Joel Martinez **00126886**

Laboratorio 5 evaluado de seguridad informática

Consideraciones importantes

- El laboratorio se puede hacer con un máximo de 2 personas
- Cualquier sospecha de copia será evaluada con la nota mínima 1.0 para todos los involucrados.
- Tiene una semana para entregarlo

Usted tiene permitido:

- Trabajar en conjunto a un compañero
- Utilizar el lenguaje que desee

Desafío: Se le solicita a usted crear un programa del tipo cliente / servidor, el cual tenga como objetivo que el cliente pueda sincronizar con el servidor una llave mediante diffie-helman para posteriormente recibir un el mensaje leído desde el archivo mensajeentrada.txt el cual debe ser encriptado por DES y poder desencriptar en nuevo archivo llamado mensajerecibido.txt

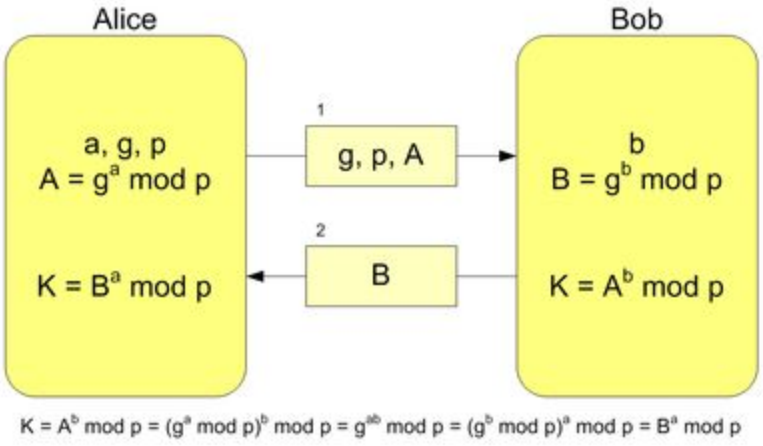
Al terminar la actividad repita lo mismo ocupando 3DES y aes con la cantidad de bytes que usted desee.

Entregable:

- Publicar su codigo en github
- Crear un video de no más de 5 minutos explicando su solución.

Desarrollo:

Se elaboró un programa cliente servidor para la sincronización de una llave.



A través de dos clases llamadas “alice” y “Bob”:

```
class alice:
    def __init__(self,g,p,a):
        self.g=g
        self.p=p
        self.a=a

    def A(self):
        A=(self.g**self.a)%self.p
        return A

    def k(self,B):
        k=(B**self.a)%self.p
        return k



a=70
g=50
p=73

#B=55
uno=alice(g,p,a)
A=uno.A()
#k=uno.k(B)
print('A='+str(A))
#print('k='+str(k))

bhu=str(g)+','+str(p)+','+str(A)+',(g;p;A)'
print(bhu)

dos=receptor()
dos.respuesta_A(bhu)

dat=dos.contacto_B()[2:-1]
B=int(dat)
print('_____')
print("B'="+str(B))

k=uno.k(B)
print('k='+str(k))
```

```
class bob:
    def __init__(self,b):
        self.b=b
        self.g=None
        self.p=None
        self.A=None

    def datos(self,s):
        self.g=s[0]
        self.p=s[1]
        self.A=s[2]

    def B(self):
        B=(self.g**self.b)%self.p
        return(B)

    def k(self):
        k=(self.A**self.b)%self.p
        return k




b=20
s=[int(dat[0]),int(dat[1]),int(dat[2])]

dos=bob(b)
dos.datos(s)
B=dos.B()
k=dos.k()
print('B='+str(B))
print('k='+str(k))
```

Estas clases sincronizan una llave a través de diffie-hellman.
(https://rico-schmidt.name/pymotw-3/socket/tcp.html)

Además de otras dos clases que realizan la comunicación entre ambos interlocutores “emisor” y “receptor” para “bob” y “alice” respectivamente.

“emisor” con las funciones contacto_A y respuesta_B que realizan la comunicación

```
import socket
import sys
import pyDes

class emisor:
    def contacto_A(self):
    def respuesta_B(self,tex):
        # Create a TCP/IP socket
        sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

        # Connect the socket to the port where the server is listening
        server_address = ('localhost', 1000)
        print('>>> connecting to {} port {}'.format(*server_address))
        sock.connect(server_address)

        try:
            # Send data
            a=tex
            message = bytes(a, "ascii")
            print('>>> sending {!r}'.format(message))
            sock.sendall(message)

            # Look for the response
            amount_received = 0
            amount_expected = len(message)

            while amount_received < amount_expected:
                data = sock.recv(16)
                amount_received += len(data)
                print('>>> received {!r}'.format(data))

        finally:
            print('>>> closing socket')
            sock.close()

class bob:
```

