



17/03/2005



<http://www.xonico.com.ar>  
xonico@gmail.com  
by xonico

Copyright © 2003-2005 xonico. Todos los derechos reservados.

# I Torneo - Hacking ético - CrashCool.com

## Nivel 1

Al empezar el torneo damos con el siguiente texto:

"A este nivel solo podras acceder a través de un Nokia 9210"

Los navegadores poseen un identificador (*User-Agent*) que los diferencia unos de otros, como así también idioma (*Accept-Language*) entre otras cosas mas en las cabeceras de petición (ver "*protocolo http*" <http://www.lfcia.org/openprojects/camllets/doc/html/node19.html> ), al realizar una petición si nuestro navegador es un Mozilla nuestra petición será del tipo User-Agent: Mozilla/4.0

Para cambiar nuestro *User-Agent*: hay muchas herramientas y hasta extensiones para Firefox que lo hacen, pero no entrare en detalles, si las buscan las encontrarán, utilizaremos un proxy man in the middle (hombre en el medio), para que nos vallamos familiarizando por que nos servira a lo largo de todo el torneo, yo personalmente les recomiendo Burp Proxy (<http://www.portswigger.net/proxy/>), también pueden utilizar el Achilles (<http://www.mavensecurity.com/achilles/>), hay muchos de este tipo, generalmente más complejos que se los utilizan para pen-testing. Los Proxys sirven para modificar y ver las peticiones y respuesta que enviamos y recibimos del servidor web.

### browser <=> proxy <=> server

Este tipo de proxy se utiliza generalmente para saltar protecciones javascript, cambiar cabeceras http, cambiar código http, entre otras cosas...

Bueno lo que debemos hacer es poner nuestro proxy e interceptar nuestras peticiones para eso activamos las conexiones proxys de nuestro navegador, ponemos el proxy local (host: localhost) con su puerto, el burp proxy por defecto es el 8080.

Ya estamos listos para interceptar, cuando nuestro navegador haga la solicitud cambiamos nuestro identificador por *User-Agent: Nokia 9210*

Veremos que nos aparece un código fuente Java:

```
/*Se ha utilizado el siguiente algoritmo
 *para codificar un numero, obteniendo
 *este resultado: 'Q8)367219
 *Encuentra dicho número.
 */
public static String codifica(int numero)
{
    .
    .
    .
}
```

A esta etapa se la puede encarar de varias formas:

- Analizar el código fuente y deducir que es lo que hace el programa, para eso tenemos que tener conocimientos sobre Java
- Ver para determinadas entradas que salida obtenemos hasta deducir como funciona el programa.
- Codificar un bucle que valla probando (bruteforce) hasta dar con el número correcto, no se necesita muchos conocimientos de Java para hacer esto.

Opte por la ultima opción, el código quedaría así:

codigo.java

```
public class codigo {
    public static void main(String[] args) {
        int i=0;
        while(!codifica(i).equals("Q8)367219"))
            i++;
        System.out.println("Resultado = " + i);
    }

    public static String codifica(int numero)
    {
        String codigo;
        char letras[];
        int i,j;
        for(i=0;i<(numero);i++){
            if(i%2==0) numero+=(i+1);
            else i+=(numero*2);
        }
        codigo=numero+":P";
        letras=codigo.toCharArray();
        for(i=0;i<letras.length;i++)
        {
            j=(int)(letras[i]);
            if(i%2==0) j=j+1;
            else j=j-2;
            letras[i]=(char)(j);
        }
        codigo="";
        for(i=(letras.length-1);i>=0;i--)
            codigo=codigo + letras[i] + "";

        return codigo;
    }
}

//compilar: javac codigo.java
//ejecutar: java codigo
```

Si no tenemos un compilador a mano podemos utilizar uno online  
[http://www.innovation.ch/java/java\\_compile.html](http://www.innovation.ch/java/java_compile.html)

Ejecutamos el programa, esperamos un momento y obtendremos la solución = **831954**

## Nivel 2

Damos con el siguiente mensaje:

“Tatiana, una linuxera un poco despistada ha perdido su contraseña y está desesperada ya que no puede acceder a su sistema y le da escalofríos usar windows. Ayuda a tatiana a recuperar su contraseña; me ha entregado un chorro de caracteres ....., ella dice que es un archivo punto tar o algo así pero yo no sé ni qué es eso. “

Al abrir el txt de tatiana nos damos con una tira de caracteres ascii, a simple vista nos podemos dar cuenta que es una codificación uuencode, sirve para codificar y decodificar archivos binarios a archivos de textos, esta codificación es utilizada generalmente para mandar binarios por correo.

Decodificamos

```
Uudecode nombre.txt nombre.tar
```

Todavía tenemos que hacer unos cambios para que funcione cambiamos el nombre `_tatiana_.txt` por `tatiana.txt` y nos falta definir el comienzo y fin del archivo uuencode, editamos `tatiana.txt`

Agregamos una línea al principio

```
begin 644 tatiana.tar
```

y otra al final

```
end
```

Decodificamos: ***Uudecode tatiana.txt tatiana.tar***

Obtenemos un archivo comprimido que contiene el shadow y passwd de un sistema GNU/Linux

Ahora nos toca desencriptar, para eso utilizaremos el “John the Ripper“

(<http://www.openwall.com/john/> )

Primero debemos combinar el shadow con el passwd para eso John trae una utilidad extra *unshadow entonces ejecutamos*

```
unshadow passwd shadow > tatiana
```

Estamos listos para desencriptar. Para desencriptar se puede seguir de la siguiente manera:

- probar con solo dígitos ( `john -i:Digits tatiana` )
- probar con solo alfas ( `john -i:Alpha tatiana` )
- probar con un diccionario ( `john -w:dicc tatiana` )
- por último el modo infinito (todos los caracteres) ( `john -i:All tatiana` )

Al desencriptar con solo caracteres alfas en pocos minutos tendremos la contraseña:

```
owwo
```

### Nivel 3

Nos aparece un formulario para ingresar... solo nos deja ingresar 3 caracteres, vamos al código fuente para ver si es una restricción javascript o simplemente html, viendo el form nos damos con `maxlength="3"`

```
.
:
.

<form name="form1" method="post" action="index.php">

    <div align="center">
        <input name="n3xj3" type="text" id="retonivel33" size="10" maxlength="3"
CLASS="textBox">
    &nbsp;
    <input type="submit" name="Submit" value="Enviar" class="boton">
    </div>
</form>

.
:
.
```

Para saltar este tipo de protecciones podemos guardar la pagina en nuestro disco y ejecutarla en nuestro navegador, pero antes debemos cambiar el `<form>` donde dice `action="index.php"` por `action="http://crashcool.com/torneo04/index.php"` y borrar el `maxlength="3"`

Yo opte por interceptar las respuestas del servidor con el proxy y borrar el `maxlength="3"`

Ahora ya podemos probar con lo que se nos ocurra, al poner una inyección sql `' or '='` nos tira el msg: "Aun no has pasado la prueba.... :P"

Interceptamos nuestra petición al server cuando mandamos `' or '='` (para mas info <http://www.willydev.net/descargas/SQLInjection.pdf> )

```
POST /torneo04/index.php HTTP/1.0
Accept: image/gif, image/x-bitmap, image/jpeg, image/pjpeg, ...
Referer: http://www.crashcool.com/torneo04/index.php
Accept-Language: es
Content-Type: application/x-www-form-urlencoded
Proxy-Connection: Keep-Alive
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows 98)
Host: www.crashcool.com
Pragma: no-cache
Cookie:
_crashcool=eG9uaWNvIGM2YjFjYWY4ZDU0NGY4YWZhMGExMmI5MTU1Y2Q4MGE5IGJj
ZjZkODI4YjQ0MTM2ZmExZWZmMDI1YmM5ZDZmMDRmIE1nPTI=;
_crash_acceder=c2k%3D;
Content-Length: 41

n3xj3=%27+or+%27%27%3D%27&Submit=Enviar
```

