Què és la injecció SQL? Per què és possible? Com evitar-la?

Aquesta presentació explora la injecció SQL, una tècnica d'atac que pot comprometre la seguretat de les bases de dades. Aprendrem què és,per què succeeix i com prevenir-la.

Què és una Injecció SQL?

Definició

Tècnica d'atac on s'introdueix codi maliciós en camps d'entrada d'una aplicació web.

Objectiu

Manipular consultes consultes SQL al servidor per accedir, accedir, modificar o o esborrar informació sensible. sensible.

Exemple senzill de consulta SQL

Consulta Original

SELECT * FROM users WHERE username = 'usuari' AND password = = 'contrasenya';

Atac

Entrada maliciosa: ' OR '1'='1

Resultat

L'atacant accedeix al sistema sense conèixer les les credencials.

Per què és possible una injecció SQL? SQL?

Falta de validació

Dades d'entrada no verificades verificades ni netejades.

Consultes estàtiques

Construcció dinàmica de consultes amb cadenes de text.

Errors detallats

Exposició de missatges d'error que revelen sintaxi SQL.

Configuracions insegures

Privilegis excessius a usuaris o comptes.

Conseqüències d'una injecció SQL

Accés no autoritzat

Exposició de dades sensibles.

Modificació de dades

Canvis en informació crítica.

Destrucció de dades

Eliminació de taules o bases de dades.

Execució de comandes

Accés a l'entorn del servidor.

Com evitar una injecció SQL?

1

Consultes preparades

Separar dades i codi amb Prepared Statements.

2

Validació de dades

Netejar i validar les dades d'entrada dels usuaris.

3

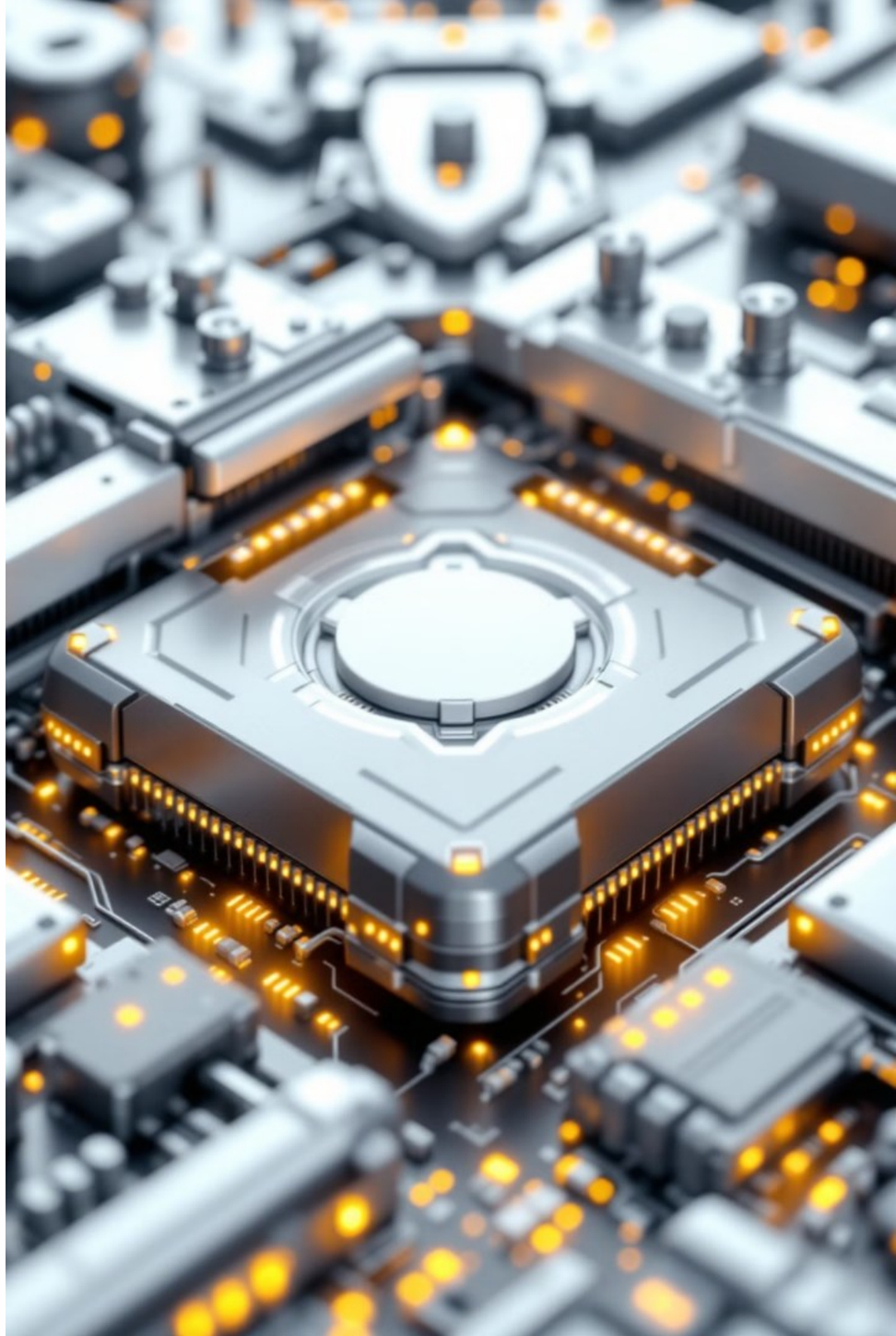
Utilitzar ORM

Eines que generen consultes SQL segures automàticament.

4

Limitar privilegis

Assignar només els permisos necessaris a cada aplicació.



Exemple de codi segur

Consulta insegura

```
$sql = "SELECT * FROM users
WHERE username = '$username'
AND password = '$password'";
```

Consulta segura

```
$stmt = $pdo->prepare("SELECT
* FROM users WHERE username =
:username AND password =
:password");
$stmt->execute(['username' =>
$username, 'password' =>
$password]);
```