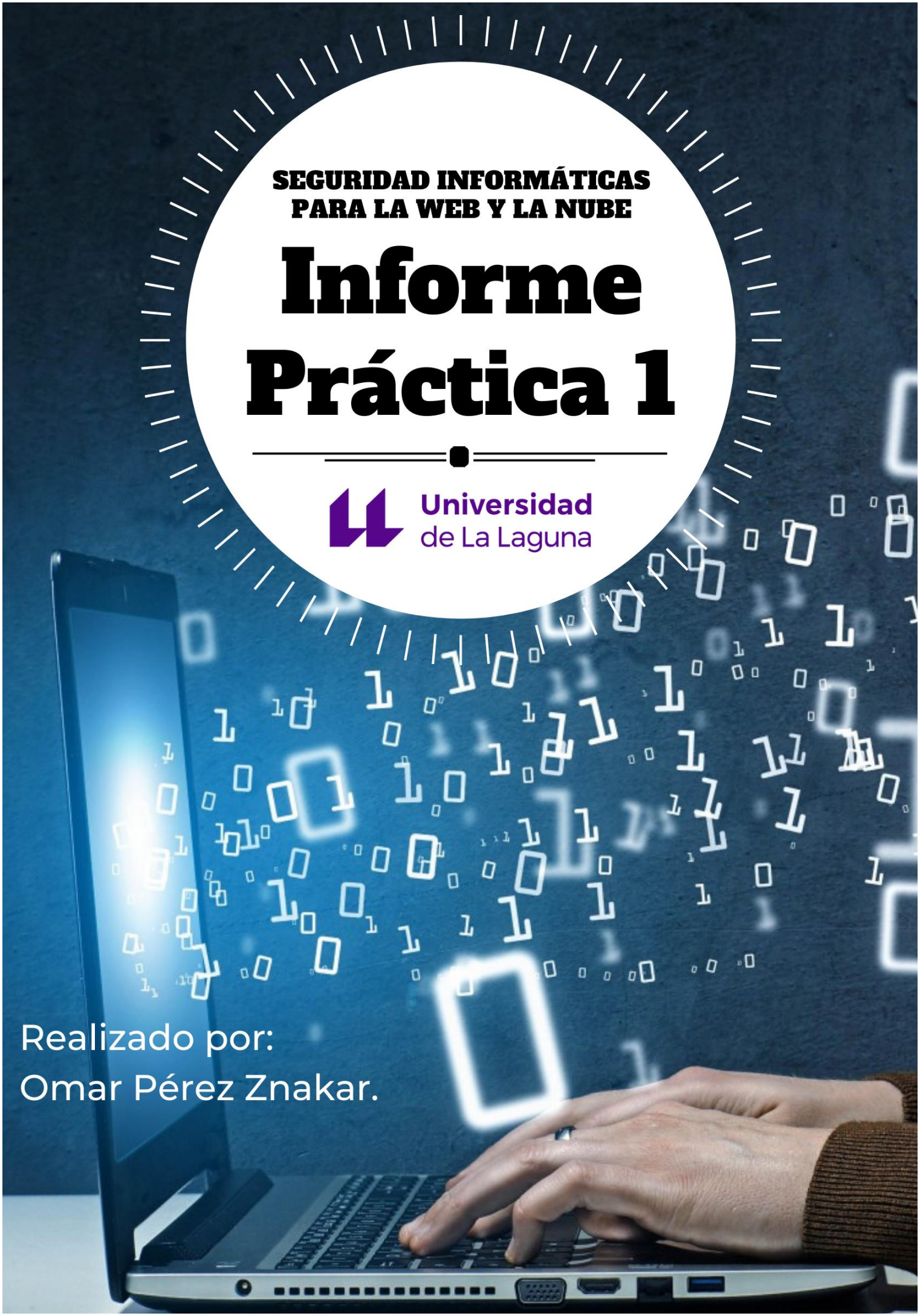


SEGURIDAD INFORMÁTICAS
PARA LA WEB Y LA NUBE

Informe Práctica 1

Universidad de La Laguna

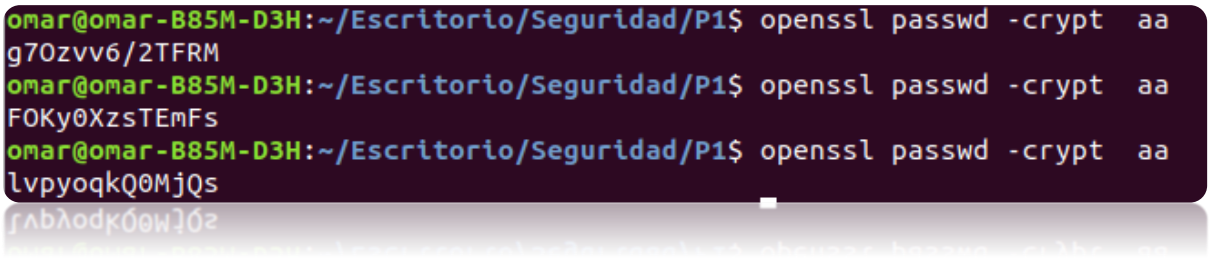
Realizado por:
Omar Pérez Znakar.

Gestión de Identidades-Control de accesos
Autorización

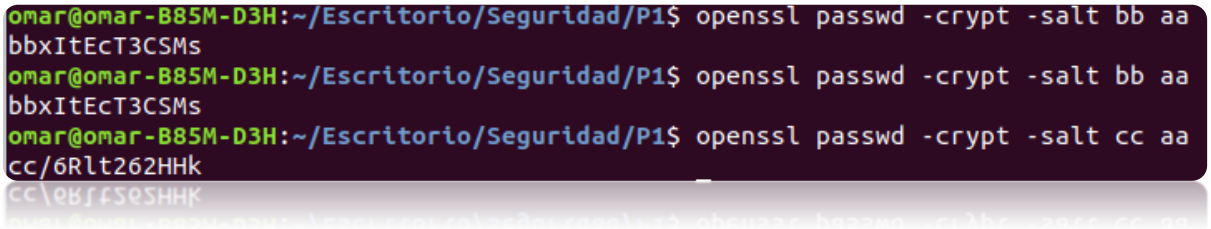
1. Generando Hash.