Vamos analizar las cookies

Cookie:

```
_crashcool=eG9uaWNvIGM2YjFjYWY4ZDU0NGY4YWZzMGE5IGJjZjZkODI4YjQ0MTM2ZmExZWZWM2MDI1YmM5ZDZmMDRmIE1nPTI=;
_crash_acceder=c2k%3D;
```

Al ver las cookies nos damos cuentas que están codificadas con **base 64**, para decodificar podemos utilizar este script <http://www.xonico.com.ar/webtools/sneak.php>  
La cookie cuenta de dos partes `_crashcool` y `_crash_acceder` en la primera parte observamos:

xonico c6b1caf8d544f8afa0a12b9155cd80a9 bcf6d828b44136fa1ec6025bc9d2f04f Mg=2 nada del otro mundo, nuestro user, nuestro pass codificado en md5 y los otros datos son un tipo de control para el nivel

Ahora vamos a la segunda parte de la cookie (`_crash_acceder=c2k%3D;`) decodificamos con base 64 y nos da esto `_crash_acceder=si;`

Que pasa si modificamos nuestra cookie por `_crash_acceder=no;` para eso tenemos que codificar **no** en base64 entonces quedaría `_crash_acceder=bm8=;`  
El servidor nos responde con otro mensaje “?¿?¿?¿”

Todavía seguimos sin pasar el nivel, veamos el código html con el que nos responde el servidor, si nos fijamos bien, antes del msg “?¿?¿?¿” hay un comentario:

**<!-- Navegador:**

**Vm0xMFIXRnJOVmhXYkdoWFItEGFjRIV3VIRGalJsWnhVV3hhVUZWVU1Eaz0= >**

Por lo visto también es base 64, decodificamos:

```
Vm0xMFIXRnJOVmhXYkdoWFItEGFjRIV3VIRGalJsWnhVV3hhVUZWVU1Eaz0=
```

Sigue teniendo forma de base64 volvemos a decodificar

```
Vm10YWFrNVhWbGhXYmxacFUwVTFjRIZxUWxaUFVUMDk=
```

de nuevo `Vmtaak5XVihWblZpU0U1cFVqQIZPUT09`

de nuevo `VkZjNWVXVnViSE5pUjBVOQ==`

de nuevo `VFc5eWVubHNIr0U9`

de nuevo `TW9yenlsbGE=`

nos da algo lógico

**Morzylla**

Navegador: Morzylla... suena a *User-Agent: Morzylla*

Entonces nuestra petición quedaría

```
POST /torneo04/index.php HTTP/1.0
. . .
User-Agent: Morzylla
. . .
_crashcool=eG9uaWNvIGM2YjFjYWY4ZDU0NGY4YWZzMGE5IGJjZjZkODI4YjQ0MTM2ZmExZWZWM2MDI1YmM5ZDZmMDRmIE1nPTI=;_crash_acceder=bm8=;
. . .
n3xj3=%27+or+%27%27%3D%27&Submit=Enviar
```

Nos aparece otro msg *"Warning: Failed opening 'retos/level3.passwd' for inclusion (include\_path=)"* "

Veamos que hay en el path que hace referencia (*retos/level3.passwd*), como estamos en <http://www.crashcool.com/torneo04/> la dirección sería <http://www.crashcool.com/torneo04/retos/level3.passwd>

**9745DFC0D5CCAA1F9A6AB0A8296A022D**

El archivo contiene un hash de 128 bits (32 caracteres) hay varios algoritmos que dan un hash de 128 bits, uno de los mas conocidos es la encriptación **MD5** no es un algoritmo simétrico como base 64 así que hay que crackearlo como lo hicimos con el shadow de tatiana y el "jhon the ripper", para descifrar el hash MD5 utilizaremos el "mdcrack" (<http://mdcrack.multimania.com/>)

*mdcrack -s [caracteres] [hash]*

Probamos de la misma manera que lo hicimos anteriormente, comencemos por los dígitos

*mdcrack -s 0123456789 9745DFC0D5CCAA1F9A6AB0A8296A022D*

en cosa de segundos bingo!

