**Illustration of HASH LSB process**

Consider a RGB pixel value of the cover frame as below

R: 10110111

G: 10010100

B: 11001001

And a byte of message to be inserted in LSB as: 10001001

LSB is lowest bit in a series of binary numbers, so in this case for R it will be 1, 0 for G and 1 for B.

The proposed technique is applied in four lowest LSBs in each pixel value. So the LSBs for the above RGB values are:

R: 0111

G: 0100

B: 1001

he message is embedded in groups of 3, 3 and 2 in the respective RGB LSBs positions. The positions are obtained from the hash function given by the equation k=p%n. The value of n number of bits of LSB is 4. Using the hash function let the position of insertion k returned for a particular iteration are,

k = 1, 2, 3 for R.

k = 4, 1, 2 for G

k = 3, 4 for B

Considering the above positions of insertion, the bits from the message are inserted in four LSB

positions and resulting RGB pixel value are as given below:

R: 10111001

G: 10011000

B: 11001001

Thus all the eight bits of the message are embedded in three bytes and number of bits actually changed

is five out of twenty four bits. Further these five bits are randomly distributed which increases the

robustness of the scheme. On decoding the message, the valid user follows the reverse step. As the hash function is known to the intended the user, it calculates the k values to get the position of insertion. Taking the same embedded RGB value as above,

R: 10111001

G: 10011000

B: 11001001

The hash function will return the following k values for this particular iteration.

k = 1,2,3 for R.

k = 4,1,2 for G

k = 3,4 for B

Using these k values which represent the four LSB positions, the data of the secret message is found as

below,10001001,Which is same as the data of secret message