uc3m | Universidad Carlos III de Madrid

PRÁCTICA 1:

ESTUDIO DE LA FORTALEZA DE CONTRASEÑAS

Carmen Abans Maciel - 100432086

Noelia Hernandez Rodriguez - 100432116

25 Septiembre 2023

Ingeniería de la Ciberseguridad

Índice

Estudio de la fortaleza de contraseñas

El objetivo de esta práctica es profundizar en el estudio de la fortaleza de contraseñas y el uso de herramientas de rotura de las mismas. Para ello, se generarán varios conjuntos de contraseñas de distinta complejidad y se estudiará empíricamente el esfuerzo necesario para romperlas usando distintas estrategias.

Generación de los datasets

Para analizar las estrategias para romper contraseñas se han generado 5 sets de 5 datasets de 100 contraseñas cada uno usando Python. Detallamos a continuación cómo se han generado.

Set de contraseñas con letras minúsculas

Este set contiene 5 datasets de contraseñas de longitud 3, …, 7 formadas únicamente por letras minúsculas. Cada dataset lo hemos denominado minus-x, siendo x la longitud de las contraseñas creadas. Para generarlas utilizamos el siguiente programa de python:

```python
def generate_password_minus(length):
      characters = string.ascii_lowercase
      return ''.join(random.choice(characters) for _ in range(length))
```

Set de contraseñas con letras mayúsculas

Este set contiene 5 datasets de contraseñas de longitud 3, …, 7 formadas únicamente por letras mayúsculas. Cada dataset lo hemos denominado mayus-x, siendo x la longitud de las contraseñas creadas. Para generarlas utilizamos el siguiente programa de python:

```python
def generate_password_mayus(length):
      characters = string.ascii_uppercase
      return ''.join(random.choice(characters) for _ in range(length))
```

Set de contraseñas numéricas

Este set contiene 5 datasets de contraseñas de longitud 3, …, 7 formadas únicamente por números. Cada dataset lo hemos denominado num-x, siendo x la longitud de las contraseñas creadas. Para generarlas utilizamos el siguiente programa de python:

```python
def generate_password_num(length):
      characters = string.digits
      return ''.join(random.choice(characters) for _ in range(length))
```

Set de contraseñas alfanuméricas con símbolos

Este set contiene 5 datasets de contraseñas de longitud 3, …, 7 formadas por caracteres alfanuméricos (tanto mayúsculas como minúsculas) y símbolos. Cada dataset lo hemos denominado ans-x, siendo x la longitud de las contraseñas creadas. Para generarlas utilizamos el siguiente programa de python:

```python
def generate_password_ans(length):
      characters = string.ascii_letters+string.digits+string.punctuation
      return ''.join(random.choice(characters) for _ in range(length))
```

Diccionario

Para crearlos hemos ideado 5 estrategias que implican el uso de palabras tomadas de este diccionario. Cada dataset lo hemos denominado dic-X, siendo X el orden de creación. A continuación, detallamos las estrategias usadas para crear cada uno de los dataset.

dic-1

La primera estrategia para crear el dataset dic1 trata sencillamente de coger 100 palabras aleatorias del diccionario. Para generarlas utilizamos el siguiente programa de python:

```python
def generate_password_dic1(length, dictionary):
      return random.sample(dictionary, length)
```

dic-2

La segunda estrategia trata de coger palabras aleatorias del diccionario pero a estas les hace cambios de minúsculas a mayúsculas y viceversa (toggle). Para generarlas utilizamos el siguiente programa de python:

```python
def generate_password_dic2(length, dictionary):
```

```
        passwords = random.sample(dictionary, length)
        for i in range(length):
              password = list(passwords[i])
              for j in range(len(password)):
                    if random.choice([True, False]):
                    password[j] = password[j].swapcase()
              passwords[i] = ''.join(password)
         return passwords
```

dict-3

La tercera estrategia trata de coger palabras aleatorias del diccionario y añadiendoles a continuación de las mismas su reflejo. Para generarlas utilizamos el siguiente programa de python:

```
Python
def generate_password_dic3(length, dictionary):
      passwords = random.sample(dictionary, lenghth)
      result = []

      for word in passwords:
            reversed_word = word[::-1]
            doubled_word = word + reversed_word
            result.append(doubled_word)

      return result
```

dict-4

La cuarta estrategia trata de coger palabras aleatorias del diccionario pero a estas les hace los siguientes cambios:

- Si hay una letra ‘o’ la cambiará por el número 0.
- Si hay una letra ‘i’ la cambiará por el número 1.
- Si hay una letra ‘e’ la cambiará por el número 3.
- Si hay una letra ‘a’ la cambiará por el número 4.
- Si hay una letra ‘s’ la cambiará por el número 5.
- Si hay una letra ‘t’ la cambiará por el número 7.
- Si hay una letra ‘b’ la cambiará por el número 8.

Para generarlas utilizamos el siguiente programa de python:

```python
def generate_password_dic4(length, dictionary):
    def replace_chars(char):
        replacements = {'o': '0', 'i': '1', 'e': '3', 'a': '4', 's': '5',
't': '7', 'b': '8'}
        return replacements.get(char, char)
    passwords = random.sample(dictionary, length)
    for i in range(length):
        passwords[i] = ''.join(replace_chars(char) for char in
passwords[i])
    return passwords
```

dict-5

La quinta estrategia trata de coger palabras aleatorias del diccionario pero revirtiendo el orden de las letras que las componen. Es decir, si la palabra era password con el cambio seria drowssap. Para generarlas utilizamos el siguiente programa de python:

```python
def generate_password_dic5(length, dictionary):
    passwords = random.sample(dictionary, length)
    for i in range(length):
        passwords[i] = passwords[i][::-1]
    return passwords
```

Metodología

Para utilizar John the Ripper (JtR) hemos cifrado las contraseñas generadas en el paso anterior con el cifrado md5crypt. Esto se puede pasar como parámetro explícitamente pero JtR lo detecta instantáneamente.

Hemos diseñado 5 estrategias diferentes para romper contraseñas usando John the Ripper. Cada estrategia viene definida por un modo de rotura y, cuando proceda, los correspondientes parámetros, reglas o fichero de configuración adaptados a cada dataset.

En JtR existen 5 métodos de rotura que son:

- Modo por diccionario (Wordlist): Estos son aquellos métodos que implican el uso de un diccionario y reglas.
- Modo single crack: Este método utiliza reglas e información adicional de las cuentas de usuario de la contraseña. En nuestro caso no tenemos ninguna información del usuario asociada a las contraseñas ya que son de creación aleatoria y no tiene mucho sentido

usarlo pero teóricamente la documentación indica que es más eficiente que el uso de wordlist.

- Modo incremental: Este es el método de fuerza bruta por excelencia. Trata de comprobar todas las combinaciones posibles de contraseñas.
- Modo externo: Este modo se utiliza añadiendo secciones al archivo de configuración (List.External:MODE) en el que podemos colocar código que genere contraseñas candidatas en el lenguaje de programación C.

