The Conclusive Netsh - Ultimate Guide

https://adamtheautomator.com/netsh/#Diagnosing_the_Windows_Firewall_and_IPsec_Events

Contents

The Conclusive Netsh : Ultimate Guide 1
Prerequisites 1
What is Network Shell (Netsh)? 1
Running Netsh Commands 1
Using Netsh Utility Commands 3
Fetching and Executing Netsh Utility Commands from a Text File 5
Executing Netsh Commands Under a User Account 5
Listing Available Network Interfaces 5
Gathering Information on Network Interfaces 6
Setting a Static IPv4 Address and DNS Server 7
Managing Wired Network Interfaces 10
Managing the Windows Firewall 11
Capturing Network Traces 11
Diagnosing the Windows Firewall and IPsec Events 12
Validating Incoming and Outgoing Traffic 13
Working with Alias and Unalias Using Netsh Command 14
Reserving a Uniform Resource Locator (URL) 14
Managing Netsh Configuration and Query Commands 15
Managing HTTP System Settings 17
Automating Netsh Commands Execution with Batch Scripts 18
Conclusion 19

Have you ever run into an issue on a Windows machine with network connectivity? Maybe it was some rogue software installer making unknown changes to registry keys, or perhaps you suspect a virus is installed and need to track down its activity. Luckily Network Shell (netsh) utility is just around to help.

Netsh is a command-line utility that allows you to configure and display the status of various network configurations of Windows machines or servers. And in this tutorial, you'll learn just about every command-line feature Netsh provides.

Ready? Read on and keep your network connectivity at its peak!

Prerequisites

This tutorial comprises hands-on demonstrations. But so long as you have a Windows PC or server, you're good to go – This tutorial uses Windows 2019 Datacenter Edition, but Netsh works in all modern Windows editions.

What is Network Shell (Netsh)?

Before you dive into executing your first `netsh` command, kick-off this tutorial by getting a high-level overview of Netsh. So, what is Netsh anyway?

You might have worked with dozens of networking command-line tools such as ping, tracert, telnet, etc., but they are bound for a specific purpose.

With the Netsh command-line utility, you can configure and work with various network components. These network components are not limited to network interfaces, Windows firewalls, server roles, etc., on computers running Windows Server.

Running Netsh Commands

You now have a basic idea of Netsh and its usefulness for monitoring and configuring your network. But how do you actually execute `netsh` commands?

You can execute `netsh` commands on command-line consoles like PowerShell or the command prompt on a Windows machine. But as the command prompt's successor, this tutorial demonstrates running the `netsh` commands on PowerShell.

1. Log in to your Windows Server using your RDP client. Windows has a default RDP client installed.

Related:The Top Free Remote Desktop Connection Managers for Windows

2. After you log in, open PowerShell as administrator, and run the `netsh` command below to access the **netsh** command-line session.

`Netsh`

```
PS C:\Users\Administrator> netsh
netsh>
netsh>help

The following commands are available:

Commands in this context:
..              - Goes up one context level.
?               - Displays a list of commands.
abort           - Discards changes made while in offline mode.
```

Accessing netsh Command-line Session

3. In the **netsh** prompt, run the `help` command to see all commands you can use inside your netsh command-line session. `help`

`help`

Pick one command you'd like to run, but this tutorial uses the `interface` command.

```
netsh>help

The following commands are available:

Commands in this context:
..              - Goes up one context level.
?               - Displays a list of commands.
abort           - Discards changes made while in offline mode.
add             - Adds a configuration entry to a list of entries.
advfirewall     - Changes to the `netsh advfirewall' context.
alias           - Adds an alias.
branchcache     - Changes to the `netsh branchcache' context.
bridge          - Changes to the `netsh bridge' context.
bye             - Exits the program.
commit          - Commits changes made while in offline mode.
delete          - Deletes a configuration entry from a list of entries.
dhcpclient      - Changes to the `netsh dhcpclient' context.
dnsclient       - Changes to the `netsh dnsclient' context.
dump            - Displays a configuration script.
exec            - Runs a script file.
exit            - Exits the program.
firewall        - Changes to the `netsh firewall' context.
help            - Displays a list of commands.
http            - Changes to the `netsh http' context.
interface       - Changes to the `netsh interface' context.
ipsec           - Changes to the `netsh ipsec' context.
lan             - Changes to the `netsh lan' context.
mbn             - Changes to the `netsh mbn' context.
namespace       - Changes to the `netsh namespace' context.
```

Help

4. Lastly, run the `interface` command below to show all available network interfaces. `interface show interface`

`interface show interface`

```
netsh>interface show interface

Admin State    State          Type             Interface Name
-------------------------------------------------------------------------
Enabled        Connected      Dedicated        Ethernet
Enabled        Connected      Dedicated
Enabled        Connected      Dedicated
Enabled        Connected      Dedicated        Wi-Fi

netsh>
```

Showing All Available Network Interfaces

Using Netsh Utility Commands

You now have access to the **netsh** prompt, and you're almost ready to manage your network connectivity. But before you dive deeper into Netsh, you'll be running basic Netsh utility commands to familiarize yourself with how Netsh works.

