Intel® Ethernet 700/800 Series

Windows Performance Tuning Guide

Rev. 1.1

February 2024

Doc. No.: 784543, Rev.: 1.1

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

This document (and any related software) is Intel copyrighted material, and your use is governed by the express license under which it is provided to you. Unless the license provides otherwise, you may not use, modify, copy, publish, distribute, disclose or transmit this document (and related materials) without Intel's prior written permission. This document (and related materials) is provided as is, with no express or implied warranties, other than those that are expressly stated in the license.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors which may cause deviations from published specifications.

Intel and the Intel logo are trademarks of Intel Corporation in the U.S. and/or other countries.

Other names and brands may be claimed as the property of others.

Copyright © 2023-2024, Intel Corporation. All rights reserved.

Contents

Revision History

Revision	Date	Comments
1.1	February 1, 2024	Updated the following sections: • Terminology • Reference Documents • Known Issues • Install Adapter and Cable • Check BIOS Settings • Method 1 – INF Driver Installation • Update the 700/800 Series Firmware • Check System Hardware Capabilities • Method 1 – Using Intel® Ethernet Cmdlets (recommended) • Ensure DDP Package is Loading Properly • SSU Output • Verify DDP Package Status • Low Receive (RX) Performance and/or Discarded Packets • Low Performance with SR-IOV/IOMMU Passthrough/VMs/ Hyper-V • Pinning Application Threads to Specific Cores Using the Process Lasso Utility • RSS Base Processor Number • Optimization for Specific Usage Models • General Tuning • BIOS Settings • SR-IOV/VMs • Adapter Teaming Added the following new sections: • Microsoft NTTTCP • Collect Event Trace Logs (ETL) • LLDP • PCI Express (PCIe) Passthrough
1.0	August 10, 2023	Initial Public Release.

1.0 Introduction

This guide provides tuning guidance for optimal networking performance using Intel® Ethernet 700 Series (700 Series) or Intel® Ethernet 800 Series (800 Series) devices in Windows environments. This guide focuses on hardware, driver, and operating system conditions, as well as settings that help improve network performance.

NOTE

Networking performance can be affected by any number of outside influences. Only the most common and dramatic of these are covered in this guide.

1.1 Terminology

Term	Description
ANS	Advanced Networking Services
DCB	Data Center Bridging
DDP	Dynamic Device Personalization
EPCT	Ethernet Port Configuration Tool
HPCC	High Performance Computing Cluster
IOMMU	Input-Output Memory Management Unit
iSCSI	Internet Small Computer Systems Interface
ITR	Interrupt Throttle Rate
LACP	Link Aggregation Control Protocol
MTU	Maximum Transmission Unit
NUMA	Non-Uniform Memory Access
NVM	Non-Volatile Memory
PCI	Peripheral Component Interconnect
PCIe	PCI Express
RSS	Receive Side Scaling
SET	Switch Embedded Teaming
SR-IOV	Single Root I/O Virtualization
SSU	System Support Utility
VF	Virtual Function
VM	Virtual Machine

continued...

Term	Description
VMQ	Virtual Machine Queue
vNIC	Virtual Network Interface Card
vSwitch	Virtual Switch

1.2 Reference Documents

- User Guide for all Intel® Ethernet adapters and devices, supporting Windows and Linux:
  - *Intel® Ethernet Adapters and Devices User Guide*
- Technical Datasheets:
  - *Intel® Ethernet Controller X710/XXV710/XL710 Datasheet*
  - *Intel® Ethernet Controller E810 Datasheet*
- Complete software bundle for all Intel® Ethernet products

  (downloads all drivers, NVMs, tools, and Release Notes):
  - Intel® Ethernet Adapter Complete Driver Pack
  - Intel® Ethernet Controller Products Release Notes (select Version for your SW Release)
- Non-Volatile Memory (NVM) Update Packages:
  - Intel® Non-Volatile Memory (NVM) Update Utility for Ethernet Adapters 700 Series-Windows
  - Intel® Non-Volatile Memory (NVM) Update Utility for Ethernet Network Adapters E810 Series-Windows

NOTE

If you have an OEM-branded Intel adapter (such as Dell, Lenovo, Cisco, HPE, or other), please contact the vendor for guidance on how to update NVM image(s). These images are not included in the base Intel NVM update package.

- Ethernet Port Configuration Tool to configure E810 devices in different port configurations

  (1x100 Gbps, 4x25 Gbps, 8x10 Gbps):
  - Ethernet Port Configuration Tool - Windows (EPCT)

NOTE

This is specific to E810 devices only.

- System Support Utility (SSU) to gather system details on Windows OS

  (hardware details, firmware/driver versions, and settings):
  - Intel® System Support Utility for Windows (SSU)

- Dynamic Device Personalization (DDP) Technology Guides:
  - *Intel® Ethernet 700 Series*
  - *Intel® Ethernet Controller E810 Dynamic Device Personalization (DDP) Technology Guide*

1.3 Known Issues

Known issues for Windows OS and all Intel® Ethernet products can be found in the following locations:

1. Known Issues section of the Release Notes, packaged with the Intel® Ethernet Adapter Complete Driver Pack.

   The file is located in the root folder of the software download package.
2. Known Issues section of the *Intel® Ethernet Adapters and Devices User Guide*, Section 9.0.
3. Additional Known Issues, specific to the Windows OS and Intel® 700/800 Series devices:
   - Intel® Advanced Networking Services (ANS) is supported on Microsoft Windows 2012 R2 and earlier, to configure Adapter Teaming.

     Intel® ANS is not supported on Microsoft Windows Server 2016 and later.

     Mitigation: See Adapter Teaming for the steps to configure Adapter Teaming on Windows Server 2016 and later.
   - Creating Switch Independent Teaming in a Microsoft Service (GUI) does not support SR-IOV. However, it does spawn a Virtual Machine Queue (VMQ) Virtual Network Interface Card (vNIC) on the VM. This can result in lower VM performance than a configuration that does support SR-IOV to spawn a VF vNIC on the VM.

     NOTE

     Switch Independent teaming is only supported on Windows VM and is not supported on Linux VM.
   - GUI Teaming and SET vSwitch:
     - GUI Teaming is available for both Windows VM and Linux VM. Windows VM supports VMQ and allows the creation of VMQ vNICs within the VM. However, VMQ is not supported in Linux VM. Additionally, GUI teaming does not support SR-IOV, which is a critical feature for Linux VM.
     - SET vSwitch is available in Windows Server 2016 and later, and offers advanced teaming capabilities for both Windows VM and Linux VM. It supports SR-IOV, allowing the creation of virtual functions (VF) for the VM vNIC, which is compatible with Linux VM as well. However, it is worth noting that the VMQ is a feature specific to Windows VM and might not be relevant for Linux VM.

- Driver versions *i40ea* 1.16.130 and later provide low throughput numbers in Windows 2022 Hyper-V VM when vSwitch is set to a Switch Independent Teaming Interface without SR-IOV enabled.

  Mitigation: Use *i40ea* version 1.18.367.0 (or later).
- If Hyper-Threading is enabled in BIOS, the driver does not utilize the hyper-threaded cores for RSS, and this can impact performance for applications using both physical and logical (HT) cores.
- *icea* drivers provide low throughput numbers in Windows 11 due to the missing registries: RssBaseProcNumber, NumaNodeId, and MaxRssProcessors, in the INF file.

2.0 Initial Checklist

2.1 Install Adapter and Cable

1. Install the adapter in an appropriate slot. See Check PCI Express (PCIe) Slot Capabilities to verify PCIe compatibility with the adapter.
2. Use the proper cabling for your device. See *Intel® Ethernet Adapters and Devices User Guide*, Section 2.3, "Connect Network Cables" for a list of supported cables and their part numbers.

NOTE

If the adapter is not detected in the system, verify the installed OS is supported by the driver. Consult the Feature Support Matrix (FSM) for OS support by product and driver. See Intel® Ethernet Controller Products Release Notes, Section 6.1, "Feature Support Matrix" for supported OSs (select the correct Software Release Version in the drop-down menu for your system).

2.2 Check BIOS Settings

1. Performance Profile is recommended for most applications and workloads, for best overall performance and throughput. Performance Profile disables C-states and sets power settings to Performance mode. If your BIOS does not have a Performance Profile, experiment with manually disabling C-States, and setting Power Management for optimal performance.
2. Hyper-Threading (if Intel platform) or Simultaneous Multithreading (if AMD platform) can be enabled or disabled based on the application.

   NOTE

   Intel recommends that you enable Hyper-Threading for the best throughput and that you disable it for the best latency.

3. Intel® VT-d (if Intel platform) or Intel® AMD-V (if AMD platform) should be enabled for environments using SR-IOV or virtualization with virtual functions (VF).

NOTE

See BIOS Settings for more details on these settings.

2.3 Update Driver/Firmware Versions

2.3.1 Update the Driver (i40ea/i40eb/icea)

There are different installation options for updating Intel® drivers and software on Windows. See the *Intel® Ethernet Adapters and Devices User Guide* for complete installation instructions. This section contains details on what is needed to install Intel drivers for optimized performance on Windows.

2.3.1.1 Method 1 – INF Driver Installation

NOTE

This method must be applied to each Ethernet port/interface on the server.

Method 1 is recommended for single-port testing, and for quick and easy driver installations outside of the Intel® PROSet GUI.