Para las metodologías que utilizan ataques por diccionario hemos ideado tres diccionarios:

1. Wordlist_1. El primero será para trabajar con los datasets de contraseñas generadas de forma aleatoria de letras (es decir los datasets de minus y mayus). Este diccionario se ha creado con 6.000 palabras aleatorias tanto con letras minúsculas como mayúsculas. Por ejemplo las tres primeras contraseñas son: [fMTdMzK, wIcjQgQ, PhFemyT].
2. Wordlist_2. El siguiente será para trabajar con alfanuméricos y contendrá tanto números como elementos ASCII. En total tendrá unas 6.000 palabras como por ejemplo: [4674481, 9746614, 1070029].
3. Rockyou-75. El tercer diccionario será el que hemos empleado para crear los datasets de dic que tiene 59.186 palabras.

Método 1. Ataque por fuerza bruta

Esta estrategia de ataque utiliza la fuerza bruta según la configuración que tenemos en cada dataset, usando el siguiente formato:

```
Unset
john --length=X --incremental=MODE archivo_hash.txt

john --min-length=X --max-length=Y --incremental=MODE archivo_hash.txt
```

Como en este caso conocemos el tamaño exacto de las contraseñas hemos decidido usar simplemente length igualado a la longitud de cada dataset (X). Si no sabemos la longitud exacta simplemente colocamos una longitud mínima y máxima. Esto lo hacemos porque en la realidad hay muchos lugares con contraseñas de una longitud exacta conocida como podría ser el pin del móvil (contraseña numérica de 4 números) o el cvv de las tarjetas bancarias (contraseña numérica de tamaño 3). Pero lo general es que se pida al usuario hacer una contraseña en un rango de caracteres determinado.

El modo incremental intenta todas las combinaciones posibles según los parámetros especificados en modo y longitud. Los modos (MODE) que admite este tipo de ataque son:

- "ASCII" (todos los 95 caracteres ASCII imprimibles)
- "LM_ASCII" (para usar en hashes LM)
- "Alnum" (todos los 62 caracteres alfanuméricos)
- "Alpha" (todas las 52 letras)

- "LowerNum" (letras minúsculas y dígitos, un total de 36)
- "UpperNum" (letras mayúsculas y dígitos, un total de 36)
- "LowerSpace" (letras minúsculas y espacio, un total de 27)
- "Lower" (letras minúsculas),
- "Upper" (letras mayúsculas)
- "Digits" (solo dígitos)
- Personal (se puede configurar un nuevo modo añadiendolo en el archivo de configuración indicado como: [Incremental:Personal])

Para cada tipo de dataset hemos ideado los siguientes enunciados atendiendo a su longitud y formato. Indicamos a continuación un ejemplo de cada tipo de dataset:

```python
# DATASET DE MINUSCULAS :
john length=3 --incremental=Lower minus_3_hash_md5.txt


# DATASET DE MAYUSCULAS :
john length=3 --incremental=Upper mayus_3_hash_md5.txt


# DATASET DE NUMEROS :
john length=3 --incremental=Digits num_3_hash_md5.txt


# DATASET DE SIMBOLOS :
john length=3 --incremental=ASCII ans_1_hash_md5.txt


# DATASETS DE DICCIONARIOS : colocamos un rango de tamaño
john --min-length=3 --max-length=15 --incremental=ASCII dic_1_hash_md5.txt
```

Método 2. Ataque por fuerza bruta con aplicación de máscara

La segunda estrategia llevada a cabo será el modo de fuerza bruta aplicando máscaras para descifrar las contraseñas. John the Ripper realiza combinaciones de palabras utilizando las máscaras especificadas consiguiendo así descifrar las contraseñas de manera más sencilla. Es un ataque parecido al anterior pero en vez de con incremental con máscara. Esto en la vida real se podría usar cuando sabemos que la contraseña debe estar compuesta de una determinada manera como con ciertos símbolos, mayúsculas y minúsculas de tamaño pequeño.

La forma de ejecutarla sería de la siguiente manera:

```unset
john --mask='MASK'archivo_hash.txt
john --mask=ARCHIVO archivo_hash.tx
```

Dentro de las comillas irían las máscaras que pueden usar los siguientes parámetros según nuestras especificaciones:

- ?: Indica que es un único elemento.
- (*): Indica que son varios elementos de x formato
- “d”: Hace referencia a los dígitos (números).
- “l”: Hace referencia a letras minúsculas.
- “u”: Hace referencia a letras mayúsculas.
- “s”: Hace referencia a caracteres especiales ASCII.
- “a”: Hace referencia a todos los caracteres ASCII.
- Archivo. Se le puede pasar también un archivo con varias máscaras en caso de querer usar más de una pero no tendría mucho sentido en nuestro caso con el límite de tiempo ya que nunca acabaría.

Para cada tipo de dataset hemos ideado las siguientes máscaras atendiendo a su longitud y formato. Indicamos a continuación un ejemplo de cada tipo de dataset:

```python
# DATASET DE MINUSCULAS : ?l indica un caracter de letra minuscula si ponemos tres es una palabra de tamaño 3 solo minusculas. Como no conocemos ningun patron no podemos usarlo pero si por ejemplo todas empezaran por fer_ podriamos poner fer?l
john --mask='?l?l?l' minus_3_hash_md5.txt

# DATASET DE MAYUSCULAS : ?u indica un caracter de letra mayuscula, si ponemos tres es una palabra de tamaño 3 unicamente de letras mayusculas.
john --mask='?u?u?u' mayus_3_hash_md5.txt

# DATASET DE NUMEROS : ?d indica un numero desconocido, si ponemos tres simboliza un numero de longitud 3 (111, 427, etc)
john --mask='?d?d?d' num_3_hash_md5.txt

# Para estos tres datasets simplemente se añaden ?x segun la longitud de las contraseñas con el formato mencionado en cada caso.

# DATASET DE SIMBOLOS : ?a indica un caracter ASCII, si ponemos tres simboliza una contraseña de longitud 3 (1ae, 8!<, etc)
john --mask='?a?a?a' ans_3_hash_md5.txt

# DATASETS DE DICCIONARIOS : igual que en el caso anterior pero sin definir longitud porque es deconocido
john --mask='?a' --min-length=4 dic_1_hash_md5.txt
```

Método 3. Ataque por diccionario

Como hemos comentado anteriormente, en JtR podemos hacer ataques por diccionario en el que tomamos un diccionario de grandes dimensiones y atacamos los hashes. Si la contraseña

original estaba en la base de datos será descifrada. Esto lo utilizamos escribiendo el comando de la siguiente manera:

```
Unset
john --wordlist=WORDLIST archivo_hash.txt
```

Estos diccionarios en la vida real podrían ser bases de datos filtradas y por lo general hay muchas personas con contraseñas similares. Algunas contraseñas muy comunes que suelen estar en estas bases de datos son: 123456, password, qwerty, 1234, 111111, abc123, letmein…

