

Curso: (62612) Diseño de aplicaciones seguras

Fernando Tricas García

Departamento de Informática e Ingeniería de Sistemas
Universidad de Zaragoza

<http://www.cps.unizar.es/~ftricas/>

<http://moodle.unizar.es/>

ftricas@unizar.es

Tema XII: Seguridad en bases de datos

Fernando Tricas García

Departamento de Informática e Ingeniería de Sistemas
Universidad de Zaragoza

<http://www.cps.unizar.es/~ftricas/>

[http://moodle.unizar.es/
ftricas@unizar.es](http://moodle.unizar.es/ftricas@unizar.es)

Seguridad en bases de datos

- ▶ Los propios gestores no tienen en cuenta estos aspectos
- ▶ Las medidas adoptadas no son estándar entre distintos sistemas
- ▶ ¡Leer la documentación!
- ▶ Hay libros completos dedicados a este tema
- ▶ Daremos algunas ideas

Los objetivos

- ▶ Los propios datos (claro)

Los objetivos

- ▶ Los propios datos (claro)
Pero también ...
- ▶ Cotas. Valores máximos o mínimos pueden proporcionar información interesante
- ▶ Existencia. Si existe un dato y se puede saber ...
- ▶ Resultados negativos. Cuando alguien (o algo) no tiene 'algo' también es información interesante.
- ▶ Valor probable

Hay que defenderse contra todas las posibilidades...

Tampoco se trata (como casi siempre) de cerrar toda la información para que no pase nada malo.



Y por supuesto...

- ▶ Consistencia interna.
 - ▶ Los datos cumplen con las reglas establecidas (no hay precios negativos, ...)
- ▶ Consistencia externa
 - ▶ Los datos reflejan la realidad.

Reglas de integridad

Además de la integridad referencial y las tradicionales

- ▶ Comprobación de los valores de los campos: contienen valores 'razonables' y/o válidos
- ▶ Comprobación de ámbito adecuado (devolver valores sólo cuando el rango es suficientemente amplio).
- ▶ Comprobaciones de consistencia.
Ej.: en la factura algo vale X, y en la tarifa vale Y...

Puede haber conflictos entre estas comprobaciones y la confidencialidad (para comprobar algún valor hay que acceder a valores privados)



Empezando por lo básico

- ▶ La conexión con la base de datos no suele ser cifrada
 - ▶ Oracle, como producto aparte
 - ▶ MySQL, con SSL
 - ▶ Otros?

Siempre se puede habilitar una red privada virtual (VPN)

Empezando por lo básico

Nuestra principal preocupación debería ser el daño que pueda realizar una entrada de datos maliciosa

- ▶ Todas (o casi todas) proporcionan autenticación mediante clave, y control de acceso a las tablas
- ▶ Sólo ofrecen protección contra el resto de usuarios de la base de datos

Control de acceso

- ▶ Niveles de acceso
 - ▶ Manipulación de datos de las tablas básicas
 - ▶ Operaciones compuestas
- ▶ Podemos:
 - ▶ Restringir las operaciones basándonos en cada usuario
 - ▶ Definir los niveles de protección para cada tipo de dato
- ▶ En todo caso ...
 - ▶ Completitud (todos los campos están protegidos)
 - ▶ Consistencia (no hay incoherencias en las reglas)

Más ideas

- ▶ Esto no ayuda con aplicaciones que tengan su propio sistema de usuarios
- ▶ En la práctica, los propios programas deben ocuparse del control de acceso a la base de datos
- ▶ Hay que ser muy cuidadosos con la información que se pueda deducir

Control de acceso

- ▶ El objetivo es proteger a unos usuarios de los otros
- ▶ Cada usuario tiene sus tablas, y gestiona quién puede acceder a ellas.
- ▶ No es una solución escalable

Control de acceso

- ▶ Generalmente:

- ▶ Usuarios

- Se autentifica con un proceso de entrada. A veces es suficiente con que el usuario se haya autenticado en el sistema operativo.

- ▶ Acciones (SELECT, INSERT, ...),

- ▶ Objetos (Tablas, otros...),

Los usuarios invocan acciones sobre los objetos. El DBMS decide si la acción está permitida o no.

Control de acceso

- ▶ Privilegios (quién, qué, herencia, ... GRANT)
- ▶ Vistas (sólo puedes ver ...)
 - ▶ Tablas virtuales como filtro de las reales
 - ▶ Se podría crear una por cada tipo de usuario del sistema

```
CREATE VIEW      empleado_sencillo AS  
SELECT nombre, despacho, telefono,  
correo, fechaIngreso  
FROM info_empleado;
```

Tambien se pueden hacer cosas mas sofisticadas...

