SQL Injection

Michael Nicholson
Alex Lopez

What is SQL

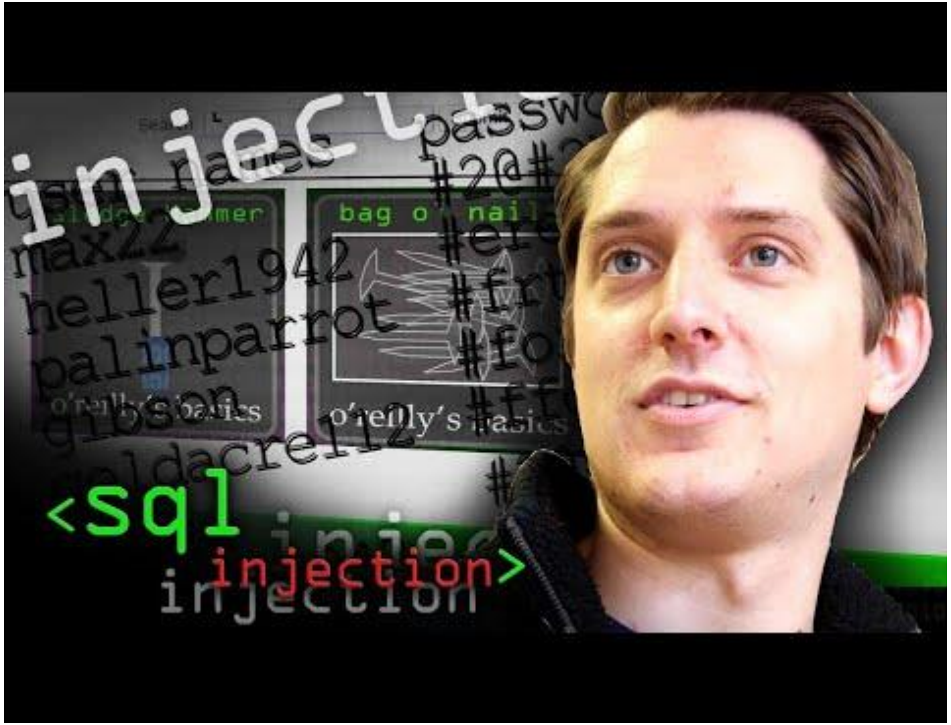
- SQL- Structured Query Language
- It is the primary way of creating and maintaining databases.
- Almost all data forms are held within SQL, or SQL derivative, controlled databases.
- SQL in itself is one of cybersecurity’s biggest vulnerabilities.

SQL Injections

- SQL injection is a hacking technique that takes advantage of vulnerabilities found in webpage inputs to run malicious code in SQL statements.
- It is one of the most common hacking techniques.
- They can be used to obtain unauthorized access into different parts of a database and/or edit and view the database.
- If this vulnerability is not protected against, it can lead to sensitive information being stolen and/or the database being destroyed.

How are SQL Injections Used?

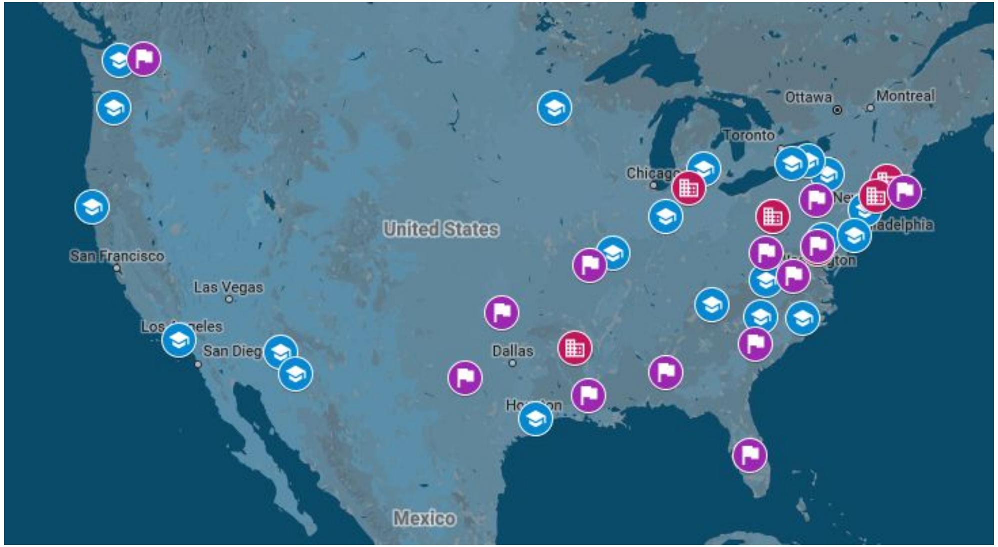
- SQL injections can be used to manipulate a database in many ways. This video shows how SQL injections can be used to obtain sensitive information that is stored in a website’s database.
- Most instances of SQL injections are simply caused by a lack awareness about the vulnerabilities, or programmers purposely cutting corners to save time/money.
- In this video Dr. Mike Pound shows how SQL can be used to retrieve hashed passwords from a website’s database.

