UAM SILICON VALLEY – EPISODIO 2

Dinesh ha perdido la clave VERDADERA que usaba para abrir su zip secreto pero gracias a DIOS tiene un archivo .raw donde puede recuperarla y necesita que le echemos una mano.

A Dinesh le encantan los mensajes con doble sentido, debéis tenerlo en cuenta...

- Archivo .raw (escoged el que mejor os venga):

https://www.mediafire.com/file/piv4t8514bp5dpg/pied_piper_bak.zip/file

https://mega.nz/#!iAUDnKwA!Y2g23qnZ9rwZvzZA3Bg8cbENe_ZtASOi1NFgrgfL8sg

Info: Las pistas os servirán a partir de que tengáis la contraseña del zip adjunto (Secretos_Dinesh.zip). Recordad que flag.txt tiene dos cifrados (leed bien README).

Descargamos el fichero de uno de los enlaces. Vemos el contenido del fichero, que además de ser muy grande (1 GB), nos muestra que es una captura de memoria de una máquina virtual:

```
nacho@kali:~/Forensic$ hexdump -C pied_piper_bak.raw | head -10
00000000  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
00001000  00 c0 d0 ff ff ff ff ff  00 30 d0 ff ff ff ff ff  |.........0......|
00001010  f0 00 7f 41 00 00 00 00  46 41 43 50 f4 00 00 00  |...A....FACP....|
00001020  04 f8 56 42 4f 58 20 20  56 42 4f 58 46 41 43 50  |..VBOX  VBOXFACP|
00001030  01 00 00 00 41 53 4c 20  61 00 00 00 00 02 7f 41  |....ASL a......A|
00001040  70 04 7f 41 00 00 09 00  2e 44 00 00 a1 a0 00 00  |p..A.....D......|
00001050  00 40 00 00 00 00 00 00  04 40 00 00 00 00 00 00  |.@.......@......|
00001060  00 00 00 00 08 40 00 00  20 40 00 00 00 00 00 00  |.....@.. @......|
00001070  04 02 00 04 02 00 00 00  65 00 e9 03 00 00 00 00  |........e.......|
```

Nos lo llevamos a la máquina Kali y analizamos la imagen con volatility. Nos sugiere distintos profiles de Windows:

```
nacho@kali:~/Forensic$ volatility -f pied_piper_bak.raw imageinfo
Volatility Foundation Volatility Framework 2.6
INFO    : volatility.debug    : Determining profile based on KDBG search...
          Suggested Profile(s) : Win7SP1x64, Win7SP0x64, Win2008R2SP0x64, Win2008R2SP1x64_24000, Win2008R2SP1
x64_23418, Win2008R2SP1x64, Win7SP1x64_24000, Win7SP1x64_23418
                     AS Layer1 : WindowsAMD64PagedMemory (Kernel AS)
                     AS Layer2 : FileAddressSpace (/home/nacho/Forensic/pied_piper_bak.raw)
                      PAE type : No PAE
                           DTB : 0x187000L
                          KDBG : 0xf80002a520a0L
          Number of Processors : 1
     Image Type (Service Pack) : 1
                KPCR for CPU 0 : 0xfffff80002a53d00L
             KUSER_SHARED_DATA : 0xfffff78000000000L
           Image date and time : 2018-10-15 10:48:27 UTC+0000
     Image local date and time : 2018-10-15 12:48:27 +0200
```

