Homework 5: Exam Review

This homework is due **Thursday, December 7** at **10 p.m.**. You will have a budget of five late days (24-hour periods) over the course of the semester that you can use to turn assignments in late without penalty and without needing to ask for an extension. You may use a maximum of two late days per assignment. Once your late days are used up, extensions will only be granted in extraordinary circumstances.

We encourage you to discuss the problems and your general approach with other students in the class. However, the answers you turn in must be your own original work, and you must adhere to the Code of Academic Integrity. Solutions should be submitted electronically via Canvas with the template at the end of this document.

Concisely answer the following questions. Limit yourself to at most 80 words per subquestion.

1. **Authentication.** Many organizations have deployed two-factor authentication through the use of key fob–sized devices that display pseudorandom codes at a fixed time interval. These codes are generated based on a built-in clock and a device-specific secret s that is also stored on a central authentication server tied to the user's account. Here is one way such a device might work: Let n be the number of minutes that have elapsed since the UNIX epoch; output the first 20 bits of $\text{HMAC}_s(n)$. Successful authentication requires the user's username and password and the current pseudorandom code from the user's device.

   (a) Name three common attacks against authentication that are mitigated by these devices.

   (b) Name one common attack against authentication that is not mitigated.

   Some devices use a counter instead of a clock and generate a single-use code each time the user presses a button on the device. One way this might work is as above, letting n be a register that is initially zero; upon each button press, display the current code for one minute and increment n on the device; on each successful authentication increment n on the server.

   (c) Describe one security advantage of single-use codes compared to time-based codes.

   (d) Describe one usability advantage of single-use codes compared to time-based codes.

   (e) A major usability problem with single-use codes in practice is that the counter on the device sometimes gets out of sync with the counter on the server (often as a result of inadvertent button presses in the user's pocket). Explain how we might extend the server to mitigate this without significantly reducing security.

As more and more organizations adopt these devices, end-users are burdened with carrying multiple devices, one for each entity to which they authenticate. Suppose instead that a central authority distributed and managed time-based devices (like the ones described above) for all users and companies, and allowed servers to verify a user's code through a public API.

(f) Describe at least three serious vulnerabilities that this would introduce.

2. **Web attacks.** Consider a fictitious social networking site called MyPlace. MyPlace has millions of users, not all of whom are particularly security-conscious. To protect them, all pages on the site use HTTPS.

(a) MyPlace's homepage has a "Delete account" link which leads to the following page:

```
<p>Are you sure you want to delete your account?</p>
<form action="/deleteuser" method="post">
  <input type="hidden" name="user" value="{{username}}"></input>
  <input type="submit" value="Yes, please delete my account"></input>
</form>
```

(The web server replaces {{username}} with the username of the logged-in user.)

The implementation of /deleteuser is given by the following pseudocode:

```
if account_exists(request.parameters['user']):
    delete_account(request.parameters['user'])
    return '<p>Thanks for trying MyPlace!</p>'
else:
    return '<p>Sorry, ' + request.parameters['user'] + ', an error occurred.</p>'
```

Assume that the attacker knows the username of an intended victim. What's a simple way that the attacker can exploit this design to delete the victim's account without any direct contact with the victim or the victim's browser?

(b) Suppose that /deleteuser is modified as follows:

```
if validate_user_login_cookie(request.parameters['user'], request.cookies['login_cookie']):
    delete_account(request.parameters['user'])
    return '<p>Thanks for trying MyPlace!</p>'
else:
    return '<p>Sorry, ' + request.parameters['user'] + ', an error occurred.</p>'
```

where validate_login_cookie() checks that the cookie sent by the browser is authentic and was issued to the specified username. Assume that login_cookie is tied to the user's account and difficult to guess.)

Despite these changes, how can the attacker use CSRF to delete the victim's account?

