Santa’s Workshop Laptop Practice Image

ANSWER KEY

Written by: RainbowDynamix

Scored Check Categories

- Forensics Questions - 2
- Access Control - 4
- Services/Maintenance - 1
- Unauthorized Files and Applications - 5
- Persistence Mechanisms - 1
- Security Policies - 7

Forensics Questions

Forensics Question #1

As a temporary file server solution during a backup of workshop resources, this system was configured to share “C:\workshop” anonymously throughout the network.

What is the configured share name for this resource?

Example: StaffDirectory$
ANSWER: <type answer here>

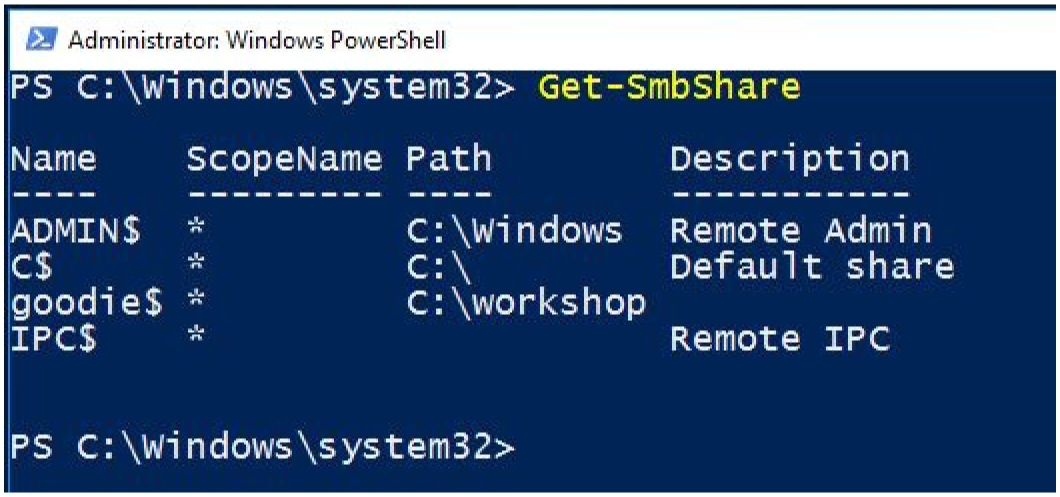
According to the share permissions of this resource, what permissions have been configured for all users? (AccessControlType, AccessRight)

Example: Allow, Read

Solution

File shares on Windows use the SMB (Server Message Block) protocol. In updated versions of Windows such as Windows 10, newer versions of PowerShell containing very useful management cmdlets are available. We will use these new cmdlets to solve this question. However this is not the only way to solve this question.

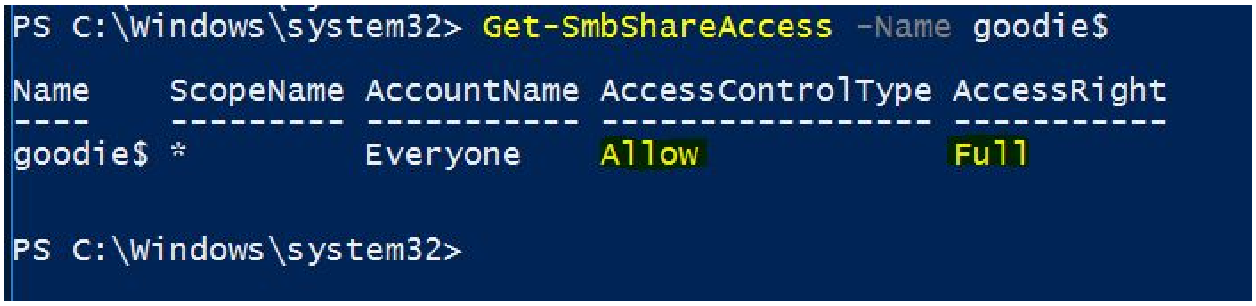
To list the file shares available on the system, we can use the *Get-SmbShare* cmdlet as shown below.



```
Administrator: Windows PowerShell
PS C:\Windows\system32> Get-SmbShare

Name     ScopeName Path         Description
----     --------- ----         -----------
ADMIN$   *         C:\Windows   Remote Admin
C$       *         C:\          Default share
goodie$  *         C:\workshop
IPC$     *                      Remote IPC


PS C:\Windows\system32>
```

As the question stated, the system is sharing “C:\workshop” under the share name, “goodie$” (fun fact: The dollar sign at the end means it’s a *hidden* file share).

If we run *Get-SmbShareAccess* on this share, we can see the share permissions that have been granted on this resource.



```
PS C:\Windows\system32> Get-SmbShareAccess -Name goodie$

Name     ScopeName AccountName AccessControlType AccessRight
----     --------- ----------- ----------------- -----------
goodie$  *         Everyone    Allow             Full


PS C:\Windows\system32>
```

As you can see, the configured permissions on this share are “Allow, Full” for the built-in *Everyone* group (all users)

ANSWER: Allow, Full

Forensics Question #2

At Santa's workshop communication is very important. Because of this, the elves decided to implement bulk instant messaging through WinSMS. During a security audit, extremely permissive file permissions were enumerated and identified in the WinSMS installation directory. These permissions could allow an unprivileged user to execute arbitrary code with system-level privileges.

According to the ACL of "C:\Program Files (x86)\WinSMS", what is the SDDL string associated with its permissions?

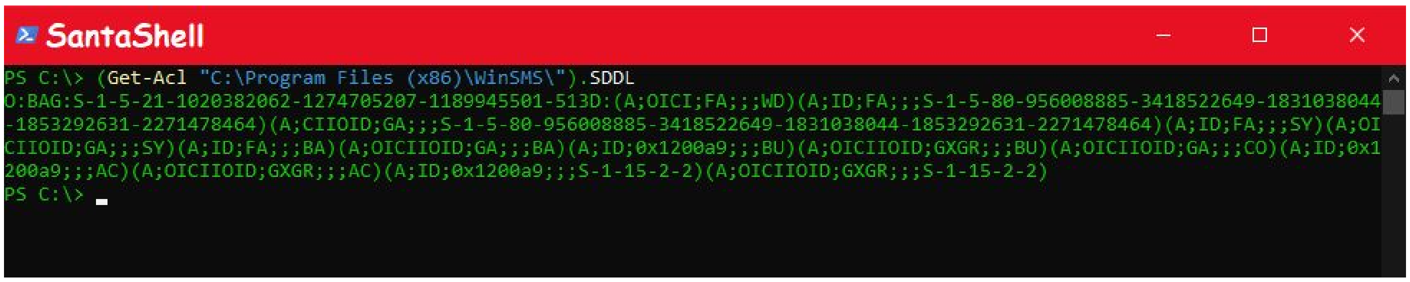
Solution

SDDL (Security Descriptor Definition Language) strings look like the example provided below. This string is a representation of the permissions configured on an object, such as a directory, file, service, etc. In this forensics question, the user was asked to identify this string for auditing purposes. This is useful because it is a single string that takes up little space and can be parsed for further investigation.

