Assignment Elective - III D3

Aim:

To understand the concept of DSA Algorithm.

Problem Statement:

Write a program to produce a DSA signature using parameter tuple(p,q,g,y), long term key pair and a message digest.

Input:

User message

Output:

tuple(p,q,g,y),x, r and s

Theory:

Digital Signature:

A digital signature is an authentication mechanism that enables the creator of a message to attach a code that acts as a signature. Typically the signature is formed by taking the hash of the message and encrypting the message with the creator's private key.The signature guarantees the source and integrity of the message.

Digital Signatures:

Generic Model:

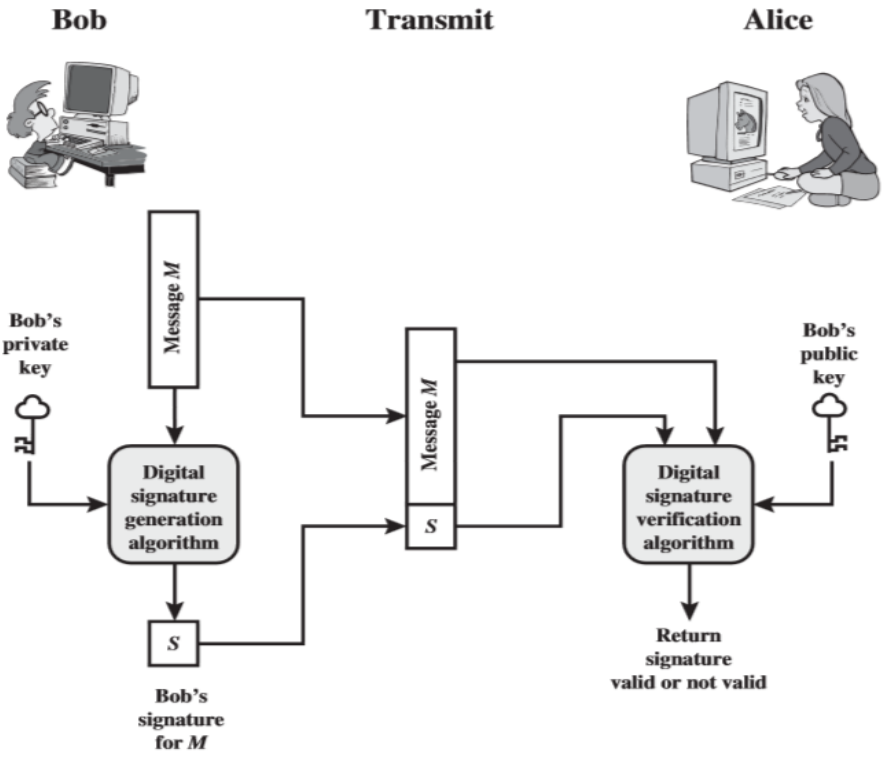

Figure 1: Generic Model of Digital Signature Proces

The Digital Signature Algorithm:

Figure 2 summarizes the algorithm.There are three parameters that are public and can be common to a group of users.A 160-bit prime number is chosen.Next, a prime number is selected with a length between 512 and 1024 bits such that divides (p-1). Finally, g is chosen to be of the form $h^{(p-1)/q} mod p$, where h is an integer between 1 and (p-1) with the restriction that must be greater than 1.

With these numbers in hand, each user selects a private key and generates a public key.The private key x must be a number from 1 to (q-1) and

should be chosen randomly or pseudorandomly.The public key is calculated from the private key as y = $g^x mod p$.The calculation of y given x is relatively straightforward. However, given the public key y,it is believed to be computationally infeasible to determine x, which is the discrete logarithm of y to the base g,mod p. To create a signature, a user calculates two quantities,

