

Ciberseguridad en Entornos de las Tecnologías de la Información

Módulo 5021 – Incidentes de Ciberseguridad

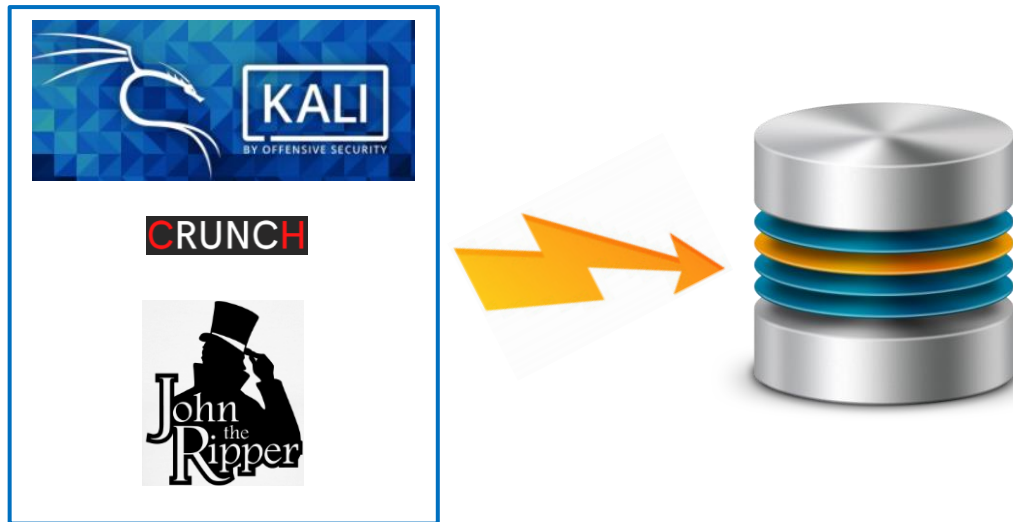
Ejercicio – Crunch y John the Ripper

Pliego de Descargo

- *Los ejercicios y conocimientos contenidos en el Módulo 5021, Incidentes de Ciberseguridad, tienen un propósito exclusivamente formativo, por lo que **nunca se deberán utilizar con fines maliciosos o delictivos.***
- *Ni el Ministerio de Educación y Formación Profesional como organismo oficial, ni el CIDEAD como área integrada en el mismo, serán responsables en ningún caso de los daños directos o indirectos que pudieran derivarse del uso inadecuado de las herramientas de hacking ético utilizadas en dichos ejercicios.*



Ataque de Ficheros con de John the Ripper



John the Ripper y Crunch – Una Pareja Fatal para las Claves

- **Crunch** es uno de los constructores de diccionarios de propósito general más utilizados en Internet, y viene instalado por defecto en Kali Linux.
- Por otra parte, **John the Ripper** es el *cracker* de passwords más popular y trabaja con base en códigos *hash*.
- Existen en Internet multitud de **herramientas de extracción de hash de diferentes tipos de ficheros**, para dar material de trabajo a John the Ripper.
- **La combinación de Crunch y John resulta ser una pareja fatal** para cualquier sistema de claves, porque puede que lleve su tiempo, pero **John siempre encuentra la clave buscada**, trabajando con Fuerza Bruta o con Diccionario.
- **La estrategia para acortar los tiempos de búsqueda está basada en Ingeniería Social**, esto es, *espionaje* puro y duro, ya sea a través de personas, de *stealers*, de *keyloggers*, o de robots de cualquier tipo.