Ejemplos de escritura de las líneas de comando de cada dataset:

```
Unset
# DATASET DE MINUSCULAS :
john --wordlist=wordlist_1.txt minus_3_hash_md5.txt

# DATASET DE MAYUSCULAS :
john --wordlist=wordlist_1.txt mayus_3_hash_md5.txt

# DATASET DE NUMEROS :
john --wordlist=wordlist_2.txt num_3_hash_md5.txt

# DATASET DE SIMBOLOS
john --wordlist=wordlist_2.txt ans_3_hash_md5.txt

# DATASETS DE DICCIONARIOS :
john -wordlist=rockyou-75.txt dic_1_hash_md5.txt
```

Método 4. Ataque por diccionario con reglas por defecto

JtR permite añadir reglas definidas en el archivo de configuración para leer de forma más eficiente los diccionarios siguiendo el siguiente esquema:

```
Unset
john --rules=RULE --wordlist=WORDLIST archivo_hash.txt
```

RULE hace referencia a alguna regla que está definida en el archivo de configuración (john.conf). Las reglas por defecto más destacadas que ofrece JtR son:

- Single: Esto es utilizado cuando se sabe algo del propietario de la contraseña. Por ejemplo si se trata de una contraseña de administrador, esta regla podría encontrar fácilmente la contraseña siempre que tenga algo que ver con el rol como admin, admin1234, 123admin, Admin, administrador, etc

- Wordlist: Este ruleset va intentando hacer varias reglas consecutivas. Algunas de ellas añaden el número uno al final de la palabra, otras hacen que siga patrones de gramática en inglés (por ejemplo sleep la transformaría a sleeping), …
- Extra: Este ruleset hace cosas parecidas. Por ejemplo algunas reglas tratan de hacer Toggle de las letras de las palabras (si tenemos sol, intentaría sol, los, ols, osl, lso, etc)
- Jumbo: Hace las reglas de single + wordlist + extra
- KoreLogic: Este ruleset incluye algunos métodos conocidos que emplea esta empresa de seguridad al realizar ataques de penetración.
- All: Realiza todas las anteriores.

Para hacer la práctica vemos que tiene más sentido hacerlo usando todas las reglas de default ya que las reglas por defecto no están adaptadas a nuestra generación de contraseñas y es muy improbable que consigan averiguar alguna de las contraseñas individualmente. En los dataset que hemos generado, tienen mayor probabilidad de éxito los creados a partir del diccionario ya que son las únicas que siguen un sentido lógico al ser contraseñas de personas reales y los números ya que la fuerza bruta actúa eficazmente en esta clase de carácter porque existen menos combinaciones por longitud de palabra a descifrar.

Un ejemplo de como lo hemos usado con cada dataset se detalla a continuación:

```Python
# REGLAS DATASET DE MINUSCULAS :
john --rules=all --wordlist=wordlist_1.txt minus_3_hash_md5.txt

# REGLAS DATASET DE MAYUSCULAS :
john --rules=all --wordlist=wordlist_1.txt mayus_3_hash_md5.txt

# REGLAS DATASET DE NUMEROS :
john --rules=all --wordlist=wordlist_2.txt num_3_hash_md5.txt

# REGLAS DATASET DE SIMBOLOS
john --rules=all --wordlist=wordlist_2.txt ans_3_hash_md5.txt

# REGLAS DATASETS DE DICCIONARIOS :
john --rules=all --wordlist=rockyou-75.txt dic_1_hash_md5.txt
```

Método 5.Ataque por diccionario con reglas específicas

La quinta metodología que hemos empleado hace uso de diccionarios pero con el uso de reglas de creación propia en vez de por defecto, escribiendo el comando de la siguiente manera:

```Unset
john --rules=RULE --wordlist=rockyou-75.txt archivo_hash.txt
```

RULE hace referencia a alguna de las reglas que hemos diseñado y añadido al archivo de configuración. Como tenemos varios tipos de dataset hemos creado las siguientes reglas adaptadas en cierta medida al dataset correspondiente.

Las reglas se han creado teniendo en cuenta que:

- l: Convierte las letras en minúsculas.
- 'N: Trunca las palabras a una longitud de N.
- u: Convierte las letras en mayúsculas.
- !X: Rechaza la palabra si contiene X.
- : : Ninguna operación: No hace nada con las palabras entrantes.
- t: Alternar entre mayúsculas y minúsculas todos los caracteres de las palabras.
- f: Reflejar: "Estuche" -> "EstucheehcutsE".
- sXY: Reemplaza todos los caracteres X de las palabras por Y.
- r: Inverso: "Estuche" -> "ehcutsE"
- /?C: rechaza las palabra a menos que contenga un carácter de clase C

[List.Rules:Minus3]	l ’3	Convierte las letras de las palabras del diccionario a minúsculas [a-z] y trunca la palabra en la posición 3.
[List.Rules:Minus4]	l ’4	Convierte las letras de las palabras del diccionario a minúsculas y trunca la palabra en la posición 4.
[List.Rules:Minus5]	l ’5	Convierte las letras de las palabras del diccionario a minúsculas y trunca la palabra en la posición 5.
[List.Rules:Minus6]	l ’6	Convierte las letras de las palabras del diccionario a minúsculas y trunca la palabra en la posición 6.
[List.Rules:Minus7]	l ’7	Convierte las letras de las palabras del diccionario a minúsculas y trunca la palabra en la posición 7.
[List.Rules:Mayus3]	u ‘3	Convierte las letras de las palabras del diccionario a mayúsculas [A-Z] y trunca la palabra en la posición 3.
[List.Rules:Mayus4]	u ‘4	Convierte las letras de las palabras del diccionario a mayúsculas y trunca la palabra en la posición 4.
[List.Rules:Mayus5]	u ‘5	Convierte las letras de las palabras del diccionario a mayúsculas y trunca la palabra en la posición 5.
[List.Rules:Mayus6]	u ‘6	Convierte las letras de las palabras del diccionario a mayúsculas y trunca la palabra en la posición 6.
[List.Rules:Mayus7]	u ‘7	Convierte las letras de las palabras del diccionario a mayúsculas y trunca la palabra en la posición 7.
[List.Rules:Num3]	/?d ‘3	Excluye toda palabra que no esté compuesta

