PWN LAB

Name: Ritesh Bardikar

Date: 8/8/2025

Email : riteshbardikar@gmail.com

# Table Of Contents

|  |  |  |
| --- | --- | --- |
| 1. | **Executive Summary** | 3 |
| 2. | **Attack Narrative** | 4 |
| 3. | **Conclusion** | 14 |
| 4. | **Recommendation** | 14 |

Executive Summary:

I have performed security assessment on 00:0C:29:77:EF:8E(PWN Lab) machine. My objective was to recover the machine's flags, and in order to do so, I had to get over a number of security flaws. Through a simulated attack, I was able to compromise sensitive systems, demonstrating a lack of adequate defences against the cyber threats. I got various weakness present that can further be used for any other activities which I had not used here for gaining the flag.

Potential threats of the security weakness may contain:

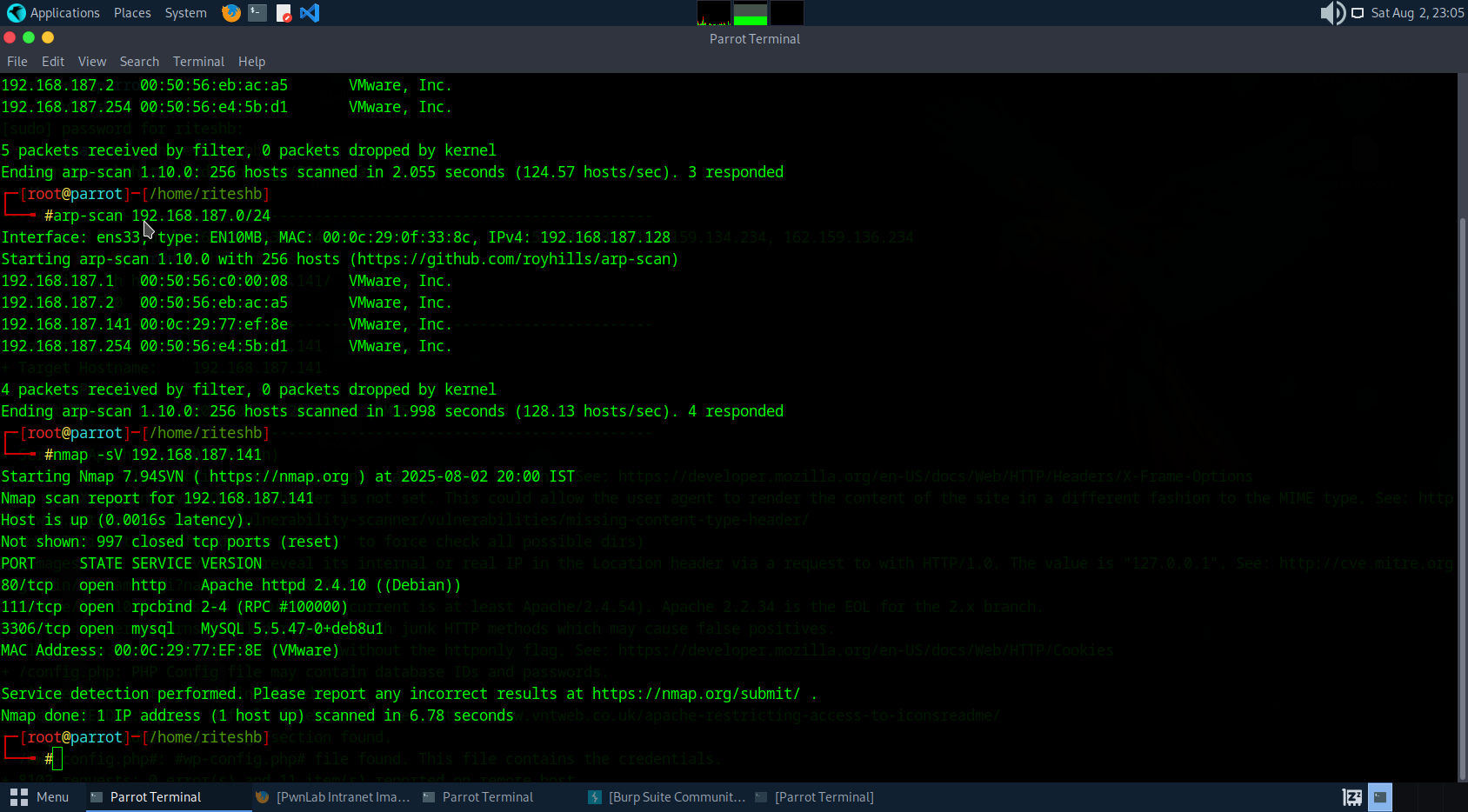
* Misusing the open ports available.
* Can manipulate the data through various methods.
* Gaining the shells access.
* Gaining the admins access.
* Backdoor can be created to future.
* Malicious file may get uploaded.
* Hidden files and directory data can be retrieved.

Summary of the Result:

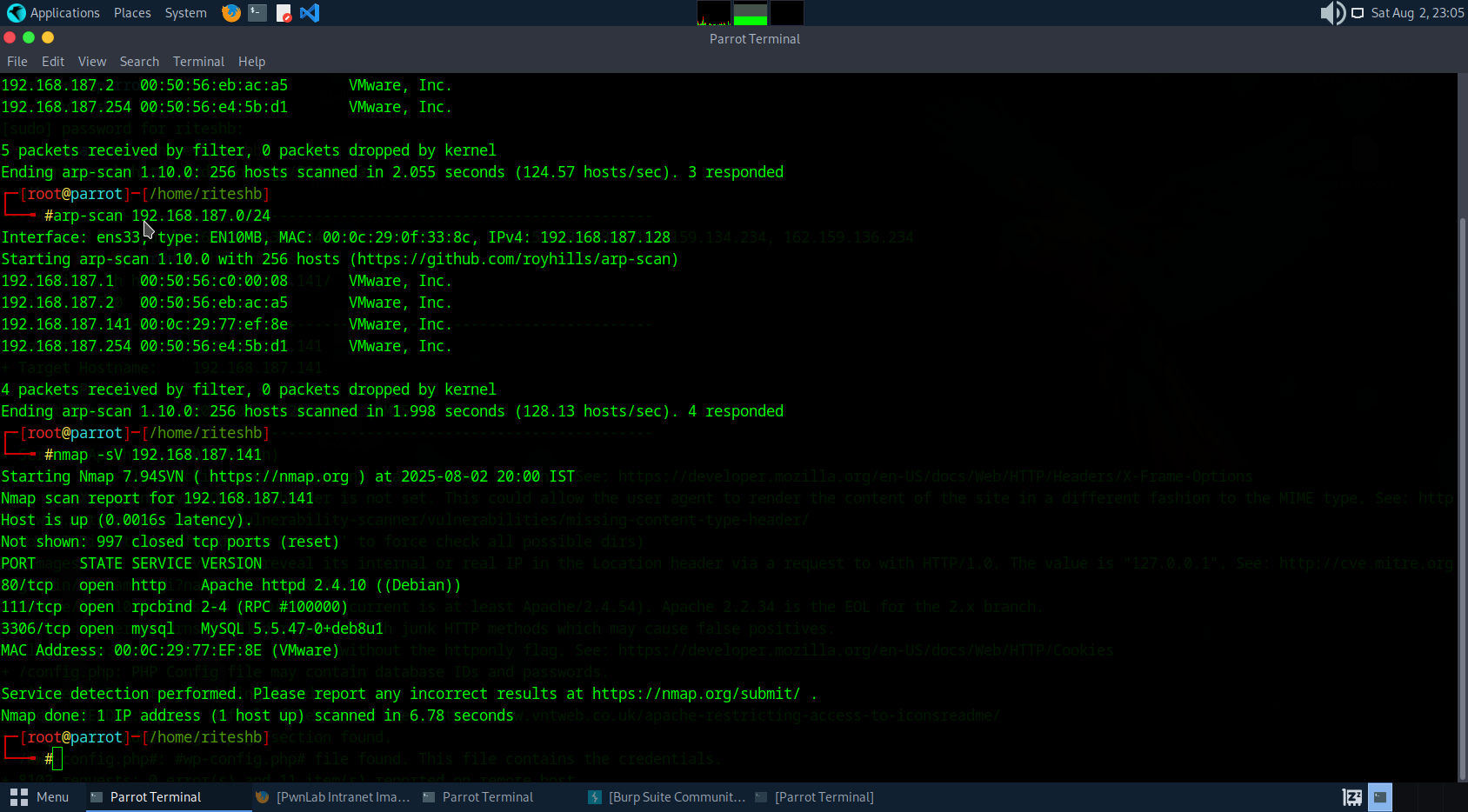
While conducting the security assessment, I discovered a number of vulnerabilities during the security evaluation that could allow an attacker to penetrate the system and alter the database that is already there. Through a number of existing vulnerabilities present, the attacker can get unauthorized access to the database. Attackers can obtain user credentials while gaining access to the database, which will enable them to utilize spoofing to carry out additional harmful actions. I found that an attacker can gain access to the shell, and with additional exploits, they can gain terminal access. Additionally, the attacker will elevate their privileges to the root level after carrying out additional command line execution. Attackers have the ability to alter data and carry out other unethical actions that could endanger the user. This site contains a number of high-risk vulnerabilities that could give an attacker the ability to obtain a ransom from an administrator or user.

Attack Narrative:

Gathering information of the system like getting IP address.

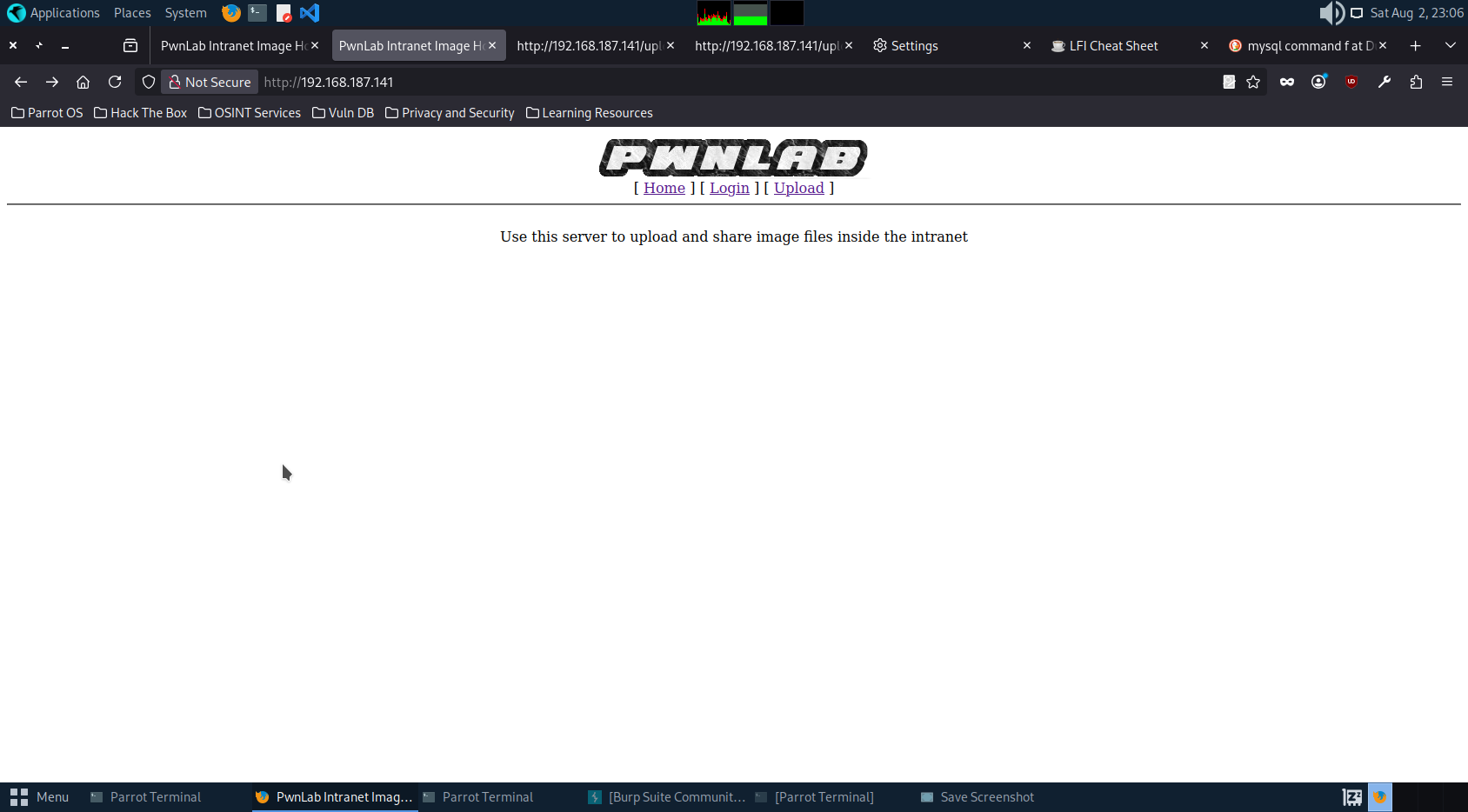


Scanning to get information on the open ports.

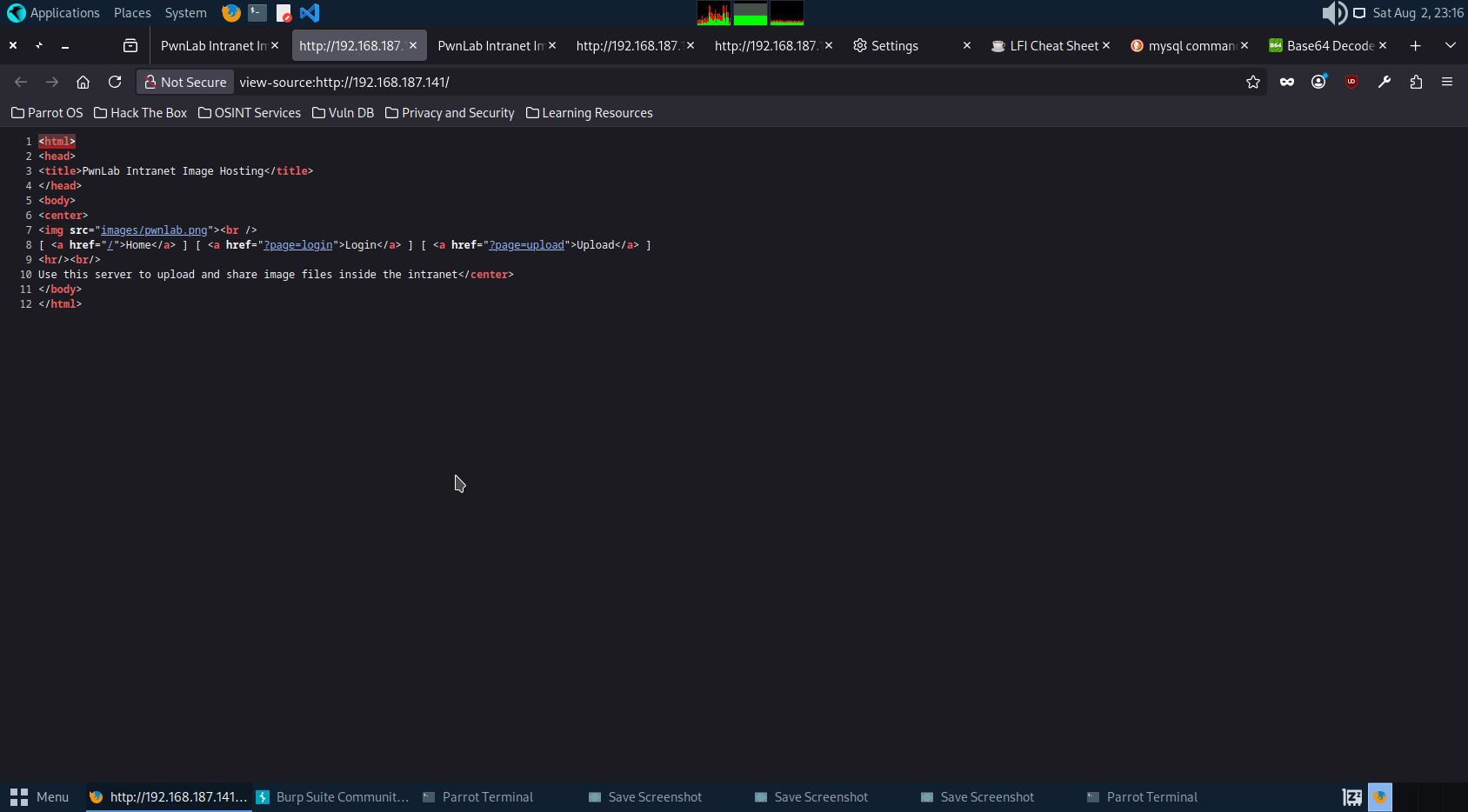


Http port is open hence checking for the website vulnerabilities.