There are two ways you can run Nesh utility commands as follows:

- Run the `netsh` command alone to access the **netsh** prompt, then run the Netsh utility command.

```
netsh
help
```

- Running the `netsh` command followed by the Netsh utility command.

```
netsh help
```

Like the `help` command you learned in step three of the "Running Netsh Commands" section, run some other Netsh utility commands below that Netsh offers:

- `/?` – Similar to `help` but commonly used for most of the commands you can use within Netsh utility.

```
netsh>/?

The following commands are available:

Commands in this context:
..              - Goes up one context level.
?               - Displays a list of commands.
abort           - Discards changes made while in offline mode.
add             - Adds a configuration entry to a list of entries.
advfirewall     - Changes to the `netsh advfirewall' context.
alias           - Adds an alias.
branchcache     - Changes to the `netsh branchcache' context.
bridge          - Changes to the `netsh bridge' context.
bye             - Exits the program.
commit          - Commits changes made while in offline mode.
delete          - Deletes a configuration entry from a list of entries.
dhcpclient      - Changes to the `netsh dhcpclient' context.
dnsclient       - Changes to the `netsh dnsclient' context.
dump            - Displays a configuration script.
exec            - Runs a script file.
exit            - Exits the program.
firewall        - Changes to the `netsh firewall' context.
help            - Displays a list of commands.
http            - Changes to the `netsh http' context.
```

Displaying all the display all the commands and context using netsh /?

- `show` – This command displays the commands available with each context.

```
netsh>show

The following commands are available:

Commands in this context:
show alias      - Lists all defined aliases.
show helper     - Lists all the top-level helpers.
show mode       - Shows the current mode.
netsh>
```

Showing Available Commands in a Context

popd allows you to restore a context from a first-in-last-out (FILO) stack, while pushd pushes the current context on a stack.

Both commands don't have outputs, but you can append the help command to see how these commands function, as shown below.

```
popd help
pushd help
```

```
netsh>popd help

Usage : popd

        Pops a context from the stack.


netsh>pushd help

Usage : pushd

        Pushes current context on stack.


netsh>
```

Popping and Pushing Stack

- The commit command commits changes made to the running configuration while in offline mode. In contrast, the abort command discards changes made in the running configuration while in offline mode.

These commands don't have output, but appending the help command lets you view the command's usage.

```
commit help
abort help
```

```
netsh>commit help

Usage: commit

Remarks:
       Commits changes to the running configuration that were made while in
       offline mode. No action is taken for this command while in online
       mode.

netsh>abort help

Usage: abort

Remarks:
       Discards changes made while in the offline mode. This does not affect
       changes made while in online mode.

netsh>
```

Committing and Aborting Changes to the Running Configuration

- quit and bye – Lets you exit out of Netsh utility.

```
netsh>quit

PS >
```

Quitting the Netsh Utility (quit)

```
netsh>bye

PS >
```

Quitting the Netsh Utility (bye)

Fetching and Executing Netsh Utility Commands from a Text File

Like every other command, the `netsh` command allows you to declare parameters while executing tasks for simplicity and flexibility.

When working with Windows network configurations, you'll find yourself dealing with Netsh. And using parameters in your `netsh` commands comes in handy. One good example is calling upon commands from a dedicated source, like a text file.

Create a file named *myfile.txt* on your desktop and add the `help` command in the file.

Now, run the `netsh` command below with the `-a` flag to specify the file's path containing the `netsh` command.

```
netsh -a C:\Users\Administrator\Desktop\myfile.txt
```

Below, you can see the `netsh` command called and executed the `help` command from the *myfile.txt* file.

If you wish to run the command on a remote windows machine, consider using the `-r` flag, as shown below. But make sure the Remote Registry service runs on the remote computer. netsh -r shanky

```
PS C:\Users\Administrator> netsh -a C:\Users\Administrator\Desktop\myfile.txt

The following commands are available:

Commands in this context:
..              - Goes up one context level.
?               - Displays a list of commands.
abort           - Discards changes made while in offline mode.
add             - Adds a configuration entry to a list of entries.
advfirewall     - Changes to the `netsh advfirewall' context.
```

Fetching a Netsh Command from a Text File

Executing Netsh Commands Under a User Account

If you like to keep track of changes you make to your network configuration, run Netsh commands under a user account. How? By using the `-u` parameter followed by the account's username.

Run the following `netsh` command to list available network interfaces under a user account (`-u`) named `shanky`.

To specify the user's password, append the `-p` parameter followed by the password.

```
netsh -u shanky -p hellopass interface show interface
```

```
PS C:\Users> netsh  -u shanky -p hellopass interface show interface

Admin State    State          Type             Interface Name
-------------------------------------------------------------------------
Enabled        Connected      Dedicated        Ethernet
```

Running `netsh` Commands Under a Different User Account

If you need to run the `netsh` command under a user account from a specific domain, consider using the `DomainName\\username` parameter.

Listing Available Network Interfaces

A network interface connects your computer and a private or public network. And knowing which interface your machine uses and the network configurations to troubleshoot is crucial.

This information helps resolve issues or provide your network configurations to third-party vendors for support.