		únicamente de números. Si la acepta la trunca en la posición 3.
[List.Rules:Num4]	/?d ‘4	Excluye toda palabra que no esté compuesta únicamente de números. Si la acepta la trunca en la posición 4.
[List.Rules:Num5]	/?d ‘5	Excluye toda palabra que no esté compuesta únicamente de números. Si la acepta la trunca en la posición 5.
[List.Rules:Num6]	/?d ‘6	Excluye toda palabra que no esté compuesta únicamente de números. Si la acepta la trunca en la posición 6.
[List.Rules:Num7]	/?d ‘7	Excluye toda palabra que no esté compuesta únicamente de números. Si la acepta la trunca en la posición 7.
[List.Rules:Ans3]	‘3	Trunca la palabra en la posición 3 ya que el diccionario que le vamos a pasar tiene palabras de longitud 7.
[List.Rules:Ans4]	‘4	Trunca la palabra en la posición 4 ya que el diccionario que le vamos a pasar tiene palabras de longitud 7.
[List.Rules:Ans5]	‘5	Trunca la palabra en la posición 5 ya que el diccionario que le vamos a pasar tiene palabras de longitud 7.
[List.Rules:Ans6]	‘6	Trunca la palabra en la posición 6 ya que el diccionario que le vamos a pasar tiene palabras de longitud 7.
[List.Rules:Ans7]	‘7	Trunca la palabra en la posición 7. En realidad esta regla no hace nada porque todas las palabras del diccionario tienen esta longitud.
[List.Rules:Dic1]	:	No hace nada con el input ya que las palabras a descifrar están en el diccionario de contraseñas más usadas.
[List.Rules:Dic2]	t	Cambia entre minúsculas y mayúsculas todas las letras de la palabra (Toggle) para ajustarse al patrón de creación del diccionario 2.
[List.Rules:Dic3]	f	Le suma a la palabra su reflejo ya que en el

		diccionario estábamos sumando a las palabras su palíndroma.
[List.Rules:Dic4]	so0 si1 se3 sa4 ss5 st7 sb8	Reemplaza los caracteres de la siguiente forma: • o → 0 • i → 1 • e → 3 • a → 4 • s → 5 • t → 7 • b → 8 ya que así era el patrón de creación del diccionario 4.
[List.Rules:Dic5]	r	Pone la palabra al revés ya que así se ha creado el diccionario 5.

Un ejemplo de como lo hemos usado con cada dataset se detalla a continuación:

```python
# REGLAS DATASET DE MINUSCULAS :
john --rules=Minus3 --wordlist=wordlist_1.txt minus_3_hash_md5.txt

# REGLAS DATASET DE MAYUSCULAS :
john --rules=Mayus3 --wordlist=wordlist_1.txt mayus_3_hash_md5.txt

# REGLAS DATASET DE NUMEROS :
john --rules=Num3 --wordlist=wordlist_2.txt num_3_hash_md5.txt

# REGLAS DATASET DE SIMBOLOS
john --rules=Ans3 --wordlist=wordlist_2.txt ans_3_hash_md5.txt

# REGLAS DATASETS DE DICCIONARIOS :
john --rules=Dic1 --wordlist=rockyou-75.txt dic_1_hash_md5.txt
```

Este ataque es útil cuando el diccionario del que se dispone es muy extenso y con muchos tipos de contraseñas. Esto puede pasar por ejemplo cuando se filtra una base de datos de un lugar con especificaciones de contraseñas muy generales y alguien la usa para atacar en otro lugar con mayor especificación de formato por lo que se perdería mucho tiempo si no se filtran primero las contraseñas.

Resultados y Análisis

Hemos ejecutado cada una de las estrategias diseñadas en el paso 2 contra cada uno de los datasets generados en el paso 1 siguiendo los ejemplos descritos en el apartado anterior. Dado que algunos datasets necesitan un tiempo muy elevado requerido para romper sus contraseñas,

hemos dado un tiempo limitado a John the Ripper de una hora y media (90 minutos) tras el cual paramos forzosamente el ataque si este no había acabado.

Una vez finalizado el proceso para cada dataset, hemos obtenido distintos resultados. Para verlos en el documento hemos ideado una tabla con las columnas:

- Porcentaje: Porcentaje de contraseñas rotas.
- Media: Media del tiempo requerido para romper una contraseña.
- Mediana: Mediana del tiempo requerido para romper una contraseña.
- Tiempo total: Tiempo de ejecución total. Aparece con el formato d:hh:mm:ss. Los que están en 1:30:00 son los que han sido abortado tras traspasar el límite de tiempo (1:30h).

Para ver y analizar los métodos y la seguridad de cada tipo de dataset vamos a organizar los resultados en dos tipos de tablas. Las primeras van organizar los datos por métodos y las segundas por dataset. En la segunda forma de tabular los resultados en los dataset generados por longitud (minus, mayús, numy ans) solo vamos a colocar la longitud mínima y la máxima para así comprobar cómo influye la longitud de la contraseña en la seguridad pero colocaremos todos los resultados de los diccionarios ya que cada uno sigue una metodología de generación.

Resultados método 1. Fuerza bruta

	Porcentaje	Media	Mediana	Tiempo total
minus-3	100%	0:00:00	0:00:00	0:00:12
minus-4	100%	0:00:03	0:00:02	0:04:51
minus-5	31%	0:02:52	0:01:44	1:30:00
minus-6	2%	0:16:57	0:16:57	1:30:00
minus-7	0%	-	-	1:30:00
mayus-3	100%	0:00:00	0:00:00	0:00:15
mayus-4	100%	0:00:05	0:00:03	0:08:14
mayus-5	33%	0:02:43	0:02:05	1:30:00
mayus-6	3%	0:29:08	0:37:05	1:30:00
mayus-7	0%	-	-	1:30:00
num-3	100%	0:00:00	0:00:00	0:00:00
num-4	100%	0:00:00	0:00:00	0:00:06
num-5	100%	0:00:01	0:00:00	0:01:01
num-6	100%	0:00:07	0:00:03	0:11:06

num-7	57%	0:01:34	0:00:10	1:30:00
ans-3	100%	0:00:07	0:00:04	0:11:49
ans-4	1%	0:16:24	0:16:24	1:30:00
ans-5	0%	-	-	1:30:00
ans-6	0%	-	-	1:30:00
ans-7	0%	-	-	1:30:00
dic-1	30%	0:03:04	0:00:36	1:30:00
dic-2	20%	0:01:49	0:00:23	1:30:00
dic-3	0%	-	-	1:30:00
dic-4	0%	-	-	1:30:00
dic-5	0%	-	-	1:30:00

Resultados método 2. Fuerza bruta con máscaras

	Porcentaje	**Media**	**Mediana**	**Tiempo**
minus-3	100%	0:00:00	0:00:00	0:00:10
minus-4	100%	0:00:04	0:00:02	0:06:07
minus-5	100%	0:00:59	0:00:30	1:37:39
minus-6	3%	0:17:13	0:17:13	1:30:00
minus-7	0%	-	-	1:30:00
mayus-3	100%	0:00:00	0:00:00	0:00:07
mayus-4	100%	0:00:02	0:00:01	0:02:55
mayus-5	100%	0:00:50	0:00:31	1:23:20
mayus-6	1%	0:29:44	0:29:44	1:30:00
mayus-7	0%	-	-	1:30:00
num-3	100%	0:00:00	0:00:00	0:00:00

num-4	100%	0:00:00	0:00:00	0:00:04
num-5	100%	0:00:00	0:00:00	0:00:36
num-6	100%	0:00:04	0:00:02	0:06:50
num-7	100%	0:01:00	0:00:30	1:40:34
ans-3	100%	0:00:04	0:00:02	0:06:42
ans-4	1%	0:46:56	0:46:56	1:30:03
ans-5	0%	-	-	1:30:00
ans-6	0%	-	-	1:30:00
ans-7	0%	-	-	1:30:00
dic-1	0%	-	-	1:30:00
dic-2	0%	-	-	1:30:00
dic-3	0%	-	-	1:30:00
dic-4	0%	-	-	1:30:00
dic-5	0%	-	-	1:30:00