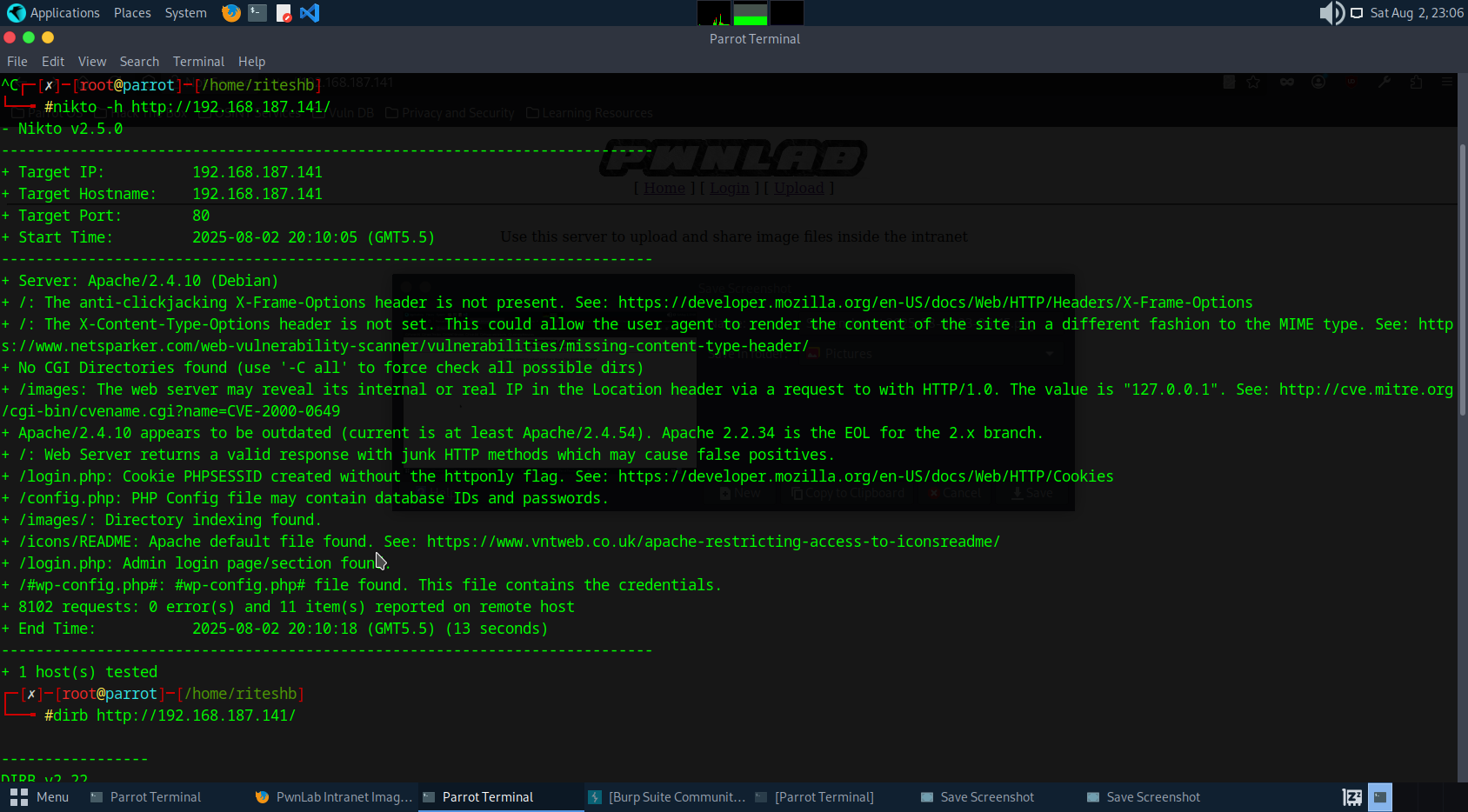
Analysing the index page.



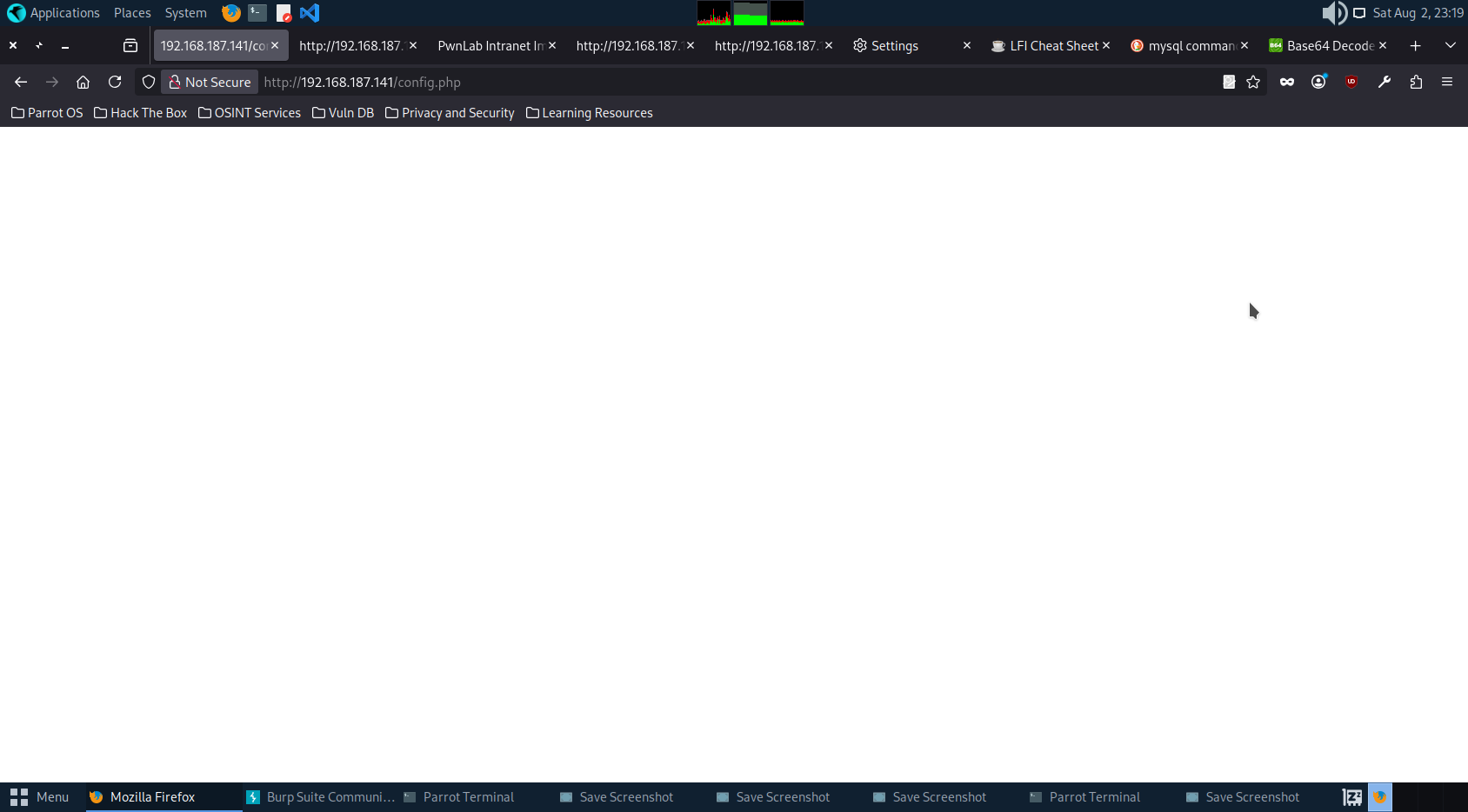
Didn’t got any information on the source code of the index page.



Scanning the website to get more information.

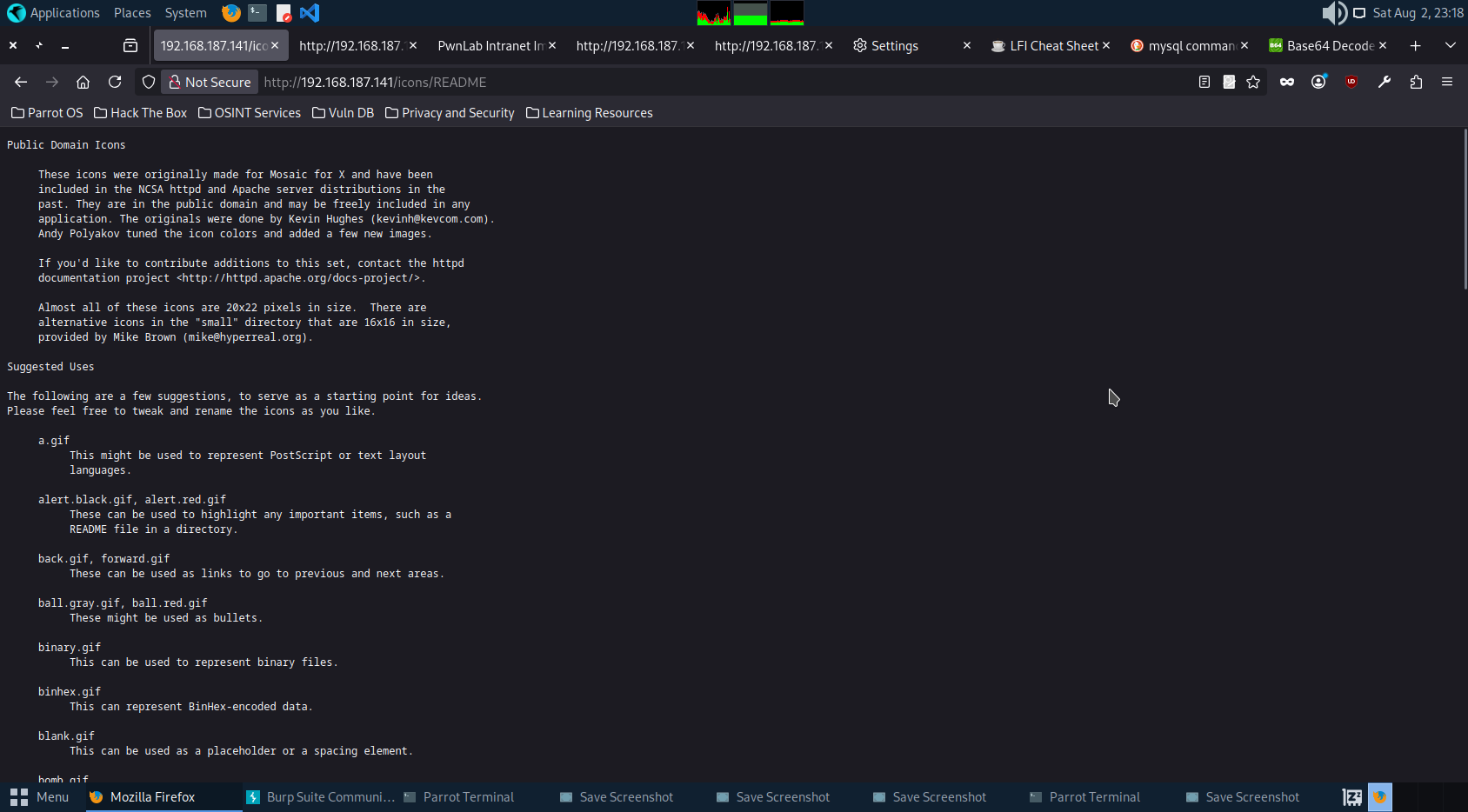


Got some leads like some php files ,README file,etc. Checking if we get some information on this pages.

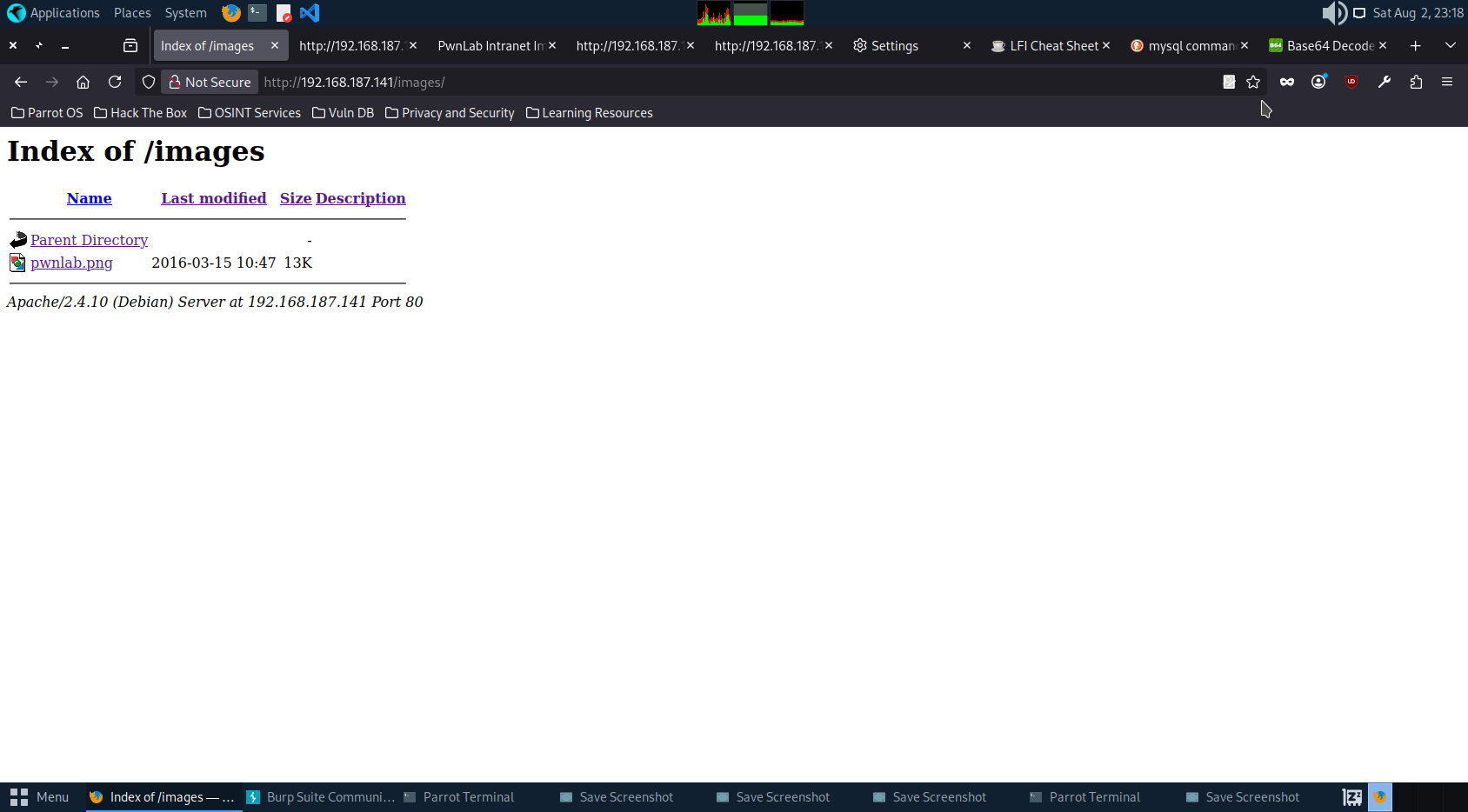


No information on config.php page.

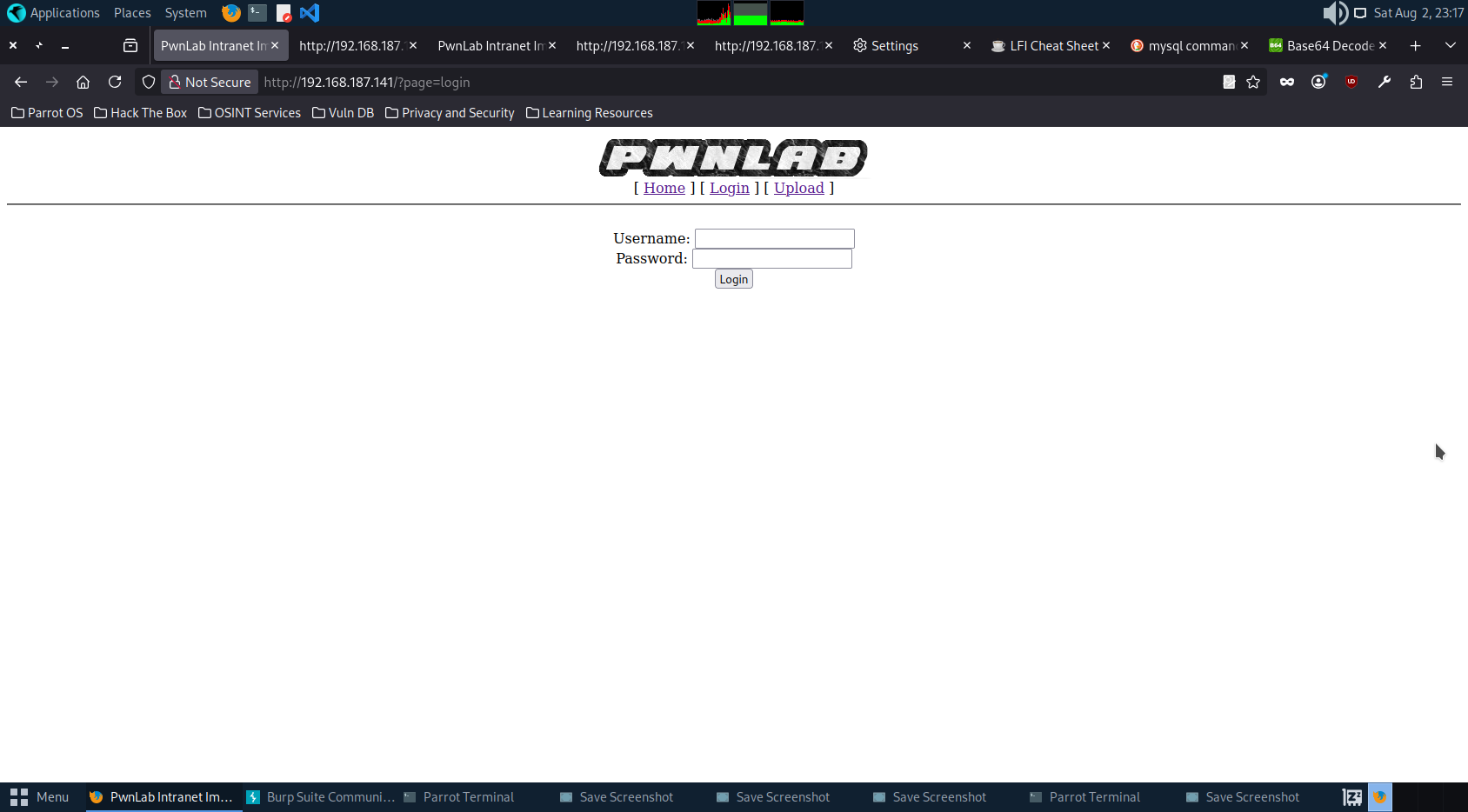
README file contained so many information but nothing was important for our activity.