Any method used to install or update the driver causes enumeration of the driver stack at the Windows OS level, which affects all Ethernet devices on the server.

NOTE

You must ensure that traffic is not running on any of the other 700/800 Series devices in the system before updating the driver, even with INF installations on a single-port.

INF driver installations are used most often within Intel engineering teams and validation teams to update drivers. If performance issues are observed using Intel® PROSet driver installations, try an INF driver installation to see if it makes a difference in performance.

Method 1 requires separate steps to update driver settings, post-driver installation, through **Task Manager** or **PowerShell** commands.

To install or update a driver:

1. Download this driver pack for the latest drivers and software.

   Intel® Ethernet Adapter Complete Driver Pack

   NOTE

   This driver pack includes the drivers and firmware for all Intel® Ethernet products for all OS's, and may take several minutes to download.

2. Extract the*Release_<version>.zip* file and locate the driver.
   - The *icea* driver is located under *Release_<version>\PROCGB\Winx64*.
   - The *i40ea/i40eb* drivers are located under *Release_<version>\PRO40GB \Winx64*.

3. Install the driver:
   a. Open Device Manager and locate the device to update.
   b. Right-click the device and select the option, **Update Driver**.
   c. Select **Browse my computer for driver**.
   d. Point the installer to:

      Release_<version>\PROCGB\Winx64

      or

      Release_<version>\PRO40GB\Winx64
   e. The installer installs the correct driver for the OS.
   f. Use the **Task Manager** or **PowerShell** commands to update the driver settings. See Performance Tuning with Intel Driver Settings (i40ea/i40eb/icea).

NOTE

Remember that INF driver installation must be applied to each 700/800 Series interface in the server. Any changes to the advanced driver settings must be applied post-driver installation directly to each interface.

2.3.1.2 Method 2 – Intel® PROSet Driver Installation

NOTE

Method 2 must be applied system-wide across all interfaces.

Method 2 installs the driver system-wide and applies the update to all 700/800 Series devices on the server. Intel® PROSet offers the ability to adjust driver settings and thresholds within the GUI.

- Older Windows versions (Microsoft Windows 2012 and earlier) support ANS Teaming within Intel® PROSet.
- Newer Windows versions (Microsoft Windows Server 2016 and later) have removed ANS capability from Intel® PROSet. See Known Issues for details.

For steps on how to install drivers with Intel® PROSet, refer to the *Intel® Ethernet Adapters and Devices User Guide*, Section 3.4, "Installing Intel® PROSet", and Section 5.1, "Installing Windows Drivers and Software".

When using Adapter Teaming, refer to the *Intel® Ethernet Adapters and Devices User Guide*, Section 4.1, "Adapter Teaming".

For tips on tuning, see Adapter Teaming in this document.

2.3.2 Update the 700/800 Series Firmware

NOTE

If you have an OEM-branded Intel adapter (such as Dell, Lenovo, Cisco, HPE, or other), please contact the vendor for guidance on how to update NVM image(s). These images are not included in the Intel NVM update package.

To update the 700/800 Series Firmware:

1. Download the latest NVM Update Utility Packages for Windows:
   - Intel® Non-Volatile Memory (NVM) Update Utility for Ethernet Adapters 700 Series—Windows
   - Intel® Non-Volatile Memory (NVM) Update Utility for Ethernet Network Adapters E810 Series—Windows

   NOTE

   The NVM Update Utility is also included with the Intel® Ethernet Adapter Complete Driver Pack. The Windows zip files are located in:
   - *Release_<version>NVMUpdatePackage\700_Series*
   - *Release_<version>NVMUpdatePackage\E810*

2. Extract the NVMUpdatePackage.zip file.

   The NVMUpdatePackage.zip file contains the self-extracting executable files:
   - *700Series_NVMUpdatePackage_<version>_Windows.exe*
   - *E810_NVMUpdatePackage_<version>_Windows.exe*

3. Install the NVM image.

   The executable files from the NVMUpdatePackage can be run directly on the server with the Intel® adapter. They can also be further extracted to reveal the individual NVM images (.bin files) and the nvmupdatew64e.exe binary tool.

   The **nvmupdate** tool can be run:
   - Using Windows File Explorer (right-click **Run as administrator**).

     or
   - Using the **PowerShell** commands with advanced features to inventory the current version of the device and the log output.

   Either method brings up a menu to select the 700/800 Series device to be updated.

   Enter the numerical menu entry (number) to start the update, for example 01.

4. After the installation completes, verify no errors were reported.
5. Reboot the server to complete the update (required).

2.4 Read the Driver Release Notes

Check the Release Notes and the README text files for known issues, and get the latest supported Windows OS details. These files are included in the Documentation section of the Intel® Ethernet Adapter Complete Driver Pack for the software release.

2.5 Check Windows OS Version

The performance tuning guidance provided in this document assumes the OS environment is running a modern version of the Windows OS. The full list of supported OSs and drivers can be found in the Release Notes provided with the Intel® Ethernet Adapter Complete Driver Pack.

2.6 Check System Hardware Capabilities

100 Gbps and 4x25 Gbps Ethernet network speeds have minimum CPU requirements and minimum system requirements to reach high-performance throughput. In general, a modern server class processor with optimal memory configuration for your platform should be sufficient. However, needs can vary depending on your workload.

All memory channels should be populated to achieve full memory bandwidth. The platform BIOS and related OS settings should be configured for high-performance mode.

Verify that your CPU and memory configurations are capable of supporting the level of bandwidth required for the intended workload. It is important to understand the architecture of your platform to ensure workloads are properly scheduled on cores and nodes local to the networking hardware.

See the following product datasheets for requirements:

- *Intel® Ethernet Controller X710/XXV710/XL710 Datasheet*
- *Intel® Ethernet Controller E810 Datasheet*

NOTE

Intel has dual port X710/XXV710/XL710 and E810 network adapters that have one underlying controller per adapter that can support up to the full line rate of the adapter across both ports (total). For example, the E810 is a 100 GbE Controller. The E810-CQDA2 dual port 100 GbE adapter, using this controller, is not intended to support 2x100 GbE, but rather a single 100 GbE interface with an active back-up port. When attempting to use line-rate traffic involving both ports, the internal switch is saturated, and the combined bandwidth between the two ports is limited to a total of 100 Gbps. XL710-QDA2 adapters can support up to 40 Gbps across both ports (total). XXV710 adapters can support up to 25 Gbps across both ports (total).

2.7 Check PCI Express (PCIe) Slot Capabilities

Verify the PCIe capabilities using **PowerShell** and Intel® Ethernet Cmdlets.

The full PCI capabilities for Intel® Ethernet devices are listed in the following product technical datasheets:

- *Intel® Ethernet Controller E810 Datasheet*
- *Intel® Ethernet Controller X710/XXV710/XL710 Datasheet*

NOTE

800 Series devices support up to PCIe 5.0 x16 connections and requires a minimum of PCIe 3.0 x16 speeds (x8 for dual-port 25G).

2.7.1 Method 1 – Using Intel® Ethernet Cmdlets (recommended)

NOTE

The Windows PowerShell Cmdlets are only supported in the Windows x64 architecture and not in the Windows x86 architecture.

To run the Intel® Ethernet Cmdlets, you must have Administrator privileges.

1. Get a list of Intel® Ethernet devices:

   ```
   Get-IntelEthernet
   ```

   NOTE

   This command displays one entry for each physical or logical device. For example, a 1x100 Gbps 800 Series Adapter configured in 4x25 Gbps mode shows as four separate entries.

2. List the device properties for a specific adapter interface:

   ```
   Get-IntelEthernet -Name <Adapter Name> | fl
   ```

3. Verify the output, specifically these settings:
   - BusType (PCI Express)
   - LinkStatus
   - NegotiatedLinkSpeed
   - NegotiatedLinkWidth

2.7.2 Method 2 – Using Non Intel® Ethernet Cmdlets

1. Get a list of network adapters:

```
Get-NetAdapter
```

NOTE

This command displays one entry for each physical or logical device. For example, a 1x100 Gbps 800 series adapter configured in 4x25 Gbps mode shows as four separate entries. This command also lists all Ethernet devices in the server, not just Intel® devices.

2. List the device details for a specific adapter:

```
Get-NetAdapterHardwareInfo -Name <Adapter Name> | fl
```

3. List the PCIe details for all network adapters:

```
Get-NetAdapterHardwareInfo
```

4. Verify the output, specifically these settings:
   - DeviceType (PCI Express endpoint)
   - PciExpressCurrentLinkSpeed
   - PciExpressCurrentLinkWidth
   - PciExpressMaxLinkSpeed
   - PciExpressMaxLinkWidth

NOTE

Some physical PCIe x16 slots are configured electrically as x8 slots. These slots have insufficient bandwidth for full performance with 100G or quad port 25G devices. The software device driver detects this situation and writes a message to the system log:

- Gb/s available PCIe bandwidth, limited by 8 GT/s x8 link at 0000:xx:xx.0 (capable of 252.048 Gb/s with 16 GT/s x16 link)

NOTE

If this error occurs, move your adapter to a true PCIe v3.0 x16 slot or greater, to avoid performance issues.

2.8 Ensure DDP Package is Loading Properly

Ensure that the driver reports the presence, and successful load, of the Dynamic Device Personalization (DDP) package file in Event Viewer by displaying this message:

- The DDP package was successfully loaded: ICE OS Default Package version X.X.X.X

In the event of a DDP package load error, the device defaults to safe mode, and many performance features are unavailable. If there are errors related to loading the DDP package, this can affect performance.

