ECE 478 / CS 419: **Network Security**
Homework 1: due Friday April 11, 10am

Instructions:

- Submit typed solutions, following the rules in the syllabus.
- Include the "disclaimer" given in the syllabus!
- "Describe an attack" means show the final result (in this case, a maliciously crafted URL) and also describe *what it does, why it does it, and your thought process of how you found it* (as if you were reporting a vulnerability to a software author).

1. How does your web browser handle unknown attributes in an HTML tag, such as:

   ```
   <a href="something" unknown="blah">
   ```

2. Study this simple form and how it works:

   http://web.engr.oregonstate.edu/~rosulekm/netsec/hw/xss-hw1.php

   Also check out the image at /~rosulekm/netsec/hw/skull.gif. Describe how to use an XSS attack to make the skull appear in place of the colored image on xss-hw1.php. Please note the file extension on skull.gif!

   Hint: include a quote character in the `color` parameter.

3. What is the HTML syntax (in general, not specific to the page above) to create an image that causes an `alert` box to appear when the users moves the mouse over the image?

   Note: there are a million ways to do this, but I'm looking for the simplest one that doesn't involve any `<script>` tags.

   Hint: search for `onmouseover`.

4. Describe an XSS attack on xss-hw1.php that causes an `alert` box appear when the user moves the mouse over the colored image.

   Note: xss-hw1.php does not allow the `color` parameter to include ">" characters, hence the restriction in the previous problem.

5. Describe an attack that achieves the same effect as in question 2 (i.e., display skull.gif), but on this page instead:

   http://web.engr.oregonstate.edu/~rosulekm/netsec/hw/xss-hw2.php

   The twist is that this page does not allow the `color` parameter to include quotes.

   Hint: is there a way to make a URL string in which the final so-many characters don't really have an effect on what file was requested?

6. The attacks in questions 2 & 5 are not stopped by Chrome's XSS auditor. Discuss why someone might still be concerned about such an XSS attack. What kind of information could an attacker gain through this attack vector?

7. Look at the following web page:

   http://web.engr.oregonstate.edu/~rosulekm/netsec/hw/xss-hw3.php

   Does it set any non-standard HTTP headers?

8. Read the following blog post:

   http://seancoates.com/blogs/xss-woes

   With that in mind, describe an XSS attack on xss-hw3.php to construct a URL that causes the form to POST to

   http://web.engr.oregonstate.edu/~rosulekm/netsec/hw/xss-hw-eviltarget.php

   In other words, when using your URL, an unsuspecting victim who submits the form will actually be submitting to a "bad" target, not the target intended by the developer.

   Note: the PHP installation on the `engr` server seems to have slightly different behavior than described in the blog post. So you'll have to try things, study how this one behaves, and do something slightly different.