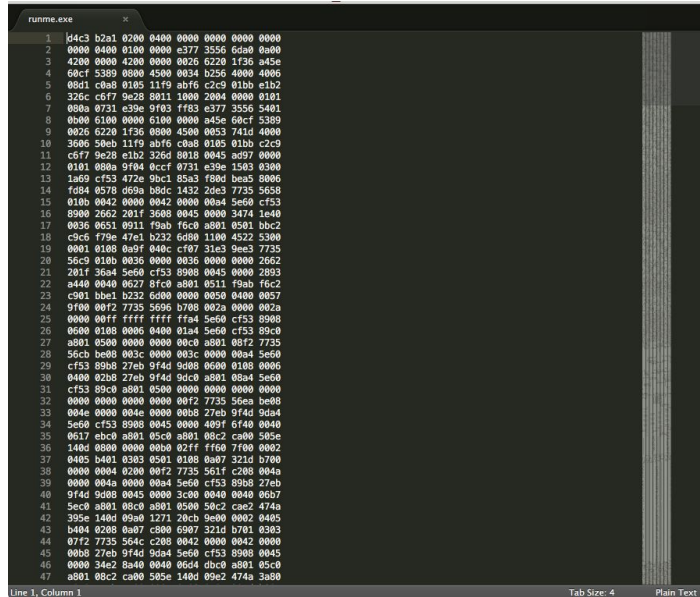


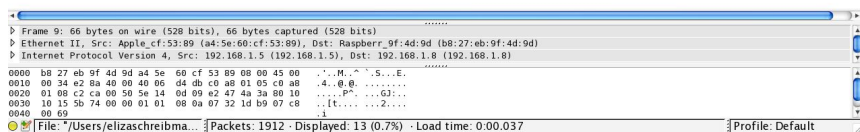
Flags

We found a flag in the runme.exe. After downloading the file and opening it in a text editor, we realized that it looked like a pcap file.



We decided to run wireshark on the file, and follow the tcp stream.

No.	Time	Source	Destination	Protocol	Length	Info
7	14.876733	192.168.1.5	192.168.1.8	TCP	78	49866->80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=32 TSval=12072
8	14.877554	192.168.1.8	192.168.1.5	TCP	74	80->49866 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_F
10	14.877678	192.168.1.5	192.168.1.8	TCP	159	49866->80 [PSH, ACK] Seq=1 Ack=1 Win=131744 Len=93 TSval=1207245
11	14.877678	192.168.1.5	192.168.1.8	TCP	66	49866->80 [FIN, ACK] Seq=94 Ack=1 Win=131744 Len=0 TSval=1207245
12	14.888205	192.168.1.8	192.168.1.5	TCP	66	80->49866 [ACK] Seq=1 Ack=94 Win=29056 Len=0 TSval=130547018 TSe
13	14.889247	192.168.1.5	192.168.1.8	TCP	66	49866->80 [ACK] Seq=1 Ack=95 Win=29056 Len=0 TSval=130547018 TSe
14	14.889568	192.168.1.8	192.168.1.5	TCP	78	80->49866 [ACK] Seq=1 Ack=95 Win=29056 Len=0 TSval=130547018 TSe
15	14.889107	192.168.1.5	192.168.1.8	TCP	66	[TCP Dup ACK 9#1] 49866->80 [ACK] Seq=95 Ack=1 Win=131744 Len=0
16	14.889529	192.168.1.8	192.168.1.5	HTTP	572	HTTP/1.1 400 Bad Request (text/html)
17	14.889531	192.168.1.8	192.168.1.5	TCP	66	80->49866 [FIN, ACK] Seq=507 Ack=95 Win=29056 Len=0 TSval=130547
18	14.889591	192.168.1.5	192.168.1.8	TCP	66	49866->80 [ACK] Seq=95 Ack=507 Win=131232 Len=0 TSval=120724932
19	14.889625	192.168.1.5	192.168.1.8	TCP	66	49866->80 [ACK] Seq=95 Ack=508 Win=131232 Len=0 TSval=120724932