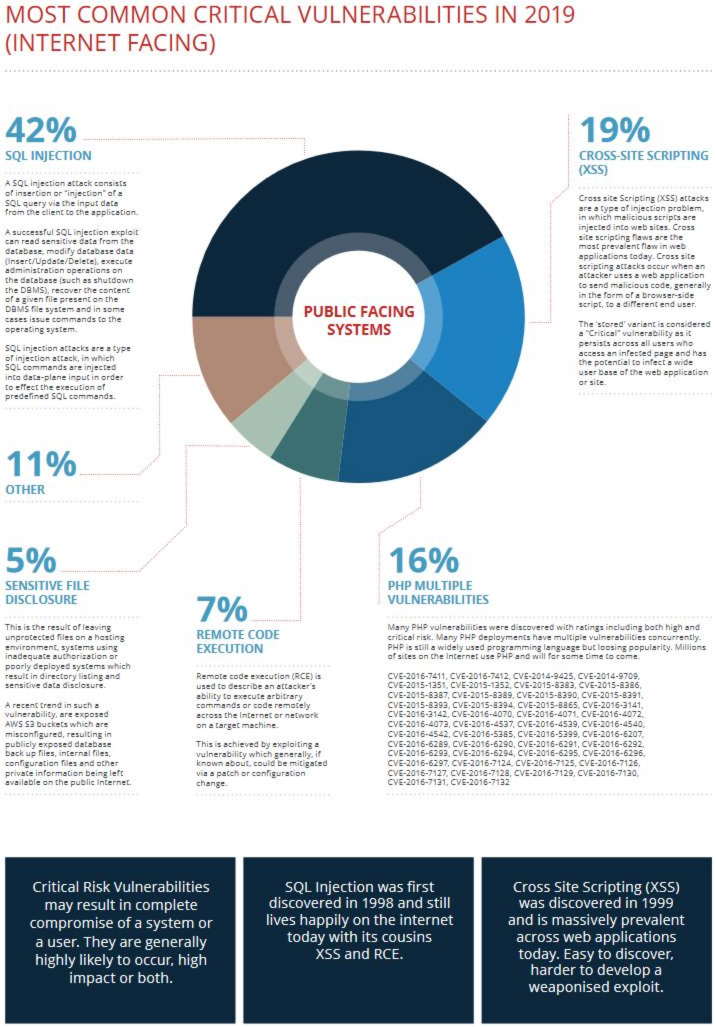


How Prevalent are SQL Injections?

SQL injections were discovered over two decades ago, but they are still the most common internet vulnerability today.

Below is a map of a SQL injection attack that affected over 60+ universities and US government agencies. This attack was carried out in 2016 by a single Russian hacker that goes by the name Rasputin. Anyone can fall victim to an SQL injection attack, so it is important to always make sure that databases the proper protection against such attacks.





MOST COMMON CRITICAL VULNERABILITIES IN 2019 (INTERNET FACING)

42%
SQL INJECTION

A SQL injection attack consists of insertion or "injection" of a SQL query via the input data from the client to the application.

A successful SQL injection exploit can read sensitive data from the database, modify database data (Insert/Update/Delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file present on the DBMS file system and in some cases issue commands to the operating system.

SQL injection attacks are a type of injection attack, in which SQL commands are injected into data-plane input in order to effect the execution of predefined SQL commands.