“receptor” con las funciones contacto_B y respuesta_A que realizan la comunicación.

```
1  import socket
2  import sys
3  import pyDes
4
5  class receptor:
6      def contacto_B(self):
42
43     def respuesta_A(self,tex):
44         # Create a TCP/IP socket
45         sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
46
47         # Connect the socket to the port where the server is listening
48         server_address = ('localhost', 10000)
49         print('>>> connecting to {} port {}'.format(*server_address))
50         sock.connect(server_address)
51
52         try:
53             # Send data
54             a=tex
55             message=bytes(a, "ascii")
56             #message = b'sd'
57             print('>>> sending {!r}'.format(message))
58             sock.sendall(message)
59
60             # Look for the response
61             amount_received = 0
62             amount_expected = len(message)
63
64             while amount_received < amount_expected:
65                 data = sock.recv(16)
66                 amount_received += len(data)
67                 print('>>> received {!r}'.format(data))
68
69         finally:
70             print('>>> closing socket')
71             sock.close()
72
73 class alice:
```

Estas realizan la sincronización de la llave entre los dos interlocutores,
“receptor empieza iniciado la comunicación con bob y después “emisor” define ‘a’ , ‘g’ y ‘p’ , para generar el ‘A’ que envía posteriormente a “receptor ”, luego “emisor inicia una comunicación con “Alice” para que después que reciba ‘a’ , ‘g’ y ‘p’ , genere un ‘B’ que envía a de vuelta a “emisor”.

Ejecución de los códigos:

- **comunicador_a.py**

```
Python 3.9.0 (tags/v3.9.0:9cf6752, Oct  5 2020, 15:34:40) [MSC v.1927 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\jmemr\OneDrive\Escritorio\lab_4_joel\comunicador_a.py ===
*** starting up on localhost port 10000
*** waiting for a connection
*** connection from ('127.0.0.1', 5019)
*** received b'50,73,69,(g;p;A)'
*** sending data back to the client
B=55
k=16
>>> connecting to localhost port 1000
>>> sending b'55'
>>> received b'55'
>>> closing socket
_____________________________________________________

texto original>>> Flor azul, espina roja, flor azul, espina roja. Esto seria mas
 facil si no fuera daltonico

Encriptacion DES

Cifrado: b'\\\x05c\xc5\x84\xbb\x941T\x85E\x12@\xc6\x17\x1a\xc97nY\xb4^D:M\xed_\x
ba\xae\xc3\x111\xfeO\xd2\xf4 2\x83\xf8\xb1-\xfb\x9f\x0fs\xe0h\xfa\x80\x1ezb@R\xa
4\xb2\x1e\xf9\x8b}>\x1c\x8fPW\x0e\x0b\x1fu\xd8\x9e\x00\x07u\xf9\x9e+\xdd\x93\x16
\xd89$\x9b\x1ev\xd9\xe7\xea?\xf6\xc4\xc1\xeb\x80'

Encriptacion 3DES

Cifrado: = b"m9\x08{\xc0\xeb\xec\x8c\xa8a;\x96\x92K\x14F\xd1\x90 \xd3\x83\x88\x8
a\x10\x8c\x1e\xa6\x89\x95eO\xe0d\xb8\xa0}\x7fW\xdf'\xb6\x03\xca\x8f\xa3S\xd0W-\x
06\xf8\xd1\x86\xd8\xca\x80\xf7(\x9cj!\xe5X\xc3cm\xf3^\xbc#n\xa1\x16:KK/J\x0b\x05
=\n\xa3\x1c\xf7U\xfcr\xea\xe0\xab\xad\xe1\x93\xb0\x82"

>>>
```

- **comunicador_b.py**

```
Python 3.9.0 (tags/v3.9.0:9cf6752, Oct  5 2020, 15:34:40) [MSC v.1927 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\jmemr\OneDrive\Escritorio\lab_4_joel\comunicador_b.py ===
A=69
50,73,69,(g;p;A)
>>> connecting to localhost port 10000
>>> sending b'50,73,69,(g;p;A)'
>>> received b'50,73,69,(g;p;A)'
>>> closing socket
*** starting up on localhost port 1000
*** waiting for a connection
*** connection from ('127.0.0.1', 5036)
*** received b'55'
*** sending data back to the client
_____________________________________________________
B'=55
k=16

Encriptacion DES

Descifrado: 'Flor azul, espina roja, flor azul, espina roja. Esto seria mas faci
l si no fuera daltonico'

Encriptacion 3DES

b'Flor azul, espina roja, flor azul, espina roja. Esto seria mas facil si no fue
ra daltonico'
>>>
```