SQL Injection and XSS

Cloud Computing and SaaS

Announcements

- Checkpoint 3 Released
  - Deadline extended to Tuesday (11/2)
  - Make sure to pull from source often!
- Autograder still in progress
  - Local tests releasing sometime tomorrow afternoon
    - Same tests as the ones on the autograder!
  - Posts service depends on functioning auth service

Last Time

- Guest Speaker - Greg Pavlik, SVP of Oracle Cloud Infrastructure
- Web Security
  - Cookies
  - Javascript
  - Frames
  - Same-Origin Policy

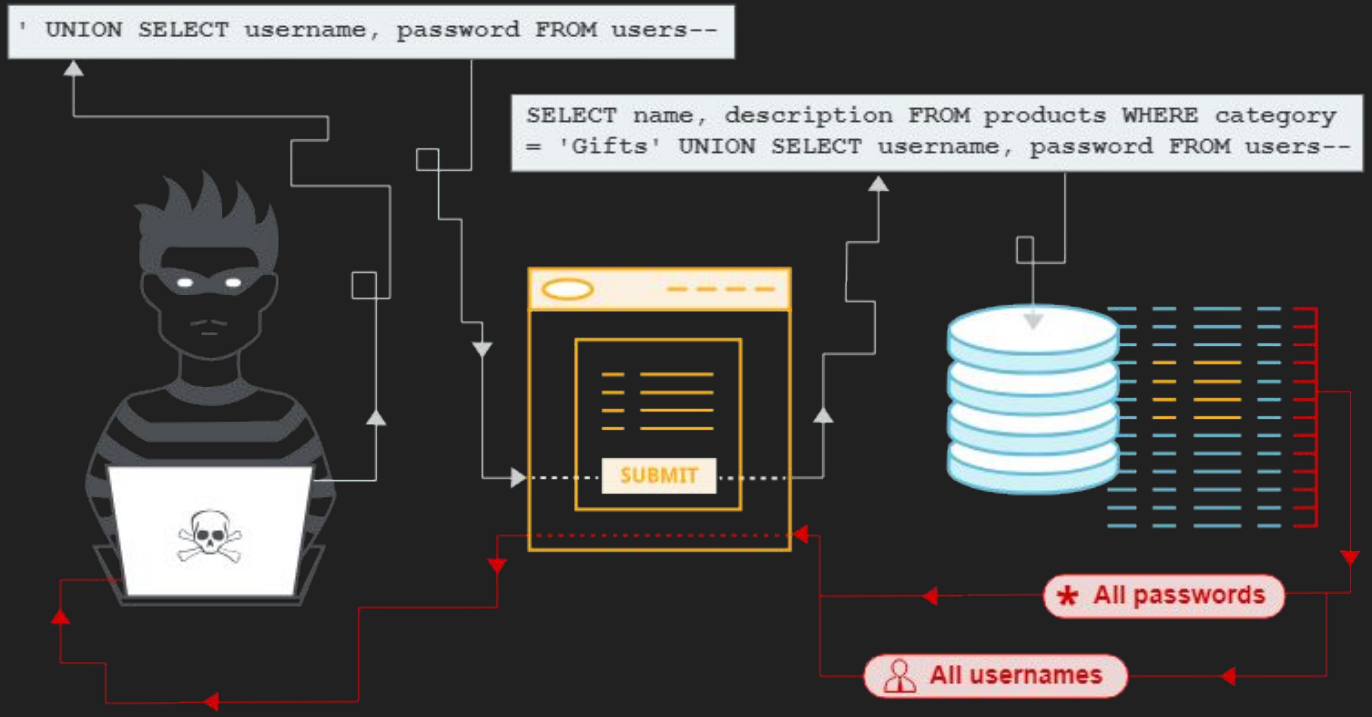
Today

- More Web Security
  - SQL Injection
  - XSS

SQL Injection

- Inject SQL through user input (usually on frontend/client)
- Recall:
  - SQL can both read *and* write to a table
  - SQL outputs/returns the result of the final query
  - Project - You directly input the JSON request body into the database
- Can end up reading sensitive user information
  - Email, hashed passwords, credit card info?

SQL Injection



SQL Injection

- Another example:
  - Schema - (ItemName, ItemNumber, ItemDescription)
  - Original Query
    - “SELECT ItemNumber, ItemDescription FROM Items WHERE ItemName= “?”;“
  - Input -> test”; DROP TABLE users --”
  - New Query
  - “SELECT ItemNumber, ItemDescription FROM Items WHERE ItemName=”test”; DROP TABLE users --”

SQL Injection

- Prevention?
  - Escaping
    - Put a “\” before every quote in the input
    - SQL Injection can abuse the fact that we can send in our own string inputs (as in previous example)
    - Escaping makes sure that any quotes in the input is part of what is inserted into the database
  - New Query -> “SELECT ItemNumber, ItemDescription FROM Items WHERE ItemName=”test\”; DROP TABLE users --\””;”

SQL Injection

- Prevention?
  - Parameterized SQL
  - Project Example:

```
var exists bool
//check if the email or username exists
err = DB.QueryRow("SELECT EXISTS (SELECT * FROM users WHERE username = ?)", credentials.Username).Scan(&exists)
if err != nil {
    http.Error(w, errors.New("error checking if username exists").Error(), http.StatusInternalServerError)
    log.Print(err.Error())
    return
}
if exists == true {
    http.Error(w, errors.New("this username is taken").Error(), http.StatusConflict)
    return
}
```

Cross-Site Scripting (XSS)

- Attacker creates a malicious script that runs on your browser
- Recall:
  - Javascript is extremely powerful
  - You can use “<script>” tags to indicate a Javascript segment
    - Browser can parse this when parsing HTML and end up executing the query

Cross-Site Scripting (XSS)

- Quick Demo

Cross-Site Scripting (XSS)

- Basic Prevention Tactics
  - HTML Encode/Escape all page content
  - Use “HTTPOnly” Cookie flag
  - Many, many more... Take CS161!