11%
OTHER

5%
SENSITIVE FILE DISCLOSURE

This is the result of leaving unprotected files on a hosting environment, systems using inadequate authorization or poorly deployed systems which result in directory listing and sensitive data disclosure.

A recent trend in such a vulnerability, are exposed AWS S3 buckets which are misconfigured, resulting in publicly exposed database back up files, internal files, configuration files and other private information being left available on the public Internet.

7%
REMOTE CODE EXECUTION

Remote code execution (RCE) is used to describe an attacker's ability to execute arbitrary commands or code remotely across the Internet or network on a target machine.

This is achieved by exploiting a vulnerability which generally, if known about, could be mitigated via a patch or configuration change.

16%
PHP MULTIPLE VULNERABILITIES

Many PHP vulnerabilities were discovered with ratings including both high and critical risk. Many PHP deployments have multiple vulnerabilities concurrently. PHP is still a widely used programming language but loosing popularity. Millions of sites on the Internet use PHP and will for some time to come.

CVE-2016-7411, CVE-2016-7412, CVE-2014-9425, CVE-2014-9709, CVE-2015-1351, CVE-2015-1352, CVE-2015-8383, CVE-2015-8386, CVE-2015-8387, CVE-2015-8389, CVE-2015-8390, CVE-2015-8391, CVE-2015-8393, CVE-2015-8394, CVE-2015-8865, CVE-2016-3141, CVE-2016-3142, CVE-2016-4070, CVE-2016-4071, CVE-2016-4072, CVE-2016-4073, CVE-2016-4537, CVE-2016-4539, CVE-2016-4540, CVE-2016-4542, CVE-2016-5385, CVE-2016-5399, CVE-2016-6207, CVE-2016-6289, CVE-2016-6290, CVE-2016-6291, CVE-2016-6292, CVE-2016-6293, CVE-2016-6294, CVE-2016-6295, CVE-2016-6296, CVE-2016-6297, CVE-2016-7124, CVE-2016-7125, CVE-2016-7126, CVE-2016-7127, CVE-2016-7128, CVE-2016-7129, CVE-2016-7130, CVE-2016-7131, CVE-2016-7132

19%
CROSS-SITE SCRIPTING (XSS)

Cross site Scripting (XSS) attacks are a type of injection problem, in which malicious scripts are injected into web sites. Cross site scripting flaws are the most prevalent flaw in web applications today. Cross site scripting attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser-side script, to a different end user.

The 'stored' variant is considered a "Critical" vulnerability as it persists across all users who access an infected page and has the potential to infect a wide user base of the web application or site.

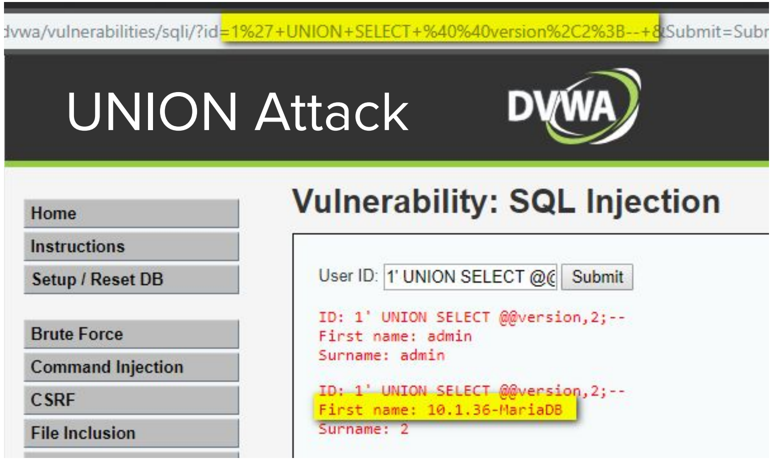
Critical Risk Vulnerabilities may result in complete compromise of a system or a user. They are generally highly likely to occur, high impact or both.

SQL Injection was first discovered in 1998 and still lives happily on the internet today with its cousins XSS and RCE.

Cross Site Scripting (XSS) was discovered in 1999 and is massively prevalent across web applications today. Easy to discover, harder to develop a weaponised exploit.