SDDL String Format Example ('C:\Windows' directory):
O:S-1-5-80-956008885-3418522649-1831038044-1853292631-2271478464**G:**S-1-5-80-956008885-3418522649-1831038044-1853292631-2271478464**D:PAI(A;OICIIO;GA;;;CO)**(A;OICIIO;GA;;;SY)(A;;0x1301bf;;;SY)(A;OICIIO;GA;;;BA)(A;;0x1301bf;;;BA)(A;OICIIO;GXGR;;;BU)(A;;0x1200a9;;;BU)(A;CIIO;GA;;;S-1-5-80-956008885-3418522649-1831038044-1853292631-2271478464)(A;;FA;;;S-1-5-80-956008885-3418522649-1831038044-1853292631-2271478464)(A;;0x1200a9;;;AC)(A;OICIIO;GXGR;;;AC)(A;;0x1200a9;;;S-1-15-2-2)(A;OICIIO;GXGR;;;S-1-15-2-2)

I know, this looks complicated, but it isn't as daunting as it seems when piecing it together. As you can see, the labels, **O:, G:,** and **D:()** have been highlighted above. **O:** for owner, **G:** for the **group** by which the **owner** is a part of, and **D:()**, for **DACL (Discretionary Access Control List)**. This information tells Windows how to delegate access to this resource

If we look at the WinSMS directory ("C:\Program Files (x86)\WinSMS"), we can see *this* SDDL string by typing in the following Get-Acl command into a PowerShell, or in this case, SantaShell, window.



```
PS C:\> (Get-Acl "C:\Program Files (x86)\WinSMS\").SDDL
O:BAG:S-1-5-21-1020382062-1274705207-1189945501-513D:(A;OICI;FA;;;WD)(A;ID;FA;;;S-1-5-80-956008885-3418522649-1831038044
-1853292631-2271478464)(A;CIIOID;GA;;;S-1-5-80-956008885-3418522649-1831038044-1853292631-2271478464)(A;ID;FA;;;SY)(A;OI
CIIOID;GA;;;SY)(A;ID;FA;;;BA)(A;OICIIOID;GA;;;BA)(A;ID;0x1200a9;;;BU)(A;OICIIOID;GXGR;;;BU)(A;OICIIOID;GA;;;CO)(A;ID;0x1
200a9;;;AC)(A;OICIIOID;GXGR;;;AC)(A;ID;0x1200a9;;;S-1-15-2-2)(A;OICIIOID;GXGR;;;S-1-15-2-2)
PS C:\>
```

ANSWER:

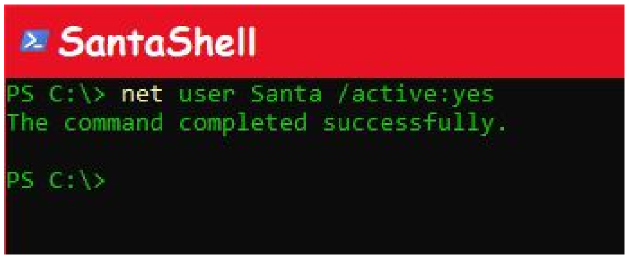
O:BAG:S-1-5-21-1020382062-1274705207-1189945501-513D:(A;OICI;FA;;;WD)(A;ID;FA;;;S-1-5-80-956008885-3418522649-1831038044-1853292631-2271478464)(A;CIIOID;GA;;;S-1-5-80-956008885-3418522649-1831038044-1853292631-2271478464)(A;ID;FA;;;SY)(A;OICIIOID;GA;;;SY)(A;ID;FA;;;BA)(A;OICIIOID;GA;;;BA)(A;ID;0x1200a9;;;BU)(A;OICIIOID;GXGR;;;BU)(A;OICIIOID;GA;;;CO)(A;ID;0x1200a9;;;AC)(A;OICIIOID;GXGR;;;AC)(A;ID;0x1200a9;;;S-1-15-2-2)(A;OICIIOID;GXGR;;;S-1-15-2-2)

Access Control

Santa's account is enabled

According to the README, Santa is an Authorized Administrator. However, on this image, Santa's user account is disabled. You must enable this account.

The quickest way to do this is through an Administrator PowerShell or cmd session with the following command.

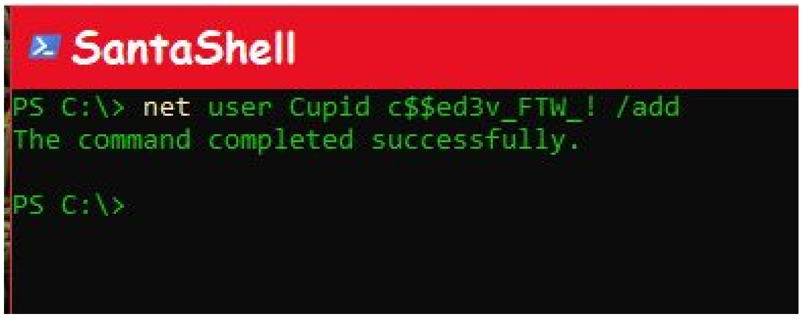


```
PS C:\> net user Santa /active:yes
The command completed successfully.

PS C:\>
```

Created user account for Cupid

According to the README, Cupid is an Authorized User. However, on this image, Cupid does not have a user account. You must create a user account for Cupid.

The quickest way to do this is through an Administrator PowerShell or cmd session with the following command. You should also create a strong password. In this case, I chose ***c$$ed3v_FTW_!***.

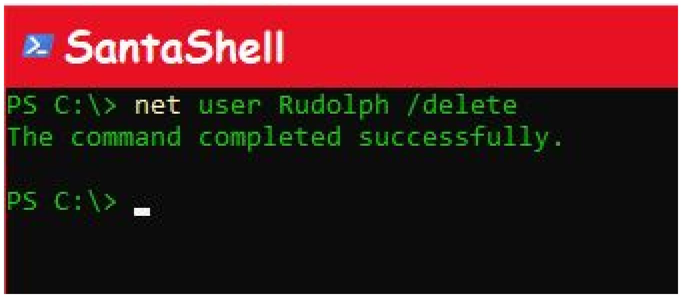



```
PS C:\> net user Cupid c$$ed3v_FTW_! /add
The command completed successfully.

PS C:\>
```

Removed Rudolph's user account

According to the README, neither the Authorized Administrators or Authorized Users lists contain the user Rudoplh. This account should be removed from the system.

The quickest way to do this is through an Administrator PowerShell or cmd session with the following command.

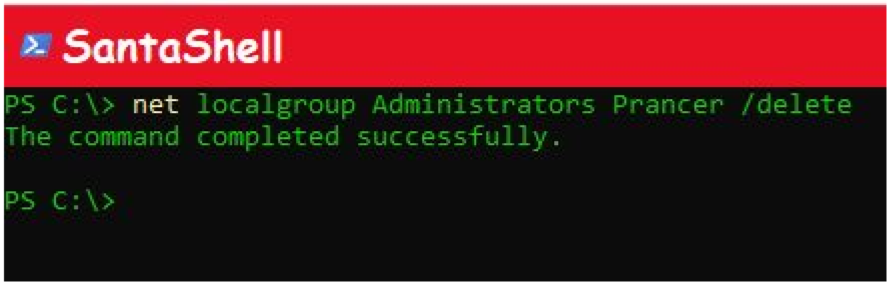


```
PS C:\> net user Rudolph /delete
The command completed successfully.

PS C:\> _
```

Removed Prancer from the Administrators group

According to the README, Prancer should be an Authorized User on the system, but not an Authorized Administrator. To do this, you must remove Prancer from the Administrators group.

The quickest way to do this is through an Administrator PowerShell or cmd session with the following command.



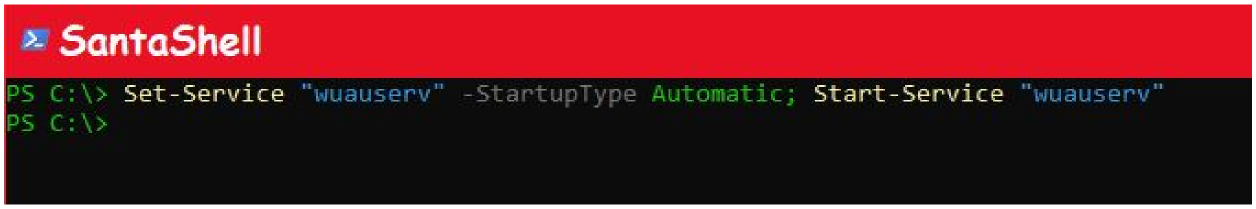
```
PS C:\> net localgroup Administrators Prancer /delete
The command completed successfully.

PS C:\>
```

Services/Maintenance

Set Windows Update service to start automatically

In order to receive the latest updates on this workstation, the Windows Update service must be running and configured to start automatically.

The quickest way to do this is through an Administrator PowerShell session with the following command.



```
PS C:\> Set-Service "wuauserv" -StartupType Automatic; Start-Service "wuauserv"
PS C:\>
```

Unauthorized Files and Applications

Removed C:\Universal Truths\Rudolph_is_dumb.png

According to the README, *"Santa expects that all of his reindeer follow the strict no media rule".* This means there should be no non-work related media files such as images, videos, applications, etc.

