HACETTEPE UNIVERSITY

BBM456

Computer and Network Security

Homework

#1

Fatma Çiğdem Tosun

216-----

Q: Why not Double-DES? Explain.

For the multiple encryption approach, the simplest form of multiple encryption has two encryption stages and two keys, and it is called Double-DES.

2DES encryption cipher is sequentially applied where the key $K_1$is used first and then the key $K_2$is used for the following DES cipher.

To reverse the encryption, Double-DES decryption uses key K_2first and then the key K_1after.

$$C = Enc(K_2, Enc(K_1, P))$$

$$P = Dec(K_1, Dec(K_2, C))$$

This can also be expressed in math as it is shown and in the mathematical expression those that are inside the parenthesis are computed or processed first.

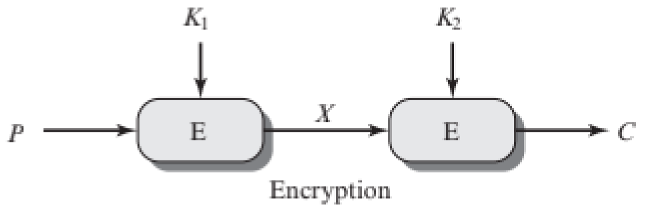
The use of two independent keys, $K_1$and K_2, are 112 bits long. Twice as long as a single key and can provide 112 bits of entropy for the ciphertext. So, the computational effort of a brute force attacker, who only analyzes the Double-DES input and the output P and C, will grow by O(2^{112}). However, the attacker can significantly reduce its complexity by conducting meet-in-the-middle attack.

$$C = Enc(K_2, Enc(K_1, P))$$

$$X = Enc(K_1, P) = Dec(K_2, C)$$

Such meet-in-the-middle attack can apply to any block encryption ciphers which are sequentially processed.

Instead of focusing only on the input and the output of the entire chain of cipher components, the meet-in-the-middle attack also stores and computes the transitional value between the cipher components. The transitional value is shown as X.



In Double-DES, the plaintext goes through the first DES encryption function with a key of K_1. And then, the output of that DES encryption gets input to another DES encryption using the key K_2. The meet-in-the-middle attacker knows that there is an intermediate value, in this case, X, which is in between the two DES functions.

Given a known plaintext or a pair of P and C that is known to the attacker, the attacker first takes the known plaintext P and computes the first DES function with the of K_1. The attacker vaires the key K_1which does not know, and stores all of the two to the 56 possible pair of values $K_2$and X. The attacker then takes the ciphertext C and computes in the backward direction to compute X that is, it will compute the decryption of C to compute X. Whenever the attacker computes the DES decryption from C, it compares the result with the X values that is computed and stored from using P and the encryption computations in the forward direction. When there is a match for X, it has a candidate key $K_1$and K_2which it can then use to try another known plaintext ciphertext pair of P and C. There can be false alarms, but the false alarm rate grows exponentially with the number of known plaintext-ciphertext pairs, limiting the number of known plaintexts that is required for the attacker.

If it were the correct key pair of K_1- K_2, then, it will always yield the correct P-C, regardless of the plaintext P that the attacker tries. This reduces the attacker effort to O(2^{56}) because now, the attacker can compute DES separately. More specifically, the attacker will need to compute all the 2^{56}computations for the first DES cipher from P. And then, half of the 2^{56} or 2^{55}computations for the second DES cipher from C on average. That is on average, the computational effort will be 2^{56}+ 2^{55}, which is O(2^{56}). This complexity of O(2^{56}) is significantly less than O(2^{112}). Taking the ratio between the two, the difference is more than 10^{21}. On the other hand, comparing it with DES, it only increases the complexity by one exponent with base two.

Therefore, Double-DES is just a naive way of using multiple DES ciphers with different keys not secured enough because the meet-in-the-middle attack exploits the vulnerability of double encryption approaches which effectively lowers the attack complexity to find the key. Meet-in-the-middle attack can be used against any double encryption approach, regardless of the cipher algorithm that was applied twice.

HACETTEPE UNIVERSITY

BBM456

Computer and Network Security

Homework

#2

Fatma Çiğdem Tosun

216-----

Q: What is the primitive root? Show an example.

