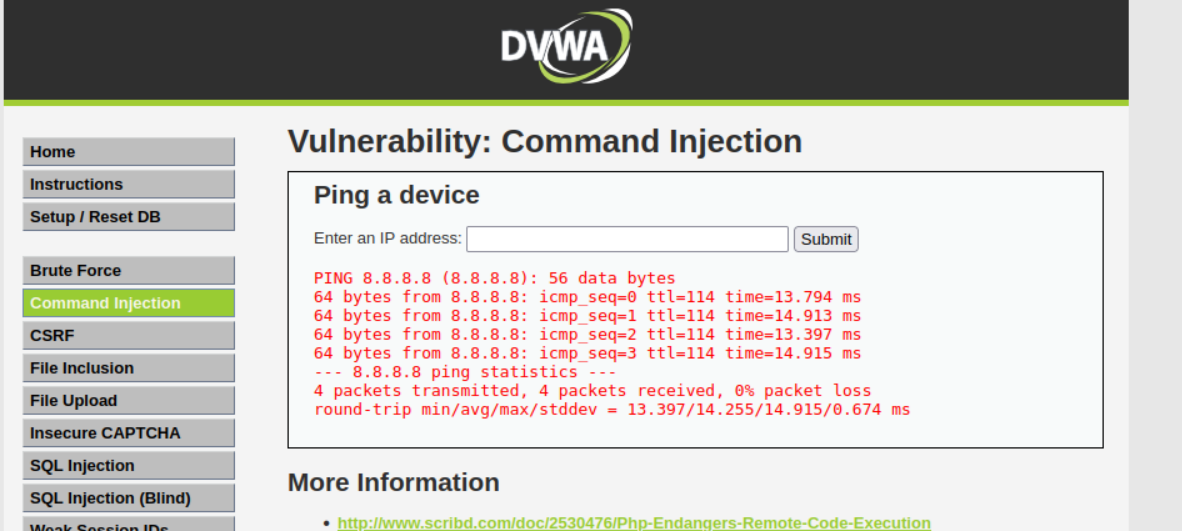
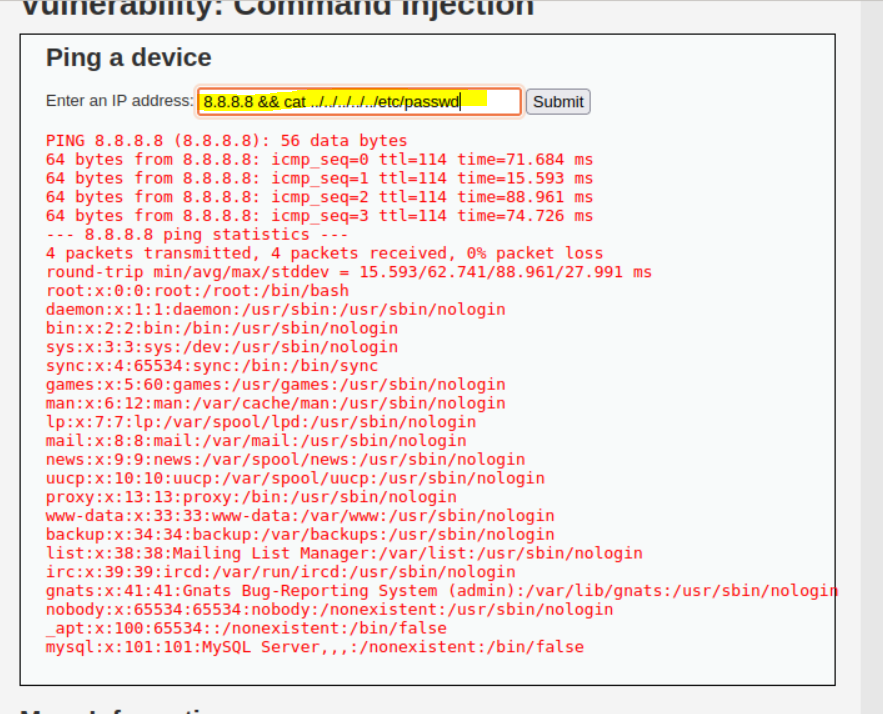
### **Web Vulnerabilities Project**

