

The latest network routers, software, management tools and information for enterprise IT administrators.

Networking
Resources

Network
Security

Network
Software

WAN and
LAN

Data
Center

Network
Management

Networking
Hardware

Unified
Communications

Slideshows

Get the Latest Scoop with Enterprise Networking Planet
Newsletter

Enter Your E-mail Address



[Home](#) [Security](#)

10 Ways to Prevent or Mitigate SQL Injection Attacks

SQL injection attacks could allow hackers to compromise your network, access and destroy your data, and take control of your machines.

By [Paul Rubens](#) | Posted Feb 23, 2010

Page of | [Back to Page 1](#)



Related Articles

[Identify and Mitigate Windows DNS Threats](#)

[Secure Your WLAN With Aircrack-ng](#)

[Build a Portable Security Tool with the ASUS Eee PC and Ubuntu](#)

[Don't Be a Googledork](#)

[Think Like a Black Hat With Offensive Security 101](#)

"Failure to Preserve SQL Query Structure (aka 'SQL Injection')" appears at number 2 in the [CWE/SANS TOP 25 Most Dangerous Programming Errors](#) list published on February 16. And for good reason: SQL injection attacks pose a massive potential threat to your organization. That's because, if successful, they could allow hackers to compromise your network, access and destroy your data, and take control of your machines.

What Is SQL Injection?

The principal behind SQL injection is pretty simple. When an application takes user data as an input, there is an opportunity for a malicious user to enter carefully crafted data that causes the input to be interpreted as part of a SQL query instead of data.

For example, imagine this line of code:

```
SELECT * FROM Users WHERE Username=' $username' AND Password=' $password'
```

which is designed to show all records from the table "Users" for a username and password supplied by a user.

Using a Web interface, when prompted for his username and password, a malicious user might enter:

'1' or '1' = '1'

'1' or '1' = '1'

resulting in the query:

```
SELECT * FROM Users WHERE Username='1' OR '1' = '1' AND Password='1' OR '1' = '1'
```

The hacker has effectively injected a whole OR condition into the authentication process. Worse, the condition '1' = '1' is always true, so this SQL query will always result in the authentication process being by passed.

<Code sample sourced from OWASP http://www.owasp.org/index.php/Main_Page>

Using characters like ";" to append another query on to the end of an existing one, and -- to comment out (and therefore cut off) a part of an existing query, a hacker could potentially delete entire tables, or change the data they contain. He could even issue commands to the underlying OS, thereby taking over the machine, and using it as a staging post to attack the rest of your network. In summary, the consequences of a SQL injection attack could be:

- Loss of data confidentiality
- Loss of data integrity
- Loss of data
- Compromise of the entire network

What Can Be Done to Prevent SQL Injection Attacks?

The most important precautions are data sanitization and validation, which should already be in place. Sanitization usually involves running any submitted data through a function (such as MySQL's `mysql_real_escape_string()` function) to ensure that any dangerous characters (like "'") are not passed to a SQL query in data.

Validation is slightly different, in that it attempts to ensure that the data submitted is in the form that is expected.

At the most basic level this includes ensuring that e-mail addresses contain an "@" sign, that only digits are supplied when integer data is expected, and that the length of a piece of data submitted is not longer than the maximum expected length. Validation is often carried out in two ways: by blacklisting dangerous or unwanted characters (although hackers can often get around blacklists) and by whitelisting only those characters that are allowed in a given circumstance, which can involve more work on the part of the programmer. Although validation may take place on the client side, hackers can modify or get around this, so it's essential that you also validate all data on the server side as well.

But sanitization and validation are far from the whole story. Here are ten ways you can help prevent or mitigate SQL injection attacks:


1. **Trust no-one:** Assume all user-submitted data is evil and validate and sanitize everything.
2. **Don't use dynamic SQL when it can be avoided:** used prepared statements, parameterized queries or stored procedures instead whenever possible.
3. **Update and patch:** vulnerabilities in applications and databases that hackers can exploit using SQL injection are regularly discovered, so it's vital to apply patches and updates as soon as practical.
4. **Firewall:** Consider a web application firewall (WAF) – either software or appliance based – to help filter out malicious data. Good ones will have a comprehensive set of default rules, and make it easy to add new ones whenever necessary. A WAF can be particularly useful to provide some security protection against a particular new vulnerability before a patch is available.
5. **Reduce your attack surface:** Get rid of any database functionality that you don't need to prevent a hacker taking advantage of it. For example, the xp_cmdshell extended stored procedure in MS SQL spawns a Windows command shell and passes in a string for execution, which could be very useful indeed for a hacker. The Windows process spawned by **xp_cmdshell** has the same security privileges as the SQL Server service account.
6. **Use appropriate privileges:** don't connect to your database using an account with admin-level privileges unless there is some compelling reason to do so. Using a limited access account is far safer, and can limit what a hacker is able to do.
7. **Keep your secrets secret:** Assume that your application is not secure and act accordingly by encrypting or hashing passwords and other confidential data including connection strings.
8. **Don't divulge more information than you need to:** hackers can learn a great deal about database architecture from error messages, so ensure that they display minimal information. Use the "RemoteOnly" customErrors mode (or equivalent) to display verbose error messages on the local machine while ensuring that an external hacker gets nothing more than the fact that his actions resulted in an unhandled error.
9. **Don't forget the basics:** Change the passwords of application accounts into the database regularly. This is common sense, but in practice these passwords often stay unchanged for months or even years.
10. **Buy better software:** Make code writers responsible for checking the code and for fixing security flaws in custom applications before the software is delivered. SANS suggests you incorporate terms from [this sample contract](#) into your agreement with any software vendor.

prubens@jupiternmedia.com

3 Comments ([click to add your comment](#))

By Chris July 11 2013 13:14 PDT

Nice one on prevention in sql injection attacks!! Love the post. Good info here as well:
<http://www.programmerinterview.com/index.php/database-sql/sql-injection-prevention/>

 [Reply to this comment](#)



By Golu Singh March 26 2012 09:17 PDT


This is one of the best articles so far I have read online. Just useful information. Very well presented. It helped me lot, there is some other good articles too which helped me in completing my task... here I'm sharing links of that post.... <http://www.mindstick.com/Blog/228/Preventing%20SQL%20Injection>
<http://msdn.microsoft.com/en-us/library/ms161953.aspx> <http://msdn.microsoft.com/en-us/library/ff648339.aspx> Thanks Everyone for your nice post!!

 [Reply to this comment](#)



By Kumar Sanu November 12 2011 01:21 PST

This is nice post. Its really helpful for me and this link <http://mindstick.com/Blog/228/Preventing%20SQL%20Injection> also helped me to complete my task. Thanks All!!

 [Reply to this comment](#)



Comment and Contribute

Your name/nickname

Your email

Subject (Optional)

Comments

(Maximum characters: 1200). You have 1200 characters left.



Type the text

[Privacy & Terms](#)



Submit



Property of QuinStreet Enterprise.

[Terms of Service](#) | [Licensing & Reprints](#) | [About Us](#) | [Privacy Policy](#) | [Advertise](#) | [Sitemap](#)
Copyright 2015 QuinStreet Inc. All Rights Reserved.