1.1. Genera una contraseña usando la librería “Openssl”. Ahora, genera la misma contraseña utilizando un salt. ¿Qué diferencia hay?

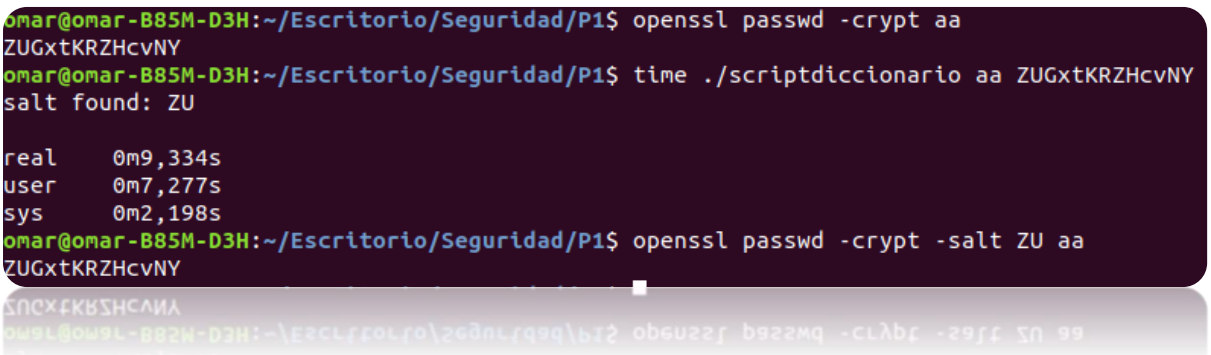
La principal diferencia es que en el caso de hacerlo sin “salt” es que, obtendremos una contraseña que se ha generado con un “salt” aleatorio. Por tanto, cada vez que se ejecute, el hash que nos devuelve será diferente. Esto lo podremos ver a continuación:



Por otro lado, al utilizar el “salt” nosotros le decimos a “openssl” que “salt”” usar. Esto lo veremos a continuación:



Como es obvio, si logramos adivinar el “salt” aleatorio (en caso de no incorporarlo) y ejecutar posteriormente dicho “salt”, con la contraseña veremos el mismo resultado. Para realizar esta investigación se ha modificado el script para adivinar el “salt” a partir de la contraseña y el hash. Un ejemplo de esto lo veremos a continuación:



2. Ataques de fuerza bruta

2.1. Utilizando el script adjunto a esta práctica (hay que darle permisos de ejecución) para romper por ataque de fuerza bruta los siguientes hashes:

Nº caracteres	Salt	Hash	Contraseña	Tiempo
2	iT	iTtle2zsSnkjY	ab	0,009s
2	Za	ZaTXT5zGz.IuM	vT	3,632s

Capturas

```
omar@omar-B85M-D3H:~/Escritorio/Seguridad/P1$ time ./scriptdiccionario iT iTtle2zsSnkjY
password found: ab

real    0m0,009s
user    0m0,003s
sys     0m0,006s
```

```
omar@omar-B85M-D3H:~/Escritorio/Seguridad/P1$ time ./scriptdiccionario Za ZaTXT5zGz.IuM
password found: vT

real    0m3,632s
user    0m2,905s
sys     0m0,799s
```

2.2. Modifica el script anterior para romper los siguientes hashes:

Nº caracteres	Salt	Hash	Contraseña	Tiempo
3	cr	crbZpEDVRIy7Q	ull	4m 58s
4	yp	yp7TQPXS8Ooho	ssi9	3h 56m 36s

Capturas

```
omar@omar-B85M-D3H:~/Escritorio/Seguridad/P1$ time ./scriptdiccionario cr crbZpEDVRIy7Q
password found: ull

real    4m58,169s
user    3m45,354s
sys     1m18,771s
```

```
omar@omar-B85M-D3H:~/Escritorio/Seguridad/P1$ time ./scriptdiccionario yp yp7TQPXS8Ooho
ssi9
password found: ssi9

real    236m36,827s
user    189m11,439s
sys     52m11,564s
```

2.3. Calcula todas las combinaciones posibles para encontrar contraseñas de 3, 4, 5, 6, 7 y 8 caracteres utilizando el alfabeto del script. Estima el tiempo medio para calcular estas combinaciones (aproximado).

Para resolver este ejercicio, se usará la fórmula de variación con repetición. Esta sería:

$$VR_n^m = n^m$$

Donde:

- VR es resultado variación.
- m es el número de dígitos.
- n es el tamaño (en este caso) del alfabeto (60).

De igual forma, he comprobado cuanto tarda en romper el hash “zzqRJ8/B7oL7E” con el salt “zz”. La contraseña que nos da es “##” (el último valor de nuestro script). El resultado dado es el siguiente:

```
omar@omar-B85M-D3H:~/Escritorio/Seguridad/P1$ time ./scriptdiccionario zz zzqRJ8/B7oL7E
password found: ##

real    0m12,365s
user    0m9,635s
sys     0m2,922s
```

Con el cálculo anterior, sabemos que tarda más o menos 12,365s (en este ordenador) en realizar un recorrido completo para dos dígitos (tiempo máximo). Por tanto, podríamos concluir con la siguiente fórmula:

$$T_n^m = m * T_{n-1}^m$$

Donde:

- T es el Tiempo máximo.
- n es el número de dígitos.
- m es el tamaño (en este caso) del alfabeto (60).

Por tanto, aplicaremos dichas fórmulas a cada elemento proporcionado. Esto lo podremos ver a continuación:

- 3 dígitos:
  - Combinaciones: $VR_{60}^3 = 60^3 = 216.000$.
  - Tiempo máximo: $T_{60}^3 = 60 * 12{,}3652 = 741{,}912\ s = 12m\ 22s$.
- 4 dígitos:
  - $VR_{60}^4 = 60^4 = 12.960.000$.
  - Tiempo máximo: $T_{60}^3 = 60 * 741{,}912\ s = 44514{,}72 = 12h\ 21m\ 55s$.

