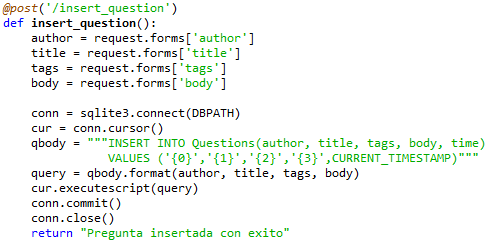
# Auditoría de seguridad web

## Declaración de autoría

Aitor Cayón Ruano, José Javier Cortés Tejada, Fernando Pérez Gutiérrez, Gabriel Sellés Salvà declaramos que esta solución es fruto exclusivamente de nuestro trabajo personal. No hemos sido ayudados por ninguna otra persona ni hemos obtenido la solución de fuentes externas, y tampoco hemos compartido nuestra solución con nadie. Declaramos además que no hemos realizado de manera deshonesta ninguna otra actividad que pueda mejorar nuestros resultados ni perjudicar los resultados de los demás.

## Vulnerabilidad 1

* **Ruta de la aplicación:** /insert\_question
* **Tipo de vulnerabilidad:** SQL Injection
* **Causante de la vulnerabilidad:**

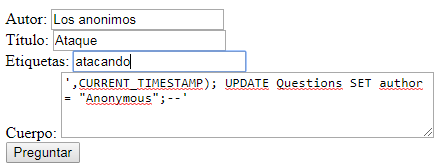
Si en alguno de los campos de entrada (Autor, Título, Etiquetas o Cuerpo) introducimos una sentencia muy concreta, podemos ejecutar sentencias SQL no planeadas por el programador de la web.

* **Repercusiones:**

Eliminación de la base de datos.

Modificar otros datos existentes en la base de datos.

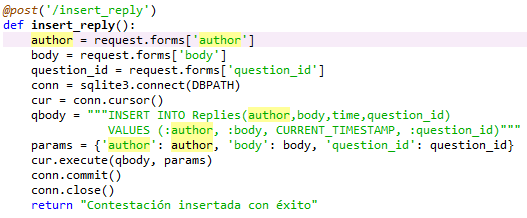
* **Ejemplo paso a paso:**



* **Medidas para mitigar la vulnerabilidad:**

Para mitigar la vulnerabilidad podríamos utilizar sentencias SQL preparadas.

## Vulnerabilidad 2

* **Ruta de la aplicación:** /insert\_reply
* **Tipo de vulnerabilidad:** StoredXSS
* **Causante de la vulnerabilidad: **

Si en alguno de los campos de entrada (Autor o Cuerpo) introducimos un código script, podremos ejecutarlo cada vez que se vuelva a acceder a la página de esa pregunta.

* **Repercusiones:** según el script podríamos desde robar las cookies del usuario hasta redirigirlo hacia una página con software malicioso.
* **Ejemplo paso a paso:**

|  |
| --- |
|  |
| 1. **ACCEDER A UNA PREGUNTA** |

|  |
| --- |
|  |
| 1. **INSERTAR EL CÓDIGO SCRIPT COMO SI FUESE LA RESPUESTA A UNA PREGUNTA O UN AUTOR** |

|  |
| --- |
|  |
| 1. **LA RESPUESTA SE INSERTA EN LA BASE DE DATOS** |

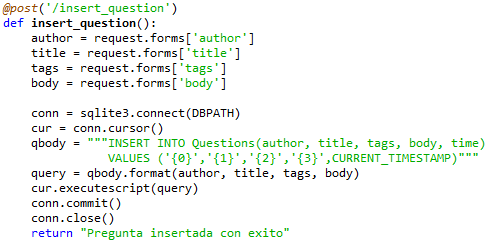
|  |
| --- |
| 1. **LA PRÓXIMA VEZ QUE ACCEDAMOS A LA PREGUNTA NOS REDIRECCIONARÁ** |

* **Medidas para mitigar la vulnerabilidad:**

Validar los datos introducidos como en los campos de texto. Para ello debemos prestar especial atención a los caracteres <>.

## Vulnerabilidad 3

* **Ruta de la aplicación:** /insert\_question
* **Tipo de vulnerabilidad:** CSRF
* **Causante de la vulnerabilidad:**

****

Al ser “conocidos” todos los parámetros y extraerse de los campos de texto de la página podríamos replicar un HTML con dichos campos y llamar al servicio.

* **Repercusiones:** el usuario operaría en una página sin saber que sus datos están siendo enviados a otra. En nuestro caso podríamos robar los datos de una cuenta.
* **Ejemplo paso a paso:**

|  |
| --- |
|  |
| 1. **EL USUARIO INTRODUCE CONFIADAMENTE SUS DATOS** |

|  |
| --- |
|  |
| 1. **LLAMAMOS AL SERVICIO Y ENVIAMOS LOS DATOS A LA OTRA PÁGINA** |

|  |
| --- |
|  |
| 1. **LOS DATOS HAN SIDO PUBLICADOS EN LA PÁGINA** |

* **Medidas para mitigar la vulnerabilidad:** validar la operación con un secreto. Solo aceptar aquellas peticiones provenientes de páginas adecuadas en nuestra web.