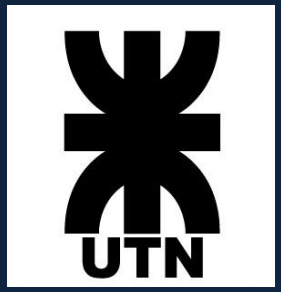
Seguridad



Usuarios

NUNCA JAMÁS

- Crear usuarios en la DB sin ningún password.
- Utilizar el usuario root de la DB en la aplicación.
- Utilizar un usuario con demasiados permisos en la aplicación.
- Utilizar passwords de diccionario para usuarios de la DB.
- Otorgar permisos a **mysql.users** a otro usuario que no sea root.
- Permitir que el usuario root se conecte por fuera de localhost.

Usuarios

SIEMPRE

- Crear un usuario con los permisos exactos (SELECT, INSERT, UPDATE, DELETE, etc) para usar en la aplicación.
- Crear un usuario con los permisos necesarios (y nunca los de control del motor) para administración por parte del programador.
- Utilizar el usuario root sólo en caso de extrema necesidad. El usuario root no puede bloquearse.

Usuarios

PREFERENTEMENTE...

- Limitar el acceso de cada usuario a nivel de hostname y/o IP.
- Limitar el acceso de cada usuario a la base de datos correspondiente.
- Dependiendo del caso, limitar el acceso de cada usuario incluso por tabla.
- Si el acceso es sólo lectura (réplicas, reportes), crear y utilizar un usuario sólo con permisos de SELECT.

Permisos

Privilege	Column	Context
CREATE	Create_priv	databases, tables, or indexes
DROP	Drop_priv	databases, tables, or views
GRANT OPTION	Grant_priv	databases, tables, or stored routines
LOCK TABLES	Lock_tables_priv	databases
REFERENCES	References_priv	databases or tables
EVENT	Event_priv	databases
ALTER	Alter_priv	tables
DELETE	Delete_priv	tables
INDEX	Index_priv	tables
INSERT	Insert_priv	tables or columns
SELECT	Select_priv	tables or columns
UPDATE	Update_priv	tables or columns
CREATE TEMPORARY TABLES	Create_tmp_table_priv	tables
TRIGGER	Trigger_priv	tables
CREATE VIEW	Create_view_priv	views
SHOW VIEW	Show_view_priv	views

Permisos

ALTER ROUTINE	Alter_routine_priv	stored routines
CREATE ROUTINE	Create_routine_priv	stored routines
EXECUTE	Execute_priv	stored routines
FILE	File_priv	file access on server host
CREATE USER	Create_user_priv	server administration
PROCESS	Process_priv	server administration
RELOAD	Reload_priv	server administration
REPLICATION CLIENT	Repl_client_priv	server administration
REPLICATION SLAVE	Repl_slave_priv	server administration
SHOW DATABASES	Show_db_priv	server administration
SHUTDOWN	Shutdown_priv	server administration
SUPER	Super_priv	server administration
ALL [PRIVILEGES]		server administration
USAGE		server administration