- 5 dígitos:
  - $VR_{60}^{5} = 60^{5} = 777.600.000$.
  - Tiempo máximo: $T_{60}^{3} = 60 * 44514{,}72 = 2670883{,}2s = 30d\ 21h\ 54m\ 43s$.
- 6 dígitos:
  - $VR_{60}^{6} = 60^{6} = 46.656.000.000$.
  - Tiempo máximo: $T_{60}^{3} = 60 * 2670883{,}2s = 160252992s \pm= 5\ años\ 1\ mes$.
- 7 dígitos:
  - $VR_{60}^{7} = 60^{7} = 2{,}79936 \times 10^{12}$.
  - Tiempo máximo: $T_{60}^{3} = 60 * 160252992s = 9615179520 \pm= 304\ años\ 11\ meses$.
- 8 dígitos:
  - $VR_{60}^{8} = 60^{8} = 1{,}679616 \times 10^{14}$.
  - Tiempo máximo:$T_{60}^{3} = 60 * 9615179520 = 576910771200 \pm= 18293\ años\ 9\ meses$.

Tras realizar todos los pasos anteriores, cabe resaltar que todos los cálculos son en caso de que se quiera realizar con un ordenador convencional. Como es obvio, en caso de que se realice con un ordenador más potente estos tiempos bajarán drásticamente.

2.4. Modifica el script anterior para romper los siguientes hashes:

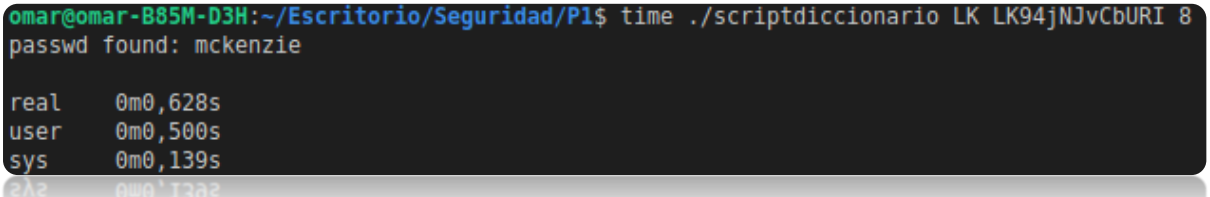
Para este apartado, se ha descargado el diccionario “rockyou.txt” y se ha creado un nuevo script con el siguiente contenido (adjuntado en la entrega):

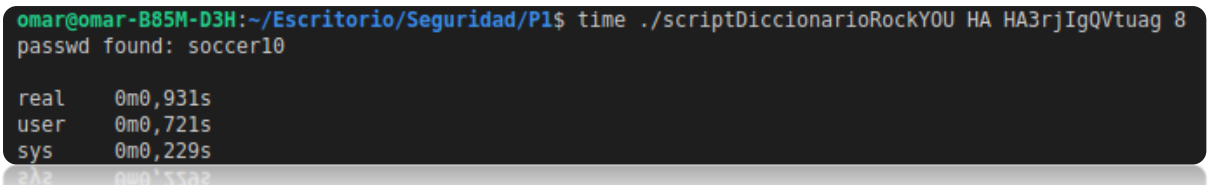
```
#script de ataque de fuerza bruta de 2 caracteres
#!/bin/bash

while read line
do
    if [ ${#line} = $3 ]
    then
        variable=$(openssl passwd -crypt -salt "$1" "$line")
        if [ "$variable" = $2 ]
            then
                echo passwd found: $line
            exit
        fi
    fi
done < ./rockyou.txt
```

Nº caracteres	Salt	Hash	Contraseña	Tiempo
8	LK	LK94jNJvCbURI	mckenzie	0,628
8	HA	HA3rjIgQVtuag	soccer10	0,931

