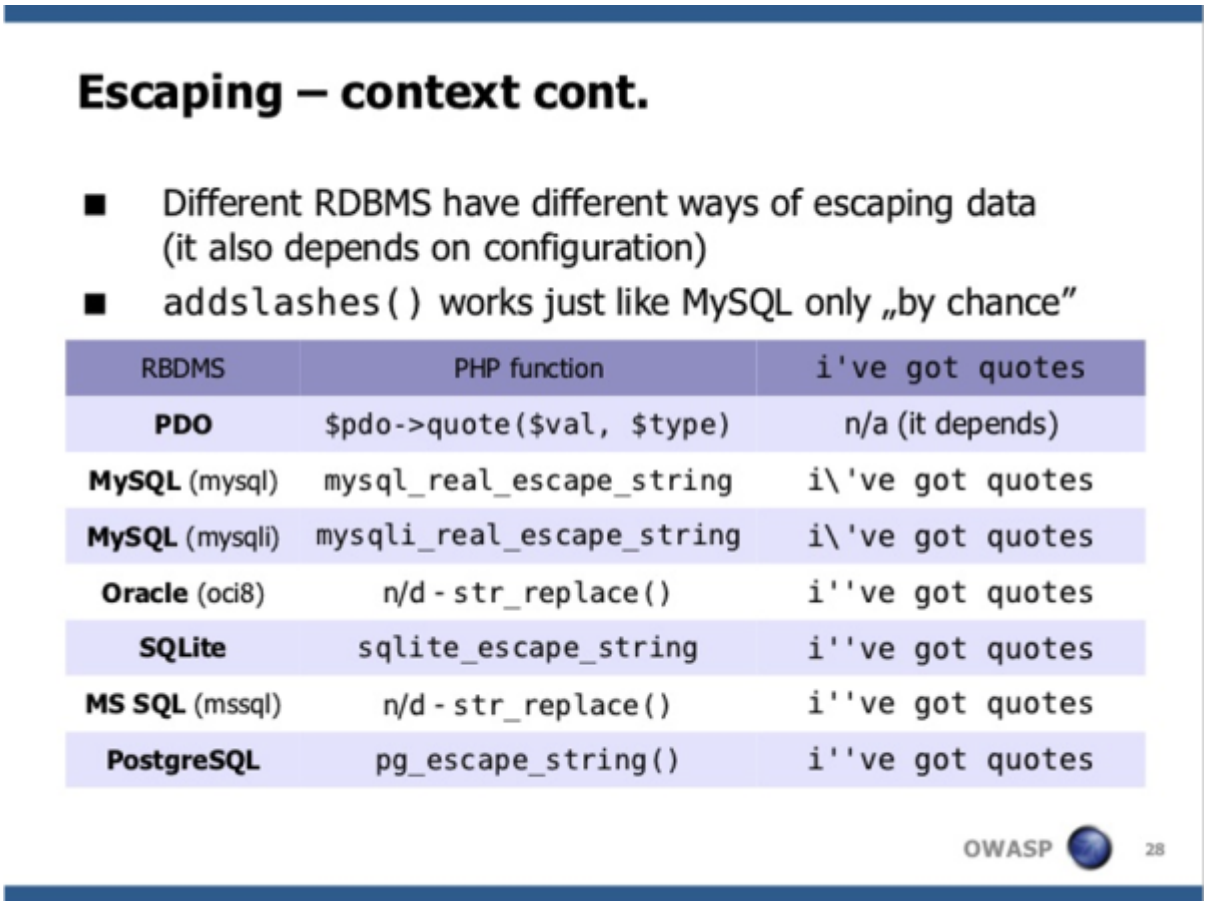
LU10c - SQLi Gegenmassnahmen

Glücklicherweise gibt gegen alle SQLi-Varianten entsprechende Gegenmassnahmen. Die hier vorliegende Liste ist daher nur ein Ausschnitt mit den prominentesten.

Escaping

Beim *escaping* werden Sonderzeichen nicht direkt in die Datenbank weitergeleitet. Die vom User eingegebenen Werte werden durch eine entsprechende Funktion/Methode, die es in vielen Programmiersprachen gibt vor der Weiterleitung gefiltert. Sonderzeichen, die den Angriffspunkt darstelle, werden von *gefährlichen* Steuerzeichen in *harmlose* Characters umgewandelt.Das „Escapen“ kann dabei auf der Client- und Serverseite geschehen.



Escaping – context cont.

- Different RDBMS have different ways of escaping data (it also depends on configuration)
- `addslashes()` works just like MySQL only „by chance”

RBDMS	PHP function	i've got quotes
PDO	`$pdo->quote($val, $type)`	n/a (it depends)
MySQL (mysql)	`mysql_real_escape_string`	i\'ve got quotes
MySQL (mysqli)	`mysqli_real_escape_string`	i\'ve got quotes
Oracle (oci8)	n/d - `str_replace()`	i''ve got quotes
SQLite	`sqlite_escape_string`	i''ve got quotes
MS SQL (mssql)	n/d - `str_replace()`	i''ve got quotes
PostgreSQL	`pg_escape_string()`	i''ve got quotes

OWASP 28

Das Hauptproblem bei SQLI besteht darin, dass Werte ungeprüft in die Datenbank weitergegeben werden, wobei Platzhalter mit vom User zur Laufzeit eingegebenen Werten angereichert werden. Enthält eine solcher zur Laufzeit generierte SQL-Anweisung Injektion, wird diese ungeprüft und ungepuffert an die DB weitergeleitet.

Parametrisierte Werteübergabe

Die *parametrisierte Werteübergabe*, kurz *Parametrisierung* genannt, kann eine mögliche Gegenmassnahme gegen SQLI sein. Dabei werden die von User eingegebenen Werte in lokale

Parameter geschrieben. Diese Werte werden dann einen Schritt später durch Platzhalter in das SQL-Statement eingefügt.

Das nachfolgende JavaScript/MySQL-Beispiel soll diese Technik verdeutlichen:

```
// mit mysql2 (Promise-API)
const sql1 = `UPDATE Production.ProductInventory
              SET Quantity = ?
              WHERE ProductID = ?`;
const [result] = await connection.execute(sql1, [quantity, productId]);
// connection.execute schützt vor SQLi, Platzhalter = ?
```

Volkan Demir

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/modul/m183/learningunits/lu10/03?rev=1758268712

Last update: **2025/09/19 09:58**