“Año De La Recuperación Y
Consolidación De La Economía Peruana”

UNIVERSIDAD PERUANA LOS ANDES

“FACULTAD DE INGENIERÍA”

ESCUELA PROFESIONAL “SISTEMAS Y COMPUTACIÓN”

Manual Semana 11: Seguridad y control de acceso

CÁTEDRA: Base de Datos II

CATEDRÁTICO: Ing. Fernandez Bejarano Raul Enrique

ESTUDIANTE: Quispe Segama Franklin Noe

CICLO: V

SECCIÓN: A1

HUANCAYO PERÚ

2025

Temas de Seguridad y Control de Acceso

1. Autenticación SQL y Windows

Enunciado: ¿Cuáles son las diferencias fundamentales entre la **Autenticación de SQL Server** y la **Autenticación de Windows** y cuáles son las prácticas recomendadas para cada una?
Explicación:

- **Autenticación de Windows:** Utiliza las credenciales del sistema operativo (Active Directory o cuentas locales) para validar al usuario. Es el método preferido (práctica segura) ya que centraliza la administración de usuarios y aprovecha las políticas de contraseñas robustas de Windows. El usuario se conecta como un *login* de dominio.
- **Autenticación de SQL Server:** Requiere que SQL Server almacene el nombre de usuario y la contraseña (hash) de forma interna. Es útil para aplicaciones o usuarios fuera de un dominio de Windows. La práctica segura es usar **contraseñas complejas**, forzar el **cambio de contraseña**, y evitar la cuenta `sa` (System Administrator) o deshabilitarla si no es estrictamente necesaria.

Código SQL: (Ejemplo de creación de un login de cada tipo, asumiendo que el login de Windows ya existe en el dominio).

```sql
-- 1. Crear un Login de SQL Server (Práctica: Usar una contraseña fuerte)
CREATE LOGIN [UsuarioSQL] WITH PASSWORD = 'Contraseña_Fuerte!2025', CHECK_POLICY = ON;
GO

-- 2. Crear un Login de Windows (Práctica: Preferido por seguridad)
-- Reemplazar 'DOMINIO\NombreUsuario' con tu usuario de Windows/Dominio
CREATE LOGIN [Aurora\frank] FROM WINDOWS;
GO

--si ya lo tienes creado simplemente asígnale acceso a la base de datos
USE QhatuPERU;
CREATE USER [Aurora\frank] FOR LOGIN [Aurora\frank];
ALTER ROLE db_datareader ADD MEMBER [Aurora\frank];
ALTER ROLE db_datawriter ADD MEMBER [Aurora\frank];
```

```
100 %
Messages
Commands completed successfully.

Completion time: 2025-11-13T07:25:08.9162207-05:00
```

2. Cuentas de Servicio y Configuración del Servidor

- **Enunciado:** ¿Qué son las **Cuentas de Servicio** de SQL Server y cómo debe configurarse su **nivel de privilegio** para una operación segura?
- **Explicación:** Las Cuentas de Servicio son las identidades de Windows bajo las cuales se ejecutan los servicios principales de SQL Server (Motor de Base de Datos,

Agente, etc.). La práctica segura esencial es seguir el principio de **mínimo privilegio**.
    - Se debe evitar el uso de cuentas de alto privilegio como `Local System` o cuentas de dominio de usuario.
    - La mejor práctica es usar **Cuentas de Servicio Administradas de Grupo (gMSA)** o, al menos, cuentas de usuario de dominio dedicadas con los permisos mínimos necesarios para operar SQL Server.
- **Código SQL:** (Este tema se configura primariamente fuera de T-SQL, en el **Administrador de Configuración de SQL Server** o en Windows. El código solo muestra un chequeo de la cuenta de servicio).

```sql
--2
-- Obtener la cuenta de servicio actual que está ejecutando la instancia de SQL Server
SELECT
    servicename,
    service_account,
    instant_file_initialization_enabled
FROM
    sys.dm_server_services
WHERE
    servicename LIKE '%' + CAST(SERVERPROPERTY('InstanceName') AS NVARCHAR(100)) + '%';
```

