上海理工大学光电信息与计算机工程学院

**《信息安全》实验报告**

****

**专　　业 计算机科学与技术**

**学生姓名　　 许耀永**

**学　 号 1712480131**

**年　　级 2017级**

**指导教师 刘亚**

**成 绩：**

**教师签字：**

报告格式要求

1. 正文字体中文为宋体，五号，行距为固定值18磅，西文为Times New Rome, 五号，行距为固定值18磅。
2. 章节标题为加粗宋体，小四号，段前段后各0.5行，行距为固定值18磅。
3. 打印时需双面打印。

目录

[实验一 古典加密算法的实现与破解 1](#_Toc493076389)

[实验二 AES的实现 2](#_Toc493076390)

[实验三 RSA的实现 3](#_Toc493076391)

[实验四 网络侦听工具及协议分析 4](#_Toc493076392)

# 实验一 古典密码实验

**一、 实验目的**

(1)实现凯撒密码的加解密

其中加密的明文使用小写字母，解密的密文用大写字母

(2)

**二、实验要求**

**三、程序的数据结构设计和算法流程图或算法描述**

**I.Caser密码**

(1)在数学形式上，将字母映射成数字

(2)选择一个整数密钥 K(K 在 1 到 25 之间)

(3)加密时，依据“加 K 模 26” ，将原字母由 L 转换成(L + K) % 26

(4)解密时，依据“减 K 模 26” ，将密文字母由 L 转换成(L - K) % 26

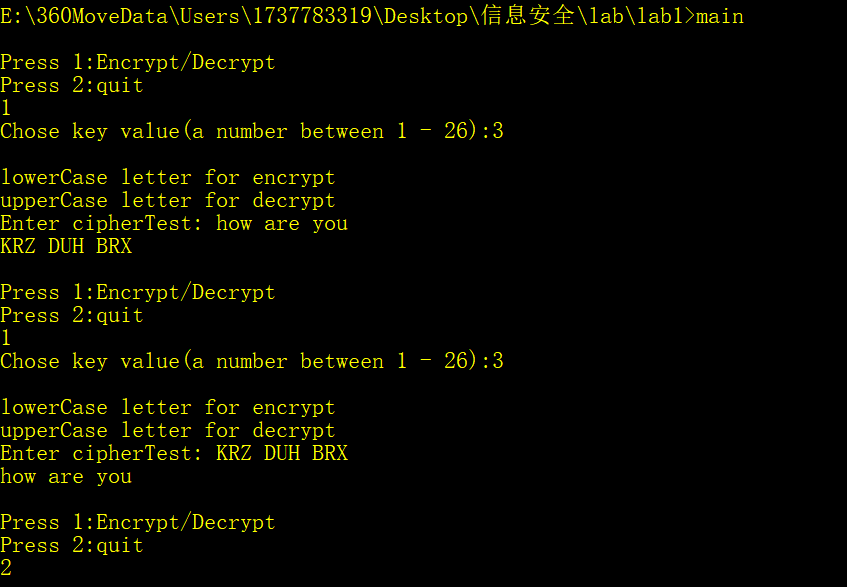
**II. Polyalphabetic密码**

(1)实现维吉尼亚密码的加解密

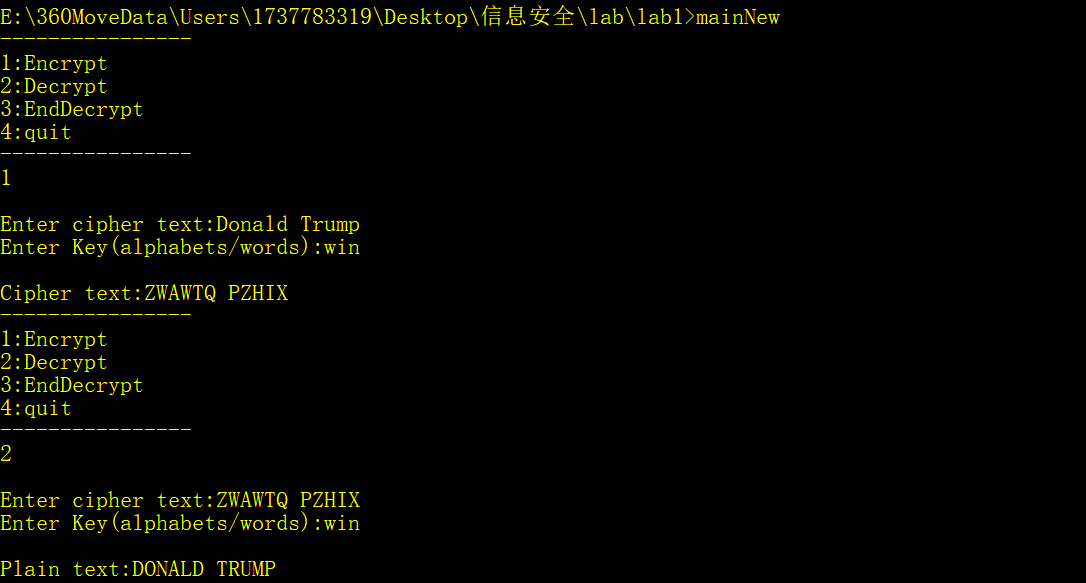
(2)实现维吉尼亚密码的破解

**四、程序运行后的结果截图**

**I.Caser密码**



**II. Polyalphabetic密码**



# 实验二 AES实验

**一、 实验目的**

实现AES的加密(Encryption)和解密(Decryption)

**二、 实验要求**

(1)AES的所有操作都是针对于一个二维数组state

一个state数组由 4行Nb列组成

(2)AES基于下列5个基本操作

1.异或(Exclusive Disjunction/XOR):当两个digit不同时输出为真

2.SubByte/InverseSubByte : 一个byte被另一个byte替代

加密时根据S\_Box查表，解密时根据Si\_Box查表

(每个byte都可以表示为两个十六进制的digit，

其中第一个代表S\_Box的行，第二个代表S\_Box的列)

3.Shift Rows(Rotation)/Inverse Shift Rows:

重排bytes，以不同的偏移量循环左移后三行

Row 0不移动

Row 1 移动一个byte

Row 2 移动两个byte

Row3 移动三个byte

解密时循环右移相对应的byte

4.MixColumns/InverseMixColumns:

二维数组state的每一列上的每一个byte映射为一个新的值

a.将state的每一列左乘一个4 \* Nb的矩阵，所得结果逐个异或而产生一个byte

b. a的乘法操作可以用两个查找表L\_Table和E\_Table来实现

step1 :一个byte，其表示为两个十六进制的digit，

而第一个digit用作L\_Table的行数，第二个digit用作L\_Table的列数

step2 : 将两个从L\_Table获得的值相加，得到一个新的byte

step3 : 在E\_Table查看这个相加的结果(按digit来查)

Note:任何数与1相乘的结果都是它本身

5.AddRoundKey:state数组的每个byte都和round key的每个byte相异或(XOR)

(3) 对于128比特的key,执行10 rounds(轮)

对于192比特的key,执行 12 rounds(轮)

对于 256比特的key,执行 14 rounds(轮)

(4) AES的框架

从第一轮到倒数第二轮，均执行:

a.SubBytes/InverseSubBytes

b.ShiftRows/InverseShiftRows

c.MixColumns/InverseMixColumns

d.AddRoundKey

最后一轮 不执行 MixColumns/InverseMixColumns

(6) Key Expansion

将一个长度为Nk的输入key生成一个长度为Nb \* (Nr + 1)的线性key

Nb : state数组的列数

Nr: 轮的个数

Nk: key的长度

记W[i]为expanded key的第i个word

**三、程序的数据结构设计和算法流程图或算法描述**

(1) Key Expansion

Begin

word temp;

for i <- 0 to (Nk - 1)

w[i] <- ((unsigned char) key [ 4 \* i ] << 24)

| ((unsigned char) key [ 4 \* i + 1 ] << 16)

| ((unsigned char) key [ 4 \* i + 2 ] << 8)

| ((unsigned char) key [ 4 \* i + 3 ] );

end for

for i <- (Nk - 1) to Nb \* Nr

temp = w[i - 1];

if (i mod Nk == 0)

temp = SubWord ( RotWord (temp))

