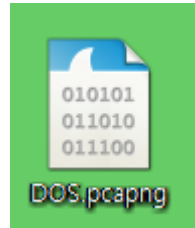


Write-Up

박준호

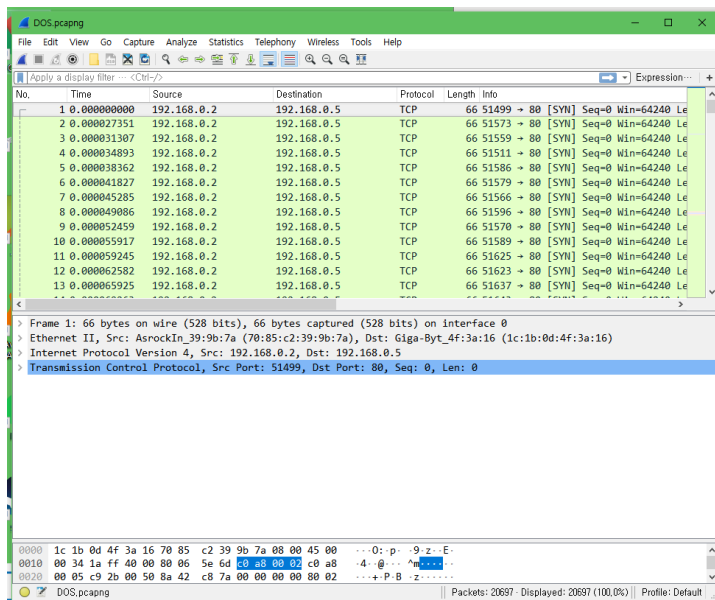
네트워크
DOS
100

->



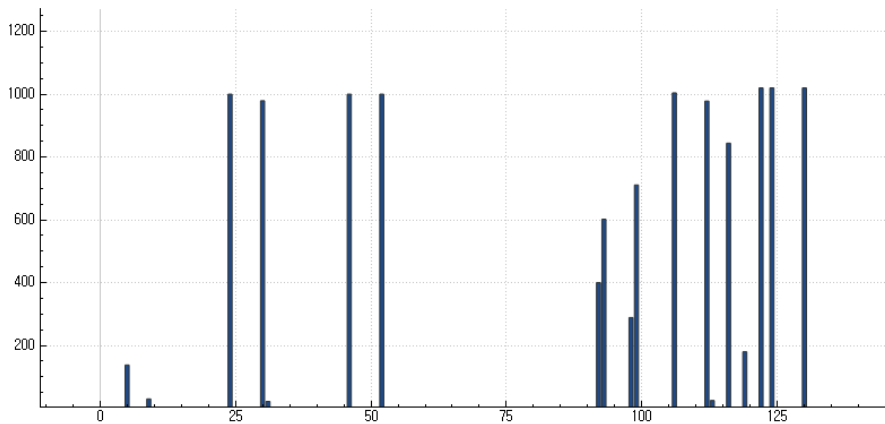
패킷 분석 문제.

이 파일을 와이어샤크로 열면

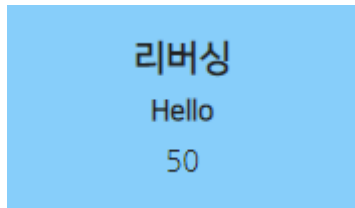


이게 나오고

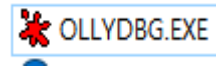
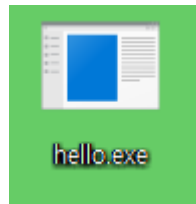
Statics -> I/O graph를 열면