Ahora consultamos el historial de comandos de la máquina (con cmdscan), por si nos da una pista de lo que se hizo antes de capturar la memoria:

```
nacho@kali:~/Forensic$ volatility -f pied_piper_bak.raw cmdscan --profile=Win7SP1x64
Volatility Foundation Volatility Framework 2.6
**************************************************
CommandProcess: conhost.exe Pid: 2200
CommandHistory: 0x1846d0 Application: cmd.exe Flags: Allocated, Reset
CommandCount: 1 LastAdded: 0 LastDisplayed: 0
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x5c
Cmd #0 @ 0x17e310: ipconfig
Cmd #15 @ 0x140158:
Cmd #16 @ 0x1839d0:
**************************************************
CommandProcess: conhost.exe Pid: 824
CommandHistory: 0x2f45d0 Application: DumpIt.exe Flags: Allocated
CommandCount: 0 LastAdded: -1 LastDisplayed: -1
FirstCommand: 0 CommandCountMax: 50
ProcessHandle: 0x5c
Cmd #15 @ 0x2b0158: /
Cmd #16 @ 0x2f38d0: /
```

Se ve que abrió un cmd, y dentro haría cosas que nos pueden dar una pista. Consultamos esos comandos con cmdline:

```
nacho@kali:~/Forensic$ volatility -f pied_piper_bak.raw cmdline --profile=Win7SP1x64
Volatility Foundation Volatility Framework 2.6
**************************************************************************
```

```
DB Browser for pid:   1836
Command line : "C:\Program Files\DB Browser for SQLite\DB Browser for SQLite.exe"
**************************************************************************
notepad.exe pid:   2616
Command line : "C:\Windows\system32\NOTEPAD.EXE" C:\Users\Richard\Desktop\piper.txt
**************************************************************************
```

Vemos dos cosas interesantes, que ha instalado un SQLite en la máquina, y que ha trasteado un fichero “piper.txt” con el Notepad. Ahora vamos a sacar el listado de ficheros de la máquina, a ver qué vemos:

```
0x0000000040501860     16      0 R--rw- \Device\HarddiskVolume2\Users\Richard\Desktop\piperdb.db
0x00000000406ac310      1      1 R--rw- \Device\HarddiskVolume2\Users\Richard\Desktop
0x000000004123a8d0      1      1 RW-rw- \Device\HarddiskVolume2\Users\Richard\Desktop\piperdb.db
0x0000000041266240     16      0 R--rwd \Device\HarddiskVolume2\Windows\Web\Wallpaper\Nature\Desktop.in
0x00000000412679a0     16      0 R--rwd \Device\HarddiskVolume2\Windows\Web\Wallpaper\Scenes\Desktop.in
0x00000000412683a0     16      0 R--rwd \Device\HarddiskVolume2\Windows\Web\Wallpaper\Landscapes\Deskto
0x00000000412699a0     16      0 R--rwd \Device\HarddiskVolume2\Windows\Web\Wallpaper\Characters\Deskto
0x000000004126af20     16      0 R--rwd \Device\HarddiskVolume2\Windows\Web\Wallpaper\Architecture\Desk
0x0000000041379070     16      0 R--rwd \Device\HarddiskVolume2\Users\Richard\AppData\Roaming\Microsoft
0x000000004146c660     16      0 R--rwd \Device\HarddiskVolume2\Users\Richard\Desktop\piper.txt
```