Capturas

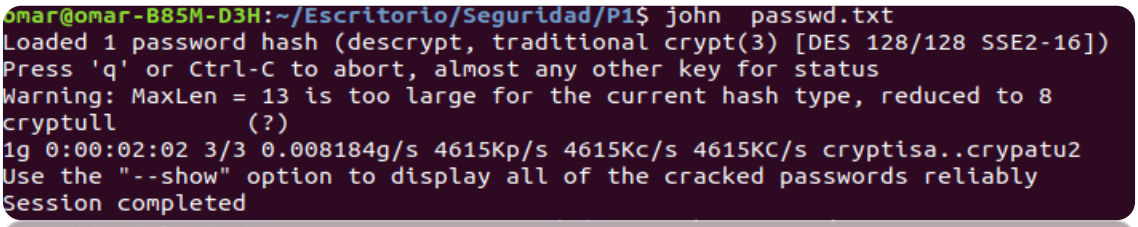


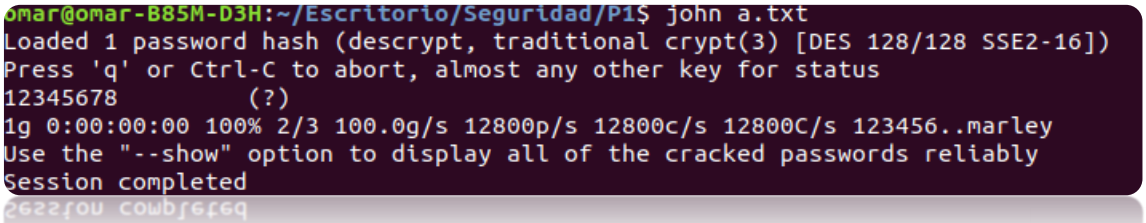


2.5. Utiliza herramientas como John the Ripper para romper los siguientes hashes:

Nº caracteres	Salt	Hash	Contraseña	Tiempo
8	wE	wEJjaGhgmQzbI	cryptull	2m 2s
8	uP	uPFsobeDFz6so	12345678	0s

Capturas





3. Shadow Passwords.

3.1. Analizar en detalle el formato de las entradas de los dos ficheros /etc/passwd y /etc/shadow.

El fichero "/etc/passwd" es el lugar donde podremos ver la información de todos los usuarios del sistema. En este, nos encontraremos una lista con elementos con la siguiente estructura "<nombre>:<password>:<uid>:<gid>:<descripción_ opcional> :<carpeta>:<shell>". Donde cada campo equivale a:

- **<nombre>:** nombre de usuario (entre 1 y 32 caracteres).
- **<password>:** en el inicio se almacenaba la contraseña del sistema. No obstante, en la actualidad aparece una "x" y está se almacena de forma cifrada en el fichero "/etc/shadow".
- **<uid>:** id de usuario (único). Este lo asigna el sistema.
- **<gid>:** la ID de grupo principal (almacenada en el fichero /etc /group).
- **<descripción opcional>:** Le permite agregar información adicional sobre los usuarios, como el nombre completo del usuario, el número de teléfono, etc.
- **<carpeta>:** ruta de la carpeta personal del usuario.
- **<shell>:** Intérprete de comando (shell) del usuario.

El fichero "/etc/shadow" es el lugar donde se almacena la contraseña y la información relacionada con esta. En este, nos encontraremos una lista con elementos con la siguiente estructura: "<nombre>:<contraseña_cifrada>:<último_cambio>:<mínimo> :<máximo>:<advertir>:<inactivo>:<vencimiento>". Donde cada campo equivale a:

- **<nombre>:** nombre de usuario.
- **<contraseña_cifrada>:** contraseña cifrada.
- **<último_cambio>:** día de última modificación de la contraseña.
- **<mínimo>:** número mínimo de días para poder cambiar una contraseña.
- **<máximo>:** número máximo de días de validez de la contraseña.
- **<advertir>:** el número de días antes de que caduque la contraseña se advierte al usuario que su contraseña debe cambiarse
- **<inactivo>:** el número de días después de que caduca la contraseña para que se desactive la cuenta.
- **<vencimiento>:** fecha para cuando esta cuenta no se puede usar.