is the public encryption function, private decryption function, respectively, of Alice and $(\mathrm{E_B}, \mathrm{D_B})$ is the public encryption function and private decryption function pair of Bob. (For example, they could be RSA exponentiation functions for encryption and decryption.) Let h denote a hash function. The following are the steps involved:



Signature Alice does the following:

1) Encrypts the message $\mathscr{M}$ with Bob's public key to create the cipher text $\mathrm{c} = \mathrm{E_B}(\mathscr{M})$.
2) Creates the hash $\mathrm{h}(\mathscr{M})$ of the message and encrypts it with her private encryption function $\mathrm{D_A}$ to create the signature $\mathrm{s} = \mathrm{D_A}(\mathrm{h}(\mathscr{M}))$.
3) Alice sends the pair (s, c) to Bob.

Verification Bob receives the pair (x, y). Bob checks whether $\mathrm{E_A}(\mathrm{x})$ and $\mathrm{h}(\mathrm{D_B}(\mathrm{y}))$ are the same. If they are the same, the signature is verified.

Let us see why this is so. Note that, on the one hand,

$$\mathrm{E_A}(\mathrm{x}) = \mathrm{E_A}(\mathrm{s}) = \mathrm{E_A}(\mathrm{D_A}(\mathrm{h}(\mathscr{M}))) = \mathrm{h}(\mathscr{M}).$$

On the other hand,

$$h(D_B(y)) = h(D_B(E_B(h(\mathscr{M})))) = h(\mathscr{M}).$$

If the message has been tampered with, the hash value of the tampered message will not match the hash value calculated from the signature.

Note that, encryption of the message is optional and Alice may encrypt the message only if it is confidential. If she doesn't encrypt the message and sends the signature, message pair $(s, \mathscr{M})$ to Bob, Bob has to check that

$$h(D_B(s)) = h(\mathscr{M})$$

Note that, only Alice could have created $D_A(h(\mathscr{M}))$ because she alone knows the decryption function D_A. So, only Alice could have sent the message.

Suppose Eve wants to create another message with the same digital signature. She will have to create a message $\mathscr{M}'$ such that $h(\mathscr{M}) = h(\mathscr{M}')$. By the properties of hash function, it is computationaly infeasible to produce a message $\mathscr{M}'$ such that $h(\mathscr{M}) = h(\mathscr{M}')$ because of the collision resistance property of hash funtions.

See page 50 of block 2 for properties of hash functions.

Suppose Alice claims at a later date that she never sent the message (s, c) and the issue is taken up by a judge. Bob can submit $(s, \mathscr{M})$ to the judge and the judge can calculate $E_A(s)$ and $h(\mathscr{M})$. If the two values are equal the judge concludes that Alice must have sent the message if they are equal because only Alice could have created $s = D_A(h(\mathscr{M}))$ because only she has access to D_A.

There are some shortcomings in this scheme. If Alice wants to deny that she sent the message, she may claim that her secret key was stolen by someone else and misused to create the message and its signature. One way to overcome this is to include a time stamp in the message and insist that Alice report any loss of key to a central authority.

Another possible problem could be that Eve could steal Alice's key at time T and she may send a message signed with Alice's signature and a time stamp before or equal to T.

Some of these problems can be avoided by using arbitrated schemes. We now discuss an arbitrated scheme that works with symmetric key cryptosystem. Suppose Alice wants to send a message to Bob. We assume that there is an arbiter trusted by both Alice and Bob. Alice shares a secret key k_A with arbiter and Bob shares a secret key k_B with the arbiter. Let (E_{k_A}, D_{k_A}) be the encryption and decryption functions corresponding to k_A and (E_{k_B}, D_{k_B}) be the encryption and decryption functions corresponding to k_B. Alice sends the message to Bob as follows:

1) Alice finds the hash of the message $h(\mathscr{M})$. She encrypts this hash value and a text X_A, that identifies that the sender of the message is Alice, to get $s = E_{k_A}(h(\mathscr{M})||X_A)$. This is the signature for the message. She sends this along with the message to tthe arbiter.

2) The arbiter knows k_A and therefore D_{k_A} and can compute

$$D_{k_A}(s) = D_{k_A}(E_{k_A}(h(\mathscr{M})||X_A)) = h(\mathscr{M}|)||X_A.$$

She extracts X_A and $h(\mathscr{M})$. From X_A, she knows that the sender is supposedly Alice. She can confirm this by comparing the value of $h(\mathscr{M})$ that she has extracted from the signature to the hash value she calculates from the message itself. If they are the same, she is satisified that the message is from Alice.

3) If $\mathscr{M}$ is a valid message for Bob, she concatenates the identity of Alice with the message $\mathscr{M}$ and signature s, encrypts it using Bob's key and sends it to Bob, i.e., she sends $E_{k_B}(X_A||\mathscr{M}||s)$. Bob can then decrypt this using D_{k_B} and obtain the identity of the sender and the message.

