Lo que haremos es separar los labos en Windows y Linux porque la última vez me faltó :D (perdón a los linuxeros).

Hoy veremos el famoso -y potente- software “john the ripper”. Está la **versión gratuita** por lo que pueden **descargarlo** de la página **oficial**: http://www.openwall.com/john/ y ahí mismo tienen su "signature" para comprobar si el archivo está o no modificado.

----------**Linux**----------

También pueden descargarlo desde el terminal de ésta manera “sudo apt-get install john”. Para los que no sepan de comandos de Linux, quédense tranquilos que más adelante los veremos a fondo.

Se descargan el que diga

John the Ripper 1.8.0 (Unix - sources, tar.gz, 5.2 MB) and its signature

El ".tar.gz" al final del nombre, nos da la referencia de que es un comprimido. Click derecho, descomprimir; o vamos con la terminal hacia el path correspondiente y usamos el comando "tar -xzvf comprimido"

Luego, con la terminal, tecleamos "**john**" y veremos como nos entrega la ayuda correspondiente y todas sus opciones.

```
Terminal
pelado@Lucho-pc:~$ john
John the Ripper password cracker, version 1.7.8
Copyright (c) 1996-2011 by Solar Designer
Homepage: http://www.openwall.com/john/

Usage: john [OPTIONS] [PASSWORD-FILES]
--single                  "single crack" mode
--wordlist=FILE --stdin   wordlist mode, read words from FILE or stdin
--rules                   enable word mangling rules for wordlist mode
--incremental[=MODE]      "incremental" mode [using section MODE]
--external=MODE           external mode or word filter
--stdout[=LENGTH]         just output candidate passwords [cut at LENGTH]
--restore[=NAME]          restore an interrupted session [called NAME]
--session=NAME            give a new session the NAME
--status[=NAME]           print status of a session [called NAME]
--make-charset=FILE       make a charset, FILE will be overwritten
--show                    show cracked passwords
--test[=TIME]             run tests and benchmarks for TIME seconds each
--users=[-]LOGIN|UID[,..] [do not] load this (these) user(s) only
--groups=[-]GID[,..]      load users [not] of this (these) group(s) only
--shells=[-]SHELL[,..]    load users with[out] this (these) shell(s) only
--salts=[-]COUNT          load salts with[out] at least COUNT passwords only
--format=NAME             force hash type NAME: DES/BSDI/MD5/BF/AFS/LM/crypt
--save-memory=LEVEL       enable memory saving, at LEVEL 1..3
```