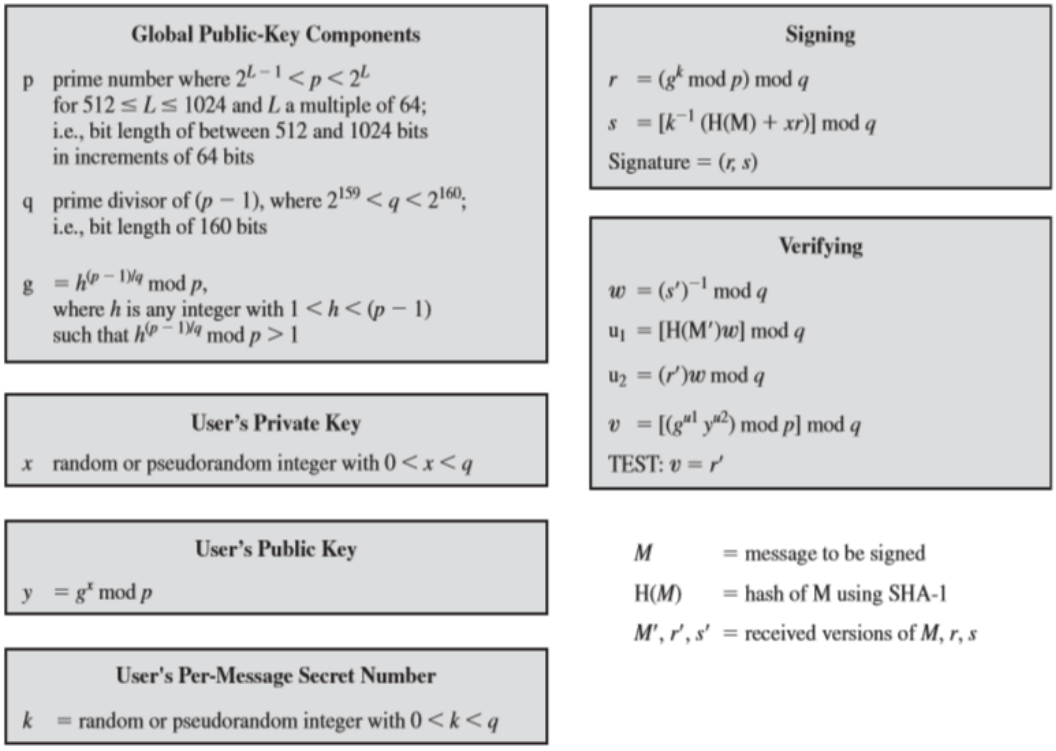

Figure 2: The Digital Signature Algorithm (DSA)

r and s, that are functions of the public key components (p,q,g), the user's private key (x), the hash code of the message H(M), and an additional integer k that should be generated randomly or pseudorandomly and be unique for each signing.

At the receiving end, verification is performed using the formulas shown in Figure 2.The receiver generates a quantity v that is a function of the public key components, the sender's public key, and the hash code of the incoming message. If this quantity matches the r component of the signature,then the signature is validated.

Figure 3 depicts the functions of signing and verifying.