This lead us to the following message:

```
Follow TCP Stream (tcp.stream eq 1)

Stream Content
Watch the video. The key is the SHA1 sum of the number, as a word in all caps, in the
video.
HTTP/1.1 400 Bad Request
Date: Sun, 01 Nov 2015 02:24:50 GMT
Server: Apache/2.2.22 (Debian)
Vary: Accept-Encoding
Content-Length: 301
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
<hr>
<address>Apache/2.2.22 (Debian) Server at 127.0.1.1 Port 80</address>
</body></html>

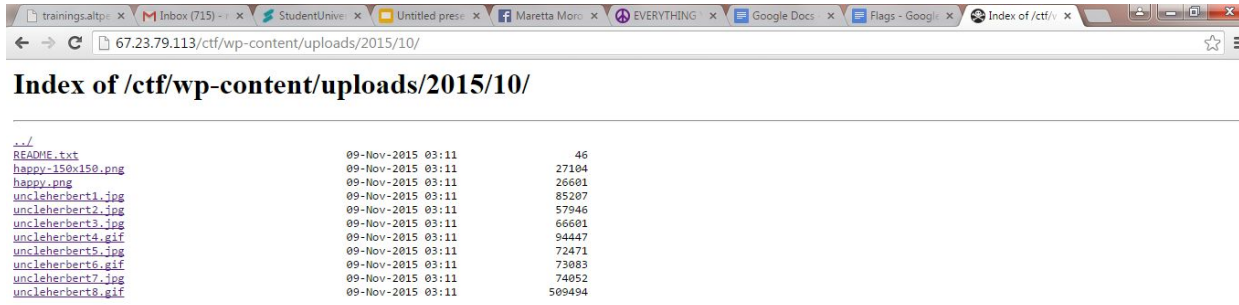
Entire conversation (599 bytes)
Find Save As Print ASCII EBCDIC Hex Dump C Arrays Raw
Help Filter Out This Stream Close
```

“Watch the video. The key is the SHA1 sum of the number, as a word in all caps, in the video.” To get this video, we had to export the object found in this stream, which produced the video sesamestreet.mp4.



We see a bunch of numbers at the very beginning of the video, so we added them and hashed the number, written out in all caps, to receive the correct flag.

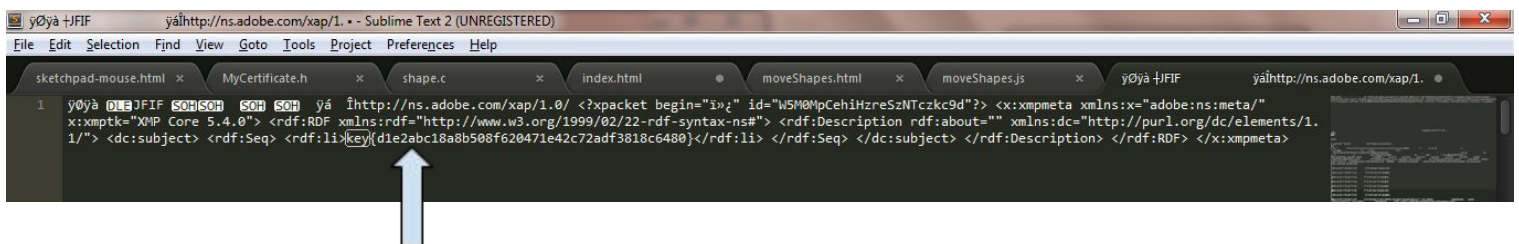
Two flags were found by exploiting the knowledge that the site was a WordPress site. All WordPress sites store uploads in the same place. Thus, even without a link to the uploads we had the ability to go to the correct page. As can be seen in the screenshot below, all of the uploads were accessible once we arrived at this page. A total of two flags were found here. The first in the uncleherbert2.jpg image and the second in README.txt



The screenshot shows a web browser window with the address bar displaying "67.23.79.113/ctf/wp-content/uploads/2015/10/". The page title is "Index of /ctf/wp-content/uploads/2015/10/". Below the title is a table listing files and their sizes.

File	Date	Size
README.txt	09-Nov-2015 03:11	46
happy-150x150.png	09-Nov-2015 03:11	27104
happy.png	09-Nov-2015 03:11	26601
uncleherbert1.jpg	09-Nov-2015 03:11	85207
uncleherbert2.jpg	09-Nov-2015 03:11	57946
uncleherbert3.jpg	09-Nov-2015 03:11	66601
uncleherbert4.gif	09-Nov-2015 03:11	94447
uncleherbert5.jpg	09-Nov-2015 03:11	72471
uncleherbert6.gif	09-Nov-2015 03:11	73083
uncleherbert7.jpg	09-Nov-2015 03:11	74052
uncleherbert8.gif	09-Nov-2015 03:11	509494

Flag found in uncleherbert2.jpg



As can be seen in the image data above, the flag was found embedded within the image header for uncleherbert2.jpg. Once we accessed the image data it was easy to pull out the flag from the other data.

