Utility and Enhancement of SQLIA Detection and Prevention Techniques

Charles Asanya and Ratan Guha

Department of EECS
University of Central Florida
Orlando Florida

charlieboy@knights.ucf.edu   guha@eeecs.ucf.edu
Abstract

• **SQL Injection Attack (SQLIA)** is a form of attack used to maliciously manipulate data stored in a database
  – User input taken from text box created by developers to allow interaction with a database is used to perform this attack.
  – Using a malformed SQL statement, attacker alters the intended query structure which can be used to break into and steal information or change and destroy a database.
    • **Eg.** `SELECT * FROM accounts WHERE name=’root’ AND password=’1234’ OR ’1=1’`  
• Different techniques has been proposed by researchers to stop this attack.
  – Research analysis has shown that these techniques cannot prevent all types of SQLIA
  – Difficult to implement
  – Some recommend combining techniques.
• The objective of this paper is to establish the usefulness of the existing detection and prevention techniques against different types of SQLIA
• Establish the right combination in order to improve the prevention of all types of SQL injection attack.
Types of SQLIA/Proposed Techniques

• Types
  – Tautologies, Union, Piggy Back, Stored Procedure, Inference, Alternate Encoding, Second Order and Illegal/Incorrect Queries.

• Most of the proposed techniques are based on using these methods;
  – Blacklist
  – Using Stored Procedure
  – Limit Privilege to Application that needs them
  – Using a Framework
  – Using Query Parameters
  – Whitelist
  – Input Type Checking
  – Escaping/Encoding of Inputs
  – Avoid Disclosing Error Information
Problems

• Among the proposed technique, none has proven to prevent all types of SQLIA.
  – Some SQL keywords cannot be escaped. There are some query actions that has no API to perform it.
  – Escaping character inside a string may cause an early escape
  – Escaping can lead to truncation as single quote are doubled by escape function.
  – There is a possibility of escaping harmless character
  – Blacklisting creates false positive
  – Hard to compile a comprehensive whitelist, and it will require frequent update.
  – Hard to determine all application needs in the development phase
  – Some Input Type checking can be subverted by Alternate Encoding
  – Some framework are unable to handle sophisticated SQL.
  – Query parameters are not able to handle multiple string literals, and cannot handle columns and table names.
  – Stored Procedures with parameters is still vulnerable.
  – Hard to implement all the techniques in one application as one may compromise the other
Solution/Work in Progress

• Since each solution has its own problem, we are experimenting with combining the following techniques to determine the prevention from all SQLIA.
  
  – Query Parameter (Ensures attacker is not able to change query intent)
  
  – Whitelist (Handle Keywords, table names or columns)
  
  – Escaping Inputs with truncation protection in Client/Server side
  
  – Hiding error Information (Limit information revealed to users caused by inference or Illegal Query attack)