Weapon of Mass Disruption

Michael Peters | June 13, 2011

Of all the attacks taking place on Web sites across the Internet today, and there have been some very high profile ones recently, SQL injection is making its mark as the cyber-criminal's most effective weapon of mass disruption (WMD) into corporate data stores. In my opinion, this threat vector against the crown jewels is public enemy number one.

Databases, data stores, data warehouses and any other data repository is the most important segment of any corporations technological resources and deserves the most attention by security and handling best practices. The reality is that it frequently is one of the technology after-thoughts. Just for curiosity purposes, I ran a search in Google on "SQL injection News" and 44 news articles came up all relating to breaches of corporate networks. Criminals are not interested in web site defacement anymore, but in financial profit. They are after credit card numbers, email addresses, personally identifying information, account numbers, social security numbers, and any bit of data that will get them into bank accounts and other sources related to stealing money.

The big question for you may be "What is SQL injection and how do I fix it," right? Like many of the most effective web application exploitation methods, SQL injection takes advantage of fundamental flaws in the way our business applications interface with their respective databases. Successful SQL injection attacks can give criminal attackers the means to access any sensitive information from the connected database, modify that database, and even blend attack vectors to take control of the server or even network segment the database resides on.

If you distill the attack methodology down into its fundamentals, the basic SQL injection attack is made possible by the fact that so many new applications hitting the Internet and Intranets today interface with some sort of database in order to offer application users easy access to information. Consider this: Every time an application user searches a commerce site to pick a surgical needle from thousands of haystack choices or searches a customer relationship management (CRM) business application to find a customer's information or even enters a search term into a social networking site such as Facebook, that application user is performing a database query.

In your typical Internet- or Intranet-facing user application, there is a method to interact with the back-office database via some type of search box. When a user enters their search terms into that box, the application essentially appends that search term into a database query, which is then run against the database in order to pull up the requested information from a particular category in the data store. That data is again passed through to the user-facing application, which formats it into the display device.

Our culprit is that application developers fail to have the user-facing application filter input. This is a frequent problem that allows a criminal the opportunity to write an SQL command in such a way within that user-facing search box I mentioned which compels the application to perform a completely different query against the database. The unfortunate end result is that far more access to information is granted to the database. Instead of a product search for that surgical needle, for instance, a criminal attacker may now get that same search application to retrieve customer credit card information stored within the database.

I know it sounds like I'm throwing application developers under the proverbial bus and you would be correct. Sorry folks, but if I were responsible for castle security and I leave the drawbridge down, renegades will now cross the middle-ware moat and storm the King's castle. The key to preventing SQL injection is to up your game by developing better code. My advice is to hone those coding skills, keep up with industry best practices, become a "Life Learner" and always look for ways to improve. Keep in mind that your work will make or break companies and affect individual's privacy and financial well-being.