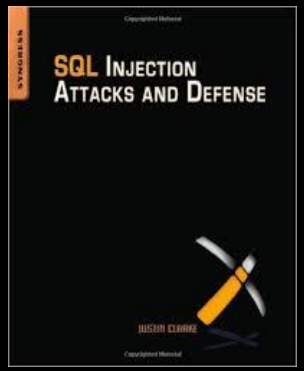
SQL Injection



SQL injection is one of the most devastating vulnerabilities to impact a business, as it can lead to exposure of all of the sensitive information stored in an application's database.

What Is SQL Injection?

It is the vulnerability that results when you give an attacker the ability to influence the Structured Query Language (SQL) queries that an application passes to a back-end database.

- Usernames.
- Passwords.
- Names.
- Addresses.
- Phone numbers.
- Credit card details.

SQL injection is not a vulnerability that exclusively affects **Web applications**; **any code that accepts input from an untrusted source** and then uses that input to form dynamic SQL statements could be vulnerable.

Web Applications 2.0

They normally consist of a back-end database with Web pages that contain server-side script written in a programming language that is capable of extracting specific information from a database depending on various dynamic interactions with the user

e-commerce application

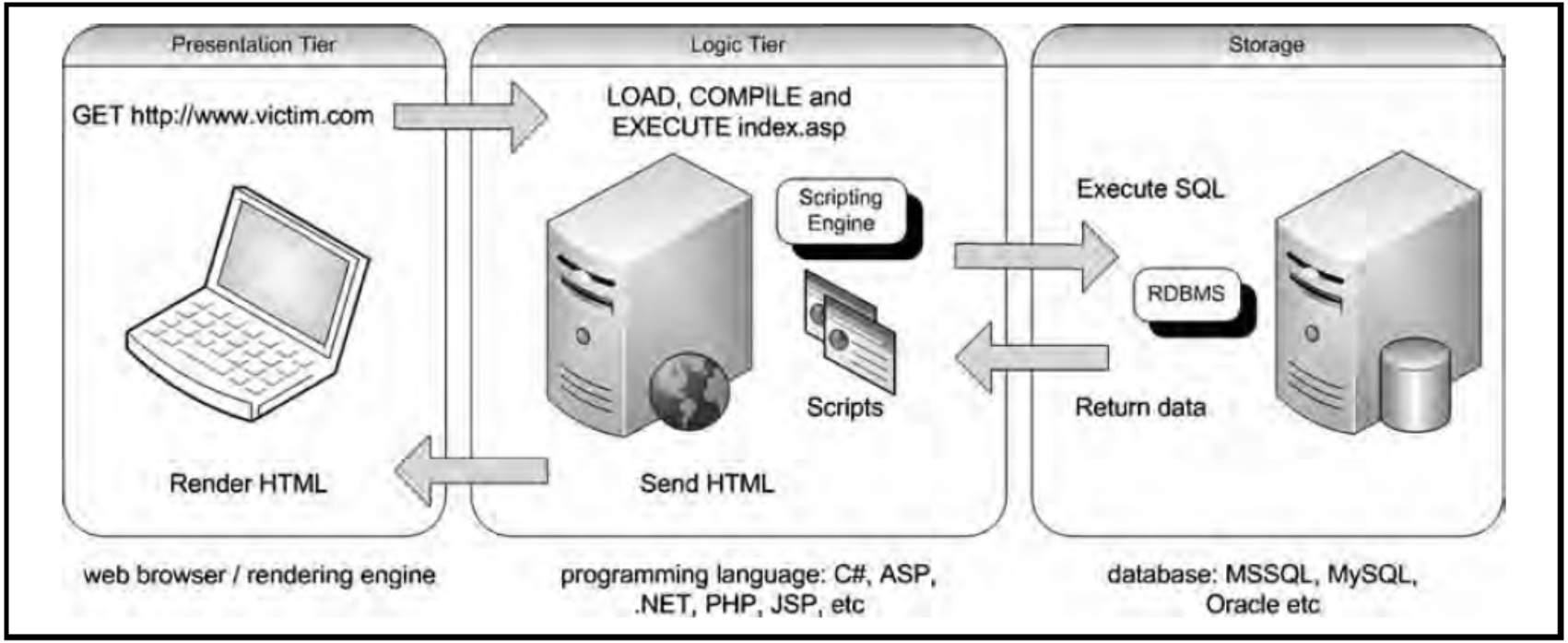
A variety of information is stored in a database.

