Windows Security Assignment 2

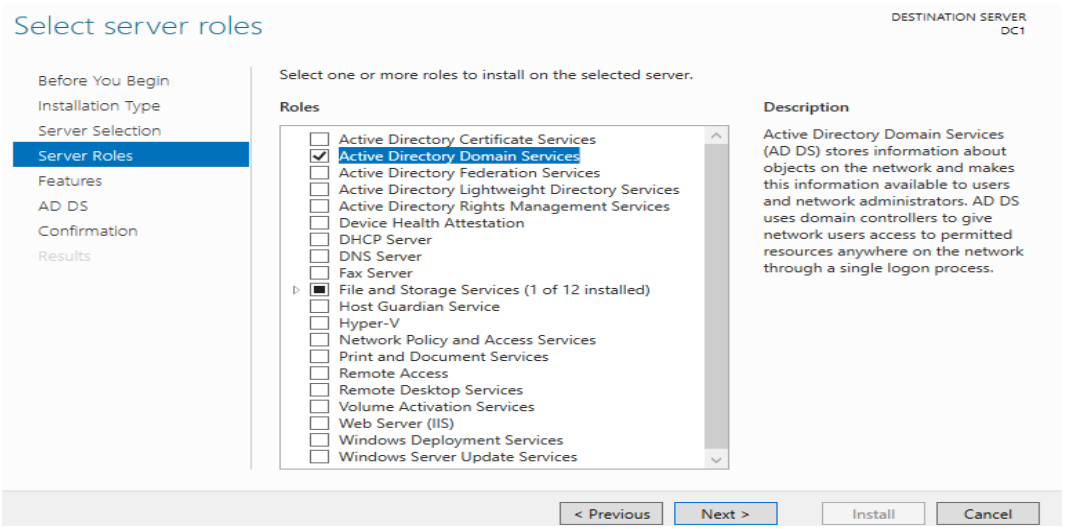
Daniel Sheehan

R00204226

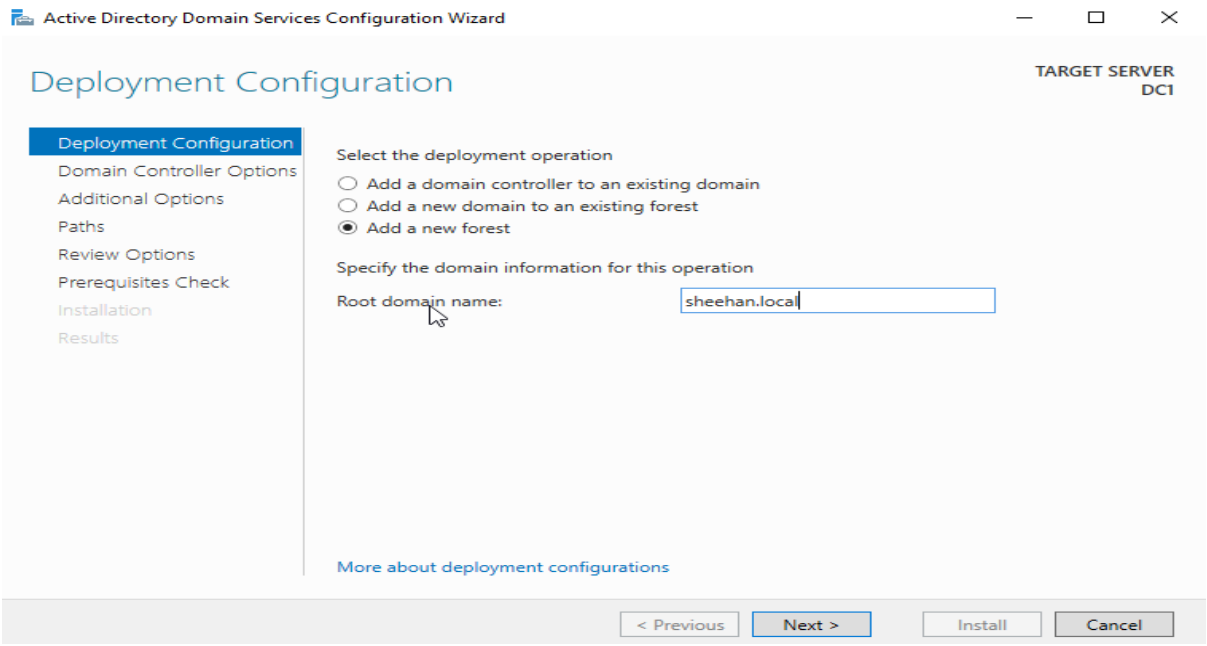
Table of Contents

Task 1: Install AD (Active Directory)

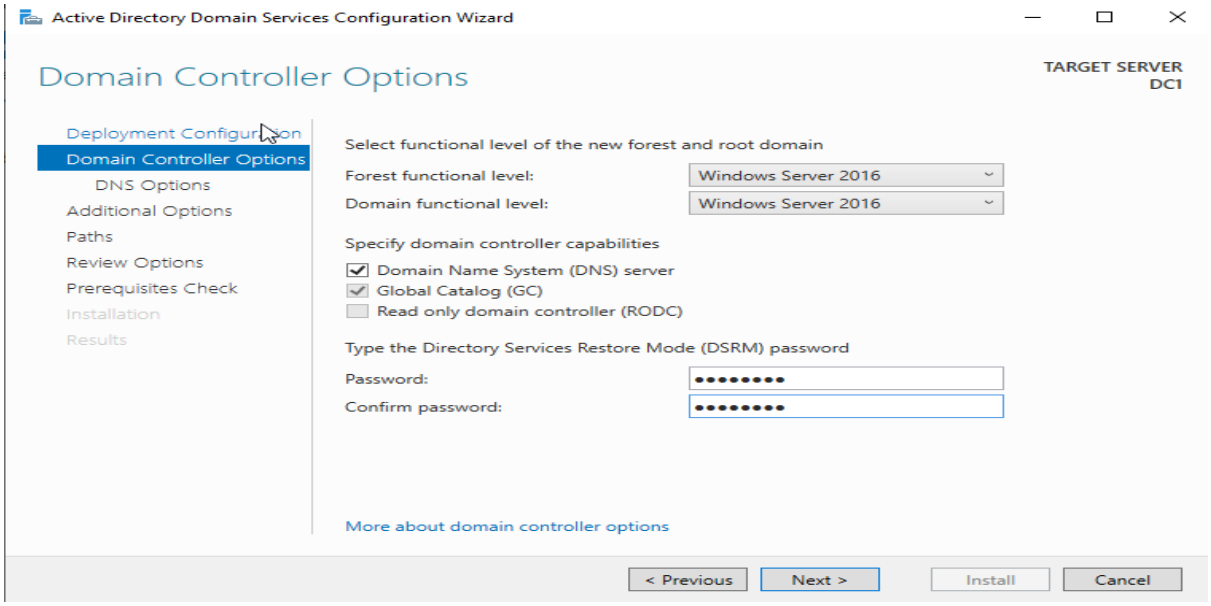
For the first task I will be installing Active Directory to a Windows Server machine. This will include setting up a domain named **Sheehan.local** and setting up a new forest. Doing this will allow me to set easily set up new users, groups and policies which I will be doing in later tasks.



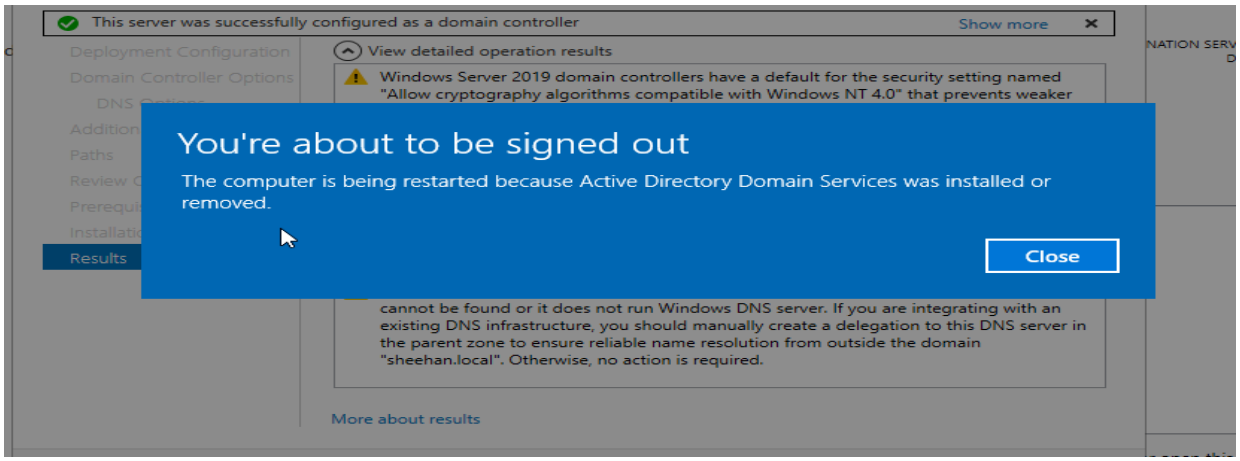
In server manager I installed new packages and selected Active Directory Domain Services (ADDS).



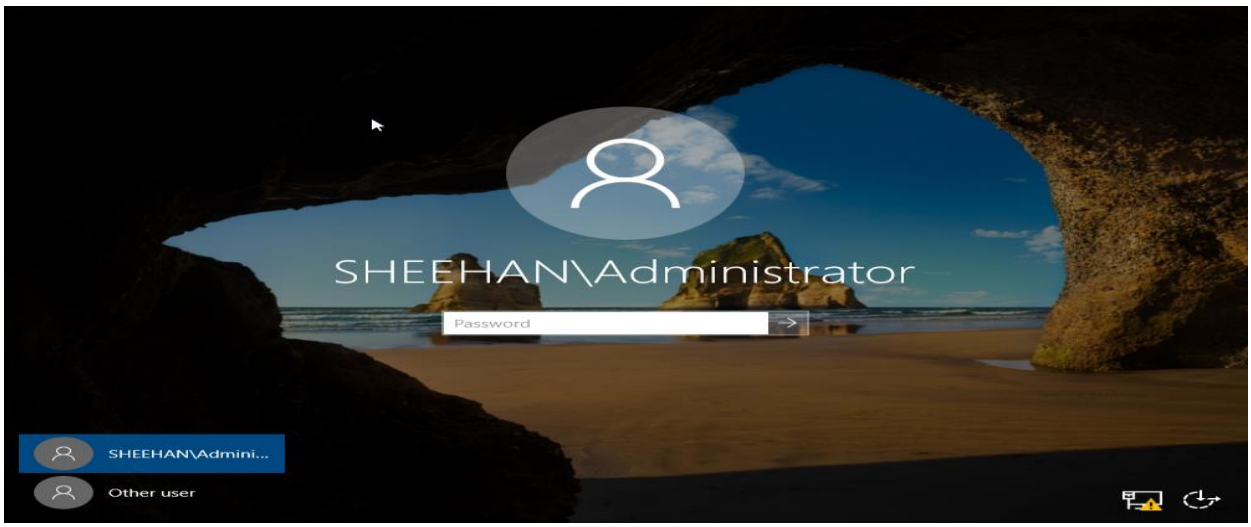
I then went to deployment configuration and added a new forest named Sheehan.local



I made it a domain controller and gave it a password as well as DNS (Domain Name Services) capabilities.



Now that Active Directory has been installed my Windows Server machine will restart for the changes to take effect in my server manager dashboard.

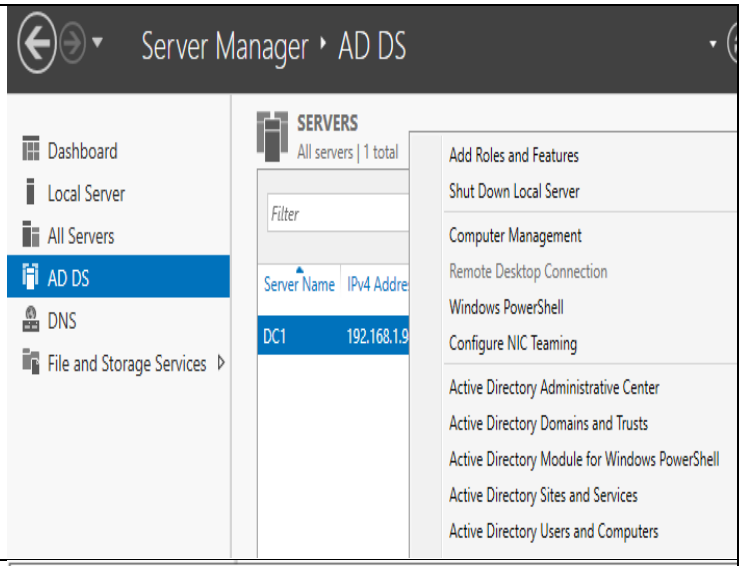
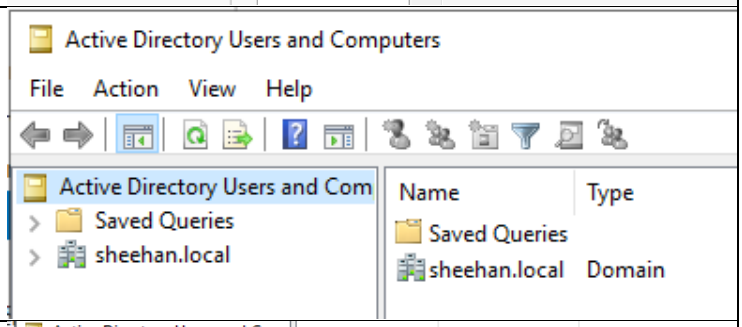
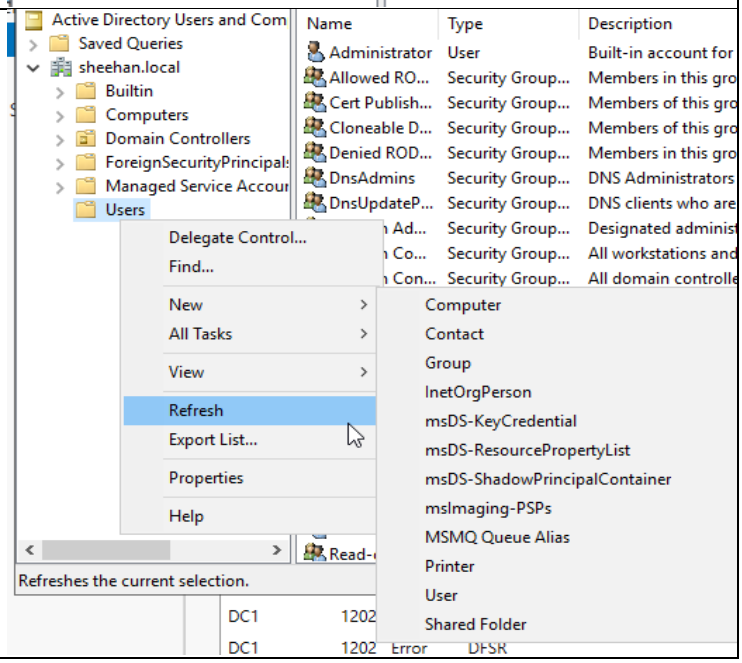


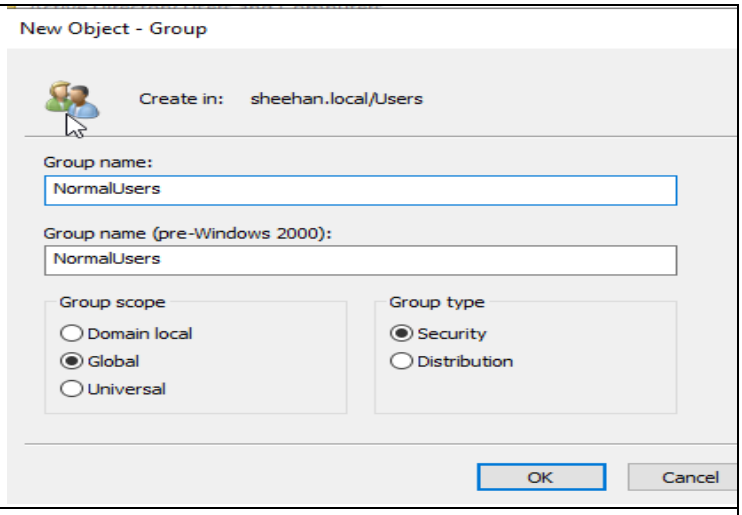
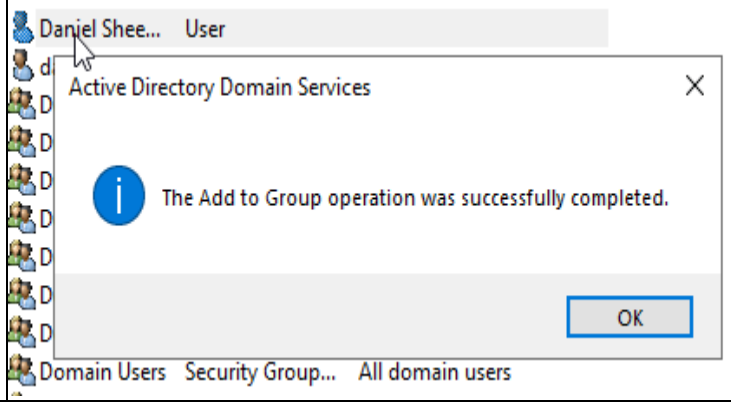
Task 1 Learning Outcome:

Through completing this task, I learned how to set up and configure Active Directory Domain Services on a Windows Server machine, this is a very important step when building a domain based network. I now understand the importance of AD as the main component for managing users, devices and security policies across an organization's internal network. Creating a new forest and domain (Sheehan.local, in my case) gave a foundation for central authentication and access control. I also gained experience using the Server Manager to add roles and features which showed me that installing ADDS also enables DNS (Domain Name Services) as it is needed for domain resolution. In the end Task 1 showed me that Active Directory streamlines network administration and prepares the network environment.

Task 2: Users & Groups

For this task I will be creating users and groups using Active Directory Domain Services on my Windows Server machine. I will make 2 user groups called NormalUsers and AdminStaff and adding users to them.

I started this task by opening up Server Manager and right clicking AD DS, I then selected Active Directory Users and Computers	
I then selected Sheehan.local (my domain) and expanded it selecting the Users folder	
Once I found the Users folder I selected it and right clicked it, I then clicked on New and from the drop down I clicked Group to begin making a new user group.	

I made the group as shown in my screenshot and repeated the same process to make the 2nd group. **New Groups:** NormalUsers AdminStaff	
This time after navigating back to the users folder I right clicked it, and this time created new users to add to the groups **New Users:** Daniel.sheehan **added to** NormalUsers group. Daniel.a.sheehan **added to AdminStaff** group.	

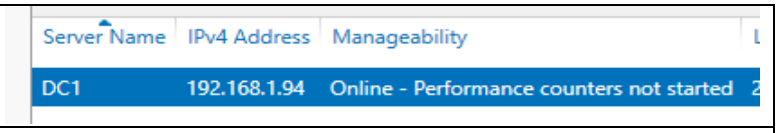
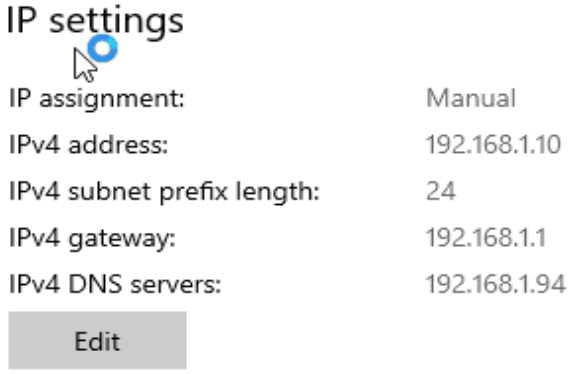
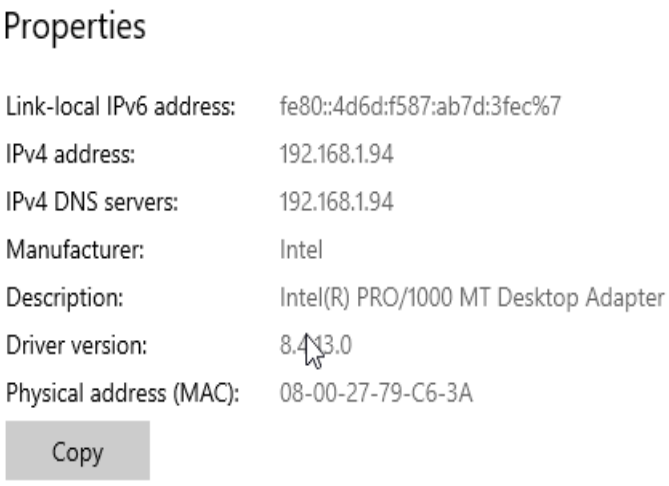
Task 2 Learning Outcome:

By completing this task , I learned how to manage users and groups using the Active Directory Users and Computers console. I now understand the importance of structuring users into groups that allow for easier ways to apply policies and permissions in a server domain. Setting up security groups like NormalUsers and AdminStaff allows network administrators to assign permissions based on what group a user is a member of rather than assigning rules to each individual user as that would be a time consuming and complicated process. I can see this being common practice in many enterprise environments that use Windows services to manage their organization.

After creating the groups I also learned how to manually create new user accounts and assign them to the groups that I had previously created in order to practise what I had learned, doing this is essential for role-based access control on a network. This task gave me hands on experience in setting up user groups and assigning users to them to maintain a well organized user management system.

Task 3: Joining Client to Server Domain

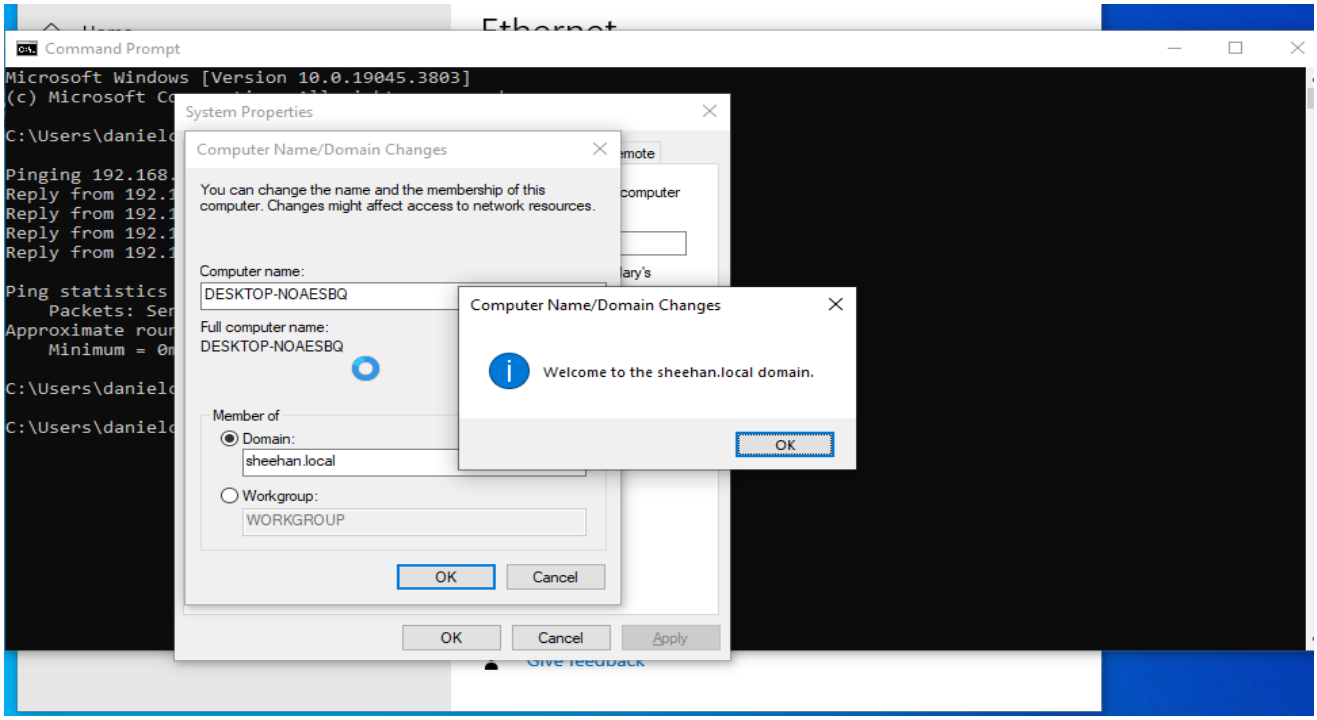
For this task I had to join the Windows client machine to the domain I set up on the server. I started by changing the DNS settings on my Windows 10 client VM to match the DNS on my server (**192.168.1.94**)

First I checked the IP address of my DNS server in my Server manager	
IN CLIENT (WIN10): On my Win10 client machine I manually configured my static IP address to use this DNS address in order for things to work properly. <u>IP Settings</u> IPV4 – 192.168.1.10 Default Gateway – 192.168.1.1 DNS Server: 192.168.1.94	
IN WINDOWS SERVER: In my windows server machine I disabled DHCP to make things easier and configured it so my ADDS/DNS server use a static IP address so that everything works at each startup without errors. <u>IP Settings</u> IPV4 – 192.168.1.94 Default Gateway – 192.168.1.1 (same gateway) DNS Server: 192.168.1.94	

After this I verified that the machines could communicate by pinging both ways across the network. Once this was successful I joined my WIN10 client to the domain of my server

I knew this was successful when I was prompted

Welcome to the sheehan.local domain



Task 3 Learning Outcome:

This task taught me how to join a Windows 10 client machine to a domain that is being managed by a Windows Server using Active Directory. I learned how to properly configure network settings, including a static IP address and assigning the DNS server to point to the domain controller's IP. The reason I did this was because it is essential so the client can locate and communicate with the domain for authentication.

I also gained a better understanding of how DNS and domain services work together. If I didn't point the client's DNS to the domain controller, the domain join process would fail. Disabling DHCP and using static IP's on both the server and client helped make sure that there was always a stable connection between the client and the server even in the event that they are shut down and restarted. Once I had all of this working successfully I used the system properties menu on the client to point towards the "sheehan.local" domain and join it. In the end I was prompted that the domain join was successful.