Creación y Cifrado de un Fichero Secreto

- Creamos un fichero de Office, por ejemplo, un Excel con contenido secreto llamado “guarida.xlsx”, y lo ciframos con una clave sencilla para que el proceso de *cracking* no se extienda demasiado: “Ab#1”

Autoguardado

guardia - Última modificación: Hace 6 min

Buscar

Franci

Archivo

Inicio

Insertar

Dibujar

Disposición de página

Fórmulas

Datos

Revisar

Vista

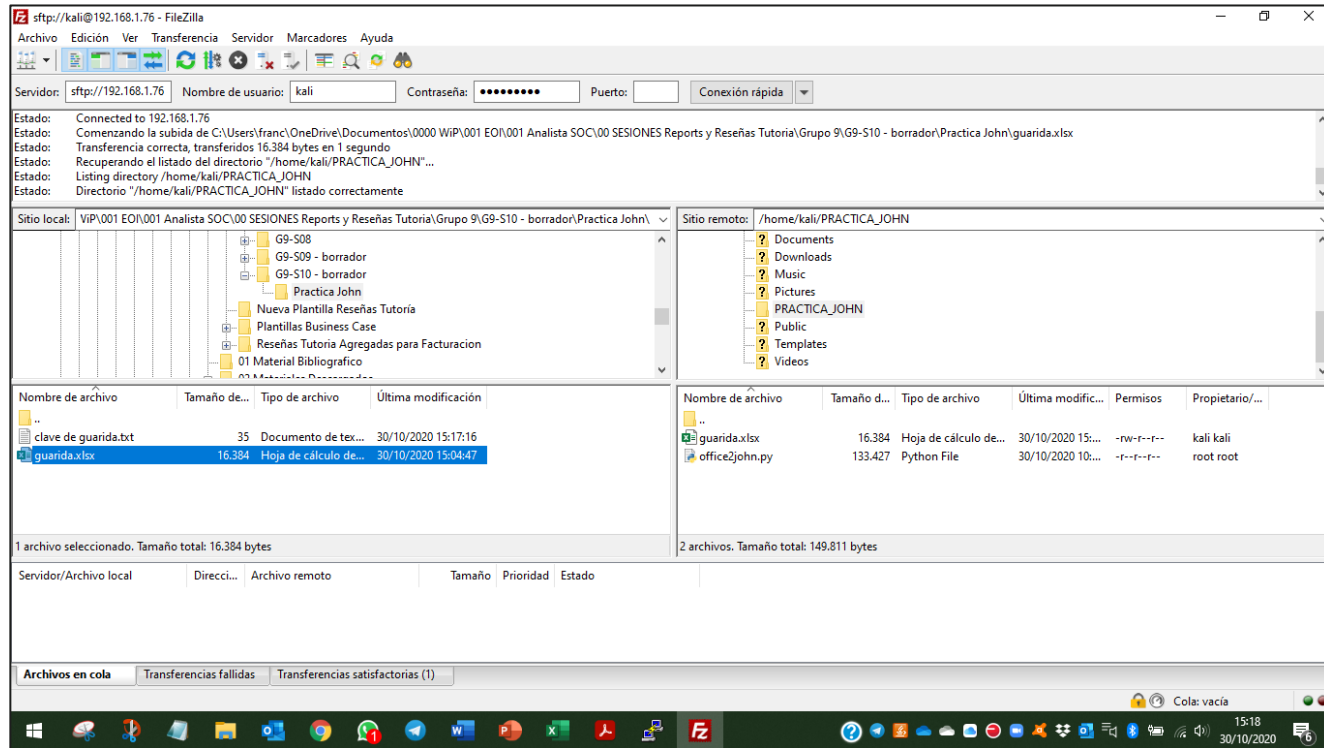
Ayuda

A1

Código

	A	B	C	D	E
1	Código	Máxima	Autor	Idioma	Traducción
2	1	Si vis pacem para bellum	Vegecio	Latín	Si quieres la paz, prepárate para la guerra
3	2	Es ist nicht leicht ein Gott zu sein	Peter Fleischmann	Alemán	No es fácil ser un dios
4	3	Homo homine lupus est	Plauto	Latín	El hombre es un lobo para el hombre
5	4	So you think you can tell heaven from hell?	David Gilmour y Roger Waters	Inglés	¿entonces crees que puedes distinguir el cielo del infierno?
6	5	Got to be some good times ahead	Farrokh Bulsara	Inglés	Tiene que haber buenos momentos más adelante
7	6	Dein Herz ist ein Pendel, es schlägt und schlägt aus	Yvonne Catterfeld	Alemán	Tu corazón es un péndulo, que va y viene

Transmisión del Fichero a Linux



- Transmitimos el fichero de Office desde el PC a la máquina Linux.
- Usamos sftp en una shell de Windows o un programa de File Transfer como Filezilla.