Resultados método 3. Ataque con diccionarios

	Porcentaje	**Media**	**Mediana**	**Tiempo**
minus-3	0%	-	-	0:01:39
minus-4	0%	-	-	0:01:35
minus-5	0%	-	-	0:01:32
minus-6	0%	-	-	0:01:32
minus-7	0%	-	-	0:01:36
mayus-3	0%	-	-	0:01:52
mayus-4	0%	-	-	0:01:29
mayus-5	0%	-	-	0:01:45

mayus-6	0%	-	-	0:01:39
mayus-7	0%	-	-	0:01:42
num-3	0%	-	-	0:01:31
num-4	0%	-	-	0:01:34
num-5	0%	-	-	0:01:38
num-6	0%	-	-	0:01:30
num-7	0%	-	-	0:01:40
ans-3	0%	-	-	0:01:47
ans-4	0%	-	-	0:01:34
ans-5	0%	-	-	0:02:09
ans-6	0%	-	-	0:01:34
ans-7	0%	-	-	0:02:00
dic-1	100%	0:00:00	0:00:00	0:00:39
dic-2	27%	0:00:02	0:00:02	0:01:08
dic-3	0%	-	-	0:01:32
dic-4	16%	0:00:05	0:00:04	0:01:18
dic-5	4%	0:00:16	0:00:16	0:01:21

Resultados método 4. Ataque con diccionarios + Reglas por defecto (all)

	Porcentaje	**Media**	**Mediana**	**Tiempo**
minus-3	100%	0:00:11	0:00:00	0:18:47
minus-4	38%	0:01:05	0:00:04	1:30:00
minus-5	5%	0:13:02	0:00:06	1:30:00
minus-6	0%	-	-	1:30:00
minus-7	0%	-	-	1:30:00

mayus-3	81%	0:00:21	0:00:01	1:30:00
mayus-4	1%	00:32:08	00:32:08	1:30:00
mayus-5	0%	-	-	1:30:00
mayus-6	0%	-	-	1:30:00
mayus-7	0%	-	-	1:30:00
num-3	100%	0:00:14	0:00:00	0:24:04
num-4	100%	0:10:43	0:00:00	0:44:52
num-5	100%	0:22:06	0:00:00	1:21:33
num-6	0%	-	-	1:30:00
num-7	0%	-	-	1:30:00
ans-3	27%	0:01:32	0:00:07	1:30:00
ans-4	0%	-	-	1:30:00
ans-5	0%	-	-	1:30:00
ans-6	0%	-	-	1:30:00
ans-7	0%	-	-	1:30:00
dic-1	100%	0:00:00	0:00:00	0:00:29
dic-2	100%	0:00:09	0:00:00	0:14:29
dic-3	69%	0:00:17	0:00:00	0:01:30
dic-4	22%	0:00:02	0:00:01	0:01:30
dic-5	100%	0:00:06	0:00:00	0:09:44

Resultados método 5. Ataque con diccionarios + Reglas propias

	Porcentaje	**Media**	**Mediana**	**Tiempo**
minus-3	100%	0:00:00	0:00:00	0:00:23
minus-4	11%	0:00:06	0:00:05	0:01:15

minus-5	0%	-	-	0:01:22
minus-6	0%	-	-	0:01:19
minus-7	0%	-	-	0:01:22
mayus-3	96%	0:00:00	0:00:00	0:00:29
mayus-4	6%	0:00:16	0:00:11	0:01:39
mayus-5	1%	0:00:17	0:00:17	0:01:28
mayus-6	0%	-	-	0:01:23
mayus-7	0%	-	-	0:01:28
num-3	100%	0:00:00	0:00:00	0:00:02
num-4	90%	0:00:00	0:00:00	0:00:17
num-5	21%	0:0:01	0:00:01	0:00:46
num-6	2%	0:00:3	0:00:3	0:01:02
num-7	0%	-	-	0:01:08
ans-3	3%	0:00:24	0:00:14	0:01:31
ans-4	0%	-	-	0:01:20
ans-5	0%	-	-	0:01:45
ans-6	0%	-	-	0:01:34
ans-7	0%	-	-	0:01:31
dic-1	100%	0:00:00	0:00:00	0:00:46
dic-2	100%	0:00:00	0:00:00	0:00:41
dic-3	100%	0:00:00	0:00:00	0:00:41
dic-4	23%	0:00:14	0:00:04	0:08:56
dic-5	100%	0:00:00	0:00:00	00:46

Resultados por datasets

Resultados minus-3

	Porcentaje	Media	Mediana	Tiempo
M1	100%	0:00:00	0:00:00	0:00:12
M2	100%	0:00:00	0:00:00	0:00:10
M3	0%	-	-	0:01:39
M4	100%	0:00:11	0:00:00	0:18:47
M5	100%	0:00:00	0:00:00	0:00:23

Resultados minus-7

	Porcentaje	Media	Mediana	Tiempo
M1	0%	-	-	1:30:00
M2	0%	-	-	1:30:00
M3	0%	-	-	0:01:36
M4	0%	-	-	1:30:00
M5	0%	-	-	0:01:22

Resultados mayus-3

	Porcentaje	Media	Mediana	Tiempo
M1	100%	0:00:00	0:00:00	0:00:15
M2	100%	0:00:00	0:00:00	0:00:07
M3	0%	-	-	0:01:52
M4	81%	0:00:21	0:00:01	1:30:00
M5	96%	0:00:00	0:00:00	0:00:29

Resultados mayus-7

	Porcentaje	Media	Mediana	Tiempo
M1	0%	-	-	1:30:00
M2	0%	-	-	1:30:00
M3	0%	-	-	0:01:42
M4	0%	-	-	1:30:00
M5	0%	-	-	0:01:28

Resultados num-3

	Porcentaje	Media	Mediana	Tiempo
M1	100%	0:00:00	0:00:00	0:00:00
M2	100%	0:00:00	0:00:00	0:00:00
M3	0%	-	-	0:01:31
M4	100%	0:00:14	0:00:00	0:24:04
M5	100%	0:00:00	0:00:00	0:00:02

Resultados num-7

	Porcentaje	Media	Mediana	Tiempo
M1	57%	0:01:34	0:00:10	1:30:00
M2	100%	0:01:00	0:00:30	1:40:34
M3	0%	-	-	0:01:40
M4	0%	-	-	1:30:00
M5	0%	-	-	0:01:08

