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

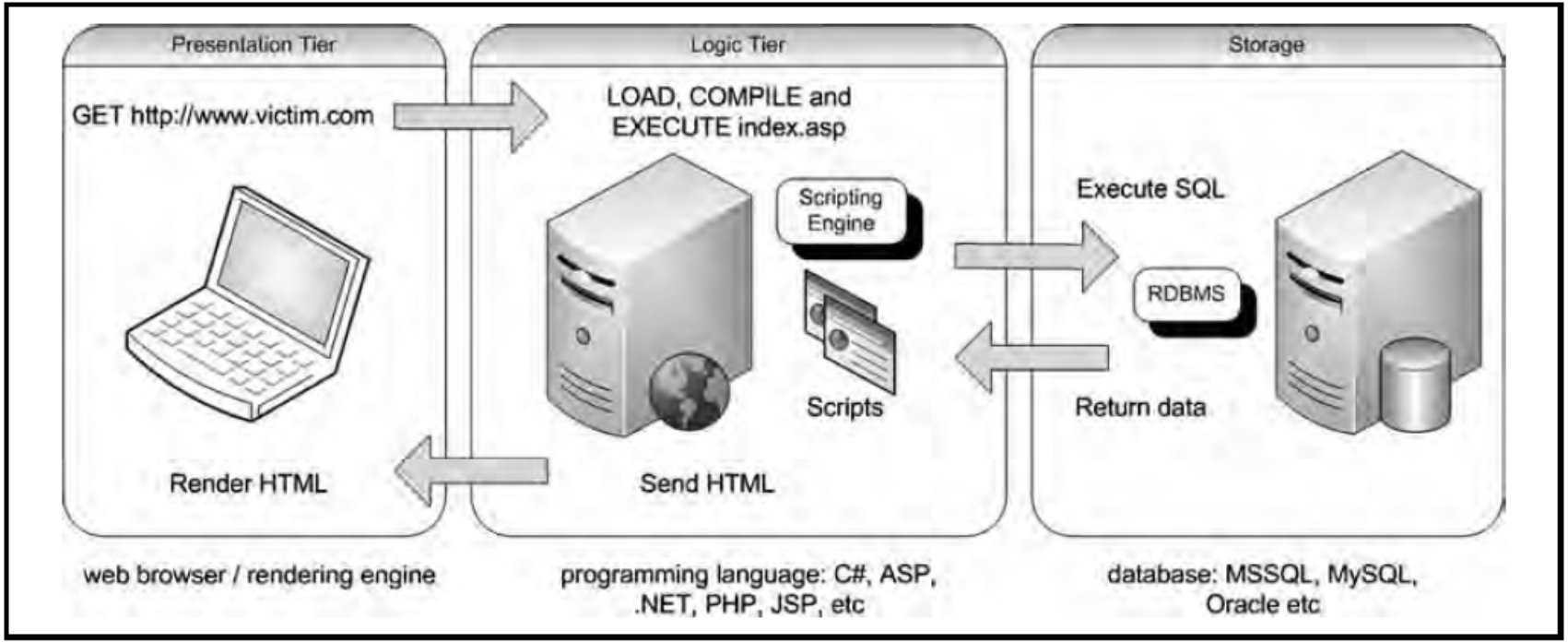
A variety of information is stored in a database.

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

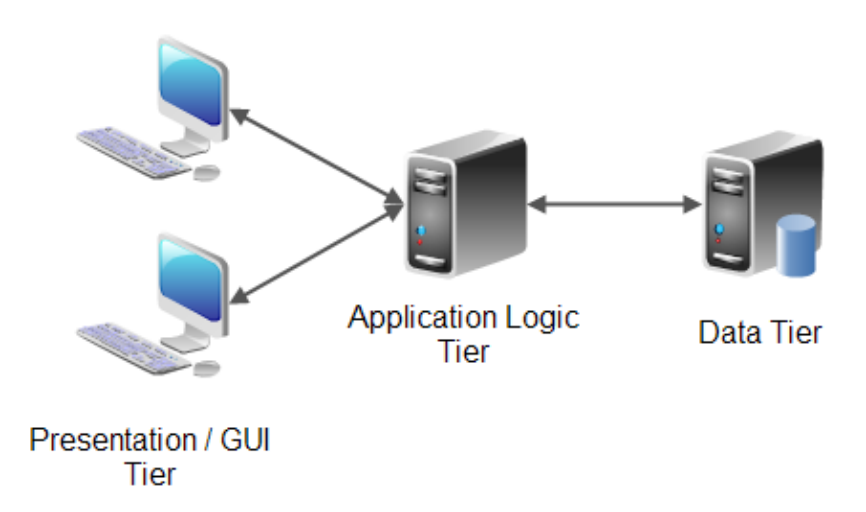
Web application commonly has three tiers

