EP2500 Networked Systems Security

Homework 1

*Emonet Camille*

*Fleitz Pierre*

*12/01/2015*

# Basic Cryptographic Primitives and Protocols

## Exercice 1 : Key establishment =

a) 1. We know that Alice and Bob use symmetric key cryptography, therefore they only need one key (for encryption and decryption).

2. And both of them need to know the key in order to communicate.

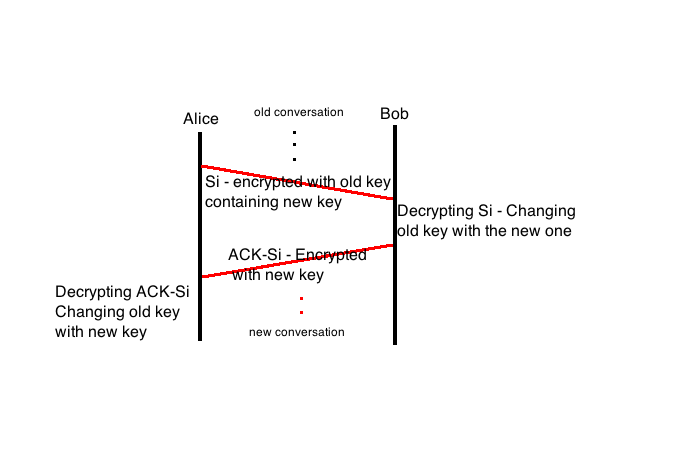
b) 1. This time Alice and Bob are using asymmetric key cryptography, therefore they gonna need 2 keys. One public key for encryption, and one private key for decryption.  
 2. Alice and Bob need to know the public key. Then Alice needs to know her private key only, just like Bob needs to know his private key only.  
  
 3. This is a solution we thought about :  
- Alice send a packet to Bob encrypted using the old share key and containing the new key.

- Bob receives it and can decrypt it. The packet contains a new shared key.

- Bob changes his public key by the new one.

- Bob sends an ACK-Si to Alice with the new key that she sent him earlier.

- Alice receives the ACK-Si and she tries to decrypt is using the new shared key. If she can it means that Bob is using the new shared key, if she can’t it means that the process failed.



## Exercice 2 : XOR encryption =

We know that we have  C1= M1 **⊕** K1 and C2 = M1 **⊕** K2 but also that K1 == K2.   
We want to give the ASCII and the binary representations of M1, M2, and K1.   
  
**M1** : Eve assume that M1 ASCII’s representation is the word : net.

In binary it gives us : 011011100110010101110100

Now we want to have the binary and ASCII of M2 and K2.   
If Eve want to have M2 she gonna have to proceed that way :

C1 = M1 **⊕** K1

C1 **⊕**M1 = M1 **⊕** K1 **⊕**M1

C1 **⊕**M1 = K1  
  
So Eve is gonna have to do the calcul C1 **⊕**M1 in order to have K1 representations :

|  |  |
| --- | --- |
| C1 | 001101000100110001101111 |
| M1 | 011011100110010101110100 |
| C1 **⊕**M1 = K1 | 010110100010100100011011 |
| ASCII | Z ) |

Now Eve do have M1, K1. She wants M2 also. Here’s what she gonna have to do :

C2 = M2 **⊕** K2

C2 = M2 **⊕** K1 (because we know that K1 == K2)

C2 **⊕** K1 = M2 **⊕** K1 **⊕** K1

C2 **⊕** K1 = M2

So this time Eve is gonna have to do the calcul C2 **⊕** K1 in order to have M2 representations :

|  |  |
| --- | --- |
| C2 | 001010010100110001111000 |
| K1 | 010110100010100100011011 |
| C1 **⊕**K1 = M2 | 011100110110010101100011 |
| ASCII | sec |

Proceeding that way will allow Eve to obtain M2 and K1.

## Exercice 3 : Symetric Key =

First attack : Man-in-the-middle Attack.  
Camille (the attacker) is listening the entire conversation. Doing that she receives RA from Alice and Ek(RA) from Bob. Then she can try to find K from the two information. If she finds K then she can impersonate Bob, or Alice in a future conversation by sending [RC(a random number), Alice] to Bob then EK(RC) later authenticating herself as Alice to Bob.

Second attack :

## Exercice 4 : Block Cipher =

a) If you want to obtain the original message from c you gonna have to : Decrypt c, there you will obtain R||m. Therefore if you want m, you gonna have to only keep the last 64 bits of R||m. There you have m, the original message.  
  
 b) What you want if you are the attacker: Send enough pairs so the oracles gives back 2 identicals block ciphers. Indeed, if you send (A1,A2) [knowing that A1 = A2] to the oracle and the oracle send back (C1,C2) (because you reach the probability to have the exact same 64 random bits) then you don’t care to know it it’s C1 or C2 that is encrypted because they are the same. You have the original message and the encrypted message, so you can try to break the algorithm. So yes the attacker can indeed infer something about the algorithm if he sends enough pairs to reach that point.

c) When we want to know if the scheme is secure, we want to know if is realisticly possible to reach the probability that both 64 bits of randoms parts are the same.  
The probability that 1 bits from each are the same is (2^2 / 2\*64). And the probability that 64 bits from both randoms messages are exactly the same is : (2^2 / 2\*64)^64 wich is approximately 5^47. It means that if the attacker wants to be sure that he will have 2 exacts same randoms 64 bits he gonna have to try 5^47 pairs. Wich is an insane number of pairs to send.   
Therefore, even if it is indeed possible to break the scheme, we can’t say that it is not secure !

# Physical Layer Security

## Exercice 5 : Jamming =

# Denial of Service (DoS)

## Exercice 6 : Flooding =

# Unauthorized Access

## Exercice 7 : Password Entropy =

## Exercice 8 : Password Storage =

# Secure Routing

## Exercice 9 : Secure Routing Theory =

## Exercice 10 : Routing Information Protocol (RIP) =

## Exercice 11 : Border Gateway Protocol (BGP) =