1. Execute the `netsh interface` command below to `show` all available interfaces currently present on your machine.

`Running netsh Commands Under a Different User Account`

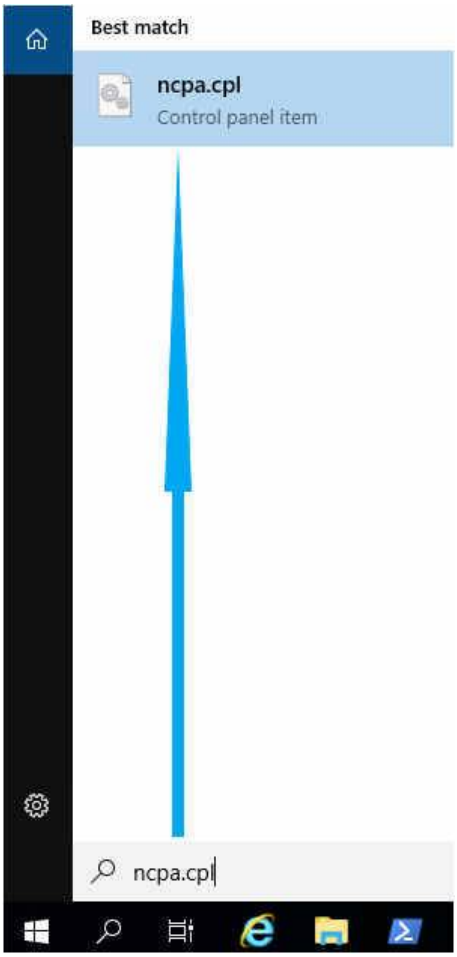
As you can see below, only one interface (**Ethernet**) takes care of all the network flow (in/out) from your machine.

```
PS C:\Users\Administrator> netsh interface show interface

Admin State    State          Type             Interface Name
-------------------------------------------------------------------------
Enabled        Connected      Dedicated        Ethernet
```

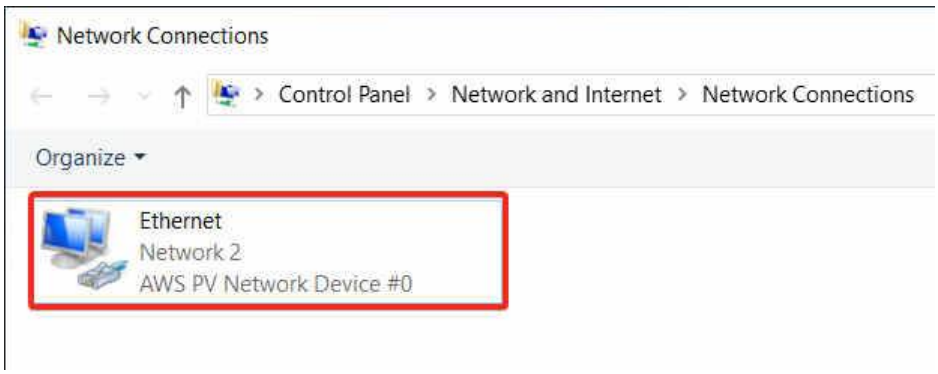
Listing All Network Interfaces

2. Now, click the start menu, type **ncpa.cpl**, as shown below, and press Enter to open the Network Connections window (step three).



Opening Network Connections

3. Finally, verify if the `netsh` command returned the correct details of interfaces. Below, you can see the name of the active interface is **Ethernet**.



Verifying Network Interfaces

Gathering Information on Network Interfaces

What other good things come with listing available network interfaces on your system? Dig deeper to get comprehensive information from your network interfaces. Why? In a large organization, other departments require those details if they need to host applications on your network.

Run the following command that works the same as listing (`show`) each network `interface` available. But this time, you'll get more details about each network interface configuration (`config`), such as **DHCP** status, **IP address**, etc.

Related:How to Manage Microsoft DHCP Server [In Depth Tutorial]

```
netsh interface ipv4 show config
```

```
PS C:\Users\Administrator> netsh interface ipv4 show config

Configuration for interface "Ethernet"
    DHCP enabled:                         Yes
    IP Address:                           172.31.40.134
    Subnet Prefix:                        172.31.32.0/20 (mask 255.255.240.0)
    Default Gateway:                      172.31.32.1
    Gateway Metric:                       0
    InterfaceMetric:                      25
    DNS servers configured through DHCP:  172.31.0.2
    Register with which suffix:           Primary only
    WINS servers configured through DHCP: None

Configuration for interface "Loopback Pseudo-Interface 1"
    DHCP enabled:                         No
    IP Address:                           127.0.0.1
    Subnet Prefix:                        127.0.0.0/8 (mask 255.0.0.0)
    InterfaceMetric:                      75
    Statically Configured DNS Servers:    None
    Register with which suffix:           Primary only
    Statically Configured WINS Servers:   None
```

Checking Information on Available Network Interfaces

*The **Loopback Pseudo-Interface** is used mainly for troubleshooting purposes and allows you to connect to servers running on the local machine for testing. But note that the loopback interface does not represent any actual hardware.*

Setting a Static IPv4 Address and DNS Server

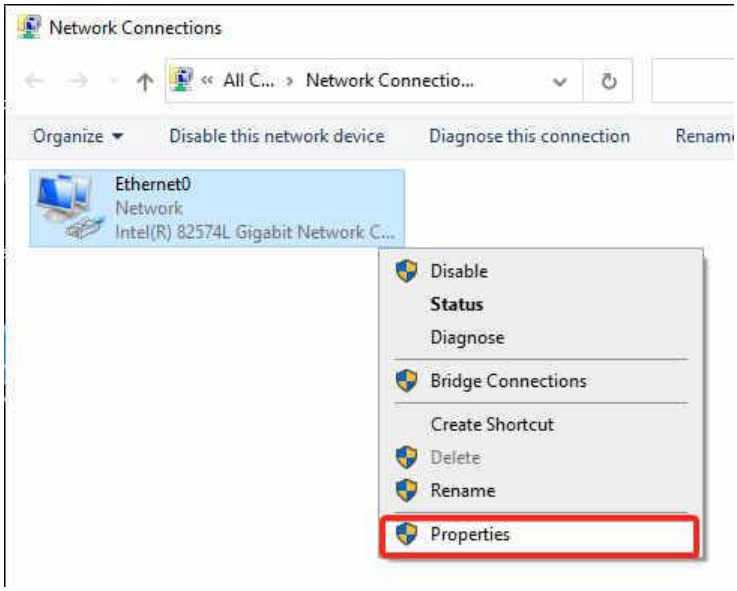
From the detailed listing of available interfaces, one of the most important concepts is the IP address of the Windows machine. The IP address for the Ethernet network interface is Dynamic. As a result, the IP address is allocated by itself from the pool of available IP addresses of the subnet.