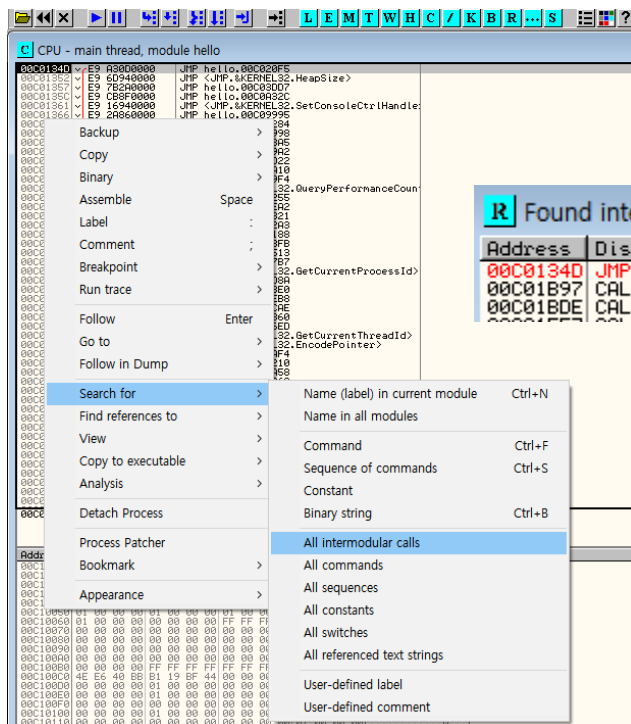
이게 19개.



->



50점짜리 리버싱, 이 파일을 ollydbg로 연다. 윈도우 64비트에서는 오류가 나므로 따로 설정을 해줄 필요가 있다. (애먹었음)



메세지를 출력하는 부분을 찾기 위해 함수 호출 부분만 골라본다.

Found intermodular calls		
Address	Disassembly	Destination
00C0134D	JMP hello.00C020F5	(Initial CPU selection)
00C01B97	CALL DWORD PTR DS:[&USER32.MessageBoxA	USER32.MessageBoxA
00C01BDE	CALL DWORD PTR DS:[&USER32.MessageBoxA	USER32.MessageBoxA

그러면 메시지 박스는 이 두가지밖에 나오지 않는다. 위의 주소로 이동하면

CPU - main thread, module hello		
00C01B85	74 23 JE SHORT hello.00C01BAA	
00C01B87	8BF4 MOV ESI,ESP	
00C01B89	6A 00 PUSH 0	
00C01B8B	68 1800C100 PUSH hello.00C10018	ASCII "Hello?"
00C01B8D	68 0000C100 PUSH hello.00C10000	ASCII "Wellcome to reversing!"
00C01B8F	6A 00 PUSH 0	
00C01B91	FF15 3C23C100 CALL DWORD PTR DS:[&USER32.MessageBoxA	USER32.MessageBoxA
00C01B93	3BF4 CMP ESI,ESP	
00C01B95	E8 2BF8FFFF CALL hello.00C013CF	
00C01B97	33C0 XOR EAX,EAX	
00C01B99	EB 45 JMP SHORT hello.00C01BED	
00C01B9B	EB 41 JMP SHORT hello.00C01BEB	
00C01B9D	68 2800C100 PUSH hello.00C10028	
00C01B9F	68 0000C100 PUSH hello.00C10000	ASCII "Wellcome to reversing!"
00C01BA1	E8 BBF8FFFF CALL hello.00C01474	
00C01BA3	83C4 08 ADD ESP,8	
00C01BA5	68 1800C100 PUSH hello.00C10018	ASCII "Hello?"
00C01BA7	68 0000C100 PUSH hello.00C10000	ASCII "Wellcome to reversing!"
00C01BA9	6A 00 PUSH 0	
00C01BAB	FF15 3C23C100 CALL DWORD PTR DS:[&USER32.MessageBoxA	USER32.MessageBoxA
00C01BAD	3BF4 CMP ESI,ESP	
00C01BAF	E8 E4F7FFFF CALL hello.00C013CF	
00C01BB1	33C0 XOR EAX,EAX	

이게 나오게 된다.

JE는 일단 의심하고 본다. 브레이크 걸어 두고 실행시키면 JE에서 멈추게 되는데.