As a workaround to resolve DDP load issues, try to unload and reload the driver (*i40ea*/*i40eb*/*icea*).

For further troubleshooting steps, refer to the DDP Technology Guides:

- *Dynamic Device Personalization for Intel® Ethernet 700 Series*
- *Intel® Ethernet Controller E810 Dynamic Device Personalization (DDP) Technology Guide*

3.0 Baseline Performance Measurements and Tuning Methodology

3.1 General Performance Tuning Methodology

Focus on one tuning change at a time, so you know what impact each change makes to your test. The more methodical you are in the tuning process, the easier it is to identify and address the causes of performance bottlenecks.

3.2 Network Performance Benchmarks

Before beginning any tuning exercise, you should have a good baseline measurement of your network performance. Also, in addition to the initial assessment of your specific application/workload performance, you should use standard network performance micro-benchmarks to verify that the network device is in a good state.

To benchmark interface speeds of 25 Gbps and higher, it is common to require that multiple traffic threads utilize the connection. iperf2 and iperf3 provide examples of common micro-benchmarks that are known to work well.

3.2.1 iperf2

Intel recommends using iperf2 over iperf3 for most benchmarking situations due to their ease of use and support of multiple threads in a single application instance. Intel recommends running with the **-P** option with 2 to 4 threads for 25G connections and around 6 to 8 threads for 100G connections.

Download the latest iperf2 Windows binary: Iperf 2 Files

To run uni-directional traffic from client to server:

1. Start Server.

```
iperf-2.1.9-win.exe -s -D
```

2. Start Client:

```
iperf-2.1.9-win.exe -c <serverIP> -P <threads>
```

To run bi-directional traffic from client to server and vice versa:

1. Start Server:

```
iperf-2.1.9-win.exe -s -D -p <port>
```

2. Start Client:

```
iperf-2.1.9-win.exe -c <serverIP> -p <port> -P <threads> --full-duplex
```

or

```
iperf-2.1.9-win.exe -c <serverIP> -p <port> -P <threads> -d
```

NOTE

Both the `--full-duplex` and `-d` options in iperf2 allow you to perform bidirectional testing. However, the `--full-duplex` option specifically focuses on full duplex testing.

3.2.2 iperf3

Using iperf3 requires multiple instances of an application to take advantage of Receive Side Scaling (RSS), multi-threads, and hardware queues.

Using iperf3 differs from using iperf2 as follows:

- iperf3 with the *-P* option set uses multiple streams of traffic inside a single I/O thread.
- iperf3 supports single-threaded benchmarking with multiple streams from one application thread.
- iperf2 with the *-P* option set spawns a separate thread for each instance of *-P*.
- iperf2 supports true multi-threaded benchmarking.

Intel recommends running with 2 to 4 application sessions for 25G connections and around 6 to 8 sessions for 100G connections. Each session should specify unique TCP port values using the **-p** option.

- Download the latest iperf3 Windows binary:
- https://iperf.fr/iperf-download.php#windows

To run uni-directional traffic from client to server:

1. Start Server.

```
iperf3.exe -s
```

2. Start Client.

```
iperf3.exe -c <serverIP>
```

To start multiple instances (threads) of iperf3:

1. Use a for-loop to map threads to TCP ports.
2. Run iperf3 in the background using this cmdlet:

```
Start-Process
```

3. Start Server with 4 instances/threads:

```
for ($i = 0; $i -le 3; $i++) {
   Start-Process powershell.exe -ArgumentList "-Command", ".\iperf3.exe -s -p
((5200+$i))"
}
```

4. Start Client with 4 instances/threads:

```
for ($i = 0; $i -le 3; $i++) {
   Start-Process powershell.exe -ArgumentList "-Command", ".\iperf3.exe -s -p
((5200+$i))"
}
```

NOTE

The *for loops* create four iperf3 server/client processes in the background on ports 5200 to 5203. A new window is created for each process.

To turn off the creation of new windows and send all output to the active window:

- Add the `–NoNewWindow` command to the end of the Start-Process commands as follows:

```
Start-Process powershell.exe -ArgumentList "-Command", ".\iperf3.exe -s -p
((5200+$i))" -NoNewWindow
```

```
Start-Process powershell.exe -ArgumentList "-Command", ".\iperf3.exe -c
<serverIP> -p ((5200+$i))" -NoNewWindow
```

NOTE

For 100G connections, increase the for-loop to create 6 to 8 instances/threads.

3.2.3 Microsoft NTTTCP

NTTTCP is an extended version of the NTTCP (Network Throughput Testing Tool) developed by Microsoft. This tool is used to measure network throughput between two Windows systems and allows users to evaluate network performance metrics such as bandwidth, latency, and packet loss.

Download the tool from the official Microsoft GitHub repository https://github.com/microsoft/ntttcp or other Microsoft download channels.

1. Start server:

   ```
   ntttcp.exe -s -m <no-of-connections>,<mapping-type>,<local-ip> -t <duration> -
   p <port-number>
   ```

   Example command:

   ```
   ntttcp.exe -s -m 8,1,192.168.11.22 -t 30 -p 7001
   ```

2. Start client:

   ```
   ntttcp.exe -r -m <no-of-connections>,<mapping-type>,<server-ip> -t <duration>
   -p <port-number>
   ```

   Example command:

   ```
   ntttcp.exe -r -m 8,*,192.168.11.22 -t 30 -p 7001
   ```

   Where * indicates that the tool should automatically detect the local IP address for the other endpoint.

4.0 Performance Troubleshooting

4.1 Collect Server Details

4.1.1 SSU Output

Collect system level details for each server in the test configuration.

1. Download the latest version of SSU:

   Intel® System Support Utility for Windows (SSU)

2. Run SSU.exe on each server involved in the testing.

The SSU output provides a full system inventory of the hardware, firmware, and driver levels. These details are extremely useful to help diagnose and debug performance issues. Any performance-related support tickets opened with Intel support teams require this information.

NOTE

SSU does not currently collect PCIe details. However, this information can be critical for diagnosing performance issues.

NOTE

PCIe network adapters must be installed in a PCIe slot that can support the device's capabilities. If an adapter is installed in a PCIe slot with lower PCIe width/speed capabilities than required by the device, performance will be largely impacted. For example, when an E810 adapter, which requires a x16 PCIe slot and a x16 PCIe riser, is installed or electrically linked at x8, performance is impacted. Follow the steps in Collect Event Trace Logs (ETL) to collect PCIe details separately.

Follow the steps in Check PCI Express (PCIe) Slot Capabilities to collect PCIe details separately. X710 devices support up to PCIe 5.0 x8 connections and require a minimum of PCIe 3.0 x8 speeds. E810 devices support up to PCIe 5.0 x16 connections and require a minimum of PCIe 3.0 x16 speeds (x8 for dual port 25G).

4.2 Collect Event Trace Logs (ETL)

ETL (Event Trace Logs) are log files generated on Windows OS using the PerfView tool provided by Microsoft. These log files include application and system-level errors, warnings, and other event data useful for debugging performance and timing issues.

To collect ETL logs:

Collect an ETL trace with PerfView

4.3 Check System Logs for Driver Errors

To check the system logs for errors and warnings in Windows, use one of the following methods:

- Event Viewer
- **PowerShell** commands

To check the system logs using Event Viewer:

1. Open Event Viewer using the search function of the Windows start menu.
2. Open the application.
3. Navigate to Windows Logs.
4. Navigate to System.
5. Sort on the first column, titled Level, to list the Error and Warning messages at the top of the system event log window.
6. Look for any Error or Warning entries for the driver (*i40ea*/*i40eb*/*icea*).
7. Look for other Errors/Warnings that may be relevant to your application or configuration, such as:
   - Hyper-V entries if you are using a virtualized environment
   - DNS Client Events for multi-host network topologies

To check the system logs using **PowerShell** commands:

1. List all events:

```
Get-WinEvent -LogName System
```

2. List only warnings (ErrorLevel 2) and errors (ErrorLevel 3):

```
Get-WinEvent -LogName System -ErrorLevel 2,3
```

3. Retrieve the latest 50 events from the Application, System, and Security logs:

```
Get-WinEvent -LogName "Application", "System", "Security" -MaxEvents 50 |
Format-Table -AutoSize
```

4.4 Verify DDP Package Status

Ensure that the driver reports the presence and successful load of the Dynamic Device Personalization (DDP) package file. In the event of a DDP package load error, the device defaults to safe mode and many performance features are unavailable.

To find the DDP package status:

- In Event Viewer, go to:

  ```
  Event Viewer > Windows Logs > System
  ```

  The status is shown with the driver (*i40ea*/*i40eb*/*icea*) as the source log.
- To filter the DDP status from *icea* driver logs:

  Use the **PowerShell** command:

  ```
  Get-WinEvent -provider icea | ? Message -like *DDP*
  ```

The following example shows a Windows system event log with the DDP Package successfully loaded.

```
Intel(R) Ethernet Network Adapter E810-C-Q2 #2
The DDP package was successfully loaded: ICE OS Default Package version 1.3.30.0
```

If there are errors related to loading the DDP package, issues this can affect performance. As a workaround to resolve DDP load issues, try to unload and reload the driver (*i40ea*/*i40eb*/*icea*).

