

# 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



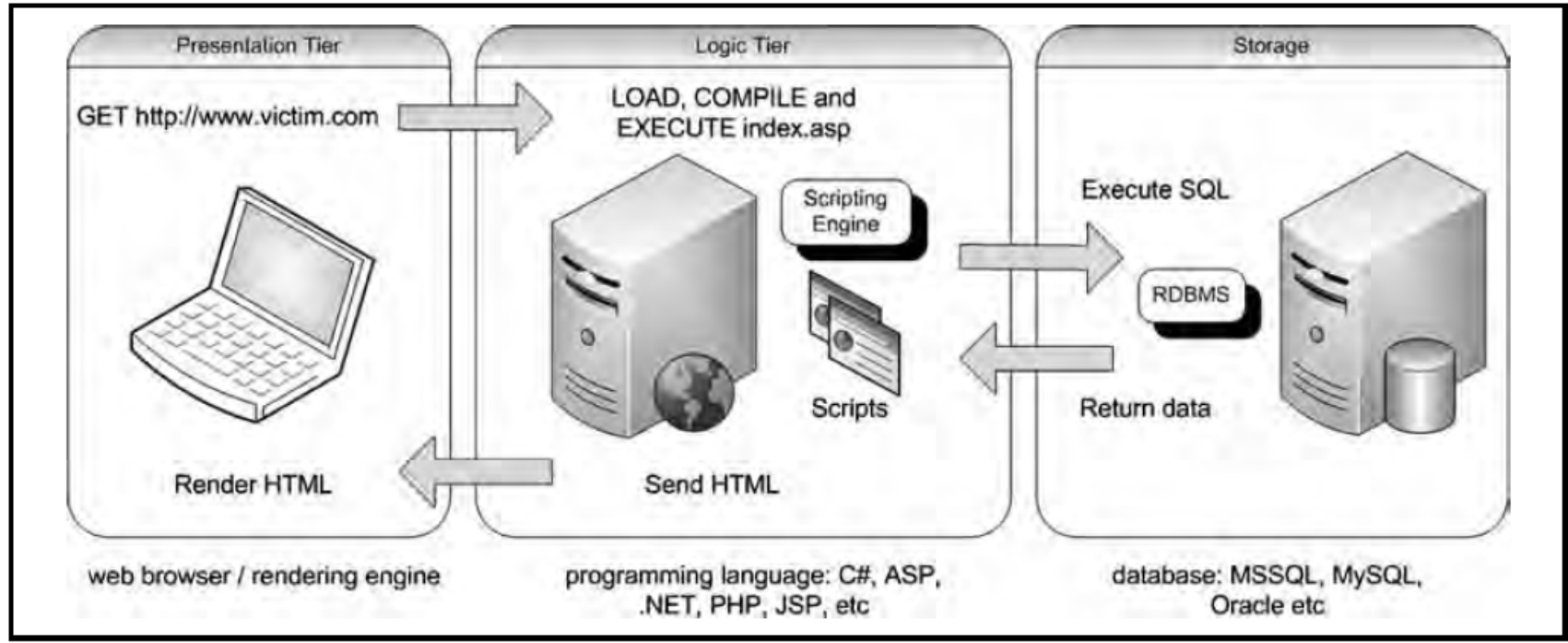
A variety of information is stored in a database.

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

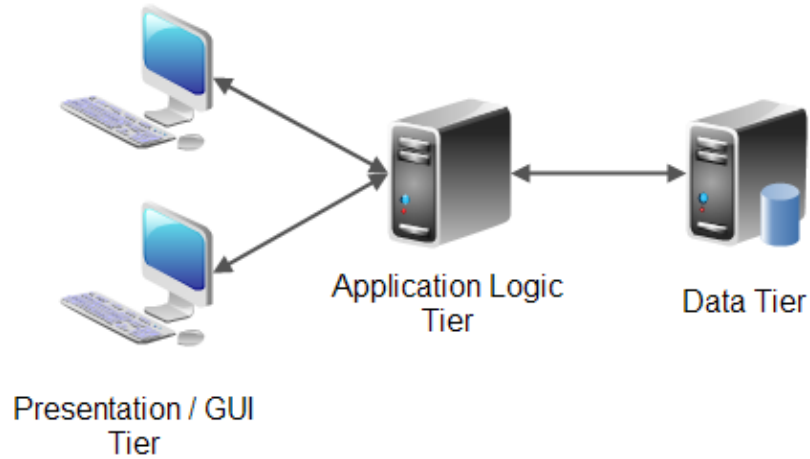
## Web application commonly has three tiers