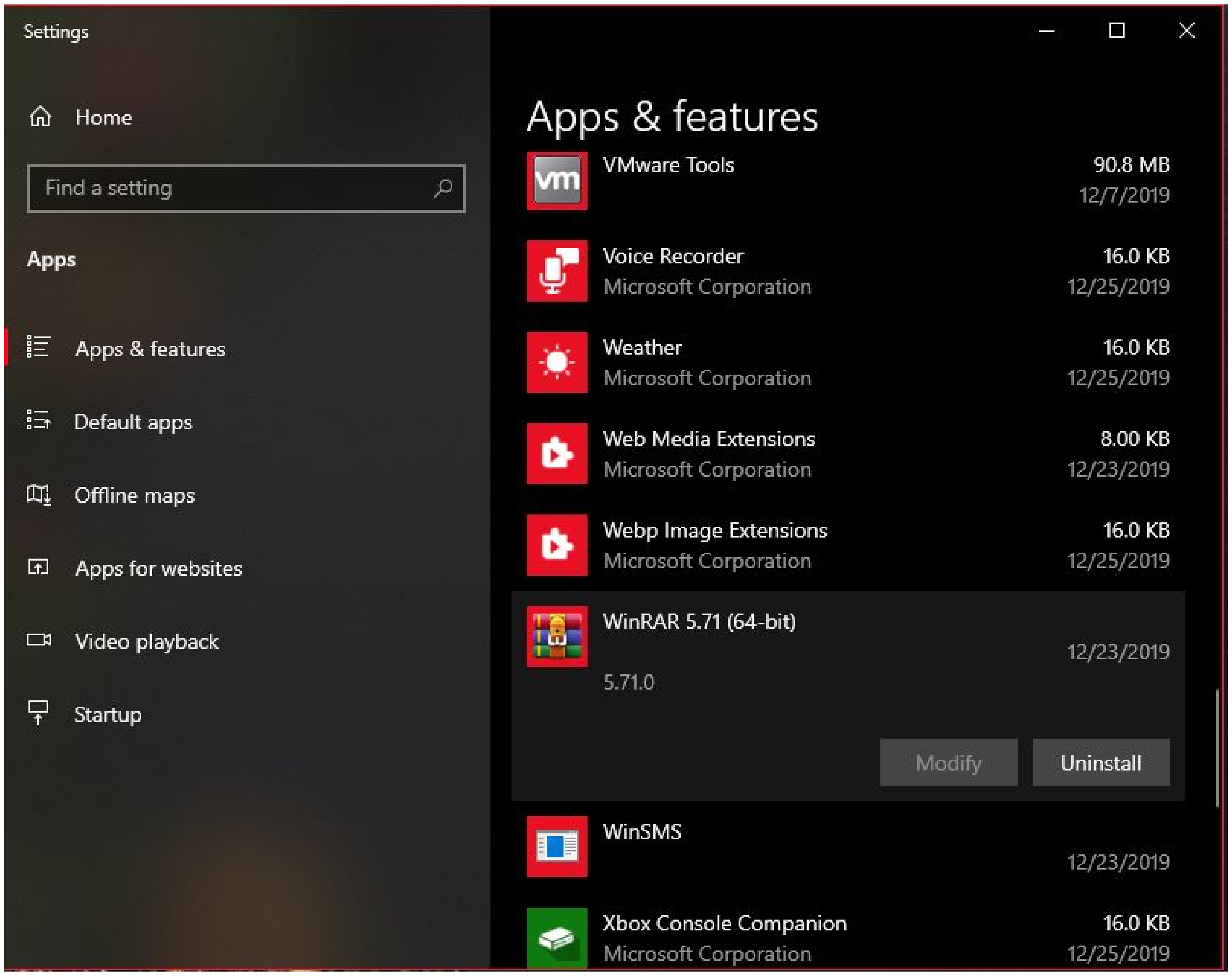
If you open the file explorer and navigate to the C drive, you'll notice a non-standard folder called *"Universal Truths"*. If you open this folder, you will find the following image (Rudolph_is_dumb.png).

Removed C:\Users\Santa\Desktop\tunes.mp3

According to the README, *"Santa expects that all of his reindeer follow the strict no media rule".* This means there should be no non-work related media files such as images, videos, applications, etc.

Removed WinRAR

According to the README, the only authorized program that should remain on the system is BioniX. Everything else should be removed. In this case, you can simply uninstall WinRAR via Apps & Features.

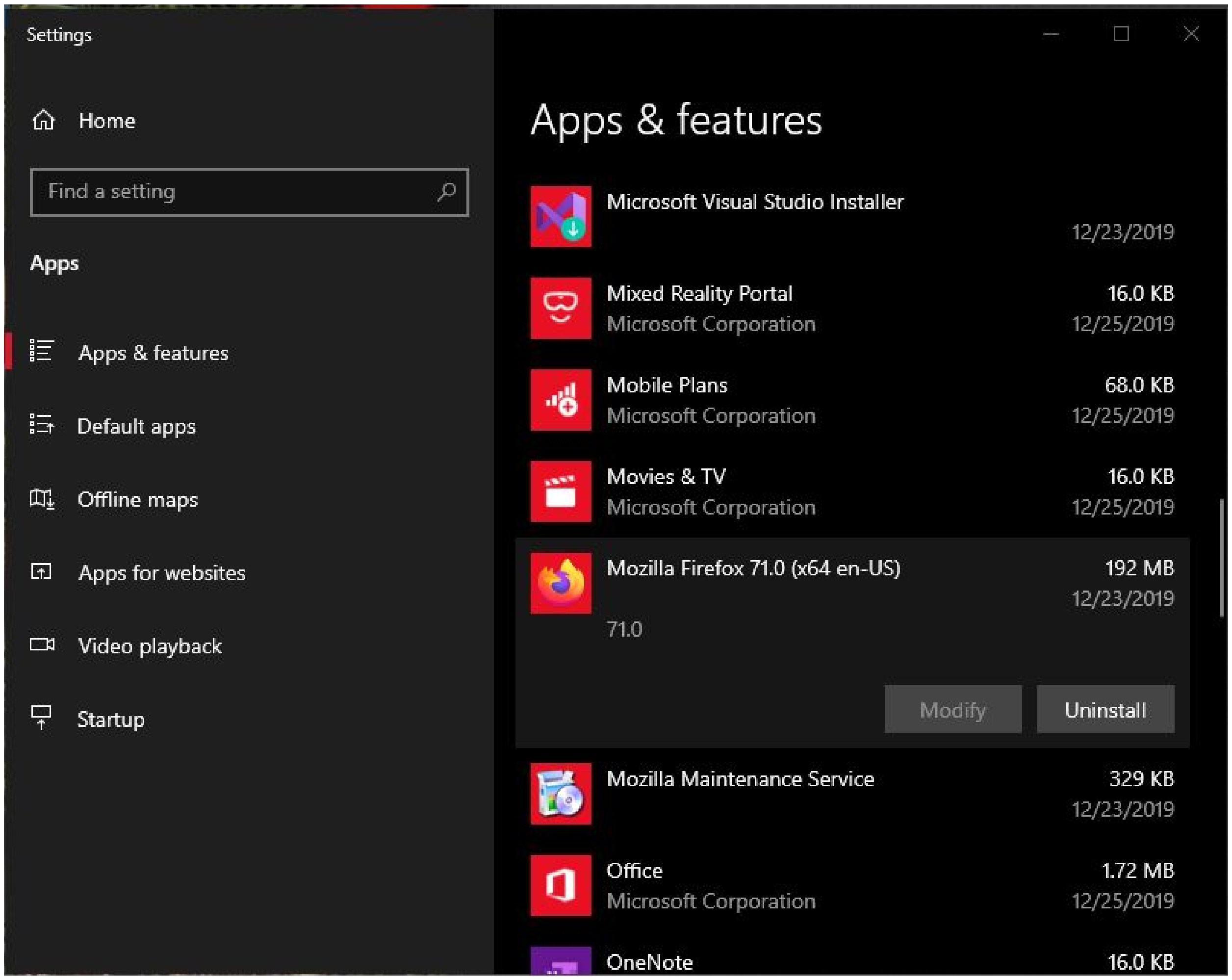


Apps & features

VMware Tools		90.8 MB
		12/7/2019
Voice Recorder		16.0 KB
Microsoft Corporation		12/25/2019
Weather		16.0 KB
Microsoft Corporation		12/25/2019
Web Media Extensions		8.00 KB
Microsoft Corporation		12/23/2019
Webp Image Extensions		16.0 KB
Microsoft Corporation		12/25/2019
WinRAR 5.71 (64-bit)		
		12/23/2019
5.71.0		
WinSMS		
		12/23/2019
Xbox Console Companion		16.0 KB
Microsoft Corporation		12/25/2019

Removed Mozilla Firefox

According to the README, the only browser that should remain on the system should be Microsoft Edge. In this case, you should uninstall Firefox via Apps & Features.