*Resultado = **64261***

#### Nivel 4

Aparece un mensaje que dice:

*“Only http://www.eG9uaWNv.com/xonico.php referer is allowed”*

Este tipo de falla es muy simple, solo tenemos que modificar la petición *referer*: por la que nos dice que es permitida

```
POST /torneo04/index.php HTTP/1.0
.
.
Referer: http://www.eG9uaWNv.com/xonico.php
.
.
```

Para mas información ver <http://www.datatrendsoftware.com/spoof.html> también hay extensiones para firefox que hacen el spoof referer como <http://refspoof.mozdev.org/>

Saltamos la restricción y nos sale otro msg:

*“<applet code= "Imagen.class" codebase="none" archive="null" width="100" height="100"> Eufrasio es un poco descuidado con el php y no ha insertado bien este sencillo applet que te mostraba una imagen gif... pero bueno no creo q eso te frene. ¿Que solución te ha dado el gif?:”*

Como no podemos ejecutar el applet en el server, lo que nos queda es descargarlo para analizarlo.

El atributo *“archive”* (*null*) nos da el nombre del archivo o archivos empaquetado con los *.class*, *“codebase”* (<http://www.crashcool.com/torneo04/none>) nos dice donde se encuentra la url base del applet.

Entonces el archivo comprimido se encuentra en <http://www.crashcool.com/torneo04/none/null>

Lo descargamos, descomprimos, contiene 3 *.class* el que nos interesa es *Imagen.class*, como es un binario tenemos que decompilarlo para ver el código fuente, podemos utilizar el *“DJ Java Decompiler”* (<http://members.fortunecity.com/neshkov/dj.html>)

Vamos a analizar un poco el código que obtuvimos al decompilar *Imagen.class*, veamos las partes que nos interesan del código fuente.

#### Imagen.java

```
String s = "";
int ai[] = {101, 98, 87, 103, 105, 61, 67};
for(int i = 0; i < ai.length; i++)
    s = s + (char)ai[i]; //va concatenando (char)101 (char)98... en 's'
//((char)num nos da el valor ascii del num
s = base64.Convertir_64(s); //codifica "s" en base64
s = MD5.getMD5(s); //ahora en MD5
String s1 = s.substring(0, 16); //toma la primera mitad de caracteres de 's'
String s2 = s.substring(16, 32); //toma la segunda mitad de caracteres de 's'
s = s2 + s1; //las invierte
img = getImage(getDocumentBase(), s + ".gif"); //guarda la imagen 's'.gif donde 's' es un hash
```

Lo que necesitamos nosotros es saber el valor de 's' que será la imagen que necesitamos, nuestro código modificado quedara así

Imagen.java

```
public class Imagen
{
    public static void main(String[] args)
    {
        String s = "";
        int ai[] = {101, 98, 87, 103, 105, 61, 67};

        for(int i = 0; i < ai.length; i++)
            s = s + (char)ai[i];

        s = base64.Convertir_64(s);
        s = MD5.getMD5(s);
        String s1 = s.substring(0, 16);
        String s2 = s.substring(16, 32);
        s = s2 + s1;

        System.out.println("http://www.crashccol.com/torneo04/" + s + ".gif");
    }
}
//javac Imagen.java
//java Imagen
```

Nos dara como resultado la ubicación de la imagen que es

**<http://www.crashcool.com/torneo04/56004ca1e9de8093dc1b3741e4a3035b.gif>**

Al descargar el gif lo único que vemos es la emblema hacker o uno de los símbolos del juego de la vida.