For further troubleshooting steps, refer to the DDP Technology Guides:

- Dynamic Device Personalization for Intel® Ethernet 700 Series
- Intel® Ethernet Controller E810 Dynamic Device Personalization (DDP) Technology Guide

4.5 Check CPU Utilization

You should check CPU utilization while I/O is active. CPU utilization per core is more relevant to performance than overall CPU utilization, since it provides an idea of the CPU utilization per network queue.

If you have only a few threads running network traffic, you might only have a few cores being used. However, if those cores are at 100%, your network throughput might be limited by CPU utilization.

In this case, it is time to:

- Tune interrupt the moderation rate as detailed in Interrupt Moderation Rate.
- Increase the number of application threads to spread out the CPU load over more cores. If all cores are running at 100%, your application might be CPU bound rather than network bound.

To check CPU utilization (per Core) using **Task Manager**:

1. Open Windows **Task Manager** and select the Performance tab.
2. View the CPU statistics window to observe which cores are being used while traffic is running.
3. View the percent utilization for each core.

NOTE

For servers with more than one CPU socket/NUMA node, best performance is to use CPU cores closest to the adapter on the same Non-Uniform Memory Access (NUMA) node.

4.6 Check System Resource Utilization

To check the system resources utilization:

1. Use the built-in **Performance Monitor** or **PowerShell**.
2. List the available system resources with performance counters:

```
Get-Counter -ListSet * | Select-Object -ExpandProperty CounterSetName
```

3. Get the counters for a specific counter set:

```
Get-Counter -Counter "\{CounterSetName}\*"
```

4. Replace{CounterSetName} with the name of the counter set you want to monitor.
5. List the IPV4 stats for all, TCP, and UDP:

```
Get-Counter "\IPv4\*"
Get-Counter "\TCPv4\*"
Get-Counter "\UDPv4\*"
```

6. List the network interface stats:

```
Get-Counter "\Network Interface(*)\Current Bandwidth"
Get-Counter "\Network Interface(*)\Bytes Total/sec"
Get-Counter "\Network Interface(*)\Bytes Received/sec"
Get-Counter "\Network Interface(*)\Bytes Sent/sec"
Get-Counter "\Network Interface(*)\Packets/sec"
Get-Counter "\Network Interface(*)\Packets Received/sec"
Get-Counter "\Network Interface(*)\Packets Sent/sec"
Get-Counter "\Network Interface(*)\Packets Received Unicast/sec"
Get-Counter "\Network Interface(*)\Packets Received Non-Unicast/sec"
Get-Counter "\Network Interface(*)\Packets Received Discarded"
Get-Counter "\Network Interface(*)\Packets Received Errors"
Get-Counter "\Network Interface(*)\Packets Received Unknown"
Get-Counter "\Network Interface(*)\Packets Sent Unicast/sec"
Get-Counter "\Network Interface(*)\Packets Sent Non-Unicast/sec"
Get-Counter "\Network Interface(*)\Packets Outbound Discarded"
Get-Counter "\Network Interface(*)\Packets Outbound Errors"
Get-Counter "\Network Interface(*)\Output Queue Length"
Get-Counter "\Network Interface(*)\Offloaded Connections"
```

NOTE

If dropped or discarded packets are suspected, check the following counters:

- IPv4 Datagrams Received Discarded Counter
- Network Interface Packets Received Discarded Counter

For tips to help resolve these issues, see Check Network Connections (netstat).

4.7 Check IP Statistics (ipstats)

Analyze the IP statistics for all IPv4 traffic system-wide.

- Show IPv4 stats:

```
netsh interface ipv4 show ipstats
```

- This output shows the details for all IP Forwarding (enabled/disabled) and for all counters for RX/TX packets (received/sent/discards).

NOTE

These metrics are system level, not for each individual adapter or network device.

4.8 Analyze Performance Counters (typeperf)

Analyze the performance counters using the built-in Performance Monitor tool, or using the **typeperf** command line tool, available with default Windows OS installations.

For example:

```
typeperf.exe counter <counter_path> -si <sampling_interval> -sc <sample_count>
```

Where:

```
<counter_path>
```

Specify the performance counter you want to monitor.

You can provide the full counter path, such as:

```
\Processor(_Total)\% Processor Time
```

or use wildcard characters (*) for multiple counters.

```
-si <sampling_interval>
```

Specify the sampling interval in seconds. This determines the frequency at which data is collected. The default is one second.

```
-sc <sample_count>
```

Specify the number of samples to collect. This determines the duration of the monitoring session. For example, if you set it to 10, it collects data for 10 seconds.

4.9 Check Network Connections (netstat)

You can install and run **netstat** in Windows Server 2012 and later. The **netstat** command can be used to display active TCP/UDP connections, listening ports, Ethernet statistics, and IP routing tables.

You can find more details about **netstat** in this article: netstat

The following example **PowerShell** commands can be used to help diagnose network issues:

- List active TCP connections and all TCP and UDP listening ports:

  ```
  netstat.exe -a
  ```

- List active TCP connections and TCP (only) listening ports, and include the application PID:

  ```
  netstat.exe -a -p tcp -o
  ```

- Display the IP routing table:

  ```
  netstat.exe -r
  ```

- Display Ethernet statistics for bytes sent/received, packets, discards, and errors:

  ```
  netstat.exe -e
  ```

NOTE

Ethernet counters are useful to monitor byte and packet counters across test runs, and to check for dropped packets (discards) or any errors on the interface. See Low Receive (RX) Performance and/or Discarded Packets for tips on resolving discarded RX packets.

4.10 Low Receive (RX) Performance and/or Discarded Packets

Discarded or dropped RX packets, can indicate an issue with system or resource contention. For example, if the CPU is not servicing buffers fast enough or resources are limited in the PCIe slot/memory/system, this can cause discarded packets.

If discarded packets are observed in the **Task Manager** or **PowerShell** commands, such as the **Get-Counter** cmdlet examples in Analyze Performance Counters (typeperf), or the **netstat -e** and **netsh interface IPv4 show ipstats** commands, the settings below may help to alleviate resource contention.

NOTE

These same tunings can help with environments where there are no discarded packets, but RX performance is lower than expected due to similar resource constraints.

Apply each of these suggestions one at a time to measure any differences in performance introduced by each change:

1. Increase Receive Buffers size to 4096. See Receive Buffers.
2. Increase Transmit Buffers size to 4096. See Transmit Buffers.

   NOTE

   Increasing the TX buffer size is not typically needed. Bottlenecks are usually caused by RX packets coming into the adapter and getting processed by the CPU. However, the TX buffer setting is worth testing, in case it does result in a performance benefit for your workload.

3. Verify the CPU Affinity for the device, and use the CPU cores local to the adapter. Best performance is typically achieved using CPU cores local to the device. However, there are situations where using remote cores or all cores may be beneficial, depending on the application and workload. See CPU Affinity.
4. Update the RSS Base Processor Number (if needed). See RSS Base Processor Number:
   a. If the device is on a NUMA node other than NUMA 0, set the RSS Base Processor Number to the first physical CPU core for the local NUMA node. For example, on a server with 2 CPUs, each with 40 cores, the cores are split 0 to 39 for NUMA 0, and 40 to 79 for NUMA 1 (on a server with consecutive CPU core layout).
   b. If the device is installed on NUMA 1, the RSS Base Processor Number should be set to 40 for the first CPU core on NUMA 1.

   c. If the device is installed on NUMA 0, it is best to avoid Core 0, which is used for OS administration tasks. For NUMA 0, use Cores 1 or 2 for the RSS Base Processor, depending on the CPU layout. It is best to avoid using Core 0 for RSS or for application traffic.

5. Reduce the Maximum Number of RSS Queues to 8 (or fewer for lower core count servers). See Maximum Number of RSS Queues.
6. Experiment with the Interrupt Moderation Rate threshold. See Interrupt Moderation Rate.
   a. Experiment with the values, *Adaptive*, *High*, and *Low*, to see if there are any resulting differences in the performance or the number of discarded packets.

   NOTE

   When a higher ITR setting is used, the interrupt rate is lower, which can result in better CPU performance for higher link speeds. With lower link speeds, a lower ITR setting may give best performance to service interrupts faster.

7. For latency-sensitive workloads, try disabling Interrupt Moderation all together. See Interrupt Moderation.

4.11 Low Performance with SR-IOV/IOMMU Passthrough/VMs/Hyper-V

1. Verify the hardware compatibility to ensure that the hardware components support the required virtualization features like SRI-IOV and IOMMU passthrough.
2. Check the BIOS settings to ensure virtualization features such as Intel® VT-x and Intel® AMD-V are enabled. Additionally, verify that IOMMU is enabled. See BIOS Settings.
3. Check for the latest drivers, firmware, and NVM updates for the network cards. Older or incompatible drivers can cause performance issues.
4. Verify the resource allocation to ensure that the VMs or virtual functions (VFs), assigned to SRI-IOV, have adequate resources allocated. See SR-IOV/VMs for details to assign virtual cores (vCPUS) and RSS queues to a VM with SR-IOV.
5. Check the VM configuration to ensure the VMs are properly configured to utilize SR-IOV and IOMMU passthrough, and that the necessary virtual devices are assigned to the VMs.
6. Analyze the network configuration, such as the switches, routers, and firewall, to ensure they are properly configured to handle SR-IOV traffic.
7. Check for software conflicts and disable unnecessary services and applications that might be consuming resources or causing conflicts.