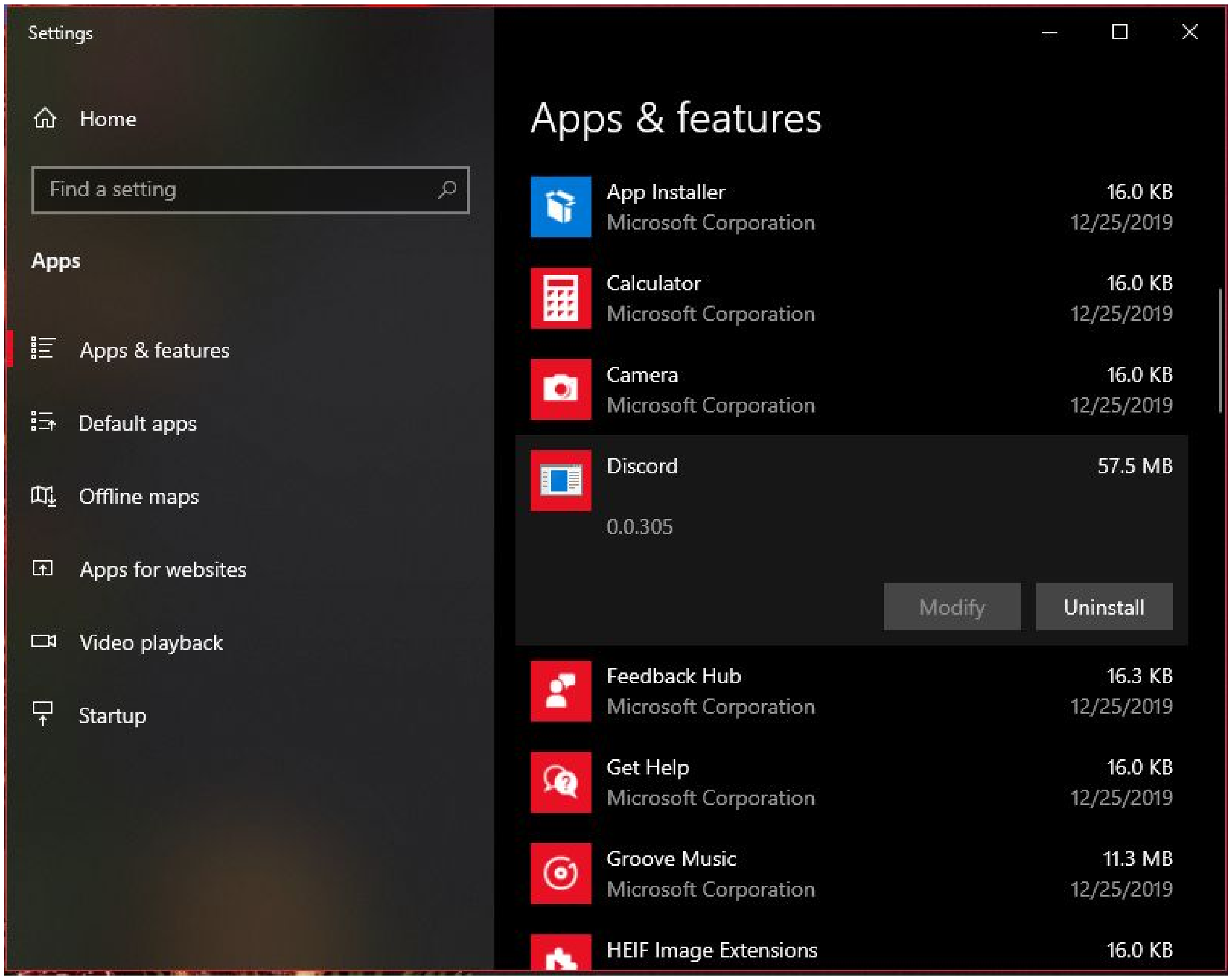


Apps & features

Microsoft Visual Studio Installer		12/23/2019
Mixed Reality Portal	Microsoft Corporation	16.0 KB 12/25/2019
Mobile Plans	Microsoft Corporation	68.0 KB 12/25/2019
Movies & TV	Microsoft Corporation	16.0 KB 12/25/2019
Mozilla Firefox 71.0 (x64 en-US)	71.0	192 MB 12/23/2019
Mozilla Maintenance Service		329 KB 12/23/2019
Office	Microsoft Corporation	1.72 MB 12/25/2019
OneNote		16.0 KB

Removed Discord

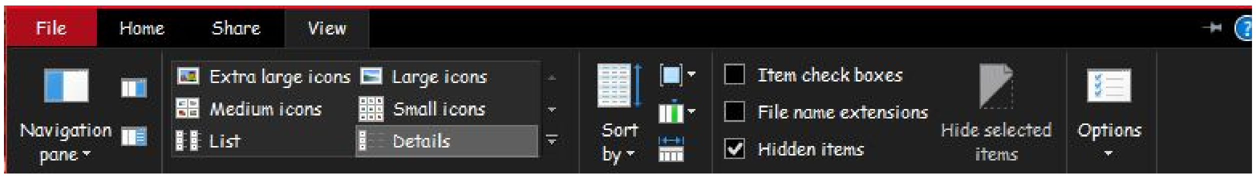
According to the README, the only authorized program that should remain on the system is BioniX. Everything else should be removed. In this case, you MUST uninstall Discord via Apps & Features AND remove artifacts from AppData.



