Arctos Morei  
  
**Week 15 Homework Submission File: Web Application**

**#### Web Application 1: Your Wish is My Command Injection**

Text

Description automatically generated

Text

Description automatically generated

Allowing command line inputs with the ping command on this website allows malicious users to inject code following a ping using “&&” followed up with any command they want to run afterwards. This is incredibly dangerous, and I would recommend disabling the ability for users to browse other directories not intended for access, and/or disabling the ability to add additional inputs into this field aside from IP addresses.

**#### Web Application 2: A Brute Force to Be Reckoned With**

Graphical user interface, table

Description automatically generated with medium confidence

I was easily able to run a brute force attack for 100 attempts and on attempt #75 with the username tonystark and password I am Iron Man, was able to get a successful login prompt. This can be prevented in the future by having a limit to unsuccessful login attempts (standard is 3) and Multi Factor Authentication (MFA). Additionally, it was mentioned that there was a systems breach that created this vulnerability, and all of these passwords contain easy to guess words. I would additionally like to recommend that a policy should be put in place for required password complexity to further strengthen security.

**#### Web Application 3: Where's the BeEF?**

Graphical user interface, text, application, email

Description automatically generatedGraphical user interface, text, application, email

Description automatically generated

An attempt was made to prevent XSS by creating a character limit in the message field; however, this was easily modified by opening the Inspect Element tool and editing maxlength="50” to “500”. I was then able to inject the code and created a BeEF hook, where I ran a couple vulnerability commands.   
  
To prevent XSS in the future, I would suggest using Input Validation to determine exactly what users can and cannot input here. Now considering this is a free-form text field, that can be a little tricky because there may be various, valid reasons why a user might need to use a ‘, “, or <. In this case, I would specifically recommend using HTML Encoding so that any HTML specific characters will be converted into their encoded versions.

More on this can be found here, specifically with Matt S’s reply:  
<https://stackoverflow.com/questions/1996122/how-to-prevent-xss-with-html-php>