XOR (Rcon [ i / Nk ] << 24) ;

else if ( Nk > 6 and ( i mod Nk ) == 4 )

temp = SubWord(temp);

end if

w[i] = w[i - Nk] XOR temp;

end for

End

(2) Encryption

Begin

state = plaintext

1.KeyExpansion

2.AddRoundkey(state,Expandedkey[0])

3.for r <- 1 to (Nr - 1)

a. SubBytes (state , S-Box)

b. ShiftRows (state)

c. MixColumns(state)

d. AddRoundKey(state,ExpandedKey[r])

4. SubBytes (state , S-Box)

ShiftRows (state)

AddRoundKey(state,ExpandedKey[Nr])

Out = cipherText

End

(3) Decryption

Begin

State = cipherText

1.Key Expansion

2.AddRoundKey(state,ExpandedKey[0])

3.for r <- (Nr - 1) to 1

a. InverseShiftRows(state)

b. InverseSubBytes(state,S-Box)

c. AddRoundKey(State,ExpandedKey[r])

4. InverseSubBytes(state,S-Box)

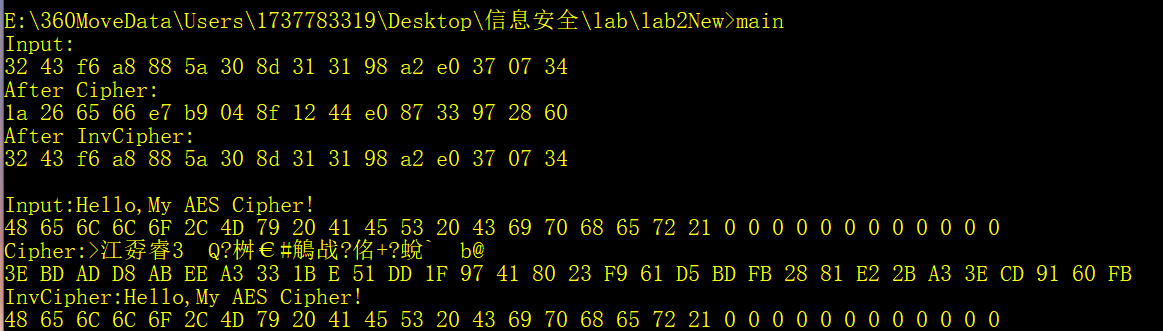
InverseShiftRows(state)

AddRoundKey(state,ExpandedKey[Nr])

Out = plaintext

End

**四、程序运行后的结果截图**

****

# 实验三 RSA实验

**一、 实验目的**

实现RSA的加密(Encryption)和解密(Decryption)

**二、实验要求**

**Operation:**

**1.** Key Generation:

**(1)** Select two large prime numbers p and q randomly, such that p is not equal to q

(2) Compute n such that n = p \* q

(3) Compute f(n) = f(p) \* f(q) = (p - 1) \* (q - 1) where f is Euler’s totient function

(4) Select an number **e** such that 1 < **e <** f(n) and gcd (e, f(n)) == 1

where e and f(n) are co-prime

(5) Compute d as the multiplicative inverse of e(mod(f(n))) i.e. de = 1 mod f(n)

(6) Publish the pair PU = (e, n) as participant’s public key

Keep the pair PR = (d, n) as the participant’s private key

Algorithm 1 : findE(phi\_n)

Begin

e <- 0

do

begin

choose an integer number e

(e must be co-prime of phi\_n)

while(!checkCoprime(phi\_n, e))

end do-while

return e

End

Algorithm2 : FindD(phi\_n,e)

Begin

Local variables: a, b , x , y , u , v , m , n , q , r , gcd

a <- phi\_n

b <- e

x <- 0

y <- 1

u <- 1

v <- 0

gcd <- b

while(a != 0)

begin

q <- gcd / a

r <- gcd % a

m <- x – u \* q

n <- y – v \* q

gcd <- a

a <- r

x <- u

y <- v

u <- m

v <- n

end while

if y < 1

begin

y <- phi\_n + y

end if

return y

End

Algorithm 3 : Generate(&n,&e,&d)

Begin

local variables : p , q , phi\_n

Enter two prime numbers and stored them in p and q respective

n <- multiply(p , q)

phi\_n <- multiply(p – 1 , q - 1)

e <- findE(phi\_n)

d <- findD(phi\_n,e)

