University of Central Florida
CGS 2545
Database Concepts

DEPARTMENT OF ELECTRICAL ENGINEERING & COMPUTER SCIENCE
COMPUTER SCIENCE DIVISION

Injection

- If you take a user input through a webpage and insert it into a SQL database, there is a chance that you have left yourself wide open for a security issue known as the **SQL Injection**.
- There are methodologies how to prevent this from happening and secure scripts and SQL statements in your server side scripts such as a PERL Script.

Injection

- Injection usually occurs when you ask a user for input, like their name and instead of a name they give you a SQL statement that you will unknowingly run on your database.
- Never trust user provided data, process this data only after validation
- As a rule, this is done by **Pattern Matching**

Injection

- In the example below, the **name** is restricted to the alphanumerical characters plus underscore and to a length between 8 and 20 characters (modify these rules as needed).

```
if (preg_match("/^\w{8,20}$/", $_GET['username'], $matches)) {
    $result = mysql_query("SELECT * FROM CUSTOMERS
        WHERE name = $matches[0]");
} else {
    echo "user name not accepted";
}
```

Injection

- To demonstrate the problem, consider this excerpt

```
// supposed input
$name = "Qadir'; DELETE FROM CUSTOMERS;";
mysql_query("SELECT * FROM CUSTOMSRS WHERE name='{$name}'");
```

Injection

- The function call is supposed to retrieve a record from the CUSTOMERS table where the name column matches the name specified by the user.
- Under normal circumstances, **$name** would only contain alphanumeric characters and perhaps spaces, such as the string ilia.
- But here, by appending an entirely new query to $name, the call to the database turns into disaster; the injected DELETE query removes all records from the CUSTOMERS table.

Injection

- Fortunately, if you use MySQL, the **mysql_query()** function does not permit query stacking or executing multiple SQL queries in a single function call. If you try to stack queries, the call fails.
- However, other PHP database extensions, such as **SQLite** and **PostgreSQL**happily perform stacked queries, executing all the queries provided in one string and creating a serious security problem.

Injection

- Preventing SQL Injection
  - You can handle all escape characters smartly in scripting languages like PERL and PHP.
  - The MySQL extension for PHP provides the function **mysql_real_escape_string()** to escape input characters that are special to MySQL.

```
if (get_magic_quotes_gpc()) {
   $name = stripslashes($name);
}
$name = mysql_real_escape_string($name);
mysql_query("SELECT * FROM CUSTOMERS WHERE name='{$name}'");
```

Injection

- The LIKE Quandary
  - To address the LIKE quandary, a custom escaping mechanism must convert user-supplied '%' and '_' characters to literals.
  - Use **addcslashes()**, a function that lets you specify a character range to escape.

```
$sub = addcslashes(mysql_real_escape_string("%str"), "%_");
// $sub == \%str\_
mysql_query("SELECT * FROM messages
   WHERE subject LIKE '{$sub}%'");
```