(c) Suppose that the HTML form in (a) is modified to include the current user's login_cookie as a hidden parameter, and /deleteuser is modified like this:

```
if request.parameters['login_cookie'] == request.cookies['login_cookie'] and
        validate_login_cookie(request.parameters['user'], request.cookies['login_cookie']):
    delete_account(request.parameters['user'])
    return '<p>Thanks for trying MyPlace!</p>'
else:
    return '<p>Sorry, ' + request.parameters['user'] + ', an error occurred.</p>'
```

The attacker can still use XSS to delete the victim's account. Briefly explain how.

3. **Networking.** The TCP protocol is intended to provide the abstraction of an in-order stream of bytes between a sending and receiving application across a network. Each TCP packet contains a sequence number that is used to correctly order packets that may be received out of order.

   (a) In the original specification, the initial sequence number was to be chosen based on clock time. Describe how an adversary could inject data into a TCP stream between two parties in this case.

   (b) Most modern implementations use a random initial sequence number. Could an adversary still inject malicious data into a TCP stream in this case? Describe how such an attack would work.

   (c) What countermeasures could an application take at the application layer to protect against these attacks?

   In response to fears of surveillance by governments or service providers, some parties have recommended using VPN services or Tor to protect web browsing or other activities on the internet.

   (d) Give an argument for using a VPN over Tor. What threats does a VPN protect against? Does a VPN prevent the TCP injection attack described above? Against what adversaries?

   (e) Give an argument for using Tor over a VPN. What threats does Tor protect against? Does Tor prevent the TCP injection attack described above? Against what adversaries?

   (f) What security measures would you recommend an individual concerned about web browsing privacy take?

4. **Secure programming.** StackGuard is a mechanism for defending C programs against stack-based buffer overflows. It detects memory corruption using a *canary*, a known value stored in each function's stack frame immediately before the return address. Before a function returns, it verifies that its canary value hasn't changed; if it has, the program halts with an error.

(a) In some implementations, the canary value is a 64-bit integer that is randomly generated each time the program runs. Why does this prevent the basic form of buffer-overflow attacks discussed in lecture?

(b) What is a security drawback to choosing the canary value at compile time instead of at run time? If the value must be fixed, why is 0 a particularly good choice?

(c) No matter how the canary is chosen, StackGuard cannot protect against all buffer overflow vulnerabilities. List two kinds of bugs that can corrupt the stack and allow the adversary to take control, even with StackGuard in place.

5. **Ethics.** Consider the following scenario: A worm is infecting systems by exploiting a bug in a popular server program. It is spreading rapidly, and systems where it is deleted quickly become reinfected. A security researcher decides to launch a counterattack in the form of a defensive worm. Whenever a break-in attempt comes from a remote host, the defensive worm detects it, heads off the break-in, and exploits the same bug to spread to the attacking host. On that host, it deletes the original worm. It then waits until that system is attacked, and the cycle repeats.

   (a) Many people would claim that launching such a counterattack in this scenario is ethically unacceptable. Briefly argue in support of this view.

   (b) Are there circumstances or conditions under which an active security counterattack would be ethically justified? Briefly explain your reasoning.

Submission Template

Submit by uploading a `txt` file to Canvas. Use the template below to organize your submission. Follow the headings precisely for grading purposes. You may use LaTeX-style math syntax if you wish.

```
# Problem 1

1a. [Answer ...]

1b. [Answer ...]

1c. [Answer ...]

1d. [Answer ...]

1e. [Answer ...]

1f. [Answer ...]

# Problem 2

2a. [Answer ...]

2b. [Answer ...]

2c. [Answer ...]

# Problem 3

3a. [Answer ...]

3b. [Answer ...]

3c. [Answer ...]

3d. [Answer ...]

3e. [Answer ...]

3f. [Answer ...]
```

```
# Problem 4

4a. [Answer ...]

4b. [Answer ...]

4c. [Answer ...]

# Problem 5

5a. [Answer ...]

5b. [Answer ...]
```