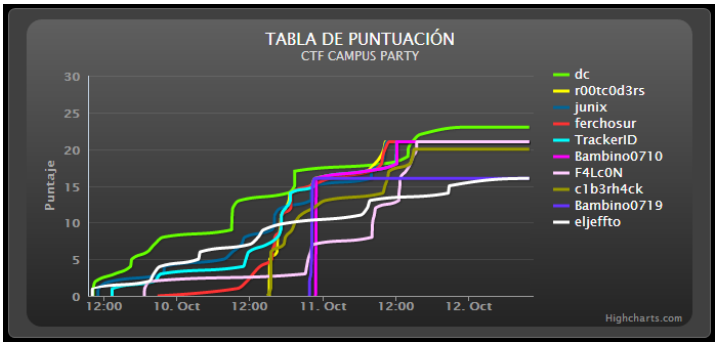
CTF Campus Party Colombia 2013

Autor: Daniel Correa (usuario wargame: dc)

Resultado general

Tanto en el juego presencial como en la gráfica se evidencia el juego no limpio, la gráfica con los valores reales queda a consideración de los organizadores.



Reto 1

Descripción

Y si lA cuLturA es cambianTe, como es eso de que la hIstoria sE repite?
¿será acaso uNa contradicción Epocal de los cientiStas sociales?

Solución

Se observan las letras que están en mayúsculas, al unirlas todas se obtiene la bandera:
Y si l**A** cu**L**tur**A** es cambian**T**e, como es eso de que la h**I**storia s**E** repite?
¿será acaso u**N**a contradicción **E**pocal de los cienti**S**tas sociales?

Respuesta

YALATIENES

Reto 3

Descripción

http://ctf.csiete.org/retos/reto3/

Solución

Se ingresa al sitio en la descripción y se verifican las cabeceras del protocolo HTTP, en una de ellas (**Set-Cookie**) se observa el valor de la bandera:

```
GET http://ctf.csiete.org/retos/reto3/
   Cabeceras pedidas:
       Host[ctf.csiete.org]
       User-Agent[Firefox/17.0]
       Accept[text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8]
       Accept-Language[en-US,en;q=0.5]
       Accept-Encoding[gzip, deflate]
       Cookie[PHPSESSID=0v8j4oq0bregu06op4jljkf4q1]
       Connection[keep-alive]
   Cabeceras recibidas:
       Date[Wed, 09 Oct 2013 19:45:01 GMT]
       Server[Apache/2.2.22 (Debian)]
       X-Powered-By[PHP/5.4.4-14+deb7u4]

Set-Cookie[AuthCookie=USER%3Anonroot%26PASS%3Anonroot123%26FLAG%3ACTFCAMPUSPARTY2013;
```

```
expires=Wed, 09-Oct-2013 19:45:02 GMT]
```

Respuesta

CTFCAMPUSPARTY2013

Reto 4

Descripción

Tienes razón, este es el reto4, usa esta pista:
505118809428593473164096519092019535 90

Solución

Se convierte el número decimal a formato hexadecimal:

```
C:\>python
Python 2.7.3 (default, Apr 10 2012, 23:24:47) [MSC v.1500 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> hex(505118809428593473164096519092019535 90)
'0x26003c0200000000f03c91fffedb0336L'
```

El número hexadecimal corresponde a una dirección IPv6:

```
2600:3c02:0:0:0:0:f03c:91ff:fedb:0336
```

Se ingresa a la ruta http://[2600:3c02:0:0:f03c:91ff:fedb:336]/retos/reto4/ y el servidor responde con el siguiente texto:

```
OK, your FLAG IS: ESTOESBRUJERIAIPV6!
```

Respuesta

ESTOESBRUJERIAIPV6!

Reto 5

Descripción

What is the smallest positive number that is evenly divisible by all of the numbers from 1 to 1337 (Give us a lowercase md5 digest)

Solución

El enunciado pertenece a un reconocido problema matemático que se encuentra en Project Euler (http://projecteuler.net/problem=5), el script usado para obtener el MD5 del menor número entero divisible exactamente por todos los números desde 1 hasta 1337 es:

```
import hashlib

def gcd(a, b):
	"""Return greatest common divisor using Euclid's Algorithm."""
	while b:
	a, b = b, a % b
	return a

def lcm(a, b):
	"""Return lowest common multiple."""
	return a * b // gcd(a, b)

def lcmm(*args):
	"""Return lcm of args."""
	return reduce(lcm, args)

def lcm_seq(seq):
	"""Return lcm of sequence."""
	return reduce(lcm, seq)

solution = lcm_seq(xrange(1,1337))
print hashlib.md5(str(solution)).hexdigest()
```

Respuesta

5a4649a23568e47a9272ba5694a0cda8

Reto 6

Descripción

http://ctf.csiete.org/retos/reto6/?text=fail

Solución

Al ingresar al sitio provisto en la descripción, se puede observar el siguiente mensaje:

```
Lo siento, fail no es una marca registrada!, tendras que mejorar tu ataque.!
```

Cuando se modifica el valor de la variable text, se ve el cambio en el mensaje mostrado:

```
Lo siento, <valor modificado> no es una marca registrada!, tendras que mejorar tu ataque.!
```

Luego de un número considerado de intentos fallidos en el envío de la variable text, y de leer continuamente el mensaje mostrado, se concluyó que se debía pasar como valor una marca registrada, identificada con el símbolo ™:

```
http://ctf.csiete.org/retos/reto6/?text=marca™
```

El servidor responde el siguiente texto:

```
OK, your FLAG IS: FLAGSFACILESPARAGENTELISTA:)
```

Respuesta

FLAGSFACILESPARAGENTELISTA:)

Reto 7

Descripción

¿Cuántas Campus Party se han hecho en Colombia?

