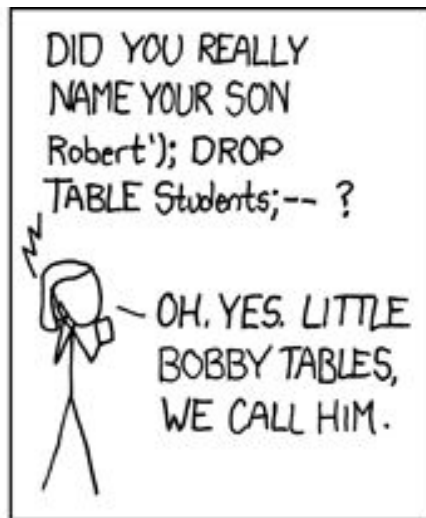
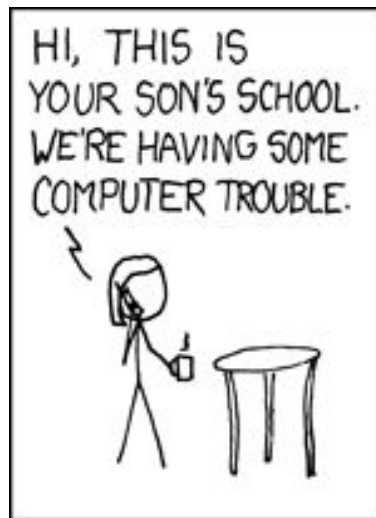


Module 2 - Lecture 9

Data Security





SQL Injection Attacks



Practice Safe Computing

- **Parameterized queries:** when done consistently, SQL injection attacks will not be possible.
- **Input validation:** limit the number of acceptable values.
- **Limit database user privileges:** the user that your application is accessing your database with should be given the least permissions necessary to function. This is called *Principle of Least Privilege*.



Securing Passwords / Secrets



Securing Passwords / Secrets

- We need to be able to verify a password, not see the password.
- A system administrator with full access to the credential database should not be able to see your password.
- A hacker that stole the credential database should not be able to see your password.
- Even with supercomputing power, no one should be able to uncover the data in a reasonable amount of time.



HASHING



HASHING

- A one way function to obfuscate the data.
- This is used prior to storage of the password.
- To verify the password, the same function is used to re-generate the hash and that is what is compared against the database.
- Passwords should also be salted prior to hashing. This will make it take even longer for someone to crack.



What makes a good hashing function?

- Inputs and outputs are constrained
- Relationship between inputs and outputs appears random
- Inputs should be evenly distributed over outputs
- Some degree of efficiency that does not sacrifice collision resistance
- Examples of hashing functions:
 - Message Digest (**MD5** - cracked via Birthday Attack)
 - Secure Hash Algorithm (**SHA-512**)



ENCRYPTION



Securing data at rest

- Data at rest can use a form of encryption called **symmetric key encryption**. Symmetric because the key works for both encrypting and decrypting.
- Requires both parties to use the key to encrypt and decrypt data.
- Any party possessing the key can read the data.
- Has difficulties of securing the symmetric key amongst multiple parties.



Securing data in transit

- Data in transit is often secured using **asymmetric encryption** or **public key encryption**.
- Asymmetric algorithms allow us to create a **public key** and a **private key**.
 - The public key is distributed freely.
 - The private key is kept to ourselves.



HTTPS & SSL/TLS

- TLS and SSL are recognized as protocols to provide secure HTTP(S) for internet transactions.
- What are Digital Certificates?
- What is a Man-in-the-Middle Attack?
- High-level, how is data secured during a web-browsing session?



Open Web Application Security (OWASP)

1. Broken Access Control
 - Path traversal, enumeration, CSRF, CORS configuration
2. Cryptographic Failures
 - Not using HTTPS, encryption, hashing, or using broken algorithms.
3. Injection
 - SQL injection, XSS, etc
4. Insecure Design
 - Security questions, lack of bot detection/prevention
5. Security Misconfiguration
6. Vulnerable or Outdated components
7. Identification or Authentication failures
 - Missing multi-factor authentication. Don't allow for weak passwords.
8. Software and Data Integrity failures
9. Security and Logging Misconfiguration failures
10. Server-Side Request Forgery
 - Allowing access to objects or services that the server should have access to, but not the client.



Secure Consumer Practices

- Use lengthy passwords
- Never repeat your passwords
- Use a password manager
- Be wary of security questions
- Use multi-factor authentication, especially on your most critical accounts
 - Your email account is very critical
- Do not trust websites that are not using HTTPS
- Upgrade software to get latest security updates
- Backup your data
- Use Anti-Virus & Firewall protections
- Avoid Phishing



QUESTIONS?