We can say that the highest possible exponent to which a number can belong (mod n) is $\boldsymbol{\Phi}(n)$. If a number is of this order, it is referred to as a **primitive root** of n.
The importance of this notion is that if $a$is a primitive root of n, then its powers

$$a,\ a^{2},\ \dots\ ,\ a^{\boldsymbol{\Phi}(n)}$$

are distinct (mod n) and are all relatively prime to n. In particular, for a prime number p, if a is a primitive root of p, then

$$a,\ a^{2},\ \dots\ ,\ a^{n-1}$$

are distinct (mod p).
Not all all integers have primitive roots. In fact, the only integers with primitive roots are those of form 2, 4, p^{a}, and $2p^{a}$, where p is any odd prime and a is a positive integer.

Let's find a primitive root modulo 23 and modulo 23^{3}.
To find a primitive root mod 23, we use trial and error. Since $\boldsymbol{\Phi}$(23) = 22, for a to be a primitive root, we just need to check that $a^{2} \not\equiv 1$(mod 23) and $a^{11} \not\equiv 1$(mod(23).

$$2^{11} \equiv 2^{5}\ .\ 2^{5}\ .\ 2 \equiv 9\ .\ 9\ .\ 2 \equiv\ -\ 11\ .\ 2 \equiv 1 \text{(mod 23)}$$

so 2 does not work.

$$3^{11} \equiv 3^{3}\ .\ 3^{3}\ .\ 3^{3}\ .\ 9 \equiv 4^{3}\ .\ 9 \equiv\ -\ 5\ .\ 9 \equiv 1 \text{ (mod 23)}$$

so 3 does not work either.

$$5^{11} \equiv (5^2)^5 .\ 5 \equiv 2^5 .\ 5 \equiv 9 .\ 5 \equiv\ -\ 1 \text{ (mod 23)}$$

and $5^2 \equiv 2$ (mod 23), so 5 is a primitive root mod 23.

Now by the proof of existence of primitive root mod p^2, using Hensel's lemma, only one lift of 5 will fail to be a primitive root mod 23^2. We need to check whether $5^{22} \equiv 1\ (mod\ 23^2)$:

$$\begin{aligned} 5^{22}\ &=\ (5^5)^4 .\ 5^2 \equiv (3125)^4 .\ 25 \\ &\equiv\ (-\ 49)^4 .\ 25 \equiv (2401)^2 .\ 25 \\ &\equiv\ 288 .\ 25 \equiv 323\ (mod\ 529) \end{aligned}$$

So 5 is a primitive root mod 529.

HACETTEPE UNIVERSITY

BBM456

Computer and Network Security

Homework

#3

Fatma Çiğdem Tosun

216-----

Q: Describe Block Ciphers vs. Stream Ciphers comparing.

Symmetric cryptography is split into block ciphers and stream ciphers, which are easy to distinguish. Figure below depicts the operational differences between stream and block ciphers when we want to encrypt *b* bits at a time, where *b* is the width of the block cipher.

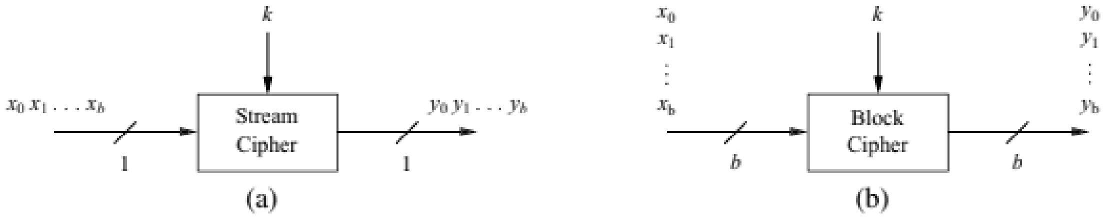


Fig. 2.2 Principles of encrypting *b* bits with a stream (a) and a block (b) cipher

A description of the principles of the two types of symmetric ciphers follows.

Stream ciphers encrypt bits individually. this is achieved by adding a bit from a key stream to a plaintext bit. There are synchronous stream ciphers where the key stream depends only on the key, and asynchronous ones where the key stream also depends on the ciphertext. If the dotted line on the figure below is present, the stream cipher is an asynchronous one. Most practical stream ciphers are synchronous ones. An example of asynchronous stream cipher is the cipher feedback (CFB) mode.