Solución

Se han realizado 6 Campus Party oficiales en Colombia y un evento llamado Campus Party Boya.ca, la flag aceptada por el sistema fue **SIETE**.

Respuesta

SIETE

Reto 8

Descripción

Definitivamente una flag (bandera) puede ser cualquier cosa.
http://www.cloudshark.org/captures/9b6bb3686322

Solución

Se realizó un análisis sobre el archivo PCAP que podía ser descargado de la dirección provista, sin resultado alguno. Dicho archivo pertenece a una copia del original que puede ser encontrado en la siguiente URL: http://packetlife.net/captures/TCP_SACK.cap

Dado que el archivo original fue cargado desde el año 2010, se descartó que dentro de dicha captura se encontrara la respuesta al reto.

Se concluye que la bandera está contenida en la misma descripción.

Respuesta

9b6bb3686322

Reto 9

Descripción

XhRvgV vM xzNkFh KzIgB

Solución

El reto fue resuelto usando una herramienta de criptografía clásica que programé hace unos años: http://www.sinfocol.org/herramientas/cryptos.php#hebraicas

El texto fue cifrado mediante la cifra Atbash (http://www.sinfocol.org/2008/10/cifras-hebraicas-atbash-albam-y-atbah/), al descifrarlo obtenemos la bandera.

Respuesta

CsIetE eN caMpUs PaRtY

Reto 10

Descripción

http://ctf.csiete.org/retos/reto10/reto10.pdf

Solución

Al descargar el archivo, se observa que la cabecera distintiva de los archivos PDF se encuentra al final del archivo %PDF-1.4, y que el archivo comienza por %%EOF. El archivo está invertido según el carácter de nueva línea (0x0a).

El siguiente script es usado para reordenar los datos contenidos en el archivo PDF:

```
<?php

$pdf = file_get_contents('reto10.pdf');
$pdf = explode("\n", $pdf);

$out = '';
for ($i = count($pdf) - 1; $i >= 0; $i--) {
       $out .= $pdf[$i] . "\n";
}

file_put_contents('reto10.ordenado.pdf', $out);
```

Al abrir el archivo PDF se observa la siguiente imagen:

Después de varios intentos fallidos se da con la respuesta verdadera.

Respuesta

Nice0NE!

Reto 11

Descripción

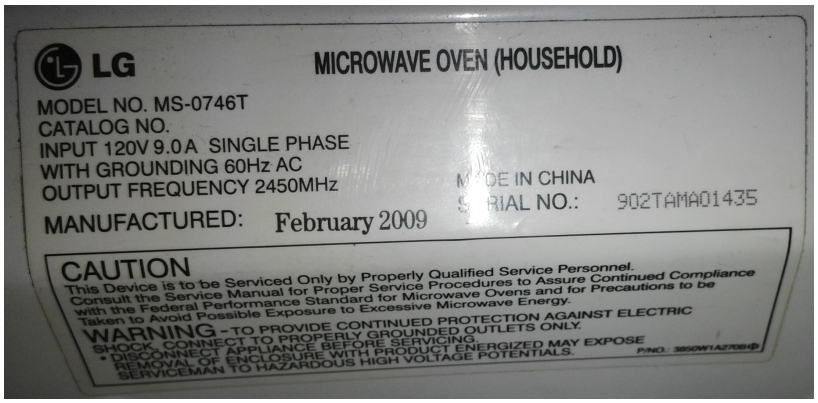
¿Cuál es el serial del elemento que se encuentra en la arena donde esta nuestra área pitágoras

que puede ser usado en una toma de evidencia forense para evitar la contaminación por tráfico de dispositivos móviles incautados?. (by @fixxx3r)

Solución

El elemento usado para evitar la contaminación por tráfico de dispositivos móviles incautados es un microondas encontrado debajo de una de las mesas en el escenario pitágoras, dicho microondas funciona como una jaula de Faraday.

La siguiente imagen es una foto de la parte trasera del microondas, donde se puede evidenciar su serial:



Respuesta

902TAMA01435

Reto 12

Descripción

Decode me #1:

4D3946514139534840393741542 66C743B46254F266C743B4631493B4625522667743B32214 7393659493D372C4038365944282 6616D703B59413A3739452826616D703B454E38565D4D2 66C743B26616D703B35540D0A2439365943393060600D0A60 (by @ftbfs)

Solución

Se convierte la cadena entregada en el reto de hexadecimal a ASCII, se obtiene el siguiente texto:

```
M9FQA9SH@97AT<F%O<F1I;F%R>2!G96YI=7,@86YD(&YA:79E(&EN8V]M<&5T
$96YC90``
`
```