If Alice denies that she sent the message, Bob can send $E_{k_B}(X_A||\mathscr{M}||s)$ to the arbitrator. The arbiter can decrypt it using D_{k_B} and obtain X_A, the identity of the sender, the message $\mathscr{M}$ and the signature $E_{k_A}(h(\mathscr{M})||X_A)$. She compares the value $h(\mathscr{M})$ computed from $\mathscr{M}$ with the value extracted from the signature s. If they are equal, arbitrator concludes that the message has been sent by Alice. This is because Bob cannot tamper with s since he doesn't know k_A. In this method, Bob cannot verify directly that the message has been sent by Alice. He needs the services of the arbitrator for verification.

We now discuss some digital signature algorithsms. A good reference for our discussion is FIPS 186-4, [9]. The document gives three different algorithms, the RSA Digital Signature Algorithm, the Digital Signature Standard (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA). In this section, we will discuss the RSA Digital Signature Algorithm, the ElGamal algorithm and the Digital Signature Standard (DSA).

8.2.1 RSA Digital Signature Algorithm

In their paper [14], Rivest, Shamir and Adleman also suggest a method for digitally signing documents using their algorithm. We now describe the algorithm. This is similar to the RSA algorithm. Suppose Alice wants to sign and send a message. She does the following:

1) Alice generates two primes p and q as she does for the RSA algorithm and computes $n = pq$. She then chooses e_A such that $(e_A, \phi(n)) = 1$. She then computes the inverse d_A of e_A using extended euclidean algorithm. Her encryption and decryption functions are, respectively, $E_A(\mathscr{M}) = \mathscr{M}^{e_A} \pmod n$ and $D_A(\mathscr{M}) = \mathscr{M}^{d_A} \pmod n$. She then publishes (n, e_A).

2) To sign a message $\mathscr{M}$ she computes the hash $h(\mathscr{M})$ of the message using some standard Hash function like SHA for h and raises the hash of the message to power d_A, i.e. $s = D_A(h(\mathscr{M})) = h(\mathscr{M})^{d_A} \pmod n$. She then sends $(\mathscr{M}, s)$ to Bob.

To verify the signature, Bob downloads (n, e_A). He finds the value $h(\mathscr{M})$ and checks whether this is the same as $E_A(s) = s^{e_A} \equiv (h(\mathscr{M}))^{d_A e_A} \pmod n$. If $s^{e_A} \equiv h(\mathscr{M}) \pmod n$, the message has been sent by Alice.

Let us now look at an example.

Example 13: Suppose the public key, private key and the modulus of Alice are $e = 13$, $d = 37$ and $n = 77$. She publishes the values $(77, 7)$ on the web. Let us suppose that the message she wants to send is $\mathscr{M} = 17$. Then, she calculates $s = \mathscr{M}^{37}$. For the sake of simplicity, we assume that she raises the whole message to the power d_A instead raising the hash value $h(\mathscr{M})$ to the power d_A. Using repeated squaring algorithm we have the following:
Values at the end of iteration 1:

$$P = 17,\ b = 58,\ m = 18.$$

Values at the end of iteration 2:

$$P = 17,\ b = 53,\ m = 9.$$

Values at the end of iteration 3:

$$P = 54,\ b = 37,\ m = 4.$$

Values at the end of iteration 4:

$$P = 54,\ b = 60,\ m = 2.$$

Values at the end of iteration 5:

$$P = 54,\ b = 58,\ m = 1.$$

Values at the end of iteration 6:

$$P = 52,\ b = 53,\ m = 0.$$

So, 52 is the signature of the message. She sends the pair $(17, 52)$ to Bob. To check the signature, Bob downloads the values $n = 77$ and $e = 13$. He then checks whether $s^{e_A} = \mathscr{M}$. In this example, Bob calculates $52^{13} \pmod{77}$. Again, using repeated squaring algorithm, we have the following: Values at the end of iteration 1:

$$P = 52,\ b = 9,\ m = 6.$$

Values at the end of iteration 2:

$$P = 52,\ b = 4,\ m = 3.$$

Values at the end of iteration 3:

$$P = 54,\ b = 16,\ m = 1.$$

Values at the end of iteration 4:

$$P = 17,\ b = 25,\ m = 0.$$

Bob finds that $52^{13} \equiv 17 \pmod{77}$. So, he is convinced that the message was indeed sent by Alice.

* * *

Here is an exercise for you to check your understanding of RSA Signature Algorithm.

E1) Suppose Alice chooses $n = 221$, $e = 11$, $d = 35$.

a) Suppose, Alice wants to send the message $\mathscr{M} = 25$ to Bob. Find the signature of the message assuming that Alice signs the message and not the hash of the message.

b) Bob receives a message-signature pair $(82, 10)$. Verify the signature.

In Example 13 on the previous page, we assumed that Alice does not want to encrypt her message. Suppose, she wants to encrypt the message using RSA as well as sign her message using RSA digital signature algorithm.