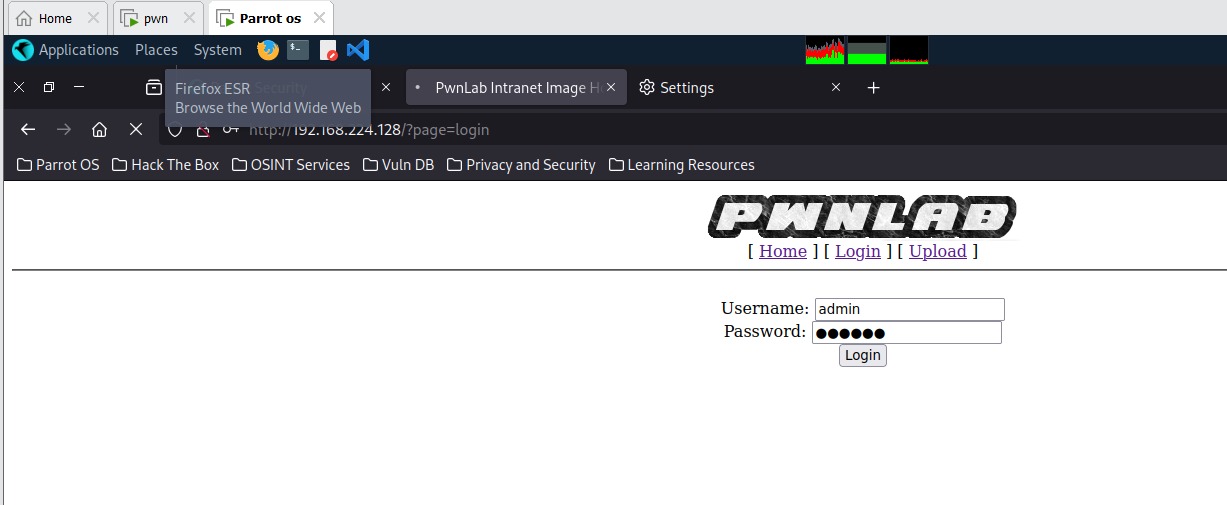
On checking image page we again didn’t got any useful information.



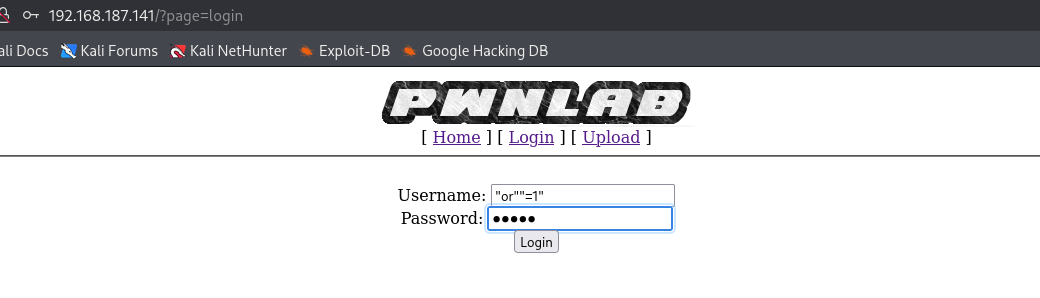
Going to login page.

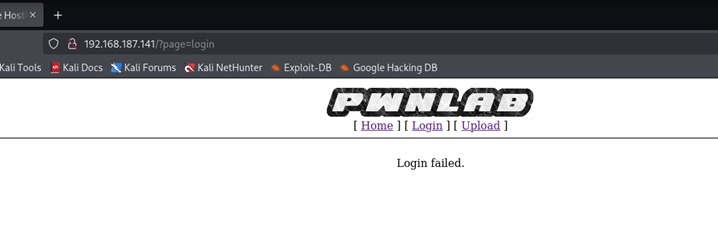


Checking for random input value and was not able to login.

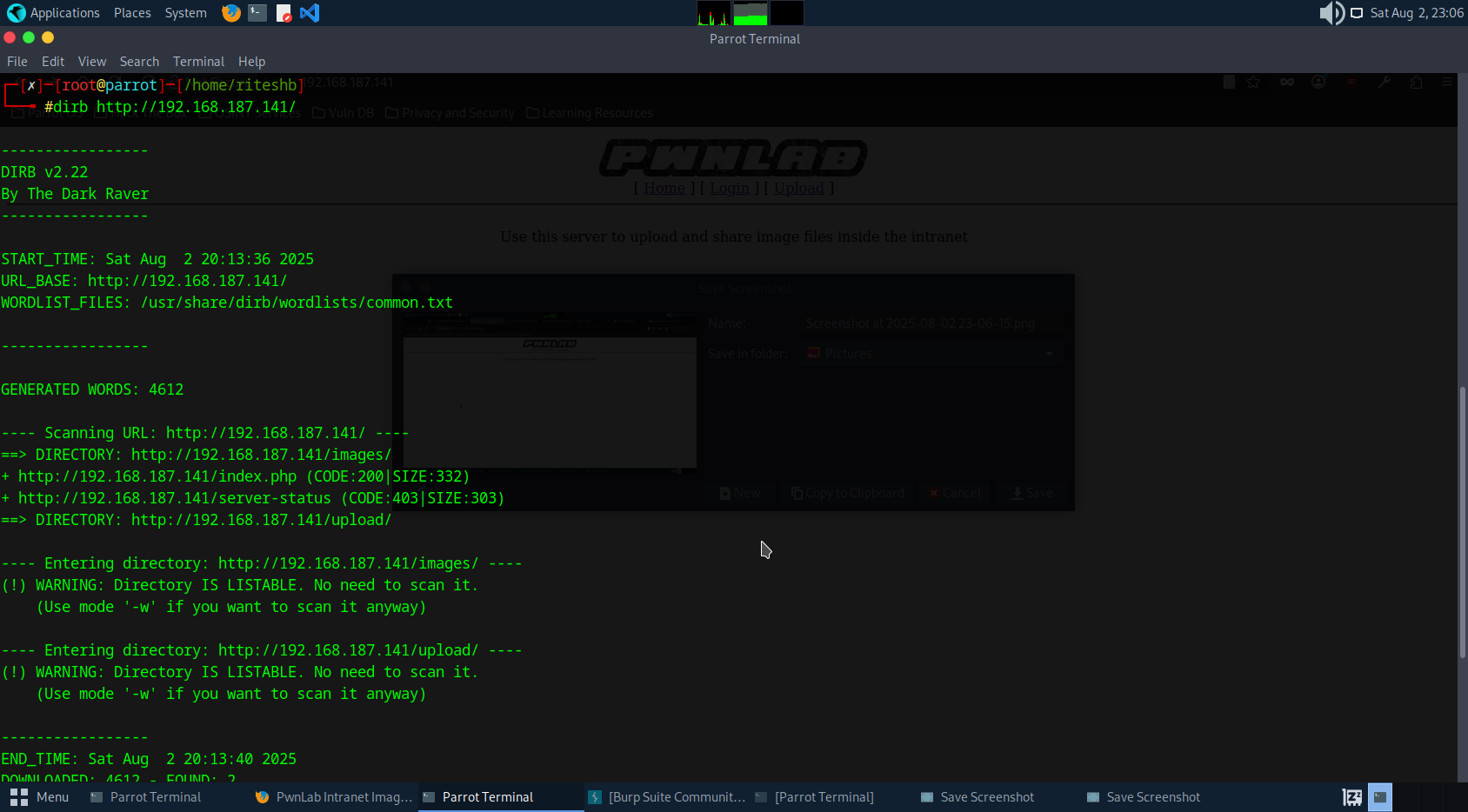


Checking for sql injection

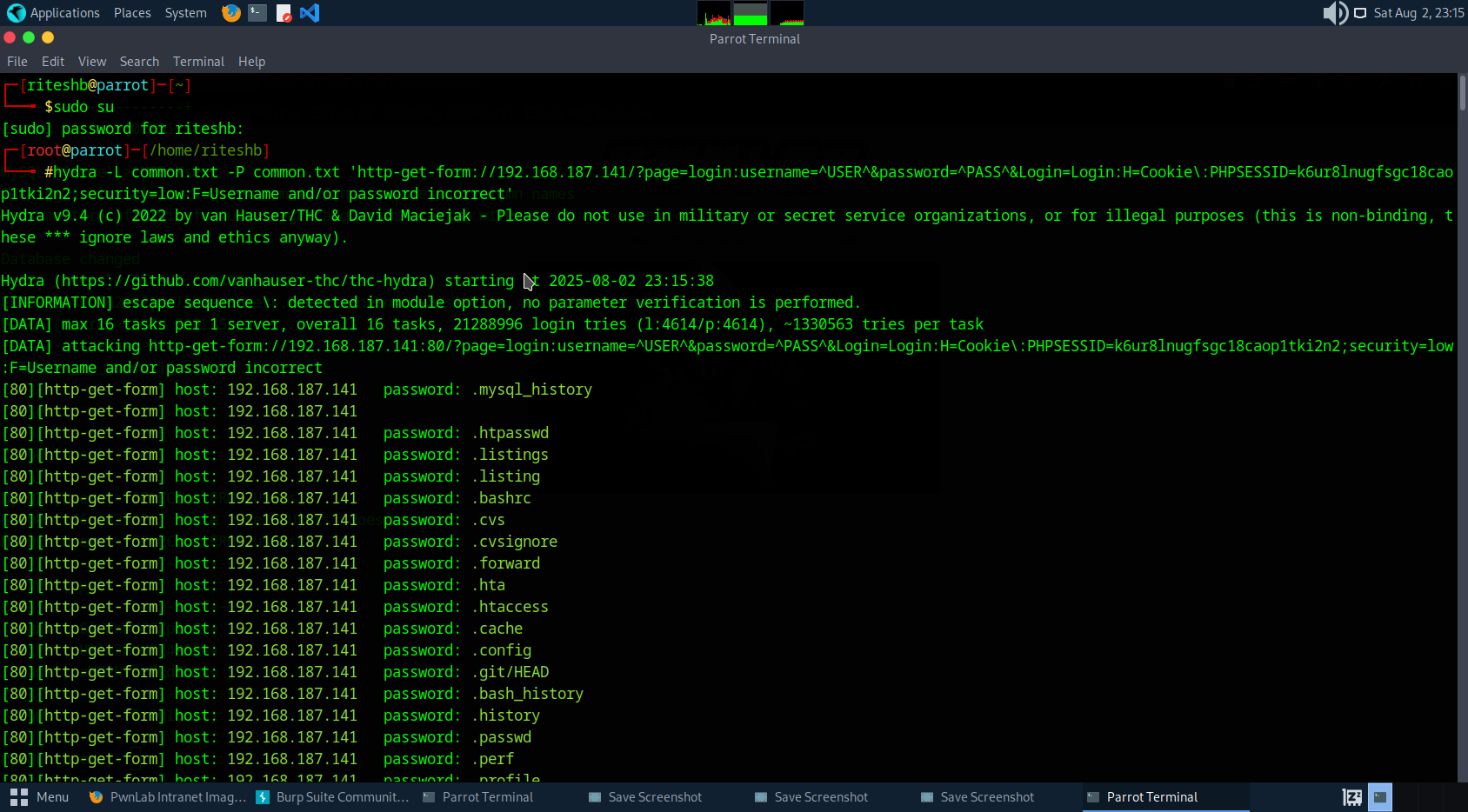




While scanning we got some wordlist.



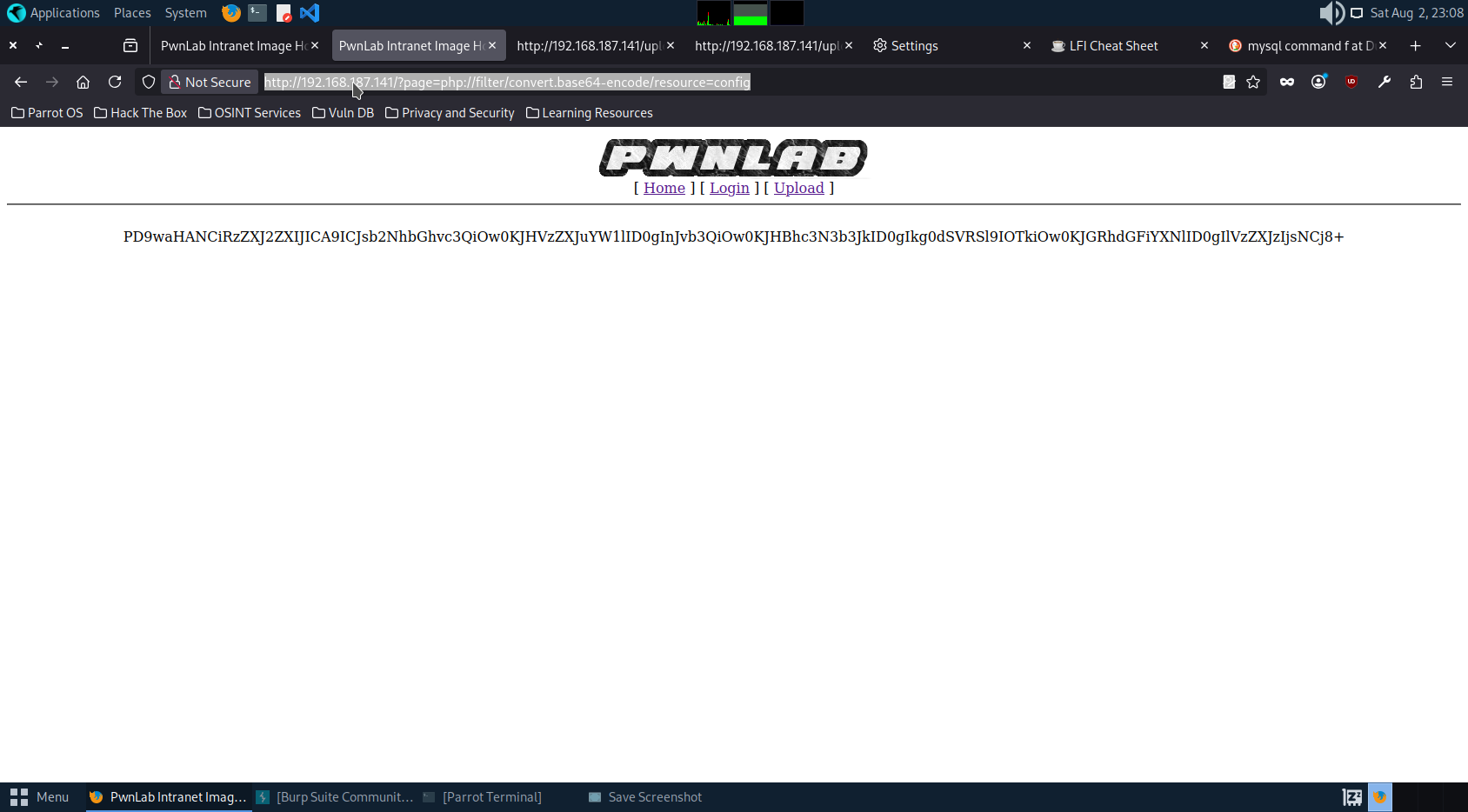
Brute forcing and checking we can get some id password.