Se decodifica utilizando la herramienta **uudecode**:

```
$ cat uueflag
begin 644 flag
M9FQA9SH@97AT<F%O<F1I;F%R>2!G96YI=7,@86YD(&YA:79E(&EN8V]M<&5T
$96YC90``
`
$ uudecode uueflag
$ cat flag
flag: extraordinary genius and naive incompetence
```

Respuesta

extraordinary genius and naive incompetence

Reto 13

Descripción

Decode me #2:

MDExMDAxMTAwMTEwMTEwMDAxMTAwMDAxMDExMDAxMTEwMDExMTAxMDAwMTAwMDAwMDExMTEwMDEwMTEwMTExMTAxMTEwMTAxMDAxMDAwMDAwMTEwMDAxMTAxMDExMDEwMDEwMDAwMDAxMTEwMDExMDExMDExMTEwMTEwMTExMDAxMTAwMDAxMTAwMTEwMDAxMDEwMDEwMDAwMDAxMTAwMDAxMDExMDExMTAwMTExMTAwMTExMTAwMTEwMTEwMTAwMDAxMDEwMDAwMDAxMTEwMDAwMDExMTEwMTExMDAxMDAxMTAwMTAxMTAwMDAxMDEwMDExMDExMTAwMTAxMTAxMDAxMTAxMTEwMTExMDEwMDAxMDExMTAwMTAwMTEwMDAwMDAxMTAwMDAxMDEwMTEwMTEwMDExMTAwMDAxMDAwMDEwMTAwMTEwMTAxMTEwMTExMDExMDAwMTEwMDAwMTEwMTAxMTAxMDAwMTAwMTEwMTAxMDExMTEwMDAxMTAxMDEwMDAxMTAxMDEwMTEwMDAwMDAxMDEwMDAwMTEwMTEwMDEwMTAxMTEwMTEwMTAwMDAwMDAwMTEwMTAwMDEwMTAxMTEwMDEwMTAxMDAwMTEwMDAwMTAxMTAxMTEwMTEwMDEwMTAxMTEwMTEwMDEx (by @ftbfs)

Solución

Se usa la herramienta SNEAK (http://snarkles.net/scripts/sneak/sneak.php), para inicialmente decodificar el texto a través del algoritmo Base64:

```
0110011001101100011000010110011100111010001000000111100101101111011101010010000001100011
0110000101101110001000000111001101101111011011000111011001100101001000000110000101101110
0111001001000000110110101100001011010100110111101110010001000000111000001110010011011110110
0001001101100011001010110110101110011001000000110101001110101011100110111010000100000111
```

```
01110110100101110100011010000010000001110000011011110111010001100001011101000110111101100
10101110011
```

Finalmente, se usa la misma herramienta para obtener el ASCII del binario:

```
flag: you can solve any major problems just with potatoes
```

Respuesta

you can solve any major problems just with potatoes

Reto 14 y 15

Descripción

Nuestra unidad de inteligencia tuvo acceso a información privilegiada con la cual se pudo determinar que durante la charla programada para las 18:00 del Jueves 10 de Octubre en el escenario Pitágoras una persona infiltrada estará extrayendo información confidencial. Su misión, si decide aceptarla, es identificar a esta persona (que seguramente lucirá un poco sospechosa) y usando técnicas de ingeniería social obtener los datos necesarios para capturar las siguientes flags:

Flag 14: md5(“color de zapatos | color de camisa/blusa/trusa”)

Ej: si los zapatos de esta persona fueran verdes y la camisa amarilla, el md5 sería: md5(verde|amarillo) = 5b2673e0ede499047b6e9ab2e9cb2670

Flag 15: md5(“nombre de la mascota de esta persona”)

Ej: si el hipopotamo de mascota se llamara “chiqui” el md5 sería: md5(“chiqui”) = 0dea3d9e9f4f257a08475bdb7c030531

Solución

El reto se realizó mediante la observación del comportamiento del público en la charla, durante el transcurso de la misma se identificaron varias personas quienes tomaban fotos, hablaban entre si, o tenían algún tipo de comportamiento extraño.

Teniendo en cuenta las pistas del reto, solo uno de estos asistentes inusuales era una mujer, por lo que se descartan los hombres y se enfocan los esfuerzos en realizar un acercamiento.

En el acercamiento se utiliza el desconocimiento y el interés para preguntar el motivo de la asistencia al evento. Se realizaron preguntas como si fuera fuese un asistente regular o si le interesa la charla, el nivel de conocimiento en temas de seguridad y si conoce de qué se trata el CTF. Ante respuestas ambiguas o inusuales como, “Soy experta en el tema” y “No sé qué es

un CTF”; “Solo asisto a este evento” y “Mi empresa patrocinó un evento similar”; “Me gustan los animales” y “Mi perro se llama Valentino”; “¿A qué te dedicas?” y dar información de los proyectos de los ponentes; dan a entender que variables como la experticia en estos eventos, con la experiencia en seguridad y no saber de qué se trata un “capture the flag” o hablar de las mascotas y mencionar sus nombres son variables a tener en cuenta para la información del reto.