Suppose she signs first and encrypts later; she first finds $s = D_A(h(\mathscr{M})) = h(\mathscr{M})^{d_A} \pmod{n}$ and then $c = E_B(\mathscr{M}) = \mathscr{M}^{e_B} \pmod{n}$. She sends (c, s) to Bob. Bob downloads (n_A, e_A), the modulus and public key of Alice. Then, using the value of e_A, he computes

$$E_A(s) = E_A(D_A(h(\mathscr{M}))) = h(\mathscr{M})^{d_A e_A} \pmod{n} = h(\mathscr{M}) \pmod{n}.$$

He then decrypts and finds the message $\mathscr{M} = D_B(c) = c^{d_B} = \mathscr{M}^{e_B d_B} \equiv \mathscr{M} \pmod{n}$. Then, he finds the hash value $h(\mathscr{M})$ and checks if $E_A(s) = h(\mathscr{M})$. If it is, Bob can be sure that the message was sent by Alice.

Now, Bob has a pair $(s, \mathscr{M})$ and if Alice denies that she sent the message, Bob can submit the pair $(s, \mathscr{M})$ to a judge. The judge checks whether $E_A(s) = h(\mathscr{M})$. If this is the case then $D_A(\mathscr{M})$ has to be s and Bob cannot produce $D_A(\mathscr{M})$ because Bob does not know D_A.

On the other hand, suppose Alice encrypts the message before signing. In this case, she computes

$$c = E_B(\mathscr{M}), s = D_A(h(c)) = D_A(h(E_B(\mathscr{M}))).$$

She sends (s, c) to Bob. Bob can recover the message by computing

$$D_B(c) = D_B(E_B(\mathscr{M})) = \mathscr{M}.$$

Bob can calculate $h(c)$ since he knows c. Bob checks whether $E_A(s) = h(c)$. If it is, he can be sure that message was sent by Alice.

In case Alice denies that she sent the message, Bob submits the pair $(s, \mathscr{M})$ to a judge. The judge finds $E_B(\mathscr{M}) = c$ and checks whether $h(c) = E_A(s)$. If this is the case, $D_A(h(c))$ must be s and Alice alone could have produced $s = D_A(h(c))$.

Now, the question is whether Alice should first encrypt the message and then sign it or sign first and encrypt afterwards. Both have problems as we will see presently.

Suppose Alice sends the sales plan $\mathscr{M}$ ="SALES PLAN" to her colleague Bob. Alice signs first and encrypts later. She computes the signature $s = D_A(h(\mathscr{M}))$. She then encrypts the message with Bob's public key to find $c = E_B(\mathscr{M})$. She then sends (c, s) to Bob. At a later time, suppose Bob has a fight with Alice and wants to make her appear guilty of divulging trade secrets to a rival, say, Charlie.

Bob can do the following: He knows D_B, so he can find $\mathscr{M} = D_B(E_B(\mathscr{M})) = \mathscr{M}$. He then encrypts it with the public key of Charlie and sends $c_1 = E_C(\mathscr{M})$ where E_C is the public encryption function of Charlie. He sends the pair (c_1, s) to Charlie and make it appear that Alice is the sender of the message. Charlie decrypts the message using his private decryption function D_C and find the message $\mathscr{M} = D_C(c_1)$. He will then find the message $\mathscr{M}$ and finds that $h(\mathscr{M}) = E_A(s)$. Charlie will be thus misled into thinking the message was sent by Alice to him. When it becomes known that the trade secret has been divulged to a rival, Alice will be blamed.

So, in this case, the signature authenticates the sender, but it cannot be used to determine who is the intended recipient. One simple solution is to include the name of the recipient in the message.

Suppose Alice encrypts first and signs later. Suppose Alice has an exciting new idea and she sends it to Bob, her Boss. She sends the message $\mathscr{M}$ ="MY IDEA" as follows: She computes $c = E_B(\mathscr{M})$ and then finds the $s = D_A(h(c))$ and sends the pair (s, c) to Bob.

Her rival Charlie may block the message and do the following in an attempt to claim credit for the idea. Since he knows E_A, he can find $E_A(s) = E_A(D_A(h(c))) = h(c)$. Then, he computes $s_1 = D_C(h(c))$ and sends (s_1, c) to Bob. Bob will be fooled into believing that Charlie is the discoverer of the new idea.

In this case, the signature can be used to determine who the recipient is, but it cannot be used to determine who the sender is. One simple solution to the issue is to include the name of the sender as a part of the message.

In general, the solution is to make these signing procedures a part of properly designed **cryptographic protocols**. A Cryptographic protocol is a step by step procedure for carrying out crytographic procedures like encryption or signing. Cryptographic protocols ensures that the cryptographic algorithms are used securely.

A thorough discussion of the issues involved is beyond the scope of this course. If you are interested in knowing more, see [3] and the references given there for a detailed discussion of the issues involved.

We end our discussion of RSA signature algorithm here. In the next sub-section, we will discuss a digital signature procedure due to ElGamal.

8.2.2 ElGamal Digital Signature

In this sub-section, we will discuss a digital signature scheme due to ElGamal. As in the case of RSA, the underlying idea behind the scheme is similar to that of RSA. Here also, the security of the scheme will depend on the difficulty in finding discrete logarithm.

Suppose Alice wants to digitally sign her messages using ElGamal Scheme. She chooses a prime p such that $p-1$ has at least one large prime factor. She then chooses a primitive root α, i.e. $\alpha \in \mathbb{Z}_p^*$ such that α generates the cyclic group $\mathbb{Z}_p^*$. She chooses another integer a such that $2 \leq a \leq p-2$. She finds $\beta = \alpha^a \pmod p$ and makes the values (p, α, β) public. She keeps the value of a secret.