Resultados ans-3

	Porcentaje	Media	Mediana	Tiempo

M1	100%	0:00:07	0:00:04	0:11:49
M2	100%	0:00:04	0:00:02	0:06:42
M3	0%	-	-	0:01:47
M4	27%	0:01:32	0:00:07	1:30:00
M5	3%	0:00:24	0:00:14	0:01:31

Resultados ans-7

	Porcentaje	Media	Mediana	Tiempo
M1	0%	-	-	1:30:00
M2	0%	-	-	1:30:00
M3	0%	-	-	0:02:00
M4	0%	-	-	1:30:00
M5	0%	-	-	0:01:31

Resultados dic-1

	Porcentaje	Media	Mediana	Tiempo
M1	30%	0:03:04	0:00:36	1:35:00
M2	0%	-	-	1:30:00
M3	100%	0:00:00	0:00:00	0:00:39
M4	100%	0:00:00	0:00:00	0:00:29
M5	100%	0:00:00	0:00:00	0:00:46

Resultados dic-2

	Porcentaje	Media	Mediana	Tiempo
M1	20%	0:01:49	0:00:23	1:30:00
M2	0%	-	-	1:30:00

M3	27%	0:00:02	0:00:02	0:01:08
M4	100%	0:00:09	0:00:00	0:14:29
M5	100%	0:00:00	0:00:00	0:00:41

Resultados dic-3

	Porcentaje	Media	Mediana	Tiempo
M1	0%	-	-	1:30:00
M2	0%	-	-	1:30:00
M3	0%	-	-	0:01:32
M4	22%	0:00:02	0:00:01	0:01:30
M5	100%	0:00:00	0:00:00	0:00:41

Resultados dic-4

	Porcentaje	Media	Mediana	Tiempo
M1	0%	-	-	1:30:00
M2	0%	-	-	1:30:00
M3	16%	0:00:05	0:00:04	0:01:18
M4	22%	0:00:02	0:00:01	0:01:30
M5	23%	0:00:14	0:00:04	0:08:56

Resultados dic-5

	Porcentaje	Media	Mediana	Tiempo
M1	0%	-	-	1:30:00
M2	0%	-	-	1:30:00
M3	4%	0:00:16	0:00:16	0:01:21
M4	100%	0:00:06	0:00:00	0:00:09

M5	100%	0:00:00	0:00:00	0:46:00

Resultados por limite de tiempo

Resumen de ataques que no han alcanzado un éxito del 100% dentro del tiempo límite:

Método	Dataset	Porcentaje	Media	Mediana	Tiempo
M3	minus-3	0%	-	-	0:01:39
M3	minus-4	0%	-	-	0:01:35
M3	minus-5	0%	-	-	0:01:32
M3	minus-6	0%	-	-	0:01:32
M3	minus-7	0%	-	-	0:01:36
M3	mayus-3	0%	-	-	0:01:52
M3	mayus-4	0%	-	-	0:01:29
M3	mayus-5	0%	-	-	0:01:45
M3	mayus-6	0%	-	-	0:01:39
M3	mayus-7	0%	-	-	0:01:42
M3	num-3	0%	-	-	0:01:31
M3	num-4	0%	-	-	0:01:34
M3	num-5	0%	-	-	0:01:38
M3	num-6	0%	-	-	0:01:30
M3	num-7	0%	-	-	0:01:40
M3	ans-3	0%	-	-	0:01:47
M3	ans-4	0%	-	-	0:01:34
M3	ans-5	0%	-	-	0:02:09
M3	ans-6	0%	-	-	0:01:34
M3	ans-7	0%	-	-	0:02:00

M3	dic-2	27%	0:00:02	0:00:02	0:01:08
M3	dic-3	0%	-	-	0:01:32
M3	dic-4	16%	0:00:05	0:00:04	0:01:18
M3	dic-5	4%	0:00:16	0:00:16	0:01:21
M5	minus-4	11%	0:00:06	0:00:05	0:01:15
M5	minus-5	0%	-	-	0:01:22
M5	minus-6	0%	-	-	0:01:19
M5	minus-7	0%	-	-	0:01:22
M5	mayus-3	96%	0:00:00	0:00:00	0:00:29
M5	mayus-4	6%	0:00:16	0:00:11	0:01:39
M5	mayus-5	1%	0:00:17	0:00:17	0:01:28
M5	mayus-6	0%	-	-	0:01:23
M5	mayus-7	0%	-	-	0:01:28
M5	num-4	90%	0:00:00	0:00:00	0:00:17
M5	num-5	21%	0:00:01	0:00:01	0:00:46
M5	num-6	2%	0:00:03	0:00:03	0:01:02
M5	num-7	0%	-	-	0:01:08
M5	ans-3	3%	0:00:24	0:00:14	0:01:31
M5	ans-4	0%	-	-	0:01:20
M5	ans-5	0%	-	-	0:01:45
M5	ans-6	0%	-	-	0:01:34
M5	ans-7	0%	-	-	0:01:31
M5	dic-4	23%	0:00:14	0:00:04	0:08:56

Resumen de ataques que no han finalizado por alcanzar el tiempo límite (abortados):

Método	Dataset	Porcentaje	Media	Mediana	Tiempo
M1	minus-5	31%	0:02:52	0:01:44	1:30:00
M1	minus-6	2%	0:16:57	0:16:57	1:30:00
M1	minus-7	0%	-	-	1:30:00
M1	mayus-5	33%	0:02:43	0:02:05	1:30:00
M1	mayus-6	3%	0:29:08	0:37:05	1:30:00
M1	mayus-7	0%	-	-	1:30:00
M1	num-7	57%	0:01:34	0:00:10	1:30:00
M1	ans-4	1%	0:16:24	0:16:24	1:30:00
M1	ans-5	0%	-	-	1:30:00
M1	ans-6	0%	-	-	1:30:00
M1	ans-7	0%	-	-	1:30:00
M1	dic-1	30%	0:03:04	0:00:36	1:35:00
M1	dic-2	20%	0:01:49	0:00:23	1:30:00
M1	dic-3	0%	-	-	1:30:00
M1	dic-4	0%	-	-	1:30:00
M1	dic-5	0%	-	-	1:30:00
M2	minus-6	3%	0:17:13	0:17:13	1:30:00
M2	minus-7	0%	-	-	1:30:00
M2	mayus-6	1%	0:29:44	0:29:44	1:30:00
M2	mayus-7	0%	-	-	1:30:00
M2	ans-4	1%	0:46:56	0:46:56	1:30:03
M2	ans-5	0%	-	-	1:30:00
M2	ans-6	0%	-	-	1:30:00