Apps & features

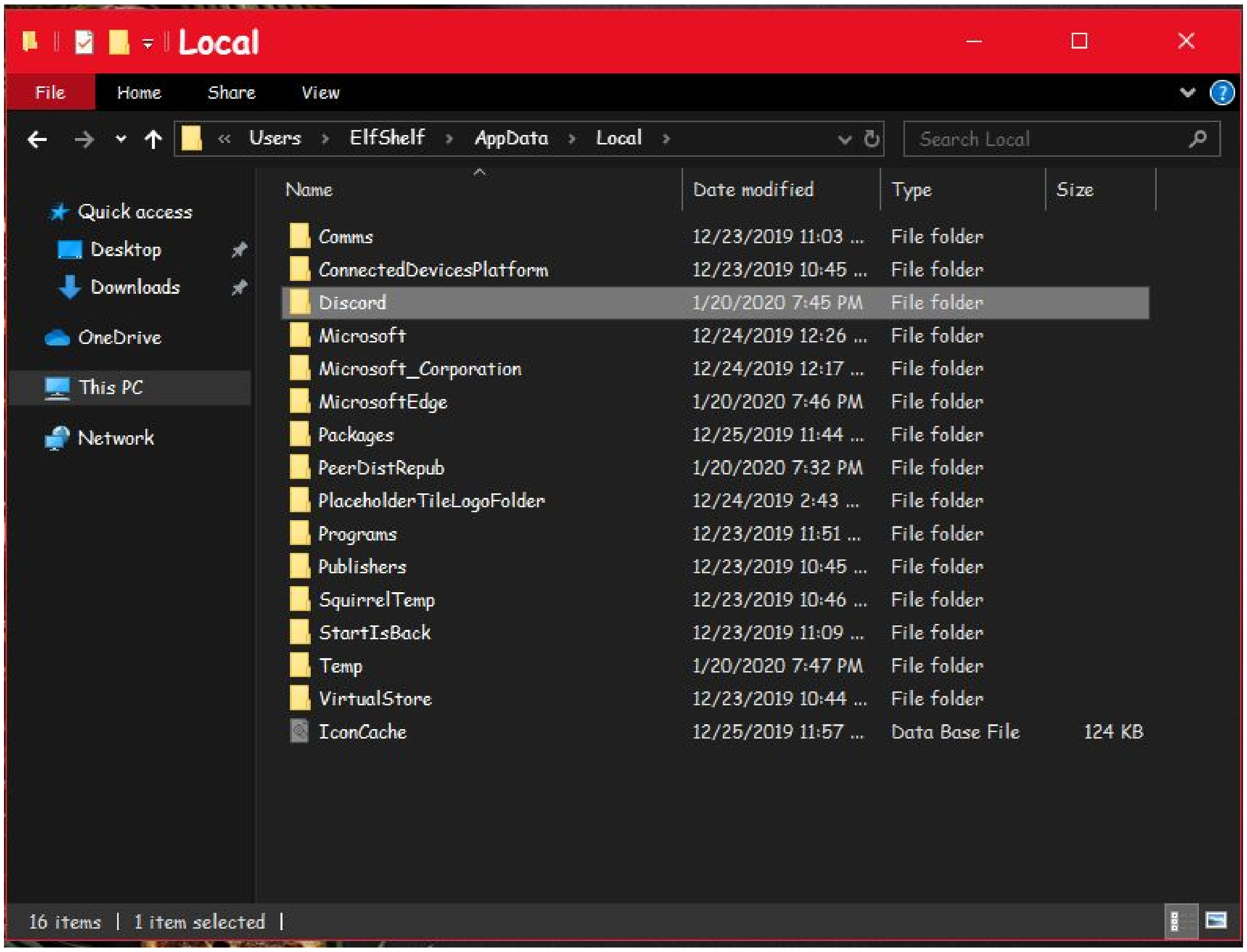
App Installer	Microsoft Corporation	16.0 KB 12/25/2019
Calculator	Microsoft Corporation	16.0 KB 12/25/2019
Camera	Microsoft Corporation	16.0 KB 12/25/2019
Discord	0.0.305	57.5 MB
Feedback Hub	Microsoft Corporation	16.3 KB 12/25/2019
Get Help	Microsoft Corporation	16.0 KB 12/25/2019
Groove Music	Microsoft Corporation	11.3 MB 12/25/2019
HEIF Image Extensions		16.0 KB

NOTE: You MUST enable Hidden items to find the Discord installation directory!



☐ Item check boxes
☐ File name extensions
✓ Hidden items

Remove *“C:\Users\ElfShelf\AppData\Local\Discord”*



Name	Date modified	Type	Size
Comms	12/23/2019 11:03 ...	File folder	
ConnectedDevicesPlatform	12/23/2019 10:45 ...	File folder	
Discord	1/20/2020 7:45 PM	File folder	
Microsoft	12/24/2019 12:26 ...	File folder	
Microsoft_Corporation	12/24/2019 12:17 ...	File folder	
MicrosoftEdge	1/20/2020 7:46 PM	File folder	
Packages	12/25/2019 11:44 ...	File folder	
PeerDistRepub	1/20/2020 7:32 PM	File folder	
PlaceholderTileLogoFolder	12/24/2019 2:43 ...	File folder	
Programs	12/23/2019 11:51 ...	File folder	
Publishers	12/23/2019 10:45 ...	File folder	
SquirrelTemp	12/23/2019 10:46 ...	File folder	
StartIsBack	12/23/2019 11:09 ...	File folder	
Temp	1/20/2020 7:47 PM	File folder	
VirtualStore	12/23/2019 10:44 ...	File folder	
IconCache	12/25/2019 11:57 ...	Data Base File	124 KB

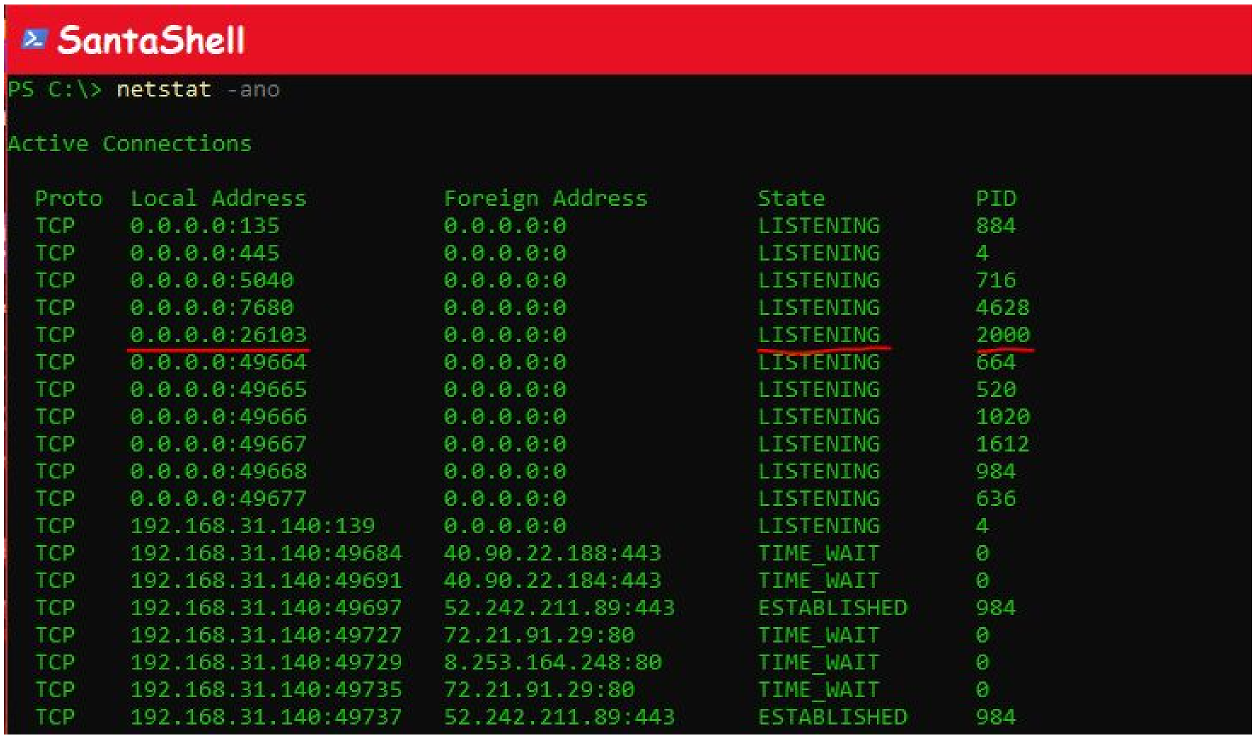
16 items | 1 item selected |

Persistence Mechanisms

Removed WinSMS backdoor and installation directory

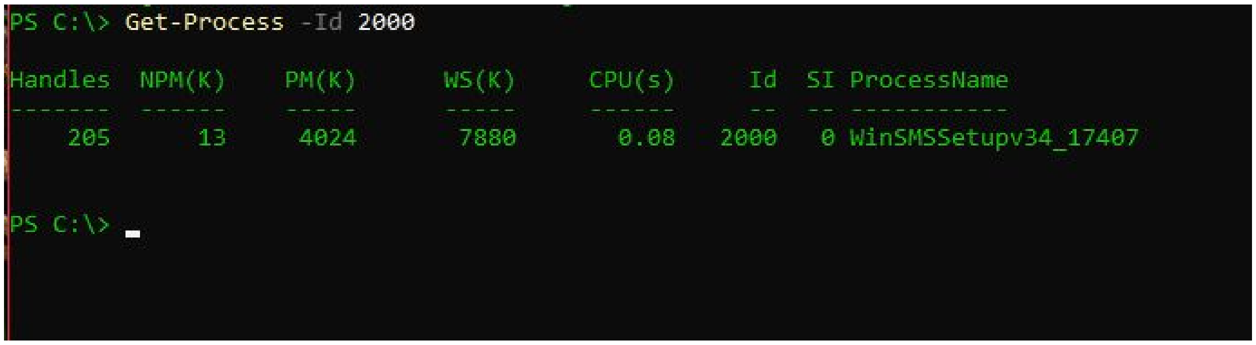
This check takes two steps. First, let's find the backdoor. In most cases, backdoors use reverse listening sockets to allow an attacker to connect to your machine. In this case, the WinSMS backdoor is a ***reverse shell*** backdoor, meaning that the attacker can remotely execute commands on this system.