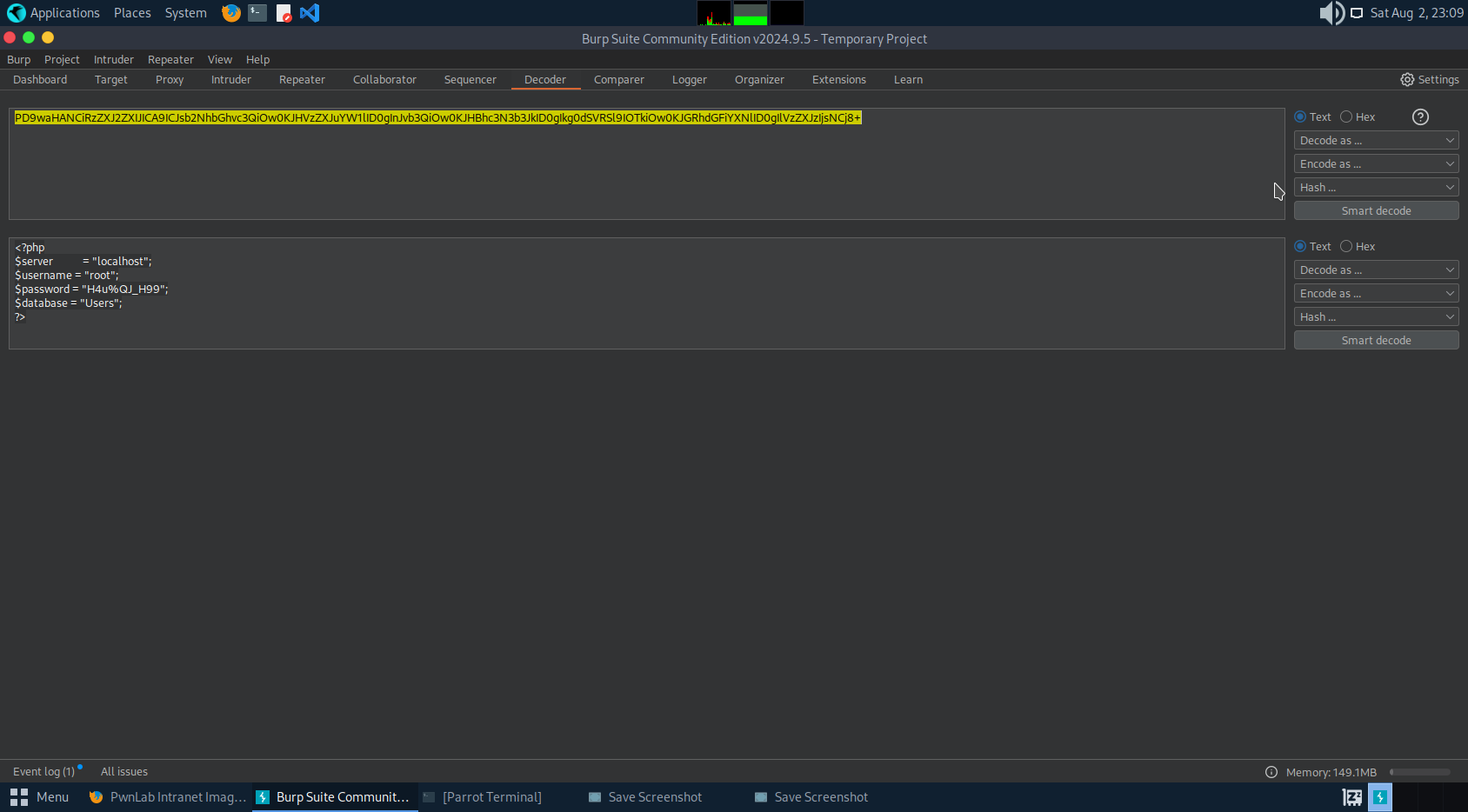
No valid id pass retrieved.

Checking for any LFI vulnerabilties.

We got some kind of hash message while checking for LFI vulnerabilities.

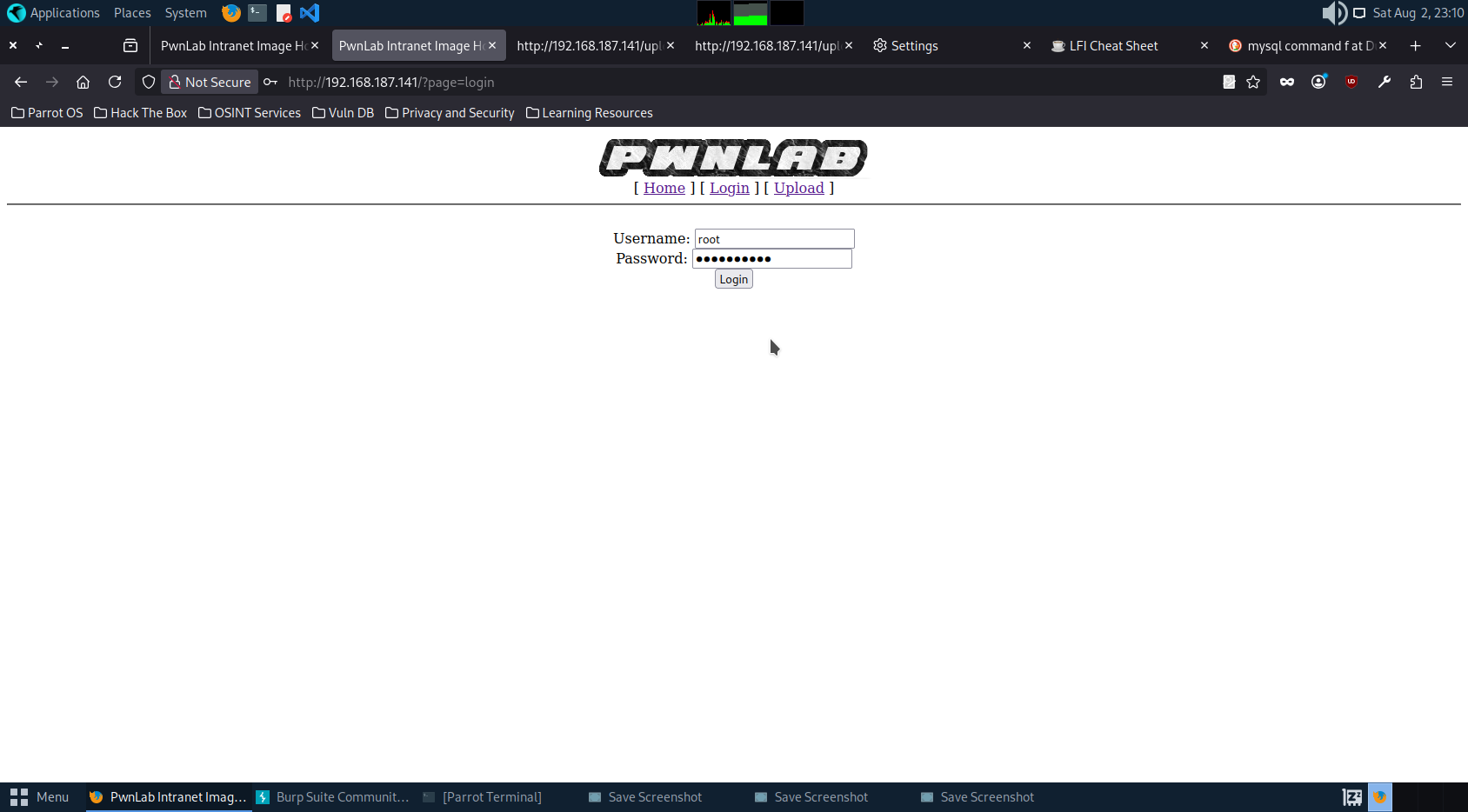


Decoding the message.

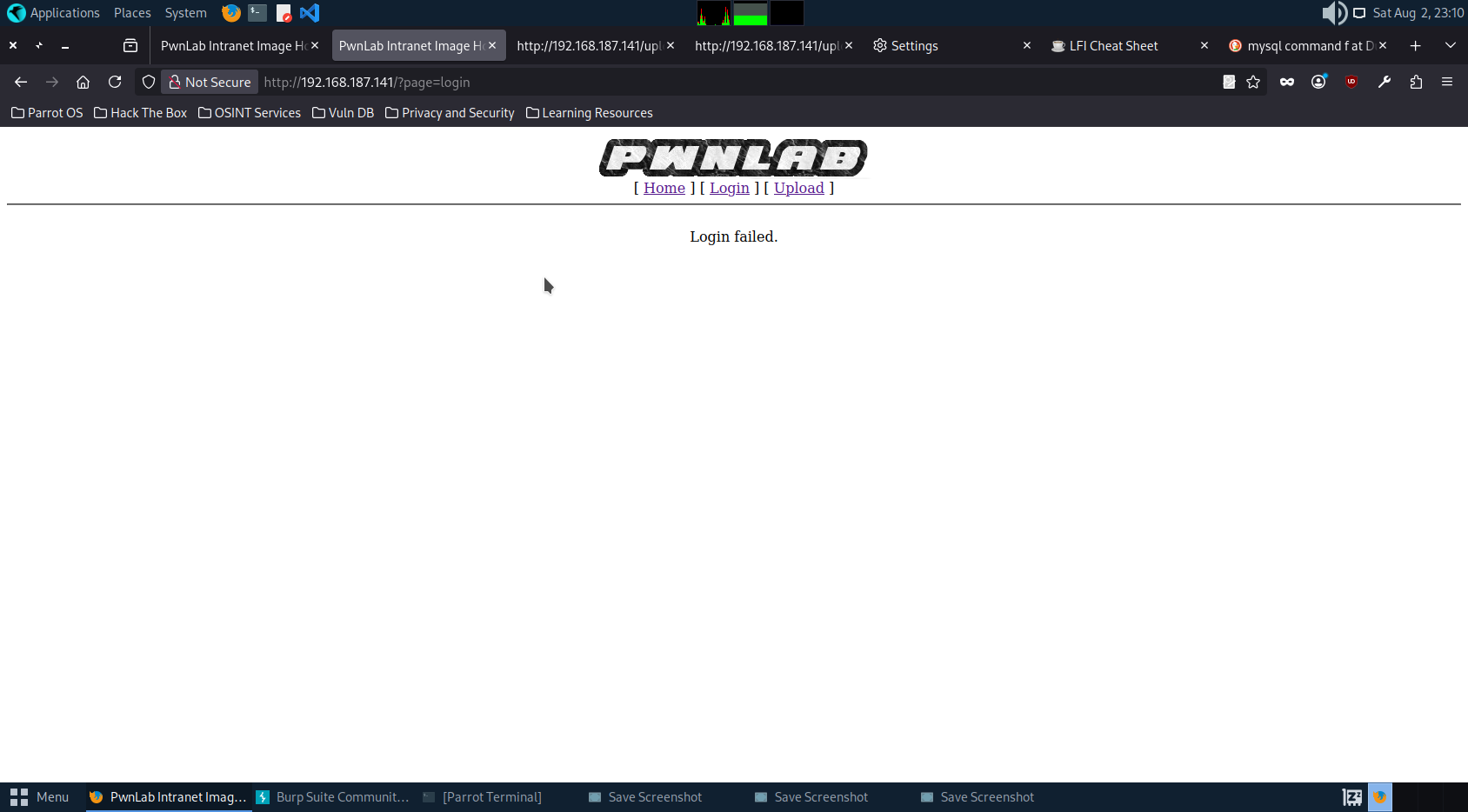


Got some critical credentials.

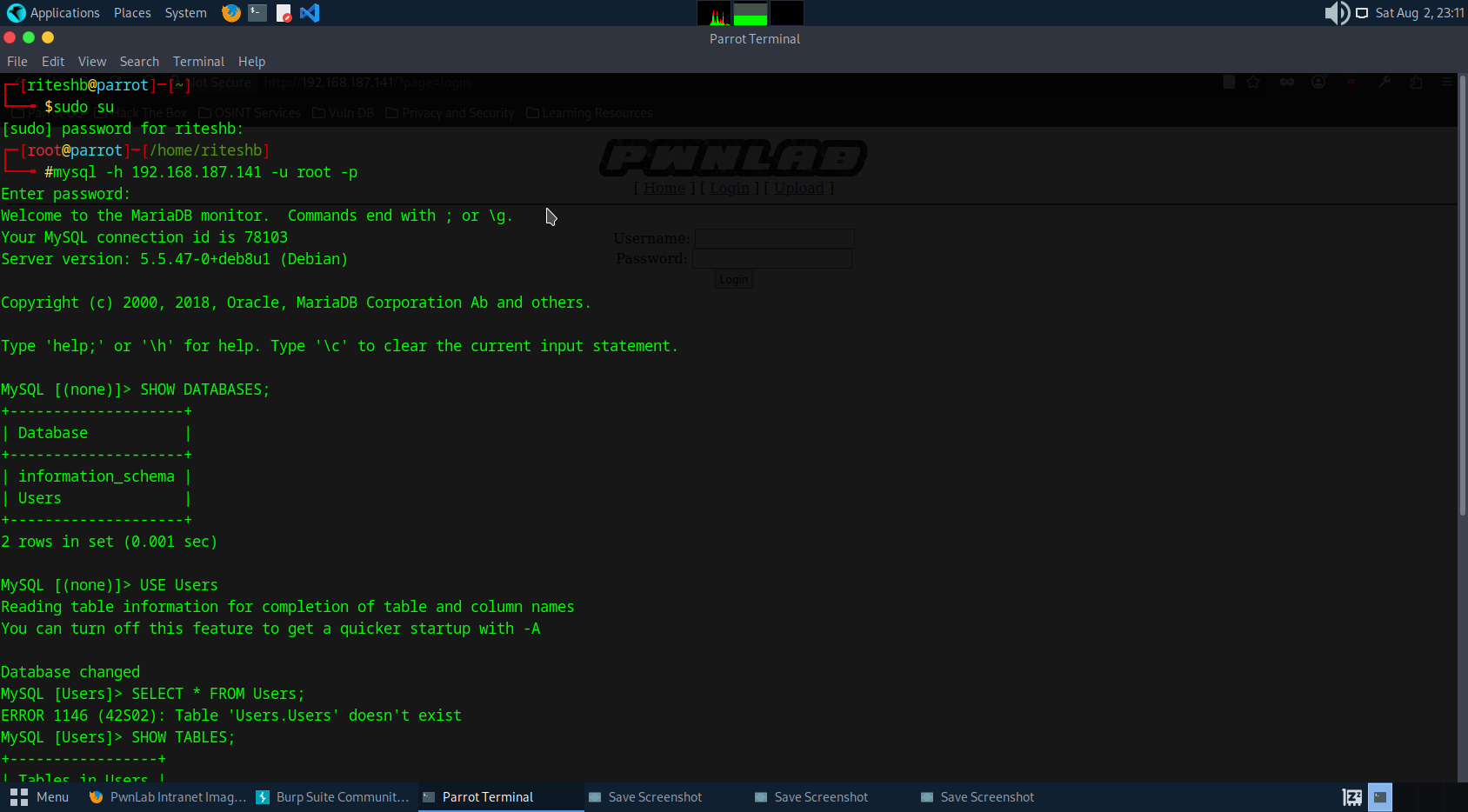
Trying to login.

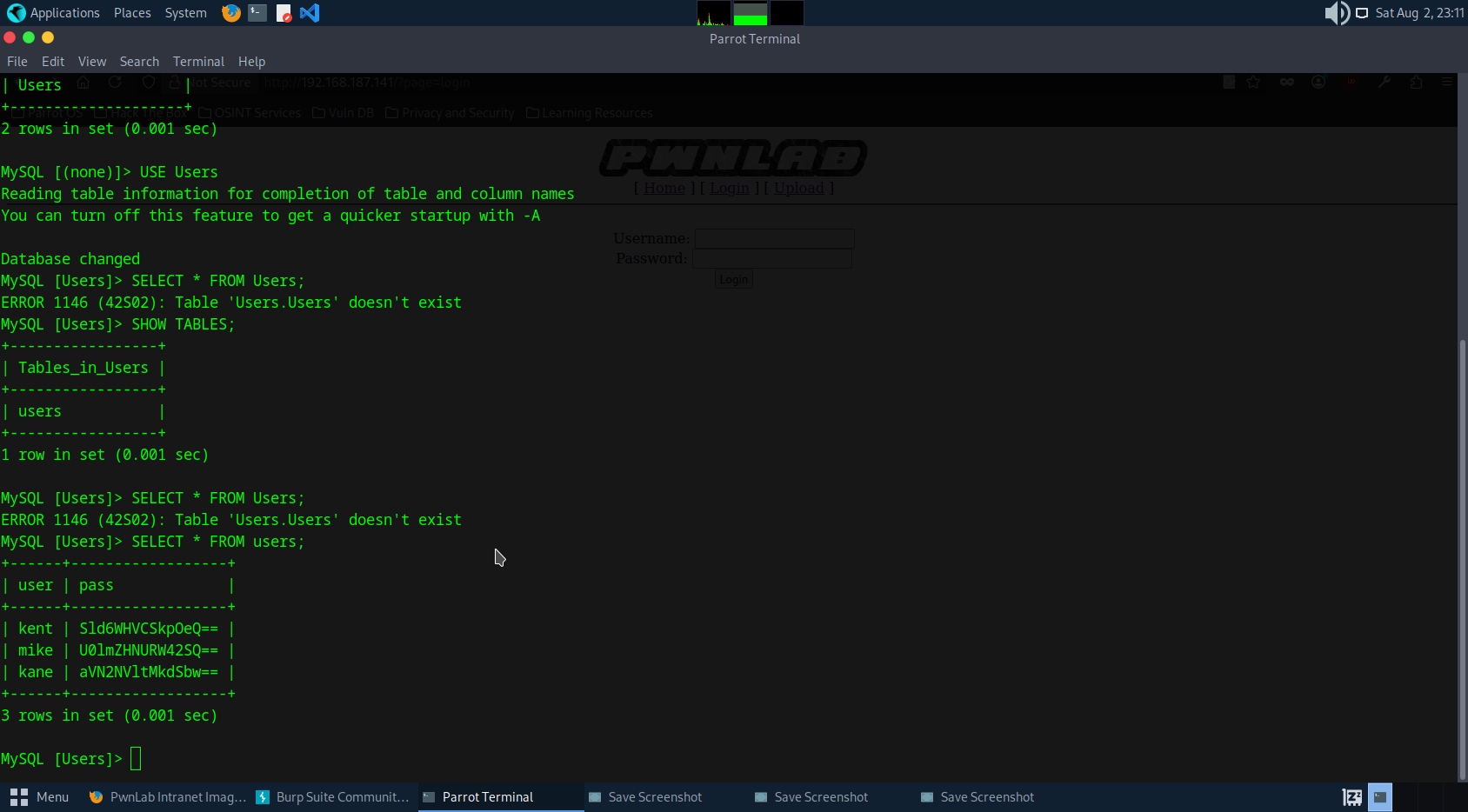


Failed to login again.