SQL Injection Examples

- Retrieve Hidden Data - Modify an SQL query to return hidden data.
- Subverting Application Logic - Change a query to interfere with the logic of the application.
- UNION Attacks - Retrieve data from different tables in the database.
- Examining The Database - Extract information about the version and structure of a database.
- Blind SQL Injection - The results of a query the hacker controls are not returned in the application's responses.



dvwa/vulnerabilities/sqli/?id=1%27+UNION+SELECT+%40%40version%2C2%3B--+&Submit=Subr

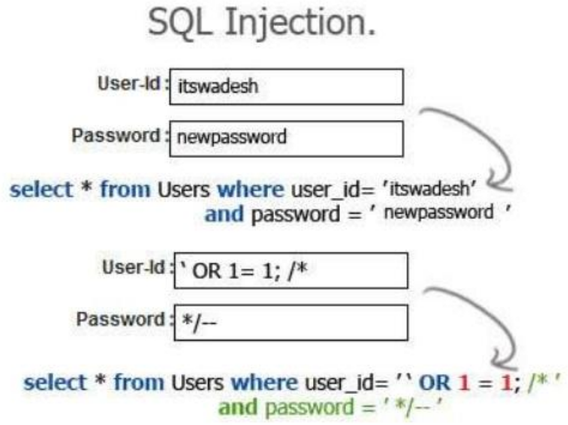
UNION Attack

Vulnerability: SQL Injection

User ID: 1' UNION SELECT @@ Submit

```
ID: 1' UNION SELECT @@version,2;--
First name: admin
Surname: admin

ID: 1' UNION SELECT @@version,2;--
First name: 10.1.36-MariaDB
Surname: 2
```



SQL Injection.

User-Id : itswadesh

Password : newpassword

select * from Users where user_id= 'itswadesh' and password = ' newpassword '

User-Id : ' OR 1= 1; /*

Password : */--

select * from Users where user_id= '' OR 1 = 1; /* ' and password = ' */-- '

How to Prevent SQL Injection Attacks

- Use Prepared Statements

  Prepared statements are one of the best ways to prevent SQL injections. By using a prepared statement, the programmer can create a template for what the SQL statement will be. In this example, the programmer uses the “bind_param()” method to force the user’s input to always be treated as a string. This then makes it impossible for malicious code to be ran.

Prepared Statements Example:

```
$stmt = $mysqli->prepare("SELECT * FROM myTable WHERE name = ? AND age = ?");

$stmt->bind_param("si", $_POST['name'], $_POST['age']);

$stmt->execute();

//fetching result would go here, but will be covered later

$stmt->close();
```

Other Methods to Protect Against SQL Injections

Three other common methods:

- Whitelisting characters - Creating a list of approved characters that can be used and the application then disallows all requests containing characters outside of the whitelist.
- Least Privilege Principle - Lower the privileges of database users so that if their account information is acquired less damage is done.
- Web Application Firewall (WAF) - Inspects the traffic at the application level and determines whether the user input is malicious or not. The WAF needs to constantly be updated because attackers will eventually find a way to bypass it.

Sources

Images:

https://3.bp.blogspot.com/-gblpmrWXm68/W3Txq4ecK8I/AAAAAAAAACk/Up3mU2405GMOdwgkbpm_IlnnlK8IjarzgCLcBGAs/s1600/Screenshot_2018-08-16-09-42-06_1534389209027.jpg

https://blogs.zeiss.com/digital-innovation/de/wp-content/uploads/sites/2/2020/05/201909_Security_SQL-Injection_1.png

https://image.slidesharecdn.com/oevewhq7slodsmcvnnjq-signature-99b7fd3b6b7771232753b0c02812cf1b4294e8f16e5997fd1371c9639cc25044-poli-170615025917/95/sql-injections-with-example-18-638.jpg?cb=1497495914

https://www.recordedfuture.com/assets/recent-rasputin-activity-1.png

https://cdn2.hubspot.net/hubfs/4118561/BCC030%20Vulnerability%20Stats%20Report%20(2020)_WEB.pdf

Information:

https://websitebeaver.com/prepared-statements-in-php-mysqli-to-prevent-sql-injection

https://portswigger.net/web-security/sql-injection

https://pentest-tools.com/blog/sql-injection-attacks/

https://www.recordedfuture.com/recent-rasputin-activity/