Assignment 7

Mark, Henry, Jun, Tolu, Philip

**XXS :** NAME

IP :

1. What vulnerability did you exploit? How does it work?
2. Is your application subject to this particular vulnerability? Why/why not?
3. How will you protect against this vulnerability in your application?

**PICTURE HERE**

**SQL Injection :** NAME

IP :

1. What vulnerability did you exploit? How does it work?
2. Is your application subject to this particular vulnerability? Why/why not?
3. How will you protect against this vulnerability in your application?

**PICTURE HERE**

**CSRF :** NAME

IP :

1. What vulnerability did you exploit? How does it work?
2. Is your application subject to this particular vulnerability? Why/why not?
3. How will you protect against this vulnerability in your application?

**PICTURE HERE**

**OS Command Injection :** NAME

IP :

1. What vulnerability did you exploit? How does it work?
2. Is your application subject to this particular vulnerability? Why/why not?
3. How will you protect against this vulnerability in your application?

**PICTURE HERE**

**Unrestricted File Upload :** Mark White

IP : 81.185.170.49

1. What vulnerability did you exploit? How does it work?

The first exploit was uploading unrestricted files by ignoring client-side restrictions by editing HTML with inspect element on Safari. I noticed that img and js directories were viewable as file systems. Inside the img directory, there is another directory called uploads which is where the uploaded are stored. One can access the uploaded file by navigating the directories, thus the uploaded file executed. <http://3.80.200.34/xvwa/img/uploads/m.php>

Using php, we can then modify the other files in the website and potential operating systems.

1. Is your application subject to this particular vulnerability? Why/why not?

Potentially, all images are uploaded to firebase which probably limits the permissions of the client to execute files on firebase servers. Locally on the IOS app, IOS restricts app abilities to execute code and services so any damage would be local to the app.

1. How will you protect against this vulnerability in your application?

Minimise the app privileges on the IPhone and do better file type checking rather than name checking.