Related:How to Use PowerShell to Get an IP Address

But at times, you need to have a static IP address allocated to your machine, such as to test applications where a reboot of the machine is required. So change the dynamic IP address settings to manual type where IPs remain persistent.

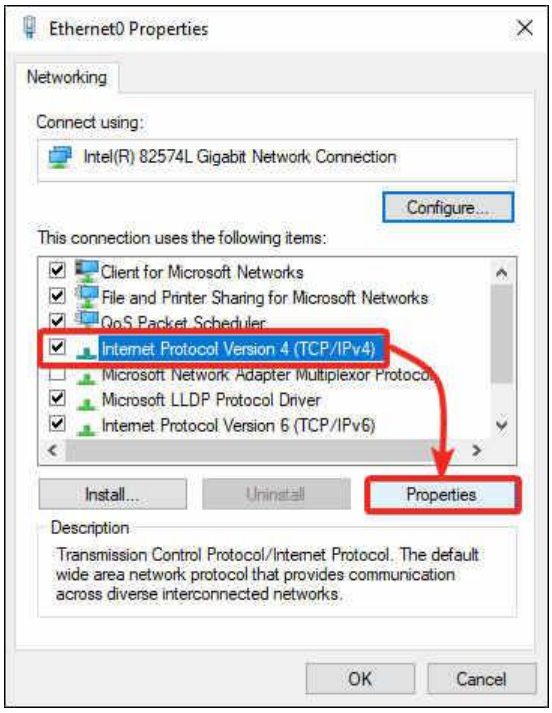
1. Open the **Network Connections** window to see available network interfaces.

2. Next, right-click on the Ethernet interface and select properties, opening Ethernet's **Properties** window.



Ethernet Properties

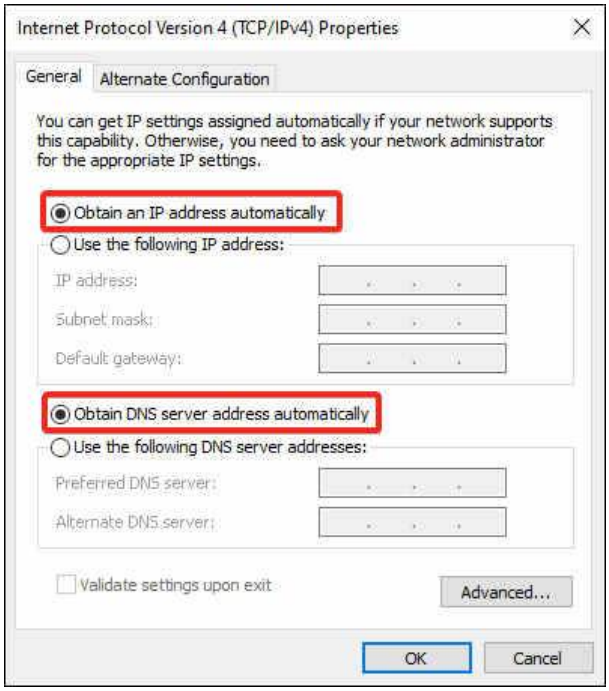
3. Select **Internet Protocol Version 4 (TCP/IPv4)**, as shown below, and click on **Properties** to open the IPV4's properties window.



Checking the IPv4 Properties

As you can see below, the IP settings are set to obtain an IP address automatically.

Click on **Cancel** to close the **Properties** window, and you'll manually change the IP address in the following step.



Viewing IPv4 Address Settings

4. Now, run the below `netsh interface` command to manually set the following IP address settings:

- IP Address – `10.0.0.100`
- Subnet mask – `255.255.0.0`
- Default gateway – `10.0.0.1`

```
netsh interface ip set address "Ethernet" static 10.0.0.100 255.255.0.0 10.0.0.1
```

To change the IPv6 address of the Ethernet interface, you will need to use `netsh int ipv6 set to address 7 2001::2`

To roll back to using a dynamic IP address, run the following command: `netsh interface ip set address "Ethernet" dhcp`

5. Run the following command to define a DNS server. Doing so allows your Windows machine to communicate to other networks.

```
netsh interface ip set address "Ethernet" static 10.0.0.100 255.255.0.0 10.0.0.1
```

To disable the Ethernet interface on your Windows system, consider running the following command: `netsh interface set interface name="Ethernet" admin=disabled`

6. Lastly, as shown below, open the IPv4 settings again as you did in steps two to three to verify the interface's IP address settings changed.

Verifying IPv4 Address and DNS Settings

Managing Wired Network Interfaces

You previously managed the Ethernet network interface, but there could be multiple interfaces, such as wireless, wired, and Bluetooth-based network interfaces. What if you need information only for wired network interfaces as they are most important?

Consider using Netsh commands for wired LAN that provide methods to configure connectivity and security settings. The Netsh `lan` commands configure the local computer or multiple computers using a logon script.