(e,n) => public key

(d,n) => private key

End

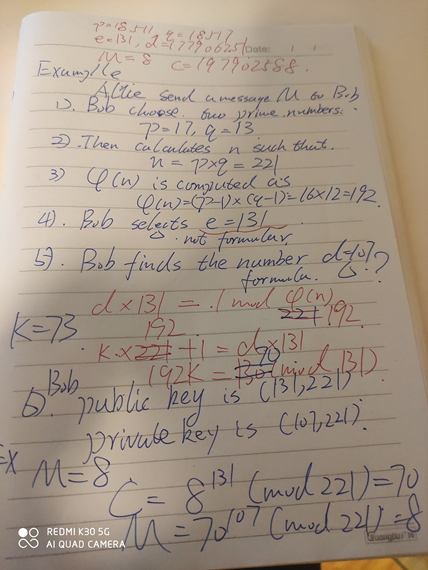
2. Encryption

C = M ^ e ( mod n)

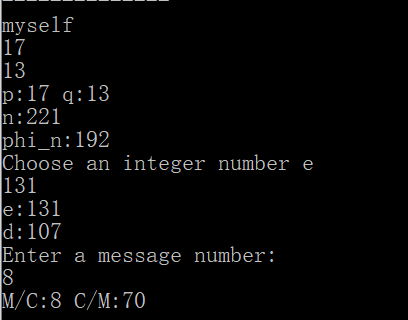
3. Decryption

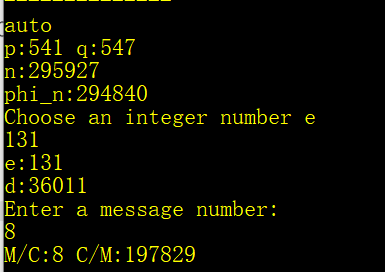
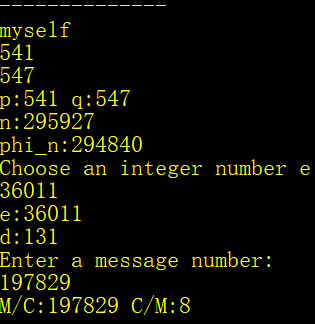
M = C ^ d ( mod n)

Example

****

**三、程序运行后的结果截图**

****



# 实验四 Wireshark实验

**一、实验目的**

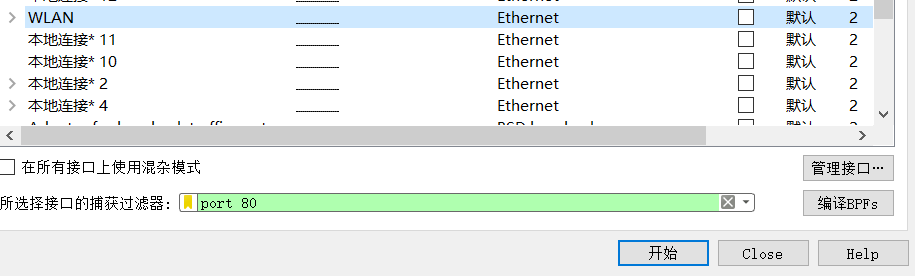
* 了解网络侦听的目的、过程以及基本手段。对协议分析有基本了解。
* 学会使用Wireshark工具。