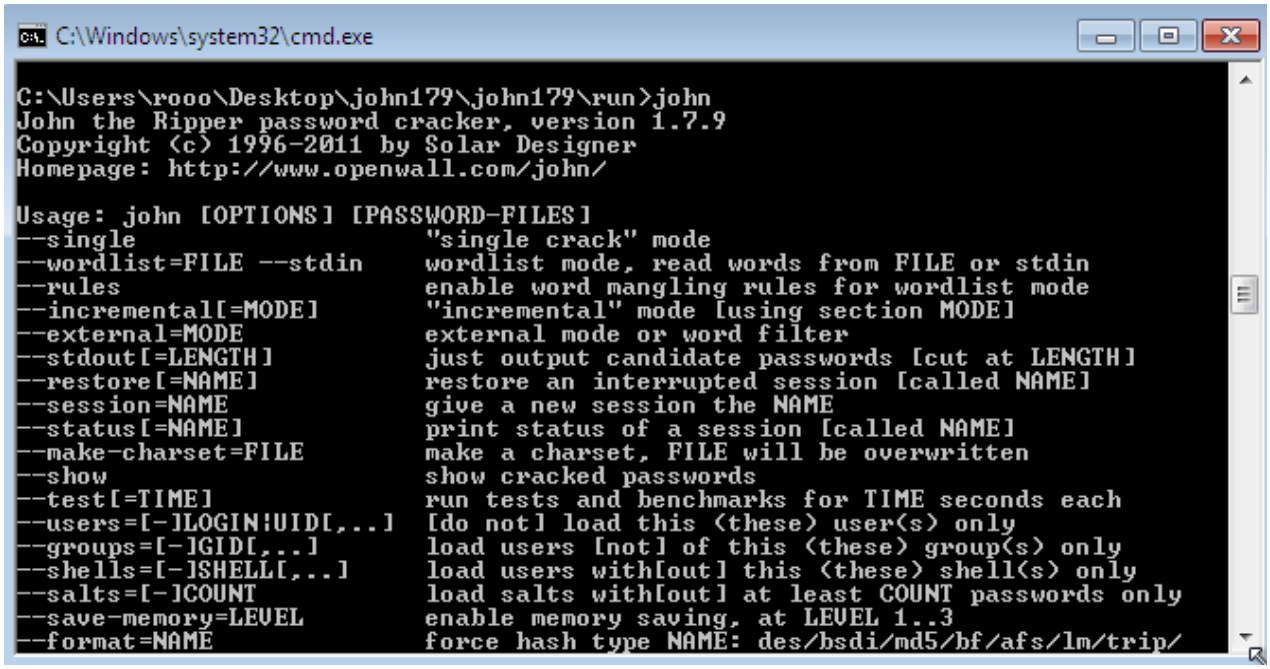
----------**Windows**--------

Se descargan el que diga

John the Ripper 1.7.9 (Windows - binaries, ZIP, 2029 KB) and its signature

Es un ".zip" que debemos descomprimir. Vamos a "ejecutar"(teclawindows+R)--->cmd---->vamos hasta la ruta de la carpeta "run" dentro de la carpeta descomprimida del john.

Lo hacemos con el comando "cd C:/lugardondelohayandescomprimido". En mi caso estaría en el escritorio por lo que sería "C:/Users/usuario/Desktop/John/run". Una vez hecho, tecleamos "**john**" y veremos la ayuda correspondiente con todas sus opciones.



De ahora en adelante, en cualquier sistema operativo que tengamos, se usa de la misma manera.

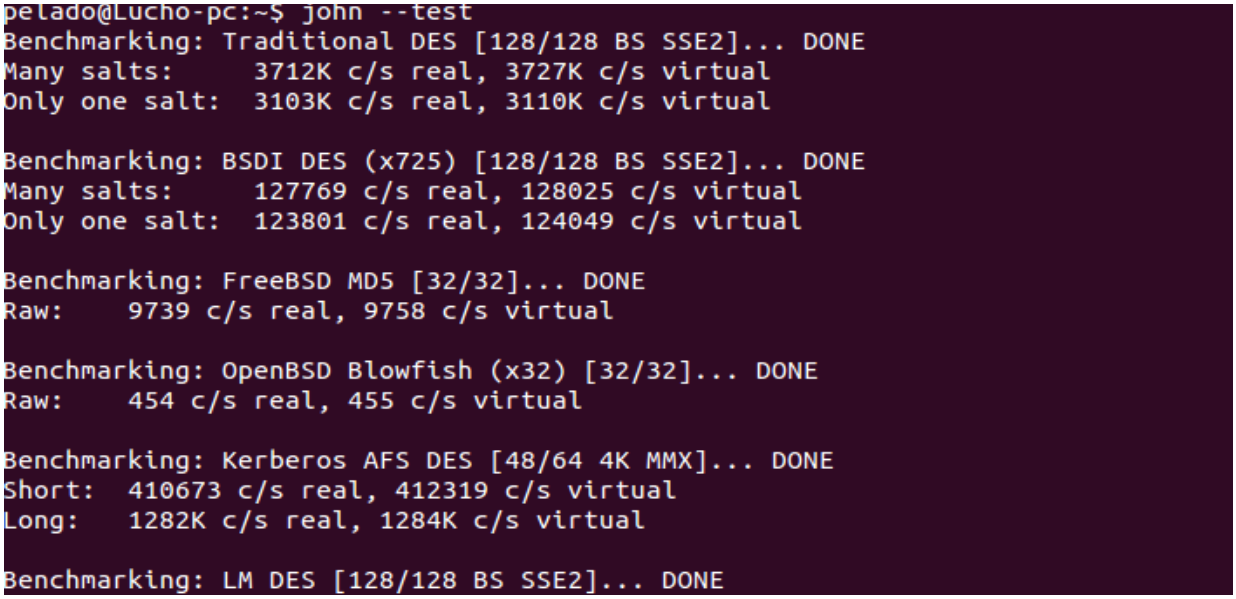
Yo en particular voy a mostrarles las imagenes de las pruebas desde la terminal de Ubuntu, pero en Windows debería ser exactamente igual.