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);
```

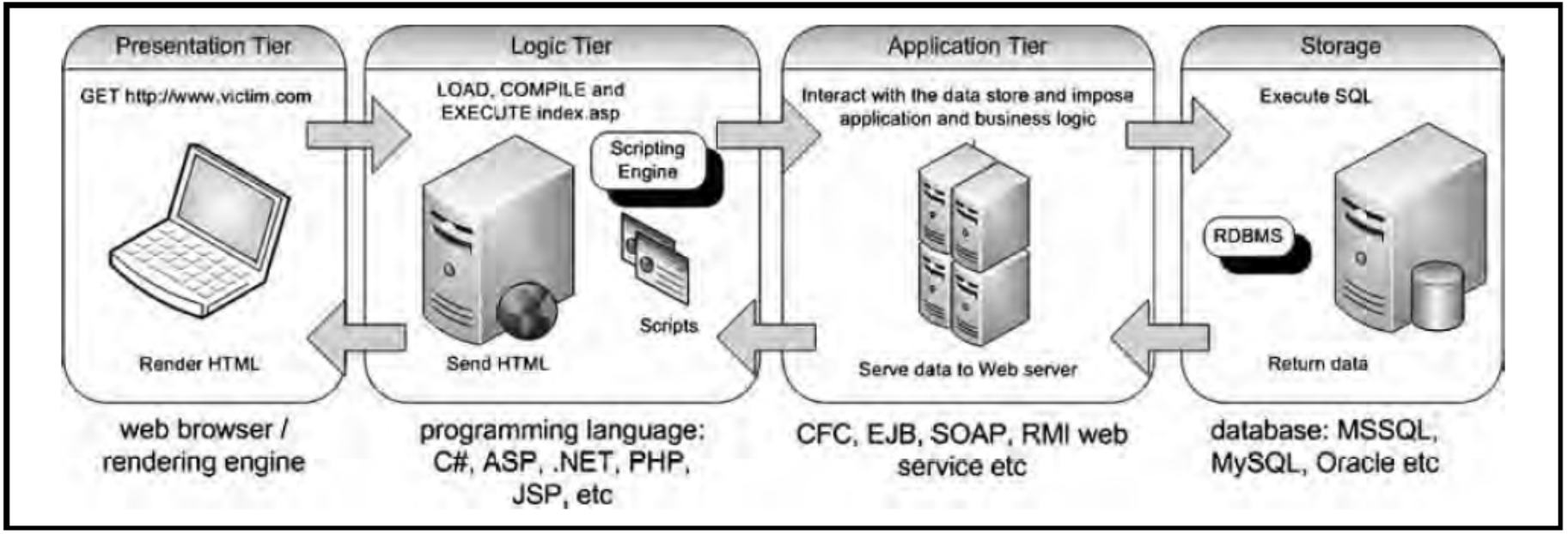
The following PHP script illustrates how the user input (val) is passed to a dynamically created SQL statement.

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.

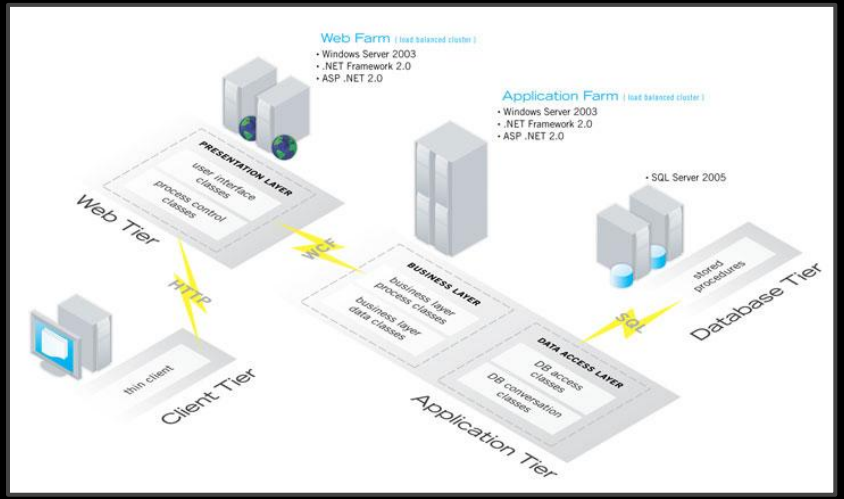
Figure 1.2 Four-Tier Architecture



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.

GET & POST Requests

GET is an HTTP method that requests to the server whatever information is indicated in the URL.

For GET requests, you can manipulate the parameters by simply changing them in your browser's navigation toolbar.

POST is an HTTP method used to send information to the Web server.This is normally the method used when you fill in a form in your browser and click the Submit button.

http://www.victim.com/cms/login.php?username=foo&password=bar

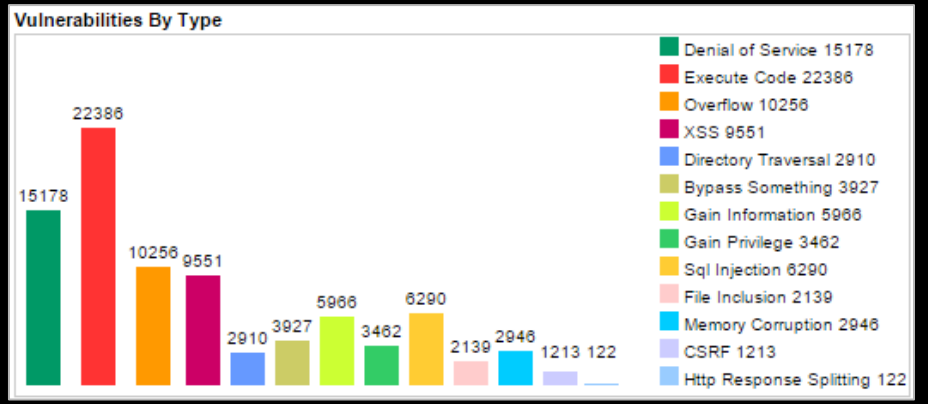
if you explicitly SAY method="POST", then, surprisingly, it's a POST.

