计算机网络编程
实验报告

班级：07111707

组长：1120171189 崔程远

成员：1120172149 吴沁璇

1120172153 张澈

1120172163 王晓媛

1120172733 张鉴昊

1120172765 曾煜瑾

1120173326 曾紫飞

北京理工大学

计算机学院

2020 年 5 月

北京理工大学

第三章 实验 5　CHAP 身份验证口令验证算法程序

1. 实验目的

CHAP 身份验证口令验证

2. 实验内容

本次实验内容为查阅 CHAP 标准规范，编写口令验证算法程序.
程序运行屏幕输出关键要点:
通过命令行参数给出要验证的口令
屏幕显示当前要验证的口令
屏幕显示当前生成的随机数
屏幕显示当前生成的 MD5 摘要值

3. 实验原理

PPP 协议的身份验证方法包括 PAP 和 CHAP，PAP 口令验证协议由于直接传输明文用户名和口令，通过网络抓包很容易进行破解。CHAP 协议由于不直接传输口令，而是采用生成随机数（不重字）和口令进行拼接和 MD5 摘要计算，通过 MD5 摘要值进行验证，安全性较高，

4. 实验环境

语言	集成开发环境	编译器
C++	Visual Studio 2017	gcc version 4.8.1
Java	Eclipse 2019	java version "1.8.0_65"
Python	Pycharm 2017	Python 3.7.0

5. 实验步骤

- C

首先检测参数数量，接收预设口令；生成随机数；打印预设口令；接收待检验口令并打印; 打印随机数; 模拟被验证方和验证放分别生成待附加段的口令的 MD5 值；打印校验结果。

ltobarr() 将随机生成的整数变为 byte 数组形式，模拟字节传输

GenRandNum() 生成随机数

CheckMD5()检验两端 MD5

genMd5()拼接口令和附加段，生成相应 MD5 值

- Python

首先检测参数数量，创建 CHAPServer 类对象；接收预设口令；打印预设口令；接收待检验口令并打印；生成随机数并打印随机数；模拟被验证方和验证放分别生成待附加段的口令的 MD5 值；打印校验结果。

- Java

程序主体部分：

```
public static void main(String[] args) throws Exception {
    int len=args.length;
    if(len != 1)
        System.out.println("Wrong Arg Num!");
    else {
        CHAPServer chap =new CHAPServer(args[0]);
        System.out.printf("Key:\t%s\n",args[0]);

        System.out.println("请输入口令: ");
        Scanner sc = new Scanner(System.in);
        String inputkey = sc.nextLine();
        System.out.printf("Input key:\t%s\n",inputkey);

        String rand = chap.genRandNum();
        System.out.printf("Rand Num:\t%s\n",rand);

        String inputs = inputkey + rand;
        MyMD5Util mds = new MyMD5Util();
        mds.md5Hex(inputs);
    }
```

随机数和口令进行拼接：

```
public String genRandNum()
{
    Random random = new Random();
    for(int i=0;i<randNumLen;i++)
    {
        randNum += String.valueOf(random.nextInt( bound: 10));
    }
    return randNum;
}
```

计算 MD5 值:

```java
public static void md5Hex(String inpu) throws Exception {
    //获的MD5消息摘要
    MessageDigest md5 = MessageDigest.getInstance("MD5");
    md5.update(inpu.getBytes( charsetName: "utf-8"));
    byte[] result = md5.digest();



    String hexString = toHexString(result);
    System.out.println("MD5:"+ hexString);
}


public static String toHexString(byte[] bytes) {
    StringBuffer sb = new StringBuffer();
    for (byte b : bytes) {
        String hex = Integer.toHexString( i: b & 0x0FF);
        if (hex.length() == 1) hex = "0" + hex;
        sb.append(hex);
    }
    return sb.toString();
```

- 实验效果:

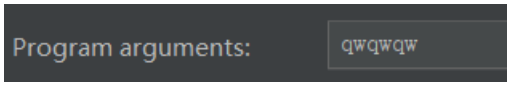
C:

```
[chez@chez-laptop C]$ ./chap 123456
Key:            123456
Please input key:
123456
Input key:      123456
Random Num:     575fc76600000000
MD5             768c9577186d0a1687eea709142f3a47
Check Answer:   True
[chez@chez-laptop C]$ ./chap 123456
Key:            123456
Please input key:
123
Input key:      123
Random Num:     cb642f5f00000000
MD5             8e04a43f9827acf87ac23a977fcdf0b5
Check Answer:   False
```

Python:

```
[chez@chez-laptop python]$ python chap.py 123456
Key:              123456
Please input key:
123456
Input key:        123456
Random Num:       2e0adecb
MD5:              91abb695c4783466c7603f15da3c5be6
Check Answer:     True
[chez@chez-laptop python]$ python chap.py 123456
Key:              123456
Please input key:
123
Input key:        123
Random Num:       453cc8e1
MD5:              45a07678c252d674d4097bbeb1a133bf
Check Answer:     False
```

Java：




```
"E:\IntelliJ\IntelliJ IDEA Community Edition 2
Key:    qwqwqw
请输入口令:
123123123
Input key:  123123123
Rand Num:   7049
MD5:9578a1c76bd744e83de01afba4d8f677

Process finished with exit code 0
```

6. 实验总结

这次实验主要考察了 CHAP 身份认证的方法，相较于明文传输，CHAP 通过 MD5 的身份验证方法更加安全。通过本次实验我更加了解 CHAP 验证中 MD5 的作用和如何确保安全性。

总之，对 CHAP 验证过程和 PPP 协议有了更深的认识。