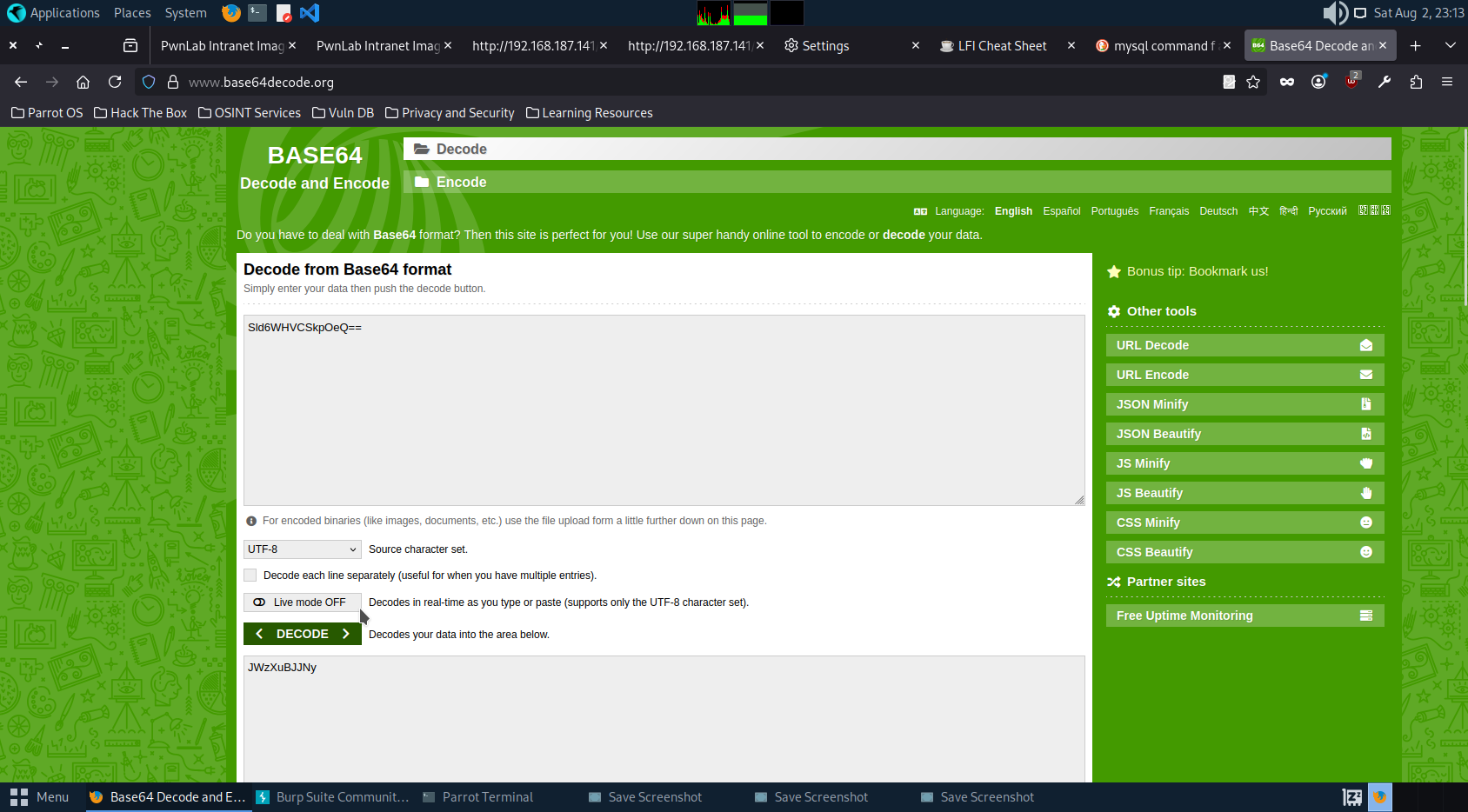
Trying to retrieve the databse.



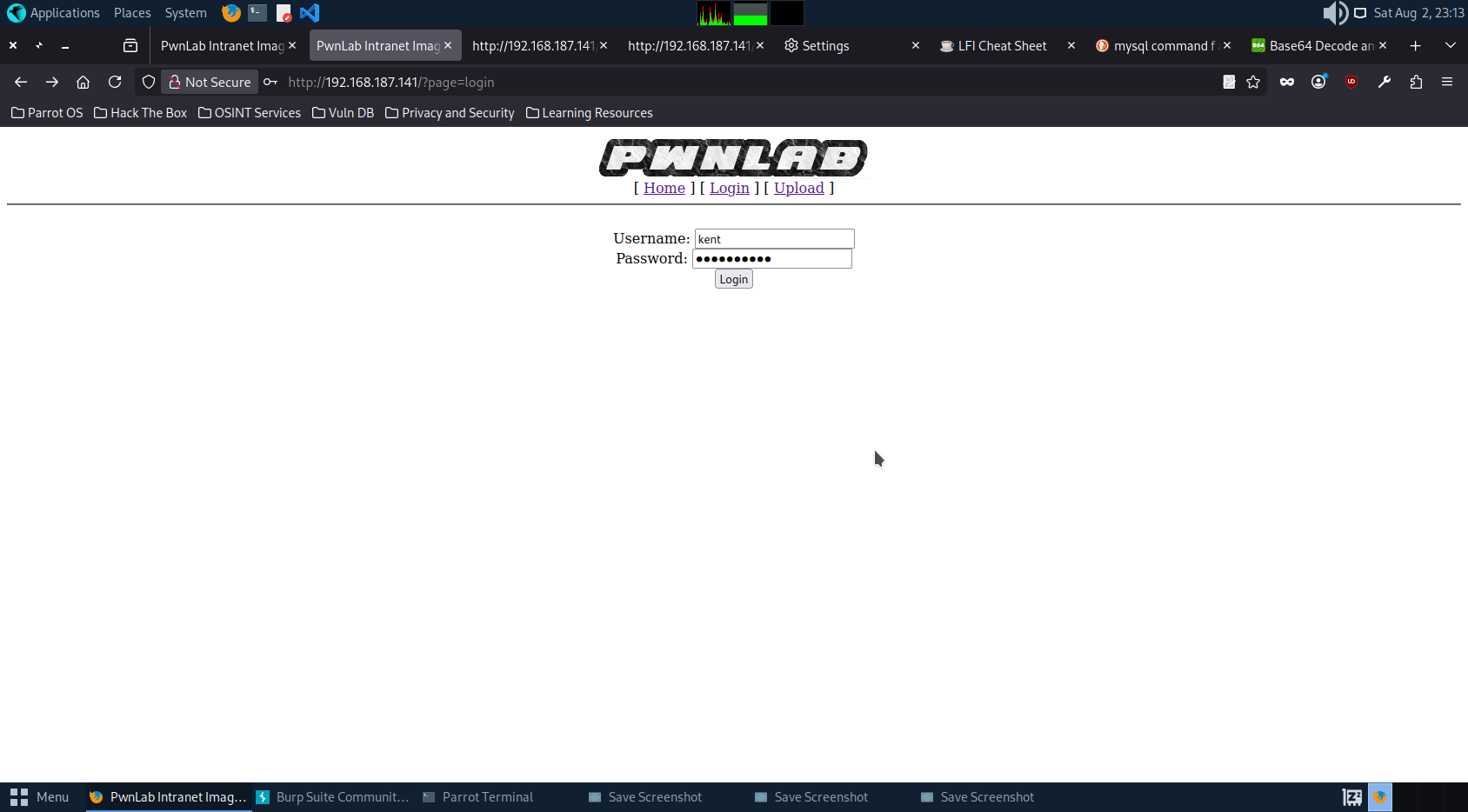


We again got some important credential but seems to be in hash.

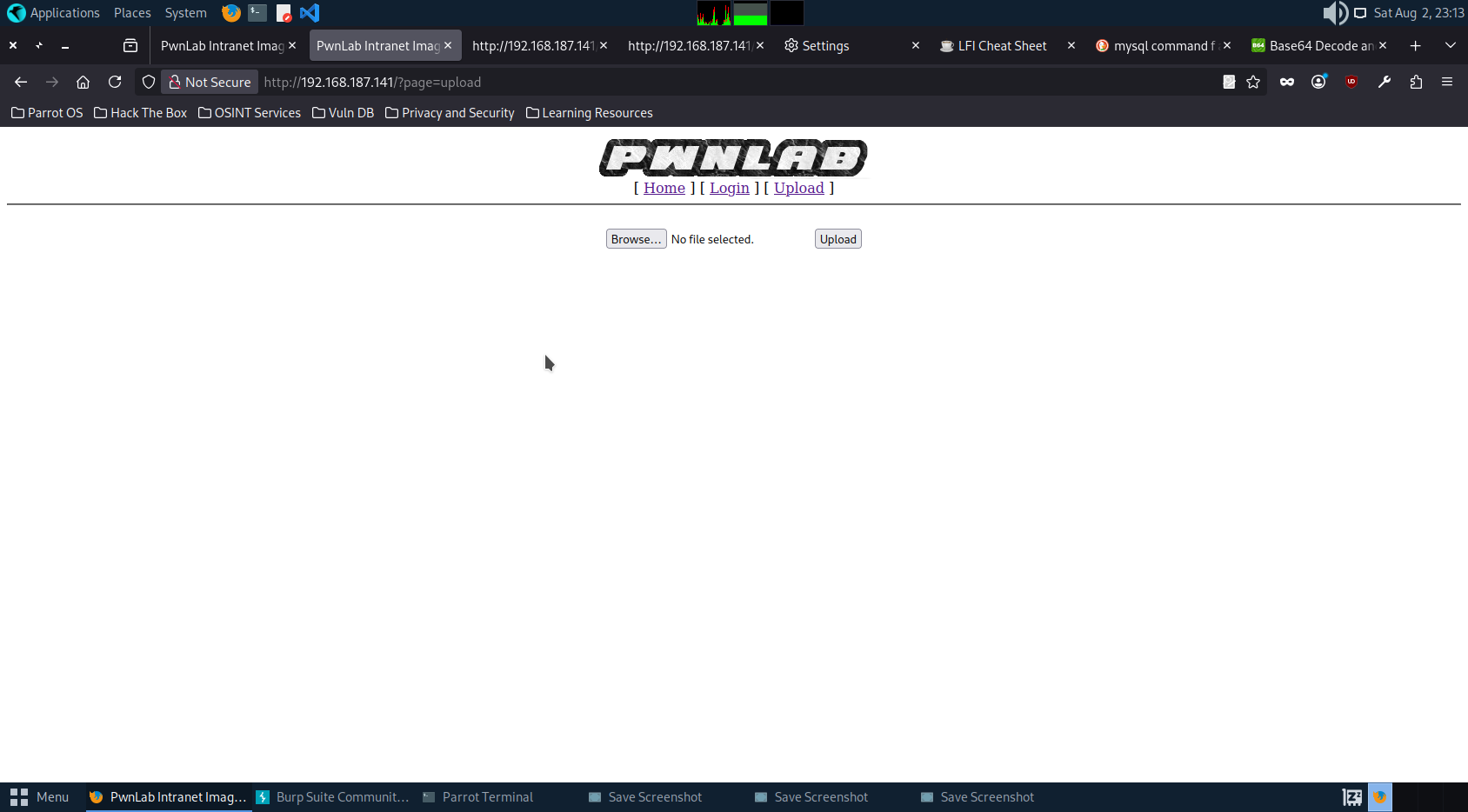
Decoding “kent” user’s password .



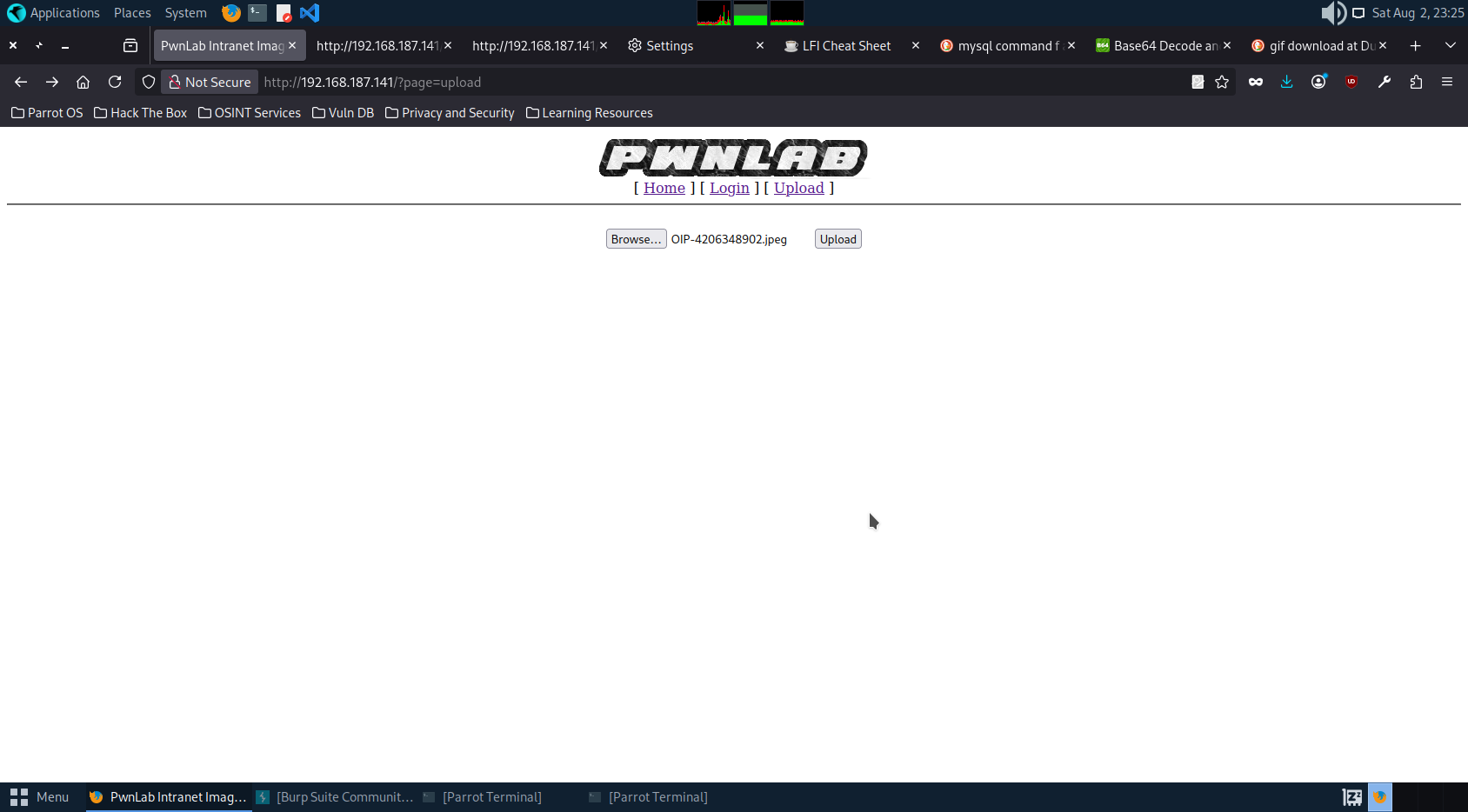
Trying to login with this password now.



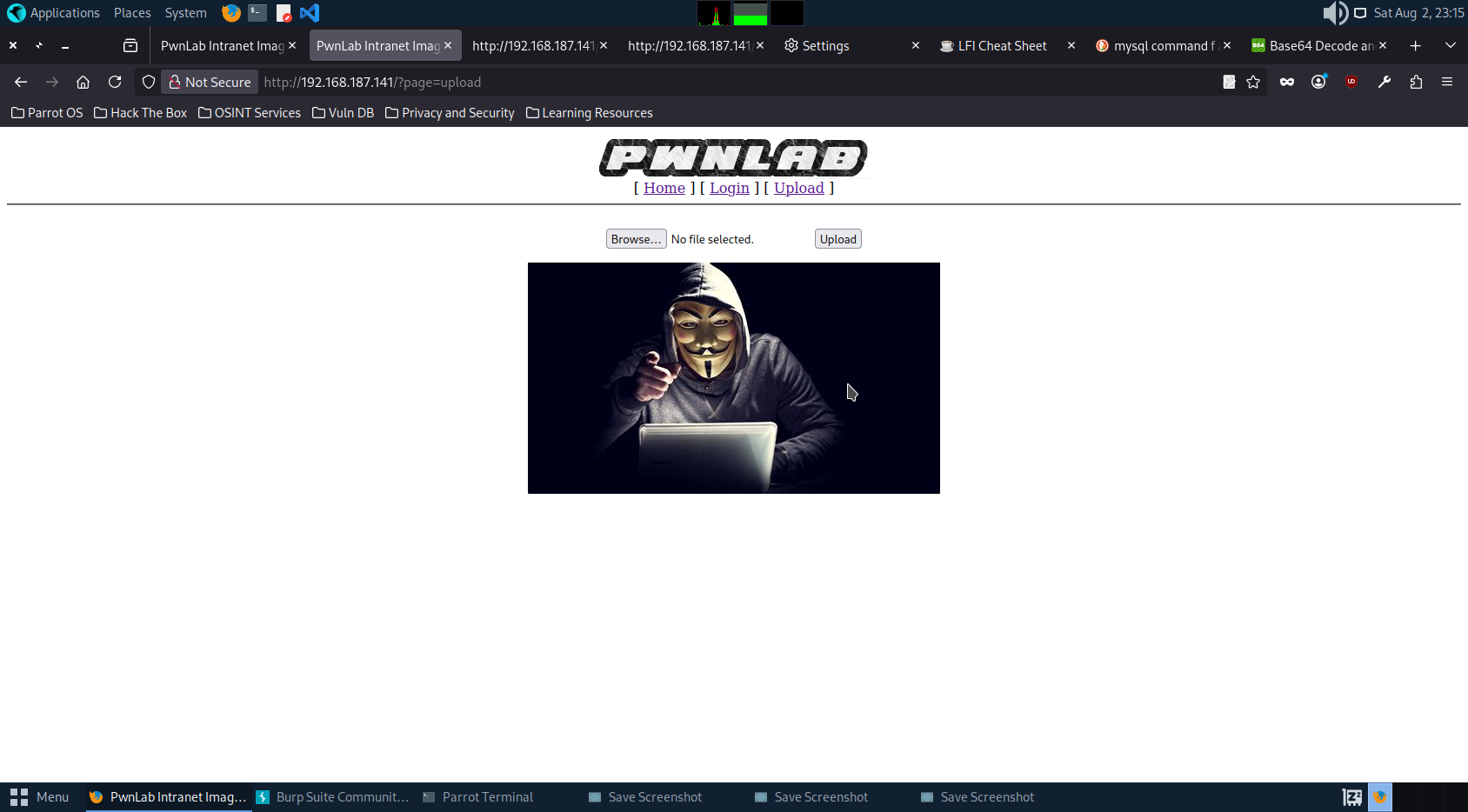
Successfully loged in.