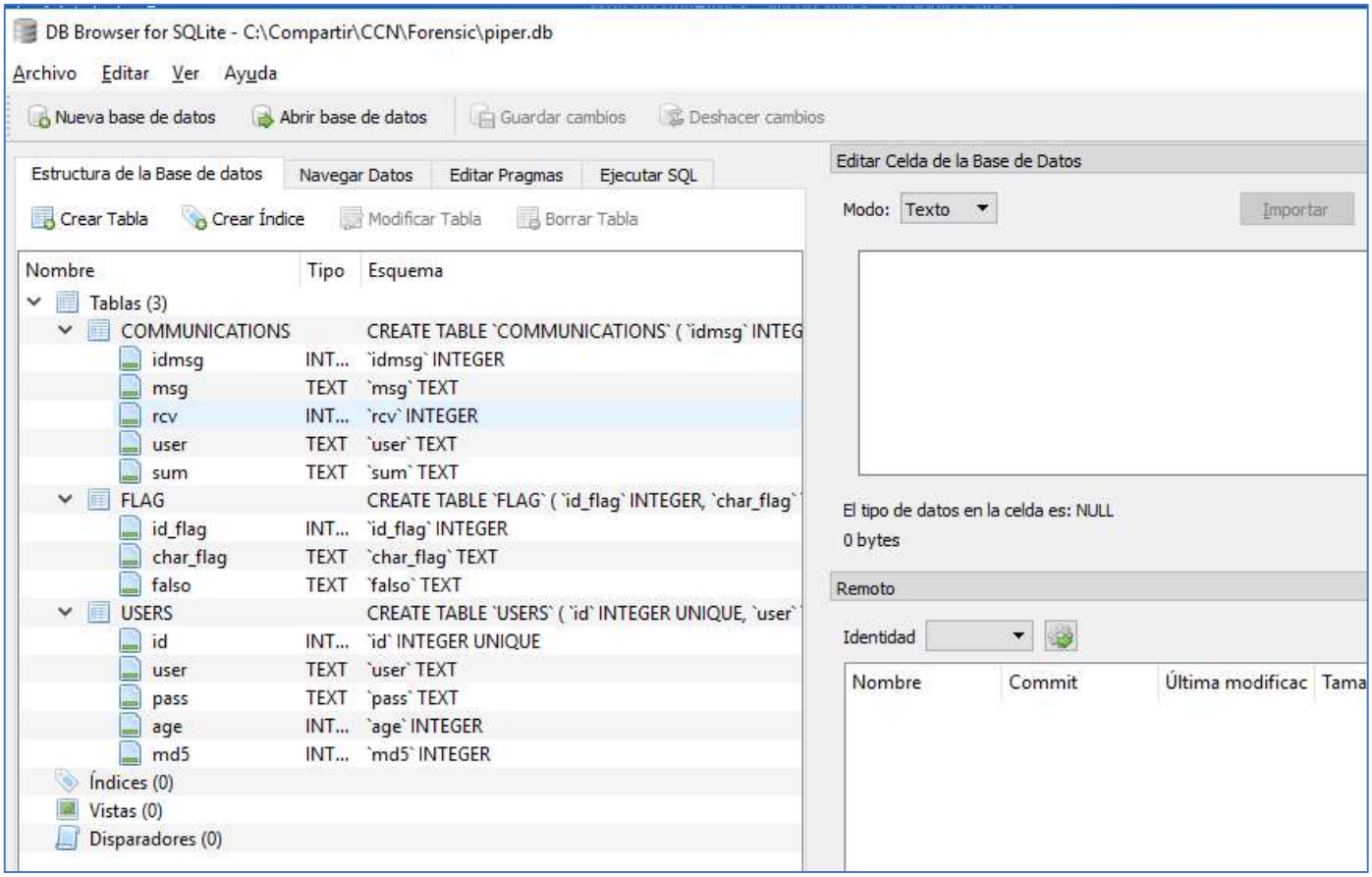
Salen dos ficheros sospechosos:

- Fichero piper.txt, que además vimos que lo habían trasteado con el Notepad. Lo intento volcar a disco a ver qué tiene, pero no recupera nada. Posiblemente tenga tamaño vacío, o alguna protección y no es capaz de recuperarlo.