00C01B83	85C0	TEST EAX,EAX	
00C01B87	74 23	JE SHORT hello.00C01BAA	
00C01B89	8BF4	MOV ESI,ESP	
00C01B8B	6A 00	PUSH 0	
00C01B8B	68 1800C100	PUSH hello.00C10018	ASCII "Hello?"
00C01B90	68 0000C100	PUSH hello.00C10000	ASCII "Wellcome to reversing!"
00C01B95	6A 00	PUSH 0	
00C01B97	FF15 3C23C100	CALL DWORD PTR DS:[&USER32.MessageBoxA	USER32.MessageBoxA
00C01B9D	3BF4	CMP ESI,ESP	
00C01B9F	E8 2BF8FFFF	CALL hello.00C013CF	
00C01BA4	33C0	XOR EAX,EAX	
00C01BA6	EB 45	JMP SHORT hello.00C01BED	
00C01BA8	EB 41	JMP SHORT hello.00C01BEB	
00C01BAH	68 2800C100	PUSH hello.00C10028	
00C01BAF	68 0000C100	PUSH hello.00C10000	
00C01BD4	EO 0000FFFF	CALL hello.00C01474	ASCII "Wellcome to reversing!"

여기서 이 부분을 실행시키면 hello와 주소 값이 같지 않으므로 점프하지 않는데, 이게 의심스러우므로 JE -> JMP로 바꾸고 프로그램을 실행시킨다

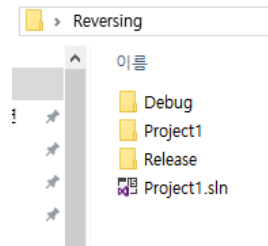
00C01B87	85C0	TEST EAX,EAX	
00C01B8B	EB 23	JMP SHORT hello.00C01BAA	
00C01B87	8BF4	MOV ESI,ESP	
00C01B89	6A 00	PUSH 0	
00C01B8B	68 1800C100	PUSH hello.00C10018	
00C01B8B	68 0000C100	PUSH hello.00C10000	

The screenshot shows a debugger window titled "CPU - main thread, module hello". The assembly list on the left shows the code after the modification, with the instruction at address 00C01B8B now being "JMP SHORT hello.00C01BAA" (highlighted in red). To the right, a message box titled "Answer" is displayed with the text "FLAG(Jmp_and_go_ah@d)" and a button labeled "확인". The comment column on the right shows the ASCII values of the pushed strings: "Answer", "FLAG(Jmp_and_go_ah@d)", and "Wellcome to reversing!".

이게 정답

리버싱
PPAP
100

->



100점짜리 리버싱 압축파일을 풀면 이게 나오는데

```
int main()
{
    string answer = "WELCOME!"; //answer
    /* vector insert to answer*/
    /*for (int i = 0; i < 9; i++)
    {
        int tmp = 0;
        for (int j = 0; j < 9; j++)
        {
            tmp += matrix[i][j] * apply[i];
        }
        answer += tmp;
    }*/

    if (answer == "")
    {
        cout << "COD3 : " << answer << endl;
    }
}
```

강 답이 적혀 있었음.

1번보다 쉬웠음.

시스템
트페의 밀장빼기
50

TP@xen.iscert.org 13872
pw : exciting_LOL

->

50점짜리 시스템 해킹. Xshell을 이용 저 서버로 들어가준다.

사용자 이름은 TP, 암호는 exciting_LOL. 13872는 포트.

```
[TP@system TP]$ ls
hint secret tmp TP
[TP@system TP]$ cat hint
#include <stdio.h>

int main()
{
    char card2[10];
    char card1[10];
    printf("input card : ");
    fgets(card1,40,stdin);

    if(strncmp(card1,"gold!",5))
    {
        printf("Not Gold!!!\n");
        return 0;
    }
    if(strncmp(card2,"sakura!",7) == 0)
    {
        setreuid(1042,1042);
        system("cat secret");
    }
}
```

[TP@system TP]\$

일단 안에는 이렇게 있고,

```
[TP@system TP]$ ./TP
input card : gold
Not Gold!!!
[TP@system TP]$
```

TP를 실행시키면 계속 이른다.