**二、实验要求**

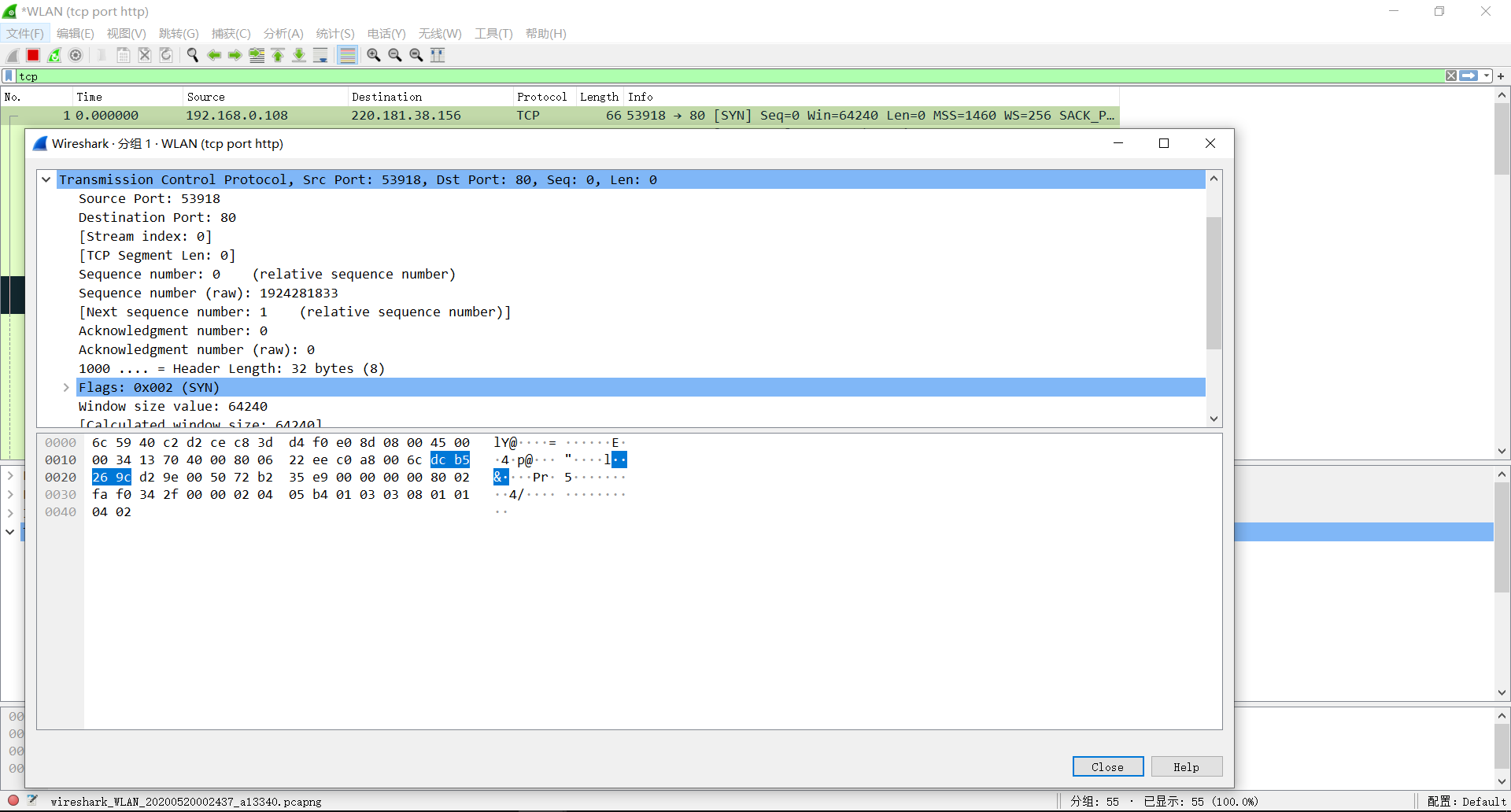
* 1. 正确安装Wireshark软件，并且能够熟练使用Wireshark的主要功能模块；
  + 菜单栏
  + 工具栏
  + 显示过滤器
  + Packet List封包列表
  + Packet Details包详情面板
* 2. 熟悉Wireshark软件的包过滤设置和数据显示功能使用；
  + 捕捉的方法
  + 使用过滤器
* 3. 利用Wireshark抓取TCP协议的数据包，深入分析“TCP三次握手协议”。
  + 确定目标地址
  + 启动抓包
  + 使用FTP服务
  + 通过显示过滤器得到相关数据包
  + 分析数据包