El resto de las preguntas se enfocaron en valerse del desconocimiento o inocencia para identificar si las anteriores preguntas se podían corroborar, por ejemplo, preguntar “¿a qué te dedicas?”, luego “¿qué hace tu empresa?”, luego “¿por qué vienes a esta charla específicamente?”. Si todas las respuestas son pensadas antes de responder, aumentan las posibilidades de que sea la persona que se está buscando. Adicionalmente se hacen preguntas de control para identificar el comportamiento de la persona al responder una u otra pregunta.

Finalmente el color de los zapatos (Café) y la blusa (Azul) se solucionan por simple observación.

Respuesta

md5("cafe|azul") = cacff0d9719d8a58709b21011fa7b0d2
md5("valentino") = 50b483d799f6b531772078e9cd0fa509

Reto 17

Descripción

La suma de los primos menores que 10 es: 2+3+5+7=17 Encuentre la suma de todos los primos menores que 10 millones. (by @ftbfs)

Solución

Se implementó el siguiente código para resolver el reto usando Java:

```
public class Reto17 {
    public static void main(String[] args) {
        Integer n = 2;
        Long primeSum = 0L;

        while (n < 10000000) {
            if (isPrime(n)) {
                primeSum += n;
            }
            n++;
       }
```

```
        System.out.println("La suma es " + primeSum);
    }

    public static Boolean isPrime(Integer number) {
        if (number <= 0 || number == 1 || (number % 2 == 0 && number != 2)) {
            return false;
        } else {
            for (Integer i = 3; i < Math.floor(Math.sqrt(number))+1; i += 2) {
                if (number % i == 0) {
                    return false;
                }
            }
        }
        return true;
    }
}
```

Ejecutando el programa compilado, se obtiene la bandera en menos de un par de minutos:

```
$ java Reto17
La suma es 3203324994356
```

Respuesta

3203324994356

Reto 18

Descripción

really simple arithmetic: Find the password for the username “ekoparty2013”. A sample username and password are given below. (Recordando la EKO ;)

```
username = "ekoparty"

password = "dada50f87fd9bf5669112bec92dd"
n = 0x10ec6f04a1271aa993f663aa0a2cfL
d = 0x302d0d91c7b585df5257b0a59363bL
m = sum(ord(c) ** i for c in username for i in range(512)) % n
c = int(password, 16);

if pow(c, d, n) == m:
     print "correct :)”
else:
     print "incorrect :("
```

Solución

El script contiene uno de los algoritmos más populares de criptografía asimétrica RSA con una pequeña variación. Se debe obtener la variable E a través del inverso multiplicativo usando los factores de N y el valor de D para calcular el password correcto del usuario “ekoparty2013”.

Se usa el siguiente script para obtener el password:

```
import gmpy

username = "ekoparty2013"

n = 0x10ec6f04a1271aa993f663aa0a2cfL
d = 0x302d0d91c7b585df5257b0a59363bL
m = sum(ord(c) ** i for c in username for i in range(512)) % n

# n = 549201213511374249930668140557998 3
# n = 420001 * 420029 * 420037 * 420041 * 420047 * 420073
# phi(n) = phi(420001) * phi(420029)  * phi(420037) * phi(420041) * phi(420047) *
phi(420073)
#
http://www.wolframalpha.com/input/?i=phi%28420001%29+*+phi%28420029%29+*+phi%28420037%29+
*+phi%28420041%29+*+phi%28420047%29+*+phi%28420073%29

phi_n = 549193368536212036157785313280000 0

e = gmpy.invert(d, phi_n)

c = pow(m, e, n)

if pow(c, d, n) == m:
       print "correct :)"
       print hex(c)
else:
    print "incorrect :("
```

Respuesta

dea7152f1dbd3e7cc834182c5c0e

Reto 19

Descripción

¿Y a que horas fue que llegó la primera campusera de #CPCO06 ?

Solución

En las noticias se encontró esta información, Sara Bustamante, primera campusera en llegar a las 4 de la mañana:

http://wradio.com.co/noticias/sociedad/una-mujer-fue-la-primera-campusera-en-llegar-al-cpco6/20130710/nota/1990228.aspx

Respuesta

4AM

Reto 20

Descripción

¿Un poco de network forensics?, http://ctf.csiete.org/files/giveme.pcap

Solución

Realizando un análisis del archivo **giveme.cap**, se encuentra información viajando a través de UDP:

```
GIVE (Source port 57181)
ACCESS CODE: 17097119
GIVE ME (Source port 59019)
ACCESS CODE: 28152063
GIVE ME A CAR (Source port 52734)
ACCESS CODE: 43769220
GIVE ME THE POWER (Source port 62620)
ACCESS CODE: 72827060
GIVE ME THE LOGINS (Source port 57162)
ACCESS CODE: 70080612
GIVE ME THE PASSWORDS (Source port 65453)
ACCESS CODE: 96608628
GIVE ME THE HINT (Source port 52684)
ACCESS CODE: 56529932
GIVE ME THE FUCKING HINT (Source port 53290)
ACCESS CODE: 86542960
GIVE ME THE ACCESS (Source port 52806)
ACCESS CODE: 63367200
GIVE ME SOMETHING (Source port 60450)
ACCESS CODE: 72237750
GIVE ME SOMETHING PLEASE (Source port 65217)
ACCESS CODE: 108847173
```

```
GIVE ME THE FLAG (Source port 61005)
```