### 

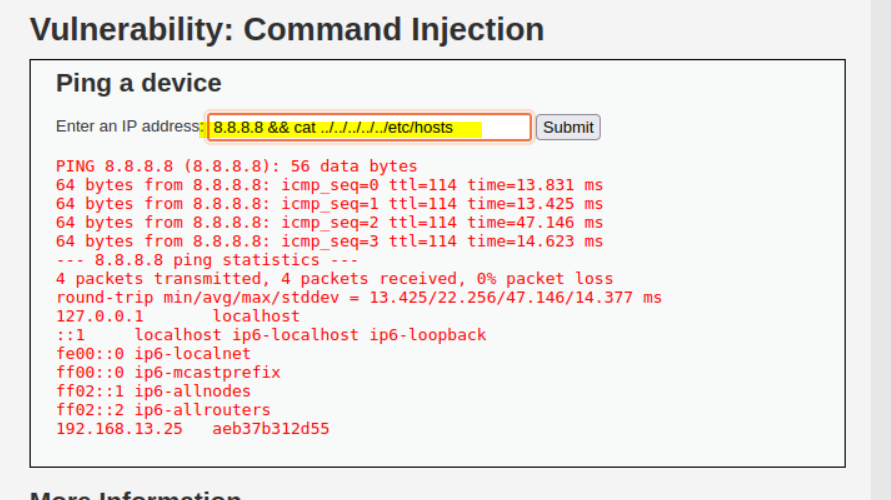
### **Web Application 1: *Your Wish is My Command Injection***

Complete the following steps to walk through the intended purpose of the web application.  
Test the webpage by entering the IP address 8.8.8.8. Press Submit to see the results display on the web application.  
  
 

Test if we can manipulate the input to cause an unintended result.  
On the same webpage, enter the following command (payload) in the field: 8.8.8.8 && pwd  
Press Enter. Note the ping results are the results of the second pwd command:  
  
 

Now that you have determined that Replicants new application is vulnerable to command injection, you are tasked with using the dot-dot-slash method to design two payloads that will display the contents of the following files:  
–/etc/passwd :  **I inserted 8.8.8.8 && cat ../../../../../etc/passwd**

– /etc/hosts  **I inserted 8.8.8.8 && cat ../../../../../etc/hosts**



**Recommended Mitigation Strategies:**:

1. Using built-in APIs instead of OS commands. Attackers can manipulate OS commands and easily inject code from the command line.
2. Input validation can prevent command injection as well. Developers must use a whitelisting approach to validate the user input.
3. Regularly do updates and patches to applications and databases.

Reference page for my mitigation strategies.

