ДОДАТОК А

Код модулю Client.py

```
#!/usr/bin/env python
# -*- coding: UTF-8 -*-
from scapy.all import *
from string import*
import struct
conf.L3socket=L3RawSocket
import logging
logger = logging.getLogger(__name__)

class TcpSession_client(object):
    def __init__(self, target):
        fin=open('/home/bell/My/text.txt')
        temp=fin.read(4)
        data="".join([format(ord(k),'b').zfill(8) for k in temp])
        print 'The secret message: ', temp
        print 'ISN_bin= ', data
        print 'ISN_dec= ',int(data,2)
        fin.close()
        self.seq = int(data,2)
        self.target = target
        self.dst = iter(Net(target[0])).next()
        self.dport = target[1]
        self.sport = 1222
        self.l4 = IP(src=target[0],
dst=target[0])/TCP(sport=self.sport, dport=self.dport, flags=0,
seq=self.seq)
        self.src = self.l4.src
        logger.debug("init: %s"%repr(target))

    def start(self):
        logger.debug("start")
        return self.send_syn()

    def handle_recv(self, pkt):
        global f
        if pkt and pkt.haslayer(IP) and pkt.haslayer(TCP):
            if pkt[TCP].flags & 0x3f == 0x12:   #SYN+ACK
                logger.debug("RCV: SYN+ACK")
                return self.send_synack_ack(pkt)
            elif (pkt[TCP].flags ==0x04):       #RST
                logger.debug("RCV: RST")
                raise Exception("RST")
            elif (pkt[TCP].flags & 0x3f == 0x01):#FIN
                logger.debug("RCV: FIN")
                return self.send_finack(pkt)
            elif (pkt[TCP].flags & 0x3f == 0x11):#FIN+ACK
```

```
            logger.debug("RCV: FIN+ACK")
            f=1
                return self.send_finack_ack(pkt)
            elif (pkt[TCP].flags & 0x3f == 0x18):#PSH+ACK
                logger.debug("RCV: PSH+ACK")
                return self.send_ack(pkt)
            elif (pkt[TCP].flags & 0x3f == 0x10 ):    #ACK
            logger.debug("RCV: ACK")
            if (f==1):
                  logger.debug("The session ended")
                  sys.exit()
        return None

    def send_syn(self):
        logger.debug("SND: SYN")
        self.l4[TCP].flags = "S"
        self.seq_next = self.l4[TCP].seq + 1
        response = sr1(self.l4,iface="lo")
        self.l4[TCP].seq += 1
        self.l4[TCP].show()
        return self.handle_recv(response)

    def send_synack_ack(self, pkt):
        logger.debug("SND: ACK")
        self.l4[TCP].ack = pkt[TCP].seq+1
        self.l4[TCP].flags = "A"
        self.seq_next = self.l4[TCP].seq
        send(self.l4,iface="lo")

    def send_fin(self):
        logger.debug("SND: FIN")
        self.l4[TCP].flags = "F"
        self.seq_next = self.l4[TCP].seq + 1
        response = sr1(self.l4, iface="lo")
        self.l4[TCP].seq += 1
        return self.handle_recv(response)

    def send_finack(self, pkt):
        logger.debug("SND: FIN+ACK")
        self.l4[TCP].flags = "FA"
        self.l4[TCP].ack = pkt[TCP].seq+1
        self.seq_next = self.l4[TCP].seq + 1
        response = sr1(self.l4, iface="lo")
        self.l4[TCP].seq += 1
        return self.handle_recv(response)

    def send_ack(self, pkt):
        logger.debug("SND: ACK")
        self.l4[TCP].flags = "A"
        self.l4[TCP].ack = pkt[TCP].seq+len(pkt[Raw])
        self.seq_next = self.l4[TCP].seq + 1
        send(self.l4, iface="lo")
```

```
    def send_finack_ack(self, pkt):
        logger.debug("SND: ACK")
        self.l4[TCP].flags = "A"
        self.l4[TCP].ack = pkt[TCP].seq+1
        self.seq_next = self.l4[TCP].seq + 1
        send(self.l4, iface="lo")
        self.send_finack( pkt)

    def snif_f(self):
        response=sniff(filter="tcp", count=1, iface="lo")
        return self.handle_recv(response[0])

if __name__=='__main__':
    logging.basicConfig(level=logging.DEBUG)
    logger.setLevel(logging.DEBUG)
    conf.verb = 0
    tcp_hs = TcpSession_client(("127.0.0.1",9999))
    tcp_hs.start()
    while(True):
       tcp_hs.snif_f()
```

ДОДАТОК Б

Код модулю Server.py

```
#!/usr/bin/env python
# -*- coding: UTF-8 -*-
from scapy.all import *
import socket
import time
import sys
conf.L3socket=L3RawSocket
import logging
logger = logging.getLogger(__name__)

ip = "127.0.0.1"
port = 9999
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
s.bind((ip, int(port)))
s.listen(1)
conn, addr = s.accept()
print 'Connection ip_adress and port:', addr[0],',', addr[1]
data='Hello World!'
conn.send(data)
conn.send(data)
conn.send(data)
print 'Connection ended'
conn.close()
```

ДОДАТОК В

Код модулю Sniff.py

```
#!/usr/bin/env python
# -*- coding: UTF-8 -*-
from scapy.all import *
from os import system
from struct import pack


while (True):
     pack=sniff(iface="lo", count=1)[0]
     if pack[TCP].flags & 0x3f == 0x02:
          isn_dec=pack[TCP].seq
          print 'ISN_dec= ',isn_dec
          isn_bin=bin(isn_dec)[2:].zfill(32)
          print 'ISN_bin= ',isn_bin
          isn_char=''.join([chr(int(isn_bin[i:i+8],2)) for i in
range(0, len(isn_bin),8)])
          print 'Secret message: ', isn_char
          f=open('/home/bell/My/message.txt', 'w')
          f.write(isn_char)
          f.close
          sys.exit()
```

ВІДОМІСТЬ БАКАЛАВРСЬКОЇ АТЕСТАЦІЙНОЇ РОБОТИ

№	Позначення	Найменування	Дод. відомості
		Текстові документи	
1	ГЮІК.ХХХХХХ.020 ПЗ	Пояснювальна записка	65 с.
		Графічні матеріали	
2		Слайд-презентація	17 слайдів
		Інші документи	
3		Електронна версія ПЗ	1 шт.

					ГЮІК.ХХХХХХ.Х020 ВД	
Змін.	Арк.	Номер докум.	Підп.	Дата		
Розроб.		Костенюк Т.А.			Мережева стеганографія на основі генерації ISN номерів	Літ: У; Арку: 1; Аркуші: 1
Перевір.						
						ХНУРЕ Кафедра БІТ
Н.контр.		Конева Н.Ф.				
Затв.		Халімов Г.З.				