If you pull up a PowerShell session, you can run the ***netstat*** command along with the arguments, ***-ano***, to view a list of listening socket connections (See netstat).



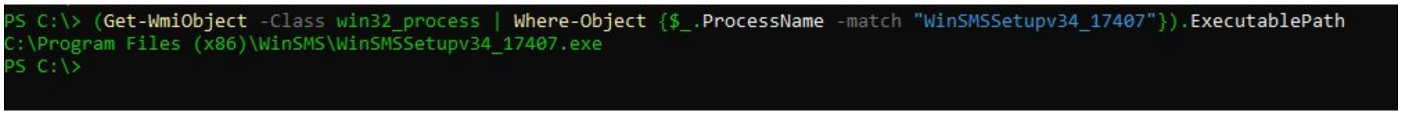
```
PS C:\> netstat -ano

Active Connections

  Proto  Local Address          Foreign Address        State           PID
  TCP    0.0.0.0:135            0.0.0.0:0              LISTENING       884
  TCP    0.0.0.0:445            0.0.0.0:0              LISTENING       4
  TCP    0.0.0.0:5040           0.0.0.0:0              LISTENING       716
  TCP    0.0.0.0:7680           0.0.0.0:0              LISTENING       4628
  TCP    0.0.0.0:26103          0.0.0.0:0              LISTENING       2000
  TCP    0.0.0.0:49664          0.0.0.0:0              LISTENING       664
  TCP    0.0.0.0:49665          0.0.0.0:0              LISTENING       520
  TCP    0.0.0.0:49666          0.0.0.0:0              LISTENING       1020
  TCP    0.0.0.0:49667          0.0.0.0:0              LISTENING       1612
  TCP    0.0.0.0:49668          0.0.0.0:0              LISTENING       984
  TCP    0.0.0.0:49677          0.0.0.0:0              LISTENING       636
  TCP    192.168.31.140:139     0.0.0.0:0              LISTENING       4
  TCP    192.168.31.140:49684   40.90.22.188:443       TIME_WAIT       0
  TCP    192.168.31.140:49691   40.90.22.184:443       TIME_WAIT       0
  TCP    192.168.31.140:49697   52.242.211.89:443      ESTABLISHED     984
  TCP    192.168.31.140:49727   72.21.91.29:80         TIME_WAIT       0
  TCP    192.168.31.140:49729   8.253.164.248:80       TIME_WAIT       0
  TCP    192.168.31.140:49735   72.21.91.29:80         TIME_WAIT       0
  TCP    192.168.31.140:49737   52.242.211.89:443      ESTABLISHED     984
```

Here, you can see what seems to be like normal connections established. However, there is one connection that is glaringly *odd* to say the least. There is a listening connection on port 26103! Not only is port 26103 extremely high, but also non-standard. We can also see that the PID (Process ID) is 2000. Keep in mind, this value will be different on your instance.



```
PS C:\> Get-Process -Id 2000

Handles  NPM(K)    PM(K)      WS(K)     CPU(s)     Id  SI ProcessName
-------  ------    -----      -----     ------     --  -- -----------
    205      13     4024       7880       0.08   2000   0 WinSMSSetupv34_17407


PS C:\> _
```

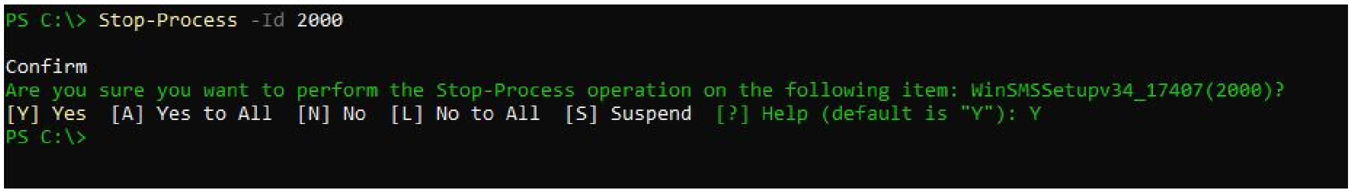
If we run *Get-Process* on it, you can see the name, WinSMSSetupv34_17407. However, this is just a name. This executable can reside anywhere on the system. Let's investigate further…



```
PS C:\> (Get-WmiObject -Class win32_process | Where-Object {$_.ProcessName -match "WinSMSSetupv34_17407"}).ExecutablePath
C:\Program Files (x86)\WinSMS\WinSMSSetupv34_17407.exe
PS C:\>
```

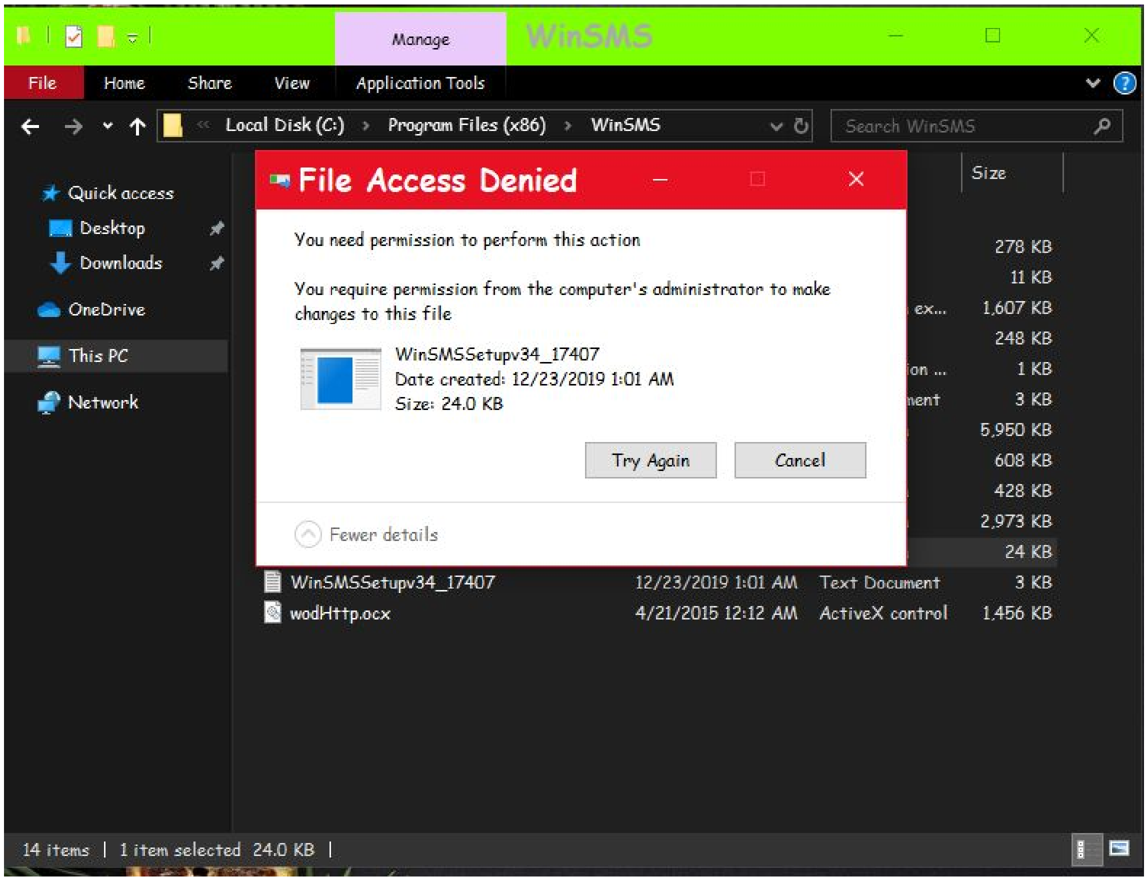
Running this *Get-WmiObject* query in PowerShell, we can see the executable path of the backdoor. Now, let's kill it!