100 %

Results | Messages

	servicename	service_account	instant_file_initialization_enabled
1	SQL Server (MSQLSERVER)	NT Service\MSSQL$MSQLSERVER	N
2	SQL Server Agent (MSQLSERVER)	NT Service\SQLAgent$MSQLSERVER	N

3. Creación de Roles Fijos y Personalizados

- **Enunciado:** ¿Cuál es la diferencia entre un **Rol Fijo de Servidor** y un **Rol Personalizado de Base de Datos**, y cómo se crea este último para agrupar permisos?
- **Explicación:**
    - **Roles Fijos de Servidor:** Son roles predefinidos (ej. `sysadmin`, `serveradmin`, `securityadmin`) que tienen un conjunto fijo de permisos a **nivel de servidor**. No se pueden modificar.
    - **Roles Personalizados de Base de Datos:** Son roles creados por el usuario a **nivel de base de datos** para agrupar usuarios y asignarles un conjunto de permisos específicos (siguiendo el principio de mínimo privilegio).
- **Código SQL:** (Creación de un Rol Personalizado llamado `RolLecturaInformes` y asignación de un permiso).

```sql
--3
USE [QhatuPERU]; -- Asegúrate de estar en la base de datos correcta
GO

-- 1. Crear el rol personalizado de base de datos
CREATE ROLE [RolLecturaInformes];
GO

-- 2. Asignar el permiso específico al rol
-- En este caso, permiso para solo leer (SELECT) en la tabla de Informes
GRANT SELECT ON [dbo].[Informes] TO [RolLecturaInformes];
GO

-- 3. Agregar un usuario al nuevo rol
ALTER ROLE [RolLecturaInformes] ADD MEMBER [UsuarioSQL]; -- Usa un usuario ya creado
```

100 %

Messages

```
Commands completed successfully.

Completion time: 2025-11-13T07:35:03.6761267-05:00
```

4. Control de Acceso mediante GRANT, DENY y REVOKE

- **Enunciado:** Explica el propósito de los comandos **GRANT**, **DENY** y **REVOKE** en el control de acceso y cómo interactúan entre sí.
- **Explicación:** Estos son los pilares del control de acceso discrecional (DAC) en SQL Server:
    - **GRANT:** Otorga un permiso a un principal (usuario o rol).
    - **DENY:** Deniega explícitamente un permiso a un principal. **DENY anula a GRANT** (es decir, si a un rol se le da GRANT, pero al usuario se le da DENY, prevalece DENY).
    - **REVOKE:** Elimina un permiso que previamente fue otorgado (GRANT) o denegado (DENY).
- **Código SQL:** (Ejemplo de uso secuencial en una tabla de Ventas).

```sql
--4
USE [QhatuPERU];
GO

-- 1. Otorgar permiso de INSERT (GRANT)
GRANT INSERT ON [dbo].[Ventas] TO [UsuarioSQL];
GO

-- 2. Denegar explícitamente permiso de DELETE (DENY)
-- Esto asegura que el usuario no pueda borrar, incluso si es miembro de un rol que tiene DELETE.
DENY DELETE ON [dbo].[Ventas] TO [UsuarioSQL];
GO

-- 3. Revocar el permiso de INSERT (si se quiere quitar)
REVOKE INSERT ON [dbo].[Ventas] TO [UsuarioSQL];
```

```
100 %
Messages
Commands completed successfully.

Completion time: 2025-11-13T07:37:10.8019711-05:00
```