Run the `netsh lan` command below to list all the wired network interfaces available on your Windows machine.

```
netsh lan show interfaces
```

```
PS C:\Users\Administrator> netsh lan show interfaces

There is 1 interface on the system:

    Name                   : Ethernet
    Description            : AWS PV Network Device #0
    GUID                   : 54b31d7e-36bf-4bbe-9ab2-106a939cd78c
    Physical Address       : 0E-BB-08-22-75-2D
    State                  : Attempting to authenticate
```

Listing All Wired Network Interfaces

Now, run the below command if you need to display the list of wired profiles configured on the computer.

If the interface parameter is defined, only the profile contents for the specified interface are displayed. Otherwise, all profiles will be displayed with their name and description.

```
netsh lan show profiles interface ="Ethernet"
```

```
PS C:\Users\Administrator> netsh lan show profiles interface = "Ethernet"

Profile on interface Ethernet
===========================================================================
Applied: User Profile

    Profile Version         : 1
    Type                    : Wired LAN
    AutoConfig Version      : 1
    802.1x                  : Enabled
    802.1x                  : Not Enforced
    EAP type                : Microsoft: Protected EAP (PEAP)
    802.1X auth credential  : [Profile credential not valid]
    Cache user information  : [Yes]
```

Displaying the List of Wired Profiles

Managing the Windows Firewall

Securing your network is a top priority achievable by managing your Windows firewall. How? By adding firewall rules with the Netsh `advfirewall` command. The `advfirewall` command allows you to control and manage your Windows firewall.

The `advfirewall` context works mainly on three profile settings; domain, private and public

Run the below command to enable (`action=allow`) the ping protocol (`icmpv4`) from your Windows firewall by setting a rule named `ALL ICMP V4`.

```
netsh advfirewall firewall add rule name="All ICMP V4" dir=in action=allow protocol=icmpv4
```

```
PS C:\Users\Administrator> netsh advfirewall firewall add rule name="All ICMP V4" dir=in action=allow protocol=icmpv4
Ok.
```

Adding Firewall Rule to Allow Ping Protocol

Now, run the below command to add a rule named `Open Port 80` on your machine.

You can choose any port that you'd like to open. But this command opens port 80, allowing traffic to flow in and out.

```
netsh advfirewall firewall add rule name= "Open Port 80" dir=in action=allow protocol=TCP localport=80
```

```
PS C:\Users\Administrator> netsh advfirewall firewall add rule name= "Open Port 80" dir=in action=allow protocol=TCP l
ocalport=80
Ok.
```

Opening Ports by Adding Firewall Rule

The `advfirewall` context provides the same functionality that the `netsh firewall` context gives. But the `netsh firewall` context might be deprecated in the future.

Capturing Network Traces

Trace is another crucial context to troubleshoot and collect event information regarding network connections. You can use the `trace` context to collect event information using Event Tracing for Windows (ETW) to log network events.

But first, you'll have to identify scenarios you can trace your network connections.

You can use Netsh to collect a packet capture without installing third-party tools like Wireshark.

Related:[How to Use Netsh Trace and PowerShell for Packet Capturing]

Run the command below to help you find available scenarios where tracing your network connections would work based on the issue you're dealing with in your network.

```
netsh trace show scenarios
```

As you can see below, multiple scenarios are available to work with `trace`.

```
PS C:\Users\Administrator> netsh trace show scenarios

Available scenarios (14):
---------------------------------------------------------------
AddressAcquisition       : Troubleshoot address acquisition-related issues
AddressAcquisitionServer : Troubleshoot address acquisition server related issues
DirectAccess             : Troubleshoot DirectAccess related issues
FileSharing              : Troubleshoot common file and printer sharing problems
InternetClient           : Diagnose web connectivity issues
InternetServer           : Set of HTTP service counters
L2SEC                    : Troubleshoot layer 2 authentication related issues
LAN                      : Troubleshoot wired LAN related issues
Layer2                   : Troubleshoot layer 2 connectivity related issues
NDIS                     : Troubleshoot network adapter related issues
NetConnection            : Troubleshoot issues with network connections
NetworkSnapshot          : Collect the current network state of the system
WFP-IPsec                : Troubleshoot Windows Filtering Platform and IPsec related issues
WLAN                     : Troubleshoot wireless LAN related issues
```

Checking Scenarios Available in Netsh Trace Context

Now, execute the below `netsh trace start` command to capture (`capture=yes`) all network details to a specified ETL file (`tracefile`) called `ata_trace.etl`. The `persistent` parameter permanently stores the file in the specified location when the value is true (`yes`).

```
netsh trace start capture=yes tracefile=c:\ata_trace.etl persistent=yes maxsize=4096
```