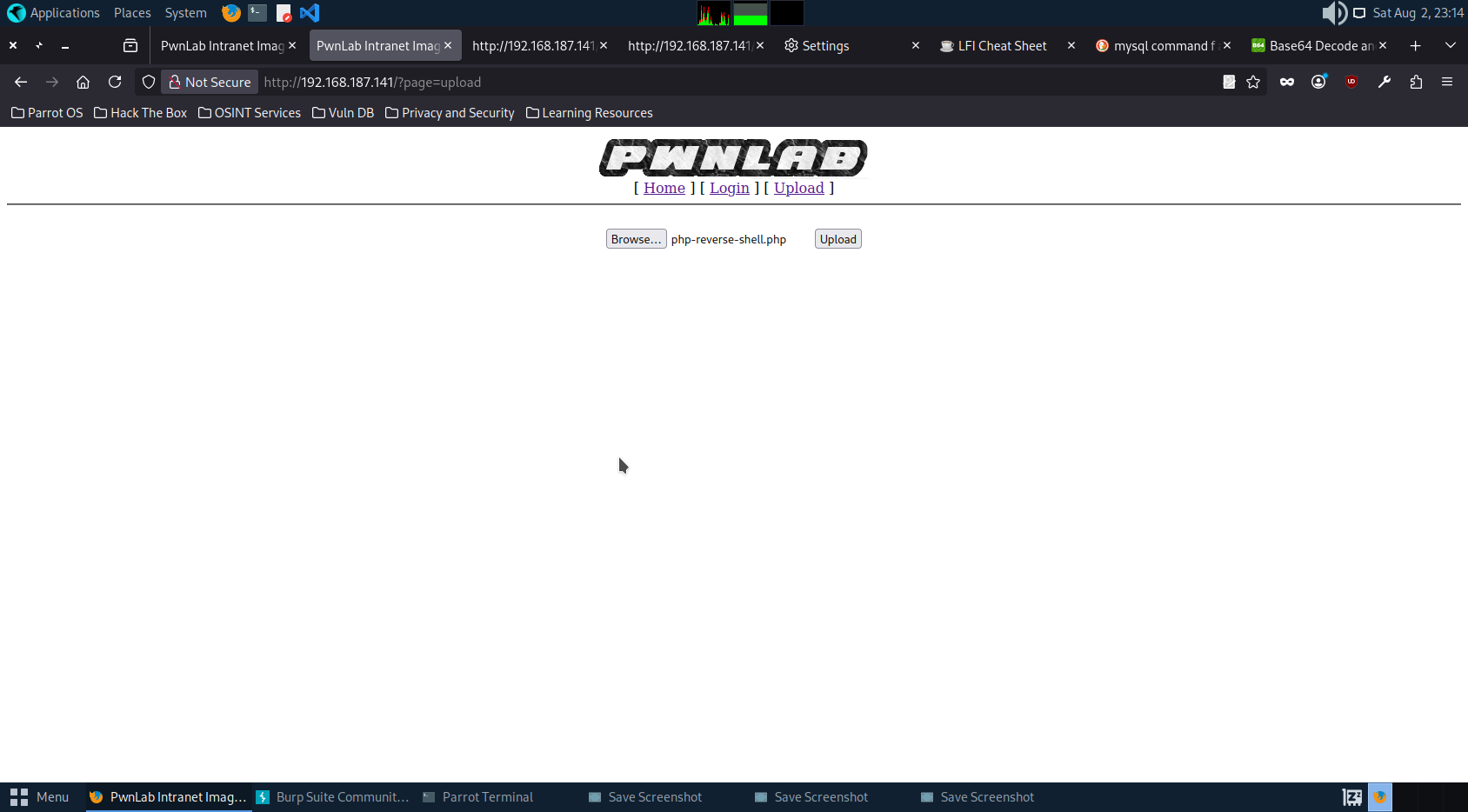
Upload page is there trying to upload some image.



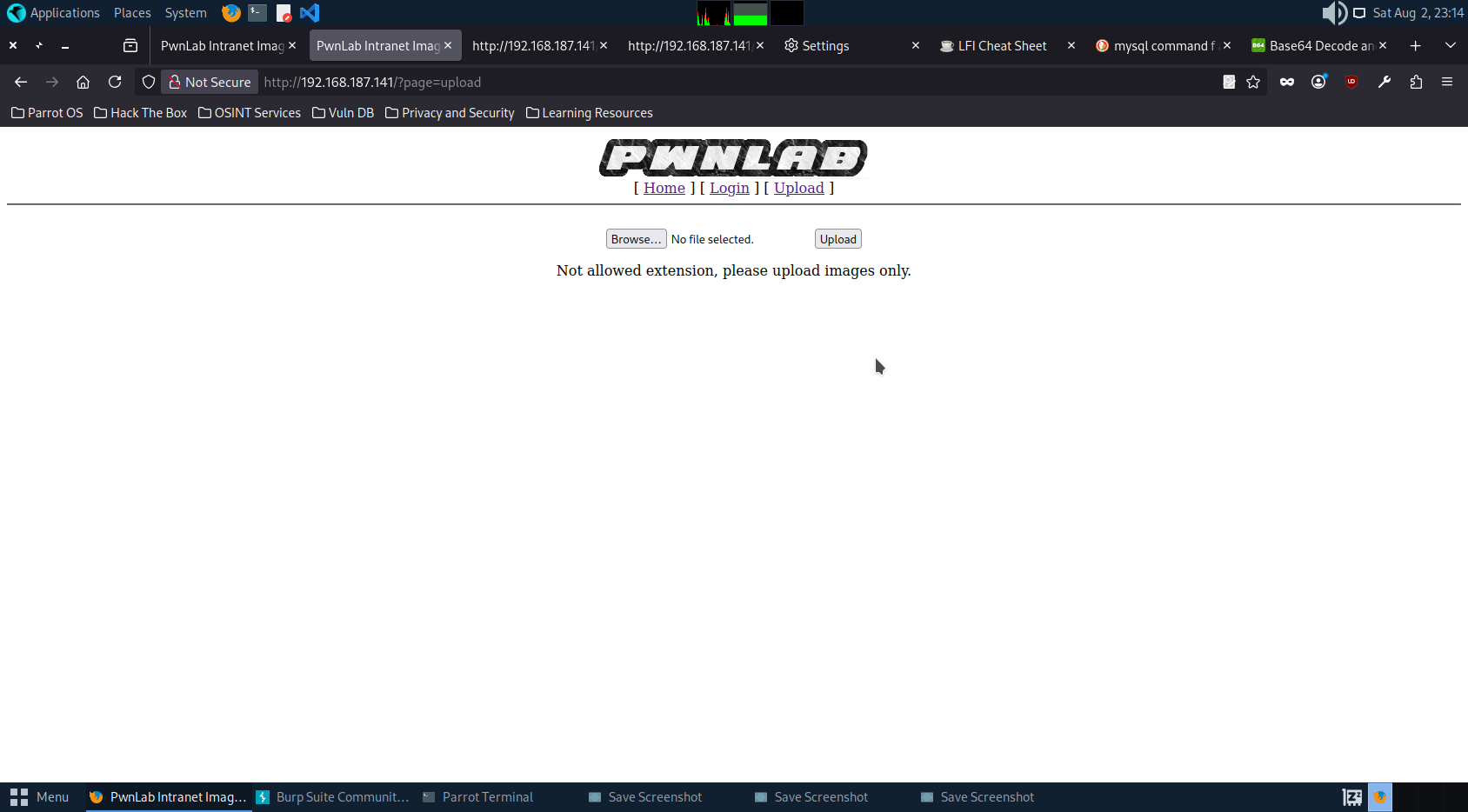
Successfully uploaded the image.



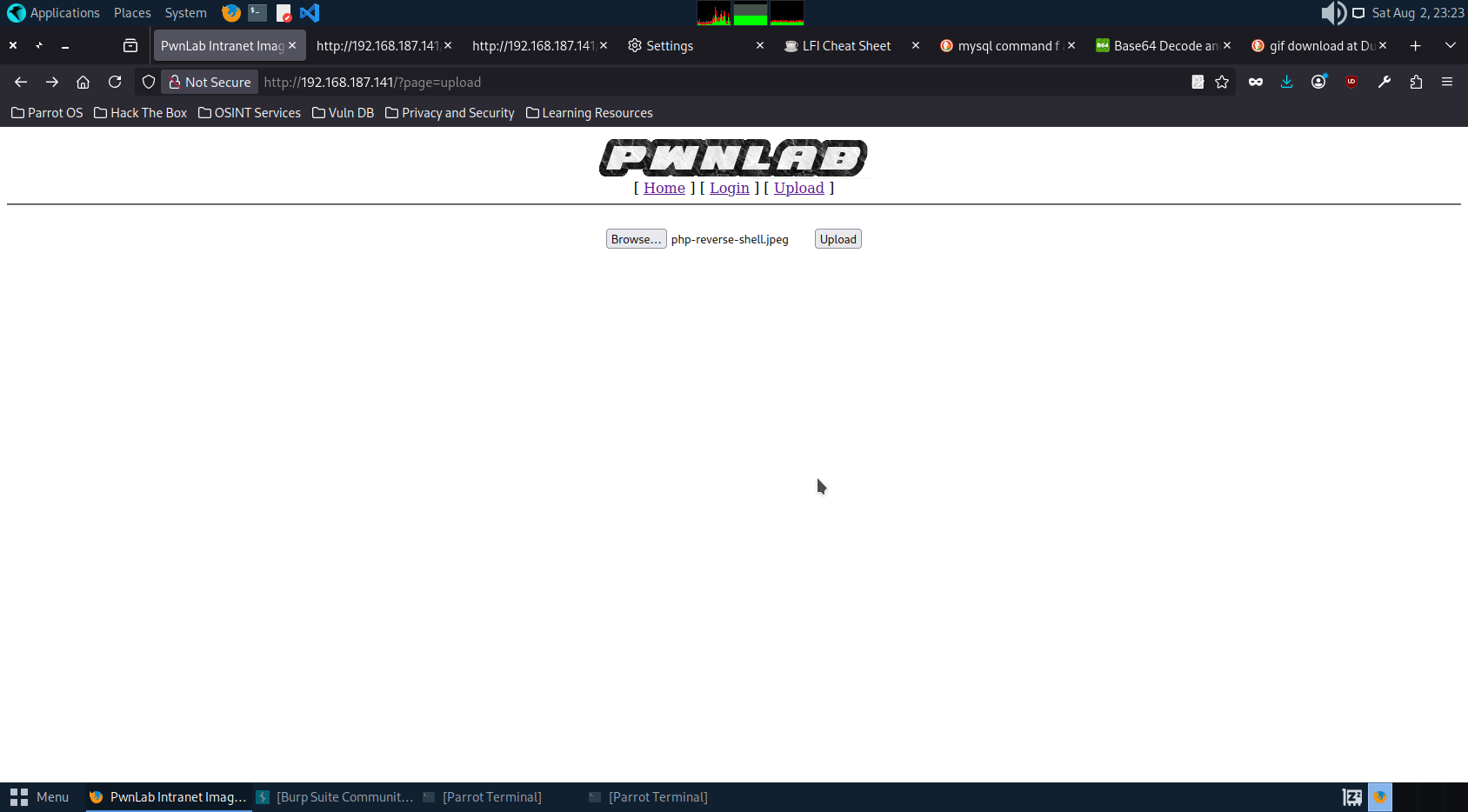
Trying to upload php file so that we can get some privilege.



Site restricted the php file.



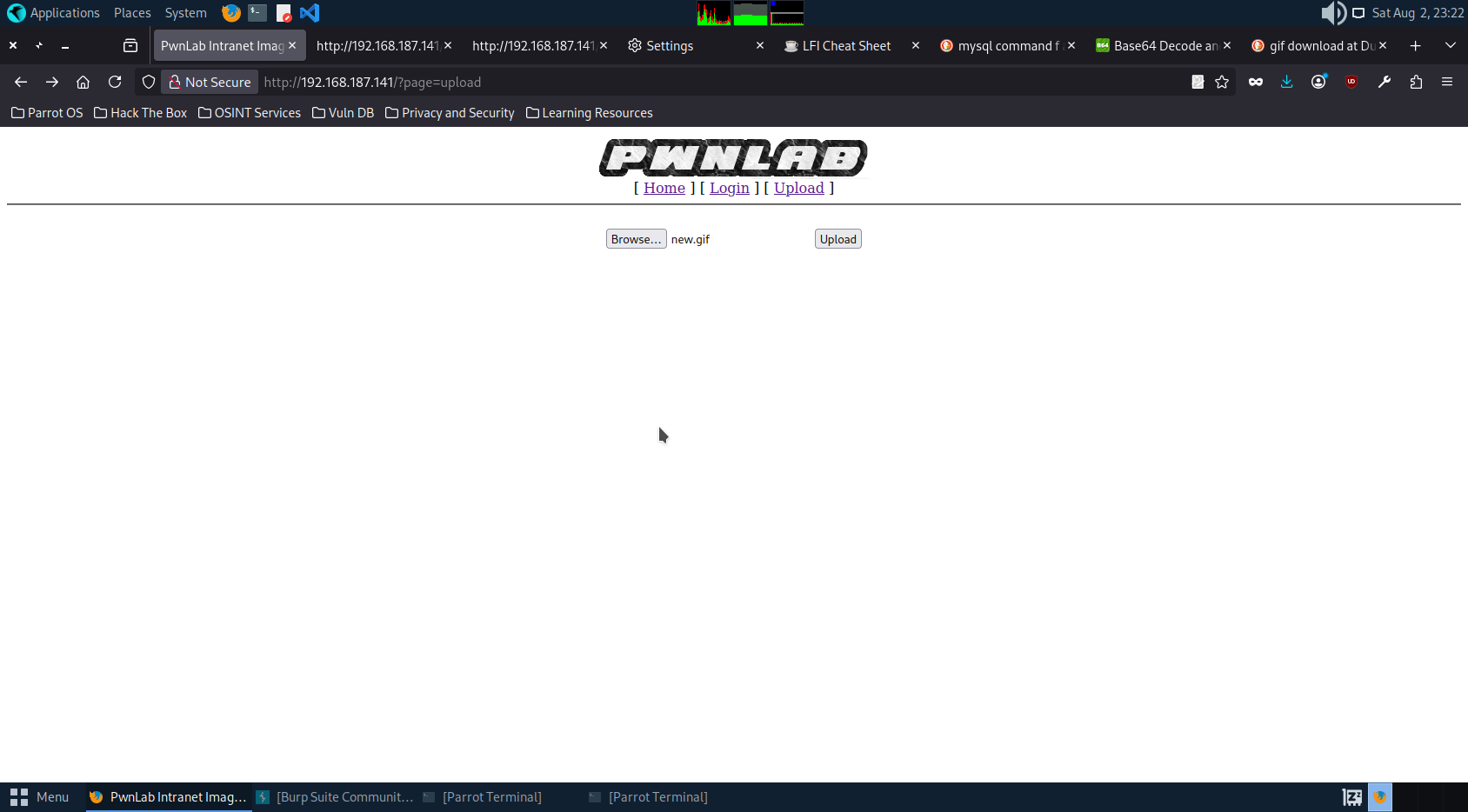
Trying to upload php file with jpeg as the extension.



Didn’t able to upload.



