Cifrado

- PHP incorpora varias funciones para cifrar datos.
- Su uso más habitual, es querer ocultar una contraseña por si algún tercero esté intentando escuchar la comunicación entre el cliente y el servidor PHP.
- Usar funciones que convierten la contraseña en un texto **hash** obtenido mediante un algoritmo de cifrado, complica mucho la obtención de la contraseña en su formato original.

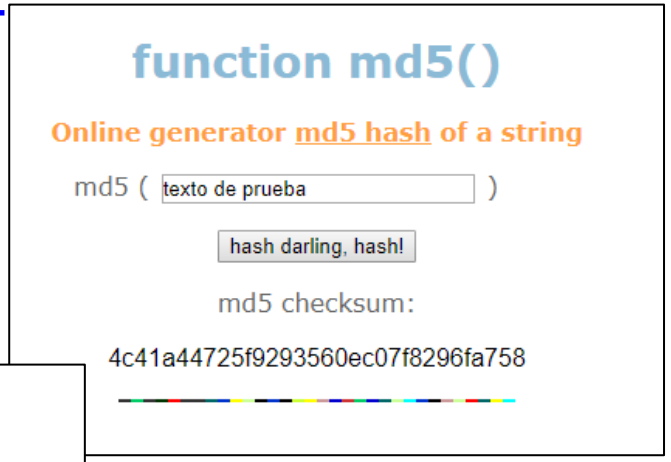
Cifrado

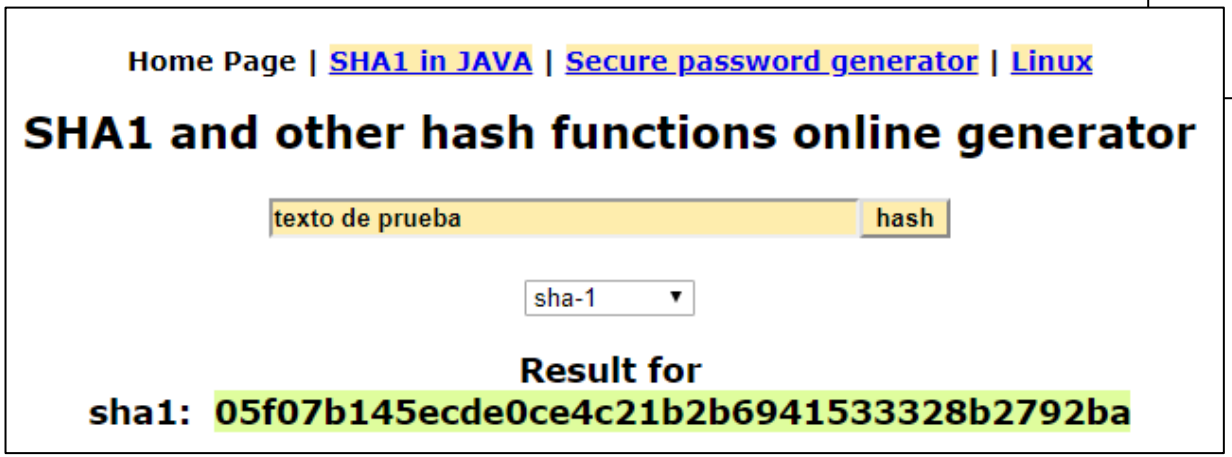
- **Hash:** a partir de un texto de entrada se obtiene un resumen del mismo utilizando un algoritmo de hashing.
- El hash es único para cada texto. La salida tiene un tamaño fijo.
- El objetivo es que a partir del hash no podamos obtener el texto original
- Algunos de los más conocidos son md5, sha-1 y sha-256.
- Por ejemplo, md5 divide el texto en bloques de 512 bits y genera un resumen (hash) de 128 bits por bloque (32 caracteres hexadecimales).

Cifrado

Algunos generadores hash online:

- http://www.sha1-online.com/
- http://www.md5.cz/





Cifrado

- PHP incluye funciones para calcular el hash según el algoritmo a utilizar:
- **Función md5()**:calcula el hash md5 de una cadena. https://www.php.net/manual/es/function.md5.php
- **Función sha1():** calcula el hash sha-1 de una cadena. https://www.php.net/manual/es/function.sha1.php

Estas funciones generan siempre el mismo hash para el mismo texto sin encriptar.

Cifrado

- **Ejemplo:**

Por simplificar el código del ejemplo no se han utilizado consultas preparadas, pero su uso añadiría seguridad en las consultas.

```
$sql="INSERT INTO usuarios(login,password)  VALUES
('".strtoupper($login)."','".md5($_POST['password’])."  '";
```

La password se guarda encriptada en la BD con la función md5()
El valor del login y de la password se escriben entre comillas.

```
$sql="SELECT * FROM usuarios WHERE
login='".strtoupper($login)."' AND
password='".md5($password)."'";
```

A la hora de comprobar credenciales, se aplica la función md5() a la variable que contiene la password, ya que genera el mismo hash para el mismo texto de entrada.