Recepción del Fichero en Linux

```
kali@kali: ~/PRACTICA_JOHN
kali@kali:~/PRACTICA_JOHN$ ls -l
total 152
-rw-r--r-- 1 kali kali 16384 Oct 30 14:18 guarida.xlsx
-r--r--r-- 1 root root 133427 Oct 30 09:09 office2john.py
kali@kali:~/PRACTICA_JOHN$
```

- Abrimos una shell de Linux y comprobamos que se ha recibido correctamente el fichero en la máquina.

Borrado del Histórico Previo de John

```
kali@kali: ~/.john
kali@kali:~/.john$ pwd
/home/kali/.john
kali@kali:~/.john$ ls -l
total 12
-rw----- 1 kali kali 4867 Oct 30 14:06 john.log
-rw----- 1 kali kali 164 Oct 30 14:06 john.pot
kali@kali:~/.john$ rm -f john.pot
kali@kali:~/.john$ ls -l
total 8
-rw----- 1 kali kali 4867 Oct 30 14:06 john.log
kali@kali:~/.john$
```

- Como comentamos en la Introducción, John the Ripper es una herramienta de cracking de claves sobre hash, que viene instalada por defecto en Kali Linux.
- Se puede instalar también fácilmente sobre el resto de distribuciones de Linux, no obstante, sólo la versión “Jumbo” es capaz de crackear con potencia suficiente.
- Borramos el histórico de passwords crackeadas por John, eliminando el fichero /home/kali/.john/john.pot

Extracción del Hash con office2john

```
kali@kali: ~/PRACTICA_JOHN
kali@kali:~/PRACTICA_JOHN$ pwd
/home/kali/PRACTICA_JOHN
kali@kali:~/PRACTICA_JOHN$ ls -l
total 152
-rw-r--r-- 1 kali kali 16384 Oct 30 14:18 guarida.xlsx
-r--r--r-- 1 root root 133427 Oct 30 09:09 office2john.py
kali@kali:~/PRACTICA_JOHN$ python office2john.py guarida.xlsx > hash_guarida
kali@kali:~/PRACTICA_JOHN$ ls -l
total 156
-rw-r--r-- 1 kali kali 16384 Oct 30 14:18 guarida.xlsx
-rw-r--r-- 1 kali kali 172 Oct 30 14:28 hash_guarida
-r--r--r-- 1 root root 133427 Oct 30 09:09 office2john.py
kali@kali:~/PRACTICA_JOHN$
```

- En Internet existen multitud de extractores de hash a partir de ficheros de diferentes formatos, como es el caso de [office2john](#), que extrae el hash de cualquier fichero cifrado de MS-Office (script de Python que se puede descargar desde GitHub).
- Extraemos el hash del fichero con office2john:

```
python office2john.py  
guarida.xlsx > hash_guarida
```
- El hash es un conjunto de caracteres que se generan a partir de una password, aplicando un algoritmo de cálculo especial.