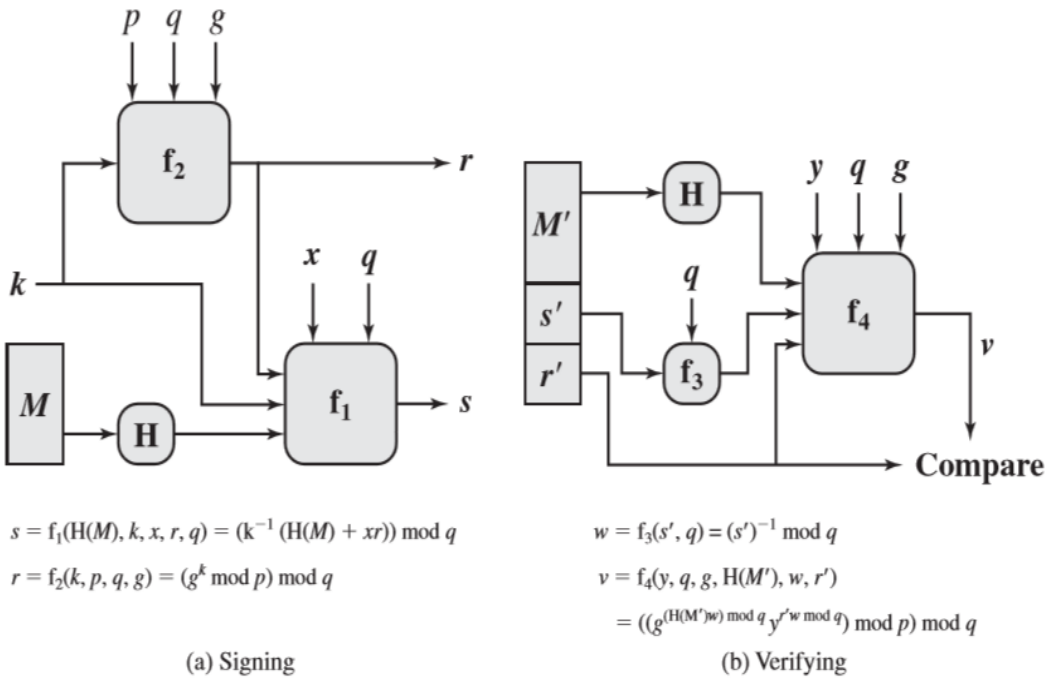

Figure 3: The Digital Signature Algorithm (DSA)

The structure of the algorithm,as revealed in Figure 3,is quite interesting. Note that the test at the end is on the value r,which does not depend on the message at all.Instead, r is a function of k and the three global public-key components.The multiplicative inverse of k (mod q) is passed to a function that also has as inputs the message hash code and the user's private key.The structure of this function is such that the receiver can recover r using the incoming message and signature,the public key of the user,and the global public key.It is certainly not obvious from Figure 2 or Figure 3 that such a scheme would work.

Given the difficulty of taking discrete logarithms, it is infeasible for an opponent to recover k from r or to recover x from s.

Another point worth noting is that the only computationally demanding task in signature generation is the exponential calculation $g^k modp$ .Because this value does not depend on the message to be signed, it can be computed

ahead of time. Indeed, a user could precalculate a number of values of r to be used to sign documents as needed.The only other somewhat demanding task is the determination of a multiplicative inverse, K^{-1}. Again, a number of these values can be precalculated.

Mathematical Modeling:

Let S be the system that represents the Distributed Booth's Algorithm.

Initially,

$$\mathbf{S} = \{\phi\}$$

Let,

$$\mathbf{S} = \{\mathbf{I},\ \mathbf{O},\ \mathbf{F}\}$$

Where,
I - Represents Input set
O - Represents Output set
F - Represents Function set

Input set - I:

$$\mathbf{I} = \{\mathbf{M}\}$$

Where,

- M - Represents the input message.

Output set - O:

$$\mathbf{O} = \{\mathbf{P},\mathbf{Q},\mathbf{G},\mathbf{Y},\mathbf{X},\mathbf{R},\mathbf{S}\}$$

Where,

- tuple(P,Q,G) - Represents Global Public-Key Components
- Y - Represents User's Public Key
- X - Represents User's Private Key
- tuple(R,S) - Represents Signature

Function Set - F:

$$\mathbf{F} = \{F_1, F_2, F_3\}$$

Where,

- F_1 - Represents a function for generating Public Key components.
  $F_1()$ -> {P,Q,G,Y}
- F_2 - Represents a function for generating Signature.
  F_2(R,D) -> {S}
  - R - Represents Signature component.
  - D - Represents the input data.
  - S - Represents generated signature.
- F_3 - Represents a function for generating Signature.
  F_3(D,R,S) -> {A,U}
  - R - Represents Signature component.
  - D - Represents the input data.
  - S - Represents generated signature.
  - A - Represents valid result.
  - U - Represents invalid result.

Conclusion:

Thus, we have studied and implemented DSA Algorithm.