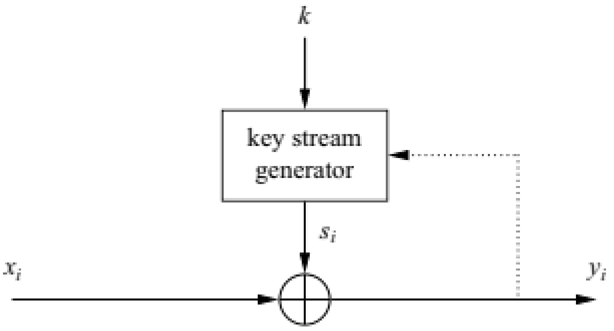


Fig. 2.3 Synchronous and asynchronous stream ciphers

Block ciphers encrypt an entire block of plaintext bits at a time with the same key. This means that the encryption of any plaintext bit in a given block depends on every other plaintext bit in the same block. In practice, the vast majority of block ciphers either have a block length of 128 bits (16 bytes) such as the Advanced Encryption Standard (AES), or a block length of 64 bits (8 bytes) such as the Data Encryption Standard (DES) or triple DES (3DES) algorithm.

In practice, in particular for encrypting computer communication on the Internet, block ciphers are used more often than stream ciphers.
Because stream ciphers tend to be small and fast, they are particularly relevant for applications with little computational resources, e.g., for cell phones or other small embedded devices. A prominent example for a stream cipher is the A5/1 cipher, which is part of the GSM mobile phone standard and is used for voice encryption. However, stream ciphers are sometimes also used for encrypting Internet traffic, especially the stream cipher RC4.
Traditionally, it was assumed that stream ciphers tended to encrypt more efficiently than block ciphers. Efficient for software-optimized stream ciphers means that they need fewer processor instructions (or processor cycles) to encrypt one bit of plaintext. For hardware-optimized stream ciphers, efficient means they need fewer gates (or smaller chip area) than a block cipher for encrypting at the same data rate. However, modern block ciphers such as AES are also very efficient in software. Moreover, for hardware, there are also highly efficient block ciphers, such as PRESENT, which are efficient as very compact stream ciphers.

Stream Cipher	**Block Cipher**
Operates on smaller units of plaintext.	Operates on larger block of data.

Faster.	Slower.
Processes the input element continuously producing output one element at a time.	Processes the input one block of element at a time, producing an output block for each input block.
Requires less code.	Requires more code.
Key is used only once.	Reuse of key is possible.
Application: SSL.	Application: Database, file encryption.
More suitable for hardware implementation.	Easier to implement im software.

HACETTEPE UNIVERSITY

BBM456

Computer and Network Security

Homework

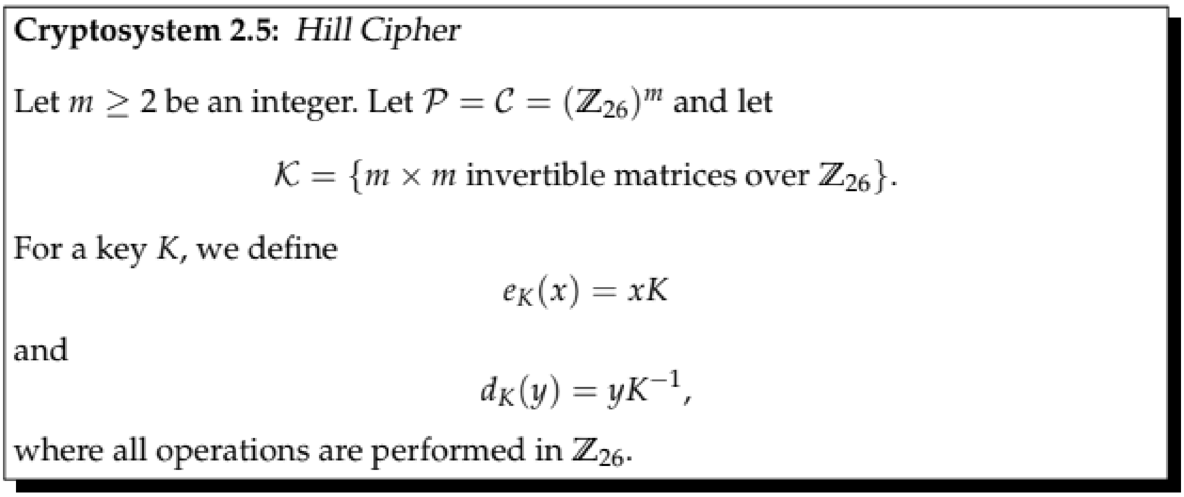
#4

Fatma Çiğdem Tosun

216-----

Q: What is Hill Cipher? How does it work? Give an example.