<b>Presentation</b>	Web browser or Rendering engine, such as Internet Explorer, Safari, Firefox, etc.
<b>Logic</b>	programming language, such as C#, ASP, .NET, PHP, JSP, etc.
<b>Storage</b>	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.



```
$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.

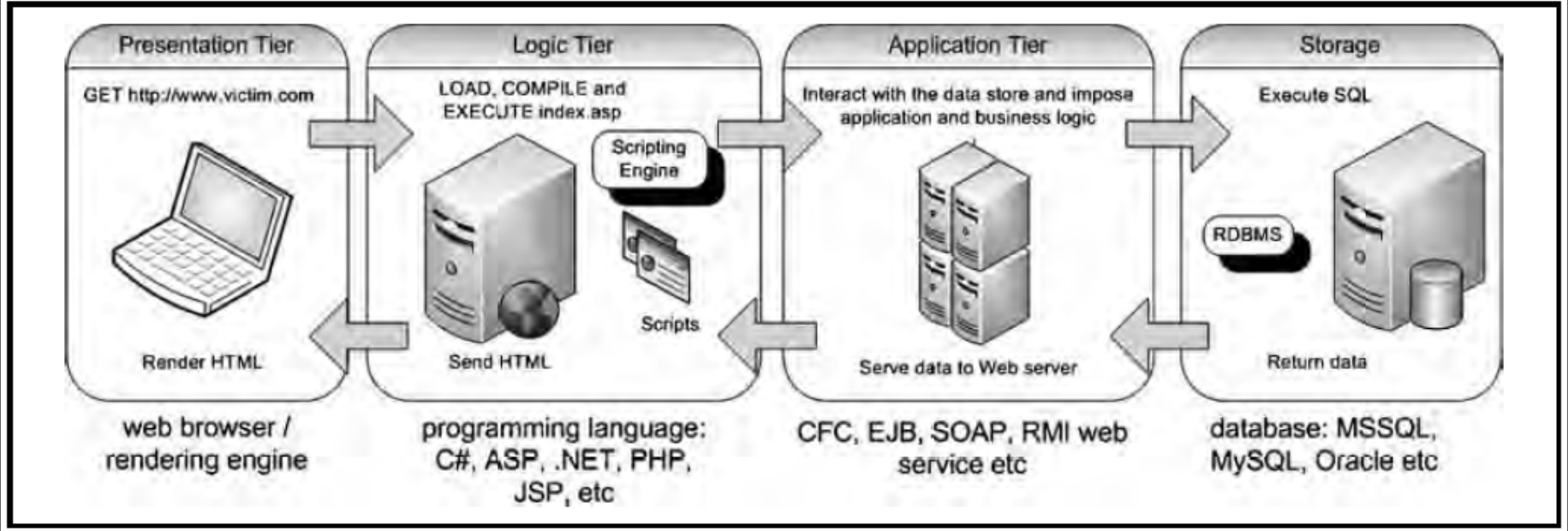


Within this a four-tier solution was devised that involves the use of a piece of

**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**



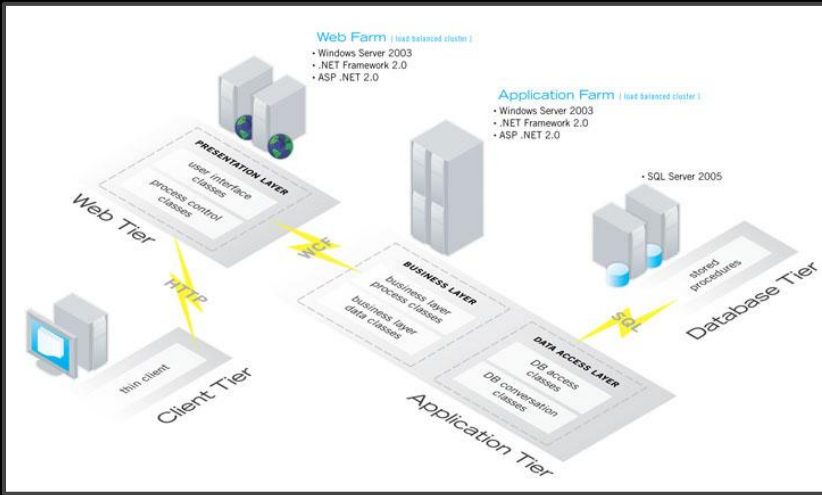
Web browser (presentation) sends requests to the middle tier (logic), which in turn calls the exposed APIs of the application server residing within the application tier, which services them by making queries and updates against the database (storage).



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.**

---



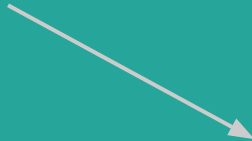
Three-tier and four-tier architectures are the most commonly deployed architectures on the Internet today...however, the n-tier model is extremely flexible.