Suppose she wants to sign a message $\mathscr{M}$. She chooses a secret integer k, $2 \leq k \leq p-2$ such that $(k, p-1) = 1$. She finds $r = \alpha^k \pmod p$. She can solve the congruence

$$kx \equiv \mathscr{M} - ar \pmod{p-1}. \tag{1}$$

for the value of x. Indeed, since k is coprime to $p-1$, the congruence Eqn. (1) has a unique solution s with $0 \leq s \leq p-2$. She sends the triple $(\mathscr{M}, r, s)$ to Bob.

To verify the signature, Bob has to check that

$$\alpha^{\mathscr{M}} = r^s \beta^r \pmod p. \tag{2}$$

Let us see why these two values should be equal. Note that since s is a solution to Eqn. (1), $sk \equiv \mathscr{M} - ar \mod p-1$. So, $sk - \mathscr{M} + ar$ is a multiple of $p-1$. Since α is an element of order $p-1$, it follows that

$$\alpha^{sk-\mathscr{M}+ar} \equiv 1 \pmod p.$$

We can rewrite the last equation as

$$\alpha^{\mathscr{M}} \equiv \alpha^{sk} \alpha^{ar} \tag{3}$$

But, $\alpha^{sk} = r^s$ since $r = \alpha^k$. Also, $\beta = \alpha^a$, so $\alpha^{ar} = \beta^r$. Substituting these values in Eqn. (3), we get Eqn. (2).

Note that, since only Alice knows the values of a, k and r only she could solve the congruence Eqn. (1).

Suppose Eve wants to forge the signature of Alice for a mess $\mathscr{M}_1$. She can choose some value k_1 and find $r_1 = \alpha^{k_1}$. If she has to convince Bob that the message was sent by Alice, she has to find a s such that

$$\alpha^{\mathscr{M}_1} \equiv r_1^s \beta^{r_1} \text{ or } r_1^s \equiv \alpha^{\mathscr{M}_1} \beta^{-r_1}.$$

For this, she has to solve the discrete logarithm problem

$$r_1^x = \gamma$$

in $\mathbb{Z}_p^*$, where we denote $\alpha^{\mathscr{M}_1}\beta^{-r_1}$ by γ. In general, this discrete logarithm problem is a problem that is difficult to solve if $p-1$ has a large prime factor. Thus, Eve will not be able to fake Alice's signature.

Let us now look at an example to understand the procedure.

Example 14: Suppose Alice chooses $p = 173$. Then 2 is a primitive root for for 173. So, she chooses $\alpha = 2$. Then, she chooses $a = 8$ and finds

$$\beta = 2^8 = 256 \equiv 83 \pmod{173}.$$

She makes the triple $(173, 2, 83)$ public.

Suppose Alice wants to send sign and send the message $\mathscr{M} = 16$ to Bob. Here, $172 = 4 \cdot 43$. She chooses $k = 9$ which is coprime to 172. So, $r = \alpha^k = 2^9 \equiv 166 \pmod{173}$. To find s, she has to solve equation $9x \equiv 16 - 8 \cdot 166 \pmod{172}$ or $9x \equiv 64 \pmod{172}$ since $16 - 8 \cdot 166 = -1312 \equiv 64 \pmod{172}$. Using extended euclidean algorithm, we get $(-19) \cdot 9 + 172 = 1$. So, $\overline{-19}$ is the inverse of $\overline{9}$ in $\mathbb{Z}_{172}^*$. We have $-19 \equiv 153 \pmod{172}$, i.e $\overline{9}^{-1}$ is $\overline{153}$ in $\mathbb{Z}_{172}^*$. Solving $9x \equiv 64 \pmod{172}$, we get $x \equiv 153 \cdot 64 \equiv 160 \pmod{172}$. She sends the value $(16, 166, 160)$ to Bob.

To verify the signature, Bob finds $\alpha^{\mathscr{M}} = 2^{16} \equiv 142 \pmod{173}$ to find the LHS value in Eqn. (2) on the preceding page.

Next, he finds the values in the RHS. First, he finds

$$r^s = 166^{160} \equiv 47 \pmod{173}.$$

He then finds

$$\beta^r = 83^{166} \equiv 84 \pmod{173}$$

Mutliplying r^s and β^r, we get

$$47 \cdot 84 = 3948 \equiv 142 \pmod{173}$$

and this agrees with LHS value. So, Bob is convinced that the message was sent by Alice.

* * *

Here is an exercise to check your understanding of the ElGamal signature scheme.

E2) Suppose Alice chooses $p = 139$. Then, 2 is a primitive root so, she chooses $\alpha = 2$. She chooses $a = 9$ and finds $\beta = \alpha^a = 2^9 \equiv 95 \pmod{139}$. She makes the triple $(139, 2, 95)$ public. Suppose she wants to send the message 25 to Bob.

i) Find the signature for the message if she chooses $k = 11$.

ii) Explain in detail how Bob will verify the signature.