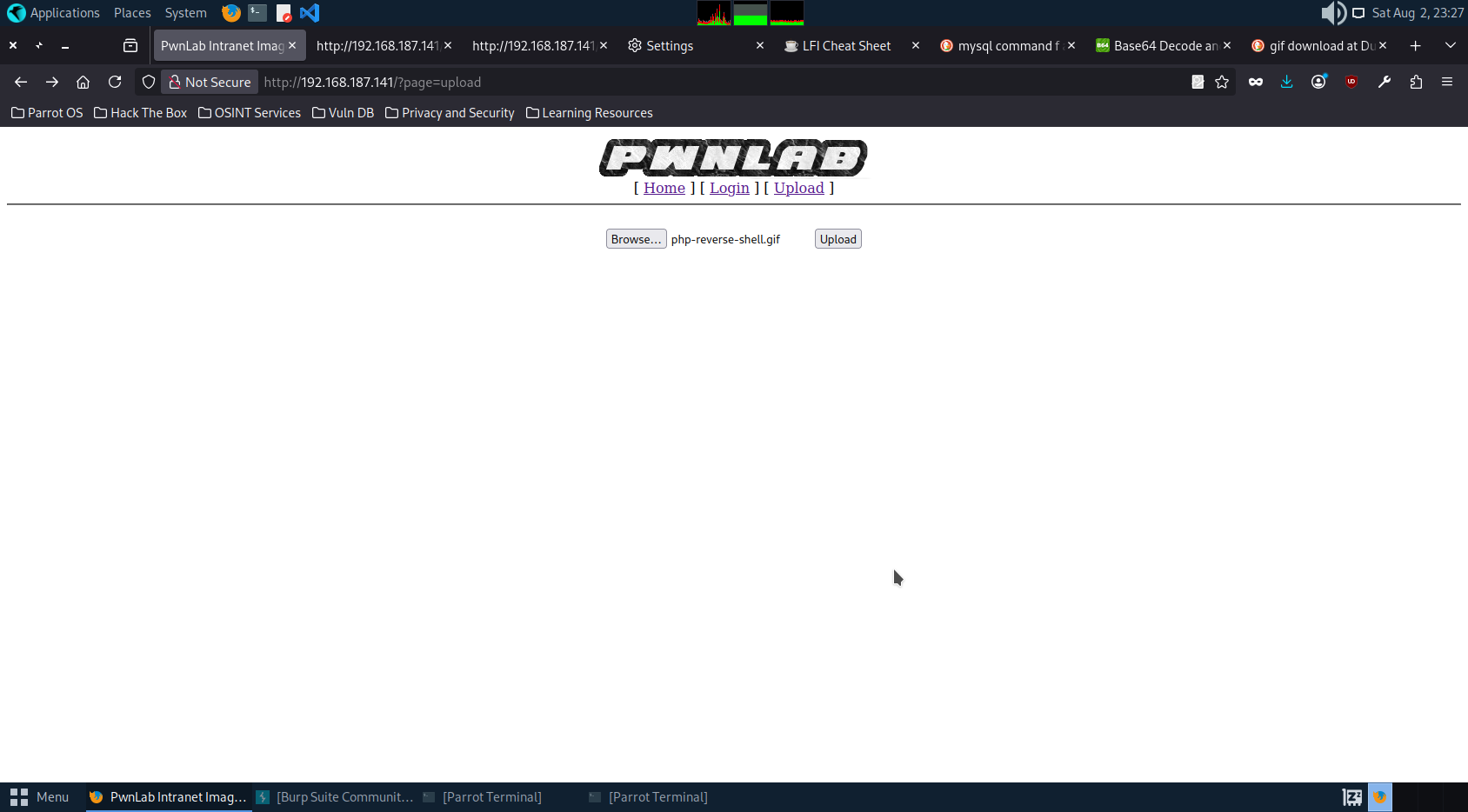
Tryinng to Upload the gif.



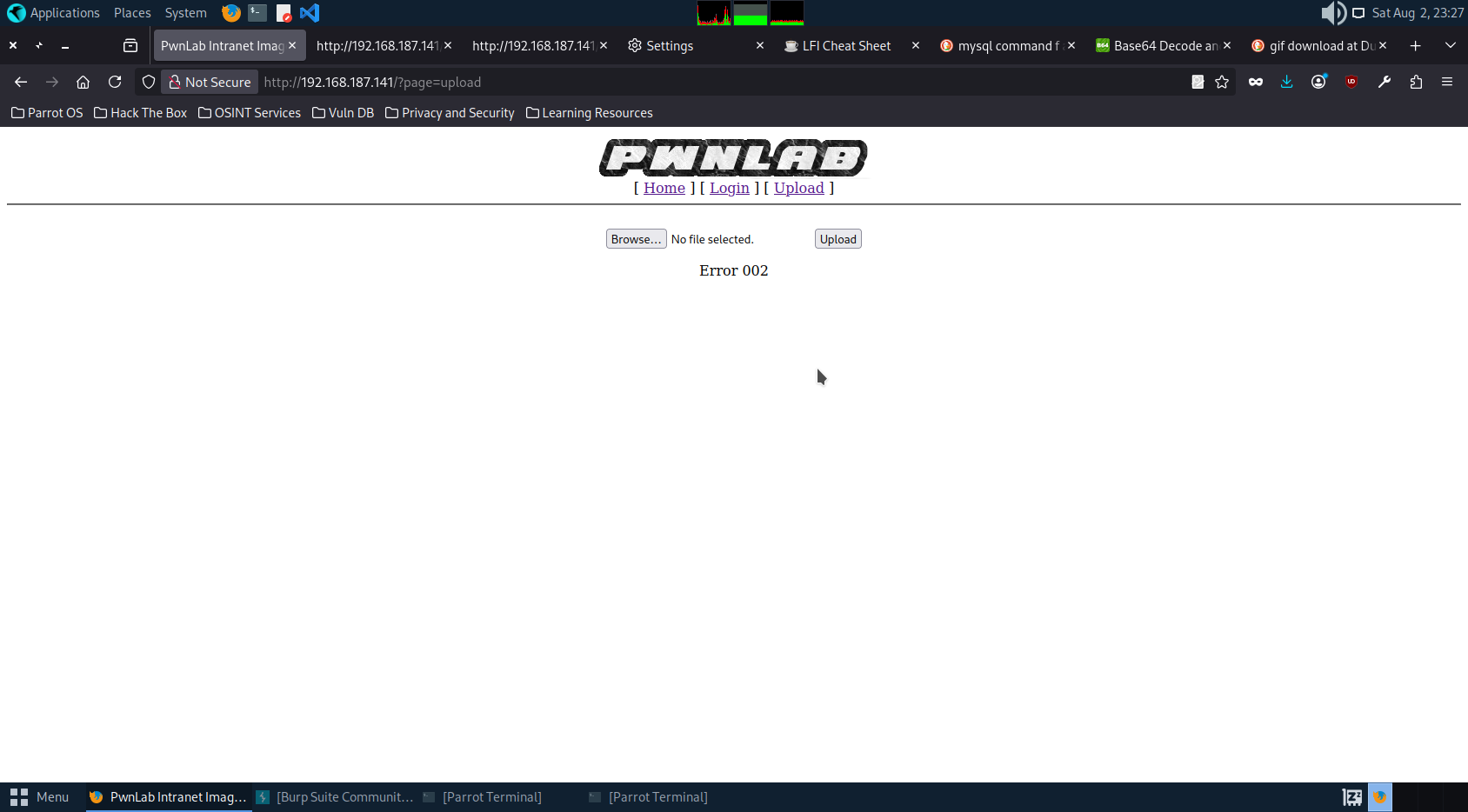
Successfully uploaded.



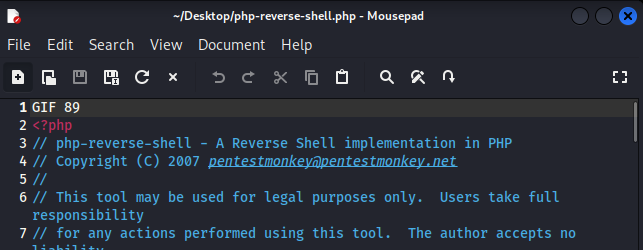
Trying to upload php file with gif as extension.

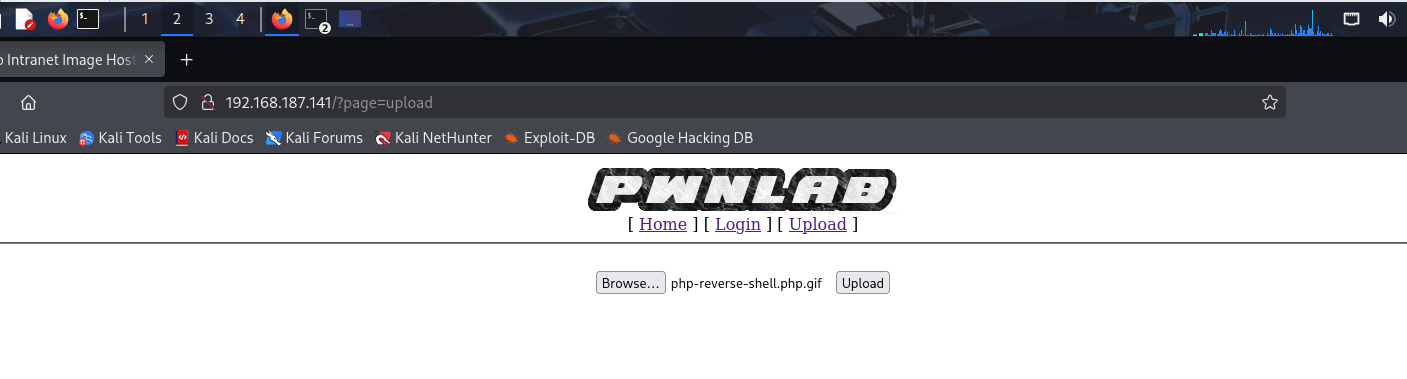


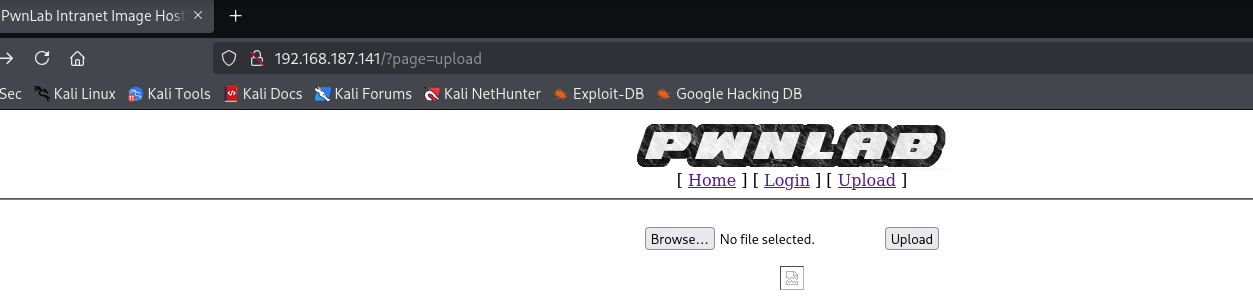
Same error and failed to upload.



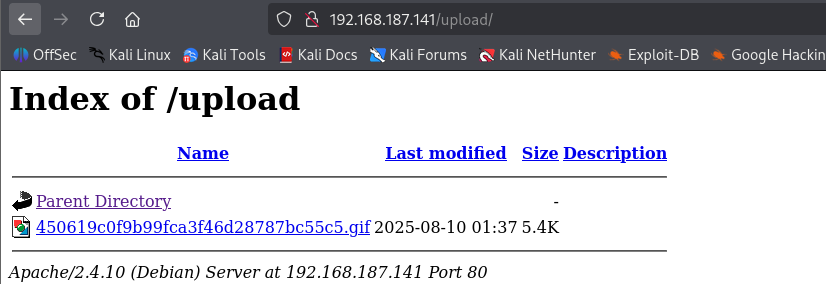
Trying to bypass with new method.



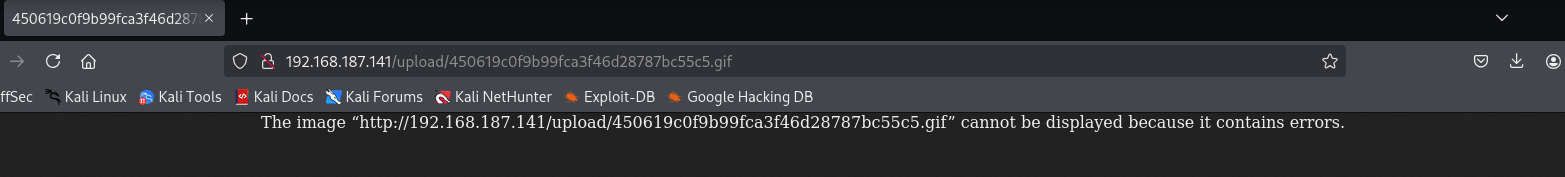




Successfully uploaded the gif file with php code in it.

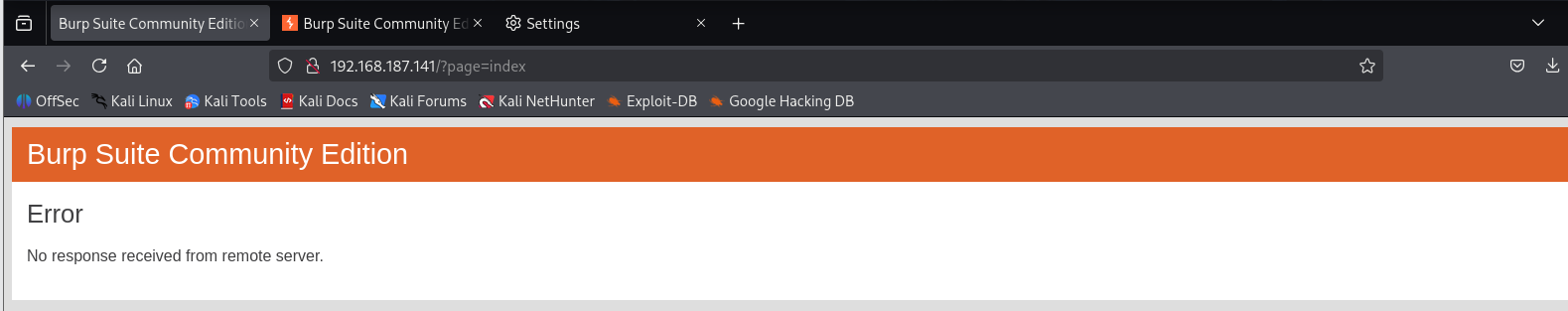


Opening the file to see if we can gain access.

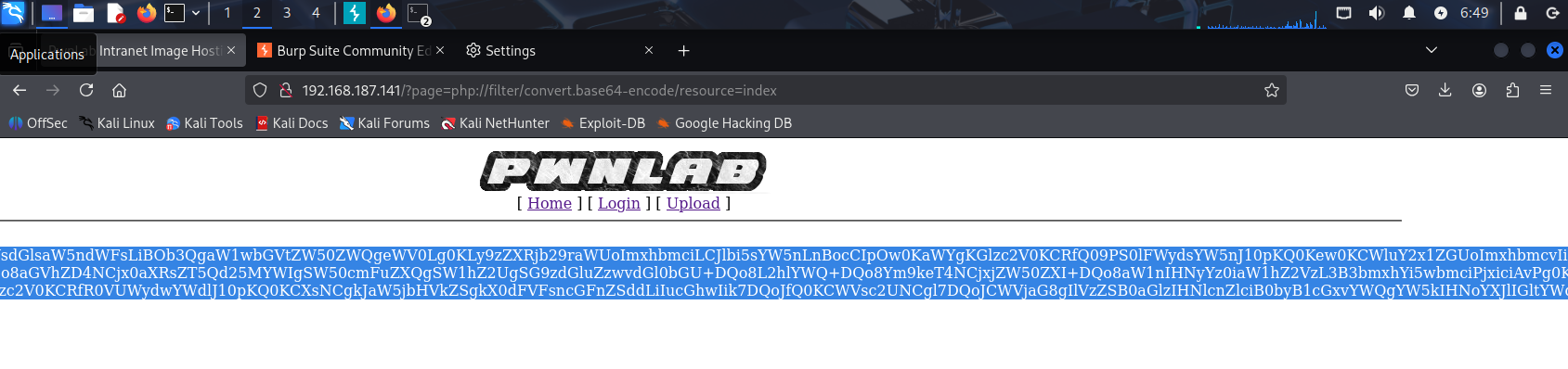


No respond on netcat command.

Trying to open index page.

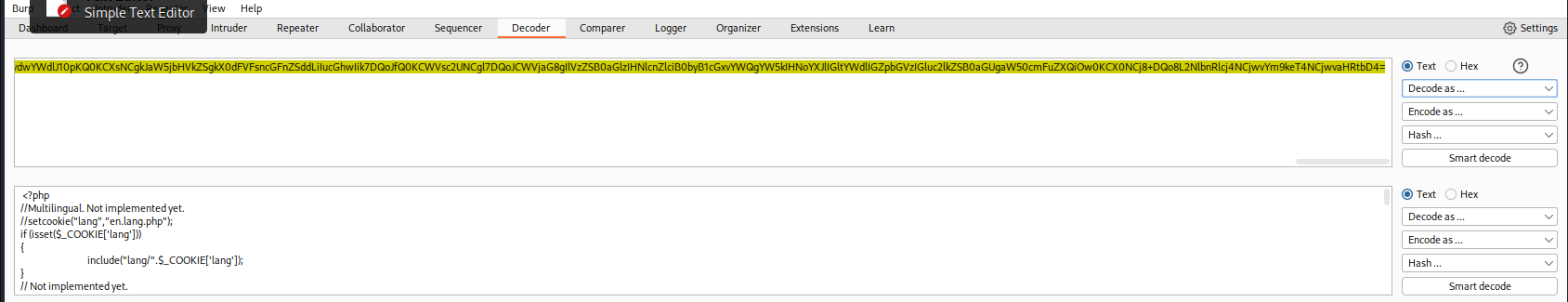


As we know there is a vulnerabilities we are using some command to get index page.



Seems we got some encrypted message.