<https://www.imperva.com/learn/application-security/command-injection/>

### 

### 

### 

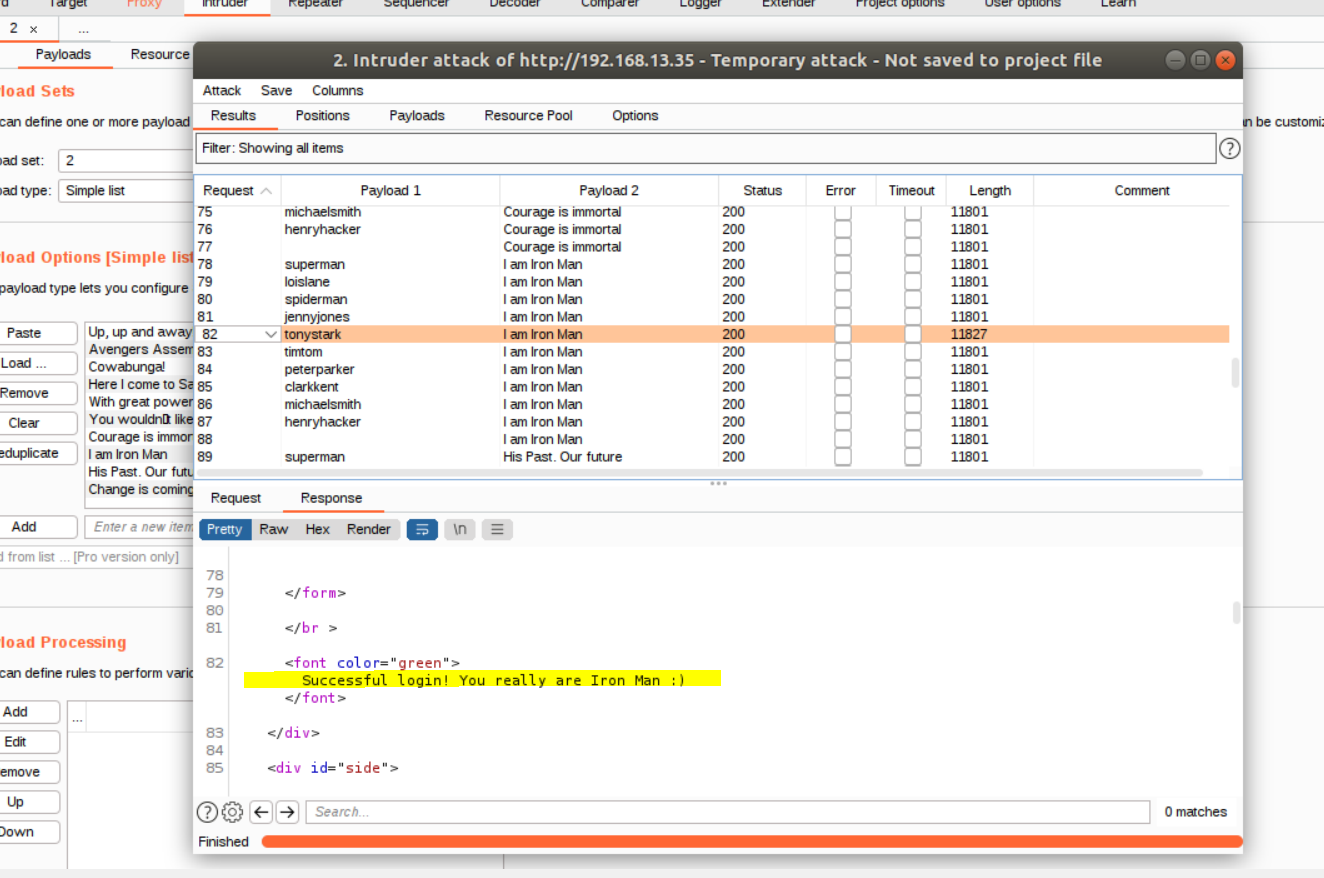
### 

### 

### 

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

**Deliverable**: Take a screen shot confirming that this exploit was successfully executed and provide 2-3 sentences outlining mitigation



Blocking Brute Force Attacks

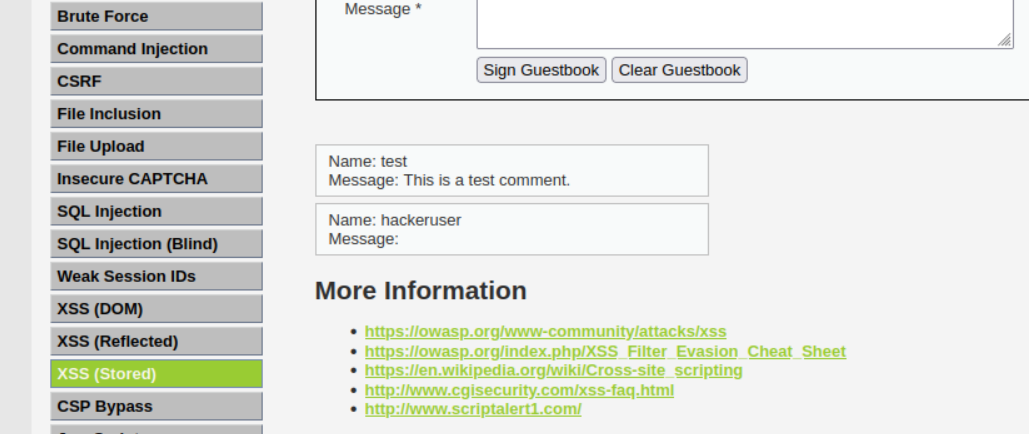
The most obvious way to block brute-force attacks is to simply lock out accounts after a defined number of incorrect password attempts. Account lockouts can last a specific duration, such as one hour, or the accounts could remain locked until manually unlocked by an administrator.

