Assignment 5: Web Security and Authentication Logic
15-316 Software Foundations of Security and Privacy

Due: **11:59pm**, Tuesday 4/24/18
Total Points: 50

1. **Spot the vulnerabilities (15 points).** Consider the PHP script below, which consults a back-end SQL database to perform password authentication of users. List the vulnerabilities present in this code, explaining what type each vulnerability is and how it can be exploited. Finally, briefly state what method you would use to prevent them. *Note: you should be able to complete this problem from the brief introduction to PHP and examples from the lecture notes. However, if you are uncertain of something, consult the PHP manual at http://php.net/manual/en/langref.php.*

```
<!DOCTYPE html>
<html>
  <body>
    <div class = "container form-signin">
       <?php
          if(!empty($_GET[’uname’]) && !empty($_GET[’passwd’])) {
            $uname = $_POST[’uname’];
            $passwd = $_POST[’passwd’];
            $sql = "SELECT id FROM admin WHERE user = ’$uname’ and pass = ’$passwd’";
            $result = mysqli_query($db,$sql);
            $count = mysqli_num_rows($result);
            // If result matched $uname and $passwd, row count must be 1
            if($count == 1) {
              echo ’Login successful. Welcome, ’ . $uname . ’!’;
            } else {
              echo ’Invalid password for user ’ . $uname . ’.’;
              echo ’<a href="’ . $_GET[’login_src’] . ’">Click here</a> to try again.’
            }
          }
       ?>
    </div> <!-- /container -->
   </body>
</html>
```

2. **It sounded like a good idea at the time (10 points).** JavaScript Object Notation (JSON) is a human-readable format that is commonly used within web applications to transport data between client and server. In a nutshell, a JSON datum is simply a textual representation of a Javascript object, using the same notation as the Javascript language for object literals. So for example, a JSON object that represents the name and age of a person might read as follows.

```
{
  "name": "John Smith",
  "age": 26,
}
```

   JSON objects are typically retrieved using the `XMLHttpRequest` method. *Note: for more information on JSON in general, refer to the Wikipedia page.*

   Because `XMLHttpRequest` is subject to the same-origin policy, developers have looked for ways to circumvent this to return JSON from remote domains. One approach utilizes the fact that `<script>` tags are not subject to SOP, so it is allowed to do the following.

```
<script src="http://otherdomain.com/getuser.php?name=John">
```

If the server hosting `otherdomain.com` returns a script containing the JSON-encoded data, then the page with the `<script>` tag will be able to read it. Suppose that the JSON contains sensitive information, but that the developer of `otherdomain.com` anticipated this and will only respond to users who present a valid session cookie obtained after logging in. Explain how a site hosted at `malicious.com` could still learn this sensitive information, and what sort of attack they would need to mount.

3. *A* **speaks for** *B* **(15 points).** Professors tend to have full schedules, which is why the CS department assigns them assistants to help with administrative work. Suppose that `mfredrik` wishes to *delegate* his authority to claim students using $\texttt{studentOf}(x, \texttt{mfredrik})$ to his assistant `bcook`, so that statements of the form $\texttt{bcook says studentOf}(x, \texttt{mfredrik})$ are treated the same as statements of the form $\texttt{mfredrik says studentOf}(x, \texttt{mfredrik})$.

   - **Part 1 (5 points).** Write an authorization logic policy formula Q_d that accomplishes this.
   - **Part 2 (10 points).** Use your policy from Part 1, in addition to the formula wherein `bcook` says that `tli2` is a student of `mfredrik`, i.e. $Q_b \equiv \texttt{bcook says studentOf}(\texttt{tli2}, \texttt{mfredrik})$, to prove the judgement below.

   $$Q_d, Q_b \vdash \texttt{mfredrik says studentOf}(\texttt{tli2}, \texttt{mfredrik})$$

   - **Part 3 (extra credit, 5 points).** Use your policy from Part 1, Q_b from Part 2, and P_1, P_2, Q_1 from the running example in Lecture 20 to prove the judgement below.

   $$Q_d, Q_b, P_1, P_2, Q_1 \vdash \texttt{admin says canOpen}(\texttt{tli2}, \texttt{cic2126})$$

4. **Unfinished business (10 points).** Prove Theorem 3 from lecture 20, i.e., the judgement $\vdash (A \texttt{ says } (A \texttt{ says } P)) \to (A \texttt{ says } P)$.