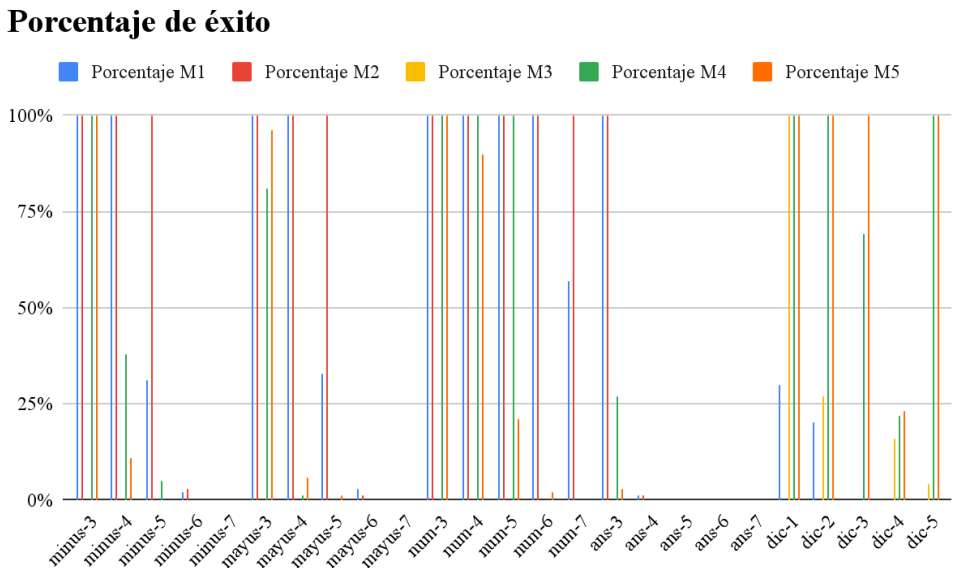
M2	ans-7	0%	-	-	1:30:00
M2	dic-1	0%	-	-	1:30:00
M2	dic-2	0%	-	-	1:30:00
M2	dic-3	0%	-	-	1:30:00
M2	dic-4	0%	-	-	1:30:00
M2	dic-5	0%	-	-	1:30:00
M4	minus-4	38%	0:01:05	0:00:04	1:30:00
M4	minus-5	5%	0:13:02	0:00:06	1:30:00
M4	minus-6	0%	-	-	1:30:00
M4	minus-7	0%	-	-	1:30:00
M4	mayus-3	81%	0:00:21	0:00:01	1:30:00
M4	mayus-4	1%	0:32:08	0:32:08	1:30:00
M4	mayus-5	0%	-	-	1:30:00
M4	mayus-6	0%	-	-	1:30:00
M4	mayus-7	0%	-	-	1:30:00
M4	num-6	0%	-	-	1:30:00
M4	num-7	0%	-	-	1:30:00
M4	ans-3	27%	0:01:32	0:00:07	1:30:00
M4	ans-4	0%	-	-	1:30:00
M4	ans-5	0%	-	-	1:30:00
M4	ans-6	0%	-	-	1:30:00
M4	ans-7	0%	-	-	1:30:00
M4	dic-3	69%	0:00:17	0:00:00	1:30:00
M4	dic-4	22%	0:00:02	0:00:01	1:30:00

Análisis de los resultados

Para facilitar el análisis vamos a representar gráficamente y analizar cada una de las métricas.

Nota: Para hacer los gráficos de media y mediana hemos puesto un límite de tiempo del eje Y de 2 horas con fines estéticos pero hay casos que necesitan más de este tiempo por contraseña.

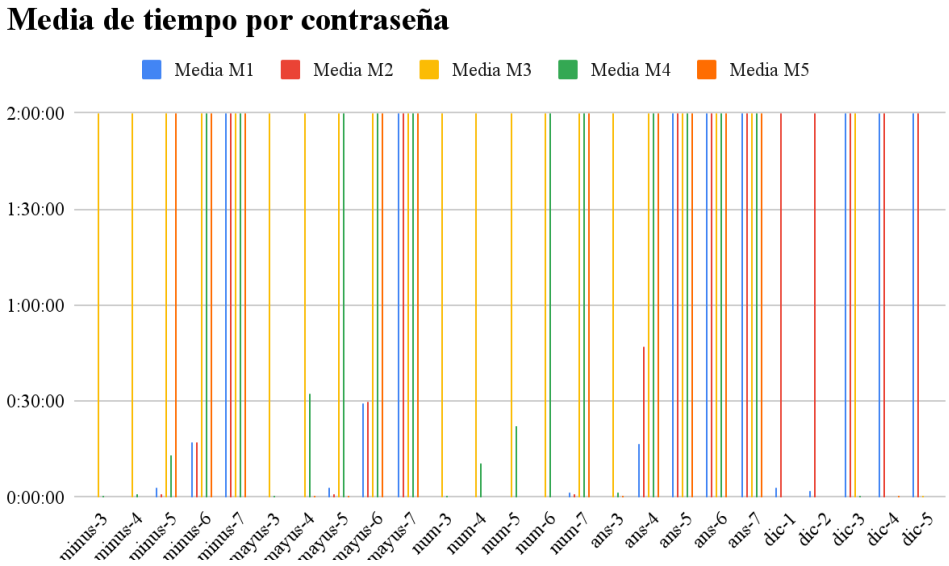
Análisis del éxito



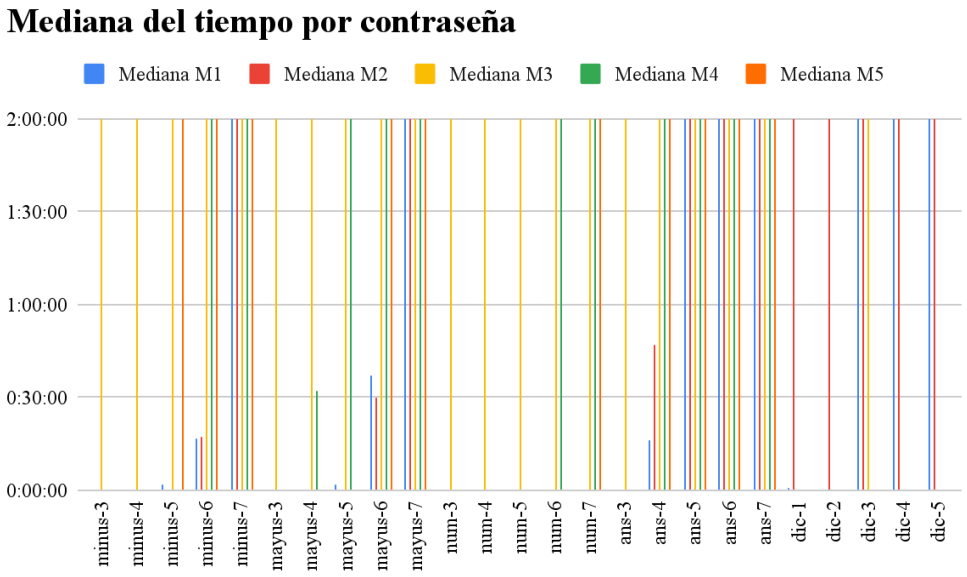
Analizando el porcentaje de éxito de los métodos podemos observar que:

- Los ataques de fuerza bruta siempre consiguen descifrar la contraseña dentro de un tiempo límite si es de una longitud pequeña.
- No hay diferencias notables de ataques por fuerza bruta con máscara de sin máscara pero en ocasiones la máscara es más rápida.
- El ataque por diccionario de contraseñas normales (palabras o/y números) es rápido y eficaz si se posee un diccionario extenso.
- Si la contraseña es corta y no sigue un patrón lógico, gramatical o un patrón personal el ataque de fuerza bruta sería el más apropiado ya que siempre consigue descifrarlas.
- Si la contraseña es alguna palabra clave o conocida el ataque por diccionarios es más eficiente ya que puede descifrarlas rápidamente a pesar de tener una longitud mayor.
- Cuanto más tipos de caracteres, mayor longitud y mayor creatividad posea la contraseña más complicado será conseguirla tanto por fuerza bruta como por diccionario debido a su originalidad.
- Las contraseñas formadas únicamente por números son las más vulnerables ya que aunque sean medianamente largas (longitud de +6) sigue teniendo un porcentaje de éxito de descifrado mayor al resto.