# 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.

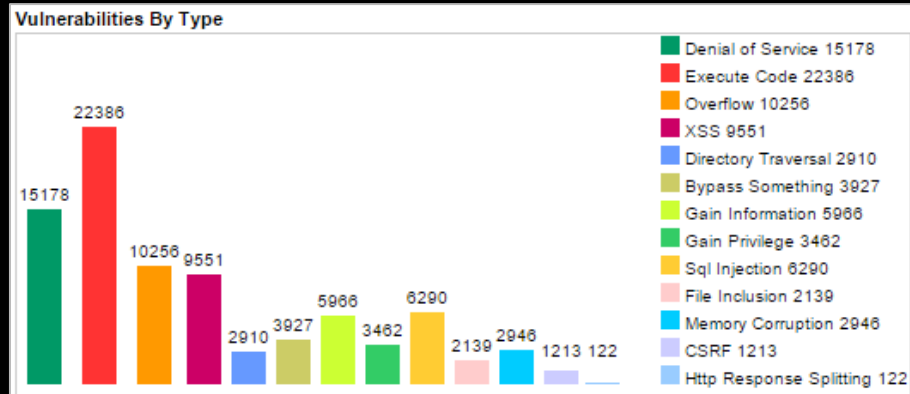
```
<form method="POST" action="SelectBeer.do">
  Select beer characteristics<p>
  <select name="color" size="1">
    <option>light
    <option>amber
    <option>brown
    <option>dark
  </select>
  <center>
    <input type="SUBMIT">
  </center>
</form>
```

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>

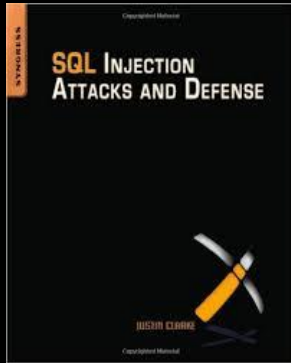




## Vulnerabilities By Type

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

# SQL Injection

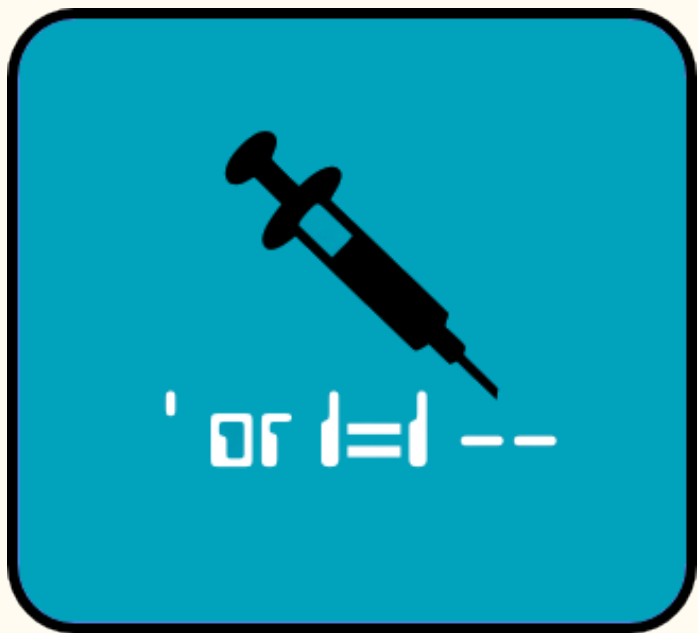


SQL injection is one of the most devastating vulnerabilities to impact a business, as it can lead to exposure of all of the sensitive information stored in an application's database.

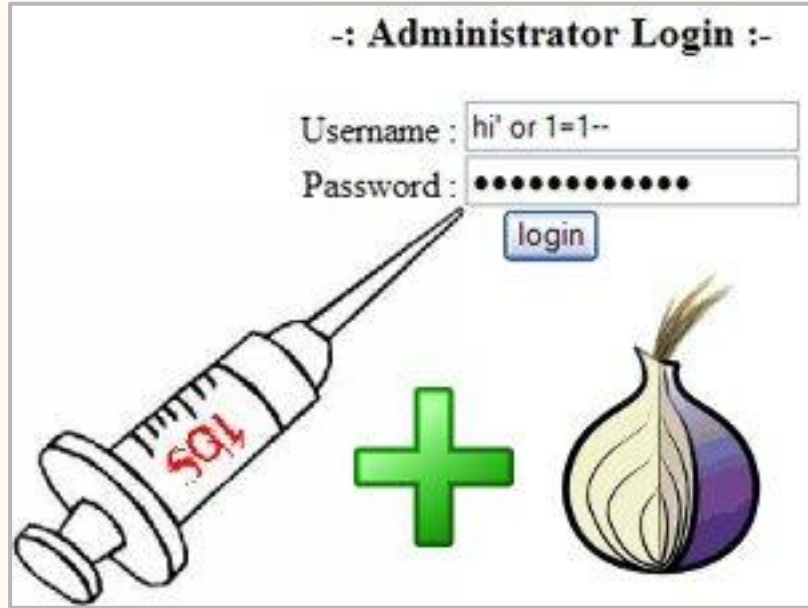
# What Is SQL Injection?

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.