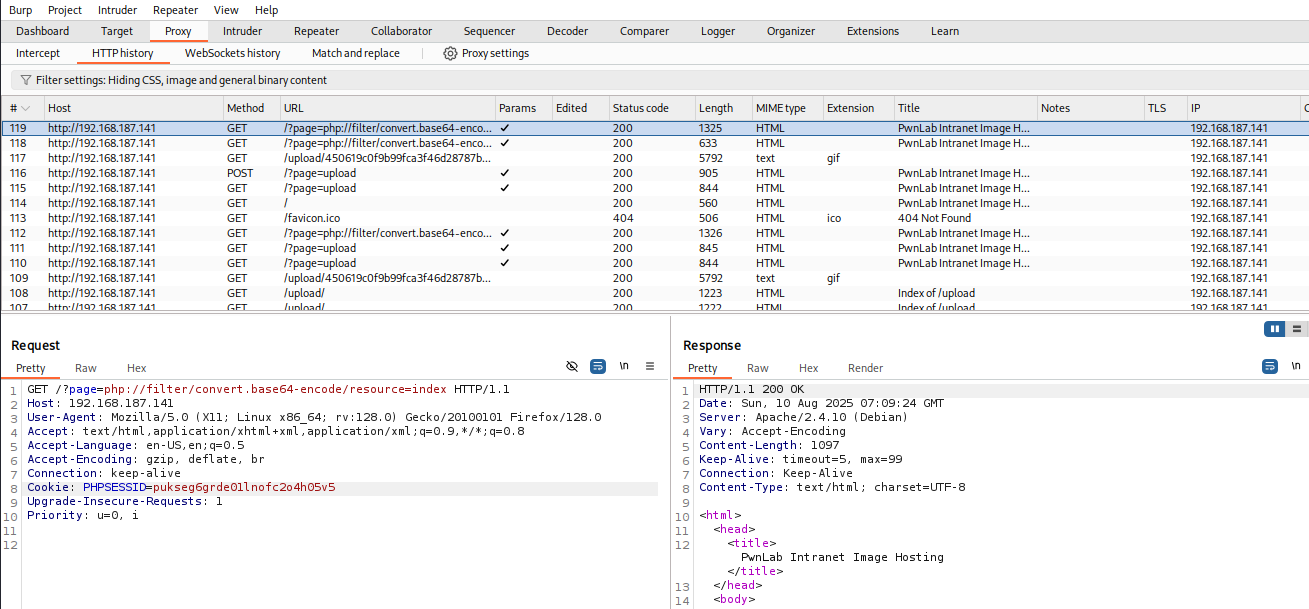
Decrypting.

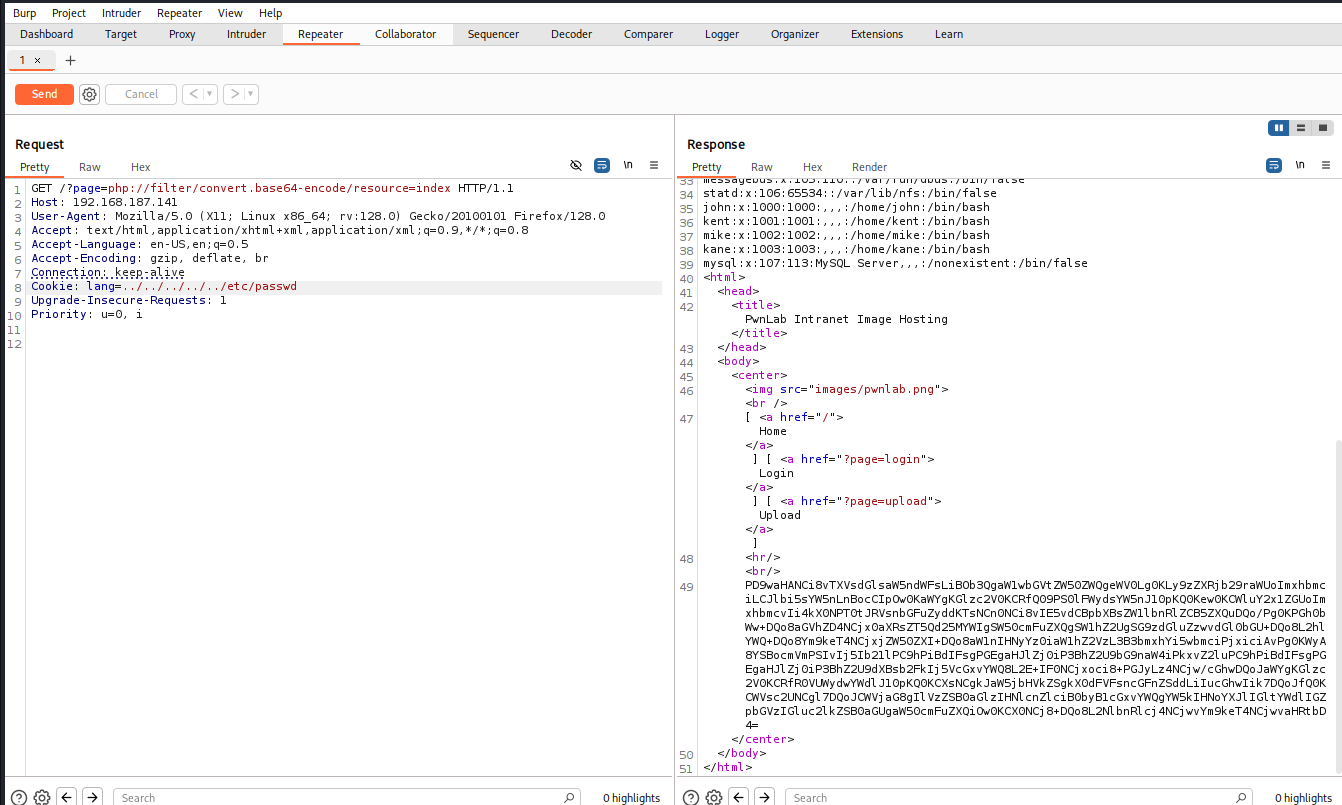


We got lang as Cookie parameter here.

Using lang for further assessment.

From HTTP history in burpsuite we go to the index page request.





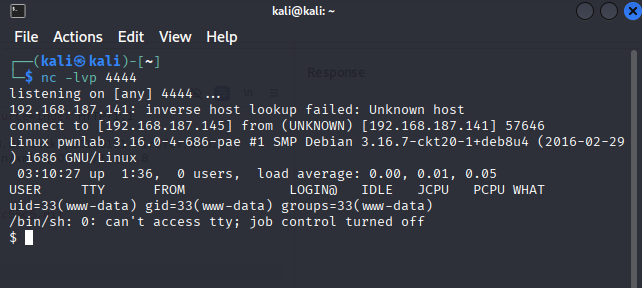
Got the code successfully. Means lang is working porperly.

Trying to run our uploaded file with this lang parameter.



Request is successfully send.

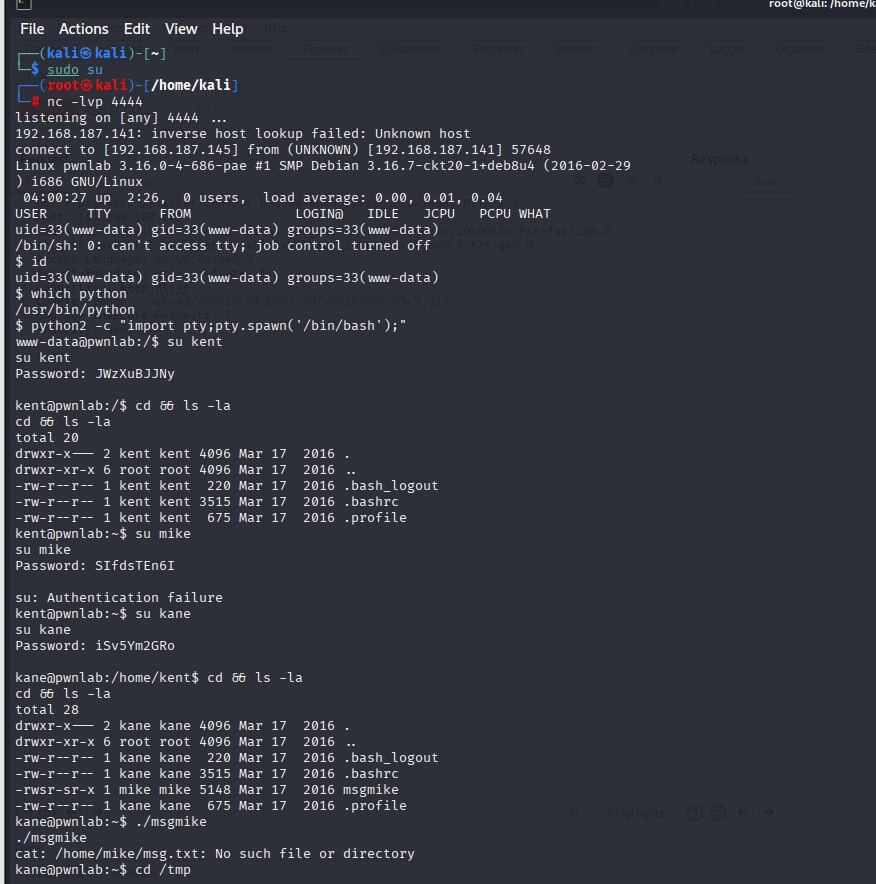
Checking if we got the access.

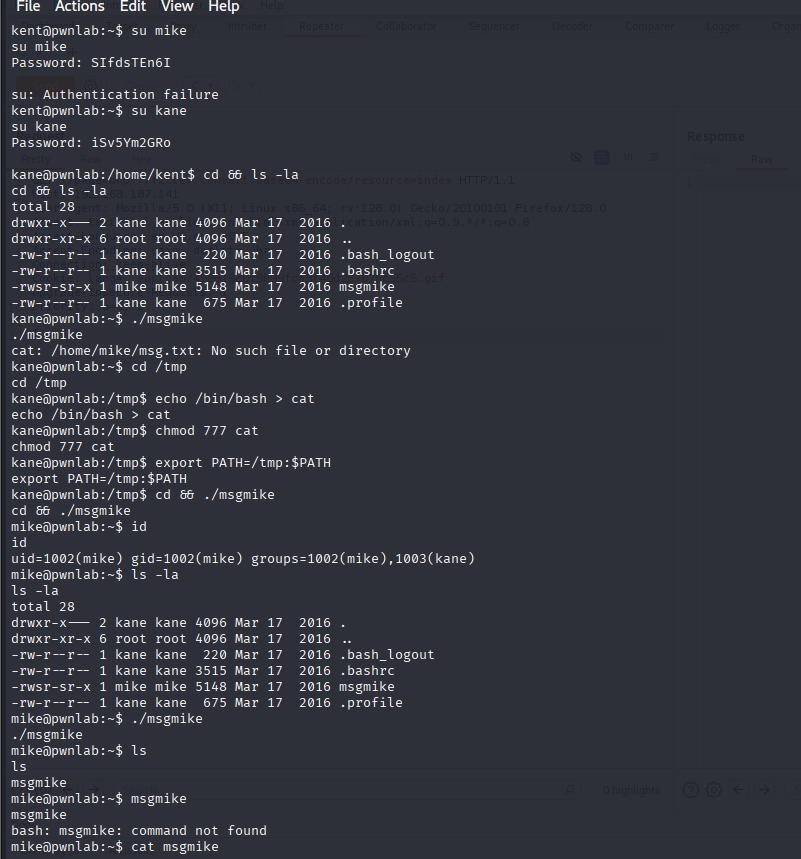


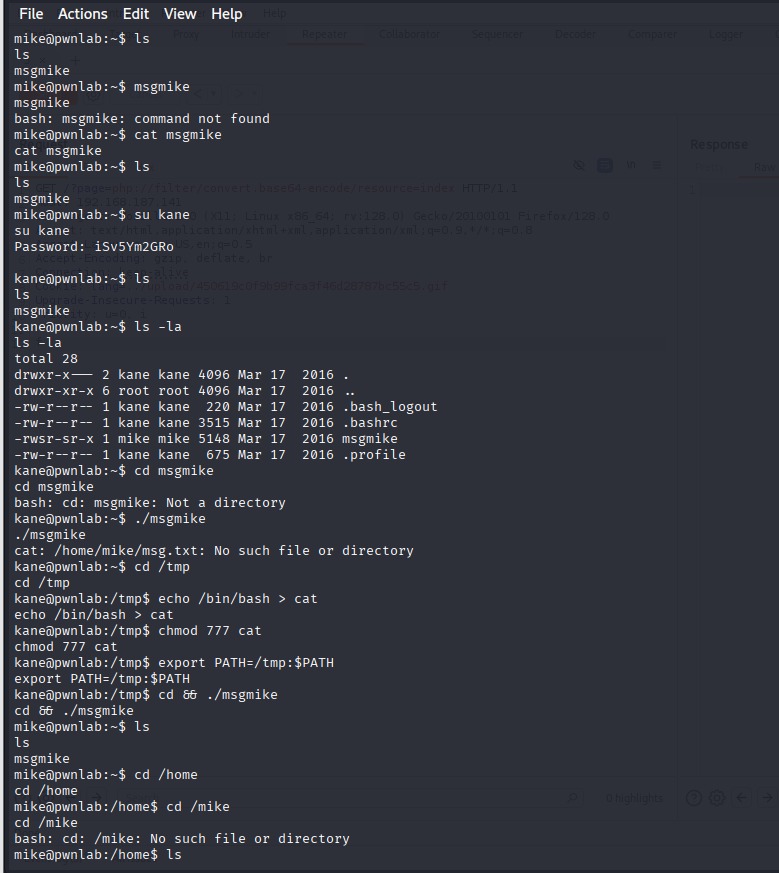
Successfully got the access of the site.

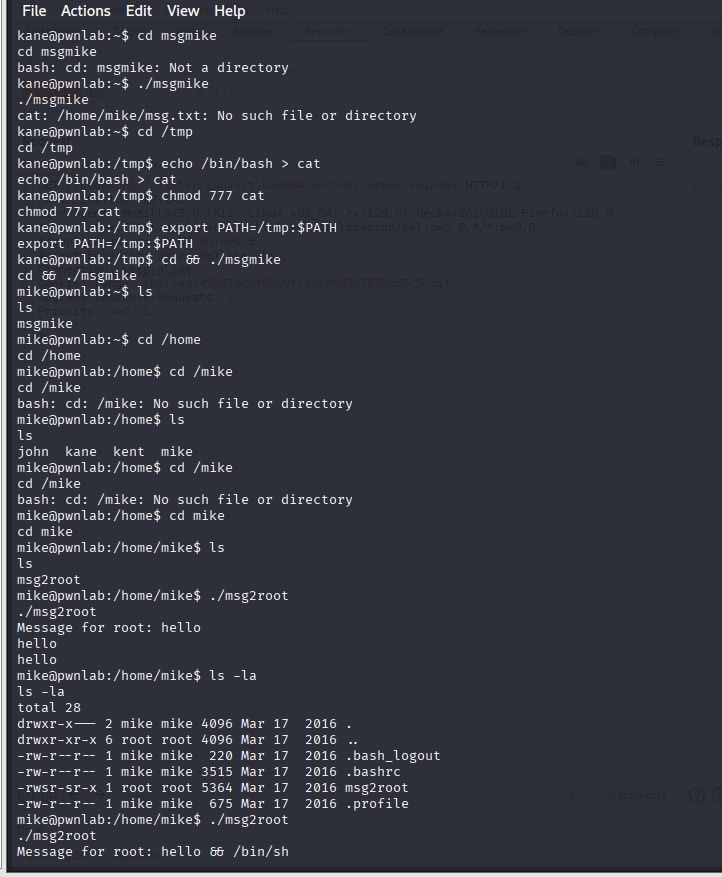
Gaining Privilege escalation.

We had the database of the users so trying to retrieve the flag.











Got the flag successfully.

Conclusion And Recommendation:

1. LFI (Local File Inclusion) with Base64 Encoding

**Description**: The application allows local file inclusion, enabling an attacker to read sensitive system files that are hidden from normal users. When combined with Base64 encoding, detection is bypassed, making it possible to exfiltrate critical data like /etc/passwd and application source code.

**Observation**: While exploiting I found the config.php page that surely should contain some information but it was not their. There I implemented this vulnerability and got the shadow code that help to get access to the database. Similarly, on index.php page we found another code that informed that cookies here contain lang as a parameter.

**Recommendation**:

* Implement strict input validation.
* Disable direct file path usage from user input.
* Use allowlists for file access.

2. MySQL Database Exploitation

**Description**: Weak authentication, exposed credentials vulnerabilities allow direct database access, enabling attackers to read, modify, or delete sensitive data.

**Observation**: After getting databases id password in config page we were able to exploit the database were we found user name and password of all the user.

**Recommendation**:

* Enforce strong database credentials.
* Restrict DB access by IP.

3. Malicious File Upload

**Description**: The system accepts unvalidated file uploads, allowing attackers to upload executable scripts or malware, potentially leading to remote code execution and full server compromise.

**Observation**: I was able to bypass the upload restriction easily and was able to upload my own php file that help me to gain access of the system.

**Recommendation**:

* Restrict allowed file types.
* Store uploads outside the web root.

4. Privilege Escalation

**Description**: Insecure configurations or exploitable flaws allow attackers to elevate their privileges from a lower-level account to root/admin, granting complete system control.

**Observation:** I had got the access to the shell at first and while exploiting the database we had got the id password of the user and surfing through kent database to kane database and at last at mike database we got our flag.

**Recommendation**:

* Patch known privilege escalation exploits
* Monitor user activities for suspicious access.
* Implement robust role-based access control (RBAC).