코드를 보면 입력은 card1만, secret에 접근하려면 card2에 sakura!가 저장 되어야 한다. = 오버플로우

TP 파일을 디버깅하자.

```
[TP@system TP]$ gdb TP
Reading symbols from /home/TP/TP...done.
gdb-peda$
[1]+  Stopped                  gdb -q TP
[TP@system TP]$ gdb TP
Reading symbols from /home/TP/TP...done.
gdb-peda$ set disassembly-flavor intel
gdb-peda$ disass main
```

->리눅스 디버거 검색하면 나옴

Disass main 하면 많이 나오는데 그중 중요한 건 [ebp - 0x28], [ebp - 0x18]부분

```

0x08048461 <+65>: lea     eax,[ebp-0x28]
0x08048464 <+68>: push   eax
0x08048465 <+69>: call   0x8048330 <strncmp@plt>
0x0804846a <+74>: add     esp,0x10
0x0804846d <+77>: test    eax,eax
0x0804846f <+79>: je      0x804848a <main+106>
0x08048471 <+81>: sub     esp,0xc
0x08048474 <+84>: push    0x8048590
0x08048479 <+89>: call    0x8048350 <printf@plt>
0x0804847e <+94>: add     esp,0x10
0x08048481 <+97>: mov     DWORD PTR [ebp-0x2c],0x0
0x08048488 <+104>: jmp     0x80484c9 <main+169>
0x0804848a <+106>: sub     esp,0x4
0x0804848d <+109>: push    0x7
0x0804848f <+111>: push    0x804859d
0x08048494 <+116>: lea     eax,[ebp-0x18]
0x08048497 <+119>: push    eax
0x08048498 <+120>: call    0x8048330 <strncmp@plt>

```

여기서 [ebp - 0x28] 후에 strncmp, [ebp - 0x18] 후에 strncmp가 나왔으므로 [ebp - 0x28]가 card1 이고 [ebp - 0x18]가 card2다 이 둘사이의 10진수로 16만 큼 차이 나고, card1이 10만큼 할당 하고 있으므로 card1에 16바이트를 입력 하게 되면 오버플로우가 일어나 card2에 저장되게 된다.

```

[TP@system TP]$ ls
hint secret tmp TP
[TP@system TP]$ cat hint
#include <stdio.h>

int main()
{
    char card2[10];
    char card1[10];
    printf("input card : ");
    fgets(card1,40,stdin);

    if(strncmp(card1,"gold!",5))
    {
        printf("Not Gold!!!\n");
        return 0;
    }
    if(strncmp(card2,"sakura!",7) == 0)
    {
        setreuid(1042,1042);
        system("cat secret");
    }
}
[TP@system TP]$

```

```

[TP@system TP]$ ./TP
input card : gold!12345123456sakura!

mIt_JJang_BBae_Gi

[TP@system TP]$

```

첫 if 문에서 card1의 5글자가 gold!가 아니면 문장을 출력하고 return 0 하므로 첫 5글자는 gold!, 나머지 11글자를 아무 값이나 넣은 후 sakura!를 입력하게 되면 card2에 sakura!가 저장 되고 secret 파일을 열어준다.

암호 Alphabet_Puzzle 50

k : HACK???????

c : ZTCBB : NGTIQYKDGONSMYOVEPJLUSYRETHPBUXFEJDPFMVG

p : ST??? : ?????CD????????II????????TU????????Z?

k = key, c = ciphertext, p = plaintext , MD5(p)= cc5ce030c3fa85c3671064d9dd13ffff

p????

50점짜리 암호. 처음엔 몰랐는데 힌트 나와서 알게 됐다. 비즈네르 암호.