```
// 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

# 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/>

## The Exploit Database

The Exploit Database (EDB) is a CVE compliant archive of exploits and vulnerable software. A great resource for penetration testers, vulnerability researchers, and security addicts alike. Our goal is to collect exploits from various sources and concentrate them in one, easy to navigate database

[Download the Exploit Database Archive](#)

# EXPLOIT DATABASE




## CVE Compliant




[Home](#)[News](#)[Events](#)[Archive](#)[Archive](#) ★[Onhold](#)[Notify](#)[Stats](#)[Register](#)[Login](#)

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

[http://www.cideko.com/pro\\_con.php?id=3](http://www.cideko.com/pro_con.php?id=3)




cideko  
BEYOND DIGITAL




**Cideko will take you beyond digital**

Air Keyboard is a wireless keyboard integrated with gyro mouse. Control the device by 3D motion sensing mouse and move the cursor in the air.

[Download Product PDF](#)



Language:  English

Home

About us

**Products**

News

Download

Contact us

FAQ

**Air Keyboard**

Combination of mini wireless keyboard and 3D gyro mouse, AVK02 Air Keyboard represents innovative technology. QWERTY standard layout and hot keys on the top brings comfort and convenience.


**Specification**

Item No.: AK02

**Feature :**

- Motion sensing air mouse
- Left and right mouse buttons
- LED indicators : low power(red) & PC mode (green)

**Photo Gallery**



Previous / Next

**Download**

Related Download

Download More

You have an error in your SQL syntax near "\" at line 1



## 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

twitter



@ksanchez\_cld

skype

ksanchez\_cld