First, we'll kill the process using the *Stop-Process* cmdlet.



```
PS C:\> Stop-Process -Id 2000

Confirm
Are you sure you want to perform the Stop-Process operation on the following item: WinSMSSetupv34_17407(2000)?
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is "Y"): Y
PS C:\>
```

Then, we can delete it!



File Access Denied

You need permission to perform this action

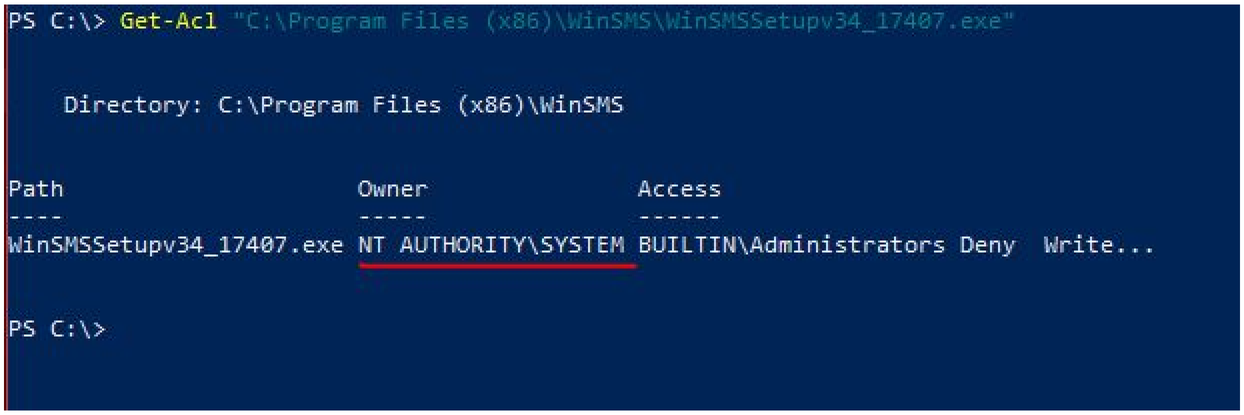
You require permission from the computer's administrator to make changes to this file

WinSMSSetupv34_17407
Date created: 12/23/2019 1:01 AM
Size: 24.0 KB

```
WinSMSSetupv34_17407   12/23/2019 1:01 AM   Text Document      3 KB
wodHttp.ocx            4/21/2015 12:12 AM   ActiveX control  1,456 KB
```

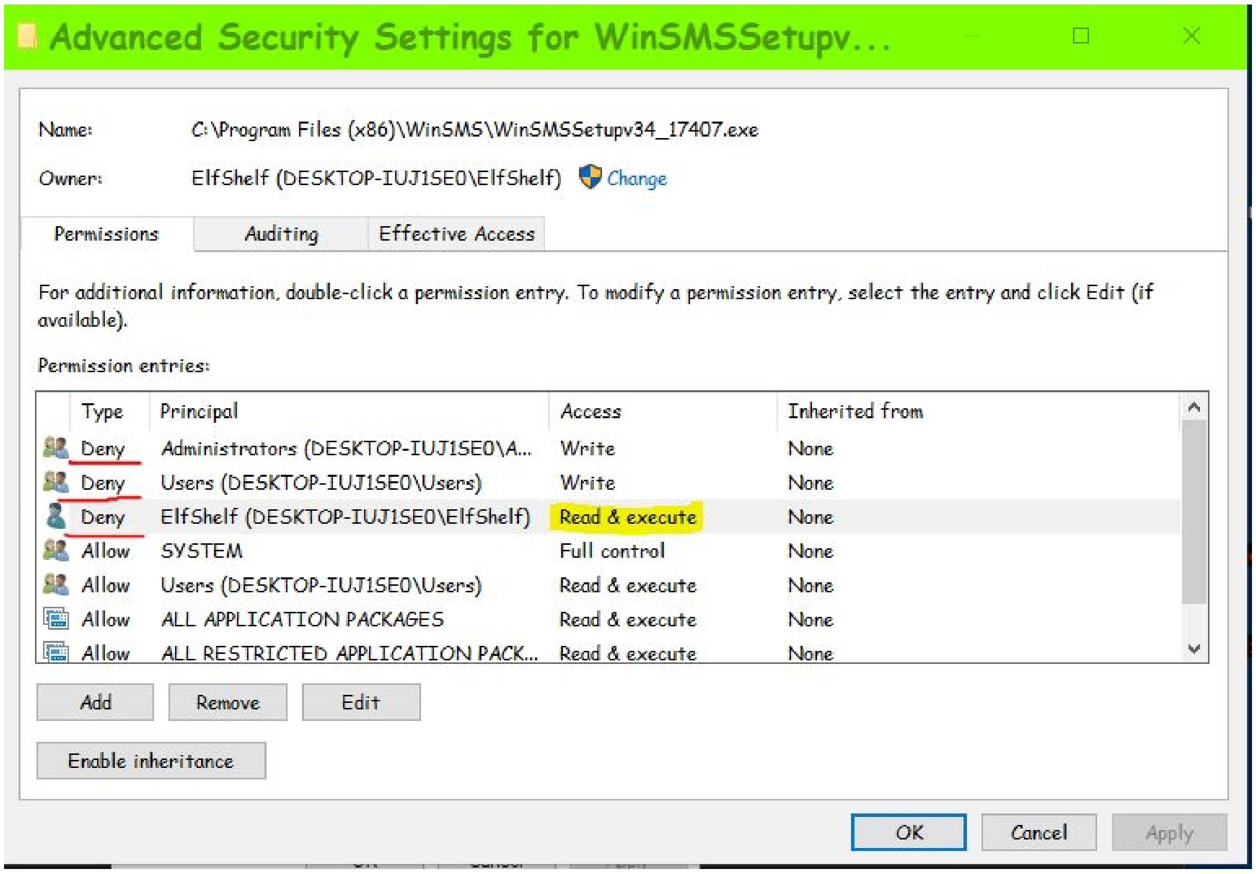
14 items | 1 item selected 24.0 KB |

Well… Not quite yet! If you look at the permissions of this executable, you will notice that it is owned by ‘*nt authority\system’.*



```
PS C:\> Get-Acl "C:\Program Files (x86)\WinSMS\WinSMSSetupv34_17407.exe"


    Directory: C:\Program Files (x86)\WinSMS


Path                      Owner               Access
----                      -----               ------
WinSMSSetupv34_17407.exe NT AUTHORITY\SYSTEM BUILTIN\Administrators Deny  Write...


PS C:\>
```

To get around this, we must take ownership of the file. Then, we can view the permissions of it.


Advanced Security Settings for WinSMSSetupv...