NOTE

The steps above are recommended to achieve best performance with SR-IOV enabled, for most VM workloads. When SR-IOV is disabled, VMQ can be used with standard virtual network adapters to enhance network performance for VMs. However, when SR-IOV is enabled, and a VM uses the VF of the SR-IOV-enabled network adapter, VMQ is generally not applicable for that VF, as the VM has direct hardware access to the adapter. Instead, VMQ might still be relevant for non-SR-IOV virtual network adapters in the same Hyper-V environment. The availability and usage of VMQ depends on the hardware capabilities, Hyper-V settings, and the specific VM configurations.

4.12 Consider Hardware Upgrades

If the server's hardware is outdated or insufficient for the workload, consider upgrading components, such as CPU, memory, or storage, to improve performance. Use the SSU utility in SSU Output to populate a list of all server hardware components to help identify bottlenecks in hardware resources.

5.0 Performance Tuning with Intel Driver Settings (i40ea/i40eb/icea)

This section contains device specific tunings for 700 Series and 800 Series devices using *i40ea*/*i40eb*/*icea* drivers. There are 2 options to set the Advanced Driver Settings for Intel® Ethernet devices:

- Option 1: Device Manager X710/E810 Adapter, Properties Advanced tab (right click).
- Option 2: Windows **PowerShell** commands.

This section uses mostly **PowerShell** examples to view and configure adapter tunings.

5.1 CPU Affinity

When passing traffic on multiple network ports using an I/O application that runs on most or all of the cores in your system, consider setting the CPU Affinity for that application to fewer cores. This should reduce CPU utilization and, in some cases, increase throughput for the device. The cores selected for CPU Affinity must be local to the Processor Node/Group of the affected network device. You may need to increase the number of cores assigned to the application to maximize throughput.

NOTE

Refer to your operating system documentation for more details on setting the CPU Affinity.

To set CPU Affinity:

1. Use this **PowerShell** command to list the cores that are local to a device:

```
Get NetAdapterRSS
```

2. List the RSS details for the device:

```
Get-NetAdapterRSS -Name <Adapter Name>
```

3. Set the CPU Affinity using **PowerShell**:

```
$process = Get-Process -Name <Application Name>
$process.ProcessorAffinity = 0x00005555
```

The CPU Affinity mask (0x00005555) is a hexadecimal bitmask that sets the CPU cores for the application, <*Application Name*>. This example sets the CPU Affinity to use the first 8 cores on the server with the local CPU Cores 0, 2, 4, 6, 8, 10, 12, 14 (0x00005555).

NOTE

Some workloads perform better when CPU Core 0 is not used for application traffic, due to contention with Windows administration tasks running on CPU Core 0. If performance issues are observed, try adjusting application traffic to use CPU cores above (excluding) CPU Core 0.

5.1.1 Pinning Application Threads to Specific Cores Using the Process Lasso Utility

The Process Lasso utility is primarily designed to manage and optimize the CPU usage of running processes on Windows. It also allows you to configure CPU affinity settings for specific applications, ensuring that they run on specific processor cores.

To pin application threads to cores:

1. Download and install Process Lasso from: https://bitsum.com/
2. Launch the utility and locate the target application. Using the list of currently running processes, locate the application for which you want to pin threads to specific cores.
3. Right-click on the target application and choose **CPU Affinity**.
4. In the **CPU Affinity** submenu, you will see a list of available CPU cores. Choose specific cores for the application.
5. Right-click on the application, navigate to Priority Class, and set the CPU priority for the application to control its access to CPU resources.

NOTE

After configuring **CPU Affinity** and the priority, you might need to restart the target application for the changes to take effect.

5.2 Flow Control

The Flow Control feature enables adapters to more effectively regulate traffic. Adapters generate Flow Control frames when their receive queues reach a predefined limit. Generating Flow Control frames signals the transmitter to slow transmission. Adapters respond to Flow Control frames by pausing packet transmission for the time specified in the Flow Control frame.

By enabling adapters to adjust packet transmission, Flow Control helps prevent dropped packets. You can improve RDMA performance by enabling Flow Control on all nodes, on the switch to which they are connected.

NOTES

- For adapters to benefit from this feature, link partners must support Flow Control frames.
- On systems running a Microsoft Windows Server operating system, enabling *QoS/Priority Flow Control disables link level Flow Control.
- Some devices support Auto Negotiation. Selecting this will cause the device to advertise the value stored in its NVM (usually "Disabled").

Layer 2 Flow Control can impact TCP performance considerably, and is disabled by default on the driver. This is the recommended setting for most workloads. A potential exception to this recommendation is where very bursty traffic (bursts not long in duration) can result is packet drops.

To manage Flow Control:

- List the current value:

```
Get-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Flow
Control"
```

- Enable Flow Control:

```
Set-SerialPort -PortName <Port Name> -FlowControl XonXoff
```

- Disable Flow Control:

```
Set-SerialPort -PortName <Port Name> -FlowControl None
```

5.3 Interrupt Moderation

Interrupt Moderation generally offers advantages in terms of system performance and resource utilization, and is enabled by default in the driver. The Interrupt Moderation Rate can be adjusted for the workload to customize how frequently interrupts are sent to the CPU for processing.

However, some use cases, such as low latency environments, benefit from having Interrupt Moderation disabled. This ensures the immediate processing of interrupts when packets arrive at the adapter, and removes any additional overhead that Interrupt Moderation might have introduced into the system (such as when aggregating interrupts and calculating rate moderation intervals).

To manage Interrupt Moderation:

- List the current value:

```
Get-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Interrupt
Moderation"
```

- Enable Interrupt Moderation:

```
Set-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Interrupt
Moderation" -DisplayValue "Enabled"
```

- Disable Interrupt Moderation:

```
Set-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Interrupt
Moderation" -DisplayValue "Disabled"
```

NOTE

Disabling Interrupt Moderation can result in higher CPU utilization as a result of the system handling a larger number of interrupts.

5.4 Interrupt Moderation Rate

The Interrupt Moderation Rate sets the Interrupt Throttle Rate (ITR), and the ITR moderates the rate at which Transmit and Receive interrupts are generated. When an event occurs, such as receiving a packet (RX), the adapter generates an interrupt. The interrupt interrupts the CPU and any applications running at that time, and calls on the driver to handle the packet.

At greater link speeds, more interrupts are created, and the CPU rates also increase, resulting in poor system performance. When you use a higher ITR setting, the interrupt rate is lower, resulting in better CPU performance.

NOTE

A higher ITR rate also means that the driver has more latency in handling packets. If the adapter is handling many small packets, it is better to lower the ITR so that the driver can be more responsive to incoming and outgoing packets.

To manage the Interrupt Moderation Rate:

1. List the current value:

```
Get-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Interrupt
Moderation Rate"
```

2. Set the Interrupt Moderation Rate:

```
Set-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Interrupt
Moderation Rate" -DisplayValue "Adaptive"
Set-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Interrupt
Moderation Rate" -DisplayValue "High"
Set-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Interrupt
Moderation Rate" -DisplayValue "Low"
```

5.5 Jumbo Frames

When the expected traffic environment consists of large blocks of data being transferred, it might be beneficial to enable the Jumbo Frames feature. Jumbo Frames support is enabled by changing the Maximum Transmission Unit (MTU) to a value larger than the default value of 1500. This allows the device to transfer data in larger packets within the network environment. This setting might improve throughput and reduce CPU utilization for large I/O workloads. However, it might impact small packet or latency-sensitive workloads.

Jumbo Packets should only be enabled if ALL devices across the network support them and are configured to use the same frame size.

To manage Jumbo Frames:

1. List the current value:

```
Get-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Jumbo
Packet"
```

2. Enable Jumbo Frames:

```
Set-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Jumbo
Packet" -DisplayValue 9014
```

3. Restore Default Frame Size:

```
Set-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Jumbo
Packet" -DisplayValue 1514
```

NOTE

End-to-end hardware must support Jumbo Frames; otherwise, packets are dropped. See *Intel® Ethernet Adapters and Devices User Guide*, Section 4.12, "Jumbo Frames" for restrictions.

5.6 Receive Buffers

Receive Buffers are allocated in the host memory and are used to store packets received by the adapter. Each received (RX) packet requires at least one Receive Buffer, and each buffer uses 2 KB of memory. You can choose to increase the number of Receive Buffers if you notice a significant decrease in the performance of received traffic or if you see dropped/discarded RX packets.

To manage receive buffers:

1. List the current value (default is 512 bytes):

```
Get-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Receive
Buffers"
```

2. Increase Receive Buffers to 4096 bytes:

```
Set-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Receive
Buffers" -DisplayValue 4096
```

NOTE

See Transmit Buffers for equivalent tuning on the Transmit (TX) side, where applicable.

5.7 RSS Queues and RSS Processors

When Receive Side Scaling (RSS) is enabled, all of the receive data processing for a particular TCP connection is shared across multiple processors or processor cores. Without RSS, all processing is performed by a single processor, resulting in less efficient system cache utilization.

5.7.1 Maximum Number of RSS Queues

In Windows, it is possible to set the number of Receive Side Scaling (RSS) queues and select which base processor the RSS queues should start on, for a given interface. It can be beneficial to increase or decrease the number of RSS queues/processors, based on the workload and volume of the ingress RX packets from the application.