Generación del Diccionario con Crunch

```
kali@kali: ~/PRACTICA_JOHN
kali@kali:~/PRACTICA_JOHN$ pwd
/home/kali/PRACTICA_JOHN
kali@kali:~/PRACTICA_JOHN$ ls -l
total 156
-rw-r--r-- 1 kali kali 16384 Oct 30 14:18 guarida.xlsx
-rw-r--r-- 1 kali kali 172 Oct 30 14:28 hash_guarida
-r--r--r-- 1 root root 133427 Oct 30 09:09 office2john.py
kali@kali:~/PRACTICA_JOHN$ crunch 4 4 -t ,@^% -o diccionario.lst
Crunch will now generate the following amount of data: 1115400 bytes
1 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 223080

crunch: 100% completed generating output
kali@kali:~/PRACTICA_JOHN$ ls -l
total 1252
-rw-r--r-- 1 kali kali 1115400 Oct 30 14:30 diccionario.lst
-rw-r--r-- 1 kali kali 16384 Oct 30 14:18 guarida.xlsx
-rw-r--r-- 1 kali kali 172 Oct 30 14:28 hash_guarida
-r--r--r-- 1 root root 133427 Oct 30 09:09 office2john.py
kali@kali:~/PRACTICA_JOHN$
```

- Creamos el diccionario con crunch y usando las pistas de ingeniería social (espionaje), esto es: una letra mayúscula, una minúscula, un símbolo y un dígito (ver instrucciones detalladas con *man crunch*):
`crunch 4 4 -t ,@^% -o diccionario.lst`
- Las dimensiones de los diccionarios crecen exponencialmente al incorporar más caracteres y más variaciones de los mismos (letras, números, símbolos).
- John también puede trabajar con su diccionario propio o sin diccionario, comprobando por orden TODAS las posibles claves (Fuerza Bruta).

Comprobación del Diccionario

```
kali@kali: ~/PRACTICA_JOHN
kali@kali:~/PRACTICA_JOHN$ pwd
/home/kali/PRACTICA_JOHN
kali@kali:~/PRACTICA_JOHN$ ls -l
total 1252
-rw-r--r-- 1 kali kali 1115400 Oct 30 14:30 diccionario.lst
-rw-r--r-- 1 kali kali 16384 Oct 30 14:18 guarida.xlsx
-rw-r--r-- 1 kali kali 172 Oct 30 14:28 hash_guarida
-r--r--r-- 1 root root 133427 Oct 30 09:09 office2john.py
kali@kali:~/PRACTICA_JOHN$ cat diccionario.lst | grep Ab#1
Ab#1
kali@kali:~/PRACTICA_JOHN$
```

- Comprobamos que el diccionario contiene efectivamente nuestra password:

```
cat diccionario.lst
| grep Ab#1
```

Cracking con John the Ripper

```
kali@kali: ~/PRACTICA_JOHN
kali@kali:~/PRACTICA_JOHN$ pwd
/home/kali/PRACTICA_JOHN
kali@kali:~/PRACTICA_JOHN$ ls -l
total 1252
-rw-r--r-- 1 kali kali 1115400 Oct 30 14:30 diccionario.lst
-rw-r--r-- 1 kali kali 16384 Oct 30 14:18 guarida.xlsx
-rw-r--r-- 1 kali kali 172 Oct 30 14:28 hash_guarida
-r--r--r-- 1 root root 133427 Oct 30 09:09 office2john.py
kali@kali:~/PRACTICA_JOHN$ john --wordlist=diccionario.lst hash_guarida
Using default input encoding: UTF-8
Loaded 1 password hash (Office, 2007/2010/2013 [SHA1 128/128 ASIMD 4x / SHA512 1
28/128 ASIMD 2x AES])
Cost 1 (MS Office version) is 2013 for all loaded hashes
Cost 2 (iteration count) is 100000 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Ab#1 (guarida.xlsx)
lg 0:00:00:12 DONE (2020-10-30 14:37) 0.07806g/s 27.47p/s 27.47c/s 27.47c/s Ab#6
..Ab#1
Use the "--show" option to display all of the cracked passwords reliably
Session completed
kali@kali:~/PRACTICA_JOHN$
```

- Lanzamos el proceso de cracking de la password sobre el fichero que contiene el hash, y utilizando el diccionario que hemos creado:

```
john --wordlist=diccionario.lst
hash_guarida
```

- Cuando john localiza la password, la muestra por pantalla y la almacena en su fichero histórico john.pot. Si se intenta crackear un hash ya procesado, john mira primero si lo tiene dentro de su histórico, y si lo encuentra no lanza el proceso. Por eso, si se cambia la password de un fichero y se quiere reintentar el proceso, primero se tendrá que borrar el histórico.