Hill Cipher is a polyalphabetic cryptosystem which was invented in 1929 by Lester S. Hill.



Cryptosystem 2.5: *Hill Cipher*

Let $m \geq 2$ be an integer. Let $\mathcal{P} = \mathcal{C} = (\mathbb{Z}_{26})^m$ and let

$$\mathcal{K} = \{m \times m \text{ invertible matrices over } \mathbb{Z}_{26}\}.$$

For a key K, we define

$$e_K(x) = xK$$

and

$$d_K(y) = yK^{-1},$$

where all operations are performed in $\mathbb{Z}_{26}$.

Let m be a positive integer, and define $P = C = (\mathbb{Z}_{26})^m$. The idea is to take m linear combinations of the m alphabetic characters in one plaintext element, thus producing the m alphabetic characters in one ciphertext element.

For example, if m = 2, we could write a plaintext element as $x = (x_1, x_2)$ and a ciphertext element as $y = (y_1, y_2)$. Here, $y_1$would be a linear combination of $x_1$and x_2, as would y_2. We might take

$$y_1 = (11x_1 + 3x_2)\ mod\ 26$$

$$y_2 = (8x_1 + 7x_2)\ mod\ 26$$

Of course, this can be written more succinctly in matrix notation as follows:

$$(y_1, y_2) = (x_1, x_2)\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$$

where all operations are performed in $\mathbb{Z}_{26}$. In general, we will take an $m\ x\ m$ matrix K as our key. If the entry in row i and column j of K is $k_{i,j}$, then we write $K = (k_{i,j})$. For $x = (x_1, \dots , x_m) \in P$ and $K \in K$,we compute $y = e_K(x) = (y_1, \dots , y_m)$ as follows:

$$(y_1, y_2, \dots, y_m) = (x_1, x_2, \dots, x_m)\begin{pmatrix} k_{1,1} & k_{1,2} & \dots & k_{1,m} \\ k_{2,1} & k_{2,2} & \dots & k_{2,m} \\ \vdots & \vdots & & \vdots \\ k_{m,1} & k_{m,2} & \dots & k_{m,m} \end{pmatrix}$$

In other words, using matrix notation, $y = xK$.
We say that the ciphertext is obtained from the plaintext by means of a linear transformation. We have to consider how decryption will work, that is, how x can be computed from y. We will use the inverse matrix $K^{-1}$to decrypt. The ciphertext is decrypted using the matrix equation $x = yK^{-1}$.

Suppose the key is

$$K = \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$$

Calculate the inverse of the key,

$$K^{-1} = \begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix}$$

Suppose we want to encrypt the plaintext $july$. We have two elements of plaintext to encrypt: (9, 20) (corresponding to ju) and (11, 24) (corresponding to ly). We compute as follows:

$$(9,20)\left(\begin{array}{cc} 11 & 8 \\ 3 & 7 \end{array}\right) = (99+60, 72+140) = (3,4)$$

and

$$(11,24)\left(\begin{array}{cc} 11 & 8 \\ 3 & 7 \end{array}\right) = (121+72, 88+168) = (11,22).$$

Hence, the encryption of $july$ is $DELW$. To decrypt, Bob would compute:

$$(3,4)\left(\begin{array}{cc} 7 & 18 \\ 23 & 11 \end{array}\right) = (9,20)$$

and

$$(11,22)\left(\begin{array}{cc} 7 & 18 \\ 23 & 11 \end{array}\right) = (11,24).$$

Hence, the correct plaintext is obtained.

At this point, we have shown that decryption is possible if K has an inverse. In fact, for decryption to be possible, it is necessary that K has an inverse.