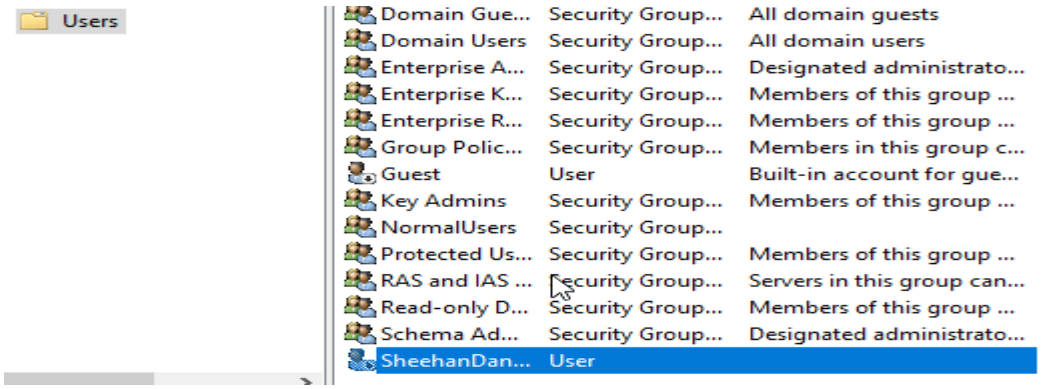
This task gave me practical experience in joining a domain on a client machine which is a useful skill to have in enterprise environments as it allows for easy network management.

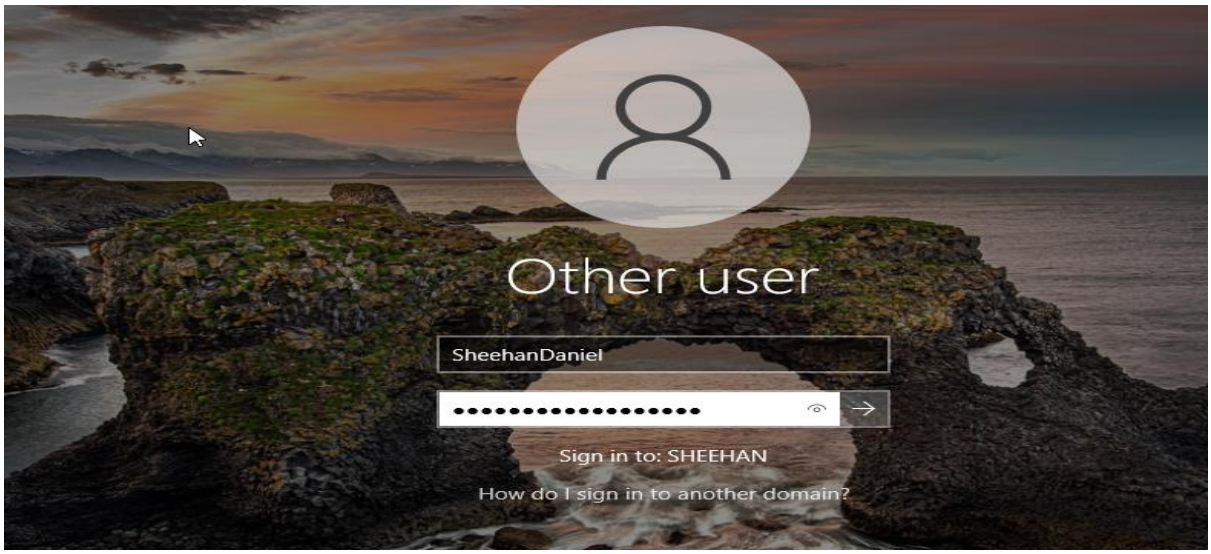
Task 4: New AD User

For this task I will be using PowerShell on my server to create a new user account and add it to the NormalUsers group. I will also make sure that the new user is not disabled by default in my cmdlet.

```
PS C:\Users\Administrator\Daniel> New-ADUser -Name "SheehanDaniel" -SamAccountName "Sheehan" -Enabled $True -AccountPassword (ConvertTo-SecureS
tring "Password101Daniel!" -AsPlainText -Force)
```

Next I went to the users folder in my ADDS server in the server manager to verify that the account was made. Once I saw the account I logged in on my client to confirm that it existed on my domain.





With this I can confirm that I have successfully made the account through PowerShell.

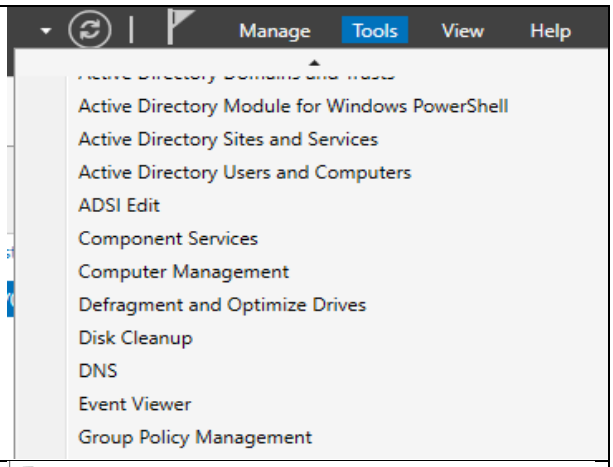
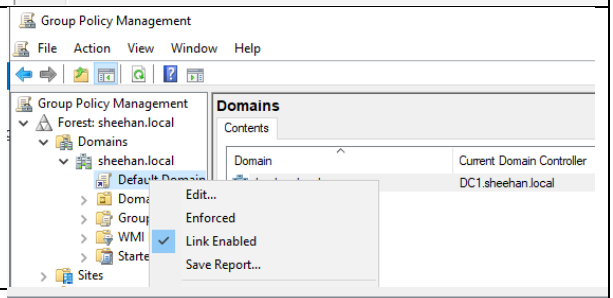
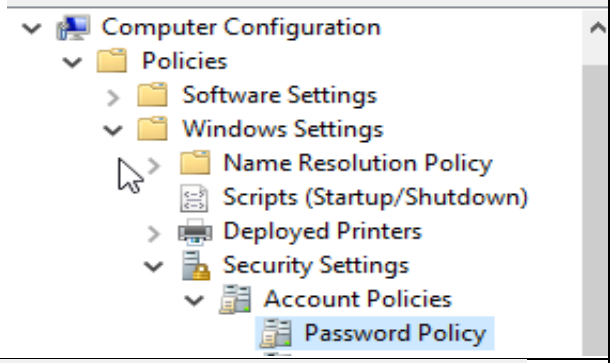
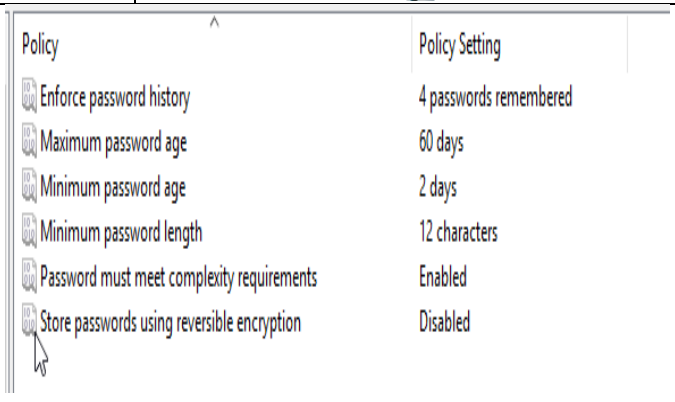
Task 4 Learning Outcome:

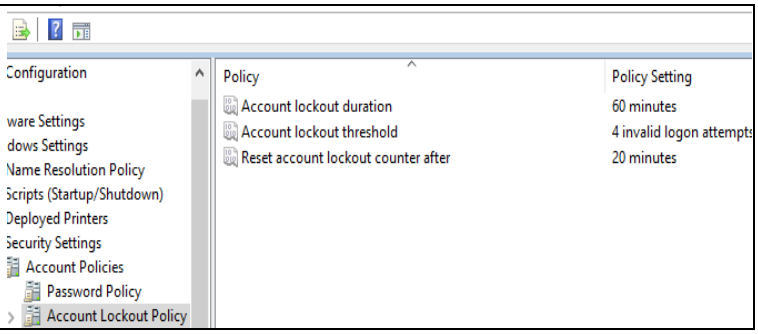
This task introduced me to creating new user accounts through PowerShell as it could be a faster and more scalable way than using the GUI. I learned how to use the **New-ADUser** cmdlet to create a user and included the **-Enabled $true** part to make sure that the account is active immediately. PowerShell makes it possible to automate the creation of multiple users which can definitely be useful in larger network environments where managing accounts manually could be difficult and time consuming. For example, if an organization had 50 new hires they could use a PowerShell script to automatically set up a new user account for each employee hired in a matter of seconds rather than setting them all up one by one with user interface. This cmdlet also let me set a password immediately on account creation and with the correct password policies implemented the user could possibly even change their password once they log in for the first time adding an extra layer of security.

Another thing that I learned was how user properties like “Name” etc can be specified using parameters in a PowerShell command adding more flexibility when creating new accounts. After I created the user, I then verified that it appeared in “Active Directory Users and Computers”, Once I saw the newly created account was displayed in the menu I then attempted to log into the new account on my client machine. I logged in successfully confirming that the new account was created and activated successfully.

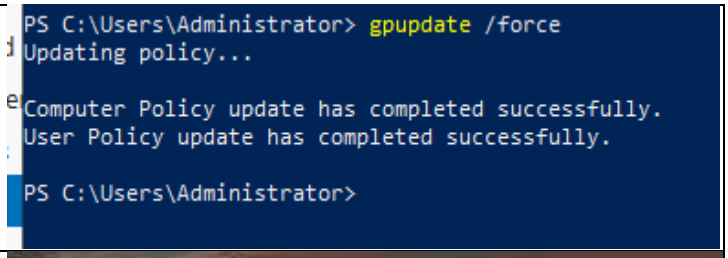
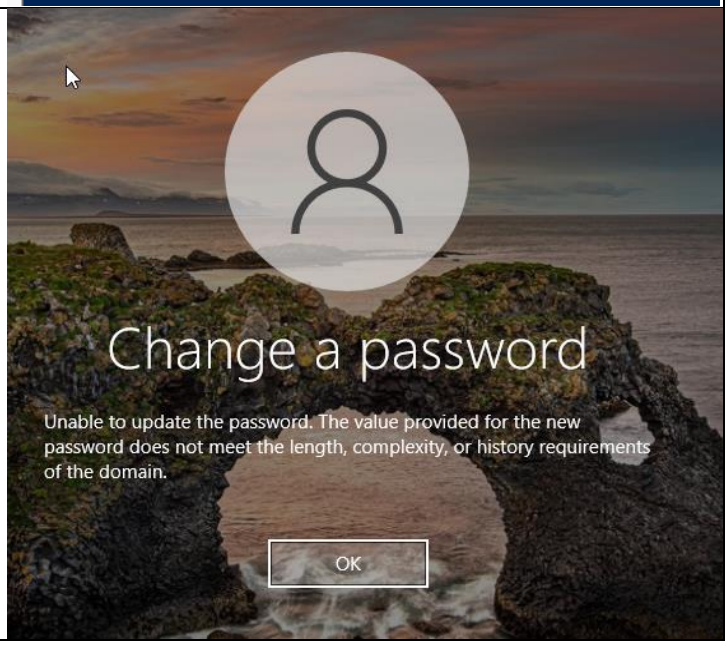
Task 5: Group Policy Objects (GPO)

For this task I will be using a GPO manager in my server dashboard to set up a password policy that remembers 4 passwords for a maximum age of 60 days a minimum password age of 2 days and a minimum password length of 12 characters. I will also set up an account lockout policy that locks accounts for a duration of 60 minutes after 4 invalid login attempts and the account lockout counter resets after 20 minutes. I will then demonstrate these changes on my WIN10 client.

In my server management dashboard I clicked tools and from the dropdown menu I selected Group Policy Management	
In the Group Policy Manager I expanded my **sheehan.local** domain and right clicked on Default domain where I then selected "Edit"	
In this I followed the path: Computer Configuration > Policies> Windows settings > Security Settings> Account Policies > Password Policy	
Here I then edited the password policy to meet the requirements of this task: Enforce Password History – 4 password remembered Max Password age – 60 days Min Password age – 2 days Min Password length – 12 characters Must meet complexity – Enabled	

I then did the same for Account Lockout Policy: Account lockout duration - 60 mins Account lockout threshold – 4 invalid attempts Reset lockout counter - 20 mins	

Testing the policy:

I applied the group policies using the command: gpupdate /force	
Here I can see that the password policy is working correctly	

Task 5 Learning Outcome:

In this task, I learned how to use Group Policy Objects (GPOs) to enforce security settings across all users on a domain. I learned that I am able to edit the default domain policy as well as create entirely new policies to implement rules for specific things such as passwords, lockout policies and more.

For this task I was able to configure both password policies and account lockout policies in order to ensure stronger account security and set better security standards for my organisation's domain. Setting a password history and minimum/maximum age helps prevent users from reusing old passwords and encourages them to regularly change their password, I also made it, so the password had a requirement for a minimum length making them more complex and harder to guess protecting users on my network against brute force and dictionary attacks.

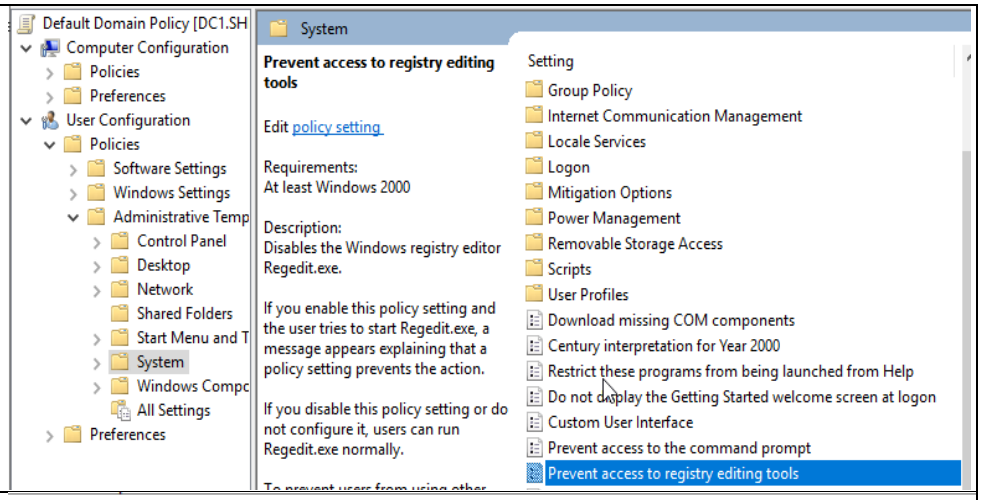
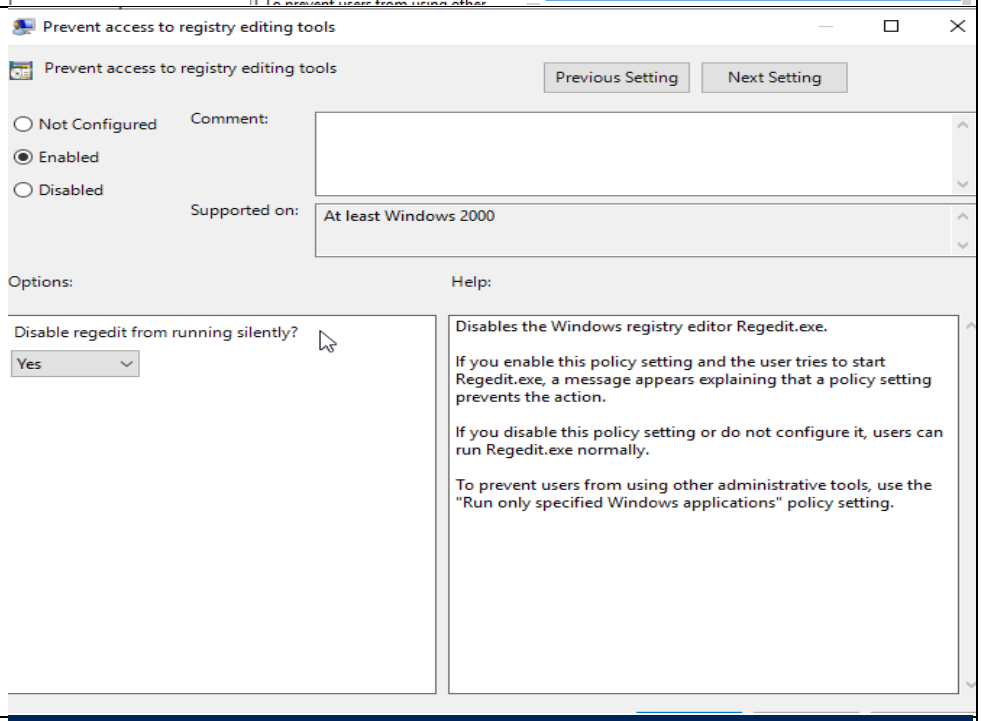
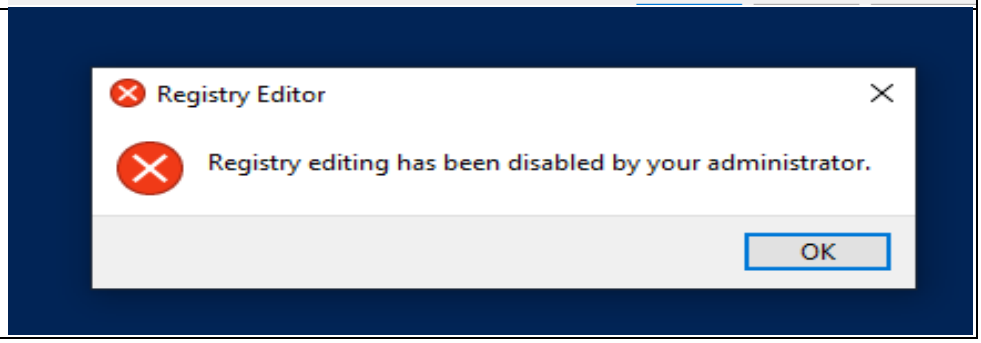
Another thing that I implemented was a account lockout policy that temporarily disables accounts after a number of failed login attempts. This helps defend against repeated logins from bots and unauthorized users. Using **gpupdate /force** I made sure that the GPO changes applied immediately across my domain, updating the policies. After doing this I tested to see if my policies had been implemented by attempting to change the password of an account without following the new rules that I set. After this I was notified that the new password didn't meet length, complexity and history requirements of my domain , showing me that it is working successfully.

This task helped me understand how GPOs can give central control over security and lockout policies to monitor and restrict user behaviour, this is extremely important for managing domain environments securely and effectively.

Task 6: GPOs

A. Prevent Registry Editor Policy

To start I navigated back to my Server Manager's Dashboard and clicked tools in the top right corner, From the dropdown I selected Group Policy Manager

In the group policy manager I followed the paths Sheehan.local>User Configuration>Policies>Administrative Templates> System I then double clicked the option to do with registry editing tools.	
In the prevent access to registry editing tools window I changed the selection on the top left to Enabled and then I clicked apply followed by ok	
I then tested the new policy that I implemented by attempting to open the registry editor where I was met with this message :	

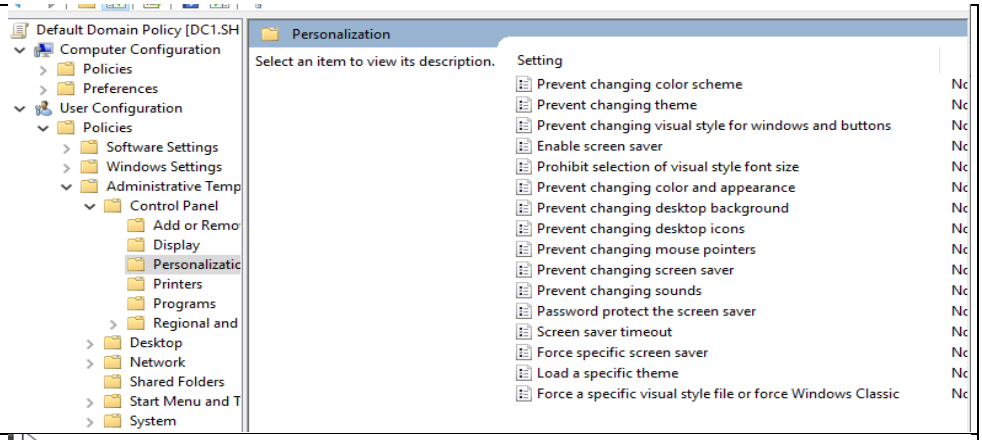
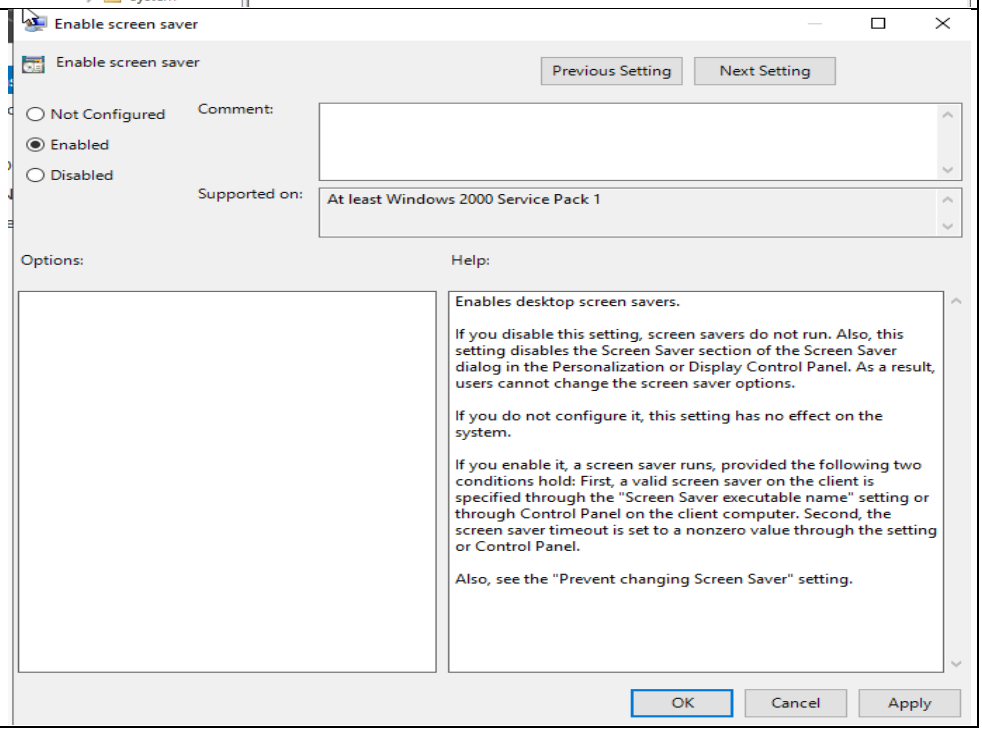
Task 6 (a) Learning Outcome:

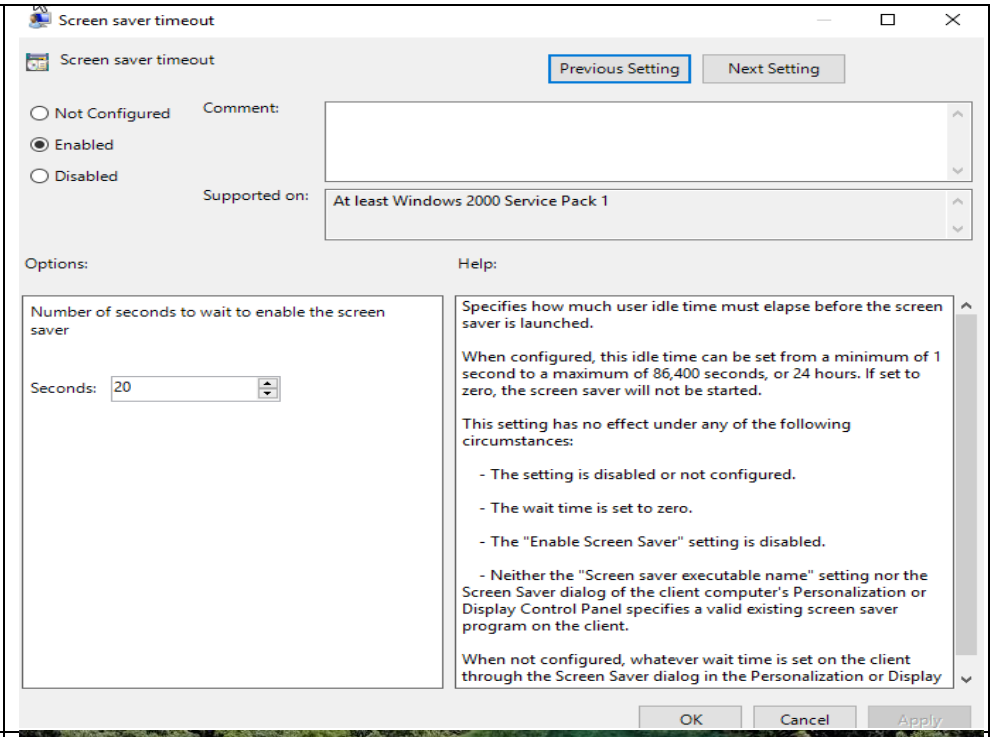
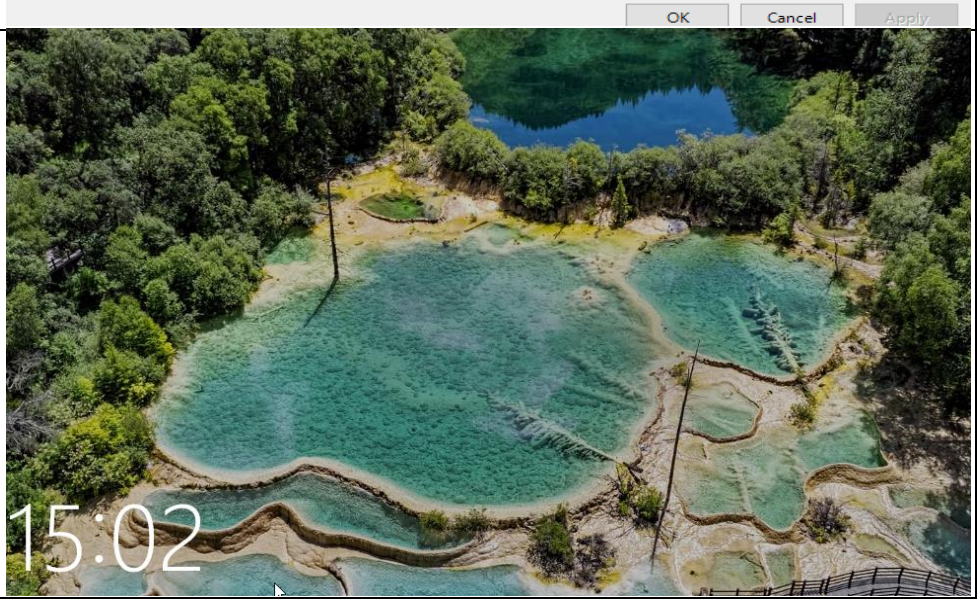
In the first part of Task 6 I learned how to use Group Policy Objects (GPOs) to restrict access to sensitive system tools like the Windows Registry Editor. Using the Group Policy Management menu under my domain, I made a new policy and followed "User Configuration" I followed the path until I found "**Prevent access to registry editing tools"** under the "System" directory, this blocks users from opening regedit.exe if they are not authorized to do so. If they had access to this tool they could potentially make changes to system settings that could affect stability, performance and security putting the whole network at risk.