¡Vamos! Siempre que se use este programa será de manera **"john --comando o -c=parámetros o parámetros"**

Primero hagamos lo que sería un **benchmark** o prueba de procesamiento:

> **"john --test"**

Les saldrá algo como esto, donde hace pruebas:



c/s significa encriptaciones por segundo. Es decir cuantos intentos por segundo puede hacer para el ataque. Cada cifrado tiene una fortaleza distinta y veamos que también nos aclara cuánto tardaría con los salts. **Cuanto más poderoso nuestro procesador, más poderoso será el ataque.**

Para las pruebas siguientes, en un fichero.txt voy a guardar **3 hashes DES**, uno fácil (**123456789**), otro de nivel medio (**roadd123456789**), y un último más complicado (**RoaddDogg.123321**). Lo guardamos donde estamos parados (en Windows sería dentro del directorio del run, en Linux donde estén en la terminal. Para saber en qué directorio estamos en Linux, usamos el comando "pwd")

EDIT: el hash DES sólo permite hasta 8 caracteres, por lo que me truncó todo a 8 caracteres maximo.

La página que utilizé para generar los hashes fue: https://www.quickhash.com/

El próximo comando que usaremos será

john --single fichero.txt

con éste, el ataque será un diccionario de ataque que recompiló algunos datos del usuario. Corto pero quizás es efectivo y es tan rápido que no está mal intentarlo. Intentémoslo con nuestro fichero:

```
pelado@Lucho-pc:~/Desktop$ john --single fichero.txt
Loaded 3 password hashes with no different salts (Traditional DES [128/128 BS SS
E2])
guesses: 0  time: 0:00:00:00 100%  c/s: 0.00
pelado@Lucho-pc:~/Desktop$
```

Como verán, el ataque no fue exitoso **(guesses: 0)**. Esto pasó porque mis datos no fueron suficientemente buenos como un diccionario de ataque.

Veamos que pasaría con un diccionario hecho y derecho. Hice uno con muchas palabras, donde una es la correcta de las 3 (la más difícil). **Veamos qué sucede cuando lanzamos el ataque con diccionario.**

john --wordlist=lista.txt fichero.txt

Encontrará sólo 1 de las palabras ya que es la única que está en coincidencia.

```
pelado@Lucho-pc:~/Desktop$ john --wordlist=lista.txt fichero.txt
Loaded 3 password hashes with no different salts (Traditional DES [128/128 BS SS
E2])
RoaddDog         (?)
guesses: 1  time: 0:00:00:00 100%  c/s: 404  trying: 123456 - RoaddDog
Use the "--show" option to display all of the cracked passwords reliably
```

¡Guesess: 1! Lo hizo muy rápido porque no era un diccionario extenso. Para saber si John ya hizo su trabajo, con el comando

john --show fichero.txt

vemos el estado de crackeo. En este caso tengo **1/3 hashes crackeados, por lo que me lo dirá.**

```
pelado@Lucho-pc:~/Desktop$ john --show fichero.txt
?:RoaddDog

1 password hash cracked, 2 left
```

Pero aún **faltan 2 hashes** y no queremos quedarnos con el pan a medio hacer. Hay un ataque muy **poderoso pero muy lento** que es

john --incremental fichero.txt

que sería el famoso fuerza bruta.

Aunque también podemos hacer un

john --incremental=RULES fichero.txt

para darle **reglas específicas**.

Las reglas pueden ser:

All : Minúsculas, Mayúsculas, dígitos, puntuación.

Alpha : Minúsculas.

Digits : Del 0 al 9.

LANMan : Similar al All, pero sin minúsculas.

Alnum: Combinación alfanumérica con solo minúsculas.

Claramente, por suerte en JTR tenemos una regla que nos satisface.

> **john --incremental=Alnum fichero.txt**

Lamentablemente esto podría llevar mucho tiempo. Hagamos la cuenta de cuánto podría llevar:

Tenemos 36 caracteres por combinación, donde puede haber 1 a 14 caracteres (roadd123456789 lleva 14 caracteres), es decir que la combinacion de esto sería algo así como 36^14 = $6.140942214 \times 10^{21}$ y que si hacemos casi 4 mil intentos por segundo tal y como decía el test, nos tomaría algo así como **48.68 mil millones de años**... Bueno, menos mal que es un ataque automatizado ¿Verdad? Lo siento, pero **el ataque se vuelve algo ilógico** cuando hacemos las cuentas.

