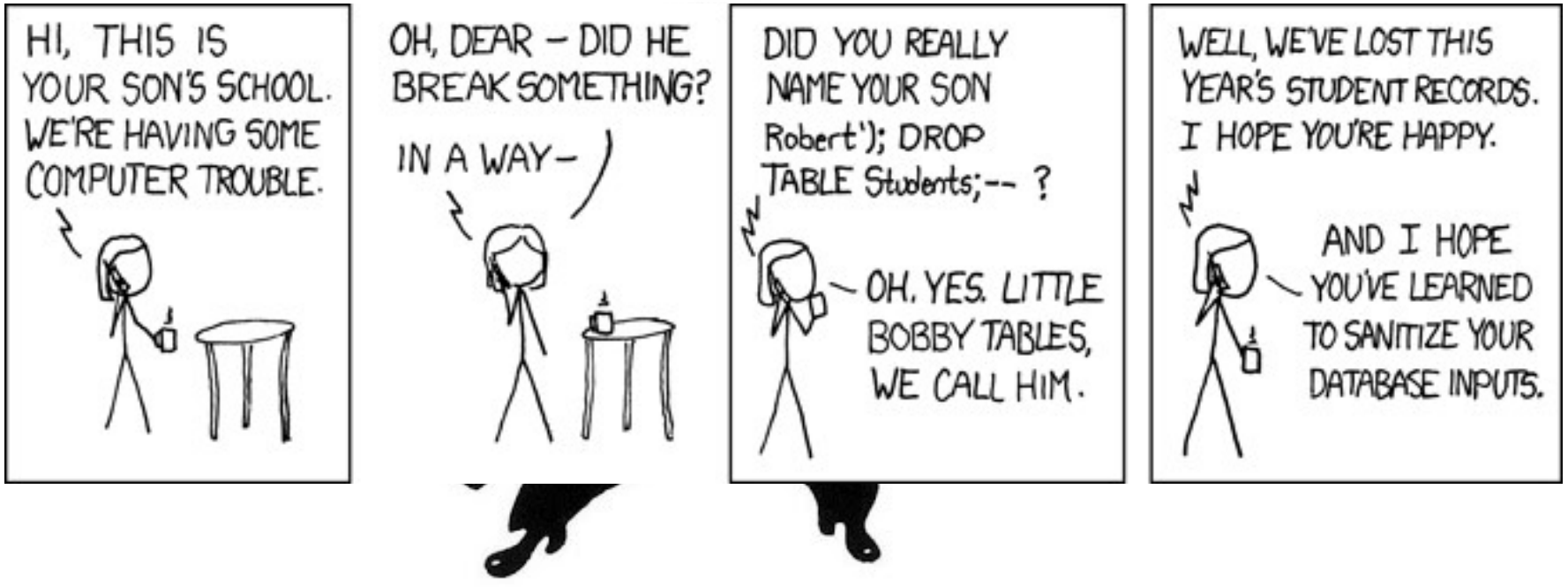
MODULE 2: DATABASE PROGRAMMING

Data Security

Stop the bad guys.



SQL Injection

1. We ask the user for their input:
    1. Please enter your name:
    2. Henry Edwards
2. We use this input to query our database
    1. string nameEntered = Console.ReadLine();
    2. string sqlCommand = “SELECT * FROM message WHERE private = FALSE AND sender_name =‘“ + nameEntered + “ ‘ ORDER BY create_date DESC”
3. We think that should be:
    1. SELECT * FROM message WHERE private = FALSE AND sender_name =‘Henry Edwards’ ORDER BY create_date DESC
4. Riiight?

SQL Injection

1. We ask the user for their input:
    1. Please enter your name:
    2. Robert'; Drop TABLE Students;--
2. We use this input to query our database
    1. string nameEntered = Console.ReadLine();
    2. string sqlCommand = "SELECT * FROM message WHERE private = FALSE AND sender_name ='" + nameEntered + " ' ORDER BY create_date DESC"
3. We think that should be:
    1. SELECT * FROM message WHERE private = FALSE AND sender_name ='**Robert'; Drop TABLE Students;--** ORDER BY create_date DESC
4. Ooops.

SQL Injection

1. We ask the user for their input:
    1. Please enter your name:
    2. ‘ OR ‘1’=‘1
2. We use this input to query our database
    1. string nameEntered = Console.ReadLine();
    2. string sqlCommand = “SELECT * FROM message WHERE private = FALSE AND sender_name =‘“ + nameEntered + “ ‘ ORDER BY create_date DESC”
3. We think that should be:
    1. SELECT * FROM message WHERE private = FALSE AND sender_name =‘**“ OR ‘1’=‘1**’ ORDER BY create_date DESC
4. Ooops.

Practice Safe Computing

- **Parameterized Queries:** If this is done consistently, SQL injection will not be possible
- **Input Validation:** Only allow certain values to be accepted. (Many of you did this in your Vending Machine)
- **Limit Database User Privileges**: A web application should always use a database user to connect to the database that has as few permissions as necessary. Never a good idea to use an admin's account

Safe Guarding Data

- Q: When you forget your password, how come the web site doesn't just send you the password?
- A: A secure web site can't see your password…ever.

1. We need to be able to *verify* a password but not *recover* it.
2. A system administrator with access to credential data should not be able to determine a password.
3. Any hacker that steals a database or set of credentials should not be able to read the passwords.
4. **Even with supercomputing capabilities, no one should be able to access the data within any reasonable amount of time**

Password Hashing

- Use a one-way function to obfuscate the plain-text password prior to storage.
- Use the password supplied by the user, re-hash it, and compare it to the stored password hash value.
- Salt the passwords in order to make it take longer to calculate all possibilities.

Hashing Requirements

- Input to outputs are constrained
  - Infinite inputs => limited output means duplicates
- Relationship between input and output should appear random
  - If TomA => asdfg then TomB should not be asdfh
- Inputs should be evenly distributed over the set of outputs

More Weapons in the good fight

- Encryption
  - The most effective way to achieve data security

- Securing Data at Rest
  - Data at rest can use a form of encryption called **symmetric key encryption**
  - 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
  - It may be necessary to allow others to send you secure data without worrying that it be intercepted.
  - Giving the secure key away would not be a good decision.
  - 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.

Securing the Web

- SSL and TLS
  - Secure Socket Layer and Transport Layer Security are examples of asymmetric key encryption.
  - SSL was developed by Netscape in 1994 to secure transactions over the WWW.
  - TLS and SSL are recognized as protocols to provide secure HTTP(S) for internet transactions. It supports authentication, encryption, and data integrity.
- Digital Certificates
  - Ownership of a public key is certified by use of a digital certificate allowing parties to rely upon the signature generated by the private key.
  - A certificate authority is a trusted third-party that provides the certificate.
  - The CA prevents the attacker from impersonating a server by indicating that the certificate belongs to a particular domain.

LET'S CODE!

WHAT QUESTIONS DO YOU HAVE?

Reading for tonight:

HTTP and Web APIs