EECE 412, Fall 2007

Quiz #4

This quiz consists of 6 **pages. Please check that you have a complete copy. You may use both sides of each sheet if needed.**

Your Family name:

_____

Your Given name:

_____

Your student ID:

_____

#	Points	Out of
1		**5**
2		**4**
3		**10**
bonus		**7**
TOTAL		**19**

Name of your left neighbor: _____

Name of your right neighbor: _____

ATTENTION: When necessary, make reasonable assumptions and state them clearly in your solutions.

2. **(5 points)** Users tend to use a single password at many different web sites. By now there are several reported cases where attackers breaks into a low security site to retrieve thousands of username/password pairs and directly try them one by one at a high security e-commerce site such as eBay. To help avoid this problem, PwdHash was been developed.

   PwdHash is a browser extension/plugin that automatically replaces the contents of a user's password with a one-way hash of the pair (i.e., password and the host name of the site). A break-in at a low security site exposes password hashes rather than an actual password.

   Here is how it works: whenever a user wants to activate PwdHash, she just has to type "@@" in front of her actual password, as shown below:

   Consider the following scenario. You
   1. go to your bank's website,
   2. click on the link to log in,
   3. log in as usual EXCEPT you type "@@" in front of your password,
   4. your bank information is displayed as usual.

   Discuss the pros and cons of this approach in light of the following usability guidelines provided below.

   Usability guidelines
   Users should be:
   1. Be reliably made aware of the security tasks they must perform
   2. Be able to figure out how to successfully perform those tasks
   3. Not make dangerous errors
   4. Be sufficiently comfortable with the interface to continue using it
   5. Have sufficient feedback to accurately determine the current state of the system

Sample answer:
Guideline #1. Be reliably made aware of the security tasks they must perform
How well PwdHash follows this guideline: It's easy for the user to forget that she should type "@@" in the password field.

Guideline #2. Be able to figure out how to successfully perform those tasks
How well PwdHash follows this guideline: Again, if the user forgets what to type in order to activate PwdHash, then the only way to recall is to go back to the PwdHash documentation.

Guideline #3. Not make dangerous errors
How well PwdHash follows this guideline: If the user happens to be phished and forgets to type "@@" in front of her "master" password, then she would accidentally reveal the master password to the phishing site, which is a dangerous error.

Guideline #4. Be sufficiently comfortable with the interface to continue using it
How well PwdHash follows this guideline: The interface seems to be easy to use provided the user remembers to use it properly.

Guideline #5. Have sufficient feedback to accurately determine the current state of the system
How well PwdHash follows this guideline: There is no feedback indicating if PwdHash is activated and the properly computed hashes are sent to the web site.

In summary, the user interface design for PwdHash appears to follow only guideline #4. All others are not followed.

2. **(4 points)** Which of the following are effective (but not necessarily most feasible) ways for a software vendor to reduce the number of security vulnerabilities in the software system being developed by that vendor? (check all applicable)
   Sample answer:
   a. Tighten the perimeter controls, e.g., firewalls
   b. Apply coding and testing standards (e.g., safe string handling)
   c. Create threat models and design countermeasures to mitigate threats
   d. Improve software development processes in order to reduce the number of all (i.e., not only security ones) defects by order of magnitude
   e. Practice principles of designing secure systems
   f. Purchase insurance against security vulnerabilities
   g. Reduce the attack surface
   h. Make sure outsiders do not know the details of the software design and implementation
   i. Conduct code reviews
   j. Outsource software development to a an offshore software vendor
   k. Prepare plan for responding to reports on new vulnerabilities and corresponding exploits
   l. Perform penetration and other security testing
   m. Employ static analysis and other tools that can flag potential vulnerabilities so that the developers can review and correct the suspects
   n. Train developers, testers, program managers, and architects annually

3. **(10 points)** Explain how buffer overflow attacks work and what can be done to avoid, prevent, and detect them.

   For a sample the description of buffer overflow, see 13-25 of the module on "Developing Secure Software"

   Sample answer about avoidance, prevention, and detection:
   Avoidance can be achieved through the use of
   1. memory-safe languages and run-time environments (e.g., Java/JVM, .Net managed languages and CLR) that perform automatic automatic bounds checking.
   2. safe versions of buffer handling APIs
   3. allowing only validated input to be used by the program.

   Detection and prevention and can be achieved through canaries, and address randomization, and other similar techniques that cause process crashing instead of allowing malware to execute arbitrary code on successful buffer overflow.

4. **(7 points) Bonus question**: Explain how cross-site scripting attacks work and what can be done to avoid, prevent, and detect them.

   Sample answer:
   Cross-site scripting (XSS) attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user via posting the malicious code on a shared web site. The victim's browser has no way to know that the script should not be trusted, and will execute the script. Because the web browser grants the malicious script same privileges as to the authentic scripts from the trusted web site, the malicious script can access any cookies, session tokens, or other sensitive information retained by the victim's browser and used with that site.

   In order to avoid XSS vulnerabilities in a web application, it can be programmed to allow the input from web application users only if it contains exclusively characters and HTML constructs that cannot have XSS exploits.