Next, we discuss some precautions that Alice has to take to make sure that the Eve is not able to forge her signature. One precaution is that Alice should not choose the same value of r for signing two different messages. Let us see what happens when Alice does this.

Suppose Aice signs two messages, $\mathscr{M}_1$ and $\mathscr{M}_2$ with the same value of r and hence the same value of k. Eve can find this out because Alice sends the value of r to Bob along with the message.

Since Alice uses the same value of r, we have

$$\mathscr{M}_1 - s_1 k \equiv ar \equiv \mathscr{M}_2 - s_2 k \mod p-1 \text{ or } s_1 k - s_2 k \equiv \mathscr{M}_1 - \mathscr{M}_2 \pmod{p-1}$$

Since s_1, s_2, $\mathscr{M}_1$ and $\mathscr{M}_2$ are known, Eve can solve the congruence

$$(s_1 - s_2)k \equiv \mathscr{M}_1 - \mathscr{M}_2 \pmod{p-1}. \tag{4}$$

for k.

Recall how we solve such congruences. Suppose, we want to solve the congruence

$$ax \equiv b \pmod{n} \tag{5}$$

and $(a, n) = d$. Then, we know from Unit 6 in MMT-003 that Eqn. (5) has a solution if and only if $d \mid b$. We let $a' = \frac{a}{d}$, and $b' = \frac{b}{d}$ and $n' = \frac{n}{d}$. We consider the congruence

$$a'x \equiv b' \pmod{n'} \tag{6}$$

Since $(a', n') = 1$, Eqn. (6) has a unique solution x_0 and we can find it by finding the inverse $\overline{a'}^{-1}$ of $\overline{a'}$ in $\mathbb{Z}_{n'}$ using extended euclidean algorithm and multiplying both sides of Eqn. (6) by $\overline{a'}^{-1}$. Then, all the solutions of Eqn. (5) are given by $x_0 + i\frac{n}{d}$, $0 \leq i \leq d-1$. So, Eqn. (5) has d solutions.

Suppose $(s_1 - s_2, p-1) = d$. Then, there are d solutions, $k_1, k_2, \ldots, k_d$, to Eqn. (4). Usually, d is small. Eve finds α^{k_1}, α^{k_2}, etc and checks for which k_i we have $\alpha^{k_i} = r$. Eve can thus find the value of k.

Once Eve finds k, she solves any one of the equations

$$ar \equiv \mathscr{M}_1 - s_1 k \pmod{p-1} \tag{7}$$

or

$$ar \equiv \mathscr{M}_2 - s_2 k \pmod{p-1} \tag{8}$$

for a. For the sake of definiteness, let us assume that Eve solve Eqn. (8). Again, if $\ell = (r, p-1)$, there are ℓ solutions to Eqn. (7), say a_1, a_2, … a_ℓ. Eve finds α^{a_1}, α^{a_2}, …, α^{a_ℓ} and checks which a_i gives $\alpha^{a_i} = \beta$. Thus, she can find the value of a. Once she finds a, Eve can easily forge Alice's signature. Let us now look at an example.

Example 15: Suppose Alice chooses $p = 173$, $\alpha = 2$, $a = 8$, $\beta = 83$, $k = 9$, $r = 166$ as in Example 14 on the preceding page. Suppose the message is $\mathscr{M}_1 = 16$. Then, we saw in Example 14 on the previous page that the signature for the message $\mathscr{M}_1 = 16$ is $s_1 = 160$.

Suppose Alice uses the same value $k = 9$ and signs the message $\mathscr{M}_2 = 40$. Then, Alice has to solve the equation

$$9x \equiv 40 - 8 \cdot 166 \equiv 88 \pmod{172}$$

to find the signature s_2. As before, in $\mathbb{Z}_{172}$, $\overline{9}^{-1}$ is $\overline{153}$. So,

$$x \equiv 153 \cdot 88 \equiv 48 \pmod{172}.$$

In other words, the signature for $\mathscr{M}_2$ is $s_2 \equiv 48 \pmod{172}$.

Eve notices that Alice has used the same r, so she knows that Alice has used the same value of k. To find the value of k she has to solve for Eqn. (4) on the facing pagefor k. Substituting the values of s_1, s_2, $\mathscr{M}_1$ and $\mathscr{M}_2$ in Eqn. (4) on the preceding page, she has to solve the congruence

$$(160-48)k \equiv -24 \pmod{172} \text{ or } 112k \equiv 148 \pmod{172} \tag{9}$$

Comparing Eqn. (5) on the facing page, we have $a = 112$, $b = 148$, $n = 172$ and $d = (112, 172) = 4$. So, we have $a' = \frac{112}{4} = 28$, $b' = \frac{148}{4} = 37$, $n' = \frac{172}{4} = 43$. Comparing with Eqn. (6) on the preceding page, we first solve the congruence

$$28x \equiv 37 \pmod{43}. \tag{10}$$

By extended euclidean algorithm, we get $20 \cdot 28 + (-13) \cdot 43 = 1$. So, $\overline{28}^{-1} = \overline{20}$ in $\mathbb{Z}_{43}$. So,

$$x \equiv 20 \cdot 37 \equiv 9 \pmod{43}$$

So, the solutions are

$$9 + 0 \cdot 43, 9 + 1 \cdot 43, 9 + 2 \cdot 43, 9 + 3 \cdot 43,$$

i.e 9, 52, 95 and 138. Finding $2^9 \pmod{173}$, $2^{52} \pmod{173}$, $2^{95} \pmod{173}$, $2^{138} \pmod{173}$, Eve finds that $2^9 \equiv 166 \pmod{173}$. So, she has the value of k.