원문	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
1	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
2	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
3	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
4	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
5	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
6	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
7	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
8	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
9	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
10	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
11	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
12	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
13	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
14	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
15	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
16	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
17	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
18	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
19	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
20	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
21	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
22	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
23	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
24	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
25	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
26	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z

키워드	S	K	Y	S	K	Y	S	K	Y	S	K	Y	S	K	Y	S	K	Y	S	K	Y	S	K
원문	d	i	v	e	r	t	t	r	o	o	p	s	t	o	e	a	s	t	r	i	d	g	e
암호문	V	S	T	W	B	R	L	B	M	G	Z	Q	L	Y	C	S	C	R	J	S	B	Y	O

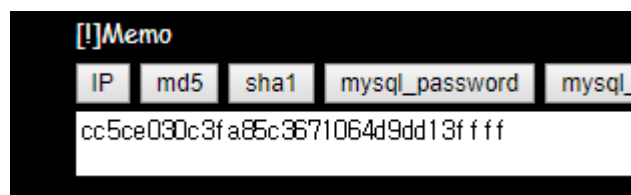
이런 식으로 테이블이 있는데 key, 평문을 조합해서 암호문을 만든다.

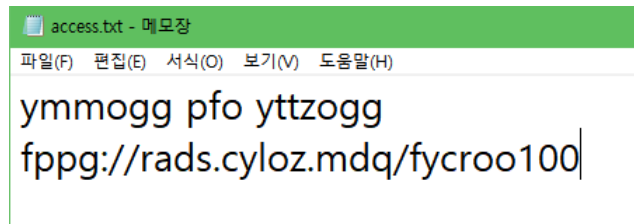
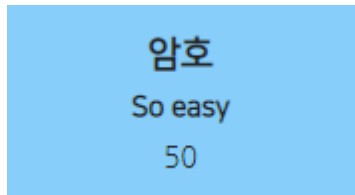
그러므로 key 암호문이 있다면 평문으로 해독하는 것도 가능하다.

그래서 열심히 해독 하니까

START:AABBCCDDEEFFGGHHIIJJKKLLMMTTUUVVWWXXYYZZ 가 나왔다.

