ITIS 6200/8200 Principles of Information Security and Privacy

Homework 3

**Question 1. Access Control (10 points)**

Alice can read and write to the file x, can read the file y, and can execute the file z. Bob can read x, can read and write to y, and cannot access z.

Q 1.1: Write a set of access control lists for this situation. Which list is associated with while file?

Q 1.2: Write a set of capability lists for this situation. With what is each list associated?

**Question 2. Cookies (20 points)**

Q 1.1: For each of the following webpages, determine whether the webpage has the same origin as <https://cci.charlotte.edu>, and provide a **brief** justification.

1. <https://cci.charlotte.edu/sis-faculty/>
2. <https://www.charlotte.edu>
3. <https://cci.charlotte.edu:443>
4. <https://cci.charlotte.edu/departments/department-of-software-information-systems/>
5. [http://cci.charlotte.edu/](http://cci.charlotte.edu/sis-faculty/)

Q 1.2: Describe how to setup a cookie so it will be sent to only <https://cci.charlotte.edu> and its subdomains.

Q 1.3: How can <https://cci.charlotte.edu> ensure that cookies are only transmitted encrypted?

Q 1.4: How can <https://engr.charlotte.edu/> set a cookie it may affect <https://cci.charlotte.edu>?

Q 1.5: **github** hosts user sites on **github.io** instead of **github.com**, i.e., **[username].github.io,** why do you think **github** do that? Why don’t Github host user sites as **[username].github.com?**

How can it help defend against cookie related attacks?

**Question 3. CSRF (25 points)**

A CSRF attack exploits cookie-based authentication to perform an action as the victim. Consider the following example. Mallory posts the following in a comment on a chat forum:

*<img src="https:// bank.com/transfer?amount=1000&to=Mallory"/>*

To successfully conduct a transaction in *bank.com*, users need to authenticate first. Then, *bank.com* sets a cookie as session taken.

Q 2.1: Explain what could happen when Alice visits the chat forum and views Mallory’s comment.

Q 2.2: Suppose *bank.com* decides to defend against CSRF attacks with a cookie-based CSRF token, as follows:

1. When a user logs in, *www.bank.com* sets a cookie csrf\_token randomly with domain attribute of *bank.com*.
2. When the user sends a POST request, the value of the csrf\_token is embedded as one of the form fields.
3. On receiving a POST request, califlower.com checks that the value of the csrf\_token cookie matches the one in the form.

If the chat forum has domain *evil.com*, can the CSRF attack against *bank.com* succeed? If the chat forum has domain *evil.bank.com*, can the CSRF attack against *bank.com* succeed?

Q 2.3: Suppose *bank.com* decides to defend against CSRF attacks by checking if the **Referer** header contains a string “bank.com”. Will the attack above work? Why or why not?

Q 2.4: Describe one way Mallory can modify her attack to always get around this check

Q 2.5: Suppose *bank.com* decides to defend against CSRF attacks with an additional cookie field **SameSite**. When **SameSite**=strict, the browser will only send the cookie if the requested domain and origin domain correspond to the cookie’s domain. Which CSRF attacks will this stop? Which ones won’t it stop? Give one big drawback of setting **SameSite**=strict.