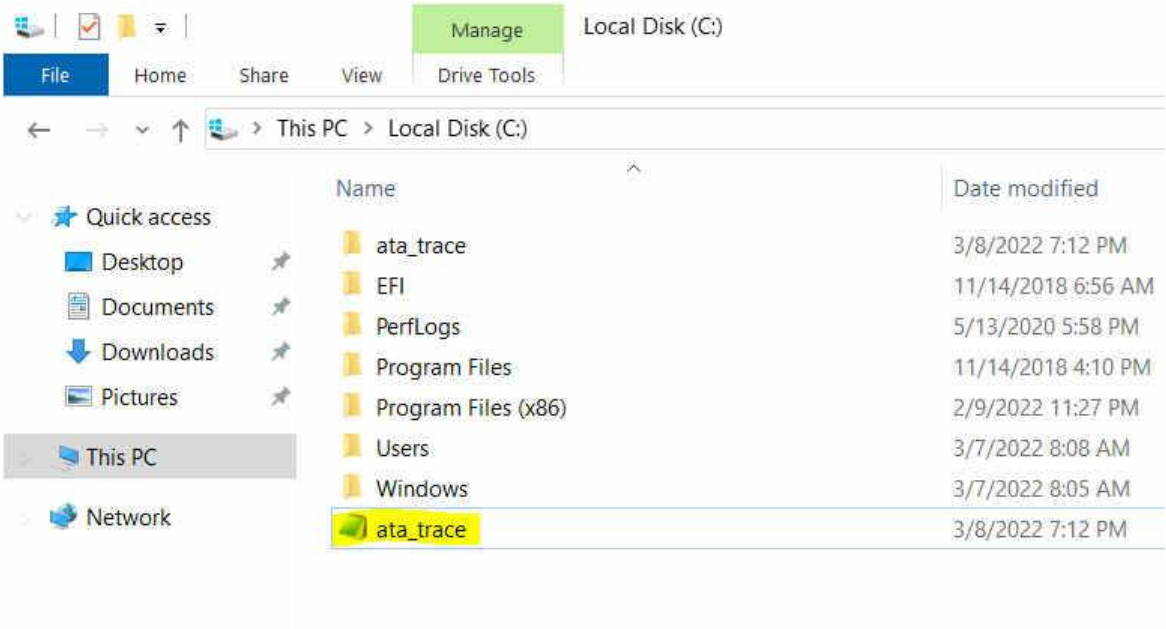
If you wish to stop the traces from getting captured, run the below command instead: `netsh trace stop`

```
PS C:\Users\Administrator> netsh trace start capture=yes tracefile=c:\ata_trace.etl persistent=yes maxsize=4096

Trace configuration:
---------------------------------------------------------------
Status:             Running
Trace File:         C:\ata_trace.etl
Append:             Off
Circular:           On
Max Size:           4096 MB
Report:             Off
```

Fetching the Traces of the Network to Store in a File

As you can see below, the file (**ata_trace**) has been created and stored in the **C:** drive.



Viewing the Trace Log File

Diagnosing the Windows Firewall and IPsec Events

Nowadays, monitoring alerts is crucial for any organization running applications or microservices. Perhaps you've decided to spin up some new Windows machine or Virtual host and check filtered data about the connectivity and firewall settings. If so, consider using the Windows Filtering Platform (WFP).

The Netsh commands for the WFP allow you to conduct checks and filtration on the firewall for systems that support Windows firewall and IPsec.

Run the following command to save the current operational state of the WFP and IPsec to an XML file (*wfpstate.xml*) on the working directory by default.

```
netsh wfp show state
```

With a copy of the state file (*wfpstate.xml*), you can further examine the WFP and IPsec state to find the root cause of a problem.

```
PS C:\Users\Administrator> netsh wfp show state
Data collection successful; output = wfpstate.xml
```

Saving Windows Firewall and IPsec Events State File

Perhaps you like to save the XML file to another location. If so, append the `file` flag and specify a save location, as shown below. Replace `location` with the actual save location and `statefile.xml` with your preferred file name.

```
netsh wfp show state file="location\statefile.xml"
```

```
PS C:\Users\Administrator> netsh wfp show state file = C:\Users\Administrator\wfpstate.xml
Data collection successful; output = C:\Users\Administrator\wfpstate.xml
```

Saving Windows Firewall and IPsec Events State File on Specified Location

Now, run the below command to save a list of network traffic events (`netevents`) to a file called *netevents.xml* in the working directory by default.

Like with the wfpstate.xml file, you can also specify a save location and file name for the network traffic events XML file (netevents.xml).

```
netsh wfp show netevents
```

```
PS C:\Users\Administrator> netsh wfp show netevents
Data collection successful; output = netevents.xml
```

Saving List of Network Traffic Events

If you need to diagnose a specific protocol only, use the `protocol` parameter below.

The below command saves a list of network traffic events only for the TCP/IP protocol in XML format, where 6 is TCP/IP's protocol number.

```
netsh wfp show netevents protocol=6
```

```
PS C:\Users\Administrator> netsh wfp show netevents protocol=6
Data collection successful; output = netevents.xml
```

Saving a list of Network Events for a Specific Protocol

Validating Incoming and Outgoing Traffic

If you're working with many applications where traffic comes in and leaves your network, you need to validate that traffic. How? Winsock is an interface that allows you to handle all the input/output requests for internet applications in a Windows system.

Execute the following `netsh winsock` command to show all the stored Windows sockets installed on your machine.

```
netsh winsock show catalog
```

Below, you can see all the Windows sockets entries and details of each one.

```
PS C:\Users\Administrator> netsh winsock show catalog

Winsock Catalog Provider Entry
------------------------------------------------------
Entry Type:                         Base Service Provider
Description:                        AF_UNIX
Provider ID:                        {A00943D9-9C2E-4633-9B59-0057A3160994}
Provider Path:                      %SystemRoot%\system32\mswsock.dll
Catalog Entry ID:                   1005
Version:                            2
Address Family:                     1
Max Address Length:                 110
Min Address Length:                 2
Socket Type:                        1
Protocol:                           0
Service Flags:                      0x20026
Protocol Chain Length:              1

Winsock Catalog Provider Entry
------------------------------------------------------
Entry Type:                         Base Service Provider
Description:                        MSAFD Tcpip [TCP/IP]
Provider ID:                        {E70F1AA0-AB8B-11CF-8CA3-00805F48A192}
Provider Path:                      %SystemRoot%\system32\mswsock.dll
Catalog Entry ID:                   1006
```