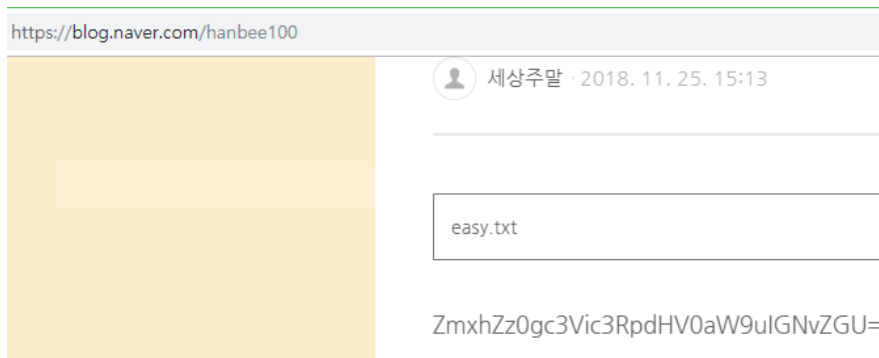
이걸 md5로 디코딩 시키면



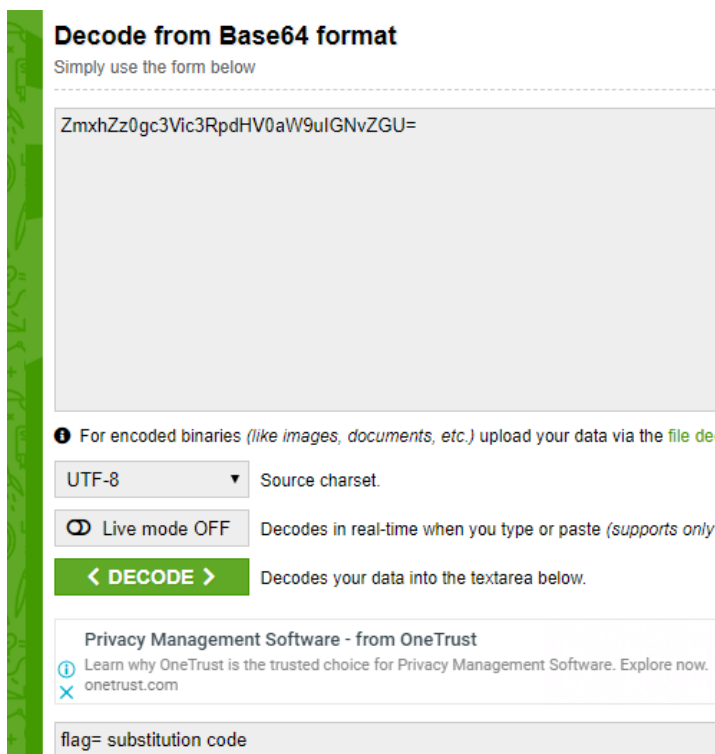


50점짜리 암호. 일단 (fppg -> http) (mdq -> org or com) (pfo -> the)로 추측.

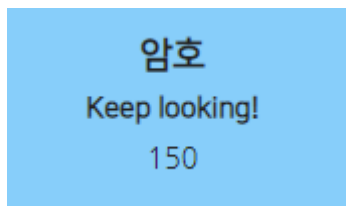
그럼 mdq 뒤가 h???ee100 이 됨 여기서 운영진 이름 하나씩 짚어보다가 정한비
생각남 그럼 (fycroo -> hanbee). (cyloz -> na?e? -> naver) (rads -> b?o? -> blog)



-> 끝에 =가 붙으면 base64.



끝.



제일 짜증났던 문제. 필독에 키워드를 커피라고 줘서 그걸로 비즈네르 엄청 돌렸는데 안 나와서 컴플레인 넣고 싶었다.

비즈네르암호 해독 해주는 사이트가 있어서 거기서 때려 맞췄다.

Input

Cipher Text:
lu euz hdo jkhnswwx ymlh ixduwd, euz blar mgy d egyxborgj
stamt pupffwo tjr

Cipher Variant: Classical Vigenere
Language: English
Key Length: 3-30
(e.g. 8 or a range e.g. 6-10)

Break Cipher Clear Cipher Text

Result

Clear text [hide]
Clear text using key "dpggff":

if you can decipher this crypto, you will get a password monje
jokachi nem

미친 사이트.

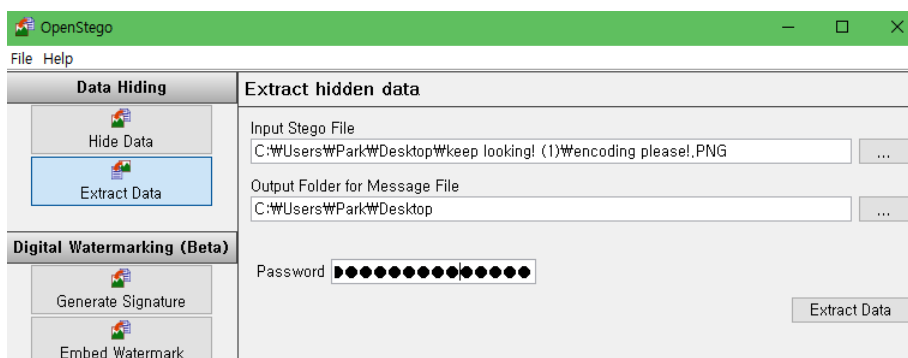
평문에 key까지 다 나왔다.

Key 보고 더 열 받았다.