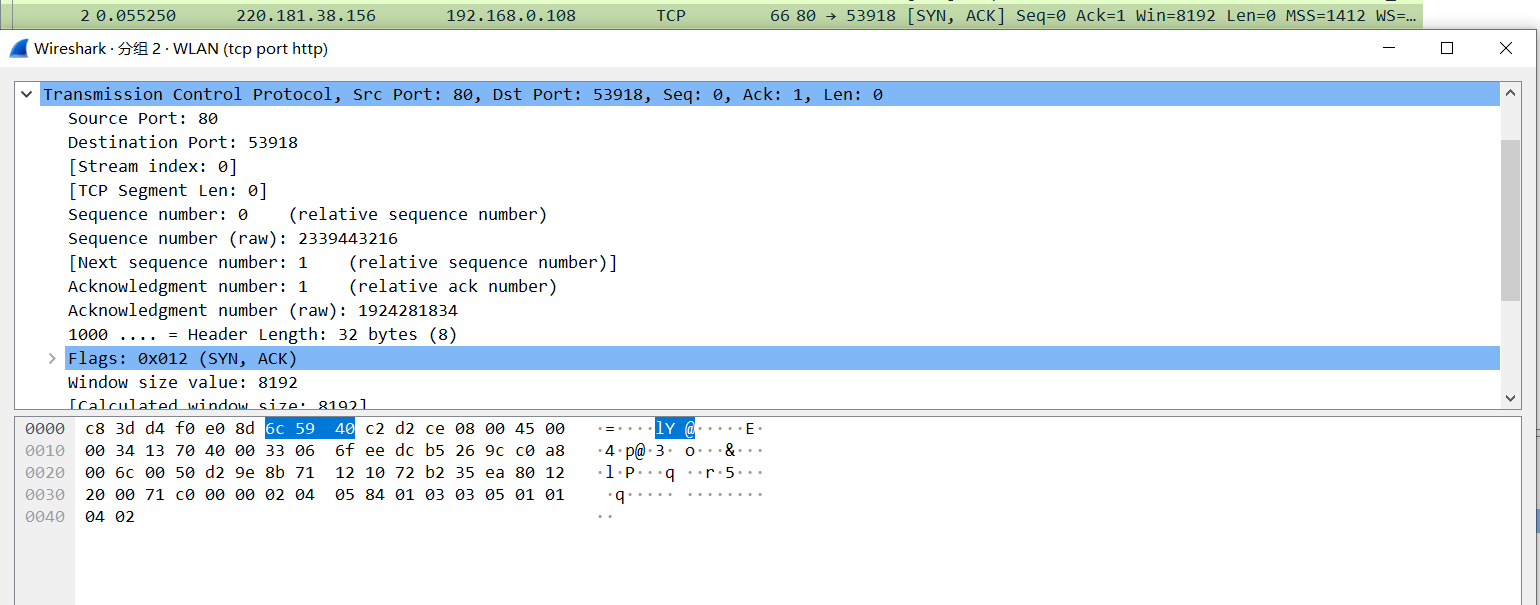
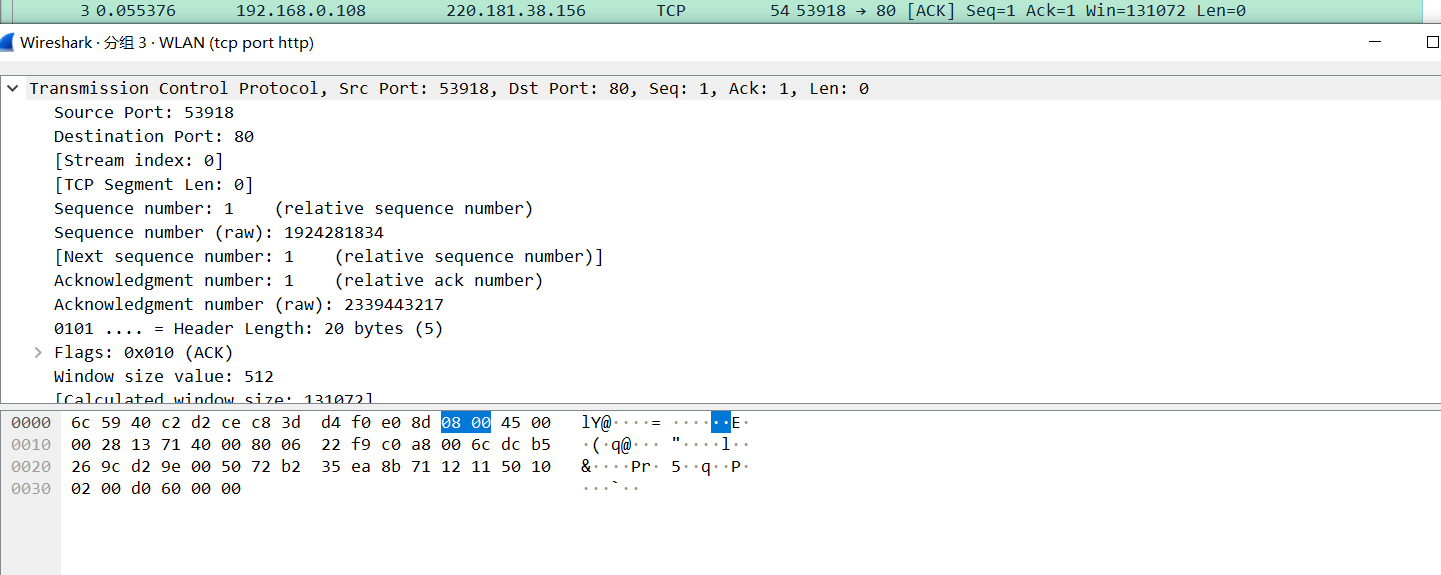
**三、实验内容及结果（写出源程序和运行程序后的结果截图）**