Análisis de la media y la mediana



En cuanto a la media, podemos observar claramente que el método con la media de tiempo por contraseña más alta es el método 3 ataque por diccionario que no utiliza ningún tipo de algoritmo, lo que nos indica que es el que más ha tardado en crackear las contraseñas. Seguido de este se encuentra el método 5 que es notable que al usar los datasets de los diccionarios la media se dispara. Además se puede notar como los tiempos de los ataques de fuerza bruta crecen de manera exponencial.



En cuanto a la mediana sigue en general la misma línea de resultados que la media.

En conjunto sacamos como conclusión que los ataques de fuerza bruta aunque efectivos necesitan mucho tiempo para obtener contraseñas normales como abcd123 por su longitud. Si a

eso le añades complejidad utilizando símbolos el tiempo necesario para descifrarla hace prácticamente imposible utilizar este método para descifrar grandes cantidades de contraseñas.

Análisis de métodos de ataque

Como podemos ver el ataque por diccionarios si sigue algoritmos simples puede ser muy eficaz contra la mayoría de contraseñas (que componen el dic-1), pero si se complican juntando muchas reglas de generación pueden llevar mucho tiempo (método 4) y no siempre llevan al éxito.

El ataque por fuerza bruta siempre con disponibilidad de suficiente tiempo puede llegar a alcanzar la solución pero si hay muchos caracteres que hacen infinitas combinaciones, en la realidad este no es un método factible en la mayoría de los casos actuales.

Conclusiones

A la hora de crear contraseñas la gente no suele dedicarle mucho tiempo a pensarlas lo que puede ser un grave error. Suelen optar por contraseñas básicas y fáciles de recordar pensando en que no van a ser a ellos a quienes les ocurra algo. Esto conlleva un gran riesgo ya que hoy en día existe un aumento de ciberdelincuencia que lo que hacen es buscar robar información para después extorsionar a estas personas.

Gracias a esta práctica hemos podido comprender cómo es el proceso de crackeo de contraseñas. Para ellos hemos usado la herramienta John The Ripper, una herramienta para crackear contraseñas. Con ella hemos aprendido que las contraseñas más seguras son aquellas con una longitud mayor y fuera de lo común. Hemos podido observar como las contraseñas más usadas y con poca variedad de caracteres son las más débiles y vulnerables y como con los diferentes métodos se pueden tardar incluso solo segundos en descifrar.

Tener conocimiento sobre todo el proceso de crackeo de contraseñas es muy importante para mejorar nuestra seguridad. Al escribir contraseñas más fuertes y seguras conseguimos proteger nuestros datos personales y profesionales de ser robados por ciberdelincuentes. Se debería concienciar más a las personas de las amenazas que pueden sufrir y así poder prevenir y proteger su información.

Referencias

Akimbo Core. (s/f). Akimbocore.com. Recuperado el 21 de octubre de 2023, de https://akimbocore.com/article/custom-rules-for-john-the-ripper/

Comprehensive Guide to John the Ripper. Part 3: How to start cracking passwords in John the Ripper (how to specify masks, dictionaries, hashes, formats, modes) - Ethical hacking and penetration testing. (s/f). Miloserdov.org. Recuperado el 21 de octubre de 2023, de https://miloserdov.org/?p=5031

Comprehensive Guide to John the Ripper. Part 5: Rule-based attack - Ethical hacking and penetration testing. (s/f). Miloserdov.org. Recuperado el 20 de octubre de 2023, de https://miloserdov.org/?p=5477

Gómez, F. P. (s/f-a). *Crear nuevas reglas*. Fpgenred.es. Recuperado el 21 de octubre de 2023, de https://www.fpgenred.es/Seguridad-Informatica-I/crear_nuevas_reglas.html

Gómez, F. P. (s/f-b). *Más sobre John The Ripper*. Fpgenred.es. Recuperado el 21 de octubre de 2023, de https://www.fpgenred.es/Seguridad-Informatica-I/ms_sobre_john_the_ripper.html

John the Ripper. (s/f). Tue.nl. Recuperado el 20 de octubre de 2023, de https://www.win.tue.nl/~aeb/linux/john/john.html

John the Ripper - command line options. (s/f). Openwall.com. Recuperado el 20 de octubre de 2023, de https://www.openwall.com/john/doc/OPTIONS.shtml

John the Ripper - cracking modes. (s/f). Openwall.com. Recuperado el 20 de octubre de 2023, de https://www.openwall.com/john/doc/MODES.shtml

John the Ripper - defining an external mode. (s/f). Openwall.com. Recuperado el 20 de octubre de 2023, de https://www.openwall.com/john/doc/EXTERNAL.shtml

John the Ripper - how to customize the configuration file. (s/f). Openwall.com. Recuperado el 20 de octubre de 2023, de https://www.openwall.com/john/doc/CONFIG.shtml

John the Ripper - usage examples. (s/f). Openwall.com. Recuperado el 20 de octubre de 2023, de https://www.openwall.com/john/doc/EXAMPLES.shtml

John the ripper/password generation - charlesreid1. (s/f). Charlesreid1.com. Recuperado el 21 de octubre de 2023, de https://charlesreid1.com/wiki/John_the_Ripper/Password_Generation

Rocha, L. (s/f). *JTR CHEAT SHEET*. Wordpress.com. Recuperado el 20 de octubre de 2023, de https://countuponsecurity.files.wordpress.com/2016/09/jtr-cheat-sheet.pdf

Salmerón, G. M. (s/f). *TFG - https://oa.upm.es/68583/1/TFG_GONZALO_MARTINEZ_SALMERON.pdf*. Upm.es.

Recuperado el 20 de octubre de 2023, de https://oa.upm.es/68583/1/TFG_GONZALO_MARTINEZ_SALMERON.pdf

What exactly is "single" mode in John the Ripper doing? (s/f). Information Security Stack Exchange. Recuperado el 21 de octubre de 2023, de https://security.stackexchange.com/questions/37072/what-exactly-is-single-mode-in-john-the-ripper-doing