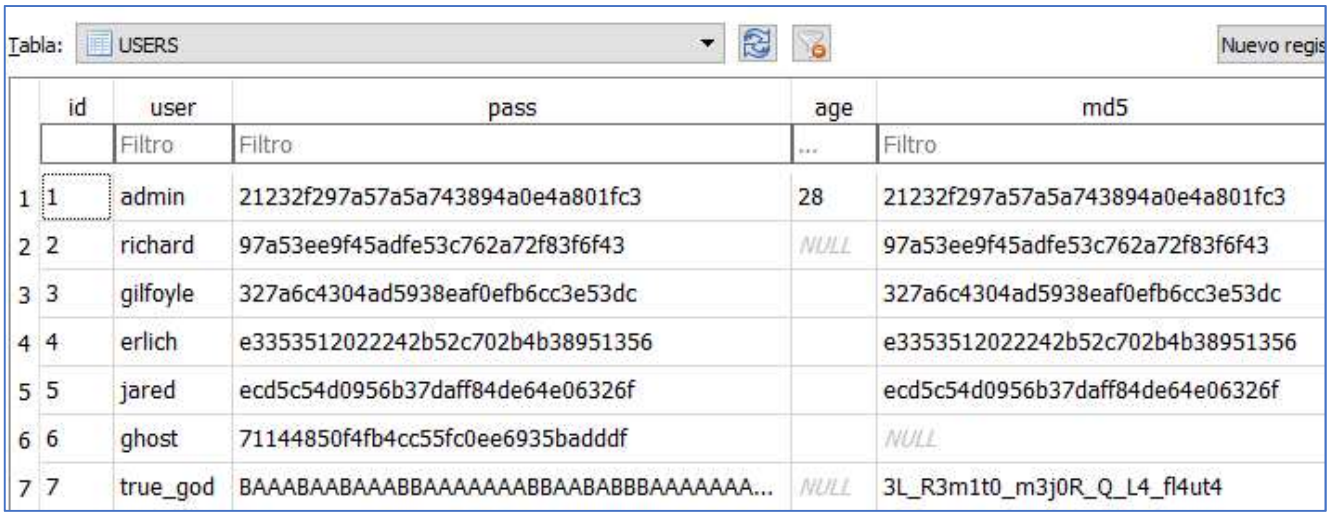
- Fichero piperdb.db, lo recupero con volatility y esta vez sí tenemos éxito:

```
nacho@kali:~/Forensic$ volatility -f pied_piper_bak.raw --profile=Win7SP1x64 dumpfiles -Q 0x000000004123a8d0
--dump-dir=files
Volatility Foundation Volatility Framework 2.6
DataSectionObject 0x4123a8d0   None   \Device\HarddiskVolume2\Users\Richard\Desktop\piperdb.db
SharedCacheMap 0x4123a8d0   None   \Device\HarddiskVolume2\Users\Richard\Desktop\piperdb.db
```

Como es un fichero “.db”, y vimos que instalaron un SQLite, vamos a probar a abrirlo a ver qué vemos. Nos lo llevamos a Windows y lo abrimos:



Tiene dentro tres tablas, vamos a ver qué datos contiene. Las tablas de FLAG y de COMMUNICATIONS están vacías, pero la de USERS no:



	id	user	pass	age	md5
		Filtro	Filtro	...	Filtro
1	1	admin	21232f297a57a5a743894a0e4a801fc3	28	21232f297a57a5a743894a0e4a801fc3
2	2	richard	97a53ee9f45adfe53c762a72f83f6f43	NULL	97a53ee9f45adfe53c762a72f83f6f43
3	3	gilfoyle	327a6c4304ad5938eaf0efb6cc3e53dc		327a6c4304ad5938eaf0efb6cc3e53dc
4	4	erlich	e3353512022242b52c702b4b38951356		e3353512022242b52c702b4b38951356
5	5	jared	ecd5c54d0956b37daff84de64e06326f		ecd5c54d0956b37daff84de64e06326f
6	6	ghost	71144850f4fb4cc55fc0ee6935badddf		NULL
7	7	true_god	BAAABAABAAABBAAAAAAABBAABABBBAAAAAAA...	NULL	3L_R3m1t0_m3j0R_Q_L4_fl4ut4

Contiene varios usuarios, y los password están la mayoría en MD5. Nos vamos a la web https://crackstation.net/ y vamos probando a sacar las cadenas originales, la mayoría si que las encontramos, pero vamos probando si es la contraseña del ZIP pero no lo son. Probamos también esa cadena del "R3m1t0_m3j0R", pero tampoco.