HACETTEPE UNIVERSITY

BBM456

Computer and Network Security

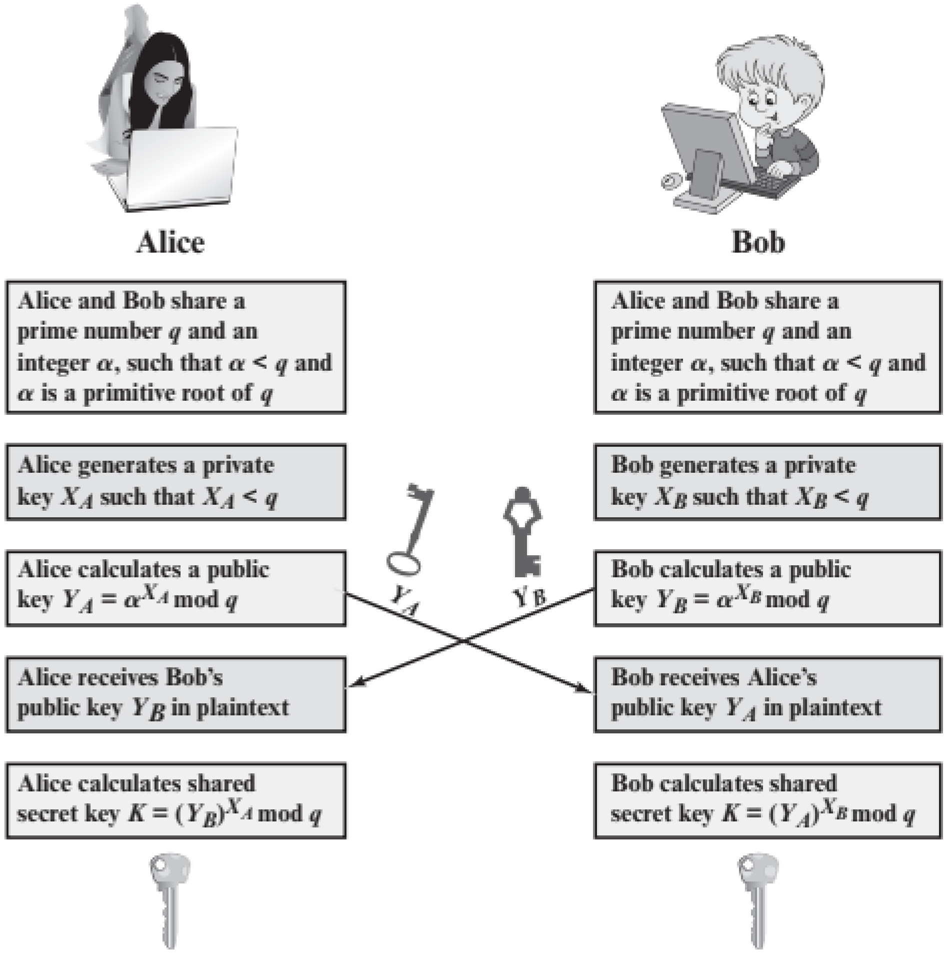
Homework

#5

Fatma Çiğdem Tosun

216-----

Q: Man-in-the-middle attack on Diffie-Hellman. Explain.



Alice and Bob share a prime number q and an integer α, such that $\alpha < q$ and α is a primitive root of q

Alice generates a private key X_A such that $X_A < q$

Alice calculates a public key $Y_A = \alpha^{X_A} \bmod q$

Alice receives Bob's public key Y_B in plaintext

Alice calculates shared secret key $K = (Y_B)^{X_A} \bmod q$

Alice and Bob share a prime number q and an integer α, such that $\alpha < q$ and α is a primitive root of q

Bob generates a private key X_B such that $X_B < q$

Bob calculates a public key $Y_B = \alpha^{X_B} \bmod q$

Bob receives Alice's public key Y_A in plaintext

Bob calculates shared secret key $K = (Y_A)^{X_B} \bmod q$

Figure 10.1 The Diffie–Hellman Key Exchange

The protocol depicted in the figure is insecure against man-in-the-middle attack. Suppose Alice and Bob wish to exchange keys, and Darth is the adversary. The attack proceeds as follows:

1. Darth prepares for the attack by generating two random private keys $X_{D1}$and $X_{D2}$and then computing the corresponding public keys $Y_{D1}$and Y_{D2}.
2. Alice transmits Y_A to Bob.
3. Darth intercepts Y_A and transmits $Y_{D1}$to Bob. Darth alsp calculates

   $K2 \ = \ (Y_A)^{x_{D2}} \, mod \, q.$
4. Bob receives Y_{D1} and calculates $K1 \ = \ (Y_{D1})^{X_B} \, mod \, q$.
5. Bob transmits Y_B to Alice.
6. Darth intercepts Y_B and transmits $Y_{D2}$to Alice. Darth calculates $K1 \ = \ (Y_B)^{X_{D1}} \, mod \, q$.