```
<form method="POST"  action="SelectBeer.do">
  Select beer characteristics<p>
  <select name="color" size="1">
    <option>light
    <option>amber
    <option>brown
    <option>dark
  </select>
  <center>
    <input type="SUBMIT">
  </center>
</form>
```

When the user clicks the "SUBMIT" button, the parameters are sent in the body of the POST request. In this example, there's just one parameter, named "color", and the value is the <option> beer color the user selected (light, amber, brown, or dark).

CVE Details

The ultimate security vulnerability datasource



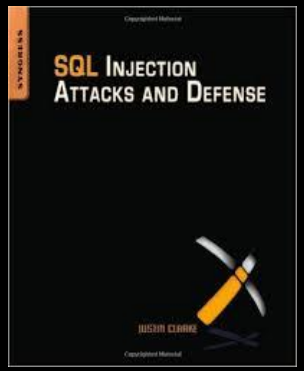
https://www.cvedetails.com/vulnerabilities-by-types.php

Vulnerabilities By Type

Year	# of Vulnerabilities	DoS	Code Execution	Overflow	Memory Corruption	Sql Injection	XSS	Directory Traversal	Http Response Splitting	Bypass something	Gain Information	Gain Privileges	CSRF	File Inclusion	# of exploits
1999	894	177	112	172			2	7		25	13	102			2
2000	1020	257	207	206		2	4	20		48	19	138			
2001	1677	402	402	297		7	34	123		83	35	219		2	2
2002	2156	497	553	435	2	41	200	103		127	74	199	2	14	2
2003	1526	381	476	370	2	49	129	60	1	61	69	144		16	8
2004	2450	580	614	409	3	148	291	110	12	145	96	134	5	38	13
2005	4934	838	1625	653	21	604	786	202	15	289	261	221	11	100	25
2006	6610	893	2719	662	91	967	1302	321	8	267	267	184	18	849	97
2007	6520	1100	2601	952	95	706	883	339	14	267	322	242	69	700	1251
2008	5632	894	2310	699	128	1101	807	363	7	288	270	188	83	170	1936
2009	5736	1035	2184	700	188	963	851	322	9	338	302	223	115	138	2108
2010	4651	1102	1714	677	342	520	605	275	8	234	282	237	86	73	1469
2011	4155	1221	1334	770	351	294	467	108	7	197	409	206	58	17	564
2012	5297	1425	1457	844	423	242	758	122	13	343	389	250	166	14	615
2013	5191	1453	1186	859	366	155	650	110	7	350	510	274	123	1	202
2014	7946	1597	1573	849	420	304	1107	204	12	457	2104	239	264	2	391
2015	4941	1326	1319	702	514	187	675	121	9	408	544	262	213	5	118
Total	71336	15178	22386	10256	2946	6290	9551	2910	122	3927	5966	3462	1213	2139	8803
% Of All		21.3	31.4	14.4	4.1	8.8	13.4	4.1	0.2	5.5	8.4	4.9	1.7	3.0	

https://www.cvedetails.com/vulnerabilities-by-types.php

SQL Injection

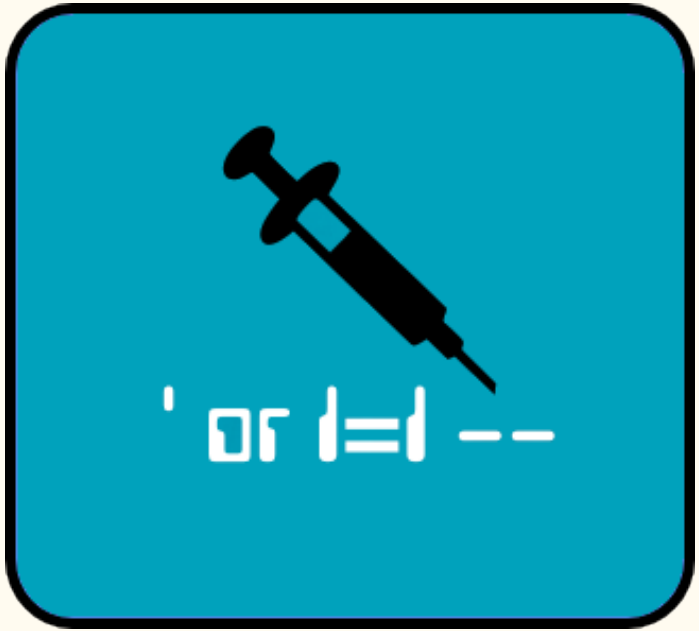


SQL injection is one of the most devastating vulnerabilities to impact a business, as it can lead to exposure of all of the sensitive information stored in an application's database.

What Is SQL Injection?

Is an attack in which SQL code is inserted or appended into application user input parameters that are later passed to a back-end SQL server for parsing and execution.

- 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.

‘ OR ‘1’= ‘1

- The primary form of SQL injection consists of direct insertion of code into parameters that are concatenated with SQL commands and executed.
- Malicious code into strings that are destined for storage in a table or as metadata.

http://www.victim.com/products.php?val=100’ OR ‘1’=’1