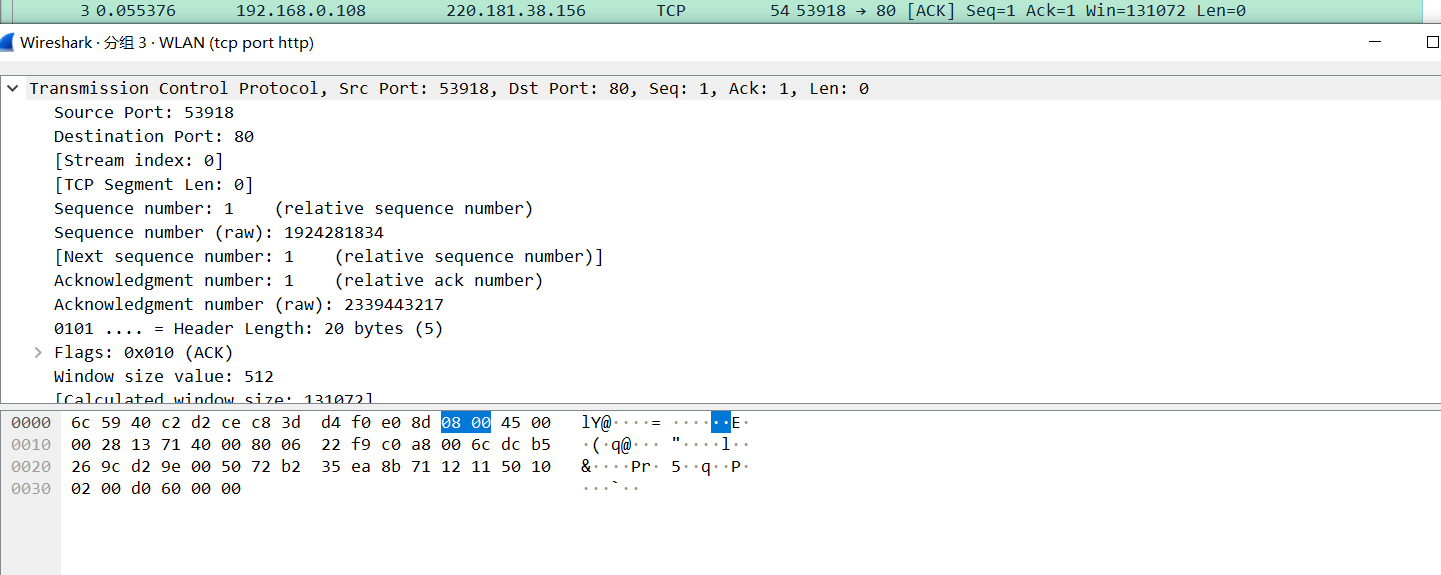
* 1. 实验前Wireshark过滤器设置



* 2. TCP第一次握手



* 3. TCP第二次握手
* 4. TCP第三次握手



* 5. 响应