Next, she solves the congruence Eqn. (7) on the facing page with $\mathscr{M}_1 = 16$, $s_1 = 160$, $k = 9$. In other words, she has to solve the congruence

$$166a \equiv 16 - 160 \cdot 9 \equiv 124 \pmod{172} \tag{11}$$

Again, comparing with Eqn. (5) on the preceding page, we see that $r = 166$, $b = 124$, $n = 172$ and $d = (166, 172) = 2$. So,

$$a_1 = \frac{166}{2} = 83, b_1 = \frac{124}{2} = 121, n_1 = \frac{172}{2} = 86.$$

Comparing with Eqn. (6) on the facing page, Eve has to solve the congruence

$$83x \equiv 62 \pmod{86} \tag{12}$$

By extended euclidean algorithm, we get $(-29) \cdot 83 + 28 \cdot 86 = 1$. Further, $-29 \equiv 57 \pmod{86}$. Hence $\overline{83}^{-1}$ is $\overline{57}$ in $\mathbb{Z}_{86}$. So, the solution to Eqn. (12) is

$$x \equiv 57 \cdot 62 \equiv 8 \pmod{86}$$

So, the solutions to Eqn. (11) are 8 and $8 + 86 = 94$. Eve easily checks that $2^8 \equiv 83 = \beta \pmod{173}$ and so, she finds that $a = 8$. Eve can now forge Alice's signature.

* * *

Try the next exercise to see if you have understood Example 15 on the preceding page.

E3) Alice has published the values $p = 173$, $\alpha = 5$, $\beta = 84$. She sends the signed messages $(25, 26, 101)$ and $(35, 26, 167)$ to Bob. Find the value of a.

In the next section, we will discuss the Digital Signature Standard, another digital signature algorithm based on discrete logarithm. This removes some shortcomings that are in the ElGamal signature scheme.

8.2.3 Digital Signature Standard(DSS)

The Digital Signature Standard (DSS) or Digital Signature Algorithm (DSA) was proposed by NIST (National Institute of Standards and Technology) in 1991 and was adopted in 1994. NIST continues to release updates on the use of algorithm from time to time. At the time of writing this Unit, the last released version of the algorithm is Federal Information Processing Standards (FIPS) 186-4. [9]. This was released in July 2013. To keep the discussion simple, we discuss only a special case of the algorithm. For a general version of the algorithm, you can refer to FIPS 186-4 which is available for download from the NIST website.

We assume that we will be signing the hash of the message and that the hash function we use produces an output of length 160 bits. To set up the signature scheme, Alice does the following:

1) She chooses a prime q of size 160 bits, i.e. a prime q such that $2^{159} < q < 2^{160}$. She then chooses another prime p which is of size 1024 bits such that $q \mid p-1$. You can refer to [9] for the details regarding choice of primes.

2) She chooses a primitive root g $\pmod p$ and sets $\alpha = g^{(p-1)/q} \pmod p$. Then, α has order q in $\mathbb{Z}_p^*$.

3) She chooses a secret value a and computes $\beta = \alpha^a$. She then makes the values (p, q, α, β) public.

As we did in RSA digital signature scheme, here also we will assume that Alice signs the message rather than the hash of the message. Suppose she wants to sign a message. She calculates the hash and gets 160-bit number $\mathscr{M}$. She generates a signature for the message as follows:

1) She chooses a secret value k such that $0 < k < q-1$ and finds $r = \left(\alpha^k \pmod p\right) \pmod q$.

2) She then solves the equation

$$kx \equiv \mathscr{M} + ar \pmod q. \tag{13}$$

The solution to Eqn. (13) is the signature s to the message. She sends (r, s) along with the message to Bob.

Bob verifies the message as follows:

1) Bob calculates the hash of the message using the same hash function used by Alice and gets the value $\mathscr{M}$.

2) Bob finds $u_1 \equiv s^{-1}\mathscr{M} \pmod q$ and $u_2 \equiv s^{-1}r \pmod q$.

3) He then finds $v \equiv (\alpha^{u_1}\beta^{u_2} \pmod p) \pmod q$. If $v = r$ the signature is valid.