This part of Task 6 helped me understand the value of implementing user level GPOs in environments where end users should not have administrative rights as well as the ability to alter registry values. I was not aware that access to specific applications and services could be restricted using Group Policy Objects. By testing the GPO edit on my Windows 10 client machine, I was able to confirm that I successfully implemented my new policy and disabled registry access for unauthorized users on the domain. Overall this showed me how to use and apply GPOs to control access to specific system tools and improve security across my domain.

B. Set Screensaver Policy

I navigated back to my Server Manager's Dashboard and clicked tools in the top right corner, From the dropdown I selected Group Policy Manager again:

Once in the Group Policy Manager I made another new policy and followed the path : User Configuration>Policies>Administrative Templates> Control Panel> Personalization	
I selected the option " Enable Screensaver" and set it to Enabled in the top left, I then clicked apply and okay	

Next I selected the option “Screensaver Timeout” and set it to Enabled in the top left, I then set the number of seconds to 20 and pressed apply then ok.	
As I can see the policies have been changed and implemented successfully as after 20 seconds of inactivity my screensaver is activated	

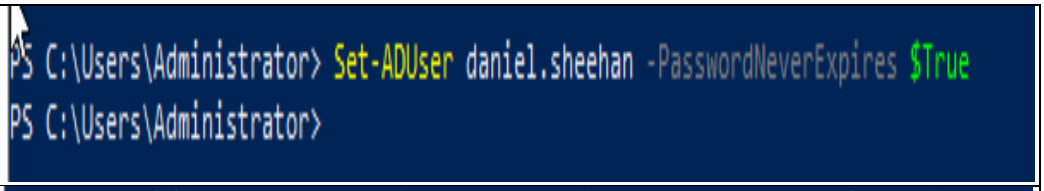
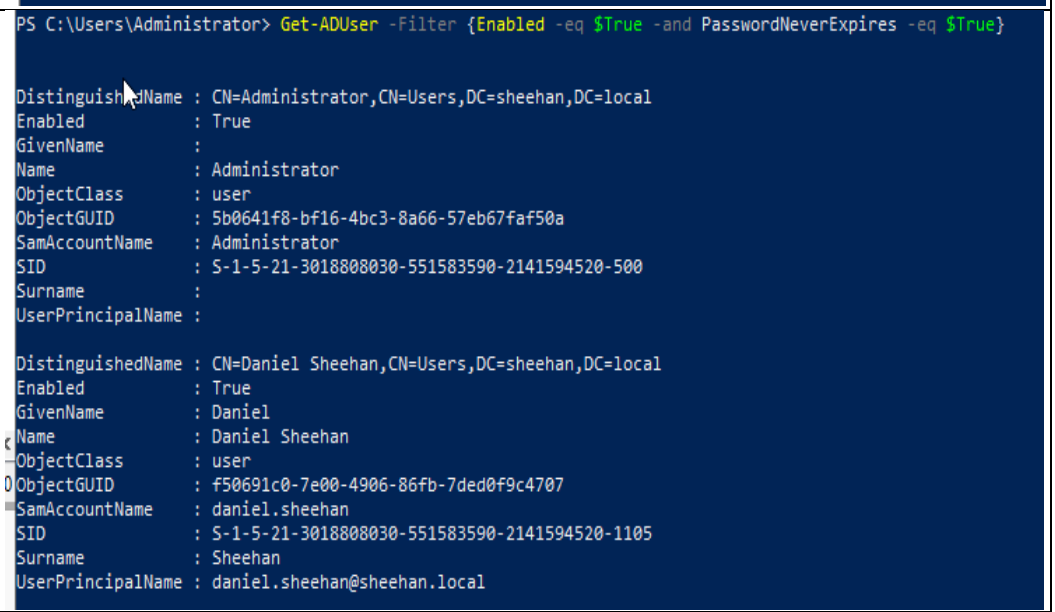
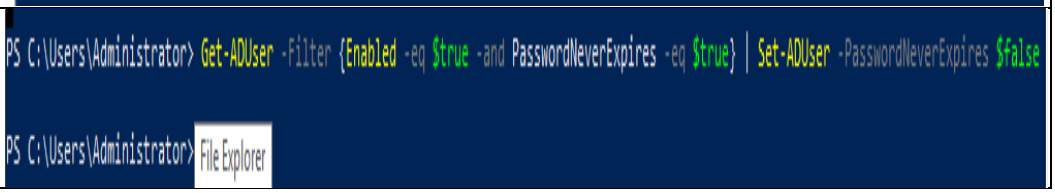
Task 6 (b) Learning Outcome:

In this part of Task 6, I learned how to use Group Policy Objects (GPOs) to configure personalisation settings for users on a domain ,specifically the screensaver timeout policy. By editing the GPO under **User Configuration>Policies>Administrative Templates> Control Panel> Personalization,** I enabled the screen saver and set its timeout value to 20 seconds. This ensures that if a user leaves their device unattended, it will automatically enter a locked state after a short period of inactivity.

This part of task 6 helped me understand how GPOs can also be used to manage user behaviour on a network domain. By editing the screensaver and it’s timeout policies I can limit the risk that if a machine is left unattended a user can gain unauthorized access to the vulnerable machine. This taught me that even things that may be overlooked such as timeout policies are an important factor to consider when taking security into account in a shared work environment.

Task 7: PowerShell Task (Password Expiration)

Next I will be using PowerShell on my server to make it so one of my user accounts' (daniel.sheehan) Password never expires. After this I will be making it so that all active directory enabled user accounts with non-expiring passwords are listed using a separate command.

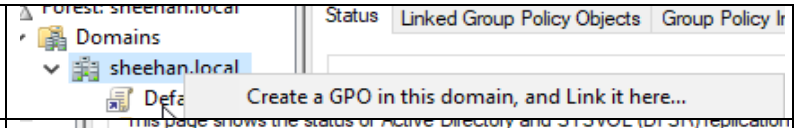
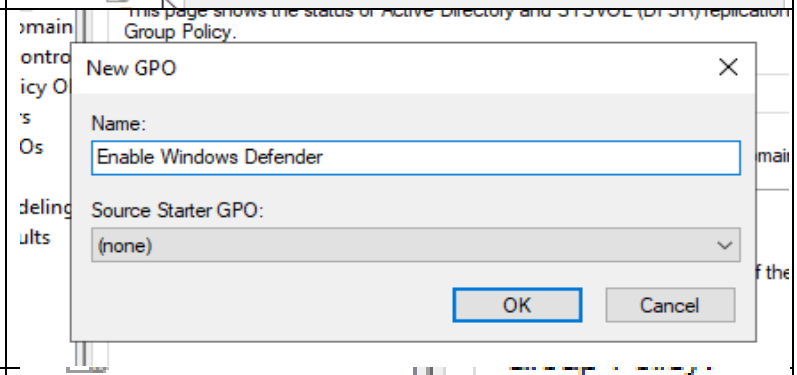
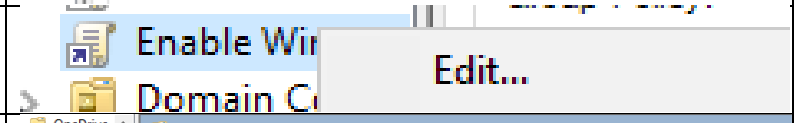
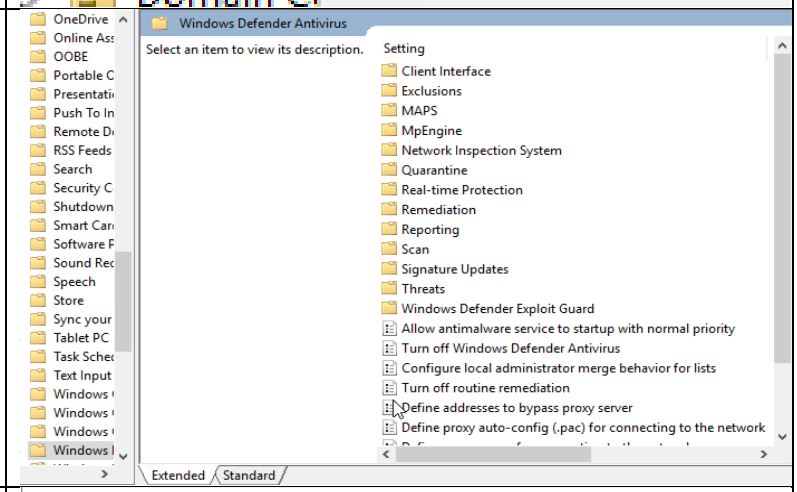
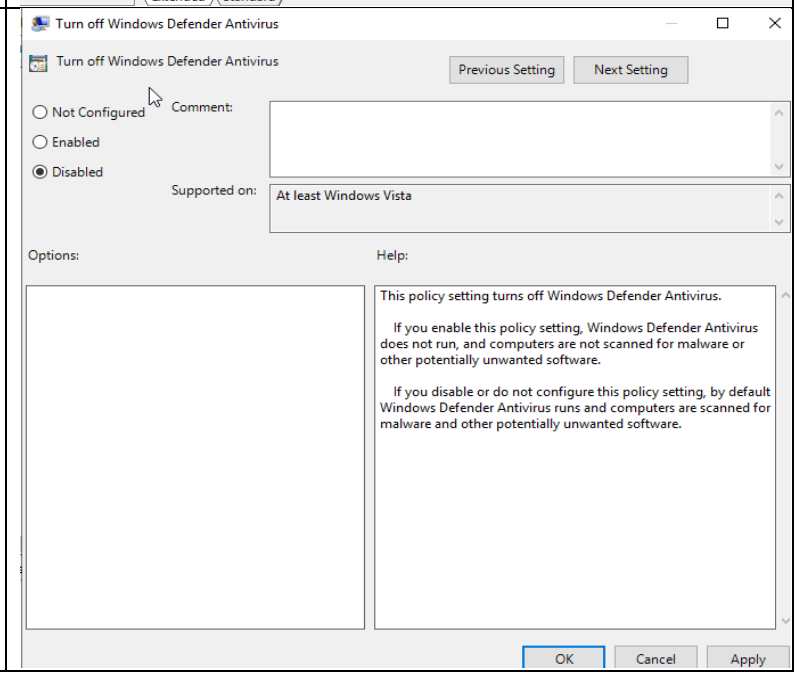
Here I set it so that for the user daniel.sheehan the password never expires	
Here I used this command to list all Active Directory Users with non-expiring passwords.	
Lastly I forced all users that were listed in the previous command to have passwords that expire again.	

Task 7 Learning Outcome:

In this task I learned how to use PowerShell to view and modify password expiration settings for Active Directory users. In this case I used the **Set-ADUser** cmdlet to configure one user account (daniel.sheehan) and make it so that the password of the account never expires. After this I used the **Get-ADUser** cmdlet with filters to list all enabled accounts where the password expiration setting was disabled so I could audit my domain and check if the changes were successful.

I the used a pipe (|) to apply a change to all users in the result of the previous cmdlet to re-enable password expiration for any accounts that I had disabled it for. This task helped me understand how to manage accounts effectively using PowerShell and again show that it can be useful when managing multiple accounts as well as their policies at once.

Task 8: GPO Windows Defender

For this task I created a GPO that enables Windows Defender	
I named this new GPO Enable Windows Defender and clicked ok.	
I right clicked on my new GPO and clicked Edit	
I followed the path as specified and double clicked “Turn off Windows Defender Antivirus”	
I then disabled “Turn off Windows Defender Antivirus” as this means it would be enabled.	

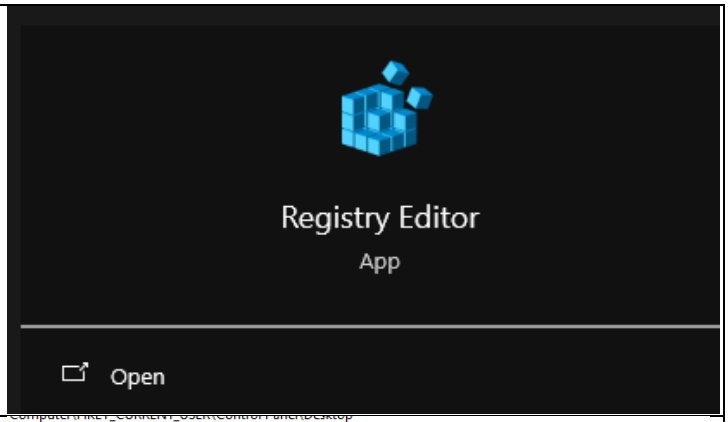
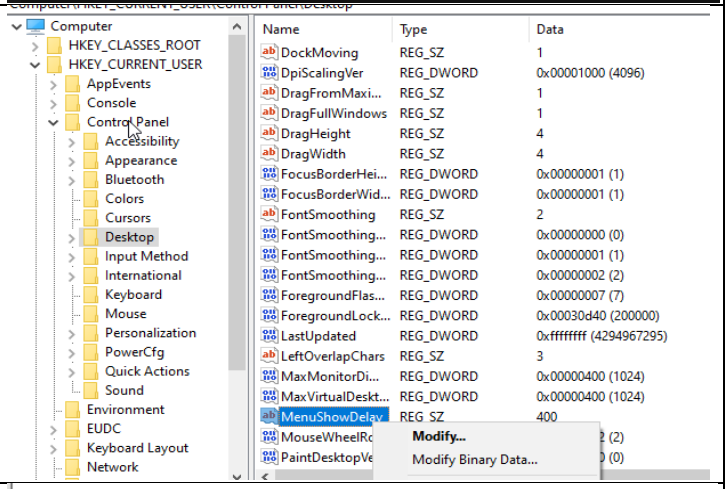
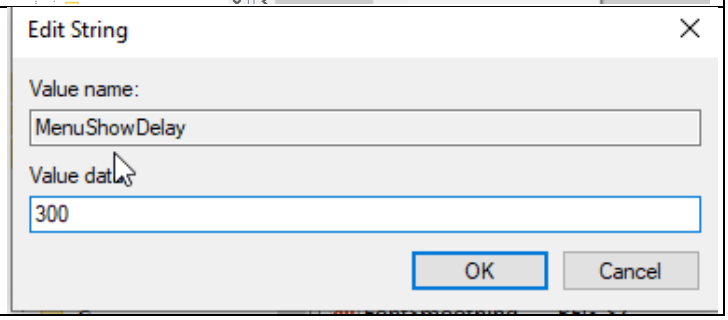
Task 8 Learning Outcome:

In Task 8 , I learned how to use Group Policy Objects to enable Microsoft/Windows Defender Antivirus on machines that are part of my domain. I created a new GPO and navigated to the setting "**Turn off Microsoft Defender Antivirus**" under **Computer Configuration > Administrative Templates > Windows Components > Microsoft Defender Antivirus.**

A weird thing that I found was that setting this option to "Disabled" actually enables Microsoft Defender making it active on the domain, I personally find this is an interesting yet odd way of wording such an important feature. I even had to double check it was enabled in the first place.

Overall this taught me how group policy isn't just used to implement password policies and set system restrictions, it can also be used to enable security features such as Microsoft Defender, The fact it is set as a domain policy also makes it extra secure as it is easier to configure through the domain controller and also makes it so that end device users on the domain network cannot manually disable it, Keeping the security standards of the domain network consistent and strong. I was unaware that GPOs could be used to enforce antivirus policies and maintain security standards , especially in larger more complex networks.

Task 9: Windows Registry

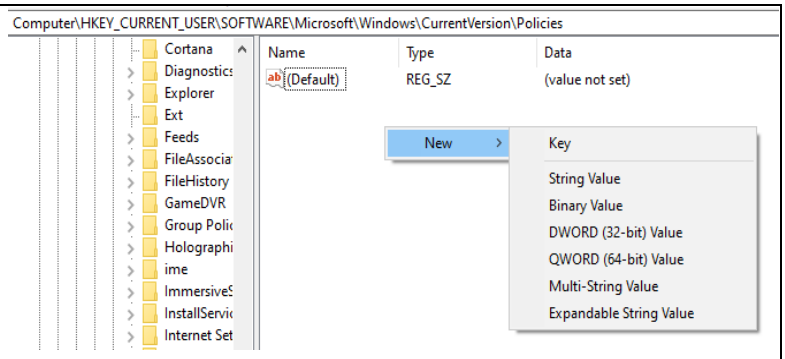
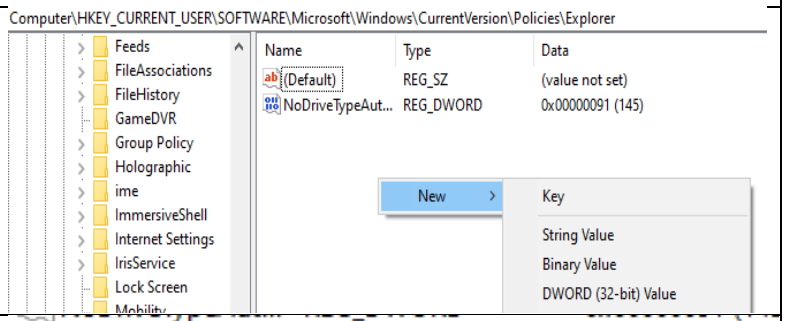
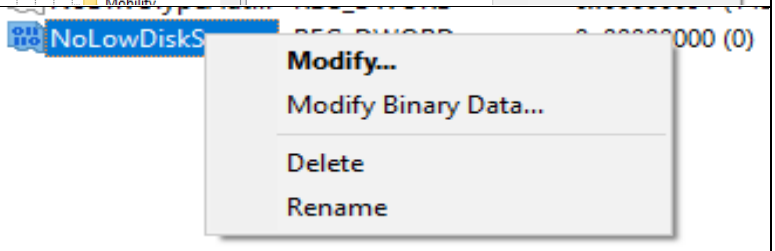
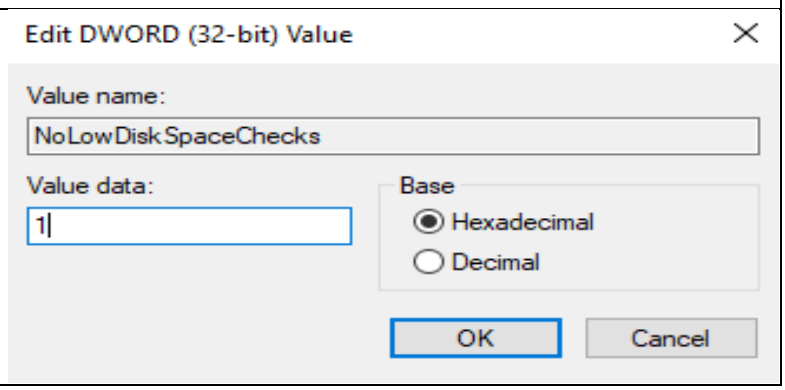
I started by opening Registry Editor as an Administrator on my WIN10 client machine	
I followed the path HKEY_CURRENT_USER > Control Panel > Desktop I then right clicked on MenuShowDelay and selected Modify.	
I changed the delay value from 400 to 300 as instructed to reduce delay.	

Task 9 Learning Outcome:

In Task 9, I learned how to use the Windows Registry Editor to improve system responsiveness by changing the behaviour of UI. In Registry Editor I navigated to HKEY_CURRENT_USER\Control Panel\Desktop and changed the value of "MenuShowDelay" from 400 to 300, The reason I did this was because that figure directly affects the time it takes for menus to appear when hovered over or clicked. By changing the figure from 400 to 300 I reduced the delay, making the UI snappier and more responsive.

This task showed me how even things such as performance settings can be customised through the registry, allowing me to optimize things like responsiveness that usually aren't available through the normal user interface. I also learned that even small changes to registry values take effect immediately and can influence user experience. However, this also made me more aware of the risks of editing the registry as changing things irresponsibly can actually lead to instability and harm the system.

Task 10: Windows Registry

Once again I opened Registry Editor in my WIN10 client. I then followed the path HKEY_CURRENT_USER > SOFTWARE> Microsoft > Windows>CurrentVersion>Policies and clicked New Key and named it Explorer	
I then selected the New Explorer Key and made a new DWORD 32 bit value	
I right clicked the new NoLowDiskSpaceChecks item and clicked modify	
I changed the value data to 1 and clicked Ok.	

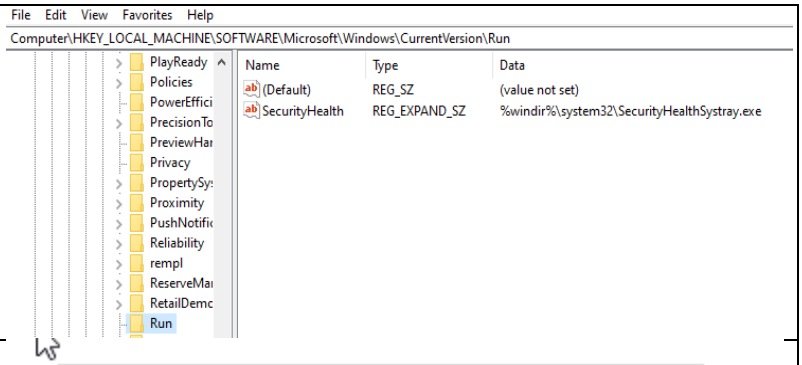
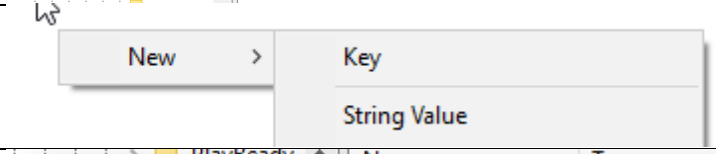
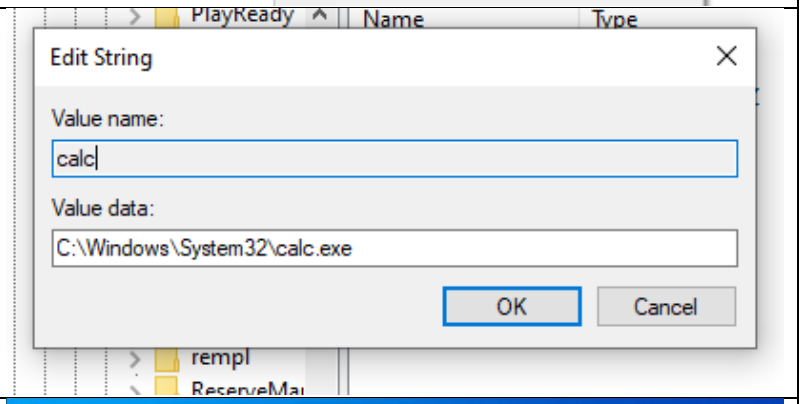
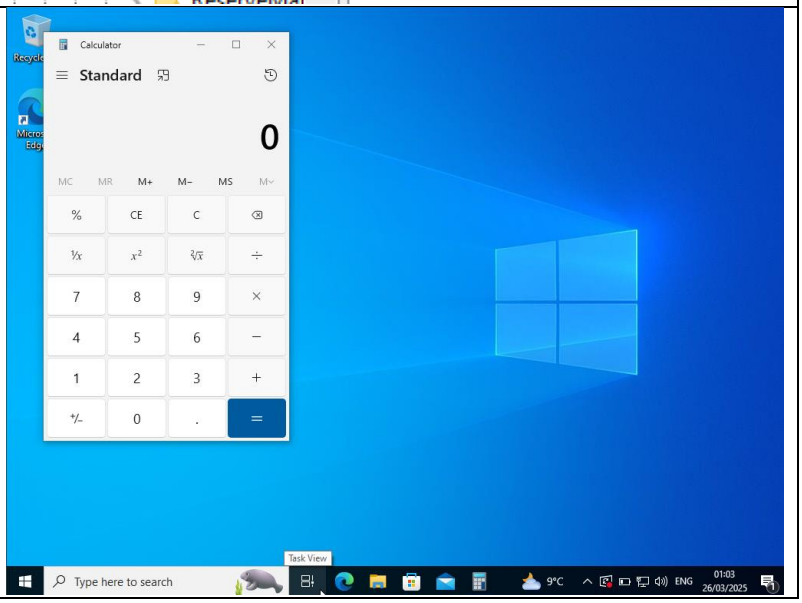
Task 10 Learning Outcome:

In Task 10, I learned how to use the Registry Editor to disable Low Disk space warnings. To start I followed the path HKEY_CURRENT_USER > SOFTWARE> Microsoft > Windows>CurrentVersion>Policies and made a new key named “Explorer”, inside the new “Explorer” key I added a new DWORD (32-Bit) Value called NoLowDiskSpaceChecks. I then modified the value setting it to 1 which disables Windows from checking for low disk space constantly. The reason for this is although low disk space notifications can be helpful they can also end up being inefficient in an environment where users are already aware of how much storage they are using. Having Windows constantly monitor the amount of disk space that is free uses CPU resources which can affect overall system performance so by disabling this feature through the registry I have optimized the system slightly by getting rid of background processes and system interruptions.

I also became more comfortable with creating new keys and DWORD values in the registry. I learned that a key organises settings into different categories and they can contain more keys, subkeys and values. I also learned that a DWORD value is a type of data entry in the registry. DWORD stands for Double Word and stores 32 bit numbers that usually represent if a setting is turned on or off (1 for enabled, 0 for disabled).

By creating the new key called Explorer to group all settings related to it together and added a DWORD value that was set to 1 to disable the warning feature.

