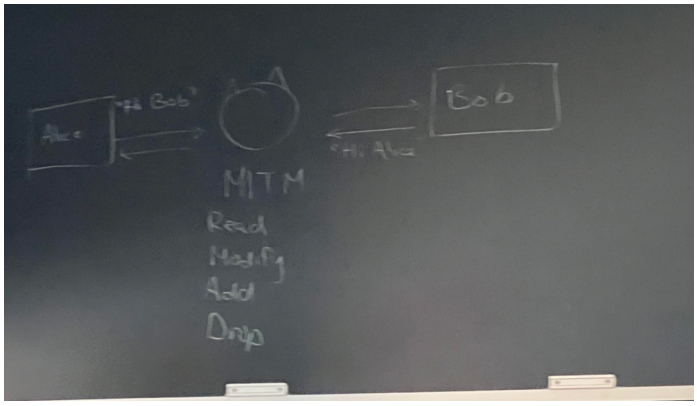
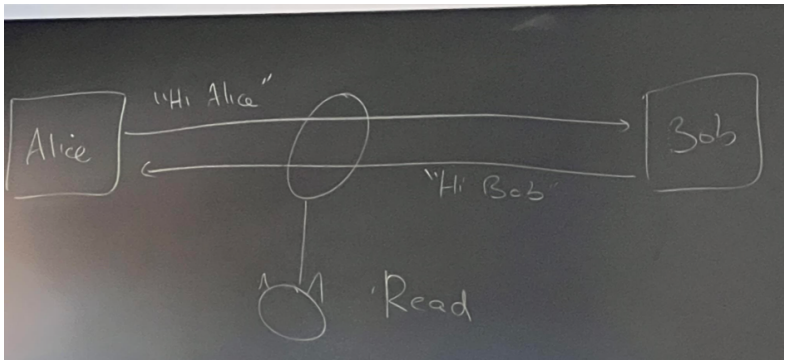
CAS CS 357

Class Note 2

1. Threat Model
    a. Evil maid attack
        1. You need physical access to the device
        2. 

    b. Man in the middle attack
        1. Need to be able to intercept (and potentially modify/add/drop) communications
        2. Examples -- Sting ray, Wi-Fi access point, Router
        3. 

    c. Eavesdropper
        1. Need to be able to send (possibly malformed) packets over the internet
        2. Examples -- Fiber optics splitter eavesdrop, Wi-Fi sniffer
        3. More powerful attack compared to “Man in the middle attack”

4. 

2. SQL Injection Attack
    a. The SQL Injection vulnerability is that form input is processed as SQL code
    b. SQL Injection is the subset of the an unverified/unsanitized user input vulnerability and the idea is to convince the application to run SQL code that was not intended
        1. The Target Intranet
            A. SQL code : SELECT fieldlist FROM table WHERE field = "steve@unixwiz.net"
                i. If the email "steve@unixwiz.net" does not exist in the table, it either returns "email address is unknown / We don't recognize your email address" or "error." If the response is an error, it is a dead giveaway that user input is not being sanitized properly and that application is ripe for exploitation
        2. Schema field mapping
            A. The first step is simply guessing field names since we do not know what the designer of the table named each column (email address or US Mail Address)

B. SQL Code: SELECT fieldlist FROM table WHERE field = ‘x’ AND email IS NULL; --’;

   i. If there exists an error here, we can assume that it occurred from the bad field name that we entered. However, if we get a valid response, it means that we guessed the name correctly.

   ii. We use AND conjunction here because the where statement will always return the value of false since no email address is simply ‘x’ and there will be no userid that is “Null” assuming that we guessed the field names correctly. If we return a true value, it might naturally ask to generate a new password to the user, causing problems

3. Finding the table name

   A. After knowing the fields, we now have to know the table name to continue attacking the data

   B. SQL Code: SELECT email, passwd FROM table where email = ‘x’ AND 1= (SELECT COUNT(*) FROM tabname);; --’;

      i. By writing code email = ‘x’ AND 1= (SELECT COUNT(*) FROM tabname);; --’;, we always get a value of false since email address of a person cannot be x and we have an “AND” conjunction. False and whatever value will always return false.

      ii. The code will print syntax error if the name of the table is incorrect in the code and will leave nothing once again if the

table name that we guessed is incorrect. We continue this step until we find the correct table name.

4. Finding some users
   A. After getting the table name, we only know one name of the user: who got the initial “here is your password” email that is chosen randomly according to the table arrangement.
   B. To find more names of the people, we use the SQL Code: “SELECT email, passwd, full_name FROM members WHERE email = ‘x’ OR full_name LIKE ‘%BOB%’;
      i. Note that %BOB% is a code of LIKE BOB in SQL so we can find names that simply includes BOB such as BOBBY, 123BOB, and etc.
      ii. Similarly, WHERE email = ‘x’ will always return false. However, if there is any name that matches the full_name LIKE ‘%BOB%’, it will return True since we are using an “OR” statement.
5. Brute-force password guessing
   A. We can try to use brute-force guessing of passwords at the login page, but many systems make an effort to detect or prevent such events that lead into account lockouts, or etc.
   B. SQL Code: SELECT email, passwd, login_id FROM members WHERE email = ‘bob@example.com’ AND passwd = ‘hello123’;

C. If the password is correct, there will be a message called “your password has been mailed to you”

6. The database isn’t readonly
   A. We can also do other things to the table such as adding new members to the table or deleting information such as the whole table
   B. SQL Code: SELECT email, passwd, login_id FROM members WHERE email = ‘x’; DROP TABLE members; --’;
      i. Here, the email = ‘x’ statement returns False and by using semicolon(;), we end the statement
      ii. After the statement, we can add new SQL code such as DROP TABLE members, which will delete all the table and the information that it contains
   C. SQL Code: SELECT email, passwd, login_id FROM members WHERE email = ‘x’; INSERT INTO members (‘email’, ‘passwd’, ‘full_name’) VALUES (‘abc@example.com’, ‘abc’, “ABC”); -- ’;
      i. Here, we added a new member to the able without proper registration by inserting values into a new row. However, in order to do this, we need to know all the columns, its order, and the data types of the corresponding columns that the table contains since there will be an error if the information does not match
7. Mail me a password
   A. We can even modify information of a person in the members table

B. SQL Code: SELECT email, passwd, login_id FROM members WHERE email = ‘x’; UPDATE members SET email = “newEmail@example.com’ WHERE email = ‘bob@example.com’;

   i. Here, after the false statement on email = ‘x’, we updated the email address of an user from bob@example.com to newEmail@example.com, which could be any email (even the attacker).

   ii. After updating such information, we can ask the website to send a password by clicking “I forgot my password” button on the website, which then the website will send the password to the email that we changed the information to. Therefore, we can now login to the website after checking the password that we received through email.