Blandskron

Laboratorio de Gestión y Seguridad de Datos de Usuarios con SQLite

Este laboratorio se centra en la creación, manipulación y aseguramiento de una pequeña base de datos SQLite que contiene información sensible de usuarios—nombre, correo y rol—y que nos servirá de escenario para diversas prácticas de desarrollo y seguridad. El punto de partida es un script Python que inicializa la base de datos y crea la tabla `usuarios`. A continuación, se inyectan registros de ejemplo que representan distintos perfiles profesionales (Analista, Pentester, Auditor y Administrador). Finalmente, un tercer fragmento de código recupera y muestra todos los registros, confirmando que la inserción ha tenido lugar correctamente.

Con esta base montada, el laboratorio se desarrolla en varias fases. Primero verificaremos el correcto funcionamiento del CRUD básico: creación de la tabla, inserción de datos y lectura de registros. Aprenderemos a modificar consultas SQL para filtrar usuarios por rol, a ordenar resultados y a paginar salidas cuando el volumen de datos crezca. A continuación, exploraremos la exportación de la información a formatos interoperables como CSV o JSON, facilitando su análisis con herramientas externas.

La segunda parte se ocupa de los riesgos asociados a la concatenación directa de parámetros en consultas SQL. Reproduciremos un escenario de inyección injuriosa donde un atacante no autenticado pueda alterar el parámetro `rol` en la consulta, obteniendo acceso a datos no autorizados. Para ello modificaremos dinámicamente el código de `query_user` o crearemos nuevas rutas que ejecuten filtros incondicionales, y observaremos en tiempo real cómo un payload malicioso puede burlar controles.

Inmediatamente después, contrarrestaremos esa vulnerabilidad implementando prepared statements con placeholders, validación y saneamiento de entrada. Compararemos el comportamiento de ambas versiones ante los mismos ataques, midiendo tiempos de respuesta y examinando mensajes de error para garantizar que no se filtra información interna sobre la estructura de la base de datos.

Paralelamente, estudiaremos la gestión de roles y privilegios a nivel de aplicación. Diseñaremos un sencillo mecanismo en Flask que limite el acceso a determinadas rutas (por ejemplo, el panel de administración) únicamente a usuarios con rol “Administrador”. Evaluaremos la efectividad de este control y cómo puede integrarse con la propia capa de base de datos (a través de vistas o triggers) para reforzar la defensa en profundidad.

Para cerrar el laboratorio, realizaremos un ejercicio de auditoría de cambios: cada vez que se inserte, actualice o borre un registro, escribiremos en un log—bien en una tabla adicional de SQLite, un archivo de texto o un sistema centralizado—la operación, el usuario actor y la marca temporal. Este registro de auditoría nos permitirá reconstruir el historial de modificaciones y detectar patrones inusuales, sentando las bases de un Sistema de Gestión de Seguridad de la Información acorde a estándares como ISO 27001 o NIST.

A lo largo de estas prácticas aprenderemos a manipular datos de forma segura, a reconocer y mitigar vulnerabilidades de inyección SQL y a aplicar controles de acceso basados en roles. Al concluir, deberías estar capacitado para diseñar arquitecturas de datos más robustas en aplicaciones Python, entender los riesgos de las consultas sin parametrizar y conocer las buenas prácticas de registro y auditoría que requieren las organizaciones que manejan información crítica de usuarios.