5. Cifrado y Protección de Datos (TDE, Always Encrypted)

- **Enunciado:** ¿Cuál es la diferencia entre **Transparent Data Encryption (TDE)** y **Always Encrypted** y qué capa de protección ofrece cada uno?
- **Explicación:** Ambos son mecanismos de cifrado críticos:
    - **TDE (Transparent Data Encryption):** Cifra los **archivos de la base de datos completa** (datos en reposo) y los archivos de *backup*. Protege contra el acceso físico a los archivos. El cifrado/descifrado es transparente para la aplicación. El dato está descifrado en memoria.
    - **Always Encrypted:** Cifra **columnas de datos específicas** y solo permite el descifrado al cliente de la aplicación que posee las claves de columna. Protege los datos **en tránsito y en uso** (el dato se mantiene cifrado en SQL Server, incluso en la memoria).
- **Código SQL:** (El setup de Always Encrypted es complejo y requiere herramientas de cliente. Este ejemplo muestra los pasos básicos de TDE).

```sql
-- Revocar (eliminar) el permiso de actualización (UPDATE) sobre la columna Email
REVOKE UPDATE ON Empleados (Email) FROM Rol_Ventas_Tech;


-- Ejemplo de la inserción de datos para el PIN de Ana Gómez
INSERT INTO Empleados (..., PIN_Acceso, ...) VALUES
(100, ..., HASHBYTES('SHA2_256', 'Gerente2025'), ...);

-- Ejemplo de cómo actualizar el PIN de un empleado:
UPDATE Empleados
SET PIN_Acceso = HASHBYTES('SHA2_256', 'NuevoPIN123')
WHERE ID_Empleado = 101;
```

```
90 %
Resultados
Los comandos se han completado correctamente.

Hora de finalización: 2025-11-12T13:15:26.5395778-05:00
```

6. Auditoría y Monitoreo de Eventos con SQL Server Audit

- **Enunciado:** ¿Cómo se utiliza **SQL Server Audit** para monitorear eventos de seguridad y cuáles son los componentes clave de su implementación?
- **Explicación:** SQL Server Audit permite rastrear y registrar eventos relevantes para la seguridad, como inicios de sesión, cambios de permisos, o modificaciones de datos. Los componentes clave son:
    - **SQL Server Audit:** El objeto a nivel de servidor que define dónde se guardará el registro (archivo, registro de eventos de seguridad, o registro de eventos de aplicación).
    - **Especificación de Auditoría de Servidor:** Define qué eventos a nivel de servidor (ej. inicios de sesión fallidos/exitosos, cambios de roles) se deben auditar.
    - **Especificación de Auditoría de Base de Datos:** Define qué eventos a nivel de base de datos (ej. SELECT, INSERT, DELETE en una tabla específica) se deben auditar.
- **Código SQL:** (Creación de un Audit a nivel de servidor).

```
--6
-- 1. Crear el objeto de SQL Server Audit
-- Reemplaza 'C:\Audit\' con un directorio seguro y existente
CREATE SERVER AUDIT [Audit_Server_Logins]
TO FILE
(    FILEPATH = 'C:\Audit\'
    ,MAXSIZE = 10 MB
    ,MAX_ROLLOVER_FILES = 5
    ,RESERVE_DISK_SPACE = OFF
)
WITH
(    QUEUE_DELAY = 1000
    ,ON_FAILURE = CONTINUE -- CONTINUAR la operación si la auditoría falla
);
GO
```

```
%
Messages
Completion time: 2025-11-13T07:54:35.7168475-05:00
```

Prácticas de Implementación

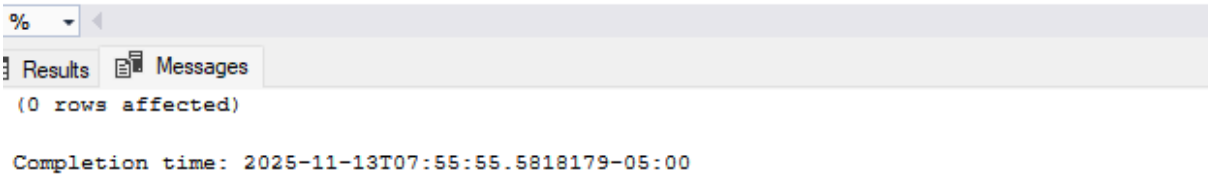
1. Configurar auditoría para seguimiento de inicios de sesión

- **Enunciado:** Implementar una **Especificación de Auditoría de Servidor** que rastree y registre los **intentos fallidos y exitosos de inicio de sesión**.
- **Explicación:** Esta práctica se realiza para asegurar la trazabilidad de quién intenta y quién logra acceder a la instancia. Utiliza el **SQL Server Audit** creado previamente y añade la especificación de eventos a nivel de servidor.