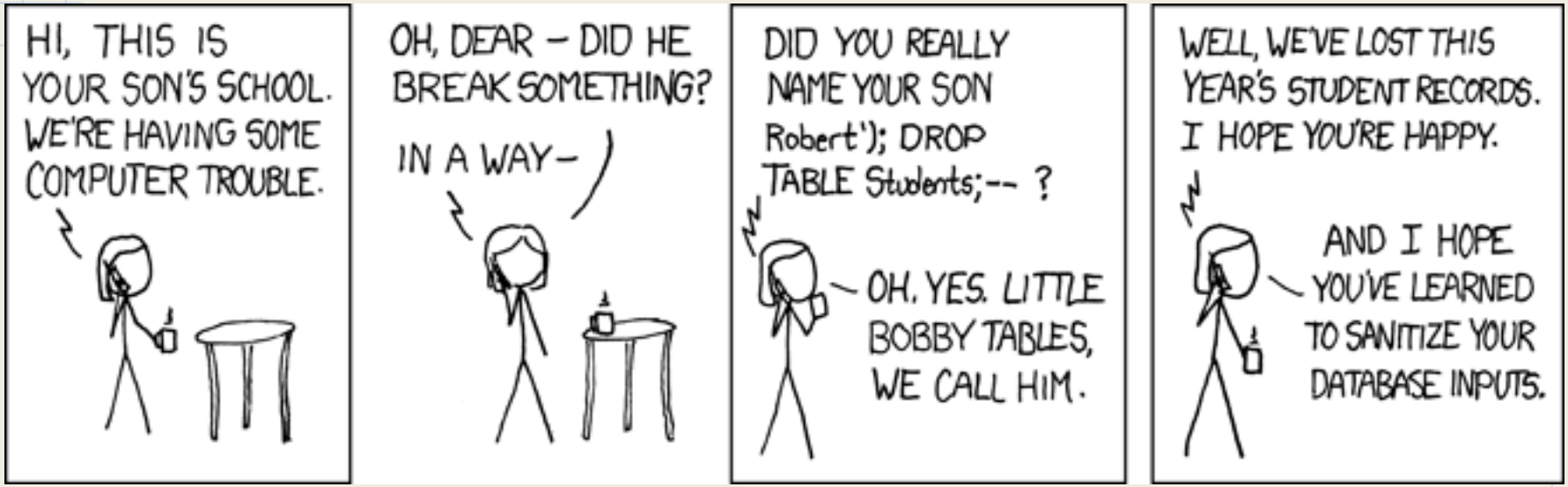
Networking

- Siempre colocar la base de datos a través de un **firewall.**
- Si el servidor está expuesto (no es interno, ni está en una VPN), limitar las conexiones por IP (con iptables o similar)
- Cambiar el puerto del server de la base de datos afecta poco y nada a la seguridad.

Networking (Encriptación)

- Si los usuarios se conectan a la DB fuera de “localhost”, encriptar la conexión con **SSL** (interno de MySQL, built-in, transparente al usuario), o **SSH para mayor seguridad**.
    - La encriptación de transferencia es para evitar “middle-man attacks”.
    - La encriptación de los datos (guardados o transferidos) depende de los endpoints y nada tiene que ver con la parte de networking.

SQL Injection



La mayoría de las librerías (MySQLdb) y ORM (sqlobject) protegen contra los casos típicos de SQL injection si se utilizan bindings.

Comic credit: http://xkcd.com/327/ “Exploits of a mom”

Passwords (y otra información sensible)

NUNCA JAMÁS

- Guardar passwords en texto plano
- Guardar passwords usando la función PASSWORD() de MySQL.
- Guardar passwords sólo hasheados.
- Dependiendo del caso, nunca guardar passwords de forma reversible.

Hashing

El hashing es un proceso en el que un algoritmo mapea datos de longitud variables en **“hashes” o “checksums”** de longitud fija.

Algoritmos

- **MD5** (128 bits), **SHA1** (160 bits), **SHA2** (hasta 512 bits)

Características

- Misma entrada, misma salida.
- Entrada similar, salida totalmente diferente.
- “Irreversible”, pero verificable.
- Baja probabilidad de colisiones (1 en $2^{n\text{-bits}}$)
- Susceptible a ataques de diccionario.

Hashing + Salting

Usar hashing para procesar passwords ya no es suficiente

Salting es el proceso de agregar un **componente aleatorio** al hashing que:

- Para la misma entrada, produce resultados siempre distintos.
- Es verificable sin necesidad de revertir hasta el password original.

La implementación recomendada en la actualidad es **bcrypt**, porque:

Es demasiado lenta para ser atacada mediante fuerza bruta.

Lo que significa que:

Es sólo cuestión de tiempo, hasta contar con el hardware necesario.

Preguntas

?