Editemos el gif para ver si encontramos algo oculto, yo lo hice con "Microsoft GIF Animator" ([http://www.jhepple.com/gif\\_animator.htm](http://www.jhepple.com/gif_animator.htm))

Observamos que cada frame tiene comentarios

frame #1

UGFyYSBvYnRlbnVyaGxhIHhBhc3N3b3JkIGRIYmVyYXMGZGVjb2Rpb2ZmljYXJsYSBjb24gc3UgbGxhdmUu

frame #2

VjJ0a1YyRlhSbGhTYmxVOQ==

frame #3

VG9kbyBlc3RhIGFxdWk=

frame #4

022637AD86B3BED8B8A3648CC280264E0444BD7854B8C3B007A31447235280D8

A esta altura supongo que ya reconocen una codificación base64, los frame 1,2,3 están codificados de esa manera, el 4to frame por el momento tiene forma hexadecimal

Veamos que nos da como resultado los primeros 3 frames

frame #1

*Para obtener la password deberas decodificarla con su llave.*

frame #2

V2tkV2FXRlhSblU9

WkdWaWFXRnU=

ZGViaWFu

*debian*

frame #3

*Todo esta aquí*

El primer frame nos dice que es un password con llave (key), la llave se supone que es "debian"... cifrados con key hay muchos los mas conocidos son DES, Triple-DES, CAST, IDEA, Blowfish, AES, también sabemos que a la vez esta codificado en hexadecimal.

Para descifrar el password nos vamos a

<http://www.instisec.com/publico/descargas/criptoaspdemo.asp?id=1>

Ponemos como clave "debian" el texto cifrado y marcamos tipo de codificación hex

Texto descifrado:

**apt-cache search solucion**

## Nivel 5

Llegamos al final y nos dice:

*\* Tatiana es la única persona del planeta que posee un novedoso servidor web, el super-ehcapA y aprovechando esto Eufrasio solo tiene que indicarle al sistema la dirección del server de Tatiana para poder leer el mensaje., como dato decir que Eufrasio accede al sistema a través de su servidor*

*\* Me han llegado unos rumores: Tatiana y Eufrasio tienen pensado hackear la nasa, y por lo que parece ya han conseguido sacar información sobre el caso O.V.N.I de Perú.*

*\* Intenta acceder a su sistema de mensajes y dime el número de informe que han sustraído, yo tan solo he conseguido la ip del servidor de Eufrasio 10.11.12.13*

Nos aparece un form para poner ip/host

Si ponemos localhost nos responde *"no creo que tatiana se halla conectado con: Apache"*

Si ponemos otro ip de cualquier server nos aparece *"no creo que tatiana se halla conectado con: Apache 2.0.7"*

Eso nos da a entender que el script lo que hace es fijarse la versión del server, eso nos da una idea, montar nuestro servidor web con la identificación de tatiana *"super-ehcapA"*

Para eso no necesitamos montar todo un apache y falsificar la versión del server, haremos algo más simple, abriremos el puerto 80 y mandaremos nuestra versión de apache server: *super-ehcapA*

Utilizaremos el nc (netcat) para montar un apache falso (<http://www.securityfocus.com/tools/139/scoreit>)

```
nc -l -p 80 < respuesta.txt
```

Si queremos que el puerto quede abierto utilizamos `-L` en vez de `-l`, el archivo `respuesta.txt` contendrá lo que mandara al recibir una petición en este caso del script

respuesta.txt

```
HTTP/1.1 200 OK
Date: Sat, 26 Feb 2005 08:39:56 GMT
Server: super-ehcapAa
X-Powered-By: PHP/4.3.10
Connection: close
Content-Type: text/html

<HTML><HEAD>
<TITLE>Spoof Apache</TITLE>
</HEAD><BODY>
Te gusta mi super-ehcapA =)?
</BODY>
```

Ahora ponemos nuestro ip o host (<http://www.showmyip.com/>) para que haga la consulta a nuestro apache falso

Nos dice: *"Tu no eres Eufrasio"*

Será por que no estamos a haciéndolo desde el host de eufrasio?... para eso al mandar nuestro ip interceptamos nuestra petición con el proxy y cambiamos el referer por el host de eufrasio

**Referer: http://10.11.12.13/**

Pasamos tatiana nos envía el siguiente msg:

Tatiana@martel:~\$telnet df4t6.nasa.gov<br>

```
.  
. .  
root@df4t6.nasa.gov:/home/research/$ cat peru-00254  
77 112 116 33 112 119 111 106 116 33 101 102 33 113 102 115 118 33 102 115 98 111  
33 53 33 110 98 115 100 106 98 111 112 116 33 113 115 112 100 102 101 102 111  
117 102 116 33 101 102 109 33 113 109 98 111 102 117 98 33 115 98 117 106 100 118  
109 106 111 45 33 101 102 116 113 118 102 116 33 101 102 33 101 98 115 116 102  
33 118 111 98 33 119 118 102 109 117 98 33 113 112 115 33 102 109 33 110 118 111  
101 112 33 119 106 111 106 102 115 112 111 33 98 114 118 106 33 122 33 111 112  
116 33 117 112 110 98 110 112 116 33 118 111 33 100 98 103 102 45 33 102 115 98  
111 33 104 102 111 117 102 33 110 98 107 98 47 47 47 71 74 79 33 69 70 77 33 74  
79 71 80 83 78 70 33 79 86 78 70 83 80 59 33 80 87 50 52 56 66  
root@df4t6.nasa.gov:/home/research/$exit
```

Nos pide el numero de informe:

Si nos fijamos no hay ningún numero menor a 33 que son los caracteres ascii de control, tampoco ningún numero mayor a 128 que son los ascii extendidos, esto nos da a entender que los números son el valor decimal de los ascii.

Para pasar los valores decimales a ascii lo podemos hacer desde

[http://www.xonico.com.ar/webtools/ascii\\_conv.php](http://www.xonico.com.ar/webtools/ascii_conv.php)

Ponemos nuestra tira de números en el cuadro de decimal ascii y en delimiter ponemos el espacio (space) en vez de la “,” calculamos y nos da la cadena

```
Mpt!pwojt!ef!qfsv!fsbo!5!nbsdjbopt!qspdfouft!efm!qmbofub!sbujdvmo-  
!eftqvft!ef!ebstf!vob!wvfmub!qps!fm!nvoep!wjojfs!brvj!z!opt!upnbnpt!vo!dbgf-  
!fsbo!hfouf!nbkb///GJO!EFM!JOGPSNF!OVNFSP;!PW248B
```

Este es otro cifrado conocido el **Caesar** (mas info <http://es.tldp.org/Manuales-LuCAS/SEGUNIX/unixsec-2.1-html/node312.html>)

Desciframos (Caesar Bruteforce <http://www.xonico.com.ar/webtools/sneak.php>) y nos da

```
Los!ovnis!de!peru!eran!5!marcianos!procedentes!de!planeta!raticulin-  
!despues!de!darse!una!vuelta!por!el!mundo!vinieron!aqui!y!nos!tomamos!un!cafe-  
!eran!gente!maja///FIN!DEL!INFORME!NUMERO;!OV248A
```

Pareciera que el numero de informe es OV248A, pero si nos fijamos el cifrado caesar solo es para caracteres alfabeticos (a-z) los caracteres que no sean no son modificados por eso los signos de admiración “!” y los números “248” no son cifrados

Lo que haremos es buscar el valor 2, 4, 8 en la tabla que utiliza el sistema caesar para cifrar y descifrar

A=0	B=1	C=2	D=3	E=4	F=5
G=6	H=7	I=8	J=9	K=10	L=11
M=12	N=13	O=14	P=15	Q=16	R=17
S=18	T=19	U=20	V=21	W=22	X=23
Y=24	Z=25				

El 2=C el 4=E y el 8=I

Ciframos en caesar "CEI" nos da "BDH" reemplazamos los valores en numero 137  
Entonces nuestro numero de Informe quedara

**OV137A**



--

< xonico@gmail.com >

<http://www.xonico.com.ar>

Security Research & Development

PGPkey 0x564034EC Available at KeyServ

BEF5 1795 BFCC DB2C 0BEF 92BD 0162 885F 5640 34EC