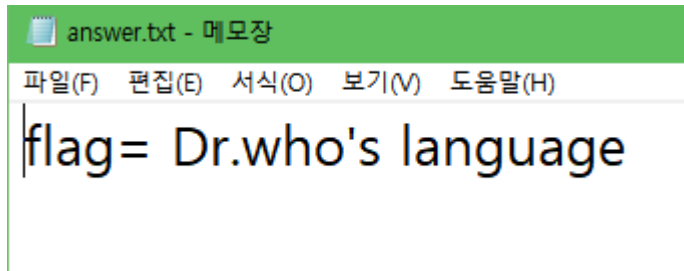
문제 진짜.

아무튼 패스워드가 나왔는데

비즈네르 암호 해독하면서 스테가노 그래피에 대해서도 엄청 찾아서 프로그램도 찾아 뒀었다. 그래서 패스워드 보자마자 눈치 챘다.



이 사진과 함께 패스워드를 입력함으로써 사진에 숨겨져 있는 텍스트가 나오게 된다.



그렇게 나온 결과 값.

처음에 openstego에서 추출할 때 계속 오류 뜨길래 내가 잘못 풀었나 했는데 버전이 달랐었음.

웹

가위바위보

50

Let's pick a winner through rock scissors paper.

Rock, Scissors, Paper

Submit

50점짜리 웹 해킹. 무조건 지는 가위바위보 게임 이기기

210.117.183.125:10021 내용:

You lose...OTL the computer gave out paper

확인

일단 뭐를 내든지 무조건 짐.

웹 하면 버프슈트지.

프록시 서버

☒ 사용자 LAN에 프록시 서버 사용(이 설정은 전화 연결이나 VPN 연결에는 적용되지 않음)(X)

주소(E): 127.0.0.1

포트(T): 8080

고급(C)

☐ 로컬 주소에 프록시 서버 사용 안 함(B)

확인

취소

기본적으로 프록시 설정 해준다.

그 다음 버프슈트를 키고 바위를 내면 이런 식으로 쿠키가 잡히는데

Rock

Target

Proxy

Spider

Scanner

Intruder

Repeater

Sequencer

Intercept

HTTP history

WebSockets history

Options

Request to http://210.117.183.125:10021

Forward

Drop

Intercept is on

Action

Raw

Params

Headers

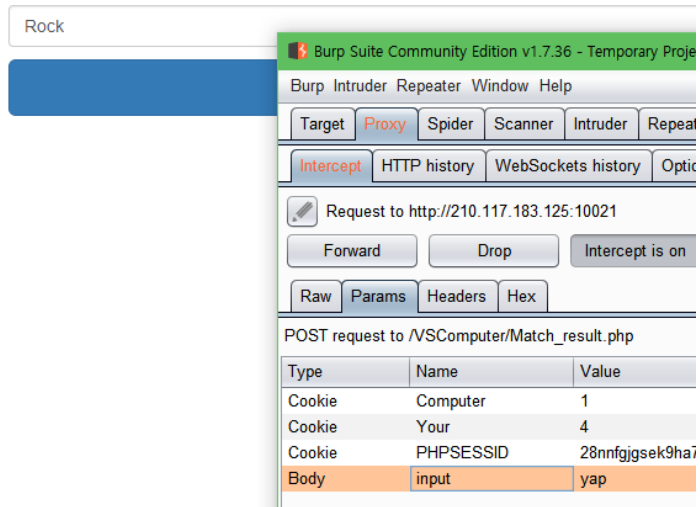
Hex

POST request to /VSComputer/Match_result.php

Type	Name	Value
Cookie	Computer	4
Cookie	Your	1
Cookie	PHPSESSID	28nnfgjgsek9ha74mrm69k3ki1
Body	input	Rock

내가 무언가를 내면 컴퓨터는 항상 저렇게 낸다. 그럼 값을 바꿔버리면. 되겠다.

Let's pick a winner through rock scissors p



일단 서로 값 바꾸고 input에 아무 값이나 넣었더니 정답을 줬다.

사실 뭘 바꿔야 하는지 기억이 나질 않아서 그냥 다 바꿔버렸다.

210.117.183.125:10021 내용:

flag{Cookie-_-Cookie}

확인