Name: C:\Program Files (x86)\WinSMS\WinSMSSetupv34_17407.exe

Owner: ElfShelf (DESKTOP-IUJ1SE0\ElfShelf) Change

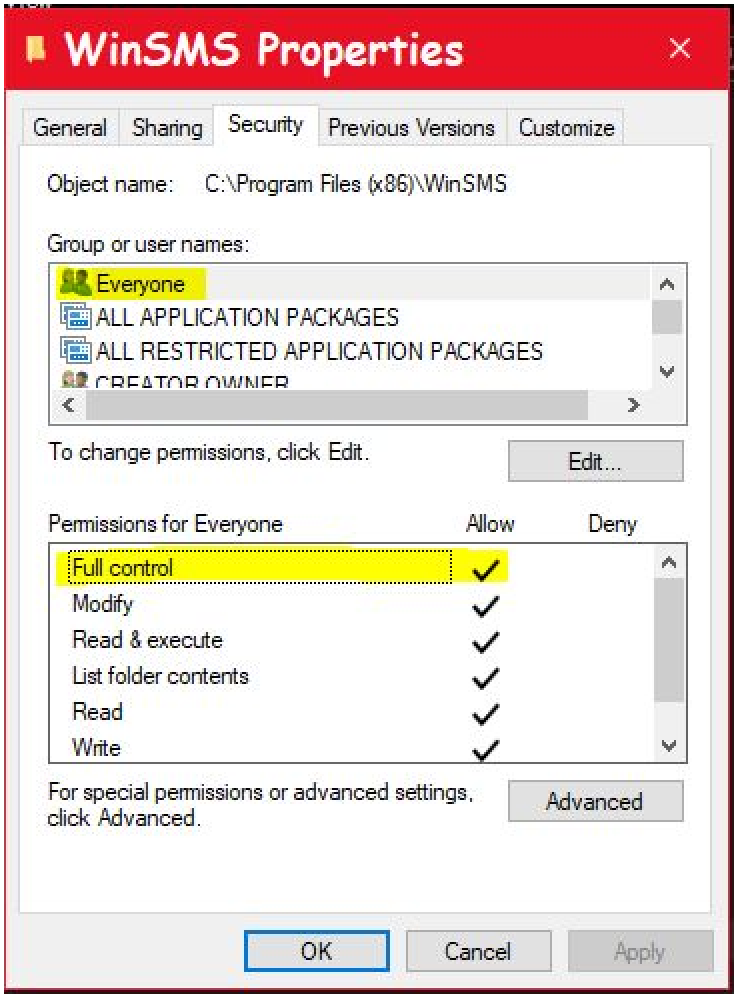
For additional information, double-click a permission entry. To modify a permission entry, select the entry and click Edit (if available).

Permission entries:

Type	Principal	Access	Inherited from
Deny	Administrators (DESKTOP-IUJ1SE0\A...	Write	None
Deny	Users (DESKTOP-IUJ1SE0\Users)	Write	None
Deny	ElfShelf (DESKTOP-IUJ1SE0\ElfShelf)	Read & execute	None
Allow	SYSTEM	Full control	None
Allow	Users (DESKTOP-IUJ1SE0\Users)	Read & execute	None
Allow	ALL APPLICATION PACKAGES	Read & execute	None
Allow	ALL RESTRICTED APPLICATION PACK...	Read & execute	None

After going through the hell of circumventing obstacles, we can see that our user is being explicitly being denied 'Read & execute' permissions on this file. We can also see two other deny statements, preventing Users and Administrators from writing to this file. If we remove these ACEs (Access Control Entries), we can then delete the file. Now we need to completely remove WinSMS, as it is insecure. See the following.

If you spent any time looking at the SDDL string for the installation directory of WinSMS or looked at the 'Security' tab in the properties of the directory, you should've seen a gaping security hole. The 'Everyone' group has "Full Control" over the WinSMS installation directory. This means any user can overwrite the contents of this application, allowing arbitrary code execution and/or privilege escalation.


WinSMS Properties

General | Sharing | Security | Previous Versions | Customize

Object name: C:\Program Files (x86)\WinSMS

Group or user names:

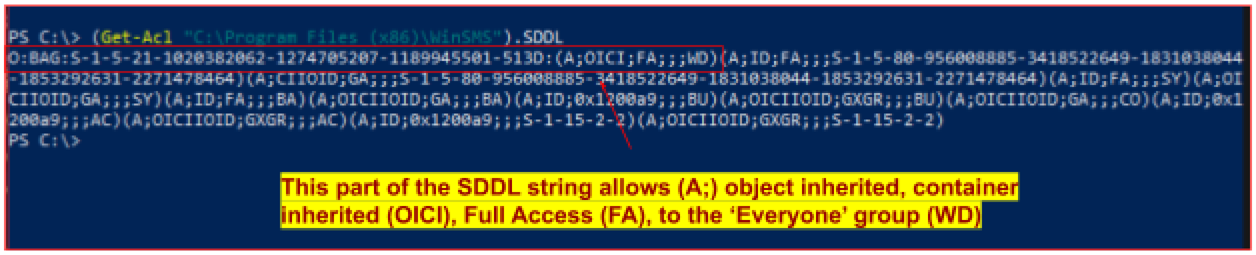
- Everyone
- ALL APPLICATION PACKAGES
- ALL RESTRICTED APPLICATION PACKAGES
- CREATOR OWNER

To change permissions, click Edit. [Edit...]

Permissions for Everyone	Allow	Deny
Full control	✓	
Modify	✓	
Read & execute	✓	
List folder contents	✓	
Read	✓	
Write	✓	

For special permissions or advanced settings, click Advanced. [Advanced]

[OK] [Cancel] [Apply]


```
PS C:\> (Get-Acl "C:\Program Files (x86)\WinSMS").SDDL
O:BAG:S-1-5-21-1020382062-1274705207-1189945501-513D:(A;OICI;FA;;;WD)(A;ID;FA;;;S-1-5-80-956008885-3418522649-1831038044
-1853292631-2271478464)(A;CIIOID;GA;;;S-1-5-80-956008885-3418522649-1831038044-1853292631-2271478464)(A;ID;FA;;;SY)(A;OI
CIIOID;GA;;;SY)(A;ID;FA;;;BA)(A;OICIIOID;GA;;;BA)(A;ID;0x1200a9;;;BU)(A;OICIIOID;GXGR;;;BU)(A;OICIIOID;GA;;;CO)(A;ID;0x1
200a9;;;AC)(A;OICIIOID;GXGR;;;AC)(A;ID;0x1200a9;;;S-1-15-2-2)(A;OICIIOID;GXGR;;;S-1-15-2-2)
PS C:\>
```

This part of the SDDL string allows (A;) object inherited, container inherited (OICI), Full Access (FA), to the ‘Everyone’ group (WD)

<u>References:</u>

https://docs.microsoft.com/en-us/windows/win32/secauthz/security-descriptor-definition-language
https://jorgequestforknowledge.wordpress.com/2008/03/26/parsing-sddl-strings/

Security Policies

Built-in Administrator account is disabled

Because this system has delegated Administrative access, the built-in Administrator account is not necessary and should be disabled. If it were enabled and not configured properly, an attacker could leverage this account to perform administrative tasks.

Audit Logon Events [Success, Failure]

Enabling this setting to audit logon successes AND failures allows for both of these events to be triggered in the event log. This is crucial for auditing purposes. If a user is attempting to bruteforce the password of another user, an Administrator can see that based on the number of failed logon events in the event log. If a user successfully logs in, the Administrator can determine *when* that user logged in.

Password complexity requirements are enforced

Enabling password complexity requirements forces users to create strong passwords. (Must meet minimum password length requirements, have uppercase and lowercase letters, numbers, and special characters).

Do not require Ctrl+Alt+Del' is disabled

This setting determines whether a user has to press ctrl alt delete before they can log on. We should disable this policy to force them to press ctrl alt delete. This is to prevent exploitation of login shells. Back when interactive services were allowed, you could apply a service at boot and have users type in their credentials into a fake login screen. However, if you type ctrl alt delete, it kills everything, and doesn't allow windows services.

Do not display last signed-in is enabled

On a shared workstation, each user is entitled to their own privacy and security, despite the workstation being shared. If a user can see who logged in before them, the likelihood of a targeted attack towards that person may increase. For the purposes of this machine being in a work environment, users should only attempt to login as themselves and should not be able to see who logged in before them.

Microsoft Network Client: Digitally Sign Communications (Always)

As mentioned in forensics question #1, SMB was a network service that was used to share files. While this machine was used as an SMB server, both servers and clients should ultimately be treated as clients in general. When this setting is enabled, all communications via SMB to other servers must be digitally signed to prevent man-in-the-middle attacks.

Restrict blank password usage at console logon

Enabling this setting prevents users from creating user accounts with blank passwords. However, if there are any users with configured blank passwords, enabling this setting will restrict the logon of these users to the interactive console, so you must physically login at the login screen