Al analizar cada respuesta se puede correlacionar que el código de acceso es múltiplo del puerto de origen y de la suma de códigos ASCII de los caracteres contenidos en los datos del paquete UDP. El siguiente script fue usado para obtener el código de acceso para la cadena “GIVE ME THE FLAG” y el puerto origen UDP 61005:

```
<?php

$s = $argv[1];
$p = $argv[2];

$n = 0;
for ($i = 0; $i < strlen($s); $i++)
    $n += ord($s[$i]);

echo $p * $n;
```

Se ejecuta con los parámetros requeridos para verificar códigos de acceso anteriores, y el código de acceso solicitado:

```
$ php giveme.php "GIVE" 57181
17097119
$ php giveme.php "GIVE ME THE FLAG" 57181
59925688
$ php giveme.php "GIVE ME THE FLAG" 61005
63933240
```

Respuesta

63933240

Reto 22

Descripción

¿Cúal es el pool (rango/CIDR) completo de direcciones IPV4 usadas en la red de #CPCO06 ?
(reto interno / Ingeniería social)

Solución

Se habló con los encargados de IData Center para requerir la información solicitada, inicialmente la respuesta fue que debía recorrer y conectarme en cada mesa del establecimiento para identificar los valores mínimos y máximos del direccionamiento. Se habló

de nuevo con ellos para evitar la labor de recorrer el establecimiento y finalmente proporcionaron los datos solicitados.

El rango de direcciones IPv4 utilizado en Campus Party fue: **190.165.160.0/19**

Respuesta

190.165.160.0/19

Reto 24

Descripción

Pista: “ekoparty - Electronic Knock Out Party - Security Conference, is a great event in South America. FTW!”

Solución

Descargar el archivo: http://ctf.csiete.org/files/data.

Al descargar el archivo se ve que comienza por una serie de signos de interrogación, se utilizan símbolos que corresponden a un lenguaje de scripting bash ($_, $12, $?).

```
????????\ ?\ ??????????????????????????-??????????????????????\ ??\ ???????\ ????????\
??????????????????;_____=$_;__________=$[++_______];__=$__________;__0__0__+=1;_________
_+=3;___=$__;___=$[++___];
_______=_______;_______=$[++______];${_____:__________:______}$3h${_____:___:__0__0__}
;___=$__;___=$[++___];__1__1__=__0__0__;
__=$_____;________=$19;________+=6;____=$__________;__________=$[--__________];__________
=$[--__________];__________=$[--__________];__________=$[--__________];__________=$[--___
_______];__________=$[--__________];__________=$[--__________];__________=$[--__________]
;__________=$[--__________];__________=$________;__________=$[++__________];_0_0_="${_
____:____:______}$9h${_____:___:__0__0__} ${__:__________:__1__1__}$6h${__:0:__1__1__}
${__:________:__1__1__}${__:$[--____]:_______}${_____:__________:1}";__=$x86;___=$_0_0_;_
_+=66;______=$__;_______=$__;__=$[--__];__=$[--__];____=$_;__=$[--__];__=$[--__];__=$[--_
_];_______=$[--_______];_______=$[--_______];_______=$[--_______];_______=$[--_______];__
_____=$[--_______];$_0_0_${_____:______:1} ${_____:_______:$[++$12]}: .${_____:$(($__ +
0x14))}
;_______=$[--_______];_______=$[--_______];__=$[--__];__=$[--__];morse_code=$?;__=$______
____;__0__0__+=1;__________+=3
```

Se puede identificar que la pista entregada tiene una similitud en longitud con los signos de interrogación:

```
????????\ ?\ ??????????????????????????-??????????????????????\ ??\ ???????\ ????????\
??????????????????
```

```
ekoparty - Electronic Knock Out Party - Security Conference, is a great event in South
America. FTW!
```

La longitud de los datos concuerdan perfectamente, pero se observan diferencias, se utiliza el metacaracter \ para escapar algunos de los espacios. Se reemplaza los signos de interrogación por la pista entregada, haciendo uso de (\ ) en todos los espacios.

```
ekoparty\ -\ Electronic\ Knock\ Out\ Party\ -\ Security\ Conference,\ is\ a\ great\
event\ in\ South\ America.\
FTW!;_____=$_;__________=$[++_______];__=$__________;__0__0__+=1;__________+=3;___=$__;__
_=$[++___];
______=_______;______=$[++______];${_____:__________:______}$3h${_____:___:__0__0__}
;___=$__;___=$[++___];__1__1__=__0__0__;
__=$_____;________=$19;________+=6;____=$__________;__________=$[--__________];__________
=$[--__________];__________=$[--__________];__________=$[--__________];__________=$[--___
_______];__________=$[--__________];__________=$[--__________];__________=$[--__________]
;__________=$[--__________];___________=$________;___________=$[++___________];_0_0_="${_
____:____:______}$9h${_____:___:__0__0__} ${__:___________:__1__1__}$6h${__:0:__1__1__}
${__:________:__1__1__}${__:$[--____]:_______}${_____:__________:1}";__=$x86;___=$_0_0_;_
_+=66;______=$__;_______=$__;__=$[--__];__=$[--__];____=$_;__=$[--__];__=$[--__];__=$[--_
_];_______=$[--_______];_______=$[--_______];_______=$[--_______];_______=$[--_______];__
_____=$[--_______];$_0_0_${_____:______:1} ${_____:_______:$[++$12]}: .${_____:$(($__ +
0x14))}
;_______=$[--_______];_______=$[--_______];__=$[--__];__=$[--__];morse_code=$?;__=$______
____;__0__0__+=1;__________+=3
```

Se ejecuta el archivo resultante como un script de bash, y se obtiene la bandera para solucionar el reto:

```
$ bash _data
_data: line 1: ekoparty - Electronic Knock Out Party - Security Conference, is a great
event in South America. FTW!: command not found

The Flag is: .South America. FTW!
```

Respuesta

.South America. FTW!