Let us see how this proceduce works. Note that, from Eqn. (13), we get

$$k \equiv s^{-1}(\mathscr{M} + ar) \pmod q \tag{14}$$

So, $q \mid (k - s^{-1}(\mathscr{M} + ar))$. Since α has order q in $\mathbb{Z}_p^*$, we have

$$\alpha^{k - s^{-1}(\mathscr{M}+ar)} \equiv 1 \pmod p$$
$$\text{or } \alpha^k \equiv \alpha^{s^{-1}\mathscr{M}}\alpha^{s^{-1}ar} \pmod p$$

Reducing both sides (mod q), we get

$$\left(\alpha^{k} (\text{mod } p)\right) (\text{mod } q) \equiv \left(\alpha^{s^{-1}\mathscr{M}} \alpha^{s^{-1}ar} \pmod{p}\right) \pmod{q} \tag{15}$$

Note that, in equation Eqn. (13) on the facing page, we do not directly reduce $\alpha^{k} \pmod{q}$ to get r. First, we reduce $\alpha^{k} \pmod{p}$ and then reduce it mod q. So, we have to proceed similarly when we calculate the RHS of equation Eqn. (15). Note that $\alpha^{k} \equiv r \pmod{p}$, so the value of the RHS in Eqn. (15) is r. On the other hand,

$$\alpha^{u_1}\beta^{u_2} = \alpha^{s^{-1}\mathscr{M}} (\alpha^{a})^{s^{-1}r} = \alpha^{s^{-1}(\mathscr{M}+ar)} \equiv \alpha^{s^{-1}\mathscr{M}}\beta^{s^{-1}r} \pmod{p}$$

Reducing mod q, we get that

$$v \equiv (\alpha^{u_1}\beta^{u_2} \pmod{p}) \pmod{q}$$

As we have already seen, the value of LHS is r, so $v = r$. Let us now look at an example.

Example 16: To keep the example simple, instead of taking a 160 bit prime q, we will suppose that Alice chooses $q = 43$ and $p = 173$. Then, she takes $g = 2$ and

$$\alpha = g^{(p-1)/q} = 2^{172/43} = 2^4 \equiv 16 \pmod{172}.$$

She then chooses $a = 7$ and finds $\beta = \alpha^{a} = 16^7 \equiv 6 \pmod{173}$. She then publishes $(173, 43, 16, 6)$.

Suppose the hash of the message she wants to sign is 30. She chooses $k = 9$ and finds $r = \alpha^{k} = 16^9 \equiv 152 \pmod{173}$ and $152 \equiv 23 \pmod{43}$, so $r = 23$. She solves the equation

$$9s \equiv 30 + 7 \cdot 23 \equiv 19 \pmod{43}$$

for s. We have $\overline{9}^{-1} = \overline{24}$ in $\mathbb{Z}_{43}^{-1}$. So,

$$s \equiv 19 \cdot 24 \equiv 26 \pmod{43}$$

She sends $(23, 26)$ along with the message.

Bob then calculates the hash of the message and finds $\mathscr{M} = 30$. He finds that $s^{-1} = \overline{26}^{-1} = \overline{5}$ in $\mathbb{Z}_{43}$. He then finds

$$u_1 = s^{-1}\mathscr{M} = 5 \cdot 30 = 150 \equiv 21 \pmod{43} \text{ and } u_2 = s^{-1}r = 5 \cdot 23 \equiv 29 \pmod{43}.$$

Bob finds

$$\alpha^{u_1}\beta^{u_2} = 16^{21}6^{29} \equiv 152 \pmod{172}$$

and $152 \equiv 23 \pmod{43} = r$. So, the signature is verified.

* * *

Here is an exercise for you to check your understanding of digital signature standard.

E4) a) Suppose Alice chooses $p = 167$, $q = 83$, $g = 5$ which is a primitive root (mod p). She chooses $\alpha = g^{(p-1)/q} = 5^{166/83} = 5^2 = 25$. She chooses $a = 8$ and gets $\beta = 25^8 \equiv 144 \pmod{167}$. Assume that she chooses $k = 13$ and the hash of the message she wants to sign is 39. Find the signature of the message if Alice Digital Signature Standard.

b) Assume that Alice has published the values $(p, q, \alpha, \beta) = (167, 83, 25, 144)$. She sends a message to Bob along with the pair $(r, s) = (31, 61)$ which was created using Digital Signature Algorithm. Bob finds that the hash of the message is 47. Explain how Bob will check whether the message has been sent by Alice or not.

We conclude this section with some remarks.

1) Note that, in ElGamal signature scheme, we need to perform three exponentiations, but in DSS, we need to perform only two exponentiations.

2) Unlike the Elgamal signature scheme, we can use the same r again. Suppose h_1 is the hash of the first message and h_2 is the hash of the second message, both signed with the same value of r. Suppose further, that the signature of the first message is s_1 and the signature of the second message is s_2. Then,

$$s_1 k - h_1 \equiv ar = s_2 k - h_2 \pmod{q}$$

Then, to find k, Eve has to solve

$$k(s_1 - s_2) \equiv (h_1 - h_2) \pmod{q}$$

for k. After solving (mod q) and Eve gets a value $k_0 \pmod{q}$. There are $2^{512-160} = 2^{342}$ numbers modulo p which reduce to k_0 modulo q. Eve has to find the value α^i for each of these 2^{342} values of i and see which of these values gives r $(\text{mod } p)$ to find the value of k. This is computationally infeasible.