Las vistas

- ▶ Son flexibles y permiten políticas de control de acceso definidas a un nivel próximo al de la aplicación
- ▶ Pueden servir para políticas de seguridad basadas en contexto o en los datos
- ▶ Invocación controlada
 - ▶ Las lecturas sin problemas
 - ▶ Las actualizaciones pueden tener problemas
- ▶ Los datos pueden ser reclasificados fácilmente

También hay desventajas

- ▶ La comprobación de acceso puede ser complicada y lenta
- ▶ Las vistas tienen que comprobarse desde el punto de vista de la corrección: ¿respetan la política de seguridad?
- ▶ Completitud y consistencia no se obtienen automáticamente.
- ▶ La parte relativa a la seguridad del DBMS se puede complicar mucho.

Protección de campos

- ▶ Ni siquiera los que tienen acceso a la BD son siempre gente de confianza
- ▶ Mejor proteger los datos delicados de su vista (criptografía)
- ▶ Opciones propias
 - ▶ Campos PASSWORD
 - ▶ Mejor no fiarse: cifrar, codificar, almacenar, decodificar, descifrar ...Problemas: ¿y para buscar? ¿y las prestaciones? ¿y el espacio?

Ataques mediante análisis estadísticos

- ▶ Cuidado con el acceso que damos a nuestra base de datos
- ▶ Eliminar el acceso a datos delicados
- ▶ Por ejemplo, datos únicos

Ejemplo:

Si conozco a un cliente de Alfamén, de 72 años, y sólo hay uno con esas características, no importa que el nombre y la dirección no aparezcan.

Ataques mediante análisis estadísticos

- ▶ Una solución sería permitir sólo funciones agregadas (AVG, COUNT, MAX, MIN, SUM, ...)
- ▶ Pero se puede inferir información

Análisis estadísticos: ejemplo

```
SELECT COUNT(*)  
FROM clientes  
WHERE ciudad = "Alfamen"  
AND edad = 72;
```

Si da 1 ...

```
SELECT AVG(sueldo)  
FROM clientes  
WHERE ciudad = "Alfamen"  
AND edad = 72;
```

Analisis estadisticos

- ▶ Restringir las peticiones por debajo de determinados numeros
- ▶ Cuidado (claro) con sus complementarias

```
SELECT AVG(sueldo)  
  FROM clientes  
  WHERE NOT (ciudad = "Alfamen"  
             AND edad = 72);
```

Pero siempre se puede conseguir algo ...

Analisis estadisticos

```
SELECT COUNT(*)  
FROM clientes  
WHERE ciudad = "Calatayud"  
AND edad = 72:
```

Devuelve 100

```
SELECT AVG(sueldo)  
FROM clientes  
WHERE ciudad = "Calatayud"  
OR (ciudad = "Alfamen"  
AND edad = 72);
```

Y ya esta!

Como?



Ataques estadísticos

- ▶ Ataque directo. Si hay pocos datos, se calcula el agregado y se deducen datos.
- ▶ Ataque indirecto. Combinación de datos de diferentes agregaciones.
- ▶ Ataque mediante predicados trazadores
 $q = \text{count}(a \text{ and } b \text{ and } c) =$
 $\text{count}(a) - \text{count}(a \text{ and not}(b \text{ and } c)) =$
 $\text{count}(a) - \text{count}(a \text{ and } (\text{not } b \text{ or not } c))$
- ▶ Fallos mediante sistemas lineales

Algunas defensas

- ▶ Los datos sensibles no deberían estar disponibles
- ▶ Auditar las preguntas de vez en cuando
- ▶ Modificar la base de datos de forma que no afecte al análisis estadístico
- ▶ Añadir ruido estadístico (no sistemáticamente)
- ▶ No hay soluciones suficientemente buenas
- ▶ Disparadores (*'Triggers'*)
- ▶ ¿El futuro?

Privacidad

“OECD Guidelines on the Protection of Privacy and Transborder Flows of Personal Data”

http://www.oecd.org/document/18/0,3343,en_2649_34255_1815186_1_1_1_1,00.html

- ▶ Colección de datos limitada. Sólo los datos necesarios, con consentimiento, obtenidos legalmente...
- ▶ Datos de calidad. Relevantes, ajustados a la realidad, completos y al día
- ▶ Claramente especificado el uso de los datos
- ▶ Uso limitado. No se proporcionan a terceros sin permiso del usuario o de la ley
- ▶ Salvaguardado: niveles adecuados de protección y seguridad
- ▶ Principio de apertura. Debería ser fácil conocer qué datos se tienen de alguien, para qué se usan...
- ▶ Participación individual. La consecuencia del anterior: fácil eliminar los datos, corregirlos, ...
- ▶ Responsabilidad. El que tiene los datos es responsable del uso que se haga de ellos.