Task 11: Windows Registry

For this task I opened Registry Editor in my WIN10 client.	
I right clicked and selected New String Value	
I named the value calc and in the Value Data I gave it the path for calculator (calc.exe) on a windows 10 machine making it run automatically on startup.	
Once I restarted my WIN10 client machine my calculator ran automatically on startup meaning the changes I made in the Registry Editor were successful.	

Task 11 Learning Outcome:

In Task 11, I learned how to make it so a Windows application launches automatically on startup by using the Windows Registry Editor. Once in the Reg editor I navigated to: HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run , I then created a new string value named “calc”. In the value data field I entered then full path to the Calculator app’s executable file (**C:\Windows\System32\calc.exe).**

After this I started up my client machine in order to test these changes I made in the Registry Editor and once I was logged in my Calculator app was opened by default proving that my registry settings had worked. This taught me that the “Run” key is used by Windows to start programs automatically for the user once they have logged in. By adding a string value to this run key I directed it to the calculator app which caused it to run on startup.

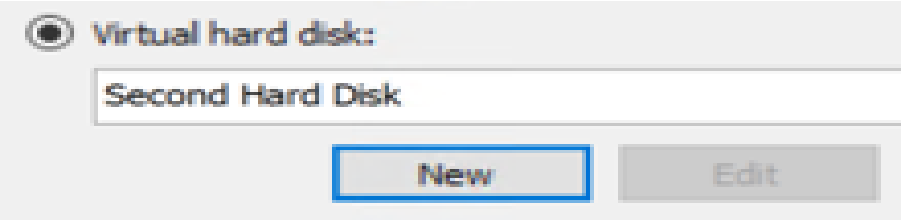
This gave me some experience working with string values in the registry, I learned that unlike DWORDS which store numerical data, string values store text like file paths or command line instructions. Here I learned that the registry can also be used practically to automate system behaviour , showing once again that small registry changes can go a long way when controlling domains and end devices that exist on their networks.

Windows Forensics:

In this part of my report I will be covering the forensics segment of my labs in detail going through what I covered as well as the methods I used to carry out each of the tasks that I needed to do.

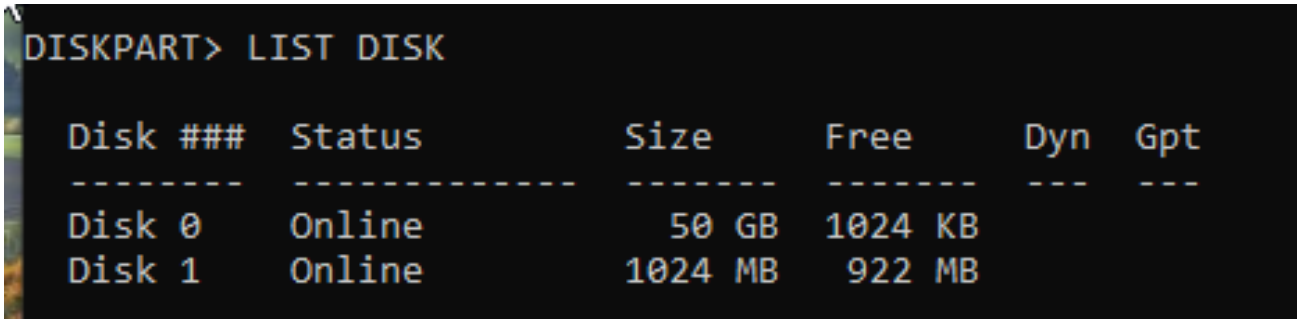
Forensics Part 1:

For part 1 of Windows Forensics I learned how to add and prepare a second virtual hard disk in a Windows 10 VM for NTFS (New Technology File System) level analysis. To start I created a 1 GB virtual hard disk but only formatted 100MB of it with NTFS using 512 byte allocations to match the size of a single sector. This just makes it easier when carrying out forensic analysis, so the mapping is 1 to 1 between sectors and clusters.



After the disk was created and attached to my Windows Virtual Machine I used the "**diskpart**" command in command prompt to clean the new disk just to double check that it is completely empty.

This was done using the command: **LIST DISK**



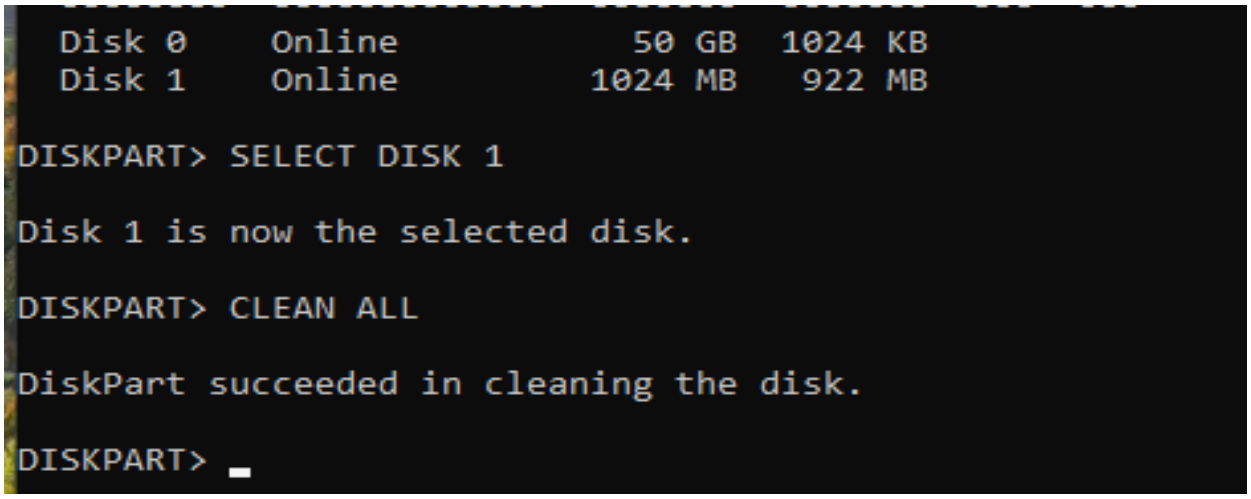
Once I entered this command I was met with 2 disks, one of which was the old disk, the other one being the new disk which I was looking for. In my case DISK 0 was the original disk and Disk 1 was the new Disk, so I followed this up by typing:

SELECT DISK 1

Followed up by

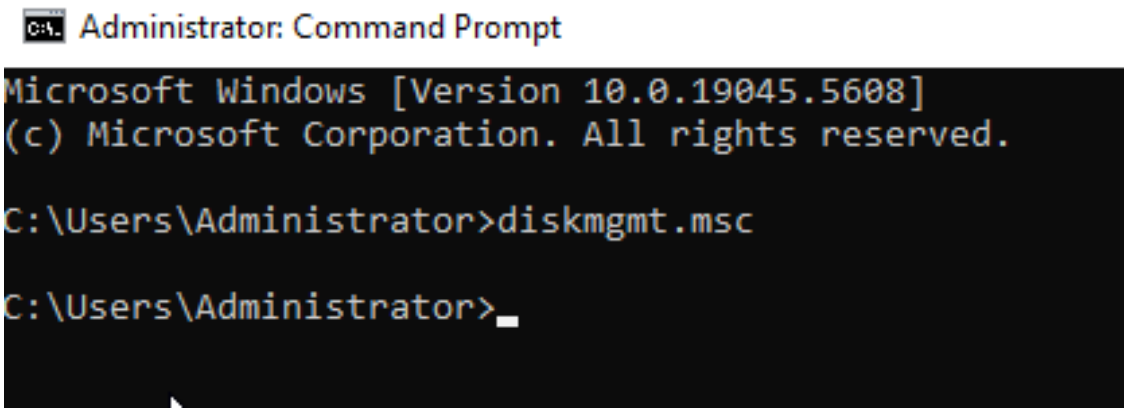
CLEAN ALL

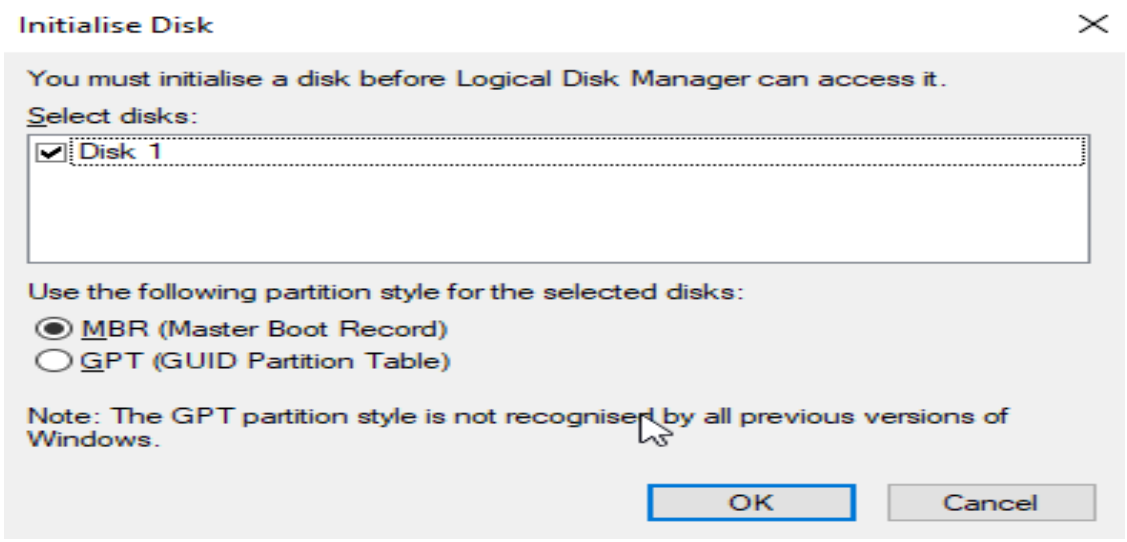
Once this cleaning process was completed I typed Exit as the disk was now definitely cleaned.



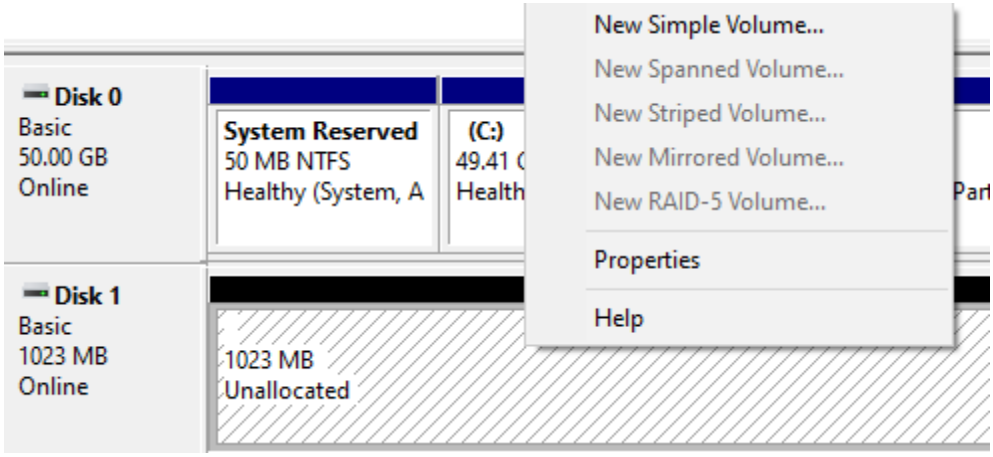
Following this I started up Disk Management by once again opening command prompt and typing:

Diskmgmt.msc

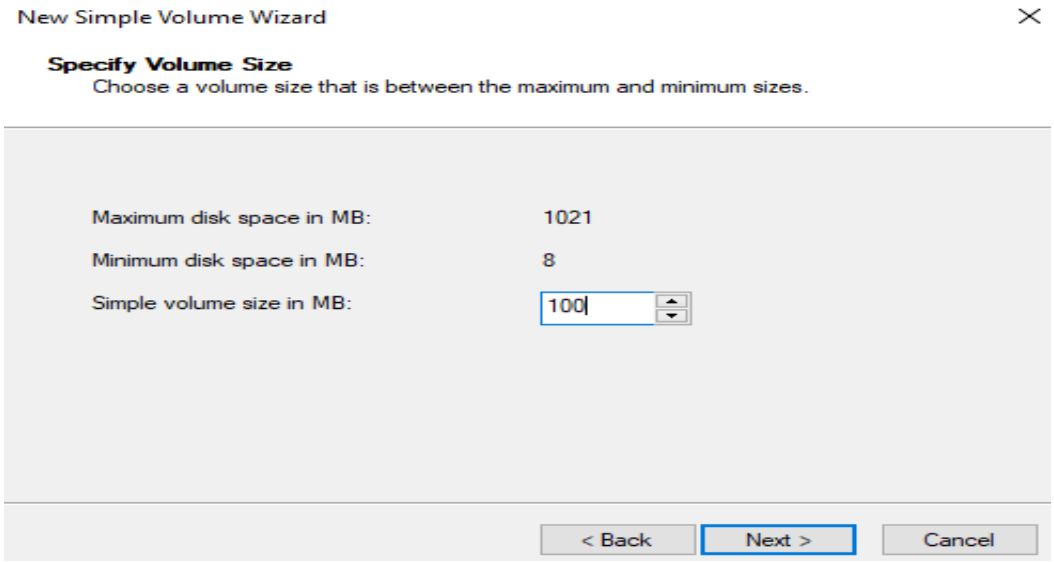




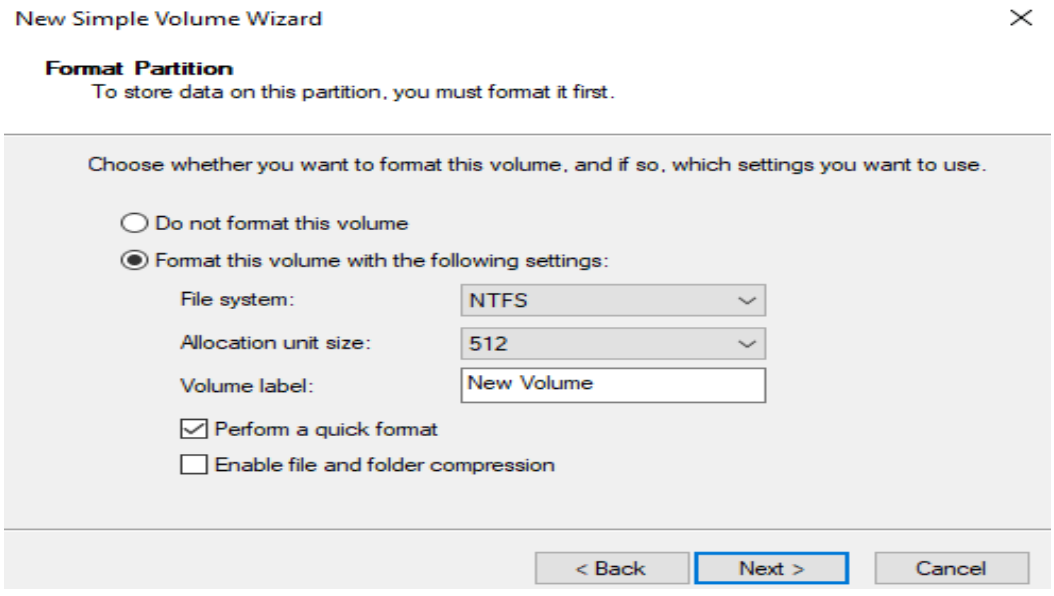
This opened the disk management application where I was met with a "Initialize Disk" box with Disk 1 selected. In the Initialize Disk box I accepted MBR (Master Boot Record) as the default choice and clicked ok, once this was finished I could see my new empty hard disk down the bottom of the screen, here I right clicked on the text "Unallocated" and selected **New Simple Volume**



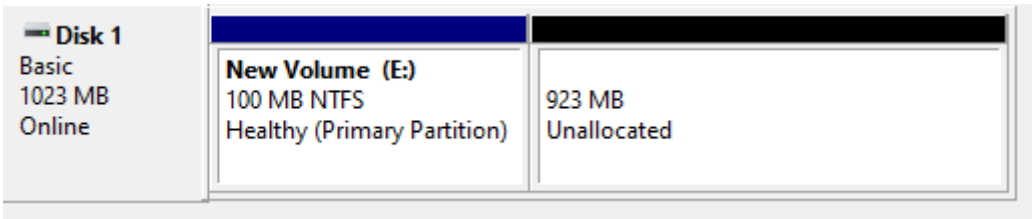
After this as I was in the **Select Volume Size** screen where I set the volume to 100mb,



After this I gave it the drive letter E as this was set by default and carried on to the Format partition section. I made sure the filesystem was set to NTFS and set the allocation unit size to 512



The disk was then formatted successfully along with the new NTFS volume that I had set up on it.

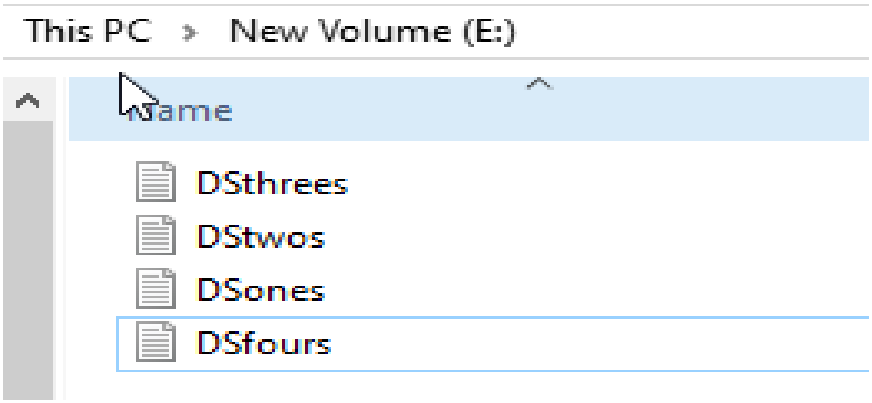


Lastly I opened file explorer and made sure that I could see the new disk as Volume E. Upon testing I could see the new disk under the “E” label telling me that task 1 of forensics has been successful and I now have an NTFS volume with cluster and sector sizes of 512 bytes.

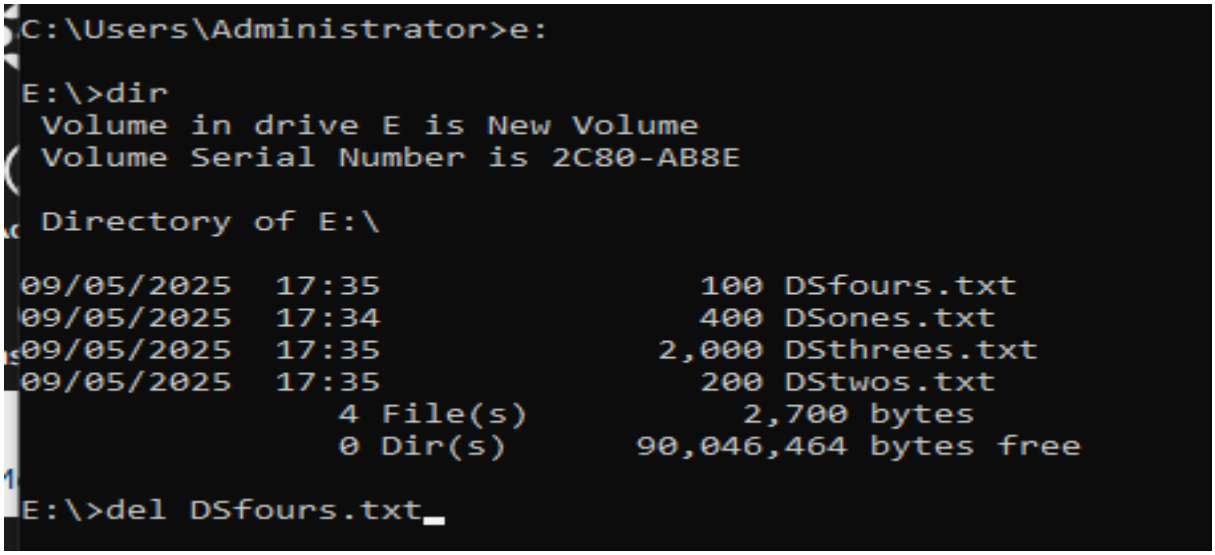
Forensics Part 2: Add 4 Files to New NTFS Volume

For the next part in the forensics segment of my report I started by downloading 4 files from canvas and putting them into my new disk volume. To do this I opened my browser in the Windows 10 client and downloaded the files: **ones.txt, twos.txt , threes.txt** and **fours.txt** on canvas.

Once these were downloaded I moved them to my New Volume and renamed them to include my initials:



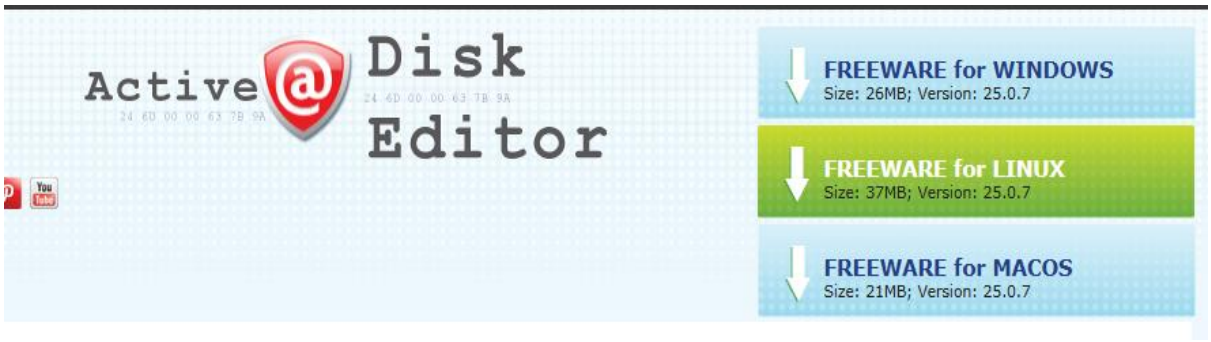
Once these were renamed I navigated to the command prompt and listed my E directory using **dir** in order to confirm each of them existed, After this I deleted DSfours using **del DSfours.txt**



Forensics Part 3: Installing Disk Editor

Next for Forensics part 3 in my Windows 10 VM, I went to my browser and went to **https://www.disk-editor.org/index.html**

In the top right corner of the screen I could see the option “Freeware for Windows”, I clicked on this and started the download

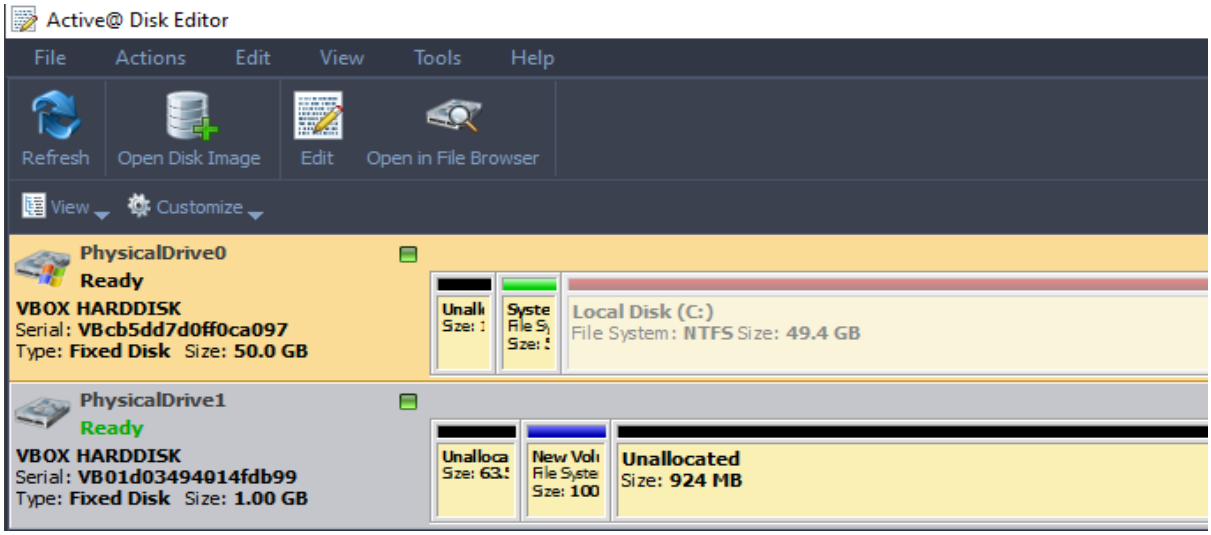


Once the .exe file was downloaded on the VM I double clicked it to run the setup wizard and proceeded with the steps as normal until the Active Disk Editor software was installed.

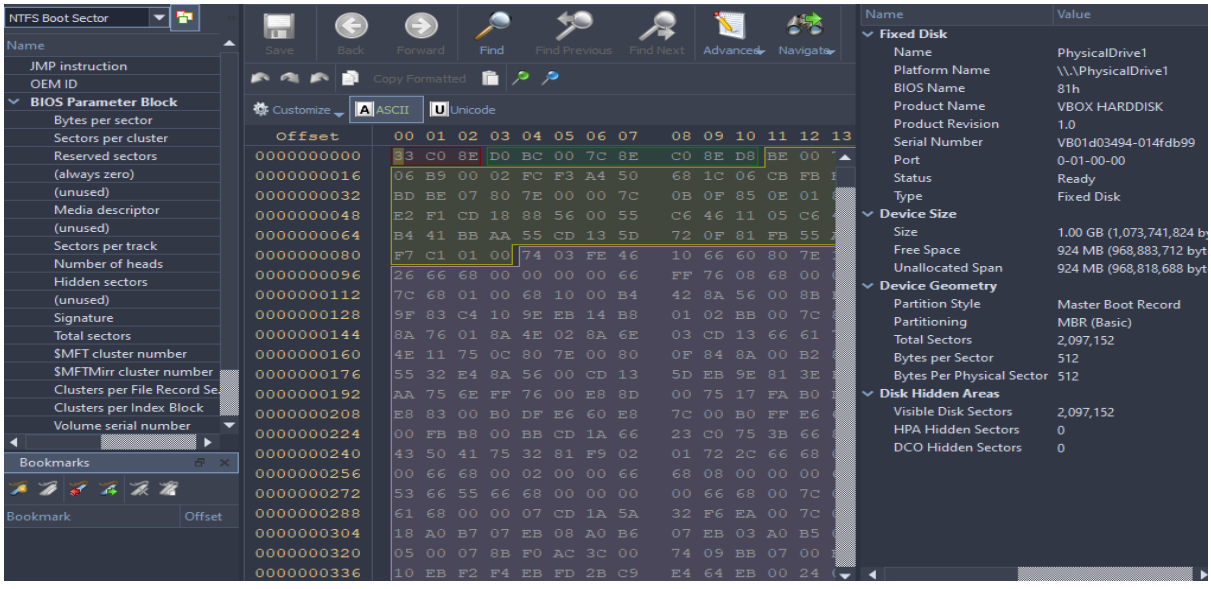
Forensics Part 4: Viewing Boot Sector in Disk Editor

Next I opened Disk editor as an administrator and also added it to the taskbar at the end of my screen as a shortcut for easy future access. In the application the first thing I was met with was a "Getting started" page where I selected **Open Disk**

Here I selected open PhysicalDrive1 which I made earlier



After opening the disk I navigated to the NTFS boot sector by changing the field in the top left menu from "Master Boot Record" to **NTFS Boot Sector**



Comparison with Lecture Notes:

When comparing my results in Disk editor to the bott sector shown in my lecture notes I noticed many similarities such as:

- The OEM ID appeared at an offset of 0x03 in both showing NTFS is the file system
- The bytes per sector in both were set to 0200 which is 512 in decimal , this is the same as my set up in my new volume

This confirms that my NTFS volume is set up properly and matches the layout shown in my lectures.