Cifrado

- PHP en su documentación oficial, **no recomienda** su uso debido a la vulnerabilidad de los algoritmos hash ya que con la potencia de las máquinas actuales, los hash se pueden descifrar utilizando ataques de fuerza bruta.
- Entonces, ¿Qué utilizar para cifrar datos desde PHP?
- Actualmente, **la función recomendada por PHP** para el cifrado, y por tanto, la que vamos a utilizar es **password_hash**().
- https://www.php.net/manual/es/function.password-hash.php

Cifrado

- **Función password_hash():** devuelve un hash de contraseña.
- Sintaxis:

```
password_hash (texto, algoritmo[,opciones );
```

  - **texto:** es el texto a cifrar, normalmente una contraseña.

IMPORTANTE: Esta función genera un hash diferente cada vez que se ejecute para el mismo texto sin encriptar. Por lo tanto solo se debe cifrar una vez y utilizar la función **password_verify()** para comprobar si el hash corresponde con un determinado texto.

Cifrado

Sintaxis: `password_hash (texto, algoritmo[,opciones);`

- **algoritmo:** indica el algoritmo a utilizar para realizar el cifrado:
  - **PASSWORD_DEFAULT:** usa el algoritmo **bcrypt**. Esta constante está diseñada para estar actualizada al algoritmo más reciente y fuerte que se ha añadido a PHP. Es la que debemos utilizar. **Se recomienda almacenar el resultado en una columna de una base de datos que pueda ampliarse a más de 60 caracteres (255 caracteres sería una buena elección).**
  - **PASSWORD_BCRYPT:** usa el algoritmo **CRYPT_BLOWFISH** para crear el hash. El resultado será una cadena de 60 caracteres.

Cifrado

Sintaxis: `password_hash (texto, algoritmo[,opciones);`

- opciones que admite el algoritmo:
  - **Salt:** (sal) es un pequeño dato añadido que complica la obtención del dato original. Si es aleatorio mejor. ***Desde la versión 7 de PHP esta opción está obsoleta, se recomienda utilizar el hash generado de forma predeterminada.***
  - **Cost:** un número entre 4 y 32 que indica el número de iteraciones en base 2 que realizará el algoritmo. Por defecto se usa 10. Poner un número más grande hace que la clave tarde más en generarse.

Cifrado

Importante:

- **IMPORTANTE: Cada vez que se llama a la función password_hash(), se genera un nuevo hash diferente**, por ello la extensión facilita la función **password_verify()** que permite comprobar si la contraseña introducida por el usuario coincide con la almacenada en la base de datos.

Cifrado

- Ejemplo:

```php
echo "<h2>hash usando password_hash('Hola clase', PASSWORD_DEFAULT)</h2>";
echo password_hash("Hola clase", PASSWORD_DEFAULT)."<br/>";
echo password_hash("Hola clase", PASSWORD_DEFAULT)."<br/>";
echo "<hr>";
```

hash usando password_hash('Hola clase', PASSWORD_DEFAULT)

$2y$10$JTVrFdKvb9dyaAbiLhl/suh7sL01UDJBFBVYhu9iC73sDPWbaZgSK
$2y$10$xZ1utJl6ZcxnK20RdE/v.O4od4r8kSaTSw0yiOaYpndnOc6UQjaBC

Observa que el hash que se genera es diferente para el mismo texto.

- El identificador **$2y$** indica el algoritmo utilizado.
- El identificador **10$** el coste.
- El resto es el texto hash

Cifrado

- En una aplicación web cuando se crea una contraseña para un usuario (bien la determine él mismo, bien la genere la propia aplicación), se debe almacenar en la base de datos encriptada usando la función password_hash() al insertarla en la tabla.
- Mas tarde, cuando el usuario se identifica con su usuario y contraseña, se comparará con la que está guardada en la BD (encriptada) para ver si es correcta o no usando la función **password_verify()**.

Cifrado

- **La función password_verify(texto, hash)** devuelve verdadero si el hash es correcto para ese texto.
- Es la función principal de verificación de una contraseña (comprobaremos si la contraseña introducida (texto), coincide con la almacenada en la BD (hash).

Cifrado

- Ejemplo

```php
<?php
    $password ="Hola clase";
    //Cada vez que ejecutemos esta función crea un hash nuevo
    $cifrado=password_hash($password, PASSWORD_DEFAULT);
    echo "Texto sin cifrar: $password<br>";
    echo "Texto cifrado: $cifrado<br><br>";

    $txtPrueba = ["Hola CLASE","hola clase", "Hola clase"];
    foreach($txtPrueba as $txt){
        if(password_verify($txt,$cifrado)){
            echo "La contraseña $txt es correcta<br/>";
        }
        else{
            echo "La contraseña $txt es incorrecta<br>";
        }
    }
```

Cifrado

- **Ejemplo:**

Por simplificar el código del ejemplo no se han utilizado consultas preparadas, pero su uso añadiría seguridad en las consultas.

```
$sql="INSERT INTO usuarios(login,password)  VALUES
('".strtoupper($login)."',
'".password_hash($_POST['password'],PASSWORD_DEFAULT))  "." '";
```

La password se guarda encriptada en la BD con la función password_hash()
El usuario y la contraseña van entre comillas

```
$sql="SELECT password FROM usuarios WHERE
login='".strtoupper($_POST['usuario']);
……
if( password_verify($_POST['password'],$passwordBD)
  echo "Credenciales correctas";
else
  echo "Credenciales incorrectas";
```

Se comprueba con la función password_verify() la password que se ha introducido en el formulario (sin cifrar) con la guardada en la BD (cifrada).