To manage the maximum number of RSS queues:

1. List the current value:

```
Get-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Maximum
Number of RSS Queues"
```

2. Set the maximum number of RSS queues:

```
Set-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Maximum
Number of RSS Queues" -DisplayValue <NumberOfQueues>
```

NOTE

If Hyper-Threading is enabled in BIOS, the driver does not utilize the hyper-threaded cores for RSS, and this can impact performance for applications using both physical and logical (HT) cores to which the RSS Queues are mapped.

5.7.2 RSS Base Processor Number

In addition to setting the Maximum Number of RSS Queues for the device port, Intel also recommends that you adjust the RSS Base Processor Number to specify the CPU core at which to start mapping the RSS queues.

The RSS queue-to-processor mappings are allocated sequentially, starting with the RSS Base Processor number, and are incremented by one local CPU core for each RSS queue, up to the Maximum Number of RSS queues/processors.

This setting is recommended for environments that have:

- Adapters with more than one active port (such as 2x25G, 4x25G, 8x10G).
- Adapters on a CPU socket or NUMA node that does not start at Core 0.
- Adapters on a CPU socket or NUMA node that does start at Core 0. Set the Base RSS Processor to start at the next physical, local core, to avoid Core 0, which is used for Windows OS administration tasks.

If you have multiple ports installed in a system, the RSS queues of each adapter port can be adjusted to use non-overlapping sets of processors within the adapter's local Non-Uniform Memory Access (NUMA) Node/Socket.

Change the RSS Base Processor Number for each adapter port, so that the combination of the base processor and the Max Number of RSS Processor settings ensure non-overlapping cores.

For Microsoft Windows systems, do the following:

1. Identify the adapter ports to be adjusted and inspect their RssProcessorArray:

   ```
   Get-NetAdapterRSS -Name <Adapter Name>
   ```

2. Identify the processors with NUMA distance 0. These are the cores in the adapter's local NUMA Node/Socket, and they provide the best performance.

   NOTE

   If Hyper-Threading is enabled in BIOS, the driver does not utilize the hyper-threaded cores for RSS, and this can impact performance for applications using both physical and logical (HT) cores.

3. Adjust the RSS Base Processor Number on each port to use a non-overlapping set of processors within the local set of processors. You can do this manually or using the following **PowerShell** command:

   ```
   Set-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "RSS Base
   Processor Number" -DisplayValue <RSS Base Proc Value>
   ```

4. Verify the right values have been set:

   ```
   Get-NetAdapterAdvancedproperty -Name <Adapter Name>
   ```

Example:

For a 4-port adapter with local processors 0, 2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30:

Set the Maximum Number of RSS Processors to 8 on the 4 interfaces:

```
Set-NetAdapterAdvancedProperty -Name "E810-1" -DisplayName "Maximum Number of RSS
Queues" -DisplayValue 8
Set-NetAdapterAdvancedProperty -Name "E810-2" -DisplayName "Maximum Number of RSS
Queues" -DisplayValue 8
Set-NetAdapterAdvancedProperty -Name "E810-3" -DisplayName "Maximum Number of RSS
Queues" -DisplayValue 8
Set-NetAdapterAdvancedProperty -Name "E810-4" -DisplayName "Maximum Number of RSS
Queues" -DisplayValue 8
```

Set the RSS Base Processor Number to 0, 8, 16 and 24 on the 4 interfaces:

```
Set-NetAdapterAdvancedProperty -Name "E810-1" -DisplayName "RSS Base Processor
Number" -DisplayValue 0
Set-NetAdapterAdvancedProperty -Name "E810-2" -DisplayName "RSS Base Processor
Number" -DisplayValue 8
Set-NetAdapterAdvancedProperty -Name "E810-3" -DisplayName "RSS Base Processor
Number" -DisplayValue 16
Set-NetAdapterAdvancedProperty -Name "E810-4" -DisplayName "RSS Base Processor
Number" -DisplayValue 24
```

NOTE

Sometimes better performance is achieved when RSS does not use Core 0, which is used for Windows OS administration tasks. If your adapter is on NUMA Node 0, experiment with the RSS Base Processor Number, starting at the next physical, local core (such as Core 2 in the example above where local cores are 0, 2, 4, 6, 8...).

5.7.3 RSS Load Balancing Profile

Setting the RSS load balancing profile, Advanced Setting, to **ClosestProcessor** can significantly reduce CPU utilization. However, in some system configurations (such as a system with more Ethernet ports than processor cores), the **ClosestProcessor** setting may cause transmit and receive failures. Changing the setting to **NUMAScalingStatic** resolves the issue.

NOTES

- Default RSS load balancing profile on 700 Series devices is **ClosestProcessor**.
- Default RSS load balancing profile on 800 Series devices is **NUMAScalingStatic**.

1. List the current value:

```
Get-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "RSS load
balancing profile”
```

2. Set the RSS load balancing profile:

```
Set-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "RSS load
balancing profile" -DisplayValue “NUMAScalingStatic”
```

5.8 Speed and Duplex

Check the Speed and Duplex settings to ensure the adapter and switch are running at full duplex.

Set both the Speed and Duplex to either Forced or Auto-Negotiate. Both the adapter and the switch must have the same Speed and Duplex configuration. keepwithnext

1. List the current value:

```
Get-NetAdapterAdvancedProperty -name <Adapter Name> -DisplayName "Speed &
Duplex"
```

2. Set 100 Gbps Full Duplex:

```
Set-NetAdapterAdvancedProperty -name <Adapter Name> -DisplayName "Speed &
Duplex" -DisplayValue "100 Gbps Full Duplex"
```

3. Set 25 Gbps Full Duplex:

```
Set-NetAdapterAdvancedProperty -name <Adapter Name> -DisplayName "Speed &
Duplex" -DisplayValue "25 Gbps Full Duplex"
```

4. Set Auto Negotiation (default):

```
Set-NetAdapterAdvancedProperty -name <Adapter Name> -DisplayName "Speed &
Duplex" -DisplayValue "Auto Negotiation"
```

5.9 Transmit Buffers

Transmit Buffers are allocated in the host memory and are used to store packets that are being transmitted out of the adapter. Each transmitted (TX) packet requires at least one Transmit Buffer, and each buffer uses 2 KB of memory. The default value is set to 512 bytes and is typically big enough for the driver to handle transmitted packets. You can choose to increase the number of Transmit Buffers if you notice a significant decrease in the performance of the egress (TX) traffic.

To manage the transmit buffers:

1. List the current value (default is 512 bytes):

```
Get-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Receive
Buffers"
```

2. Increase Receive Buffers to 4096 bytes:

```
Set-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Receive
Buffers" -DisplayValue 4096
```

NOTE

See Receive Buffers for equivalent tuning on the Receive (RX) side, where applicable.

5.10 Optimization for Specific Usage Models

Table 1 provides optimization guidance for specific server usage models.

For best performance, disabling the Firewall is recommended when possible. However, it is understandable that disabling the Firewall is not always possible for production environments and data centers.

General guidance for systems that support more than one NUMA node:

- Set the preferred NUMA node on each adapter to achieve better scaling across NUMA nodes.

  NOTE

  If the preferred NUMA node is not set, the scheduler grabs from any and all available cores, including cross-node, which can have very unreliable results.

- Pin the application threads to the CPU cores on the NUMA node local to adapter.

  See CPU Affinity.

- Set the RSS Base Processor to a CPU core on the local NUMA node.

  See RSS Base Processor Number.

It is generally best to avoid Core 0 due to the administrative overhead tasks processed by the OS on Core 0.

NOTE

Some settings are applied system-wide and can impact performance across all Ethernet devices on a server. Settings such as BIOS, Power Management, and Firewall are impacted.

Table 1. Server Usage Optimization Models

Optimize For	Useful For	Optimization Tasks
Quick response and Low Latency	Video, audio, and High Performance Computing Cluster (HPCC) servers	• Disable hyper-threading. • Minimize or disable interrupt moderation rate. • Disable offload TCP segmentation. • Disable jumbo packets. • Increase transmit descriptors. • Increase receive descriptors. • Increase RSS queues. • Use an established benchmark like: netperf -t TCP_RR netperf -t UDP_RR • Pin benchmark to a single core on NUMA node local to the adapter.
Throughput	Data backup/retrieval and file servers	• Enable jumbo packets.[1] • Increase transmit descriptors.[2] • Increase receive descriptors. [2] • On systems that support NUMA, set the Preferred NUMA Node on each adapter to achieve better scaling across NUMA nodes.
CPU utilization	Application, web, mail, and database servers	• Maximize interrupt moderation rate. • Keep the default setting for the number of receive descriptors; avoid setting large numbers of receive descriptors. • Decrease RSS queues. • In Hyper-V environments, decrease the maximum number of RSS CPUs.
IP Forwarding	Single port to port forwarding	• Disable hyper-threading. • Disable hardware-assisted virtualization, such as Intel® VT-d (or IOMMU) and Intel® AMD-V. • Turn off power management. • Minimize or disable interrupt moderation rate. • Increase transmit descriptors. • Increase receive descriptors. • Enable port forwarding
Virtualization	SR-IOV, IOMMU, Hyper-V, VMs	• Can achieve efficient resource utilization, improved performance, and enhanced security. • Enable Virtualization in BIOS. • Enable hardware-assisted virtualization, such as Intel® VT-d (or IOMMU) and Intel® AMD-V. • Enable SR-IOV on the interface in device manager, Hyper-V vSwitch manager, and VM hardware acceleration settings.[3] • Allocate appropriate hardware resource. • Use dynamic memory allocation for VMs.

continued...