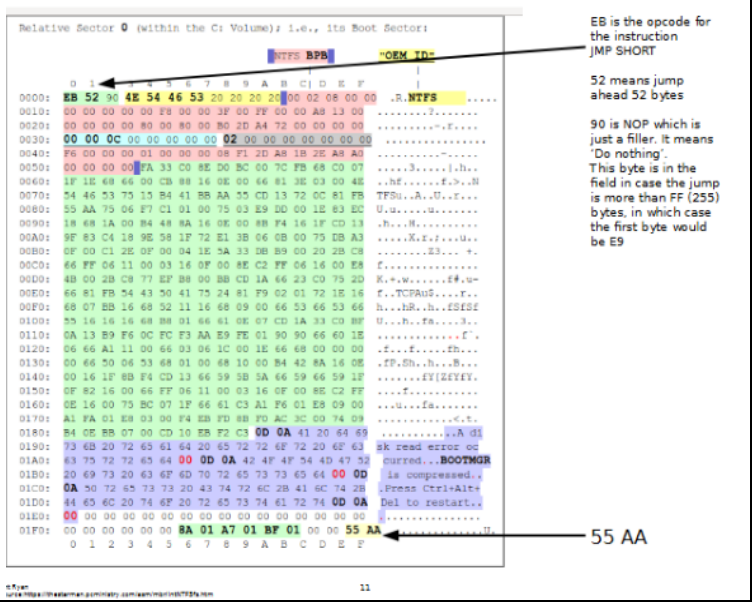
Boot Sector Layout

NTFS Reference Sheet

NTFS Boot sector

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	Jump Instruction			OEM ID								Bytes/ Sector		Sect/ clust	res	
10	0x000000			unused		Media desc	0x0000		Sect / track		Number heads		Hidden Sectors			
20	unused								Total Sectors							
30	Logical Cluster of $MFT								Logical Cluster of $MFTMirr							
40	Clust / File record segment			Clusters / Index Block					Volume Serial Number							
50	Checksum								Boot Code							
60	Boot Code															
1E0	Boot Code															
1F0	Boot Code													55	AA	

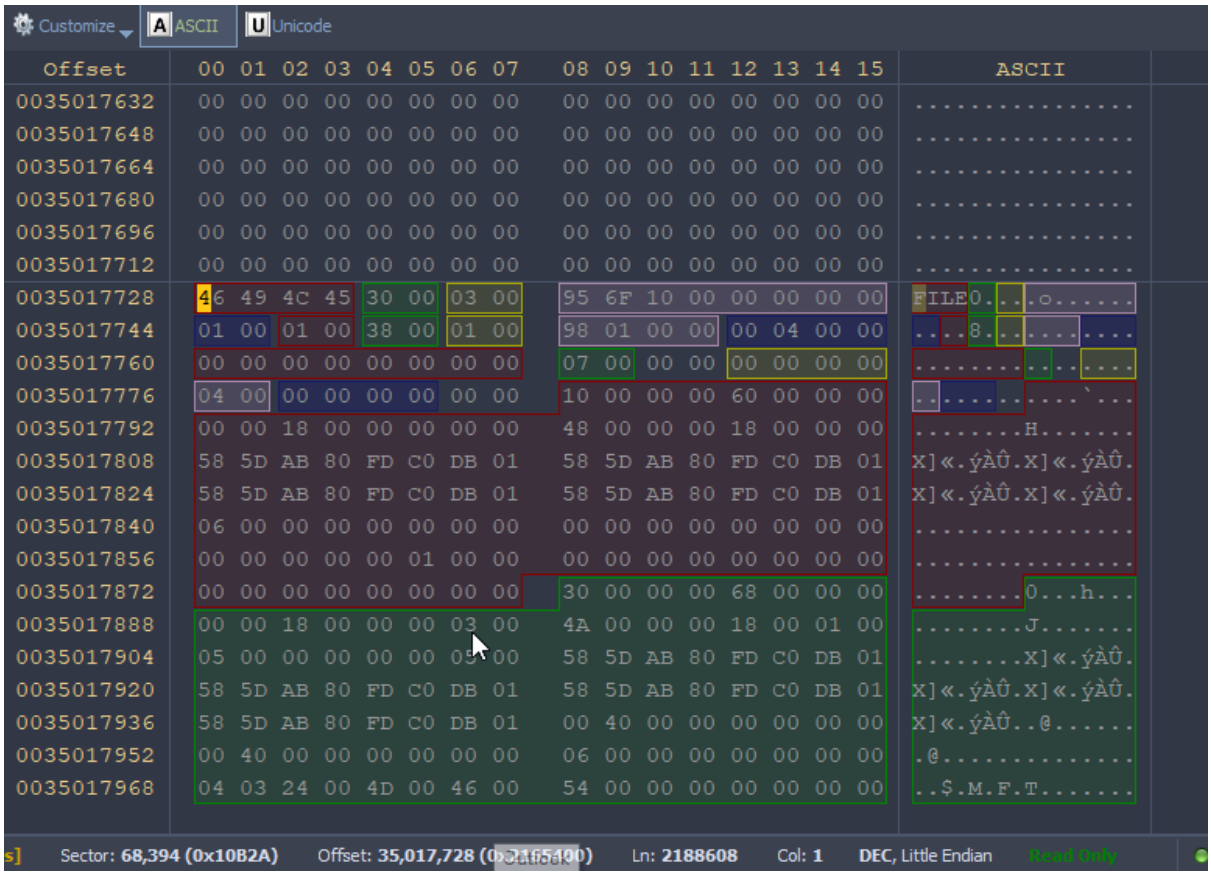
Notes : On NTFS volumes, the MFT is not located in a predefined sector (as it is on FAT16 and FAT32 volumes). For this reason, the MFT can be moved if say there is a bad sector in its normal location.
30H (30 Hex) gives the start location of the MFT and 38H gives the start of MFTMirr. The last two bytes of any Boot Sector are always 55 AA (also written as the hex number: 0xAA55)



Forensics Part 5: Viewing MFT Records

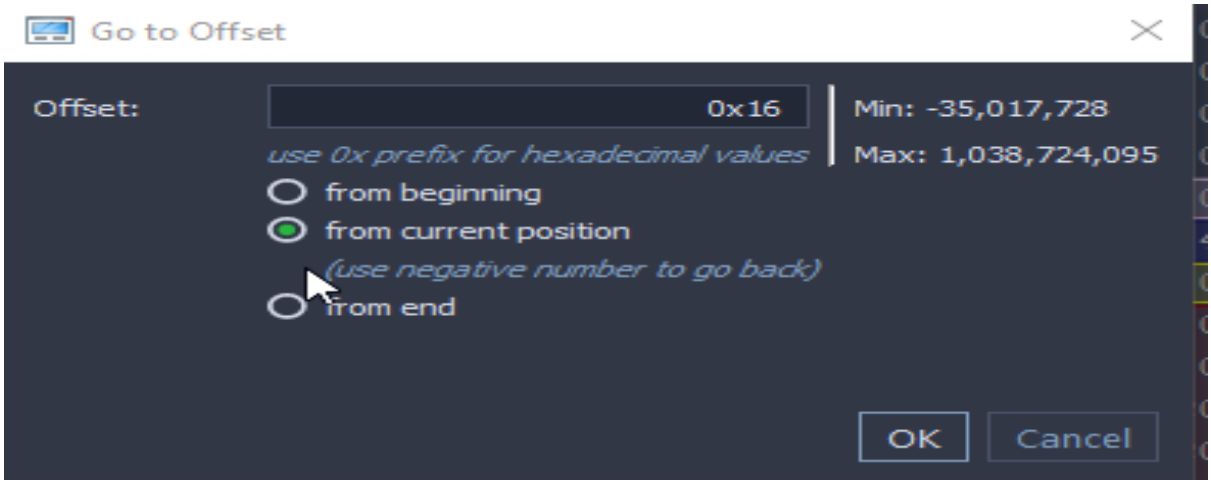
Next I navigated to the File table (MFT) in disk editor using the menu on the left and selecting "NTFS MFT File Record"

I used the navigate menu and went to Navigate > Primary NTFS > $MFT

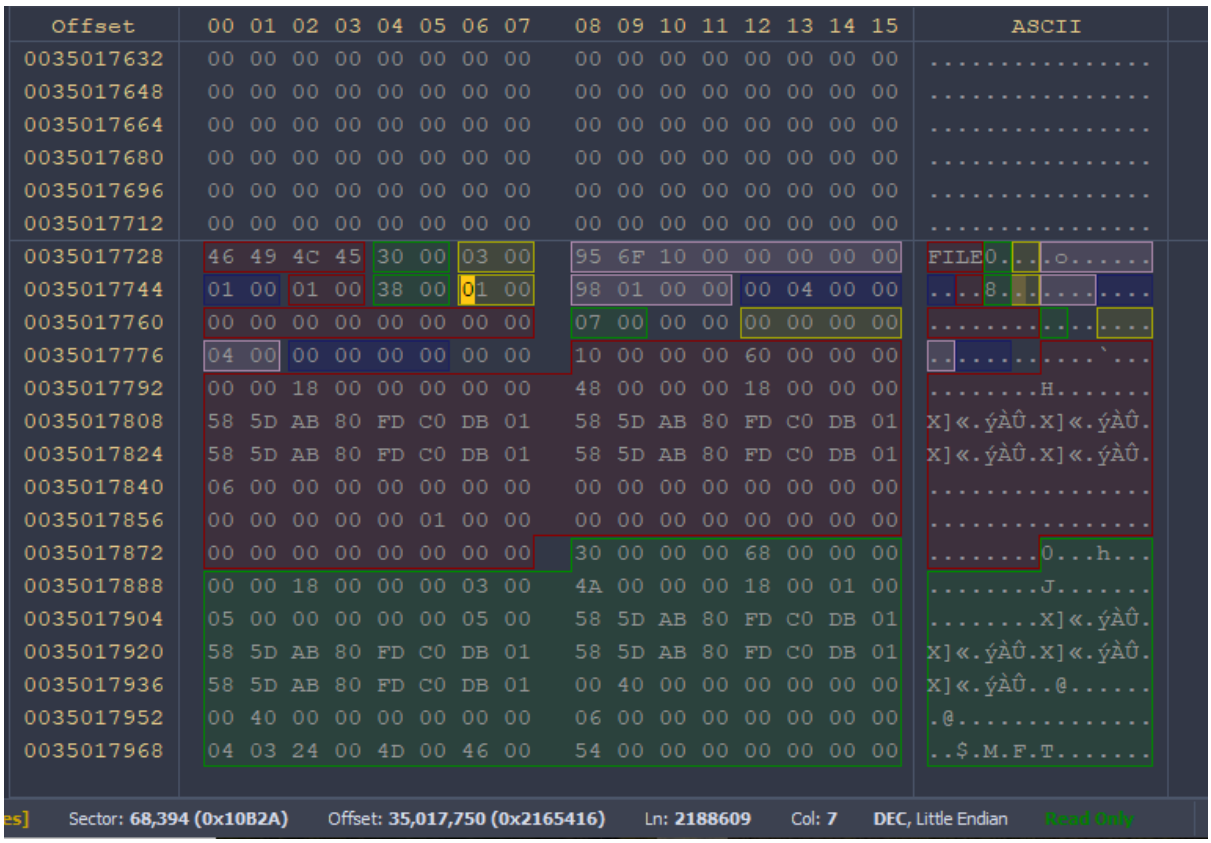


This brought me to the start of the MFT area where I could see the header for one of the MFT entries

I then used navigate> go to offset> from current position with an offset of 0x16



This took me to the offset of 0x16 where I was shown the 2 bytes 0x16 (01) and 0x17 (00) giving me the flag value of 0x0001 which based on my MFT entry header slide in my lecture notes means that the file is in use and is a regular file not a directory.



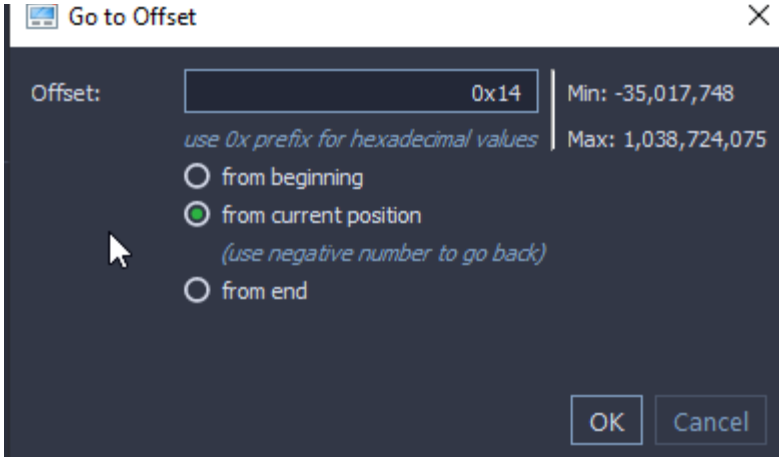
Offset	00 01 02 03 04 05 06 07	08 09 10 11 12 13 14 15	ASCII
0035017632	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....
0035017648	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....
0035017664	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....
0035017680	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....
0035017696	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....
0035017712	00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....
0035017728	46 49 4C 45 30 00 03 00	95 6F 10 00 00 00 00 00	FILE0...o.....
0035017744	01 00 01 00 38 00 01 00	98 01 00 00 00 04 00 00	....8.......
0035017760	00 00 00 00 00 00 00 00	07 00 00 00 00 00 00 00	.....
0035017776	04 00 00 00 00 00 00 00	10 00 00 00 60 00 00 00	....`...
0035017792	00 00 18 00 00 00 00 00	48 00 00 00 18 00 00 00	....H...
0035017808	58 5D AB 80 FD C0 DB 01	58 5D AB 80 FD C0 DB 01	X]«.ýÀÛ.X]«.ýÀÛ.
0035017824	58 5D AB 80 FD C0 DB 01	58 5D AB 80 FD C0 DB 01	X]«.ýÀÛ.X]«.ýÀÛ.
0035017840	06 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00	.....
0035017856	00 00 00 00 00 01 00 00	00 00 00 00 00 00 00 00	.....
0035017872	00 00 00 00 00 00 00 00	30 00 00 00 68 00 00 00	....0...h...
0035017888	00 00 18 00 00 00 03 00	4A 00 00 00 18 00 01 00	....J.......
0035017904	05 00 00 00 00 00 05 00	58 5D AB 80 FD C0 DB 01	....X]«.ýÀÛ.
0035017920	58 5D AB 80 FD C0 DB 01	58 5D AB 80 FD C0 DB 01	X]«.ýÀÛ.X]«.ýÀÛ.
0035017936	58 5D AB 80 FD C0 DB 01	00 40 00 00 00 00 00 00	X]«.ýÀÛ..@.....
0035017952	00 40 00 00 00 00 00 00	06 00 00 00 00 00 00 00	.@.....
0035017968	04 03 24 00 4D 00 46 00	54 00 00 00 00 00 00 00	..$.M.F.T.....

es] Sector: **68,394 (0x10B2A)** Offset: **35,017,750 (0x2165416)** Ln: **2188609** Col: **7** **DEC**, Little Endian Read Only

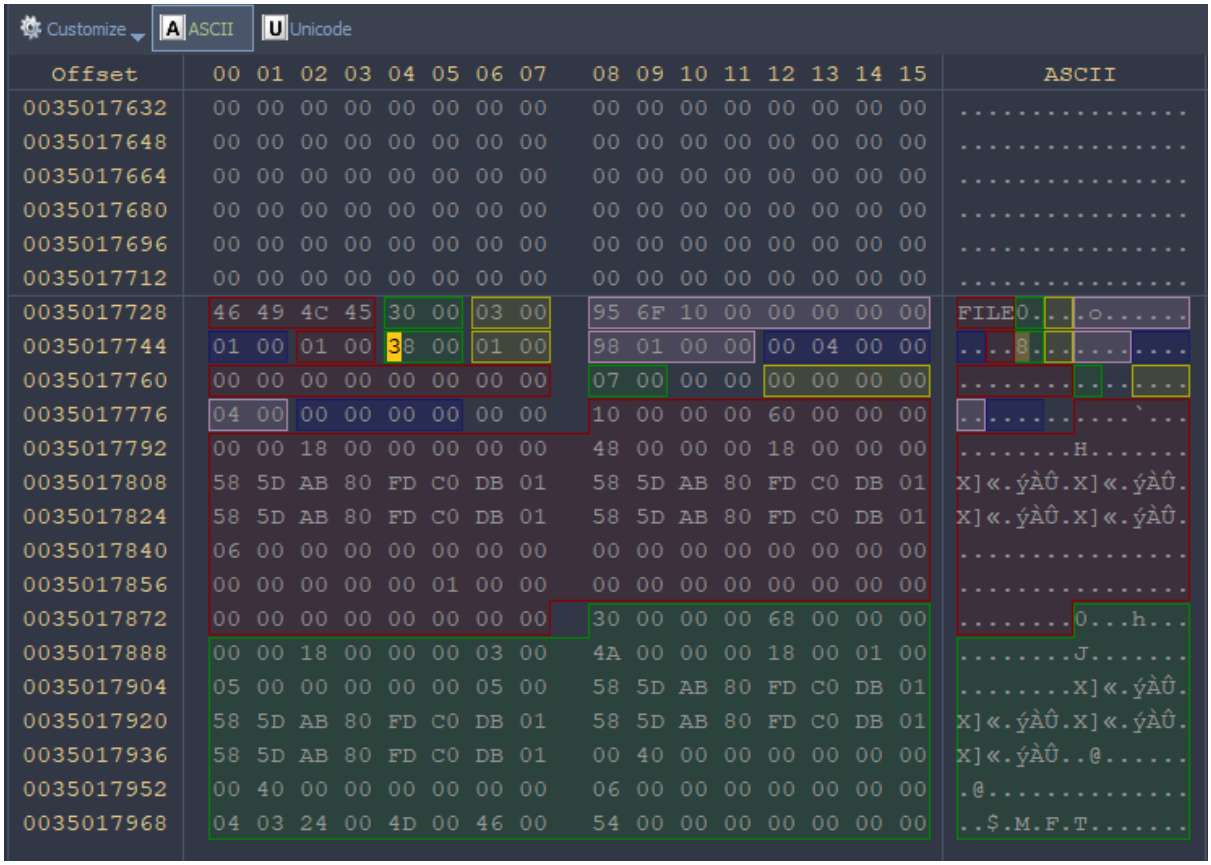
What this tells me is that the file in 0x17 had been deleted with it's flag being changed to 00.

Forensics Part 6: Viewing MFT Records Part Two

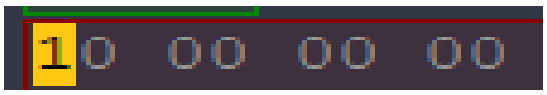
To start Part 6 I clicked the back button to return to the beginning of the MFT file table. From here I once again went to the navigate menu and entered an offset of 0x14 from my current position.



Here at offset 0x14 and 0x15 I was met with **38 00** which converted to little endian = **0x0038 = 56** in decimal. This tells me that the first attribute begins at byte 56

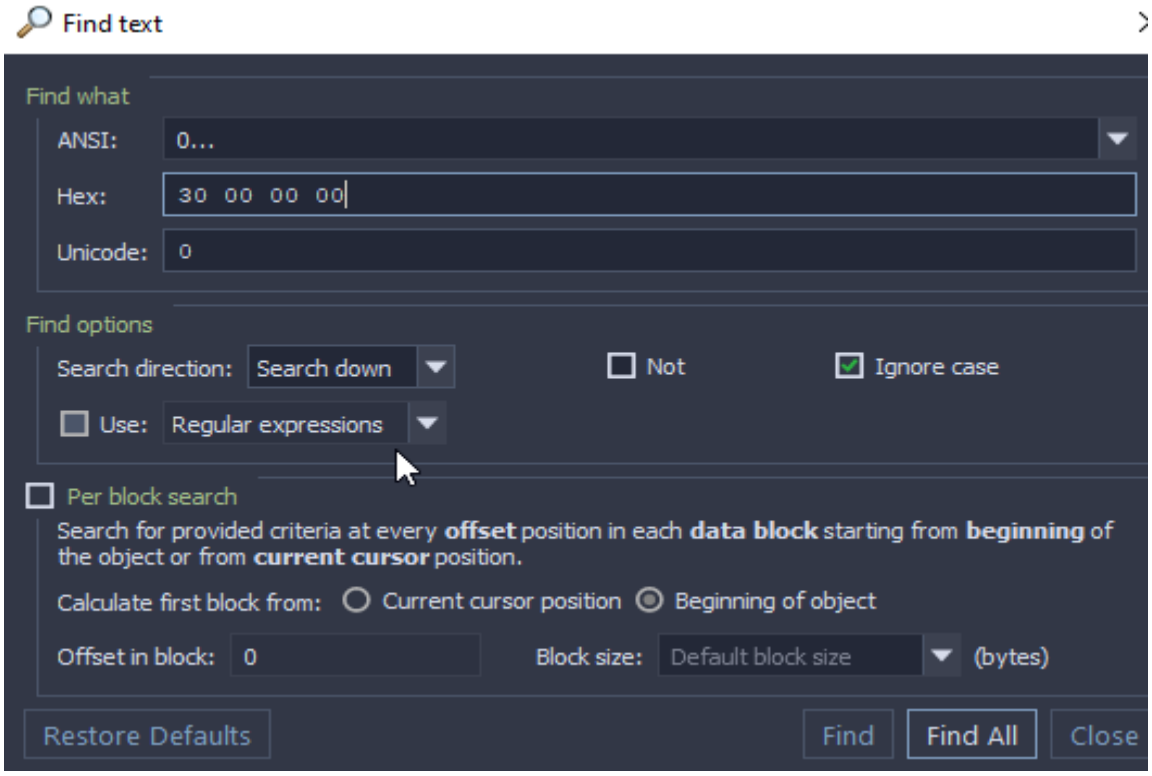


Taking this further at offset 0x0038 from the MFT entry's start I have: **10 00 00 00** which is the standard information attribute:



Further down I then noticed the file name attribute beginning with: **30 00 00 00**

I will now find this using disk editor using find and entering “30 00 00 00” under hex.



Lecture Notes:

$FileName attribute (1)

The first 4 bytes of the $FileName attribute header are

30 00 00 00 (Ox00000030)

It is a resident attribute, and is not named, so the length of the header for this attribute is 0x18 or 24.

Vincent Ryan 48

$FileName attribute (2)

$FILE_NAME		
Offset	**Size**	**Description**
0x00	8	File Reference to parent directory
0x08	8	File creation time.
0x10	8	File modification time
0x18	8	MFT modification time
0x20	8	File access time.
0x28	8	Allocated size of file
0x30	8	Real size of file
0x38	4	Flags
0x3c	4	Used by EAs and Reparse
0x34	4	Security Id
0x40	1	Filename length in unicode characters
0x41	1	Filename namespace
0x42		File Name in Unicode

Table 7: Layout of the $FILE_NAME (0x30) Attribute

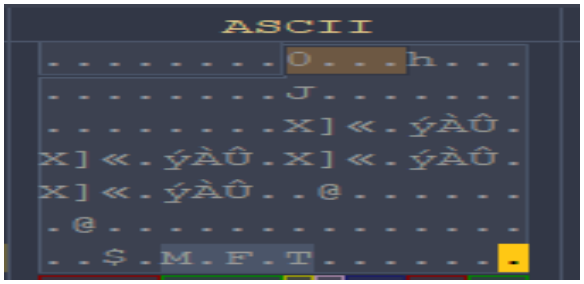
The time values are given in units of 100 nanoseconds since January 1, 1601, UTC

Source: http://www.cse.scu.edu/~tschwarz/coen252_07Fall/Lectures/NTFS.html

Vincent Ryan 49

My notes state that this attribute is resident meaning all the file information including the name is stored directly in the MFT record. Because it's a standard attribute the header is 0x18 (24 bytes) long, Following the notes I moved 24 bytes forward from the start of 30 00 00 00 and at that position there was a file name stored in Unicode visible in the ASCII column as "$MFT"

ASCII MFT Text:

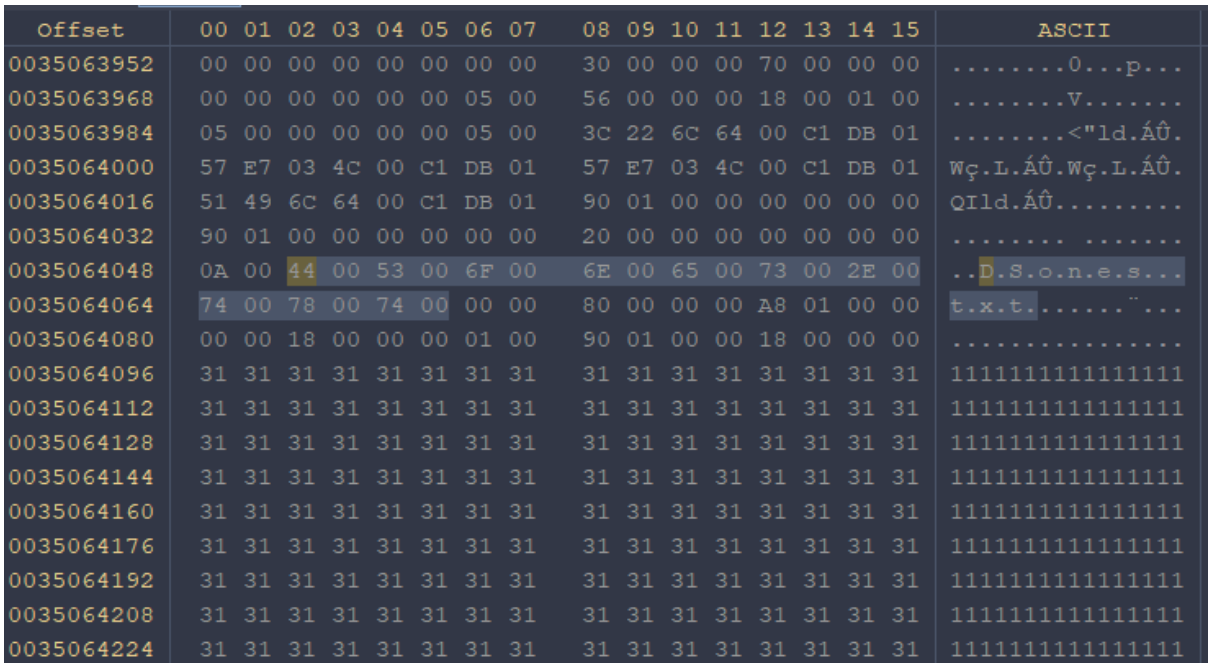


This helped show me that NTFS actually embeds filename metadata inside of MFT entries and that attributes can be found using hex and offset calculations.