Reto 26

Descripción

Necesitas la segunda llave para pasar este reto: http://ctf.csiete.org/files/captura.pcap

Solución

El archivo descargado es un archivo de texto ASCII que contiene datos de intercambio en MIFARE.

Analizando el protocolo según la especificación de MIFARE http://www.nxp.com/documents/data_sheet/MF1S503x.pdf, se identifican los intercambios de información para realizar autenticación.

El contenido interesante dentro del protocolo:

```
 +     688:   :      93  20
 +     66:   0: TAG 9e  cc  96  df  1b
 +   1984:    :      93  70  9e  cc  96  df  1b  dd  07
...
 +     112:   0: TAG 50  bf  94  a2
 +     668:   :      1b  f8  aa  84  8c  fc  fa  6a   !crc
 +     64:   0: TAG d0  01  b7! c1!
```

Se puede obtener la segunda llave utilizando la herramienta **mfkey** del proyecto **proxmark3:** https://code.google.com/p/proxmark3/source/browse/branches/cdc/tools/mfkey/?r=669

```
$ ./mfkey64 9ecc96df 50bf94a2 1bf8aa84 8cfcfa6a d001b7c1
MIFARE Classic key recovery - based 64 bits of keystream
Recover key from only one complete authentication!

Recovering key for:
  uid: 9ecc96df
   nt: 50bf94a2
 {nr}: 1bf8aa84
 {ar}: 8cfcfa6a
 {at}: d001b7c1

LFSR succesors of the tag challenge:
  nt': 8d787b12
 nt'': 39bc5169

Keystream used to generate {ar} and {at}:
  ks2: 01848178
  ks3: e9bde6a8

Found Key: [e69275506ffe]
```

Respuesta

e69275506ffe

Reto 28

Descripción

UNAFLAGREGALADAMEHACE???FELIZ

Solución

Es cuestión de adivinar que los signos de interrogación corresponde a la palabra MUY.

Respuesta

UNAFLAGREGALADAMEHACEMUYFELIZ

Reto 29

Descripción

leeloo dallas multipad: The following cryptograms were intercepted last night. The encryption key changes every five minutes. It is believed all of them share the same key, but it is unknown whether or not they are salted:

['R0f9LTuk4BAgo4T1YofCwH4Haz+aVAw4e6y+T7RCzJyF93IsLckjwf6tNnGzSFcXKFVjEo9tp5tK7RAyl2fc3Vd6ABCVTvd0C5if6iOa','NDSobmntpVUs7YT4YovCjn4EayKaTgxse7W+RbRezNOFsnItLcgjlf6xNn6zDFcBKBZjFI9up9VK/hA4lyLcz1dhAAqVFvc9C56f9yOa','WUfnLTuk5BBgo9e7eM7CjgpCI2vfDVU4uVwHCuYQid+FsjxiYp13wbbifzf9SBBEKBYhW9oo89VKrlF3l2efmxgzRFPQGrM9C9fMpHaa8cdU','XCLaSGnN1hBYy+G7Eauh/Bs2aw32bGs4YfzQReBYhZHC2yEDYcpimK2DdGT8BAIQbVo6I+Ba']

Solución

Se realizan varias combinaciones utilizando XOR entre las cuatro cadenas proporcionadas en la descripción, y se encuentra que la última cadena pareciera ser la clave al reto.

Asignamos a cada uno de los datos el valor:

```
data1 = R0f9LTuk4BAgo4T1YofCwH4Haz+aVAw4e6y+T7RCzJyF93IsLckjwf6tNnGzSFc...
```

```
data2 = NDSobmntpVUs7YT4YovCjn4EayKaTgxse7W+RbRezNOFsnItLcgjlf6xNn6zDFc...
data3 = WUfnLTuk5BBgo9e7eM7CjgpCI2vfDVU4uVwHCuYQid+FsjxiYp13wbbifzf9SBB...
data4 = XCLaSGnN1hBYy+G7Eauh/Bs2aw32bGs4YfzQReBYhZHC2yEDYcpimK2DdGT8BAIQbVo6I+Ba
```

El script adjunto es usado para realizar las siguientes operaciones, teniendo en cuenta lo siguiente:

```
data1 = data1_plaintext ^ key
data2 = data2_plaintext ^ key
data3 = data3_plaintext ^ key
data4 = data4_plaintext ^ key
var_1 = (data4_plaintext ^ key) ^ (data2_plaintext ^ key)
var_2 = (data4_plaintext ^ key) ^ (data3_plaintext ^ key)
```

El resultado de dichas operaciones es igual a:

```
var1 = data2_plaintext ^ data4_plaintext
var2 = data3_plaintext ^ data4_plaintext
```

Debido a que cada uno de los caracteres en el texto plano de data1, data2, y data3, están separados por espacios, se puede evidenciar parte del texto plano de data4 sin conocer la clave.

