An injection that causes ailments!

Over the last couple of years, but especially the last 12 months, there has been an increase in the number of web servers around the world that are being attacked, and compromised, via "SQL Injection".

This sounds like a very painful procedure, and one that should only be accompanied with a general anaesthetic! However, the reality is far more mundane, but potentially more troublesome.

SQL Injection is best described as the manipulation of information that is transmitted from a web browser to a web server that is using an SQL based database server for content management. This content can be anything from the images and text displayed through to the customer details queried from a back end system. This is the model, at a very generic level, that most online systems such as banks, online retailers etc deploy.

The manipulation referenced above occurs when a command is created at the web browser to be sent to the web server to display a piece of information or complete a process such as a login. At this point, the malicious user changes some of the parameters in the command being sent and the database server that receives this newly crafted, malicious, piece of code will process it and return the results of that action to the user's web browser.

Many organisations with a web site today that is not a complex multi-tier design feel that they should be immune from such issues. They do not have a database that is storing user information; there is no area for logons to be processed etc. Traditionally, these organisations would probably have been immune to attacks such as SQL injection as they would have had no reason for a database server. However, many web development and content management tools today require a database server to manage the information on the web site. This has led to many organisations having database servers on their web server without even knowing it. This in turn means that the database server installed is not hardened or protected in anyway compared to the installation of the web server and the operating system hosting it.

The worst thing about all of this is that it is relatively straightforward to prevent a web server and database server combination from being compromised via SQL injection attacks. Good coding practises, server side validation of all data received from user browsers, database hardening and separation from web servers etc will all help prevent a compromise occurring. One of the most important principles for application developers to adhere to is that of using Stored Procedures for all interaction with the database and remove access to Dynamic SQL commands.

There are also a number of defence technologies that can be deployed at various levels in the infrastructure. At the network layer, Intrusion Prevention/Detection Systems (IPS/IDS), and at the server level host based IDS/IPS as well as application layer firewalls can be deployed. All of these tools have signatures that will alert when SQL injection attacks are spotted, or in the case of IPS, prevent the attack succeeding.

It is also critical to ensure that the proposed solution has been tested from a security perspective prior to deployment. Areas such as SQL injection should be specifically

included in the test plan if there are any database components involved in the content management of the site or providing information for customers from back end systems.

There is a responsibility on the companies that provide web development or content management services to ensure that the underlying infrastructure such as database servers are appropriately located in the design and secured.

Rits Group Head Quarters
Information Security Centre
2052 Castle Drive
Citywest Business Campus
Co. Dublin
Ireland

Tel: +353 (0) 1 6420500
Fax: +353 (0) 1 4660468
Email: info@ritsgroup.com
Web: www.ritsgroup.com