Optimize For	Useful For	Optimization Tasks
		• Proper storage configuration. • Optimize networking.
Teaming	Configure 2x 700 or 800 series interfaces using a supported Teaming method	• Teaming ensures uninterrupted connectivity, provides load balancing, and improves overall network reliability. • Teaming can be configured using GUI, PowerShell, or Intel® ANS. • Update supported drivers and NVM/firmware. • Adjust team member properties. • Select the appropriate teaming mode. • Configure load balancing algorithms.
Notes: 1. Enable jumbo frames for workloads that require large amounts of data that need to be transferred efficiently. See BIOS Settings. 2. In situations where low RX, low TX, or dropped packets are observed, experiment with increasing the RX and TX buffer sizes. See Low Receive (RX) Performance and/or Discarded Packets. 3. Best performance is generally achieved with SR-IOV enabled, using VF(s) for the SR-IOV enabled network adapter. For environments with SR-IOV disabled, VMQ can be used with standard virtual network adapters to enhance network performance for VMs. See SR-IOV/VMs.		

5.11 Performance Profiles

Performance Profiles are supported on Intel® 100GbE Adapters and allow you to quickly optimize the performance of your Intel® Ethernet Adapter. Selecting a performance profile automatically adjusts some Advanced Settings to their optimum setting for the selected application. For example, a standard server has optimal performance with only two RSS queues, but a web server requires more RSS queues for better scalability.

Intel® PROSet is required for Performance Profiles. See the Intel® Ethernet Adapters and Devices User Guide, Section 3, "About Intel® PROSet" for steps to install Intel® PROSet.

To change settings in Intel® PROSet, use the Performance Profiles in Table 2.

Table 2. Performance Profiles

Profile	Description
Standard Server	This profile is optimized for typical servers.
Web Server	This profile is optimized for IIS and HTTP-based web servers.
Virtualization Server	This profile is optimized for Microsoft's Hyper-V virtualization environment.
Storage Server	This profile is optimized for Fibre Channel over Ethernet or for iSCSI over DCB performance. Selecting this profile disables SR-IOV and VMQ.
Storage + Virtualization	This profile is optimized for a combination of storage and virtualization requirements.
Low Latency	This profile is optimized to minimize network latency.

NOTE

Not all options are available on all adapter/operating system combinations.

If you have selected the Virtualization Server Profile or the Storage + Virtualization Profile, and you uninstall the Hyper-V role, you should select a new profile.

This setting is found on the Advanced tab of the device's Device Manager property sheet or in the Adapter Settings panel in the Intel® PROSet Adapter Configuration Utility.

- To change this setting in Windows **PowerShell**, use this cmdlet:

```
Set-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Profile" -
DisplayValue "Standard Server"
```

For example:

```
Set-IntelNetAdapterSetting -Name <Adapter Name> -DisplayName "Profile" -
DisplayValue "Standard Server"
```

5.12 Reset Driver Settings (Default Values)

- To reset a specific driver setting to the default value:

```
Reset-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName <Display
Name>
```

- To reset the maximum number of RSS queues:

```
Reset-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "Maximum
Number of RSS Queues"
```

- To reset all driver settings to the default values (*icea*/*i40ea*/*i40eb*):

```
Reset-NetAdapterAdvancedProperty -Name <Adapter Name> -DisplayName "*"
```

6.0 Platform Tuning (Driver Non-Specific)

6.1 General Tuning

1. Use the proper cabling for your device.
2. Verify PCIe compatibility with the Intel® Ethernet device(s).

   See Check PCI Express (PCIe) Slot Capabilities
3. Adjust BIOS settings for performance workloads. See BIOS Settings.
4. Disable the Firewall (if possible).
5. Adjust the Network Adapter properties for the workload.

   Common options to consider are:
   - Speed and Duplex
   - Jumbo Packet
   - Interrupt Moderation
   - Receive and Transmit Buffers
   - Offloading features that offload processing tasks from the CPU to the network adapter, such as:
     - Checksum offloading
     - TCP/IP offloading
     - Receive Side Scaling (RSS)

   See Baseline Performance Measurements and Tuning Methodology.
6. Use CPU cores local to the adapter for application traffic. See CPU Affinity.
7. Increase the number of TCP and Socket resources from the default value.
   - For Windows-Based systems, no system parameters have been identified that significantly impact performance, other than TCP Window Size.

6.2 BIOS Settings

The specific BIOS settings and options can vary depending on the hardware vendor, model, and BIOS version.

Set the BIOS settings as follows:

1. Disable or enable Hyper-Threading.

   NOTE

   Depending on the desired configuration, select the option to enable or disable Hyper-Threading.

2. Set the Power Management Setting to Performance or High Performance to ensure that the system operates at maximum performance levels.
3. Disable the CPU C-states, or set them to a lower level (such as C1E).

SR-IOV specific BIOS settings:

1. Enable SR-IOV support.
2. Enable the virtualization technology (Intel® VT-x for Intel® Processors or Intel® AMD-V for AMD® Processors) to enable hardware-assisted virtualization, which is a prerequisite for SR-IOV functionality.
3. Enable Intel® VT-d or AMD® IOMMU to provide hardware support for input/output virtualization, which is necessary for SR-IOV.

6.3 Power Management Settings

By default, Windows may apply power-saving settings to network adapters, which can affect performance.

To optimize performance:

1. Open Device Manager.
2. Expand the Network Adapters category.
3. Right-click on your network adapter and select Properties.
4. In the Power Management tab, uncheck the option:

   Allow the computer to turn off this device to save power.
5. Click**OK** to apply the changes.

NOTE

See *Intel® Ethernet Adapters and Devices User Guide*, Section 4.21 "Power Options" for additional power tuning and power saving options.

6.4 Additional Power Management Guidance

A high-level overview of the power management features in 3rd Generation Intel® Xeon® Scalable Processors, as well as guidance on how these features can be integrated at a platform level, can be found in this document:

Power Management - Technology Overview Technology Guide

7.0 Operating System Tuning

7.1 Firewalls

Firewalls can impact performance, particularly latency performance. Disable the Firewall if it is not required.

7.2 OS Version

Most modern versions of the Windows OS are reasonably well optimized for performance. However, depending on your use case, updating the OS can provide improved performance.

If you are seeing a performance issue that appears to be OS-specific, see Known Issues on page 8 to see if there is a mitigation to use a newer version of the OS.

7.3 Application Settings

Often a single thread (which corresponds to a single network queue) is not sufficient to achieve maximum bandwidth. You should experiment with increasing the number of threads used by your application if possible.

Consider using tools like **Task Manager** or **PowerShell** to pin application threads to the NUMA node or CPU cores local to the network device. For some workloads, such as storage I/O, moving the application to a non-local node provides benefit.

7.4 IP Port Forwarding

IP Port Forwarding is a network configuration that can be enabled to direct packets from one network interface to another network interface, using the destination IP address. When IP Port Forwarding is enabled, the system can route data between different networks, and act as an intermediary for network traffic.

IP Port Forwarding allows routers and network devices to direct traffic between different networks. However, IP Port Forwarding can indirectly impact performance, depending on the efficiency of the routing process, the hardware capabilities of the network devices, the routing protocols used, and the overall network design.

Windows OS supports IP Port Forwarding. However, it is disabled by default, because most consumer-oriented systems do not require routing capabilities. To use the IP Port Forwarding capabilities, you must enable IP Port Forwarding in the Windows OS.

NOTE

The IP Port Forwarding setting is applied system-wide and is enabled or disabled for all Ethernet devices in the system.

To manage IP Port Forwarding:

1. View the current settings:

```
Get-NetIPInterface | Select-Object InterfaceAlias, Forwarding
```

2. Enable IP Port Forwarding:

```
Set-NetIPInterface -InterfaceAlias <Adapter Name> -Forwarding Enabled
```

3. Disable IP Port Forwarding:

```
Set-NetIPInterface -InterfaceAlias <Adapter Name> -Forwarding Disabled
```

7.5 LLDP

Link Layer Discovery Protocol (LLDP) support is not natively built into Windows operating systems. However, there are third-party tools available that can be used to enable LLDP functionality on Windows machines.

NOTES

1. Enabling LLDP on Windows has no effect on system performance. LLDP packets are relatively light, and the protocol runs in the background without consuming a lot of resources.
2. LLDP is not a requirement for basic Windows system performance. However, in specific network management scenarios where detailed device information is required, it can be useful.

8.0 Tuning Recommendations for Common Performance Scenarios

8.1 SR-IOV/VMs

The tunings listed below can be used to improve performance between VM to VM and VM to Host when SR-IOV is enabled in Hyper-V. SR-IOV must be enabled in BIOS prior to the steps listed below, see BIOS Settings.

NOTE

See *Intel® Ethernet Adapters and Devices User Guide*, Section 4.31.1, "Single Root I/O Virtualization (SR-IOV)" for detailed steps to configure SR-IOV.