7. Alice receives $Y_{D2}$and calculates $K2 = (Y_{D2})^{X_A} mod\ q$.

At this point, Bob and Alice think that they share a secret key, but instead Bob and Darth share secret key $K1$and Alice and Darth share secret key $K2$.All future communication between Bob and Alice is compromised in the following way.

1. Alice sends an encrypted message M: $E(K2,\ M)$.
2. Darth intercepts the encrypted message and decrypts it to recover M.
3. Darth sends Bob $E(K1,\ M)$ or $E(K1,\ M')$, where M' is any message. In the first case, Darth simply wants to eavesdrop on the communication without altering it. In the second case, Darth wants to modify the message going to Bob.

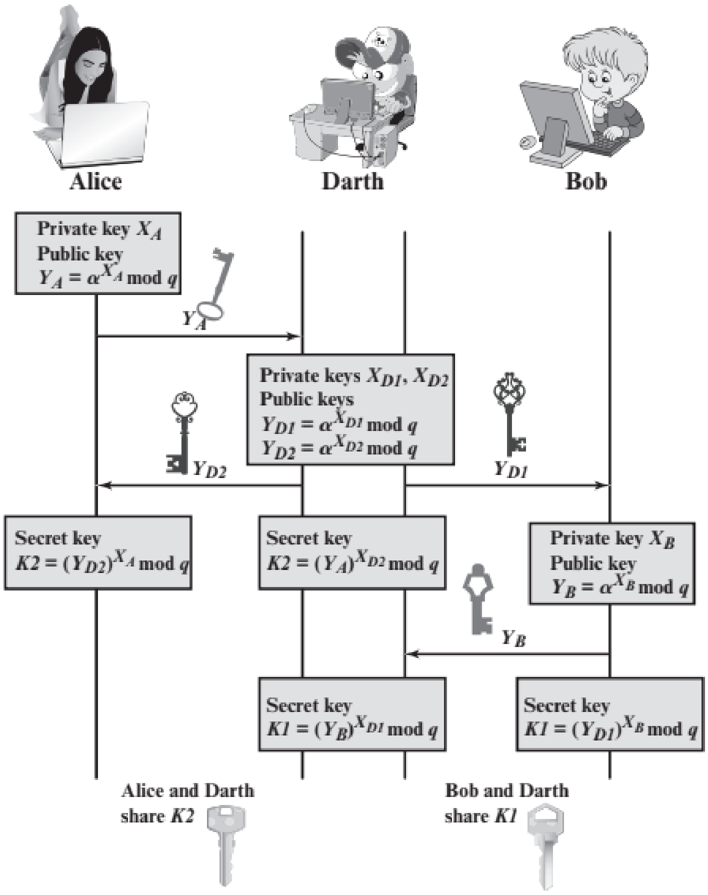

Alice
Darth
Bob
Private key X_A
Public key
$Y_A = \alpha^{X_A}$ mod q
Y_A
Private keys X_{D1}, X_{D2}
Public keys
$Y_{D1} = \alpha^{X_{D1}}$ mod q
$Y_{D2} = \alpha^{X_{D2}}$ mod q
Y_{D2}
Y_{D1}
Secret key
$K2 = (Y_{D2})^{X_A}$ mod q
Secret key
$K2 = (Y_A)^{X_{D2}}$ mod q
Private key X_B
Public key
$Y_B = \alpha^{X_B}$ mod q
Y_B
Secret key
$K1 = (Y_B)^{X_{D1}}$ mod q
Secret key
$K1 = (Y_{D1})^{X_B}$ mod q
Alice and Darth share $K2$
Bob and Darth share $K1$

Figure 10.2 Man-in-the-Middle Attack

The key exchange protocol is vulnerable to such an attack because it does not authenticate the participants. This vulnerability can be overcome with the use of digital signatures and public-key certificates.

HACETTEPE UNIVERSITY

BBM456

Computer and Network Security

Homework

#6

Fatma Çiğdem Tosun

216-----

Q: Download and install PGP. Create encrypted and signed message traffic. Prepare 3-4 pages report with screenshots.

Çözümü sildim çünkü 2 mail adresimin de görünmesini istemiyorum lol