8.3 SUMMARY

In this Unit we have discussed

1) the definition of digital signature;
2) the functionalities provided digital signatures;
3) direct and arbitrated signature schemes;
4) how to sign a message using RSA algorithm and verify a signature created with RSA algorithm;
5) how to sign a message using ElGamal algorithm and verify the signature created using ElGamal algorithm;and
6) how to sign a message using Digital Signature Standard and verify the signature created using Digital Signature Standard.

8.4 SOLUTIONS/ANSWERS

E1) a) She has to find $d_A(25) = 25^{35} \pmod{221}$. Using repeated squaring algorithm, we have the following:
Values at the end of iteration 1:

$$P = 25,\ b = 183,\ m = 17$$

Values at the end of iteration 2:

$$P = 155,\ b = 118,\ m = 8$$

Values at the end of iteration 3:

$$P = 155,\ b = 1,\ m = 4$$

Values at the end of iteration 4:

$$P = 155,\ b = 1,\ m = 2$$

Values at the end of iteration 5:

$$P = 155,\ b = 1,\ m = 1$$

Values at the end of iteration 6:

$$P = 155,\ b = 1,\ m = 0$$

So, we have $25^{35} \equiv 155 \pmod{221}$.

b) We find $82^{11} \pmod{221}$ using repeated squaring algorithm:
Values at the end of iteration 1:

$$P = 82,\ b = 94,\ m = 5$$

Values at the end of iteration 2:

$$P = 194,\ b = 217,\ m = 2$$

Values at the end of iteration 3:

$$P = 194,\ b = 16,\ m = 1$$

Values at the end of iteration 4:

$$P = 10,\ b = 35,\ m = 0$$

So, we have $82^{11} \equiv 10 \pmod{221}$. So, the signature is correct.

E2) i) We have

$$r = \alpha^k = 2^{11} = 102 \pmod{139}$$

To find the signature s she has to solve the equation

$$11x = 25 - 9.102 \equiv 73 \pmod{138}$$

Using extended euclidean algorithm, we get $(-25)\cdot 11 + 2\cdot 138 = 1$, and $-25 \equiv 113 \pmod{138}$. So, inverse of $\overline{11}$ in $\mathbb{Z}_{138}^*$ is 113. Solving for x, we get $x \equiv 113 \cdot 73 \equiv 107 \pmod{138}$. So, the signature is 107.

ii) To verify the signature Bob first finds

$$\alpha^{\mathscr{M}} = 2^{25} \equiv 110 \pmod{139}.$$

He then finds $r^s = 102^{107} \equiv 123 \pmod{139}$. He also finds $\beta^r = 95^{102} \equiv 80 \pmod{139}$. Next, he finds $\beta^r r^s = 80 \cdot 123 \equiv 110 \pmod{139}$ and this is the same as the value of $\alpha^{\mathscr{M}}$. So, the message must have been sent by Alice.

E3) We have to solve the congruence

$$(101 - 167)k \equiv 25 - 35 \pmod{172} \text{ or } 66k \equiv 10 \pmod{172} \qquad (16)$$

We have $(66, 172) = 2$ and 2 divides 10. So, we have to solve the congruence

$$33k \equiv 5 \pmod{86}$$

Solving, we get $k \equiv 21 \pmod{172}$. So, the solutions to Eqn. (16) on the previous page are 21 and $21 + 86 = 107$. Finding 5^{21} and 5^{107}, we find that $5^{21} \equiv 26 \pmod{173}$, so $k = 21$.

To find a, we have to solve

$$26a \equiv 140 \pmod{172} \tag{17}$$

We have $(26, 172) = 2$ and 2 divides 140. So, we have to solve the congruence

$$13a \equiv 70 \pmod{86}$$

Solving, we get $a = 12$. so, the solution to Eqn. (17) are 12, $12 + 86 = 98$.

We see that $\alpha^{12} \equiv 84 \pmod{173}$, so $a = 12$.

E4) a) We have $r = \alpha^k = 25^{13} = 114 \pmod{167}$ and $114 \equiv 31 \pmod{83}$, so $r = 31$. We have to solve the equation

$$13x \equiv 39 + 8 \cdot 31 \equiv 38 \pmod{83}$$

Using extended euclidean algorithm, we get $\overline{13}^{-1}$ is $\overline{32}$ in $\mathbb{Z}_{83}$, so

$$x \equiv 32 \cdot 38 \equiv 54 \pmod{83}.$$

So, the signature of the message is $s = 54$.

b) Bob finds that $s^{-1} = \overline{61}^{-1}$ in $\mathbb{Z}_{83}^*$ is $\overline{49}$. He finds that $u_1 = 49 \cdot 47 \equiv 62 \pmod{83}$ and $u_2 = 49 \cdot 31 \equiv 25 \pmod{83}$. He then finds $\alpha^{u_1} = 25^{62} \equiv 157 \pmod{167}$ and $\beta^{u_2} \equiv 144^{25} \equiv 22 \pmod{167}$. Further, $\alpha^{u_1}\beta^{u_2} = 157.22 \equiv 114 \pmod{167}$ and $114 \equiv 31 \pmod{83}$ and this is equal to the value of r. So, Bob is convinced that the message was sent by Alice.