```
// connect to the database

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

// dynamically build the sql statement with the input

$query = "SELECT userid FROM CMSUsers WHERE user =
'$_GET["user"]' " "AND password = '$_GET["password"]'";
```

http://www.victim.com/cms/login.php?username=foo&password=bar' OR '1'='1

The problem with the code is that the application developer believes the number of records returned when the script is executed **will always be zero or one.**

The logic of the application means that **if the database returns more than zero records, we must have entered the correct authentication** credentials and should be redirected and given access to the protected site.

We will normally be logged in as the first user in the table.

SQL injection vulnerabilities most commonly occur when

Web application developer does not ensure that values received from a Web form, cookie, input parameter, and so forth **are validated or encoded before passing them to SQL queries** that will be executed on a database server.0

If an attacker can control the input that is sent to an SQL query and manipulate that input so that **the data is interpreted as code instead of as data,** he may be able to execute code on the back-end database.

OWASP

The Open Web Application Security Project

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.

34706

Exploits Archived

https://www.exploit-db.com/

http://www.zone-h.org/

ACERCA DE MI

- Ingeniero en Sistemas Informáticos Universidad Central del Este (UCE)
- Profesor Universidad Dominicana O&M
- Maestria en Gerencia y Productividad - Universidad APEC
- Postgrado de Auditoria de Sistemas - Universidad O&M
- Comptia Security+ Certified

@ksanchez_cld

ksanchez_cld