Necesitamos algo más específico. Como por ejemplo, saber que las passwords tienen entre **9 y 14 caracteres.** Hagamos la cuenta, 36^5 = 60466176 con 4 mil de intentos por segundo sería igual a **4 horas.** Si esto es verdad, nos ahorraríamos muchísimo tiempo. Necesitamos cambiar las reglas por defecto.

En Linux, las reglas estarán en /etc/john/john.conf. En Windows estarán en el john.ini que se encuentra en el run.

Editemos ese archivo y vayamos (o busquemos) al lugar donde diga

> **[Incremental:All]**
> **File = $JOHN/all.chr**
> **MinLen = 0**
> **MaxLen = 8**
>
> **CharCount = 95**

Veamos que tenemos dos parámetros muy buenos, **MinLen=0 y MaxLen=8**. Obviamente, indican la longitud mínima y máxima respectivamente. Editémoslo de tal manera que el **mínimo** sea **9 y el máximo sea 14 y lanzemos el ataque.**

EDIT: Me equivoqué al editar el [Incremental:All] ya que en realidad debería haber editado el [incremental:Alnum]. Por eso me dejó realizar el ataque al fichero.txt aunque DES acepte hasta 8 caracteres máximo. Si cambian las propiedades de Alnum, en este caso, les dará error.

```
pelado@Lucho-pc:~/Desktop$ john --incremental=Alnum fichero.txt
Loaded 3 password hashes with no different salts (Traditional DES [128/128 BS SS
E2])
Remaining 2 password hashes with no different salts
12345678         (?)
guesses: 1  time: 0:00:07:54  c/s: 3096K  trying: lkc0up9 - lkc011a
Use the "--show" option to display all of the cracked passwords reliably
Session aborted
pelado@Lucho-pc:~/Desktop$
```

Bueno, al toque sacó una de las dos: la numérica. Pero la verdad es que no tengo tiempo de esperar la otra y mejor lo dejo para otro momento. Para esto puedo hacer ctrl+C y JTR abortará. Eso sí, por suerte JTR piensa en nosotros y nos dejará **restaurar el ataque** por donde íbamos para no perder el tiempo que ya se tomó en intentar antes, si es que nos tomamos el trabajo de indicarle una sesión específica. Esto lo hacemos con

john --session:miAtaque fichero.txt

donde "miAtaque" es el nombre que tendrá la sesión.

```
pelado@Lucho-pc:~/Desktop$ john --incremental=Alnum --session:miAtaque fichero.t
xt
Loaded 3 password hashes with no different salts (Traditional DES [128/128 BS SS
E2])
Remaining 1 password hash
guesses: 0  time: 0:00:00:11  c/s: 2754K  trying: nb38ti - nb391t
Session aborted
```

Y ahora tenemos ganas de volver a **seguir la sesión ya iniciada**, y usamos el comando

john --restore:miAtaque

```
pelado@Lucho-pc:~/Desktop$ john --restore:miAtaque
Loaded 3 password hashes with no different salts (Traditional DES [128/128 BS SS
E2])
Remaining 1 password hash
guesses: 0  time: 0:00:00:20  c/s: 2839K  trying: kcrdul - kcrmij
```

O podemos ver el estado del ataque con

john --status=miAtaque

```
pelado@Lucho-pc:~/Desktop$ john --status=miAtaque
guesses: 0  time: 0:00:00:21  c/s: 2703K
```

Por último, voy a agregar que otra de las cualidades que tiene es que podemos **especificar** cuál es nuestro **objetivo**, aclarando MD5, DES, BF, LM, AFS, etc.

Esto lo hacemos mediante

john --format=HASHNAME fichero.txt

Y bueno, hasta aquí será hoy. La próxima ya es el repaso del examen y el examen. No sé si el

examen irá antes o después del concurso, pero claramente no irá mientras.

Cualquier cosa pueden mandarme mail a: r0add@hotmail.com

Para donaciones, pueden hacerlo en bitcoin en la dirección siguiente:

1HqpPJbbWJ9H2hAZTmpXnVuoLKkP7RFSvw

Roadd.