Viewing Windows Sockets Entries

The Winsock catalog tends to contain incorrect entries or becomes corrupt. In that case, run the `netsh` command below to perform a Winsock reset. This command sets the Winsock catalog and associated registry settings to their defaults

Related:[How and Why to Use the Netsh Winsock Reset Command (In 2021)]

```
netsh winsock reset
```

```
PS C:\Users\Administrator> netsh winsock reset

Sucessfully reset the Winsock Catalog.
You must restart the computer in order to complete the reset.

PS C:\Users\Administrator>
```

Resetting the Winsock Catalog to Default

Working with Alias and Unalias Using Netsh Command

Alias is an essential concept in networking. An alias instructs the shell to replace one string with another while executing the commands. Similarly, Netsh also uses Alias, a user-defined character string, which Netsh treats as a look-alike to another character string.

The syntaxes to add and delete an alias are below:

```
# Adds an alias
netsh add alias
# Deletes an alias
netsh delete alias
```

Reserving a Uniform Resource Locator (URL)

Reserving a URL properly in your system is crucial to defining the syntax of a URL endpoint to a web application. A reserved URL is defined for both the Web service and the web portal when configuring the applications on a server.

Three options are used along with the `netsh` command, `add urlacl`, `show urlacl`, and `delete urlacl`.

1. Run the below command to reserve a URL entry in the system where, `url` specifies the fully qualified URL, and `user` specifies the user or user-group name (`DOMAIN`).

```
netsh http add urlacl url=https://127.0.0.1:80/ user=\Everyone
```

```
PS C:\Users\Administrator> netsh http add urlacl url=https://127.0.0.1:80/ user=\Everyone

URL reservation successfully added
```

reserve a URL entry in the system

Next, run the below command to show the list of all access control lists for a particular reserve URL (`https://127.0.0.1:80/`)

```
netsh http show urlacl url=https://127.0.0.1:80/
```

```
PS C:\Users\Administrator> netsh http show urlacl url=https://127.0.0.1:80/

URL Reservations:
-----------------

    Reserved URL            : https://127.0.0.1:80/
        User: \Everyone
            Listen: Yes
            Delegate: No
            SDDL: D:(A;;GX;;;WD)
```

Listing Access Controls

3. Finally, execute the command below to delete reserved URLs.

```
netsh http delete urlacl url=https://127.0.0.1:80/
```

```
PS C:\Users\Administrator> netsh http delete urlacl url=https://127.0.0.1:80/

URL reservation successfully deleted
```

Deleting Reserved URLs

Managing Netsh Configuration and Query Commands

Nowadays, broadband and networking are crucial on laptops, computers, and mobiles. And knowing how to query and configure mobile broadband settings and parameters comes in handy.

But first, you must ensure the WWAN AutoConfig service is running.

1. Run the below command to `start` the `wwanSvc` service. `net start wwanSvc`

```
net start wwanSvc
```

```
PS C:\Users\Administrator\Desktop> net start wwanSvc
The WWAN AutoConfig service is starting.
The WWAN AutoConfig service was started successfully.

PS C:\Users\Administrator\Desktop>
```

Starting the WWAN AutoConfig Service

2. Next, run the `netsh add` command below to add a network `profile` (`Wi-Fi-Marriott Bonvoy.xml`) in the mobile broadband interface's profile data store configuration table.

Be sure to change `Wi-Fi-Marriott Bonvoy.xml` with the network profile (XML file) you like to add.

To check all the Mobile Broadband interfaces, run the below command. `netsh mbn show interface`

`netsh wlan add profile filename="c:\New folder\Wi-Fi-Marriott Bonvoy.xml"`

```
PS C:\Users\Administrator> netsh wlan add profile filename="c:\New folder\Wi-Fi-Marriott Bonvoy.xml"
Profile Marriott Bonvoy is added on interface Wi-Fi
```

Adding a New WLAN Profile in the System

Perhaps you like to export a network profile. If so, run the following command, where `SSID` is the network interface, and `destination` is the network profile's save path: `netsh wlan export profile "SSID" key=clear folder=destination`

3. Run the following command to `connect` to a particular mobile broadband network interface (`delhi_Sagar_5GHz`). Replace `Wi-Fi Network` with your Wi-Fi network name.

`netsh wlan connect name="Wi-Fi Network"`

```
PS C:\Users\Administrator> netsh wlan connect name="[redacted]"
Connection request was completed successfully.
```

Connecting to a Wi-Fi Network

4. Execute the below command to `delete` a network config profile (`Marriott Bonvoy`).

Deleting a network config profile from the profile data store is similar to running add command. The difference is that you'll replace `add` with the `delete` sub-context in the command.

`netsh wlan delete profile name="Marriott Bonvoy"`

```
PS C:\Users\Administrator> netsh wlan delete profile name="Marriott Bonvoy"
Profile "Marriott Bonvoy" is deleted from interface "Wi-Fi".
```

Deleting a WLAN profile in the system

5. Now, run the following `netsh dump` command to capture and display interface configuration scripts.

`netsh dump`

```
PS C:\Users\Administrator\Desktop> netsh dump
#========================
# Interface configuration
#========================
pushd interface


popd
# End of interface configuration



# ---------------------------------
# 6to4 Configuration
# ---------------------------------
pushd interface 6to4

reset



popd
# End of 6to4 configuration



# ---------------------------------
# IPHTTPS Configuration
```

Capturing Interface Configurations Details

If you wish to have a copy of the configuration details you can later review, run the following command. This command doesn't print anything on the console but instead saves the output to a text file (`netconfig.txt`).

