CSE 503
Introduction to Computer Science for Non-Majors

Dr. Eric Mikida
epmikida@buffalo.edu
208 Capen Hall

Day 30
Injection Attacks

User Input

Interactive applications involve some form of user input

For example, the MusicRater WebApp allows users to input a song

We gave them fields to enter the ID, title and artist

User Input

Interactive applications involve some form of user input

For example, the MusicRater WebApp allows users to input a song

We gave them fields to enter the ID, title and artist

What will our users type into these text boxes?

User Input

Interactive applications involve some form of user input

For example, the MusicRater WebApp allows users to input a song

We gave them fields to enter the ID, title and artist

What will our users type into these text boxes?

What if they type something we don't expect?

User Input

Interactive applications involve some form of user input

For example, the MusicRater WebApp allows users to input a song

We gave them fields to enter the ID, title and artist

What will our users type into these text boxes?

What if they type something we don't expect?

What if they have malicious intent?

HTML Injection

Users can type anything they want into a textbox

Their input could include HTML, ie:

```
<b>Some bold text</b>
```

HTML Injection

Users can type anything they want into a textbox

Their input could include HTML, ie:

```
<b>Some bold text</b>
```

Depending on how that input is handled, their input may be incorporated into the HTML of our webpage, and therefore rendered as HTML…

HTML Injection

Users can type anything they want into a textbox

Their input could include HTML, ie:

```
<b>Some bold text</b>
```

Depending on how that input is handled, their input may be incorporated into the HTML of our webpage, and therefore rendered as HTML…

Our users can now inject their own HTML into OUR webpage…

HTML Injection

So…users can make their text bold…what's the harm in that?

What if they type something more complex:

```
<button onclick="alert('You\'ve been hacked!!');">Click This</button>
```

HTML Injection

So…users can make their text bold…what's the harm in that?

What if they type something more complex:

```
<button onclick="alert('You\'ve been hacked!!');">Click This</button>
```

Still ultimately harmless…but you can see where this could lead

HTML Injection

But what about this:

```
<META HTTP-EQUIV="refresh" CONTENT="1;url=http://www.buffalo.edu">
```

HTML Injection

How do we prevent this? Any ideas?

HTML Injection

How do we prevent this? Any ideas?

Do not incorporate user input directly

HTML Escaping

Do not incorporate user input directly

Use an HTML "escape" mechanism which allows us to distinguish the data from the program

Characters like: < > & "

Get converted to: < > & "

http://doc.locomotivecms.com/making-blog/2-6-html-escaping

HTML Escaping

In Python:

```
import html

safeMessage = html.escape(message)
```

SQL Injection

MusicRater3.0 has added a search feature! It allows users to look at just the songs by a particular artist. We have seen in previous lectures how this could be accomplished using SQLite queries.

If we enter the text `Alt-J` into the search box, what do we expect to see?

SQL Injection

Users can still type any text into the text field…uh oh…

If the text gets incorporated into an SQL query, bad things can happen…

What if a user types `' OR '1'='1' --` into the search box?

SQL Injection

Users can still type any text into the text field…uh oh…

If the text gets incorporated into an SQL query, bad things can happen…

What if a user types `' OR '1'='1' --` into the search box?

Our SQL command becomes:

```
SELECT * FROM songs WHERE 'artist'='' OR '1'='1' --
```

SQL Injection

Users can still type any text into the text field…uh oh…

If the text gets incorporated into an SQL query, bad things can happen…

What if a user types `' OR '1'='1' --` into the search box?

