**Lab 4. Packet Sniffing**

* **Task 1**

1. Your Answer

secret1: flag{e43y\_p4ck37\_sn1ffNg}

(2) Writeup

[Record how you solve this task here.]

根据提示：{"result":0,"message":"Wrong username or password.","hint":"Try to login as GUEST"}

用户名：GUEST

获得提示 ：{"result":0,"message":"Wrong username or password.","hint":"GUEST's password is TEMP\_PASSWD"}

试图直接改变result，由于修改位置位于中间部位，修改会导致前后的字符发生变化，为确保前后字符不变，result只能直接调为0或8，无法解决问题。

然后根据代码写了一个decode函数的解析函数。

解析函数：

#include <stdio.h>

#include <string.h>

#include<iostream>

#include<malloc.h>

using namespace std;

unsigned char\* decode(const char\* secretStr, int\* length) {

const char\* secret = secretStr;

int BabyCipher\_SecretLookup[]={0xFF, 0xFF, 26, 27, 28, 29, 30, 0x1F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF};

int secretLength = strlen(secret);

int bytesLength = secretLength \* 5 / 8;

unsigned char\* bytes = (unsigned char\*)malloc(bytesLength);

int i = 0;

int index = 0;

int offset = 0;

while (i < secretLength) {

int lookup = secret[i] - 0x30;

if (lookup >= 0) {

int\* v6 = BabyCipher\_SecretLookup;

if (lookup >= sizeof(BabyCipher\_SecretLookup) / sizeof(BabyCipher\_SecretLookup[0])) {

break;

}

int digit = v6[lookup];

if (digit == 0xFF) {

break;

}

if (index > 3) {

index = (index + 5) % 8;

bytes[offset] |= (digit >> index);

if(offset==10){

cout<<"n1 digit:"<<digit<<"index:"<<index<<"bytes[10]:"<<bytes[10]<<endl;

}

offset++;

if (offset >= bytesLength) {

\*length = bytesLength;

cout<<"digit:"<<digit<<"index:"<<index<<endl;

return bytes;

}

bytes[offset] |= (digit << (8 - index));

if(offset==10){

cout<<"n2 digit:"<<digit<<"index:"<<index<<"bytes[10]:"<<bytes[10]<<endl;

}

} else {

if(offset==10){

cout<<index<<endl;

}

index = (index + 5) % 8;

if (index == 0) {

bytes[offset] |= digit;

if(offset==10){

cout<<"n3 digit:"<<digit<<"index:"<<index<<"bytes[10]:"<<bytes[10]<<endl;

}

offset++;

if (offset >= bytesLength) {

\*length = bytesLength;

return bytes;

}

} else {

bytes[offset] |= (digit << (8 - index));

if(offset==10){

cout<<"n4 digit:"<<digit<<"index:"<<index<<"bytes[10]:"<<bytes[10]<<endl;

}

}

}

}

i++;

}

\*length = offset;

return bytes;

}

然后根据POST新写了一个POST，OVZWK4TOMFWWKPKHKVCVGVBGOBQXG43XN5ZGIPIUKQCEKBCNKACF6BCQIECFGBCTK4CEI（解析即为{"username":"GUEST","password":"TEMP\_PASSWD"}）。然后发现直接改也可以OVZWK4TOMFWWKPKHKVCVGVBGOBQXG43XN5ZGIPKUIVGVAI2QIFJVGV2E（解析即为{"username":"GUEST","password":"TEMP#PASSWD"}）改为OVZWK4TOMFWWKPKHKVCVGVBGOBQXG43XN5ZGIPKUIVGVAX2QIFJVGV2E（解析即为{"username":"GUEST","password":"TEMP\_PASSWD"}），然后拦截后放行就进去了。

* **Task 2**

(1) Writeup

[Record how you solve this task here.]

我先通过阅读<https://portswigger.net/burp/documentation/>中的案例，然后模仿示例写的程序。

导入所需的模块和类：

IBurpExtender 和 IHttpListener 是 Burp Suite 插件的接口。

sys 模块用于重定向标准输出。

定义 BurpExtender 类，该类实现了 IBurpExtender 和 IHttpListener 接口。

registerExtenderCallbacks 方法是插件的入口点，在其中进行初始化工作：

获取 Burp 插件 API 的回调对象 callbacks。

获取用于处理 HTTP 请求和响应的工具 helpers。

设置插件名称为 "Custom Content Printer"。

注册 HTTP 监听器，以便拦截和处理 HTTP 请求和响应。

将标准输出重定向到 Burp Suite 的 Extender-Output 窗口。

实现 processHttpMessage 方法，用于处理拦截的 HTTP 请求和响应消息：

首先检查是否为代理拦截的请求，确保只处理请求消息。

如果是请求消息：

获取请求消息的内容，并进行解析和处理：

获取请求头和请求体。

将请求体转换为字符串形式。

检查请求体中是否包含指定字符串。

如果存在指定字符串，将其替换为新的值。

更新请求消息的内容。

打印请求的头部和体到 Extender-Output 窗口。

如果是响应消息：

获取响应消息的内容，并进行解析和处理：

获取响应头和响应体。

打印响应的头部和体到 Extender-Output 窗口。

定义 print\_content 方法，用于将请求/响应内容转换为字符串并打印到 Extender-Output 窗口。