Filtrado - Web 300

Descripción:

Este sistema muestra las cuentas de correo electrónico que han sido filtradas en el darkweb, pero no estamos seguros si es fidedigna la información, para estar seguros, puedes ayudarnos a recuperar la contraseña filtrada de carlosmora@mail.com?

El reto está aquí:

http://18.188.52.184:3120

El reto nos proporciona la IP y el número de puerto para acceder a la página. Al ingresar al sitio web nos encontramos con un campo de búsqueda para los correos filtrados.

Al ver que en la página tenemos dos botones que no hacían nada especial y un campo para introducir texto supimos que se trataba de un reto de inyección de comandos, aunque aún no estábamos seguros si sería una SQL inyection (SQLi) o un Cross-Site Scripting (XXS).

Comenzamos por probar varias queries para descubrir una posible SQLi. Al probar con la query `'OR '1'='1` nos retornaba la información de uno de los correos en la Base de Datos (mexa1990@mail.com), con lo cual confirmamos que la página tenía una vulnerabilidad de SQLi que debíamos explotar.

Experimentando un poco descubrimos que era posible pasar nuestro propio código SQL mientras se comentará el resto de la línea utilizando `--`.
El problema era que algunos caracteres como él `;` eran filtrados por lo que no podíamos ejecutarlos. Intentamos diferentes métodos para hacer bypass de ese filtro pero ninguno fue satisfactorio. Finalmentalmente y como último recurso decidimos utilizar un `UNION` el cual

nos permitiría realizar una segunda consulta dentro de esa misma query. Al funcionar supimos que se trataba de un SQL Union Attack

A partir de eso supimos que debíamos hacer una consulta a la Base de Datos con una query de la forma

```
‘) SELECT column_name FROM table_name where email='carlosmora@mail.com'
--
```

El problema era que aún desconocíamos tanto el nombre de la columna como el de la tabla para hacer la consulta. Lo que sí sabíamos a partir de los errores que se generaban era que se estaba utilizando SQLite3.

Investigando un poco en Internet encontramos que SQLite3 contiene una tabla llamada `information_schema.tables` la cual contiene información de la Base de Datos. Por lo que a partir de ella podíamos extraer el nombre de la tabla que contenía la información. Suponiendo que la columna que nos interesaba se llamaría `password`, construimos la siguiente query que nos permitiría conocer el nombre de la tabla:

```
') UNION SELECT password from information_schema.tables--
```

El resultado de ejecutar esa query fue el siguiente:

Como podemos observar, esa consulta nos devolvió toda la información necesaria para construir nuestra nueva query y hacer la consulta de la contraseña a la Base de Datos. La query quedó de la siguiente forma:

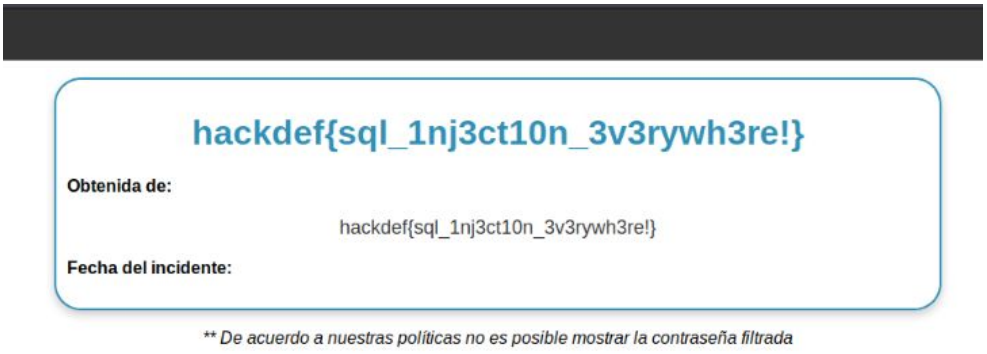
```
') UNION SELECT password from accounts where
email='carlosmora@mail.com'--
```

Desafortunadamente la consulta no se realizó de manera satisfactoria ya que el número de columnas necesarias no coinciden. Para esto, decidimos determinar el número de columnas necesarias manualmente pasando un `null` al valor de la query y agregando uno más cada vez hasta que la consulta no nos mostrará el error del número de columnas. Finalmente determinamos que el tamaño requerido de la columna era de 5.

En base a lo anterior pudimos construir nuestra query final:

```
') UNION SELECT password,password,password,password,password from
accounts where email='carlosmora@mail.com'--
```

Al pasar la query nos mostró la siguiente información la cual contenía nuestra flag.



Flag:

```
hackdef{sql_1njecti0n_3v3rywh3re!}
```