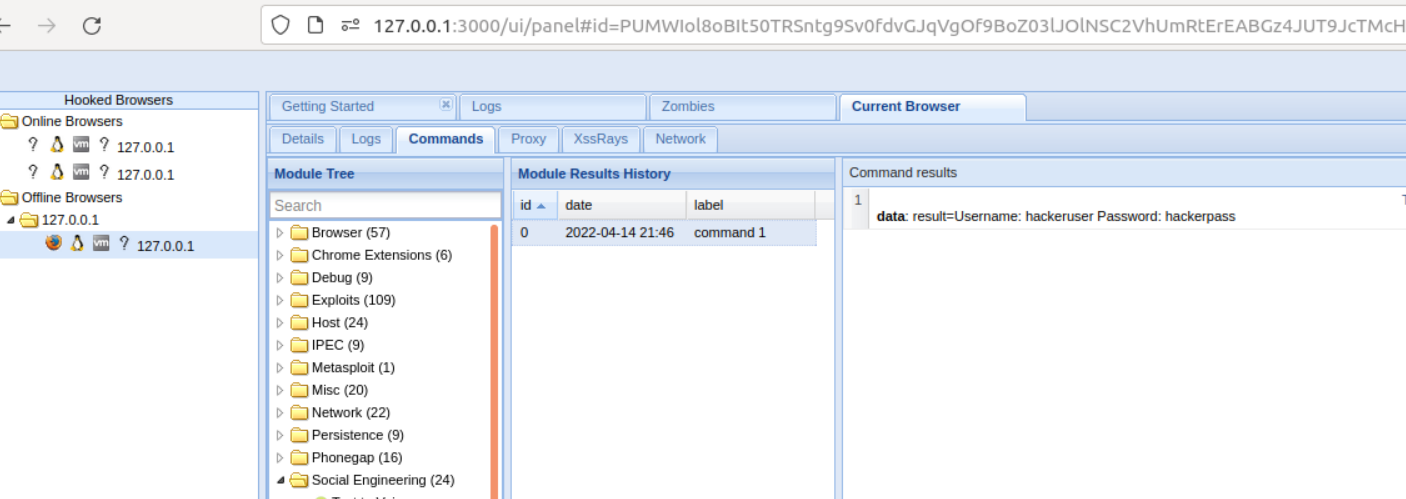
Reference page for my mitigation strategies.

<https://owasp.org/www-community/controls/Blocking_Brute_Force_Attacks>

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

**Deliverable**: Take a screen shot confirming that this exploit was successfully executed and provide 2-3 sentences outlining mitigation strategies.





### **Outlining mitigation strategies against this kind of attacks**

Keep your system up to date.

The majority of browser based exploits require JavaScript in some capacity. NoScript helps mitigate these attacks.

The Google Safe browsing API is used by Firefox and Chrome by default to prevent you from reaching a site that is known to be leveraging browser based attacks. However this does nothing for a targeted BeEF attack.

Some Anti-Viruses will plug into your browser and prevent an exploit from loading. This could prevent a targeted BeEF attack. But AV's aren't perfect and can be fooled.

Plan on failure, consider doing most of your day to day browsing inside a VM. Restore this VM to a virgin state on a regular basis (once a week, or once a month). Assume that you have been compromised and change your passwords regularly.

Reference page for my mitigation strategies.

<https://security.stackexchange.com/questions/22828/what-are-methods-for-preventing-browser-hooking-drive-by-downloads>