Flag found in README.txt

This flag was even simpler to access than in the case of the flag in uncleherbert2.jpg. Once the README.txt was accessed the flag was the only item in the file.

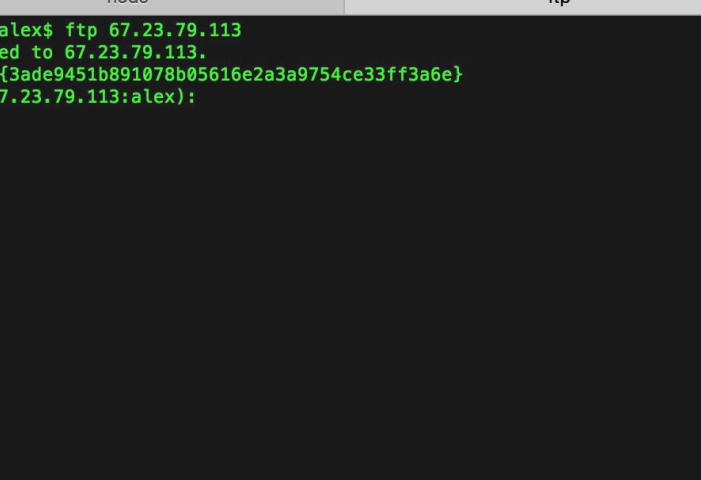
Flag found in crying.gif

The screenshot shows a Sublime Text 2 editor window with the title bar 'GIF87aÉ +'. The menu bar includes 'File', 'Edit', 'Selection', 'Find', 'View', 'Goto', 'Tools', 'Project', 'Preferences', and 'Help'. The tab bar shows several open files: 'sketchpad-mouse.html', 'MyCertificate.h', 'shape.c', 'index.html', 'moveShapes.html', 'moveShapes.js', and 'GIF87aÉ +'. The main editor area displays a large block of obfuscated JavaScript code, which is heavily commented with Cyrillic text. The code is displayed on a dark background with syntax highlighting. The code includes various escape sequences and string literals, and is likely a proof of concept for a buffer overflow. The code is displayed on lines 97 through 108.

A key was found at the end of the data for the crying.gif file found on the scoreboard. By viewing the gif data and searching for the string "key" it was an easy process to find the hidden flag.

Flag found via FTP

A simple nmap scan found an unsecured FTP process running on Port 21 of the webserver. Attempting to connect resulted in a key being given as the login prompt.

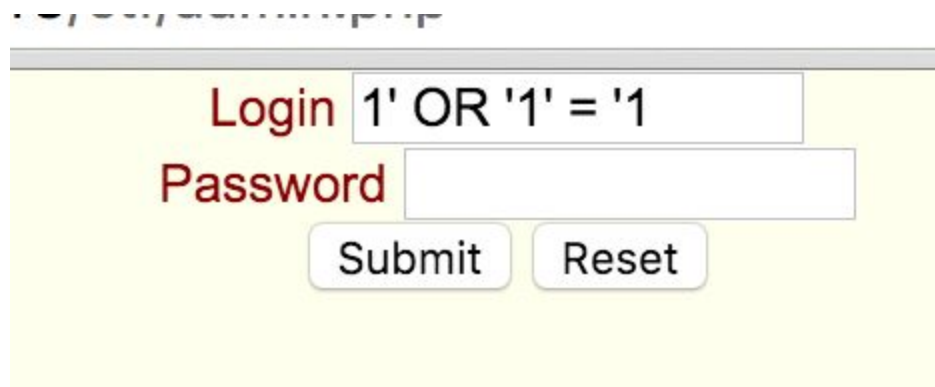


The image shows a terminal window titled "alex - ftp - 80x24". The window has a title bar with three colored buttons (red, yellow, green) on the left and a close button (a circle with a cross) on the right. The terminal content is as follows:

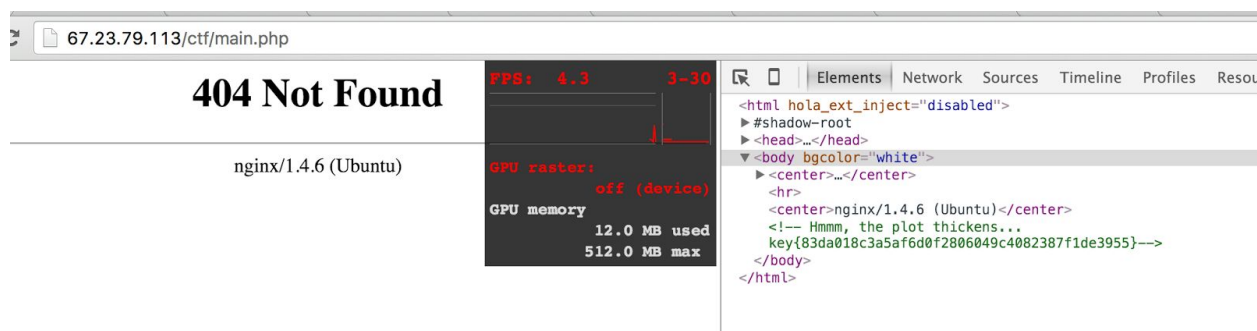
```
node
ftp
Luna:~ alex$ ftp 67.23.79.113
Connected to 67.23.79.113.
220 key{3ade9451b891078b05616e2a3a9754ce33ff3a6e}
Name (67.23.79.113:alex):
```

Flag found by SQL Injection

We found a link towards the bottom of the board page that lead to a login page.



Naturally, the first thing that came to mind was to try was SQL injection. Since there is a login, there has to be some form of data storage implemented, most likely using SQL. If the creator did not properly clean out the input from the fields and just used raw SQL statements built using string concatenation, then this SQL injection should work. Luckily for us, that's precisely what the person did. Using `1' OR '1' = '1` in the username field allowed us to bypass a password check completely. However, this went to a 404 page. But since this is capture the flag, we decided to inspect the page source anyway just in case it was a fake 404 page.



After inspecting the source, we can see that the 404 page was not an actual 404 page. Inside the center element was a comment containing a key.

Through the database intrusion we discovered user login user names, one being bobo.


```
Applications ▾ Places ▾ $ Terminal ▾ Sat 20:32 root@kali: ~
File Edit View Search Terminal Help
[+] Interesting header: SERVER: nginx/1.4.6 (Ubuntu)
[+] Interesting header: X-POWERED-BY: PHP/5.5.9-1ubuntu4.14
[!] Registration is enabled: http://67.23.79.113/ctf/wp-login.php?action=register
[+] XML-RPC Interface available under: http://67.23.79.113/ctf/xmlrpc.php
[!] Upload directory has directory listing enabled: http://67.23.79.113/ctf/wp-content/uploads/

[+] WordPress version 4.1-alpha-20141016 identified from meta generator
[+] WordPress theme in use: twentytwelve - v1.5
[+] Name: twentytwelve - v1.5
| Location: http://67.23.79.113/ctf/wp-content/themes/twentytwelve/
| Style URL: http://67.23.79.113/ctf/wp-content/themes/twentytwelve/style.css
| Theme Name: Twenty Twelve
| Theme URI: http://wordpress.org/themes/twentytwelve
| Description: The 2012 theme for WordPress is a fully responsive theme that looks great on any device. Features...
| Author: the WordPress team
| Author URI: http://wordpress.org/

[+] Enumerating plugins from passive detection ...
[+] No plugins found
[+] Starting the password brute forcer
Brute Forcing 'bobo' Time: 00:03:29 < > (10326 / 14344393) 0.07% ETA: 80:55:13
[+] [SUCCESS] Login : bobo Password : supermodel

+-----+-----+-----+-----+
| Id | Login | Name | Password |
+-----+-----+-----+-----+
|   | bobo  |     | supermodel |
+-----+-----+-----+-----+

[+] Finished: Sat Nov 7 20:30:32 2015
[+] Requests Done: 10391
[+] Memory used: 8.273 MB
[+] Elapsed time: 00:03:32
root@kali:~#
```

Using the wordlist brute force method, we found the password: supermodel. This combination granted us entry to the wordpress site where another key was discovered.