`netsh dump > netconfig.txt`

6. Execute the `netsh set` command below to set the configuration of specified interfaces, such as the following:

- Setting the mobile broadband data auto-connect state for the given interface.
- Turning the mobile broadband data on or off for the specified profile set or interface.
- Setting the mobile broadband data highest connection category for the given interface.

`netsh set`

```
PS C:\Users\Administrator\Desktop> netsh set

The following commands are available:

Commands in this context:
set machine    - Sets the current machine on which to operate.
PS C:\Users\Administrator\Desktop>
```

Configuring Specified Interfaces

Managing HTTP System Settings

Another important `netsh` command to look at is `netsh http`, which allows you to query and configure HTTP.sys settings and parameters.

These HTTP.sys settings can be, but are not limited to the following:

- The cache of the HTTP service.
- All the cached URL resources of the HTTP service.
- All certificate bindings.
- A snapshot of all the HTTP services.

Windows HTTP Services provides developers with an HTTP client API to send requests through the HTTP protocol to other HTTP servers.

Execute the below `netsh` command to `show` you a list of all the commands you can run to view the `http` service-related settings and configurations.

```
netsh http show
```

Below, you'll see various options you can use with the `netsh http` command.

```
PS C:\Users\Administrator\Desktop> netsh http show

The following commands are available:

Commands in this context:
show cacheparam - Shows the cache parameters of HTTP service .
show cachestate - Lists cached URI resources and their associated properties.
show iplisten   - Displays all the IP addresses in the IP listen list.
show servicestate - Shows a snapshot of the HTTP service.
show setting    - Shows the setting values of the service.
show sslcert    - Displays SSL certificate bindings.
show timeout    - Shows the timeout values of the service.
show urlacl     - Displays URL namespace reservations.
PS C:\Users\Administrator\Desktop>
```

Viewing Available Commands for `netsh http` Command

Now, run the below command to verify the HTTP settings and configuration, cacheparam perhaps.

```
netsh http show cacheparam
```

```
PS C:\Users\Administrator\Desktop> netsh http show cacheparam

HTTP service cache parameters:
------------------------------

maxcacheresponsesize (per-uri cache limit): 262144 bytes
cacherangechunksize (range chunk): 65536 bytes

PS C:\Users\Administrator\Desktop>
```

Viewing HTTP Cache Parameters

Automating Netsh Commands Execution with Batch Scripts

So far, you learned everything you need to know about Netsh utility, so consider yourself a pro. But at times, you need to run dozens of commands in one go and store the results. So why not automate executing Netsh commands with Batch scripts?

Create a file named *myfile.bat* on your desktop and copy/paste the code below to the *myfile.bat* file. The code below shows all network interfaces and configurations of all network interfaces.

```
# Show all the interfaces of the Windows System
interface show interface
# Show the configurations of all the interfaces
interface ipv4 show config
# Show all the Wired lan network interfaces
lan show interfaces
# Updating the firewall rule to add ping protocol
advfirewall firewall add rule name="All ICMP V4" dir=in action=allow protocol=icmpv4
# Show all the events in the network
wfp show events
# Show all the network events
wfp show netevents
```

Next, run the below `netsh` command to execute the `myfile.bat` script from the specified folder path (`-f C:\Users\Administrator\Desktop\`).

```
netsh -f C:\Users\Administrator\Desktop\myfile.bat
```

As you can see below, the `netsh` commands inside the batch file (*myfile.bat*) were executed successfully.

```
PS C:\Users> netsh -f C:\Users\Administrator\Desktop\myfile.bat

Admin State    State          Type             Interface Name
-------------------------------------------------------------------------
Enabled        Connected      Dedicated        Ethernet

Configuration for interface "Ethernet"
    DHCP enabled:                         Yes
    IP Address:                           172.31.40.134
    Subnet Prefix:                        172.31.32.0/20 (mask 255.255.240.0)
    Default Gateway:                      172.31.32.1
    Gateway Metric:                       0
    InterfaceMetric:                      25
    DNS servers configured through DHCP:  172.31.0.2
    Register with which suffix:           Primary only
    WINS servers configured through DHCP: None

Configuration for interface "Loopback Pseudo-Interface 1"
    DHCP enabled:                         No
    IP Address:                           127.0.0.1
    Subnet Prefix:                        127.0.0.0/8 (mask 255.0.0.0)
    InterfaceMetric:                      75
    Statically Configured DNS Servers:    None
    Register with which suffix:           Primary only
    Statically Configured WINS Servers:   None

There is 1 interface on the system:

    Name                  : Ethernet
    Description           : AWS PV Network Device #0
```

Executing Netsh Commands with a Batch Script

Conclusion

Throughout this tutorial, you've learned different `netsh` commands to control your network configuration and diagnose potential issues. Netsh utility is a free tool with many features you can use whether you're just checking out your network configuration or monitoring your network closely.

With this newfound knowledge, you can now monitor interfaces, run trace commands and scripts, and even convert all IP address settings.

Now that you're a Netsh guru, how do you plan to use it? Why not automate command executions by compiling commands into a PowerShell script you can run at will?