# Cybersecurity

## Secure Coding Techniques

# Input Validation

- How do we handle errors in code?
  - Errors are going to happen, proper error handling involves capturing the log file and reporting it
    - Then, it can be fixed later
  - We do not want the errors reported to an attacker
- Why do we validate inputs?
  - Validating inputs is making sure what is supposed to be input is being input
  - This helps stop XSS, XSRF, and other attacks from happening
- <u>Normalization</u> is the first step of validating inputs
  - It checks to make sure if the answer looks “normal”
    - For example, if asking for a first name, “jsmith@gmail.com” would not be normal

# Stored Procedures

- Stored procedures are defined functions that are stored in a database engine
  - These procedures can be used with input validation
- They do not alter the database
  - They just get the information from the database
- Really secure databases will only used stored procedures and not allow other methods of getting data

# Obfuscation/Camouflage

- Obfuscation is making something super easy and making it very hard to understand
    - Developers will take very simple code and make it very unreadable
        - Developers will keep the simple code, and give the users the hard to read code
        - Does not change what the code does, just makes it harder to follow
    - This makes finding security holes much harder
        - Attackers have to figure out what is happening
        - Takes them more time, doesn’t make it impossible, just more time

# Code Reuse/Dead Code

- Code reuse is using old code to help in the making of new applications
  - This is as simple as copy and pasting
  - Saves the developers a lot of time
  - If that old code has vulnerabilities, it will also be in the new code
    - Creates a ripple effect
- Dead code may or may not be used
  - But what it produces is useless to the rest of the code
  - Need to eliminate if possible
    - There are compiler options that check for dead code

# Server-side vs. Client-side execution and validation

- Server-side validation is checking for errors on the server
  - This helps protect against malicious users
    - Malicious users could be trying to use a different interface
- Client-side validation checks for errors on the client's app
  - This can be faster for users
- Can use both
  - A little more secure with server-side
    - Use more server-side validations than client-side

# Memory Management

- As a developer, you must be mindful of how memory is used
  - Many opportunities to build vulnerable code
- Bad memory management can lead to leaks
  - These leaks spread over time
  - Become a security risk

# Third-Party Libraries and SDKs

- Third-party libraries and software development kits (SDK) can help programmers
  - Save a lot of time
  - Increase the functionality of a language
- These are also huge security risk
  - Who is writing the code?
  - Could be very secure, could not be!
  - Always test the code before using it

# Data Exposure

- Data exposure is losing control of data during operations
    - You must always protect data
- Data exposure must be limited
    - Must protect the user’s data!
- Data must be protected when…
    - Stored (data at rest)
    - Being communicated (data in transit)
    - While being used (data in use)