Vemos que una de las password es una cadena con "A"s y "B"s. Esto es una codificación binaria, en vez de 1 y 0, con A y B. Existe una codificación llamada Bacon's Cipher que basa su alfabeto cambiando los 1 y 0 por los caracteres A y B. En internet sacamos el alfabeto y vamos traduciendo:

https://en.wikipedia.org/wiki/Bacon's_cipher

Letter	Code	Binary	Letter	Code	Binary
A	aaaaa	00000	N	abbab	01101
B	aaaab	00001	O	abbba	01110
C	aaaba	00010	P	abbbb	01111
D	aaabb	00011	Q	baaaa	10000
E	aabaa	00100	R	baaab	10001
F	aabab	00101	S	baaba	10010
G	aabba	00110	T	baabb	10011
H	aabbb	00111	U	babaa	10100
I	abaaa	01000	V	babab	10101
J	abaab	01001	W	babba	10110
K	ababa	01010	X	babbb	10111
L	ababb	01011	Y	bbaaa	11000
M	abbaa	01100	Z	bbaab	11001

Letra a letra nos saca la cadena final:

```
BAAAB  R
AABAA  E
ABBAA  M
AAAAA  A
BBAAB  Z
ABBBA  O
AAAAA  A
AAAAB  B
AAAAA  A
AAABA  C
ABBBA  O
ABBAB  N
ABAAA  I
AAAAA  A
ABBAB  N

REMAZOABACONIAN
```

Pruebo ese password y el ZIP se descomprime. Pasamos a la siguiente fase.

En el ZIP nos encontramos dos ficheros, uno con una cadena cifrada, y otro con dos pistas:

2Dd!E2(^as/MoI>2)$U91G(::/MJn20JtF90J+t:/N#@:2)?gA1+b1>/N#772)Hm=2D$U>/N#@:0OcUk1+b@B/MK+82)Hm=2D$U?/MJk12)[$D1bCCA/N#=90KC^B1+b1:/MJn21hIk2@<3Q#Bk;05+F.B<FCcS7F_,)l+Dk\-Df[N

Y las pistas son de traca, a cualquier cosa en este reto del UAM le llaman pistas.. 😉

1. "We are the DATE" https://www.youtube.com/watch?v=tYIYRRLj-n4

2. La clave final de todo está en el corazón de Telegram, en sus comienzos...

Viendo el video, sale la canción “We are the World”. Por el texto “We are de DATE”, interpreto que buscamos una fecha. La canción pone claramente que se creó en 1985, o sea que ese año es una pista seguro. Intento buscar cosas relacionadas con cripto de ese año, busco cifrados que se inventaran ese año, o que tuvieran eventos o acontecimientos importantes. Agua, no encuentro nada.

Le doy a la pista, esta es buena, como se nota que es de pago: 😉 Nos dice que lo importante es el año, pero solo las dos ultimas cifras (85). Esto si que es una buena pista, enseguida encontramos algo relacionado, el base85. Buscamos una web para verlo online:

ASCII85 (Base85) - Decoder, Encoder, Solver, Translator - dCode
https://www.dcode.fr/ascii-85-encoding ▾ Traducir esta página
ASCII 85 is used in PDF file format for example. ... **Encryption** uses the binary code of the text (which depend on the encoding used: ASCII, Unicode, etc.).
How to encrypt using ... · How to decrypt ASCII85 ...

Metemos la cadena y la resuelve enseguida. Ya tenemos la primera parte del cifrado:

Vamos a la segunda parte, la pista nos dice: “La clave final de todo está en el corazón de Telegram, en sus comienzos”.

