Q.1)

→ a) So in the given Question we have to find the possible worst case and expected case exhaustive search attempts made by an attackes.

Here double encryption DES is used and Let us assume that attacker has a one set of plain text as well as cipher text which he will use.

Above Question assumes that key K is b bits long. i.e. K_1 and K_2 are b bits long.

So, to Find the key in DES by exhaustive search is 2^n computations.

Exhaustive Search in double DES for worst case will generate 2^n encrypted values That is For key K_1 and then the 2^n encrypted values are futher encrypted by using other key i.e. K_2

So, this result in combination of 2^n exhaustive Searches.

Therefore Worst case exhaustive Search is 2^{2b}

In double encryption DES attacker then look for a collision through "meet in the middle attack", where attacker encrypts on one side and decrypts on the other side and meet in the middle.

So attacker will encrypt from one end as mentioned above would take 2^b combinations and also decrypt from the other cipher end which would take another 2^b combinations.

∴ Expected case exhaustive search $= 2^b + 2^b$

$= 2(2^b)$

$= 2^{b+1}$

∴ Worst case exhaustive search is 2^{2b}

∴ Expected case exhaustive search is 2^{b+1}

→ b)

By using 3 DES encryption with three independent keys i.e K_1, K_2, K_3 transforms 64 bit plaintext block which is P into ciphertext

$$C = E_{K_3}(D_{K_2}(E_{K_1}(P)))$$

Here E and D are encryption and decryption respectively.

In 3 DES attacker then look for a collision through "meet in the middle attack" where attacker meet in the middle attack against this form of DES assumes 3 different know plaintext, cipher text pairs i.e. (P_i, C_i) and works as follow:

i) So, for each 2^{56} values of key K1, compute $R = E_{C1}(P_0)$ and from the results build in memory a data structure that allows quickly finding the values of key K1 yielding a particular 64 bit result.

ii) After that for each 2^{112} values of K2 and K3 respectively compute $R = E_{K2}(D_{C3}(C_0))$ and try to find that whether R in the memory table.

So above is the theoretically way to perform an attack although there is a way to perform an attack it is practically impossible to work it will take.

It will take 24 billion years if we perform each subset of 2 about 2^{112} times performming one memory lookup per loop.

Q. 2)

→ 1) When using AES algorithm observe that the 64-byte message will result in 4 plaintext blocks, since AES algo is about 128 bit i.e 16 byte block cipher

So a single bit error in ciphertext can only affect one of the 4 blocks of ciphertext. A corrupted ciphertext block will always cause its corresponding plaintext block to fail to decrypt properly.

2) With CBC, the ciphertext block is also used to decrypt the next plaintext block via XOR'ing with AES decryption of the next ciphertext block.

However, the effects of the corrupted ciphertext block will not extend further, given that the other ciphertext blocks do not have errors. Thus, at the most 2 of 4 blocks will decrypt incorrectly. If the transmitted IV has a single-bit error, then that will result in an incorrect decryption of the first block of plaintext. Since it's computed as the XOR of the IV with the AES decryption of the 1st cipherblock.

3) However, neither the IV nor the first block of plaintext is used in subsequent decryptions, so in this scenario only one of the 4 blocks will decrypt incorrectly.

If the key has a single bit error, then when decrypting all the AES operations will fail to produce the correct value.
So every block will decrypt incorrectly.

Q.3)

```
declare Com_no = 0;
define as MAX 51
define datatype of type unsigned long int
declare key length
define encrypted_text
define decrypted_text
declare a, b and seed of type big
declare key[MAX], msg[500]

Void Symmetric()
  {
    //take the input of seed from user and
     store it in variable
     While (1)
      {
        print " Enter your choice :
        print 1. Encryption 2. Decryption
      }// Enter choice.
      ch = getch();
      // IF choice is 1 then Encryption
      // Else if choice is 2 then Decryption
      // Else display wrong choice.
  }
}

Void encode()
  {
    declare i of type int
    declare ch of type char
```

Get user message and store in array msg[]
call generate - key()

// Display encoded message

```
for (i=0; msg[i] ≠0 ; i++)
 {
    ch = (msg[i]) XOR (key[i % (keylength / sizeof (char))]);
    display ch
    call function char_to_bin (ch);
 }
  display encrypted text
}

Void decode ()
   {
     declare i of type int;
     i=0;
     declare ch and str[10] of type char;
     call the function generate_key()
     display received message;
     While ( encrypted_text[i] ≠ 0)
      {
        for ( int j=0; j<=8; j++)
        {
          str[j] = encrypted_text[i++]
        }
        ch = bin_to_dec[str);
        display ch;
```

```
        ch = (ch) XOR (key [ctr % (keylength / sizeof(char))]
        msg [ctr] = ch;
    }
    Display decoded message
}

Void generate_key()
{
    i = int;
    i = a = 0
    while (i < MAX-1)
    {
        if (not a)
        {
            b = rand_key (b)
            Set a = b
        }
        key [i++] = a % 100
        Set a = a/100
    }
    key [i++] = NULL
    keylength = i
    for (i = 0; i < keylength ; i++)
        display key [i]
}

Function rand_key (big p)
{
    Seed = Seed % 1015;
    next = (Seed) XOR (com_no)
    p = result;
```

```
next = next%15;
result = (result) XOR (next

result = result + com_no
result = result %1015;
Set Seed = next;
return result;
}
```

Q.4)

→

a)

i) So, the AES key 'k' is required to decrypt the message i.e m

ii) Wallance's public key is itself encrypted by using kw.

This ensures that even if the attacker has the codeword, it can't get a hold of the key until they have access to Vw, which is only available to Wallace.

iii) So, this mechanism state that only wallance can unlock the key to the encrypted message and this it protects Gromits confidentiality of the message.

Therefore this protocol ensure that the confidentiality of Gromit's message.

b)

i) Yes, the protocol ensure authentication as well as data integrity for every text message wallence receives.

Since it checks the signed tuple (c, c') against the tuple it receives i.e (c, c')

ii) Here, tuple (c, c') is signed with the private key Va and sent to wallace's and this ensures that wallace knows when he decrypts the message that it sent by Gromit because only if the tuple is encrypted with the Gromits unique private key can it be decrypted by its public key Kg.

iii) So, Integrity of the message is ensured by the comparison of the two tuples to ensure that there has been no data loss else it is rejected.

c)

i) No, in this type of situation it can be easily comprised. This situation can potentially be reliable if both the Financial advisor (Service) and the third party auto Financial heading Service are trust worthy.

ii) If gromit trusts both the parties even then if due to data leak on the third party Service all the data shared with third party will became Vulnerable.

iii) At the stage of the book up third party. The output of the protocol is going to be the text message with sensitive information it can be manipulated before sending it off to the third party Service. ie changing the 1000 to 10,000 or changing the target of transferring money. The method using which this message is forwarded, the level of Security is also a concern. that makes Gromit's account Vulnerable to on MIM attack.

iv) On the other hand. his data can be leaked by Gromit to any other party without Grants knowledge. leaving it open for cyber theft