- **Código SQL:** (Asumiendo que `[Audit_Server_Logins]` ya está creado y habilitado).

```sql
-- 1. Crear la Especificación de Auditoría de Servidor
CREATE SERVER AUDIT SPECIFICATION [Login_Attempts_Spec]
FOR SERVER AUDIT [Audit_Server_Logins]
ADD (SUCCESSFUL_LOGIN_GROUP), -- Para inicios de sesión exitosos
ADD (FAILED_LOGIN_GROUP)      -- Para intentos de inicio de sesión fallidos
WITH (STATE = ON);
GO

-- 2. Verificar que la especificación está activa
SELECT * FROM sys.server_audit_specifications WHERE name = 'Login_Attempts_Spec';
```



6. Implementar un esquema de cifrado para una columna sensible (ej. DNI o tarjeta)

- **Enunciado:** Describir los pasos para usar **Always Encrypted** para cifrar una columna sensible (`NumeroDNI`) en una tabla llamada `Clientes`.
- **Explicación:** Always Encrypted requiere configurar claves en el lado del cliente (aplicación/herramientas) para que solo la aplicación pueda ver los datos claros. En T-SQL, se define la **Clave Maestra de Columna (CMK)** y la **Clave de Cifrado de Columna (CEK)**, y se modifica la columna.
- **Código SQL:** (El proceso completo de AE es extenso y requiere claves en el *Key Store* de Windows, pero este es el código base en SQL).

```sql
--2
USE [QhatuPERU];
GO

-- 1. Crear la Clave Maestra de Columna (CMK)
-- (La clave real reside en un almacén de claves como Azure Key Vault o un almacén de certificados)
CREATE COLUMN MASTER KEY [CMK_AlwaysEncrypted]
WITH (
    KEY_STORE_PROVIDER_NAME = 'MSSQL_CERTIFICATE_STORE', -- Ejemplo: Usando Certificado
    KEY_PATH = 'Certificado_CN' -- El nombre común (CN) del certificado
);
GO

-- 2. Crear la Clave de Cifrado de Columna (CEK)
CREATE COLUMN ENCRYPTION KEY [CEK_DNI]
WITH VALUES (
    COLUMN_MASTER_KEY = [CMK_AlwaysEncrypted],
    ALGORITHM = 'RSA_OAEP', -- Algoritmo de cifrado
    ENCRYPTED_VALUE = 0x... -- Este valor se genera usando SSMS o PowerShell
);
GO
```

```sql
-- 2. Crear la Clave de Cifrado de Columna (CEK)
CREATE COLUMN ENCRYPTION KEY [CEK_DNI]
WITH VALUES (
    COLUMN_MASTER_KEY = [CMK_AlwaysEncrypted],
    ALGORITHM = 'RSA_OAEP', -- Algoritmo de cifrado
    ENCRYPTED_VALUE = 0x... -- Este valor se genera usando SSMS o PowerShell
);
GO

-- 3. Modificar la tabla para cifrar la columna 'NumeroDNI'
ALTER TABLE [dbo].[Clientes]
ALTER COLUMN [NumeroDNI] NVARCHAR(20)
ENCRYPTED WITH (
    COLUMN_ENCRYPTION_KEY = [CEK_DNI],
    ENCRYPTION_TYPE = DETERMINISTIC, -- Permite búsquedas de igualdad
    -- ENCRYPTION_TYPE = RANDOMIZED, -- Mejor seguridad, no permite búsquedas
    -- (El tipo de dato debe ser consistente con el cifrado)
    -- Tipo de dato para cifrado
    -- Note: El cifrado cambia el tipo de dato subyacente a VARBINARY
    -- Esto se hace típicamente con el asistente de SSMS o PowerShell.
    -- Este código es solo indicativo de la definición de cifrado.
);
```