Forensics Part 7: Viewing MFT Records Part Three:

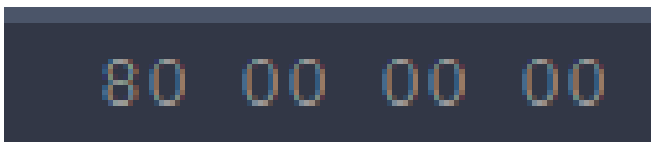
Once again using the “Find” tool in Disk editor I used Unicode to find my ones file (DSones.txt)

This brought me to the FILE_NAME attribute for the MFT entry of my DSones file. Next I would check if this was a resident file.



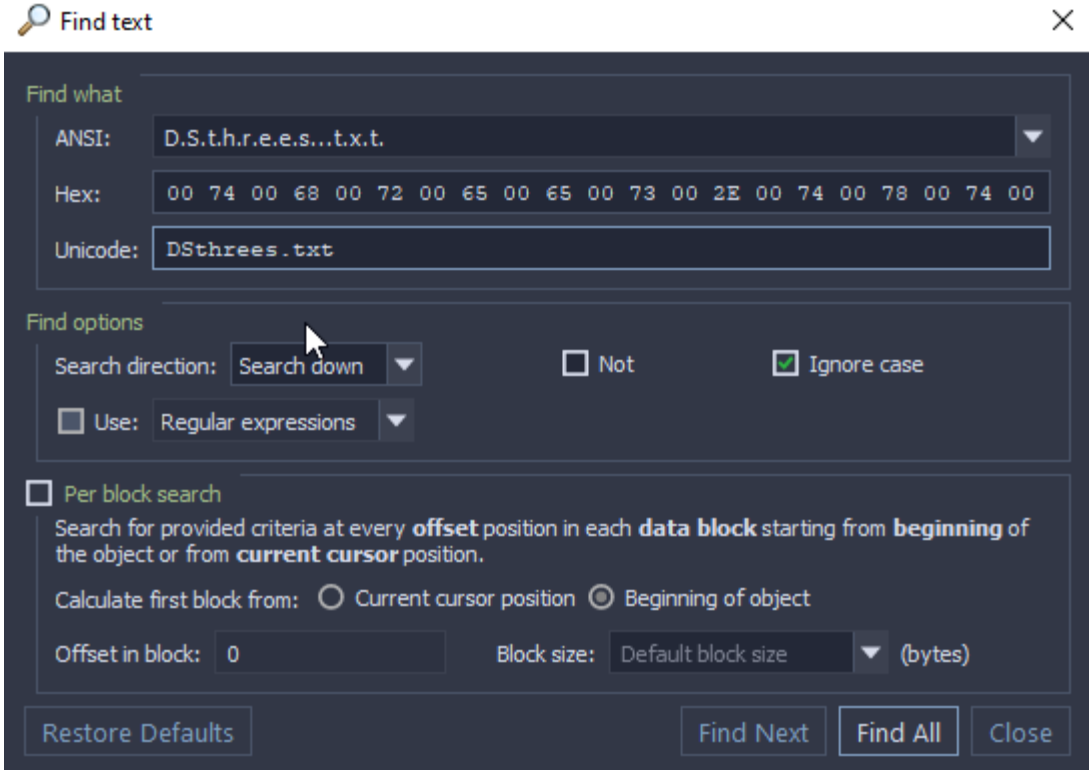
Offset	00 01 02 03 04 05 06 07	08 09 10 11 12 13 14 15	ASCII
0035063952	00 00 00 00 00 00 00 00	30 00 00 00 70 00 00 00	........0...p...
0035063968	00 00 00 00 00 00 05 00	56 00 00 00 18 00 01 00	........V.......
0035063984	05 00 00 00 00 00 05 00	3C 22 6C 64 00 C1 DB 01	........<"ld.ÁÛ.
0035064000	57 E7 03 4C 00 C1 DB 01	57 E7 03 4C 00 C1 DB 01	Wç.L.ÁÛ.Wç.L.ÁÛ.
0035064016	51 49 6C 64 00 C1 DB 01	90 01 00 00 00 00 00 00	QIld.ÁÛ........
0035064032	90 01 00 00 00 00 00 00	20 00 00 00 00 00 00 00	....
0035064048	0A 00 44 00 53 00 6F 00	6E 00 65 00 73 00 2E 00	..D.S.o.n.e.s...
0035064064	74 00 78 00 74 00 00 00	80 00 00 00 A8 01 00 00	t.x.t.....¨...
0035064080	00 00 18 00 00 00 01 00	90 01 00 00 18 00 00 00	....
0035064096	31 31 31 31 31 31 31 31	31 31 31 31 31 31 31 31	1111111111111111
0035064112	31 31 31 31 31 31 31 31	31 31 31 31 31 31 31 31	1111111111111111
0035064128	31 31 31 31 31 31 31 31	31 31 31 31 31 31 31 31	1111111111111111
0035064144	31 31 31 31 31 31 31 31	31 31 31 31 31 31 31 31	1111111111111111
0035064160	31 31 31 31 31 31 31 31	31 31 31 31 31 31 31 31	1111111111111111
0035064176	31 31 31 31 31 31 31 31	31 31 31 31 31 31 31 31	1111111111111111
0035064192	31 31 31 31 31 31 31 31	31 31 31 31 31 31 31 31	1111111111111111
0035064208	31 31 31 31 31 31 31 31	31 31 31 31 31 31 31 31	1111111111111111
0035064224	31 31 31 31 31 31 31 31	31 31 31 31 31 31 31 31	1111111111111111

When looking down I could see where the DATA attribute begun and once I looked through my slides I noticed that that there was a resident flag present (8) in 80 00 00 00 . I went to byte 8 of the DATA attribute and realized that it was 00 meaning the DATA attribute is in fact resident as if it was not resident I would be unable to see the data because it would be located somewhere else on the disk.



Forensics Part 8: Viewing MFT Records Part Four:

Next I will be finding the MFT entry for my DSthrees.txt file in order to figure out it's disk size and whether or not it is resident or non-resident. To start I opened the find tool and searched for my file "DSthrees.txt" under the Unicode heading.



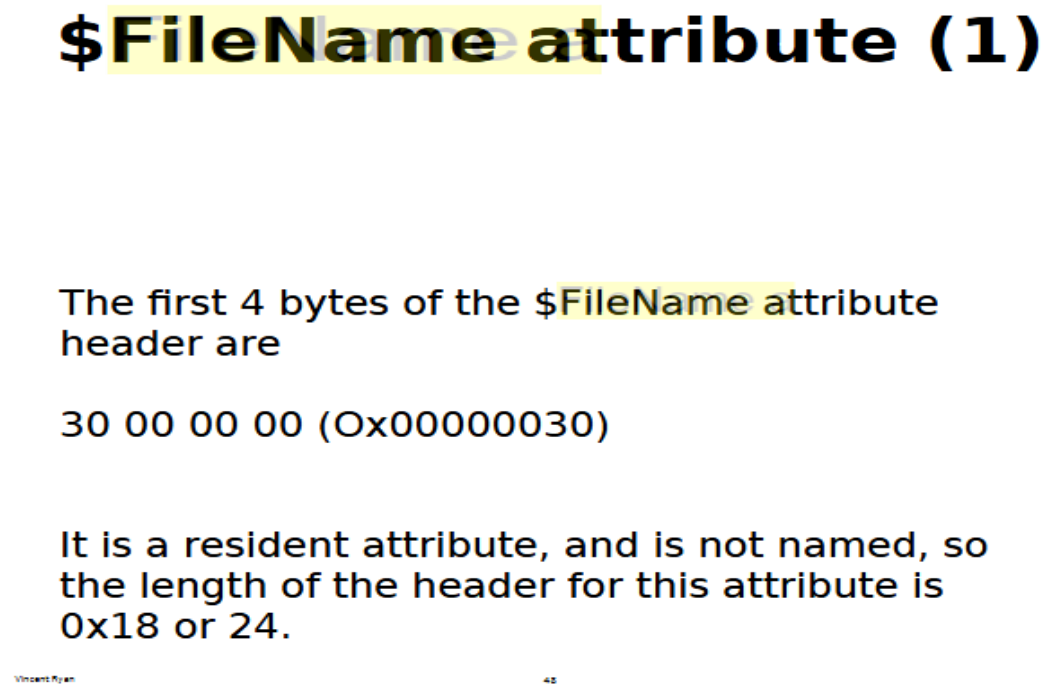
This let me locate my file on the disk

Offset	00 01 02 03 04 05 06 07	08 09 10 11 12 13 14 15	ASCII
0000212816	06 00 00 10 00 00 00 00	01 03 2E 00 00 00 00 00	.....
0000212832	2D 00 00 00 00 00 01 00	68 00 56 00 00 00 00 00	-.....h.V.....
0000212848	05 00 00 00 00 00 05 00	3C 22 6C 64 00 C1 DB 01	.....<"ld.ÁÛ.
0000212864	57 E7 03 4C 00 C1 DB 01	A0 66 9D 97 00 C1 DB 01	Wç.L.ÁÛ. f...ÁÛ.
0000212880	F9 F3 6F 64 00 C1 DB 01	90 01 00 00 00 00 00 00	ùódo.ÁÛ.....
0000212896	90 01 00 00 00 00 00 00	20 00 00 00 00 00 00 00	....
0000212912	0A 00 44 00 53 00 6F 00	6E 00 65 00 73 00 2E 00	..D.S.o.n.e.s...
0000212928	74 00 78 00 74 00 75 00	2B 00 00 00 00 00 01 00	t.x.t.u.+.....
0000212944	70 00 5A 00 00 00 00 00	05 00 00 00 00 00 05 00	p.Z.....
0000212960	F7 C0 65 64 00 C1 DB 01	6E 82 BB 55 00 C1 DB 01	÷Àed.ÁÛ.n.»U.ÁÛ.
0000212976	02 F5 18 A0 00 C1 DB 01	27 7C 69 64 00 C1 14 00	.õ. .ÁÛ.'|id.Á..
0000212992	00 08 00 00 00 00 00 00	D0 07 00 00 00 00 00 00	.....Ð.....
0000213008	20 00 00 00 00 00 00 00	0C 00 44 00 53 00 74 00	.....D.S.t.
0000213024	68 00 72 00 65 00 65 00	73 00 2E 00 74 00 78 00	h.r.e.e.s...t.x.
0000213040	74 00 00 00 00 00 01 00	2C 00 00 00 00 00 01 00	t.....,.....
0000213056	68 00 56 00 00 00 00 00	05 00 00 00 00 00 05 00	h.V.....
0000213072	FA 2D 69 64 00 C1 DB 01	0F EF C5 4F 00 C1 DB 01	ú-id.ÁÛ..ïÅO.ÁÛ.
0000213088	BF EA 1F 9B 00 C1 DB 01	98 F2 6C 64 00 C1 DB 01	¿ê...ÁÛ..òld.ÁÛ.
0000213104	C8 00 00 00 00 00 00 00	C8 00 00 00 00 00 00 00	È.......È.....
0000213120	20 00 00 00 00 00 00 00	0A 00 44 00 53 00 74 00	.....D.S.t.
0000213136	77 00 6F 00 73 00 2E 00	74 00 78 00 74 00 75 00	w.o.s...t.x.t.u.
0000213152	24 00 00 00 00 00 01 00	88 00 74 00 00 00 00 00	$.....t.....

I then saw the bytes 20 00 00 00 00 00 00 00 located at 0x28 bytes from the start of the FILE_NAME attribute beginning at 30 00 00 00 , this basically represents the allocated file size. I translated this to little endian 0x0000000000000020 = 32 bytes (0 x 20 in decimal is 32)

```
20 00 00 00 00 00 00 00
```

This is following the logic of my FileName attribute slide

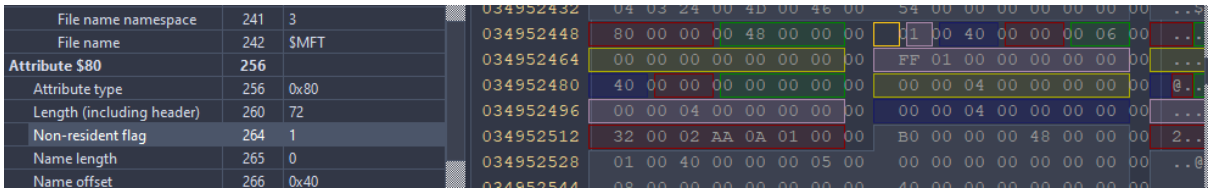


The filename is resident.

Forensics Part 9: Non Resident File:

To confirm that DSthrees.txt is a non-resident file I inspected it's MFT entry in Disk editor. I searched again for the file using Unicode and located the $DATA attribute that began with 80 00 00 00

According to my notes the Non Resident Flag is stored with 8 bytes after the start of this attribute, So I went to the offset 0x08 from the start of 80 00 00 00 block and found the value: 01

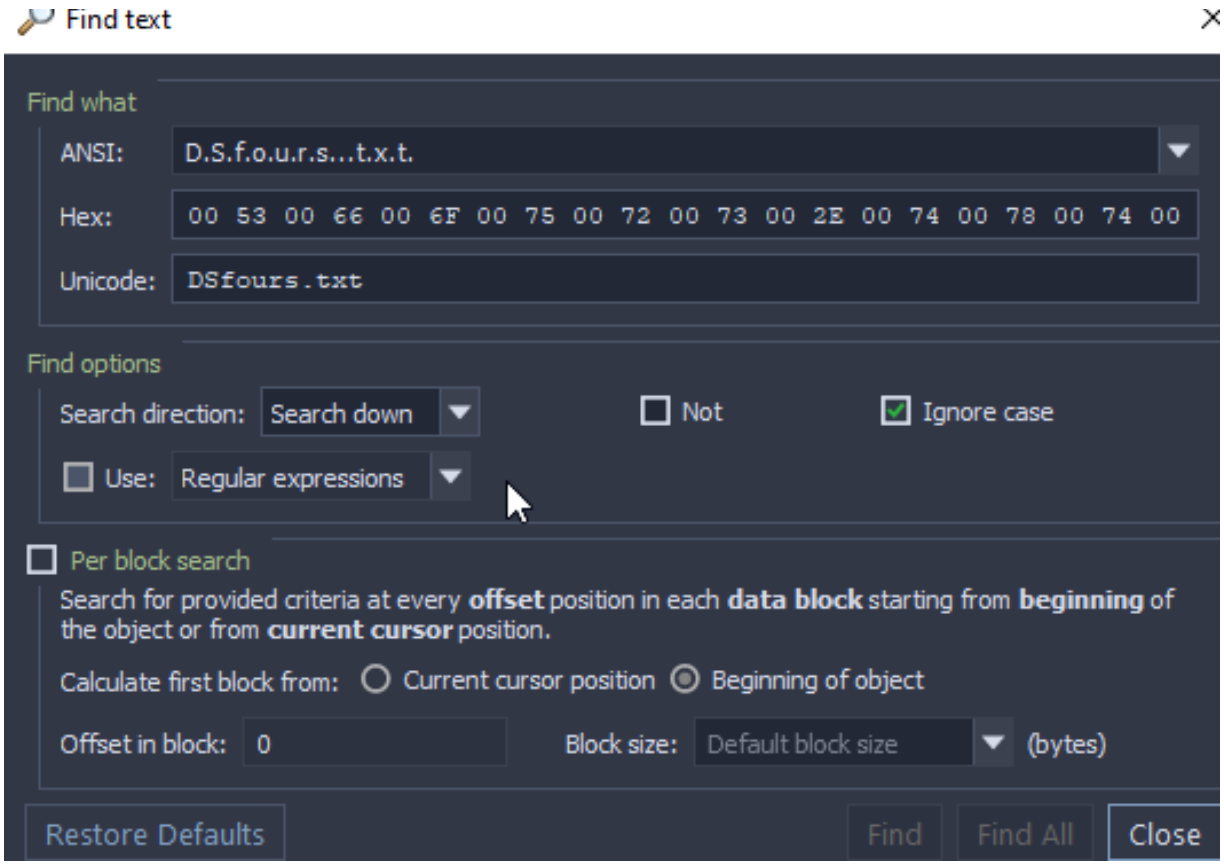


Non – resident flag 1 = yes its non-resident.

This means the flag is set to 1 , which tells me that the file is non-resident and it's data is not stored inside the MFT entry but somewhere else on the disk instead. This also matches what I found in part 8 where I calculated that the file used 32 bytes of disk space outside the MFT record. This confirms to me that DSthrees.txt is non-resident.

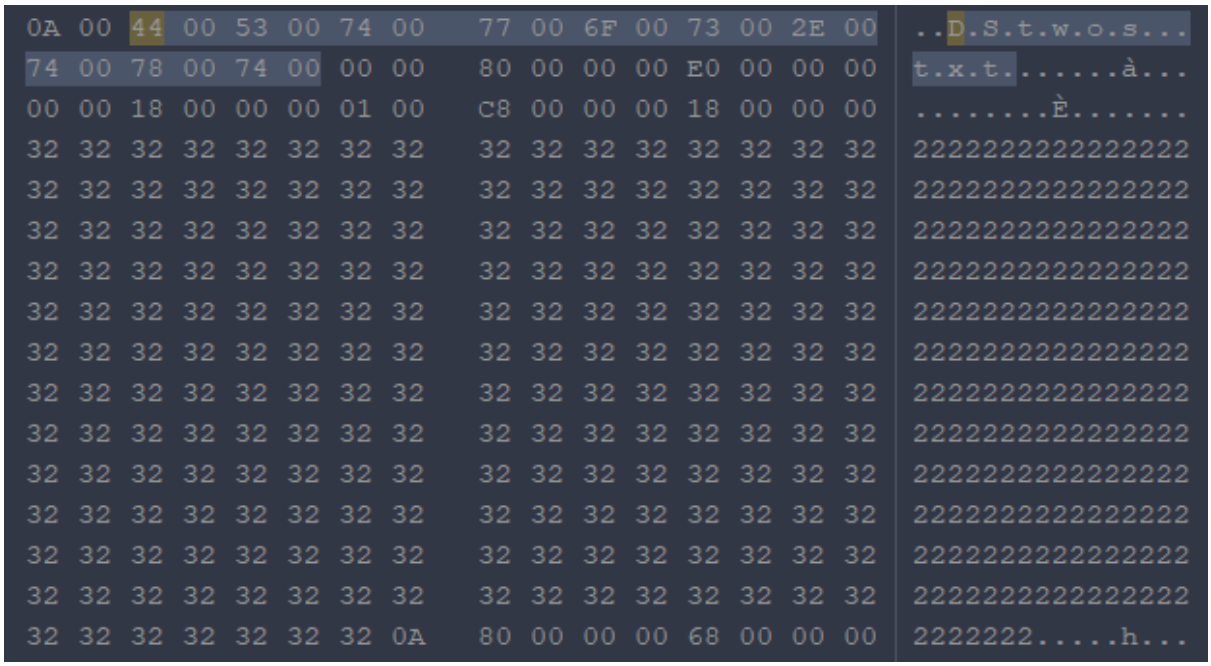
Forensics Part 10: Find the Deleted File:

In part 10 I searched for the deleted file DSfours.txt using the Unicode find tool, I was unable to find evidence of the deleted file on my Disk.

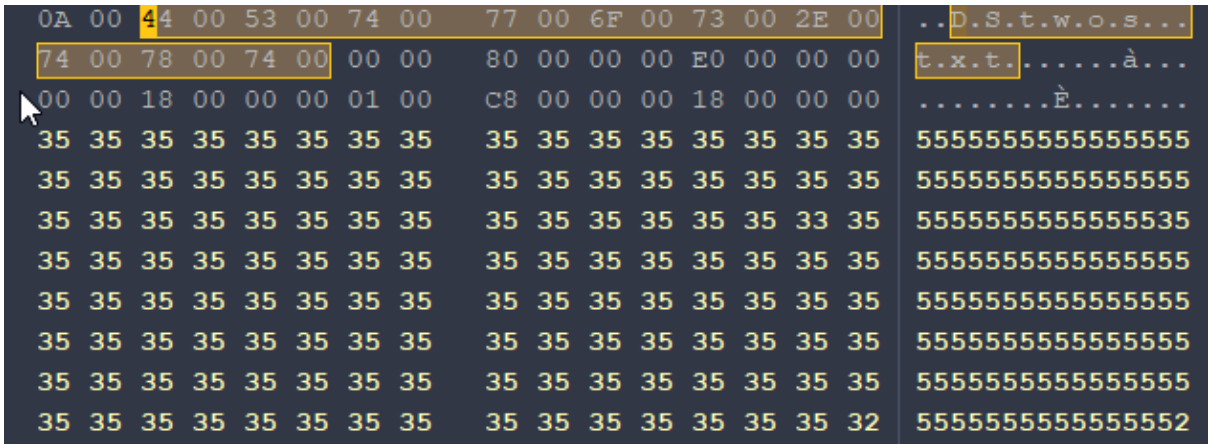


Forensics Part 11: Editing DStwos.txt Content:

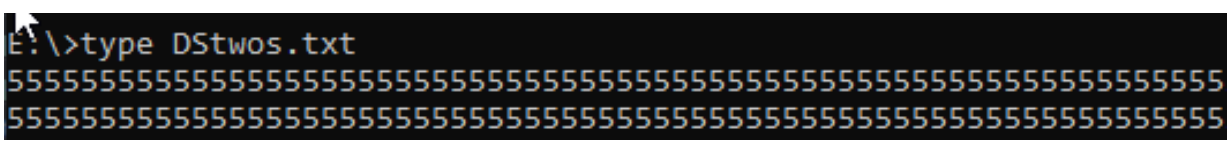
In this part I edited the contents of DStwos.txt using the Unicode search in disk editor to find it first. Once I was sure I was viewing the file's data attribute I identified the ASCII content of the file as a sequence of 0x32 bytes which directly translate to the character "2"



To edit the file, I selected Edit > Allow edit content in disk editor and changed some of the hex values from 32 to 35 changing the character in the contents from 2 to 5.

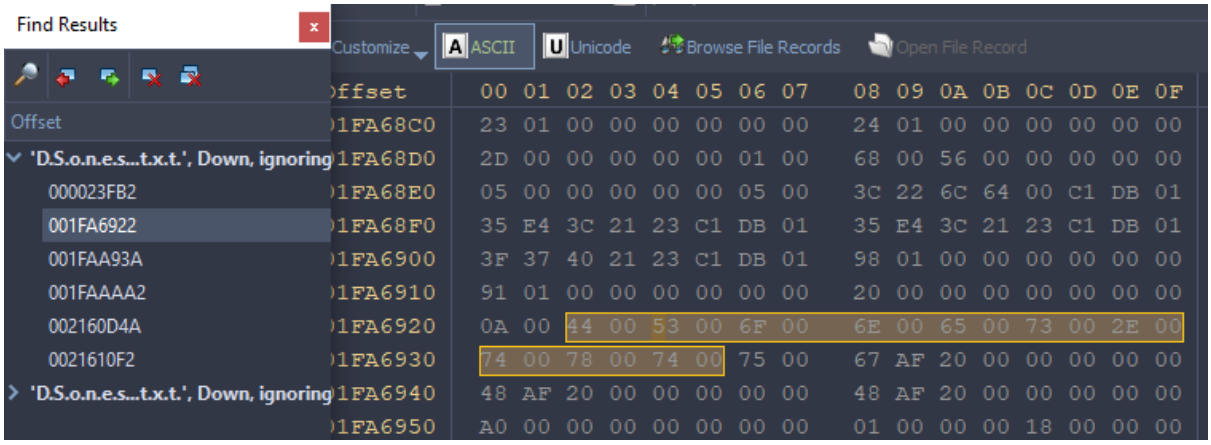


I then saved my changes opened the file contents in command prompt:



This shows that I have changed the entries successfully.

