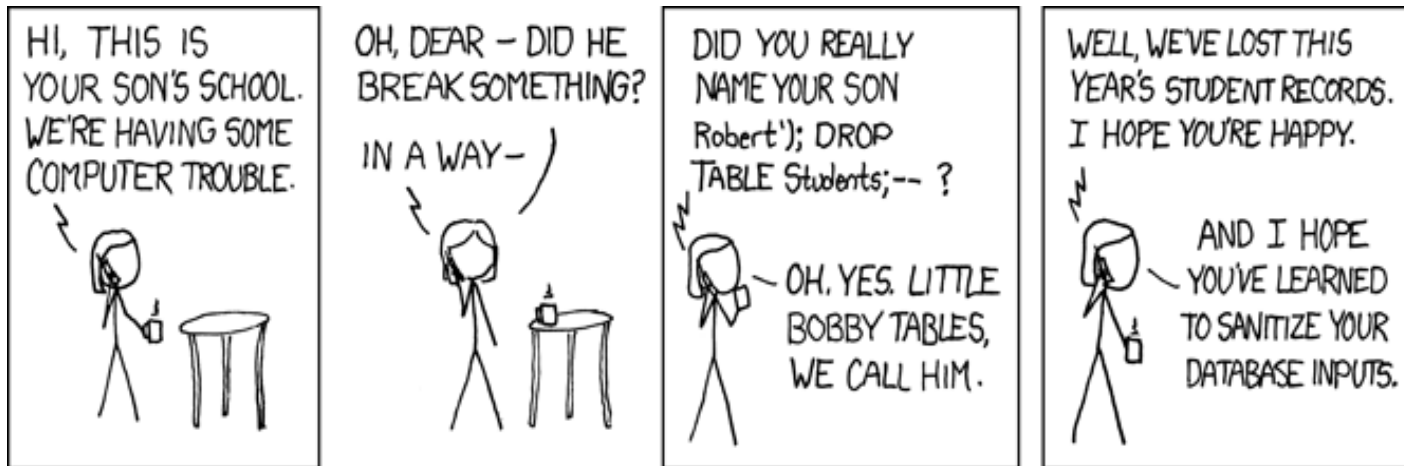


Avoiding SQL Injection Attacks



Ben Crenshaw

Greg Wendt

What is SQL Injection?

- The use of user input to execute unintended SQL code.
 - A subset of input vulnerabilities focused on SQL (buffer overflows being a different subset)
- Where do they happen?
 - Anywhere that users can input text that eventually interacts with a database
 - Most common – web applications over http:// or https://

Demo 1: Basic Example of Sql Injection

A Few Types of Attacks:

Line Comments

Line comments are used in attacks to escape the remainder of an SQL query to avoid throwing errors.

An unprotected login query

- o "SELECT user FROM members WHERE user=' " & input1 & "' AND password=' " & input2 & "';"

Can become an automatic admin login if input1 = "admin'--"

- o SELECT username FROM members WHERE username = 'admin' --' AND password = 'password'

A Few Types of Attacks: Stacking Queries

Some databases allow more than one query to execute within one transaction (i.e. SqlServer)

- If input1 = `''; DROP members--`", then...

- o `SELECT` username `FROM` members `WHERE`
username = `'`; `DROP` members `--` `'` `AND`
`password` = `'password'`

A Few Types of Attacks: Union Injections

Many databases support UNION queries, which can be used in attacks to find data across different tables

- Example: a query to display product info

- o "SELECT productName, description FROM products WHERE id=" & input1 & ";"

- If input1 = "1 UNION ALL SELECT username, password FROM members", then...

- SELECT productName, description FROM products WHERE id=1 UNION ALL SELECT username, password FROM members

Safeguarding from SQL Injection Attacks

- Decrease the amount of attackers that have access to the application
- Decrease the amount of damage that successful attacks can do
- Attempt to prevent successful SQL injection within the application code

Decrease the amount of attackers

- Internet vs. Intranet?
- Require authentication to access
- Limit the number of users to the minimum necessary

Decrease the Damage

- Applications should NOT connect to the database under an administrator account
- Database should be installed with minimum machine privileges
 - i.e. SqlServer by default runs service under Local System account, increasing risk of an attacker being able to gain machine access (through xp_cmdshell, etc.)

Decrease the Damage, contd.

- Encrypt critical data that must not be stolen (i.e. SSNs, credit cards, passwords)
 - Many attacks aim to gain unauthorized access to the application, or display data hidden in your database
- Backup data that you cannot afford to lose
 - Some attacks are simply to cause data loss or denial of service (i.e. DROP TABLE)

Attempt to Prevent SQL Injection within your code

- All it takes is one vulnerable query in an application
- Treat every user input as if it was a SQL Injection attack
- Validate all user inputs with **server-side** code, and do not execute SQL query if errors are found.
 - Client-side code (i.e. JavaScript) should be used for user-friendliness, but is easily bypassed

Demo 2: Examples of Validation

Validating text fields?

- Escaping quotes – (' becomes ")
 - Not 100% hacker-proof depending on your database
- Avoid string concatenation to build dynamic queries.
 - Difficult to make secure
 - Also a risk in stored procedures
- Better solution – Parameterized SQL
 - Most reliable way to perform queries from code

Demo 3: Examples

Second Order Attacks

- A query may be safely inserted into your database, with no attack succeeding
 - `INSERT INTO products (id, name) VALUES (1, ''); DELETE FROM products--');`
- The data can later be queried, and it becomes a successful attack
 - `SELECT id FROM products WHERE name = ''; DELETE FROM products--'`

Black Lists and Filtering

- Black Lists are functions that search for possible SQL injections within text
 - Remove possible attacks
 - Can be used to log attempted attacks
 - Not 100% full-proof, but can help to prevent second-order attacks

Demo 4: Black List example

Additional Tips

- Protect your source code
 - Easiest way to make an attack is to know the code
- Don't display detailed error messages or SQL queries
 - Just as helpful to attackers as it is to developers
- Set and validate minimum lengths to avoid buffer overflows
- Keep databases patched to avoid buffer overflows
- Educate other developers regarding SQL Injection

Questions?