Consulta del Histórico de John

```
kali@kali: ~/PRACTICA_JOHN
kali@kali:~/PRACTICA_JOHN$ pwd
/home/kali/PRACTICA_JOHN
kali@kali:~/PRACTICA_JOHN$ ls -l
total 1252
-rw-r--r-- 1 kali kali 1115400 Oct 30 14:30 diccionario.lst
-rw-r--r-- 1 kali kali 16384 Oct 30 14:18 guarida.xlsx
-rw-r--r-- 1 kali kali 172 Oct 30 14:28 hash_guarida
-r--r--r-- 1 root root 133427 Oct 30 09:09 office2john.py
kali@kali:~/PRACTICA_JOHN$ john --show hash_guarida
guarida.xlsx:Ab#1

1 password hash cracked, 0 left
kali@kali:~/PRACTICA_JOHN$
```

- Para ver si un fichero de hash ya fue crackeado, se puede consultar el histórico con el comando siguiente, aplicado sobre el fichero que contiene el hash:

```
john --show hash_guarida
```

La Complejidad Creciente de los Diccionarios

```
kali@kali: ~/PRACTICA_JOHN

(kali@kali)-[~/PRACTICA_JOHN]
$ crunch 4 4 -t @@@@ -o diccionario
Crunch will now generate the following amount of data: 2284880 bytes
2 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 456976

crunch: 100% completed generating output

(kali@kali)-[~/PRACTICA_JOHN]
$
```

- Diccionario con palabras de 4 minúsculas: 2 MB.

- En las siguientes capturas de pantalla se puede comprobar como la complejidad del diccionario escala cuadráticamente a medida que se van añadiendo caracteres.
- Con la complejidad del diccionario también escala el tiempo que dura un ataque, por lo que si se quiere complicar la labor a los *hackers*, lo mejor es usar claves complicadas y cambiarlas a menudo.

La Complejidad Creciente de los Diccionarios

```
kali@kali: ~/PRACTICA_JOHN

(kali@kali)-[~/PRACTICA_JOHN]
$ crunch 5 5 -t @@@@ -o diccionario
Crunch will now generate the following amount of data: 71288256 bytes
67 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 11881376
```

- Diccionario con palabras de 5 minúsculas: 67 MB.

```
kali@kali: ~/PRACTICA_JOHN

(kali@kali)-[~/PRACTICA_JOHN]
$ crunch 6 6 -t @@@@@ -o diccionario
Crunch will now generate the following amount of data: 2162410432 bytes
2062 MB
2 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 308915776
```

- Diccionario con palabras de 6 minúsculas: 2 GB.

La Complejidad Creciente de los Diccionarios

```
kali@kali: ~/PRACTICA_JOHN

(kali@kali)-[~/PRACTICA_JOHN]
$ crunch 7 7 -t 00000000 -o diccionario
Crunch will now generate the following amount of data: 64254481408 bytes
61277 MB
59 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 8031810176
```

- Diccionario con palabras de 7 minúsculas: 59 GB.

```
kali@kali: ~/PRACTICA_JOHN

(kali@kali)-[~/PRACTICA_JOHN]
$ crunch 8 8 -t 0000000000 -o diccionario
Crunch will now generate the following amount of data: 1879443581184 bytes
1792377 MB
1750 GB
1 TB
0 PB
Crunch will now generate the following number of lines: 208827064576
```

- Diccionario con palabras de 8 minúsculas: 1750 GB.