Forensics Part 12: Timestamps Part 1



Timestamps before editing:

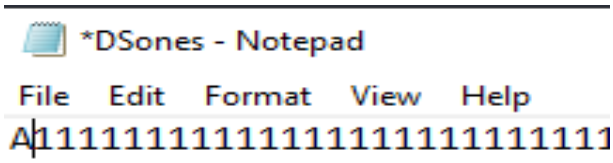
3C 22 6C 64 00 C1 DB 01 = Creation Time

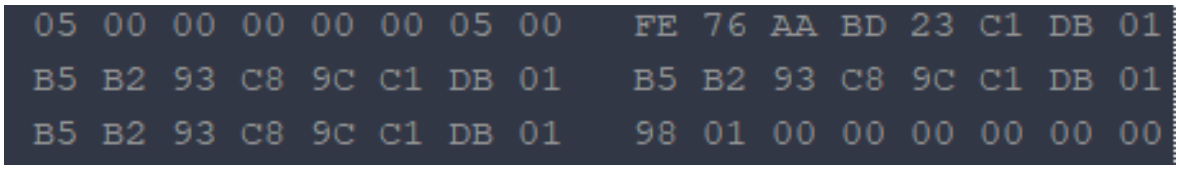
35 E4 3C 21 23 C1 DB 01 = Modified Time

3F 37 40 21 23 C1 DB 01 = MFT Changed Time

98 01 00 00 00 00 00 00 = Accessed Time

Next I edited the contents of the file and saved it:

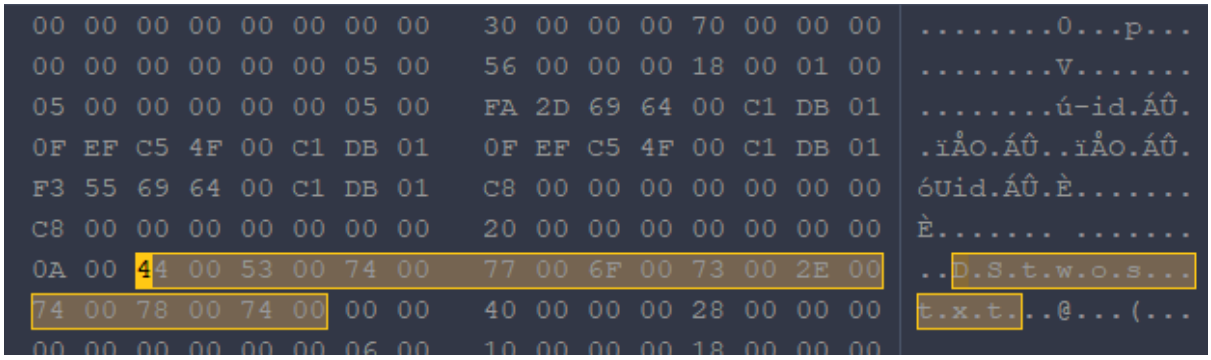




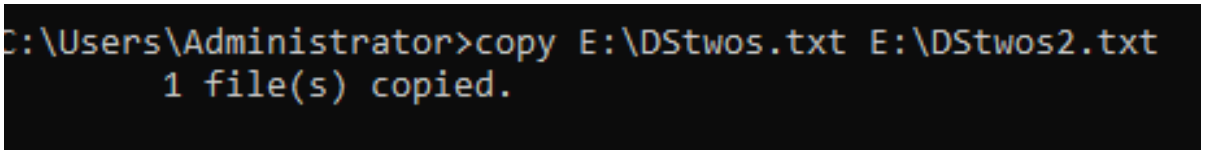
I can see that the Creation Time as well as the modified and MFT changed times have changed however I noticed the access time of 98 01 00 00 00 00 etc has been left unchanged.

Forensics Part 13: Timestamps Part 2:

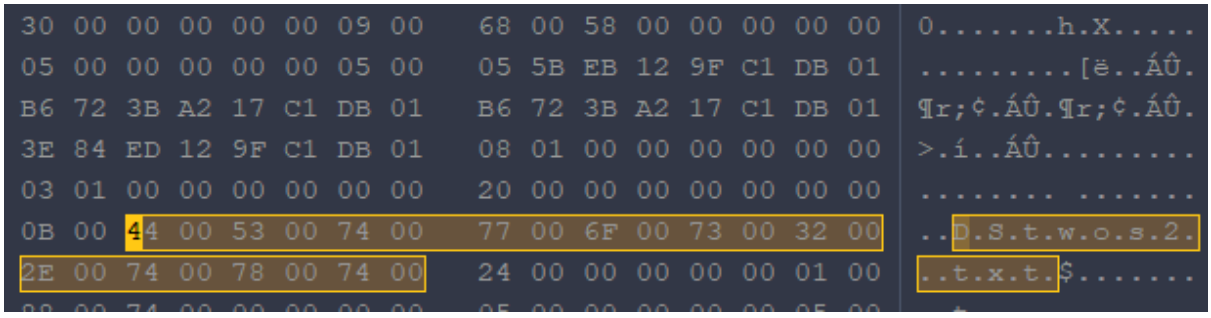
DStwos.txt:



I then copied the file in PowerShell naming the copy DStwos2.txt:



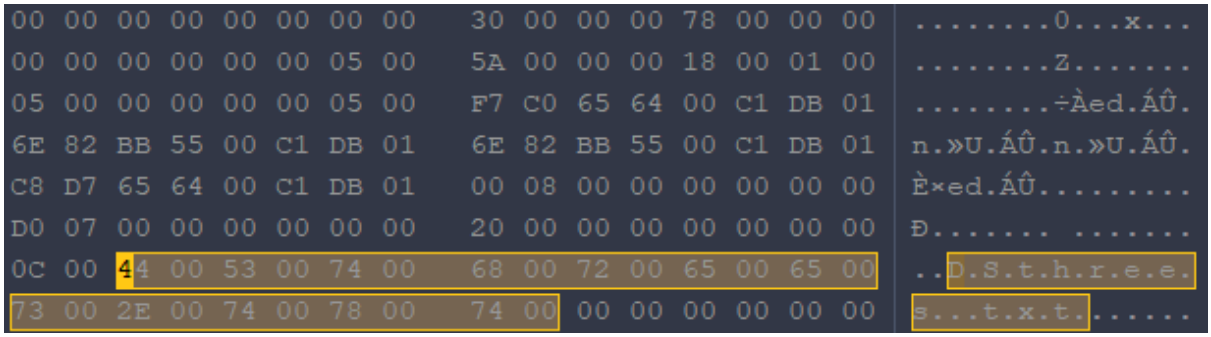
DStwos2.txt



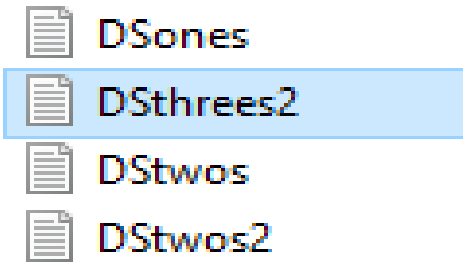
After using Disk editor to look at the original DStwos.txt and it's copy DStwos2.txt I found that all of the timestamps were all different, suggesting to me that the copied file is treated as a new file entirely because it has different timestamps for Created, Modified, MFT modified and accessed.

Forensics Part 14: Timestamps Part 3:

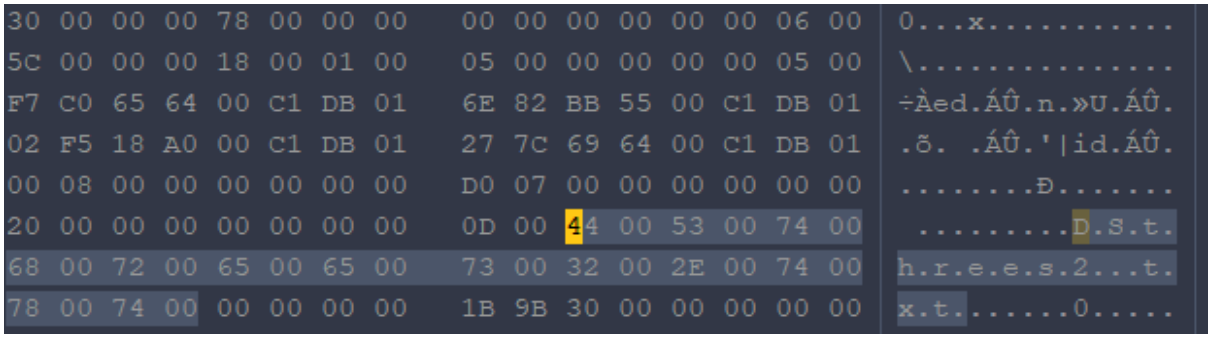
DSthrees.txt:



After this I renamed the file in file explorer from DSthrees.txt to DSthrees2.txt



Renamed File:



After using Disk editor to view the original DSthrees.txt file and the same file after renaming it I noticed that the File name attributes (created, modified, MFT modified and Accessed have all changed. This suggests to me that renaming a file can change it's metadata and all of it's timestamps are updated

Forensics 2 Try Hack Me:

I signed up to TryHackMe.com using my own personal email.

After this, I went to https://tryhackme.com/room/windowsforensics1 and joined the room where I then proceeded to answer Tasks 1,2,3,6,7 and 8

Task 1:

Answer the questions below

What is the most used Desktop Operating System right now?

Microsoft Windows ✓ Correct Answer

Task 2:

Answer the questions below

What is the short form for HKEY_LOCAL_MACHINE?

HKLM ✓ Correct Answer

Task 3:

Answer the questions below

What is the path for the five main registry hives, DEFAULT, SAM, SECURITY, SOFTWARE, and SYSTEM?

C:\Windows\System32\Config ✓ Correct Answer Hint

What is the path for the AmCache hive?

C:\Windows\AppCompat\Programs\Amcache.hve ✓ Correct Answer

Task 6:

Answer the questions below

What is the Current Build Number of the machine whose data is being investigated?

19044 ✓ Correct Answer Hint

Which ControlSet contains the last known good configuration?

1 ✓ Correct Answer

What is the Computer Name of the computer?

THM-4n6 ✓ Correct Answer

What is the value of the TimeZoneKeyName?

Pakistan Standard Time ✓ Correct Answer

What is the DHCP IP address

192.168.100.58 ✓ Correct Answer

What is the RID of the Guest User account?

501 ✓ Correct Answer Hint

Task 7:

Answer the questions below

When was EZtools opened?

2021-12-01 13:00:34 ✓ Correct Answer Hint

At what time was My Computer last interacted with?

2021-12-01 13:06:47 ✓ Correct Answer Hint

What is the Absolute Path of the file opened using notepad.exe?

C:\Program Files\Amazon\Ec2ConfigService\Settings ✓ Correct Answer

When was this file opened?

2021-11-30 10:56:19 ✓ Correct Answer Hint

Task 8:

Answer the questions below

How many times was the File Explorer launched?

26 ✓ Correct Answer Hint

What is another name for ShimCache?

AppCompatCache ✓ Correct Answer

Which of the artifacts also saves SHA1 hashes of the executed programs?

AmCache ✓ Correct Answer

Which of the artifacts saves the full path of the executed programs?

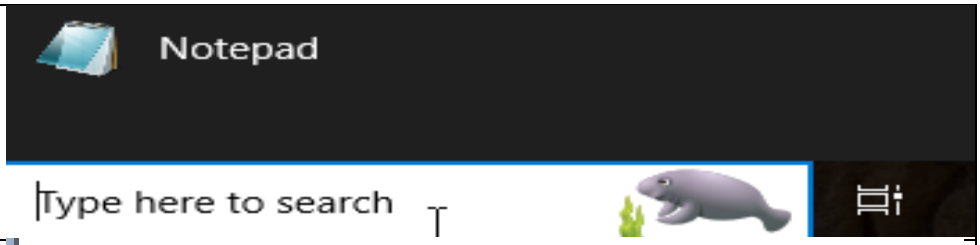
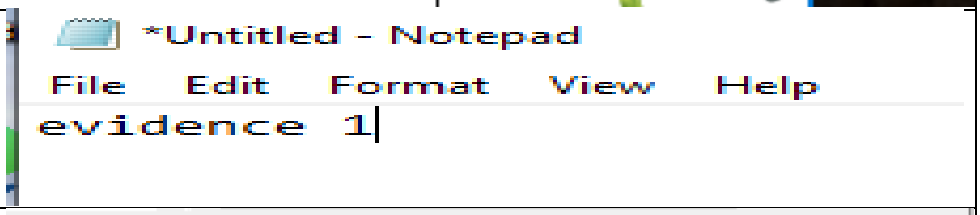
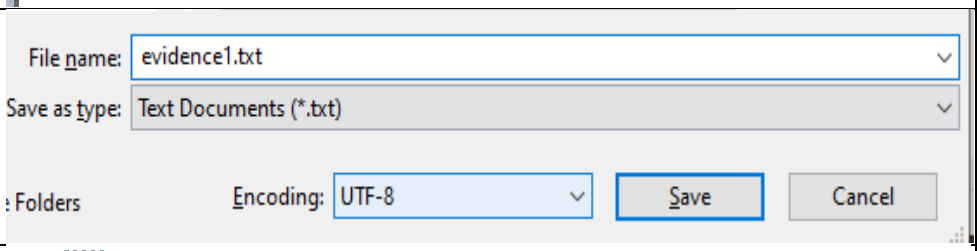
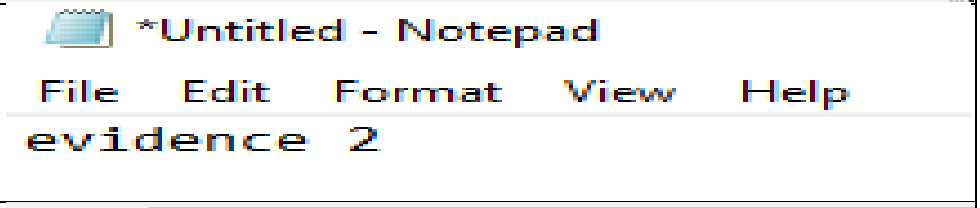
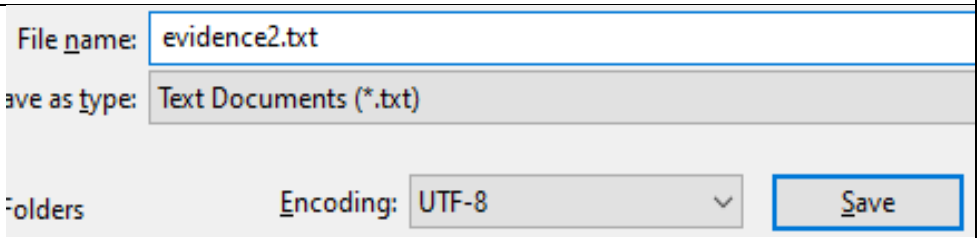
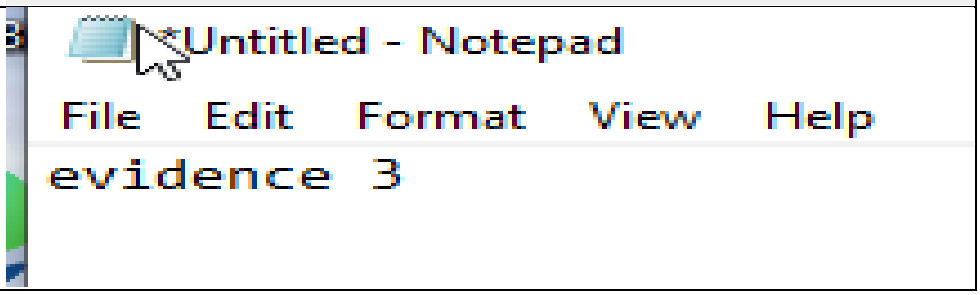
BAM/DAM ✓ Correct Answer

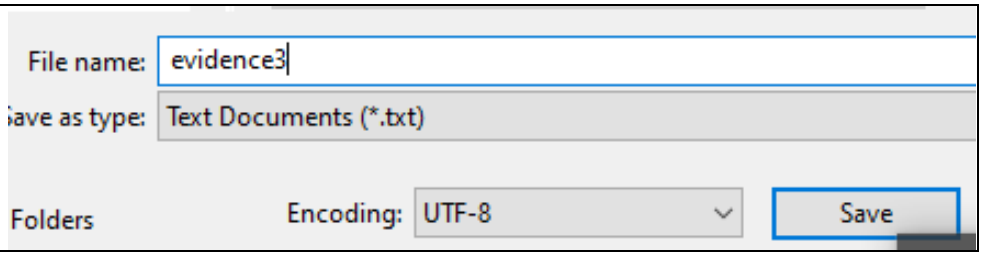
Final Completion:

Forensics 3

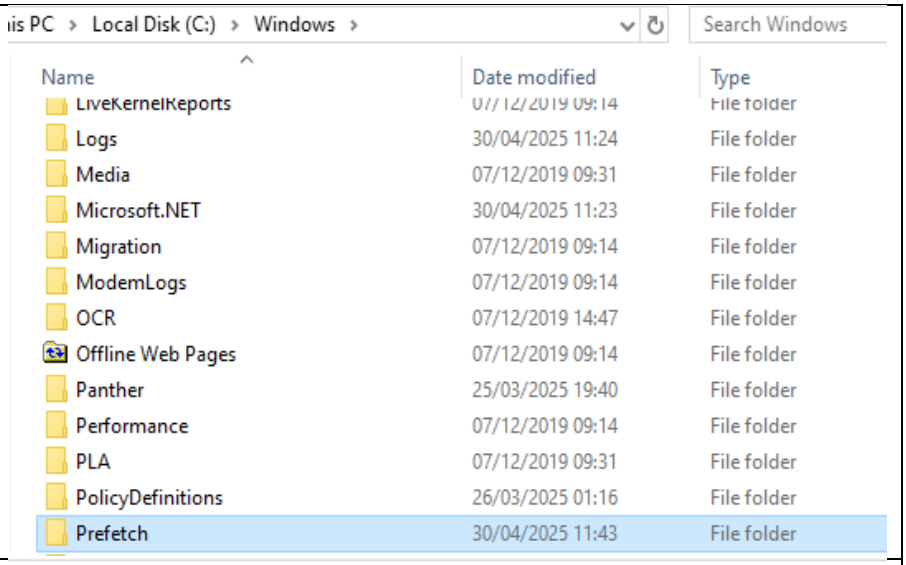
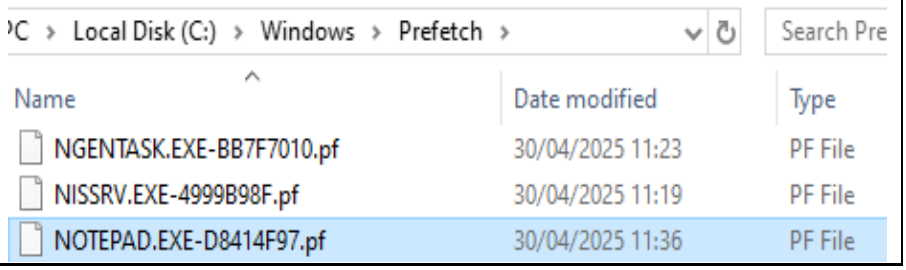
For the final task of Assignment 2 I will be creating evidence in my Windows 10 Machine

Task 1:

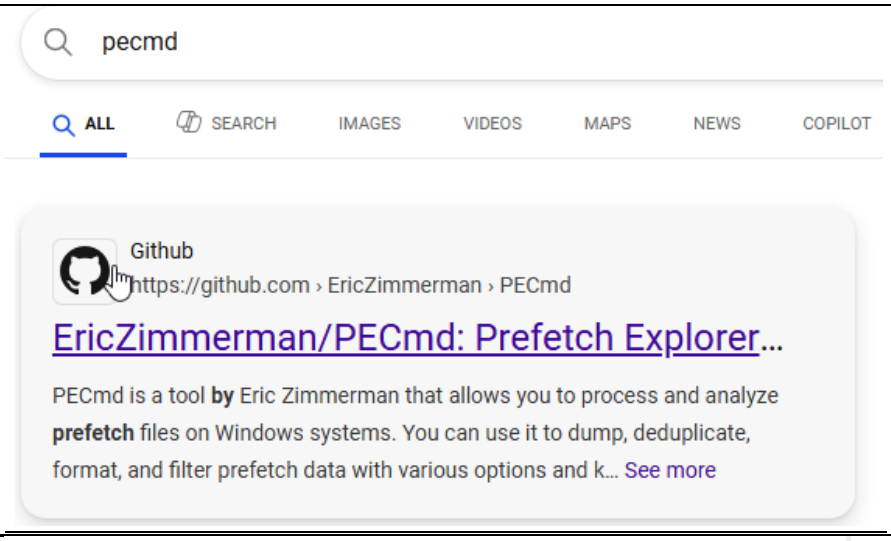
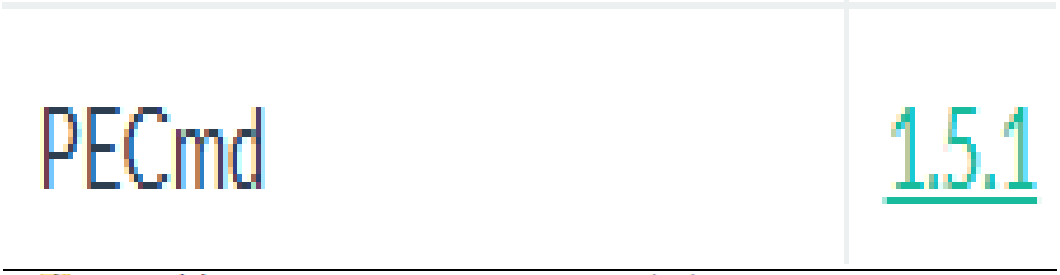
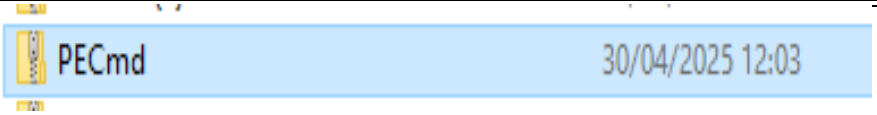
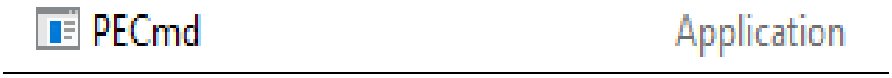
I started by running notepad from the start menu	
I created a text file and wrote: **"evidence1"**	
I saved it as **evidence1.txt**	
I created another text file and wrote: **"evidence2"**	
I saved it as **evidence2.txt**	
I created one last text file and wrote: **"evidence3"**	

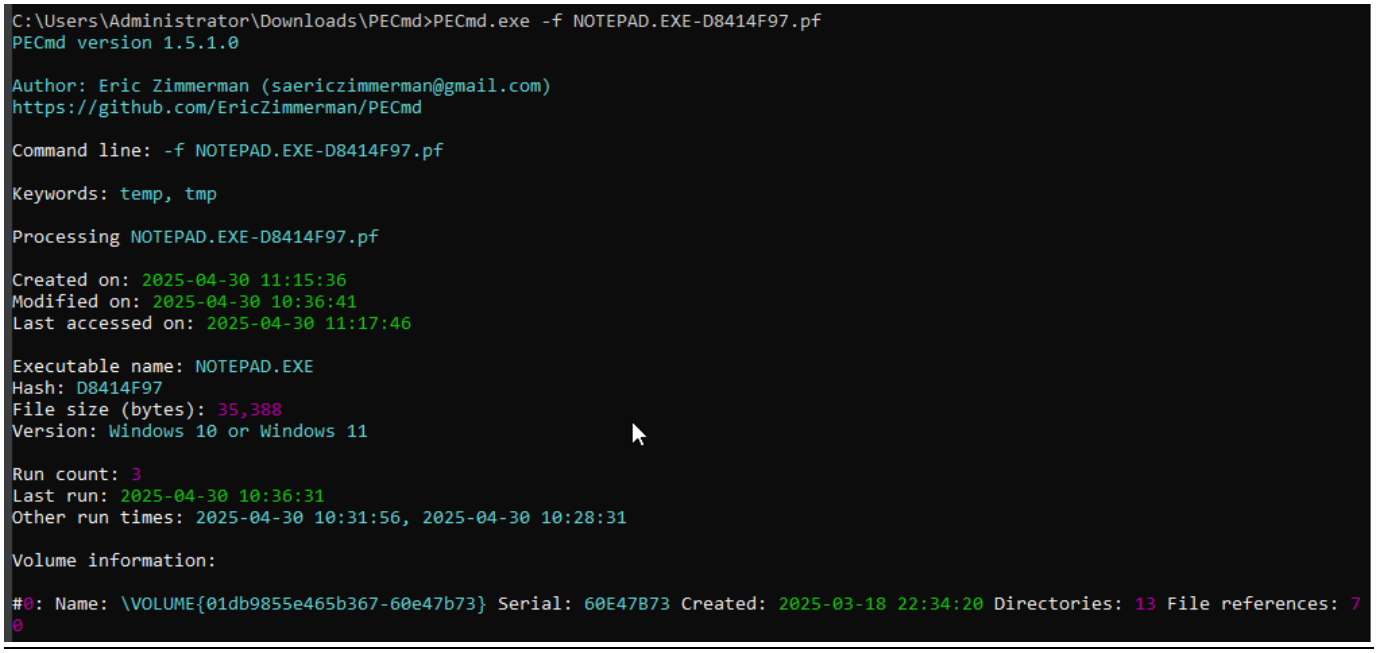
I saved it as **evidence3.txt**	

Locating the Prefetch file:

Next I opened the file explorer on my Windows 10 VM and navigated to C:Windows>Prefetch Here my prefetch files are stored.	
I located the prefetch file in relation to Notepad successfully.	

Task 2:

Next I went to https://ericzimmerman.github.io/#!index.md	
Here I downloaded a tool named "PECmd" that I would be using to analyse the prefetch file I found earlier.	
I unzipped the file	
I then moved the program into a folder with the notepad prefetch file	



After this I opened the command prompt and ran the PECmd.exe program directing it to the notepad prefetch file.

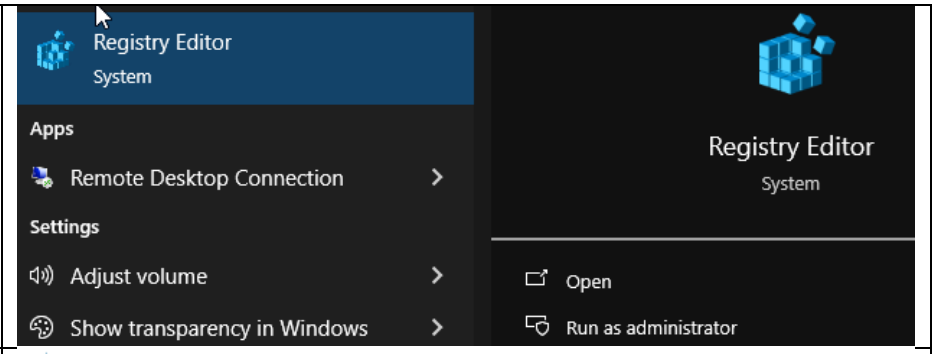
The 1st thing I noticed is that the output confirms the prefetch file belongs to notepad. It also displayed file activity Timestamps such as:

- **Created on: 2025-04-30 11:15:36**
- **Modified on: 2025-04-30 10:36:41**
- **Last accessed: 2025-04-30 11:17:46**

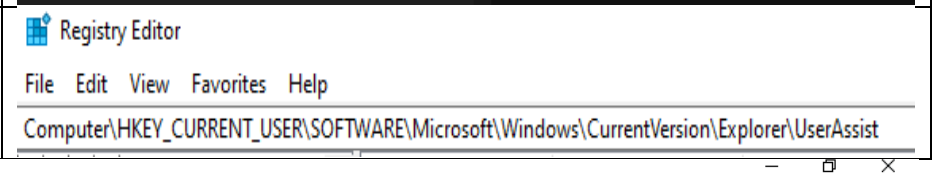
It also gave me a run count telling me Notepad was run 3 times which is correct as shown by the evidence I created. It also shows me 13 directories and 7 files that Notepad used when it was run.