- Product information.
- Stock levels.
- Prices.
- Postage.
- Packing costs.

Web application commonly has three tiers

Presentation	Web browser or Rendering engine, such as Internet Explorer, Safari, Firefox, etc.
Logic	programming language, such as C#, ASP, .NET, PHP, JSP, etc.
Storage	a database such as Microsoft SQL Server, MySQL, Oracle, etc.

Figure 1.1 Simple Three-Tier Architecture



This tier keeps data independent from application servers or business logic. Giving data its own tier also improves scalability and performance.

The presentation tier sends requests to the middle tier which services the requests by making queries and updates against the database.

Three-tier solutions are not scalable, so in recent years the three-tier model was reevaluated and a new concept built on scalability and maintainability was created: the n-tier application development paradigm.


PROTOCOLO
PROTOCOLO
http://www.victim.com/products.php?val=100
PROTOCOLO
PROTOCOLO

```
$conn = mysql_connect("localhost","username","password");

$query = "SELECT * FROM Products WHERE Price <
'$_GET["val"]' " .

"ORDER BY ProductDescription";

$result = mysql_query($query);

while($row = mysql_fetch_array($result, MYSQL_ASSOC))

{

echo "Description : {$row['ProductDescription']} <br>"

"Product ID : {$row['ProductID']} <br>"

"Price : {$row['Price']} <br><br>"

}
```

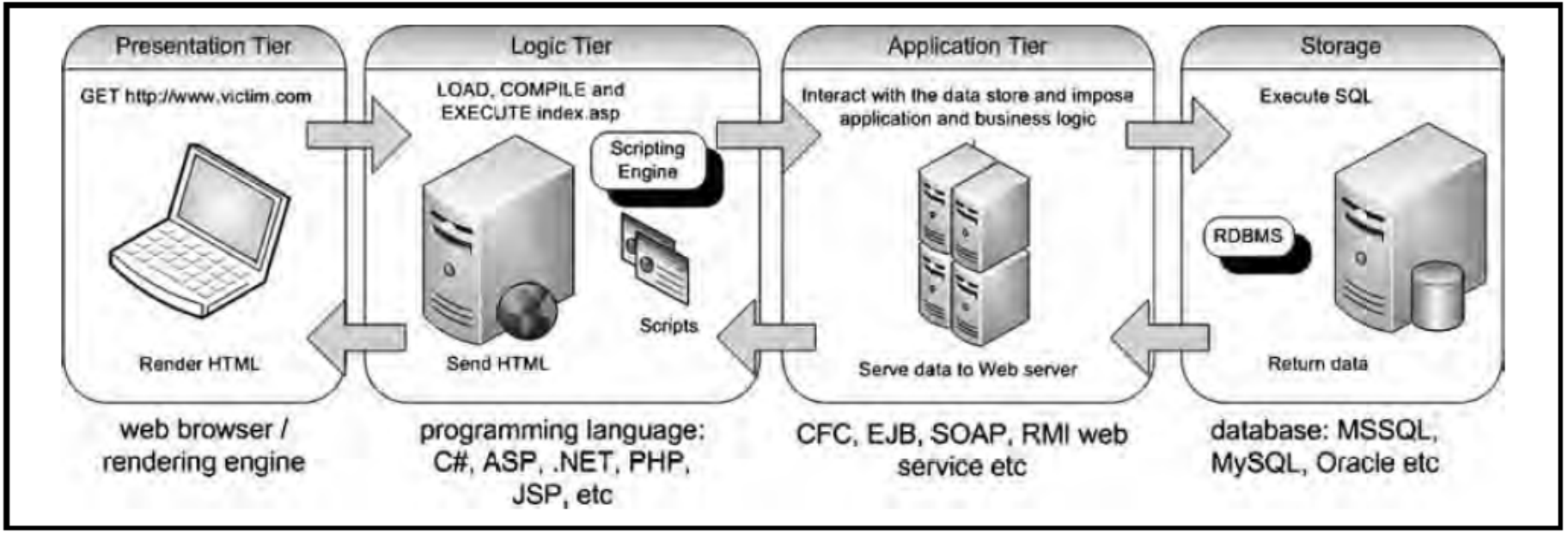
The following PHP script illustrates how the user input (val) is passed to a dynamically created SQL statement.