El siguiente script es usado para retornar el texto plano de data4 sin conocer la clave de cifrado:

```
<?php

$cipher =
array('R0f9LTuk4BAgo4T1YofCwH4Haz+aVAw4e6y+T7RCzJyF93IsLckjwf6tNnGzSFcXKFVjEo9tp5tK7RAyl2
fc3Vd6ABCVTvd0C5if6iOa',
'NDSobmntpVUs7YT4YovCjn4EayKaTgxse7W+RbRezNOFsnItLcgjlf6xNn6zDFcBKBZjFI9up9VK/hA4lyLcz1dh
AAqVFvc9C56f9yOa',
'WUfnLTuk5BBgo9e7eM7CjgpCI2vfDVU4uVwHCuYQid+FsjxiYp13wbbifzf9SBBEKBYhW9oo89VKrlF3l2efmxgz
RFPQGrM9C9fMpHaa8cdU',
'XCLaSGnN1hBYy+G7Eauh/Bs2aw32bGs4YfzQReBYhZHC2yEDYcpimK2DdGT8BAIQbVo6I+Ba');

function xor_encrypt($x, $y) {
    $out = '';
    for ($i = 0; $i < strlen($x); $i++) {
        $out .= chr(ord($x[$i]) ^ ord($y[$i % strlen($y)]));
     }

     echo $out . PHP_EOL;

     return $out;
}

$w = base64_decode($cipher[0]);
```

```
$x = base64_decode($cipher[1]);
$y = base64_decode($cipher[2]);
$z = base64_decode($cipher[3]);

$a = xor_encrypt($z, $x);
$b = xor_encrypt($z, $y);

for ($i = 0; $i < strlen($a); $i += 2) {
    echo $a[$i] . $b[$i + 1];
}

echo xor_encrypt($out, ' ');
```

La salida del anterior script es el siguiente:
HERE IS THE SECRET FLAG :ÇNothingIsAlwaysAbsolutelyXOR

Respuesta

NothingIsAlwaysAbsolutelyXOR

Reto 30

Descripción

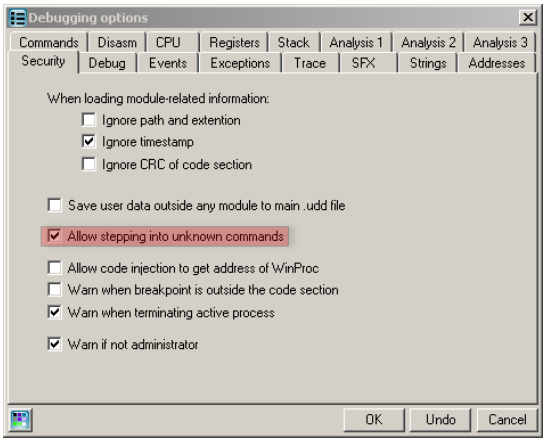
Exception Catcher in the Rye
Hey!, danos una mano y encuentra un serial valido para el usuario: “eko2013” Descargalo de: http://ctf.csiete.org/files/CatcherInTheRye.exe

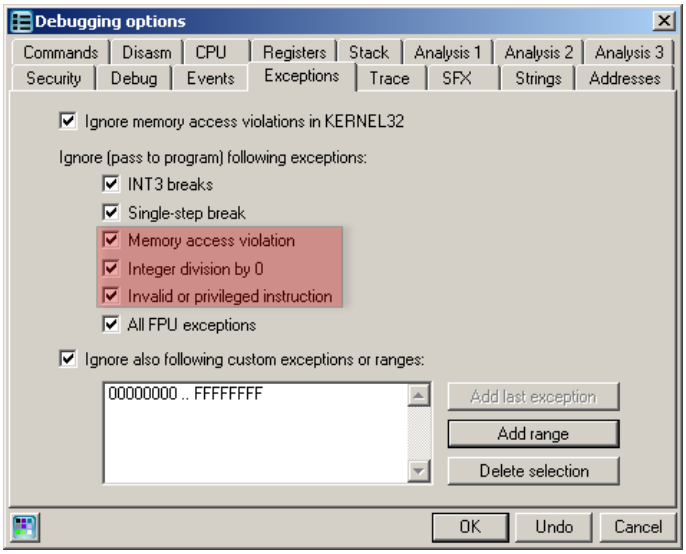
Solución

La respuesta a este reto no fue aceptada por el sistema, igualmente se realiza la respectiva resolución:

El reto consiste en un binario que utiliza controladores de excepción para ofuscar el código al ejecutar instrucciones de procesador que no son visibles desde el lugar de ocurrencia de la excepción.

