Actividades en clase

Actividad 1

1- Diagrama de flujo de datos de logueo

2- Analizar y enumerar las funcionalidades de seguridad. Debe incluir tipos o niveles de usuarios, tipos de vistas, selección de cifrado, permisos (justificando dicha elección).

3- ¿Con respecto a la capa Humana, qué medidas planifican aplicar para evitar un ataque contra la seguridad de los datos?

Actividad 2

Requisitos:

1- Tener un sistema web funcional en PHP o Python, local o remoto.
2- Tener instalado Python 3, Bash (en Linux), o WSL en Windows.
3- Descargar y descomprimir el paquete de scripts

- audit_archivo.sh: script para activar auditoría de acceso a un archivo (usando auditctl).
- log_login.php: script PHP para registrar intentos de login.
- detectar_sql.php: script PHP que detecta posibles inyecciones SQL en variables GET.
- check_password_strength.py: script en Python para evaluar la fortaleza de una contraseña.
- site_path_finder.py: script en Python que busca rutas comunes en un sitio web (simula un escaneo básico de directorios).

Ejecuten el script con permisos de root (sudo). Usen el comando: bash

```
sudo ausearch -k config_audit
```

https://computernewage.com/2019/01/13/scripting-linux-bash-crear-ejecutar-script/

¿si no me ejecuta sudo? Veamos si tenemos los privilegios

```
sudo whoami
```

a) Si te pide contraseña y dice root, tenés acceso.

b) Si te dice algo como “is not in the sudoers file”, entonces tu usuario no tiene permisos de sudo. =(

Registro en informe:

- Ruta auditada
- Eventos detectados (captura o transcripción)

2- **Fortalece tus contraseñas**

¿Una contraseña segura?

Consejo: Usa un gestor de contraseñas (KeePassXC, Bitwarden, 1Password, LastPass). Así evitas tratar de recordar cadenas complejas y puedes generar contraseñas únicas para cada cuenta.

- Ejecuten el script check_password_strength.py
- Prueben contraseñas reales (con cuidado) o simuladas.
- Modifiquen el script para reforzar el análisis.

Registro en informe:

. Contraseñas probadas (pueden ser ficticias)

. Nivel de seguridad detectado

➢ Extra: Cifrar con GPG un archivo de texto que contenga contraseñas de pruebas. Mostrar cómo, sin la clave privada, nadie puede leerlo. ¿Qué pasa si un atacante intercepta el archivo cifrado? Sin la clave no puede ver su contenido, pero aún podría eliminar el archivo para afectar la disponibilidad.

3) **Rutas ocultas o mal protegidas**

- Ejecuten site_path_finder.py
- Ingresen la URL de tu sistema (puede ser http://localhost/ o una IP)
- Observen qué rutas devuelve como accesibles

Registro en informe:

Rutas encontradas

Qué información exponen

¿Deberían estar accesibles?

4) **Auditoría de entradas de usuario**

- [x] Detectar intentos de inyección SQL.
- [x] Incluyan detectar_sql.php en sus proyectos.
- [x] Prueben enviar parámetros como ?user=admin' OR '1'='1
- [x] Verifiquen si se genera un log de advertencia en logs/inyeccion.log

Registro en informe:

Entradas usadas

Logs generados

¿Se detectó el intento?

5) **Registrar accesos al sistema.**

- [x] Agreguen log_login.php en tu proceso de login.
- [x] Ingresen usuarios reales y ficticios.
- [x] Verifiquen el contenido de logs/login.log.

Registro en informe:

Usuarios que accedieron

Timestamps registrados

¿Hay registros de intentos fallidos?