Intento convertir a ASCII la cadena, a ver si tiene alguna traducción que de una pista. Nada, es ilegible…

```
00000000h: 64 75 7C 49 50 03 03 01 05 00 06 54 53 52 08 51 ; du|IP......TSR.Q
00000010h: 54 06 04 54 0B 52 57 07 54 06 05 00 56 54 09 53 ; T..T.RW.T...VT.S
00000020h: 09 52 04 01 4F                                  ; .R..O
```

No tengo claro como sacar la clave, pero tampoco el algoritmo de cifrado. Estoy perdido, acudo a la segunda pista de pago: *Los grupos de Telegram pueden durar mucho tiempo pero no todo el mundo sobrevive a un XOR.*

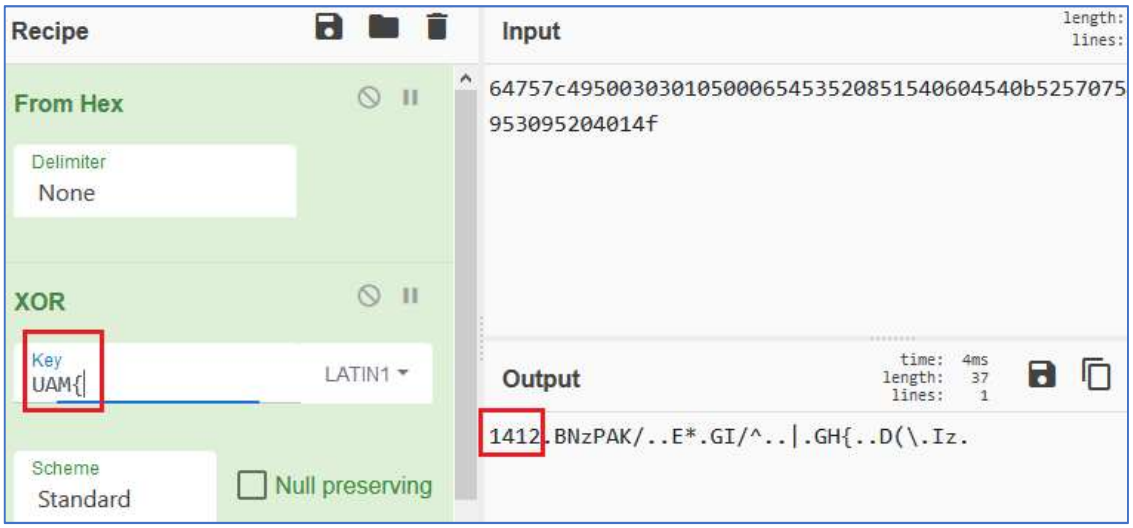
Esto es una buena pista, ya tenemos el algoritmo de cifrado, ahora nos falta encontrar la clave. Tiene que ver con telegram, empiezo a buscar cosas que estén relacionadas con los orígenes de Telegram. Pruebo claves de los nombres de los creadores, de su API, MTPRoto, etc. Nada.

Pienso en que también podría ser una fecha la clave, busco fechas relacionadas con el origen del telegram, se creó la primera versión el 14 de agosto de 2013.

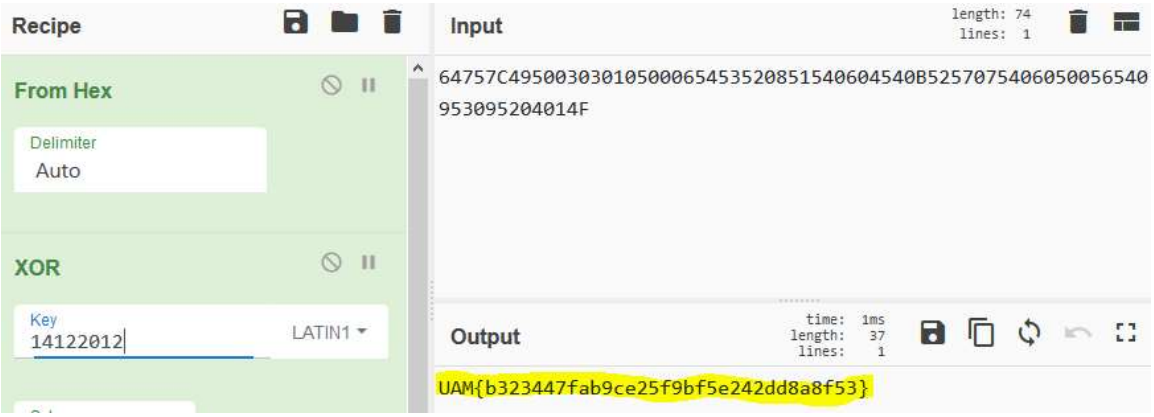
Me monto un panel en Cyberchef que me importa la cadena en Hexadecimal, y después le hace el XOR. A la cadena le quito los guiones, porque no deberían estar contenidas en el texto cifrado, o al menos eso supongo:

Pongo la fecha de creación como clave (14082013) y no resuelve nada, pero me da una pista. La cadena resultado empieza por UA, y termina en llave. Oh! Tiene pinta de que estamos cerca, realmente la salida debería ser el formato de flag de UAM{md5}. Efectivamente, cuento la longitud de la cadena original, y son justos 37 bytes, que corresponden por tamaño a los 32 del md5 más los 5 del texto UAM{}. Estamos cerca…

Si voy jugando con los números, puedo llegar a hacer que la cadena empiece por UAM{, aunque todavía no sé por qué de esa clave. Uso un truco, la operación XOR tiene la propiedad de que si A XOR B = C, entonces A XOR C = B. En este caso, sé la cadena original (A) y puedo suponer que la resultado (C) debe empezar por "UAM{", por tanto si hago ese XOR contra la clave "UAM{", me sacará los 4 primeros caracteres de la clave:



Por tanto, ya sabemos que la clave empieza por "1412", para poder formar la cadena resultante "UAM{". Sigo probando números, voy a completar una fecha cercana a la creación de Telegram, por ejemplo "14122012":



Bingo! Tengo ya una salida en formato flag, pero el problema es que no es la única. Otras fechas también generan un formato similar, con un MD5 válido dentro, como por ejemplo 14122010, 14122015, 14122017, etc.

Pruebo en la plataforma las tres flag más cercanas a esa fecha del año 2013 que se creó Telegram, generadas con las claves 14122010, 14122012 y 14122015. Ninguna es la flag correcta. La del año 14122017 la descarto inicialmente, no puede ser esa porque ese año no tiene ya nada que ver con los orígenes de Telegram.

Sigo buscando fechas en internet que pudieran cuadrar con temas de Telegram, nada, no hay forma. Al día siguiente decido buscar todas las combinaciones de clave que me generen un formato final valido, tengan o no tengan sentido o relación con Telegram. Parece que puede haber muchas, pero no son tantas. Años menores del 2000 no generan salida valida, al final voy probando y me salen unas 12 flags posibles en total. Hay tres que ya probé ayer, pero las vuelvo a probar, por si hice algo mal. Nada. Pruebo finalmente la flag UAM{b326447fab9fe25f9bf0e242dd8d8f53} correspondiente a la clave 14122017, y bingo!! ¡Es la flag y hemos superado el reto!

He averiguado la clave mitad por lógica (primera parte) relacionada con la función XOR, y la otra mitad por fuerza bruta (aunque con muy pocas combinaciones posibles) y por lógica también, aunque no en este caso por saber el origen de esa fecha.

Pregunto por privado al admin con qué evento de Telegram tiene relación esa fecha y me indica que la fecha está relacionada con la creación del grupo de Telegram de los retos del UnaAlMes. Claro, ahora si que caigo, yo me había ido por otro lado, el de la creación de la aplicación de Telegram.

Bueno, al fin y al cabo, esto es la demostración de que, a veces, hay distintas maneras de superar un CTF, y no todas son iguales.. 😉

Fdo: José Ignacio de Miguel González (nachinho3). Telegram: @jignaciodemiguel