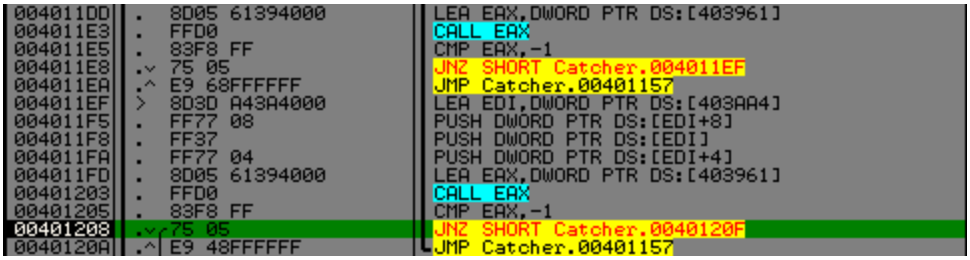
Para depurar tranquilamente habilitamos las siguientes opciones en Ollydbg:





Sin dar mucho detalle de como funciona la aplicación, estos son los puntos clave con los que puede obtener el serial del usuario:

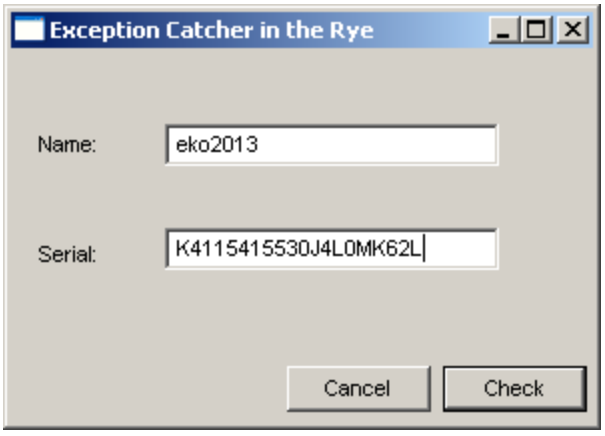
- bp 401208 (Técnica antidepuración): Se debe cambiar el valor de la bandera Z para evitar que la aplicación muera:



- bp 401BE2 (Comparación byte por byte del serial correcto con el ingresado por el usuario): En cada ciclo se debe editar la respuesta para coincidir con la esperada:

```
00401BD4  .  6A 00          PUSH 0
00401BD6  .  E8 63060000    CALL Catcher.0040223E
00401BDB  .  8B0485 B03A4000 MOV EAX,DWORD PTR DS:[EAX*4+403AB0]
00401BE2  .  381C30         CMP BYTE PTR DS:[EAX+ESI],BL
00401BE5  .v 74 05          JE SHORT Catcher.00401BEC
00401BE7  .v E9 AF000000    JMP Catcher.00401C9B
00401BEC  >  46             INC ESI
00401BED  .  47             INC EDI
00401BEE  .  49             DEC ECX
00401BEF  .  0BC9           OR ECX,ECX
00401BF1  .v 75 02          JNZ SHORT Catcher.00401BF5
00401BF3  .v EB 05          JMP SHORT Catcher.00401BFA
00401BF5  >^ E9 D7FEFFFF    JMP Catcher.00401AD1
00401BFA  >  68 B9394000    PUSH Catcher.004039B9
00401BFF  .  A0 B8394000    MOV AL,BYTE PTR DS:[4039B8]
00401C04  .  50             PUSH EAX
00401C05  .  A0 B7394000    MOV AL,BYTE PTR DS:[4039B7]
00401C0A  .  50             PUSH EAX
00401C0B  .  E8 AF030000    CALL Catcher.00401FBF
00401C10  .  50             PUSH EAX
00401C11  .  A1 FB3F4000    MOV EAX,DWORD PTR DS:[403FFB]
00401C16  .  FFD0           CALL EAX
```

Después de varias iteraciones se encuentra el serial adecuado para el usuario “eko2013”:



Se evidencia desde la depuración que el mensaje de serial correcto va a ser mostrado:

```
00401C60  .  50             PUSH EAX
00401C61  .  E8 59030000    CALL Catcher.00401FBF
00401C66  .  5B             POP EBX
00401C67  .  6A 00          PUSH 0
00401C69  .  68 AD394000    PUSH Catcher.004039AD                ASCII "Good Boy!"
00401C6E  .  A0 AC394000    MOV AL,BYTE PTR DS:[4039AC]
00401C73  .  50             PUSH EAX
00401C74  .  A0 AB394000    MOV AL,BYTE PTR DS:[4039AB]
00401C79  .  50             PUSH EAX
00401C7A  .  E8 40030000    CALL Catcher.00401FBF
00401C7F  .  50             PUSH EAX
00401C80  .  68 9F394000    PUSH Catcher.0040399F                ASCII "Nice shoot!"
00401C85  .  A0 9E394000    MOV AL,BYTE PTR DS:[40399E]
00401C8A  .  50             PUSH EAX
00401C8B  .  A0 9D394000    MOV AL,BYTE PTR DS:[40399D]
00401C90  .  50             PUSH EAX
00401C91  .  E8 29030000    CALL Catcher.00401FBF
00401C96  .  50             PUSH EAX
00401C97  .  6A 00          PUSH 0
00401C99  .  FFD3           CALL EBX
00401C9B  >  C9             LEAVE
00401C9C  .  C2 0800        RETN 8
```

Respuesta

K4115415530J4L0MK62L