Our SQL command becomes:

```
SELECT * FROM songs WHERE 'artist'='' OR '1'='1' --'
```

-- starts a comment in SQL, so anything after it would be ignored

SQL Injection

The SQL command

```
DROP TABLE someName
```

removes the table whose name is `someName`, ie:

```
DROP TABLE songs
```

or

```
DROP TABLE ratings
```

SQL Injection

What if the users search the following (the ; separates multiple commands in SQL):

```
Alt-J'; DROP TABLE songs; --
```

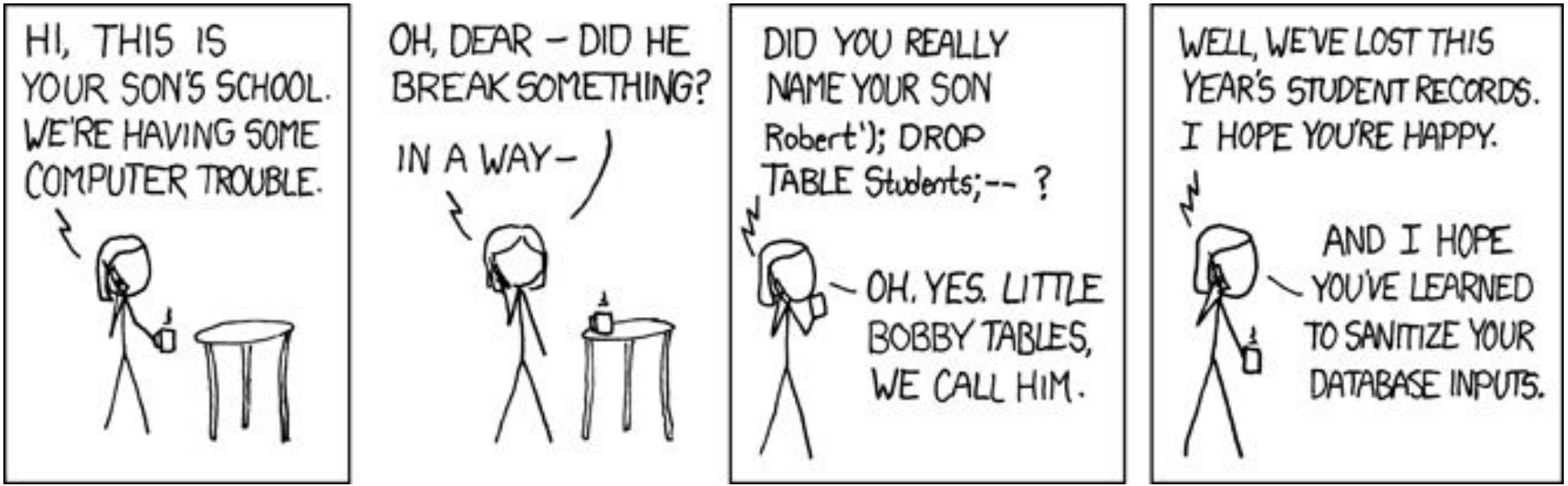
SQL Injection

What if the users search the following (the ; separates multiple commands in SQL):

```
Alt-J'; DROP TABLE songs; --
```

Luckily for us, nothing. The **`execute`** function in Python's SQLite3 library does not allow multiple commands. Phew...

SQL Injection



https://xkcd.com/327/

SQL Safe Substitution

Usually your SQL operations will need to use values from Python variables. You shouldn't assemble your query using Python's string operations because doing so is insecure; it makes your program vulnerable to an SQL injection attack (see https://xkcd.com/327/ for humorous example of what can go wrong).

Instead, use the DB-API's parameter substitution. Put ? as a placeholder wherever you want to use a value, and then provide a tuple of values as the second argument to the cursor's execute() method.

https://docs.python.org/3/library/sqlite3.html

SQL Safe Substitution

Usually your SQL operations will need to use values from Python variables. You **shouldn't assemble your query using Python's string operations because doing so is insecure;** it makes your program vulnerable to an SQL injection attack (see https://xkcd.com/327/ for humorous example of what can go wrong).

Instead, use the DB-API's parameter substitution. **Put ? as a placeholder wherever you want to use a value, and then provide a tuple of values as the second argument to the cursor's execute() method.**

https://docs.python.org/3/library/sqlite3.html

Morale of the Story

We have to be careful when handling user input!

Never allow user input to be interpreted as code (HTML, SQL, or other)