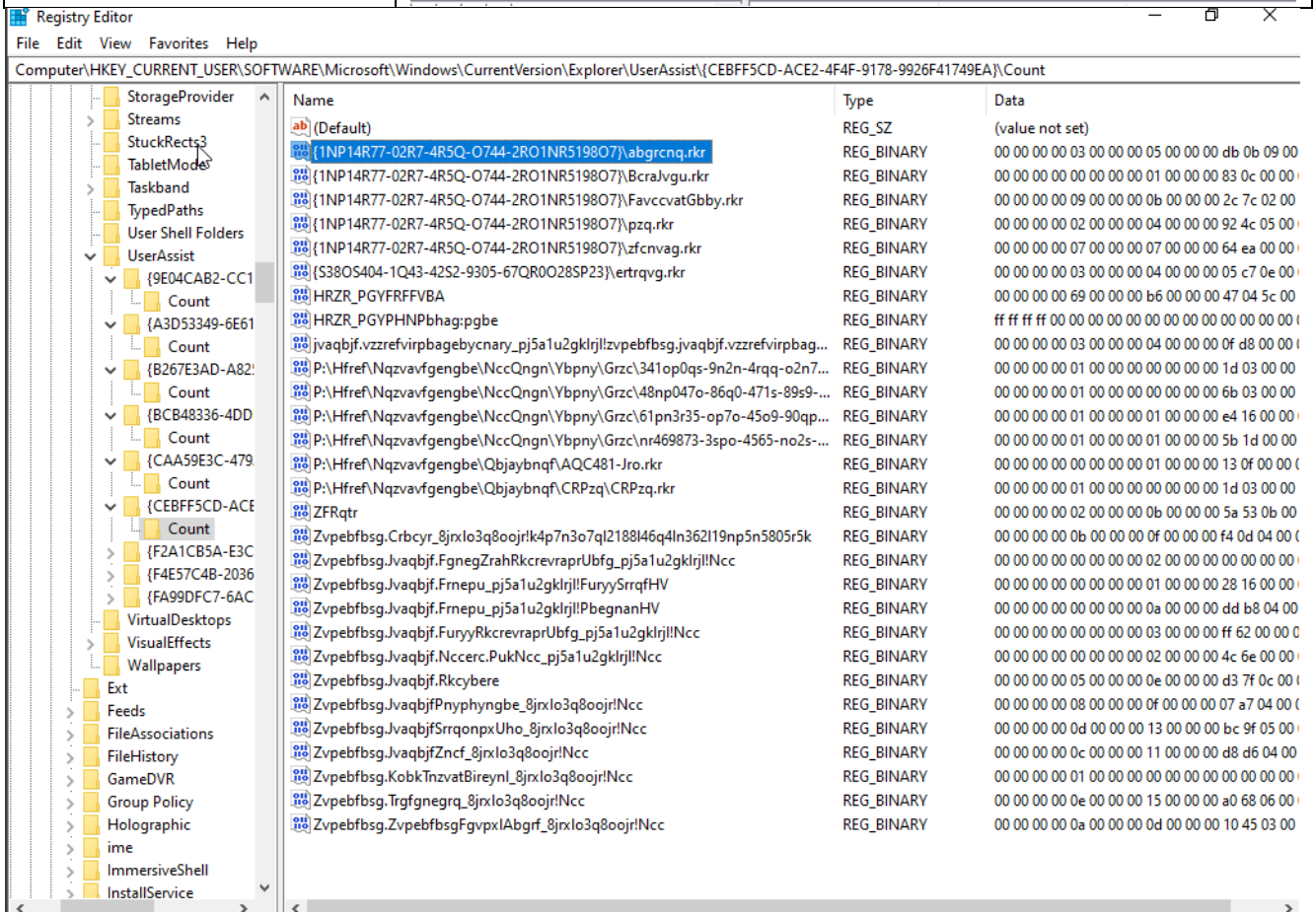
Task 3:

Next I opened Registry Editor in order to locate the UserAssist key used to run notepad

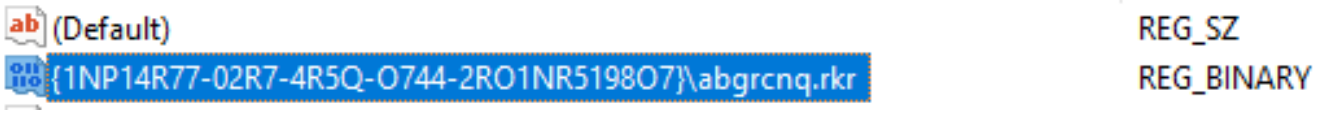


I followed the path shown in the image

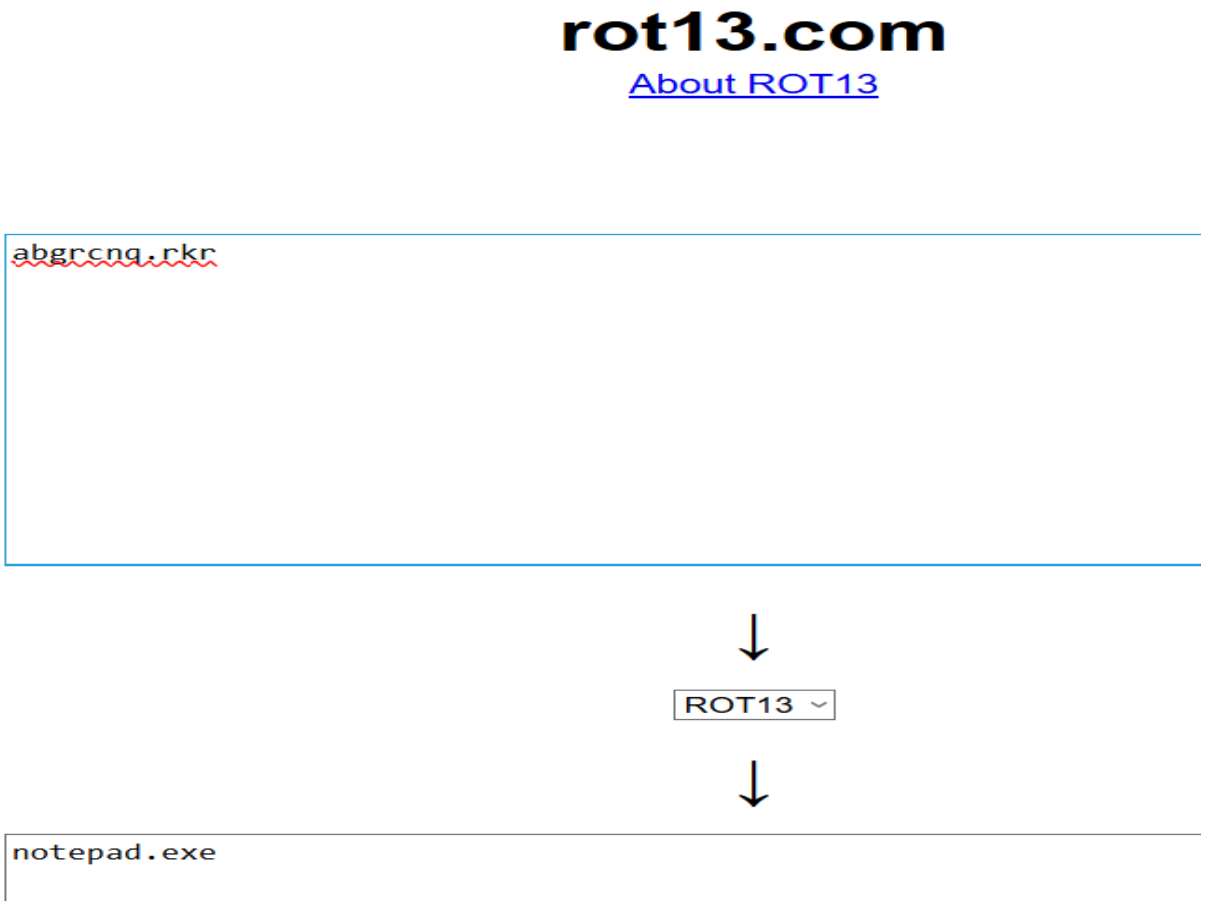




I then looked through each UserAssist registry key until I stumbled upon **abgrcnq.rkr**



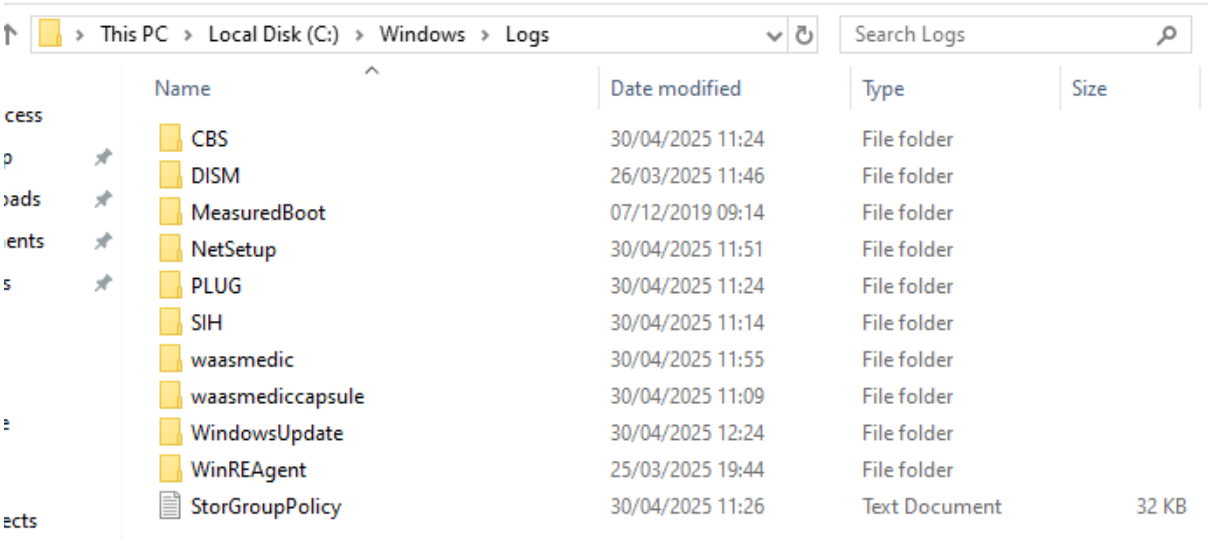
Once I got this registry key I navigated to rot13.com which is used to decode rot13, Upon entering the registry key name "**abgrcnq.rkr"** I noticed that it was translated to **"notepad.exe"** confirming to me that notepad was run by a user and tracked in the registry

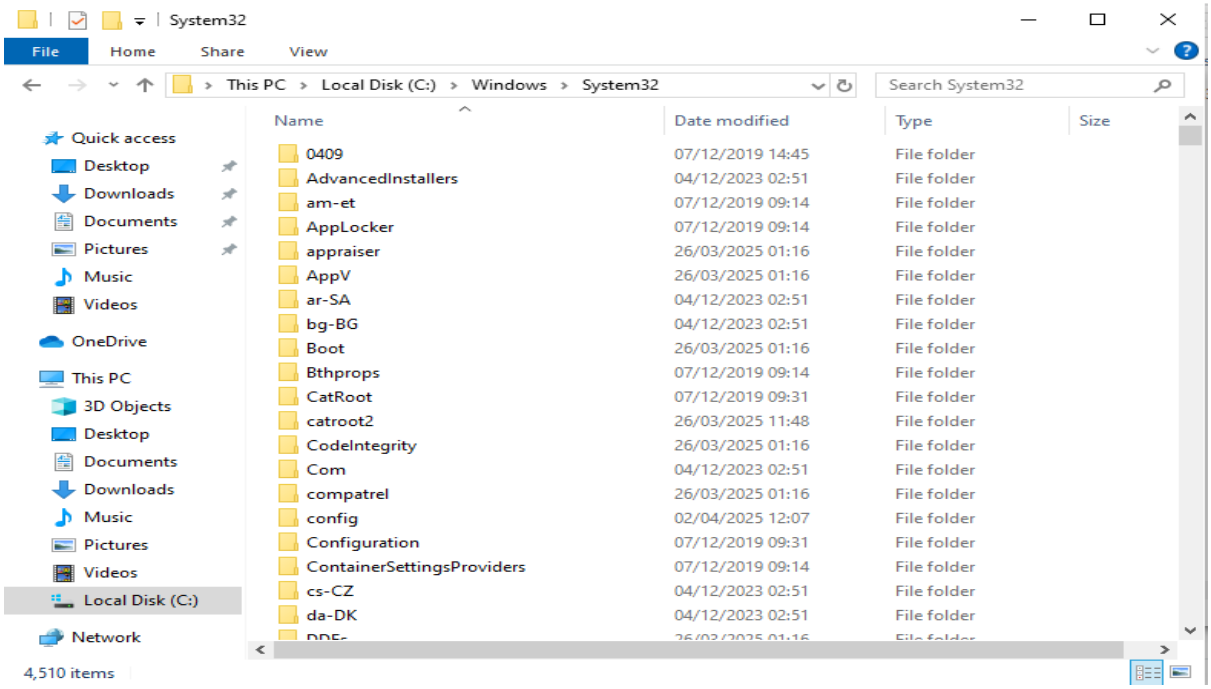


Task 4:

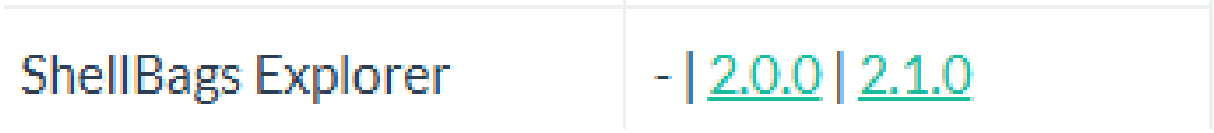
Next I created some evidence by going to my file explorer and navigating to the Directories:

C://Windows/Logs and **C://Windows/System32**

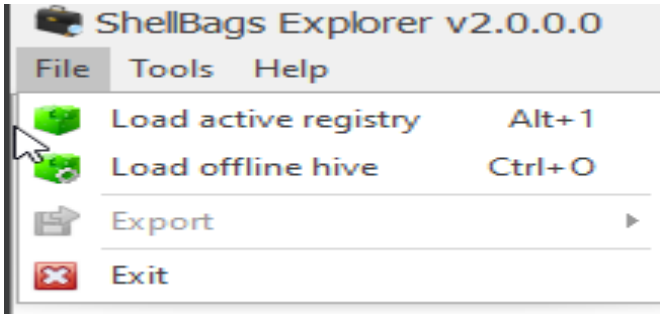




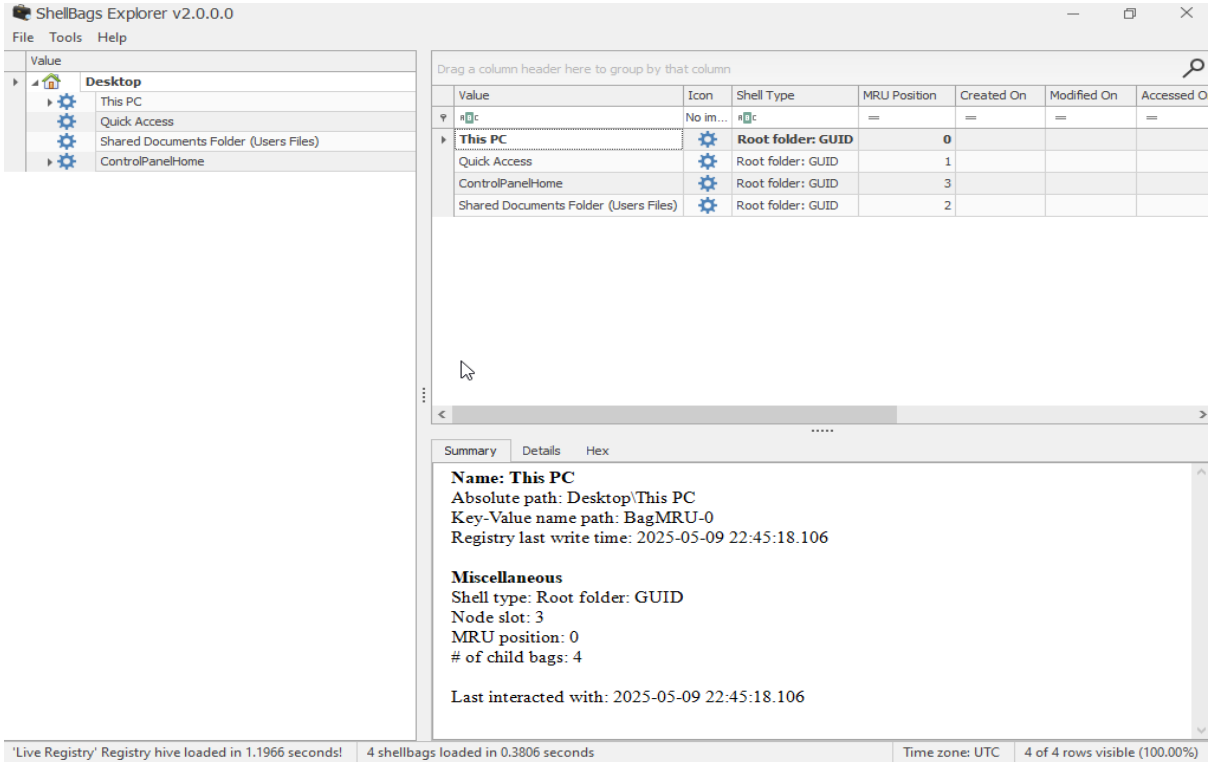
After this was done I went back to https://ericzimmerman.github.io/#!index.md and installed ShellBags explorer from the GitHub page



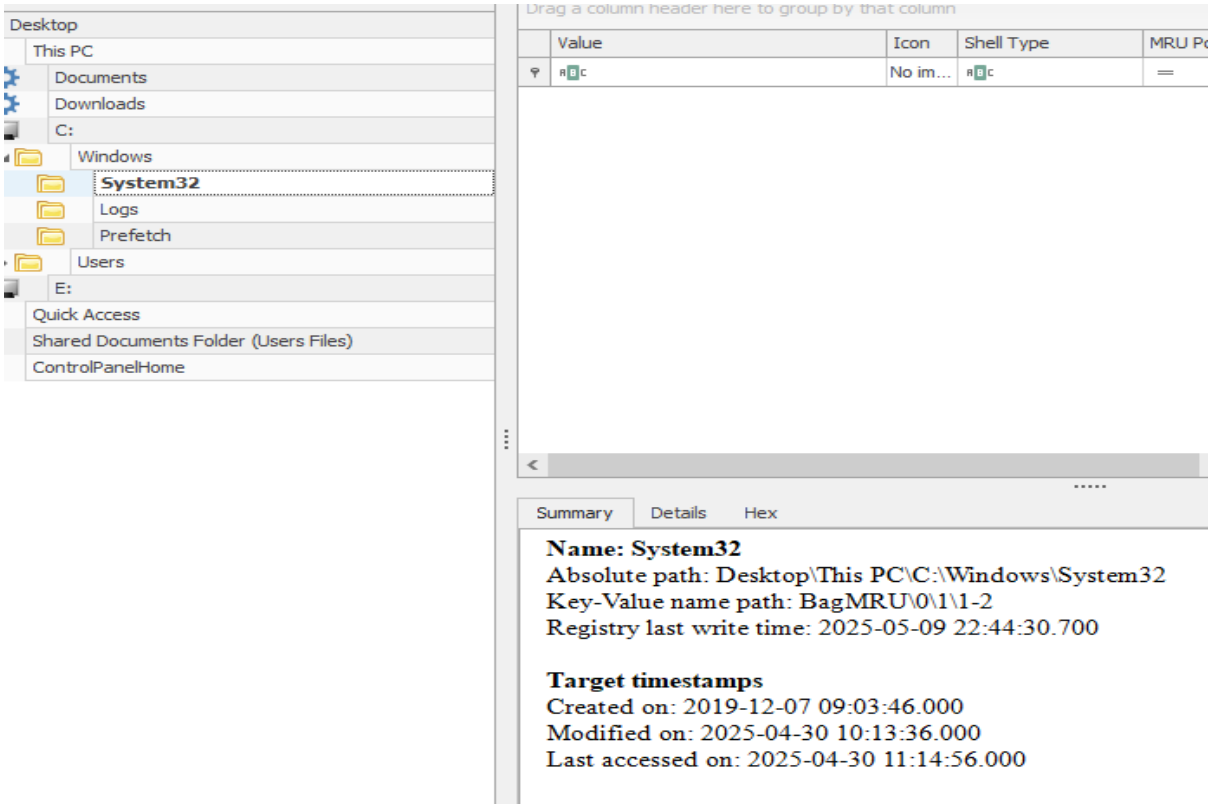
After navigating into both folders and backing out I then used the ShellBags explorer program that I installed to load the active registry hive and view recorded shell bag entries.

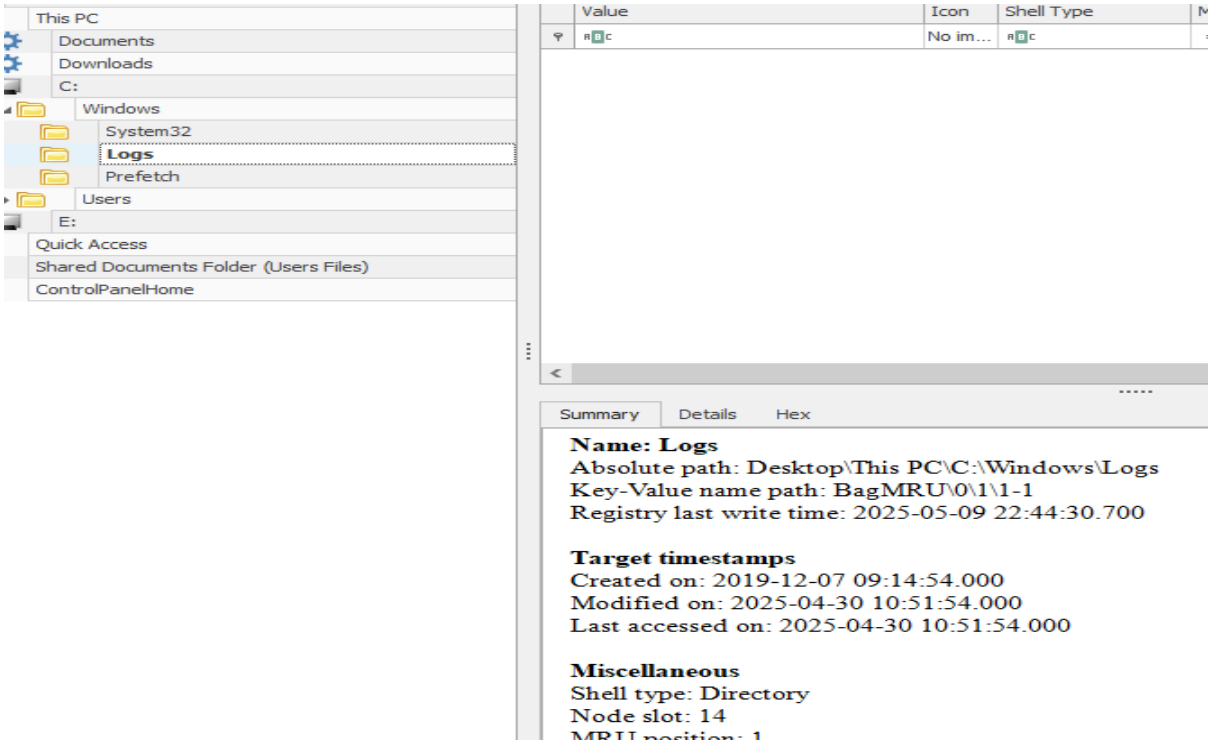


Once the Hive was loaded I was met with this home page:



Next I expanded the “This PC” field on the left and followed the path to both of the respective folders that I visited earlier.


Desktop
This PC
Documents
Downloads
C:
Windows
System32
Logs
Prefetch
Users
E:
Quick Access
Shared Documents Folder (Users Files)
ControlPanelHome
Drag a column header here to group by that column
Value
Icon
Shell Type
MRU Po
No im...
Summary
Details
Hex
Name: System32
Absolute path: Desktop\This PC\C:\Windows\System32
Key-Value name path: BagMRU\0\1\1-2
Registry last write time: 2025-05-09 22:44:30.700
Target timestamps
Created on: 2019-12-07 09:03:46.000
Modified on: 2025-04-30 10:13:36.000
Last accessed on: 2025-04-30 11:14:56.000


This PC
Documents
Downloads
C:
Windows
System32
Logs
Prefetch
Users
E:
Quick Access
Shared Documents Folder (Users Files)
ControlPanelHome
Value
Icon
Shell Type
No im...
Summary
Details
Hex
Name: Logs
Absolute path: Desktop\This PC\C:\Windows\Logs
Key-Value name path: BagMRU\0\1\1-1
Registry last write time: 2025-05-09 22:44:30.700
Target timestamps
Created on: 2019-12-07 09:14:54.000
Modified on: 2025-04-30 10:51:54.000
Last accessed on: 2025-04-30 10:51:54.000
Miscellaneous
Shell type: Directory
Node slot: 14

After investigating both directories I was met with the following evidence:

System32:

Absolute path: C:\Windows\System32

Last accessed: 2025-04-30 11:14:56.000

Registry last write time: 2025-05-09 22:44:30.700

Logs:

Absolute path: C:\Windows\Logs

Last accessed: 2025-04-30 10:51:54.000

Registry last write time: 2025-05-09 22:44:30.700

Conclusion: These results showed me that the ShellBag data was created and updated when the folders were opened earlier in the File explorer, confirming that Windows records activity at a registry level, this is useful for forensic investigations as system admins can see exactly when a user interacts with a folder.