Within this a four-tier solution was devised that involves the use of a piece of

Middleware, typically called an application server, between the Web server and the database.

In an n-tier architecture is a server that hosts an application programming interface (API) to expose business logic and business processes for use by applications.

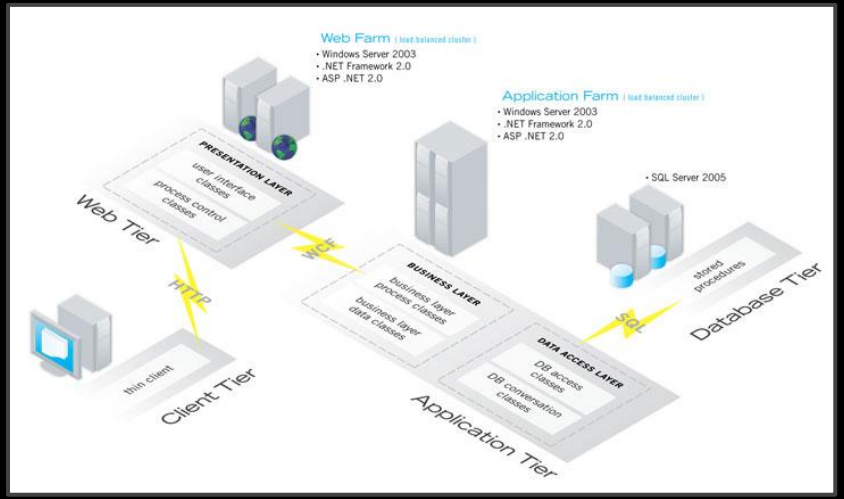
Figure 1.2 Four-Tier Architecture



Web browser (presentation) sends requests to the middle tier (logic), which in turn calls the exposed APIs of the application server residing within the application tier, which services them by making queries and updates against the database (storage).

Separating the responsibilities of an application into multiple tiers makes it easier to scale the application, allows for better separation of development tasks among developers, and makes an application more readable and its components more reusable.

The approach can also make applications more robust by eliminating a single point of failure.



Three-tier and four-tier architectures are the most commonly deployed architectures on the Internet today...however, the n-tier model is extremely flexible.

OWASP CHEAT SHEET

OWASP Top Ten Cheat Sheet

A1 Injection	A6 Sensitive Data Exposure
A2 Weak authentication and session management	A7 Missing Function Level Access Control
A3 XSS	A8 Cross Site Request Forgery
A4 Insecure Direct Object References	A9 Using Components with Known Vulnerabilities
A5 Security Misconfiguration	A10 Unvalidated Redirects and Forwards

https://www.owasp.org/index.php/OWASP_Top_Ten_Cheat_Sheet

A1 Injection

Render:

- Set a correct content type
- Set safe character set (UTF-8)
- Set correct locale

On Submit:

- Enforce input field type and lengths.
- Validate fields and provide feedback.
- Ensure option selects and radio contain only sent values.

A2 Weak authentication and session management

Render:

- Validate user is authenticated.
- Validate role is sufficient for this view.
- Set "secure" and "HttpOnly" flags for session cookies.
- Send CSRF token with forms.

A3 XSS

Render:

- Set correct content type
- Set safe character set (UTF-8)
- Set correct locale
- Output encode all user data as per output context
- Set input constraints

On Submit:

- Enforce input field type and lengths.
- Validate fields and provide feedback.
- Ensure option selects and radio contain only sent values.