From the Host OS on both Host 1 and Host 2:

1. Enable RSS on the PF and the vSwitch.

```
Enable-NetAdapterRss -name "ADAPTER_NAME"
```

2. Enable 4 queues per VF:

```
Set-VMNetworkAdapter -VMName "YOUR_TEST_VM_NAME" IovQueuePairsRequested 4
Get-VmNetworkAdapter -VMName * | where {$_.SwitchName -eq
"YOUR_TEST_SWITCH_NAME"} | Set-VmNetworkAdapter -IovQueuePairsRequested 4
```

3. Ensure the VMs have at least twice as many vCPUs as Receive Side Scaling (RSS) queues. In this case, set the number of total processors in the VM to 8.
   a. To do this, use Method 1 – PowerShell or Method 2 – Windows GUI.

8.1.1 Method 1 – PowerShell

```
# Get all the VMs on the Hyper-V host
$VMs = Get-VM

# Loop through each VM
foreach ($VM in $VMs) {
    # Get the number of vCPUs assigned to the VM
    $vCPUs = $VM.CPUCount
    # Get the number of RSS queues for the VM
    $RSSQueues = $VM.VirtualMachineSubnet
    $RSSQueuesCount = $RSSQueues | Measure-Object | Select-Object -ExpandProperty
Count
    # Compare the number of vCPUs with the number of RSS queues
    if ($vCPUs -lt ($RSSQueuesCount * 2)) {
        Write-Host "VM $($VM.Name) does not have at least twice as many vCPUs as
RSS queues."
        # You can perform additional actions here, such as adjusting the vCPUs or
notifying an administrator.
    }
    else {
        Write-Host "VM $($VM.Name) meets the requirement."
    }
}
```

8.1.2 Method 2 – Windows GUI

1. Open Hyper-V Manager on your Windows Server.
2. In the left pane, select the Hyper-V host server where the VMs are located.
3. In the center pane, you'll see a list of VMs running on the selected host.
    a. Right-click on the VM and select Settings from the context menu to open the VM Settings window.
4. In the left pane of the VM Settings window, select Processors.
5. In the right pane, find the Virtual Machine Processor Settings.
    a. Note the value specified in the *Number of virtual processors* field.
    This represents the number of vCPUs assigned to the VM.
6. In the left pane, switch to the Network Adapter section.
7. In the right pane, find the network adapters assigned to the VM.

NOTE

The number of RSS queues specified for the VM can vary depending on the network adapter and configuration.

8. Compare the number of vCPUs with twice the number of RSS queues.
    If the vCPUs are less than twice the RSS queues, the requirement is not met.
9. Repeat Steps 3 through 8 for the other VMs on the Hyper-V host.
10. For Windows Server 2012 and later, issue the following command while the VM is in the off state:

```
Set-VMProcessor -VMName "YOUR_VM_Name" -HwThreadCountPerCore 1 -Count 8
```

The HwThreadCountPerCore setting refers to the hardware thread count for each CPU core inside the VM, and specifies the number of simultaneous hardware threads that can be executed on each physical core of a processor in the VM.

The default value is HwThreadCountPerCore=0, which disables the setting and allows multiple threads for each CPU core inside the VM. The default setting of 0 essentially enables hyper-threading inside the VM, which can have a severe impact on VM performance.

For most VM workloads, Intel recommends using:

```
-HwThreadCountPerCore 1 -Count N
```

This configures 1 thread per core inside the VM, using *N* CPU cores on the host. This gives optimal performance for most VM workloads.

NOTE

This command is available only in Microsoft Windows Sever 2012 and later.

In the OS of both guest VM1 and guest VM2:

1. Set the RSS queues to 4 for all VFs in the guest OS:

```
Set-NetAdapterRss -InterfaceDescription *adaptive* NumberOfReceiveQueues 4
```

2. Update the number of queues in the guest OS:

```
Set-NetAdapterAdvancedProperty -Name "your_adapter_name_from_guest_os" -
DisplayName "Maximum Number of RSS Queues" -DisplayValue "8 Queues"
```

NOTE

In the locations where there are settings for the number of queues, the value can be anything from 1 to 16. If you want more total throughput, increase the number of queues. When updating the number of queues, you must set *IovQueuePairsRequested* to a value that is equal to or greater than the number of queues you want to use in the VM.

8.1.3 PCI Express (PCIe) Passthrough

The PCIe passthrough feature is commonly used in virtualization setups to assign dedicated hardware to virtual machines (VMs). This can be for improved performance or to run applications that require direct access to specific hardware.

To setup PCIe passthrough in Windows:

1. Ensure that the hardware supports IOMMU (Input/Output Memory Management Unit), Intel® VT-d, or Intel® AMD-Vi (AMD) virtualization technology.
2. Enable virtualization support (Intel® VT-x or Intel® AMD-V) in server BIOS/UEFI settings.
3. Install hypervisor on Windows host system.
4. Create virtual machine (VM) in hypervisor and allocate resources (CPU, RAM, storage) to the VM.

5. Identify the PCIe device to pass through, such as the network card or any other PCIe device, and associate it with the VM.
6. Inside the VM, install the necessary drivers for the assigned PCIe device.

NOTE

Without PCIe passthrough, VMs share host resources, and the hypervisor manages resource allocation, which has an impact on performance (impact on CPU, memory, and I/O). In addition, VMs have limited access to physical hardware, with the hypervisor acting as an intermediary.

8.2 Adapter Teaming

Adapter teaming is a feature that allows multiple network adapter ports to be configured into a single virtual network interface. Windows adapter teaming provides many benefits, including capabilities such as fault tolerance, high availability, load balancing, increased bandwidth, and integration with server virtualization.

When you create a team, and all members of the team support Performance Profiles, you are asked which profile to use, and the profile is synchronized across the team. If there is not a profile that is supported by all team members, then the only option available is Use Current Settings, and the team is created normally.

Adding an adapter to an existing team works in much the same way. If you attempt to team an adapter that supports Performance Profiles with an adapter that does not, the profile on the supporting adapter is set to the Custom Settings, and the team is created normally.

Common teaming modes in Windows:

- Switch Independent
- Switch Dependent
- Static
- Link Aggregation Control Protocol (LACP)

NOTE

Enabling LLDP on a Windows machine with LACP teaming can provide additional visibility into the network and the Windows host for neighboring devices. However, the functionality of LACP itself is not directly affected by the presence of LLDP.

There are different ways to configure adapter teaming, depending on the operating system and the network infrastructure. Teaming is no longer configurable through Intel® PROSet ACU. On Microsoft Windows Server 2019, Microsoft Windows* 10 Version 1809, and later, use Windows **PowerShell** or Windows GUI.

Common methods used are:

- Command-Line configuration using **PowerShell** commands
- GUI-based configuration using a Microsoft GUI service
- Configuration through vendor-specific tools and scripting, or automation

8.2.1 Method 1 – Teaming Configuration Using PowerShell Commands

- Static Teaming:

```
New-NetLbfoTeam -Name <TeamName> -TeamMembers "Adapter1", "Adapter2" -
TeamingMode Static
```

- Switch Independent Teaming:

```
New-NetLbfoTeam -Name <TeamName> -TeamMembers "Adapter1", "Adapter2" -
TeamingMode SwitchIndependent
```

- LACP Teaming:

```
New-NetLbfoTeam -Name <TeamName> -TeamMembers "Adapter1", "Adapter2" -
TeamingMode LACP
```

NOTE

LACP is a switch dependent teaming mode, hence switch level LACP configuration is required.

8.2.2 Method 2 – Teaming Configuration Using Windows GUI

1. Click the Server Manager icon on the Taskbar to open Server Manager.

   Or search for Server Manager in the Start menu to open it.
2. In Server Manager, in the left-hand pane, click Local Server.
3. In the main window, locate the Network Adapters section.
4. Click on the link next to NIC Teaming (such as Enabled or Disabled).

   The NIC Teaming window opens.
5. From the Tasks drop-down menu, select New Team.
6. In the New Team window, provide a name for the team in the*Team Name* field.
7. Check the boxes of the network adapters you want to include in the team.
8. From the Teaming Mode drop-down menu, select the desired Teaming Mode:
   a. Static
   b. Switch Independent
   c. LACP (IEEE 802.3ad)
9. Configure any additional settings or parameters specific to the selected Teaming Mode.

   Available options might vary depending on the selected mode.
10. Click on the Additional Properties link to access the Advanced Settings, such as load balancing algorithms, standby adapter configuration, and teaming and standby adapter properties.
11. Once you have configured all the desired settings, click **OK** to create the team.

8.2.3 Switch Embedded Teaming (SET)

Switch Embedded Teaming (SET) is a specialized teaming that is specifically designed for virtualized environments, particularly in Hyper-V scenarios. While traditional NIC teaming modes like Switch Independent and Switch Dependent (Static or LACP) provide teaming at the host level, SET operates at the virtual switch level.

SET is integrated directly into the Hyper-V Virtual switch, allowing teaming functionality to be applied to virtual machines (VMs) without relying on external switches.

SET supports switch independent mode, where the team members can connect to different switches or switch ports, as well as switch dependent mode, where the team interfaces with the switch as a single logical interface. SET can be managed through **PowerShell** cmdlets.

NOTE

SET is only available in Windows Server 2016 and later versions, and its usage is primarily focused on Hyper-V virtualization scenarios.

To create SET through **PowerShell** cmdlets:

1. Create SET Switch:

```
New-VMSwitch -Name SETswitch -NetAdapterName "Ethernet 1","Ethernet 2" -
EnableEmbeddedTeaming $true -EnableIov $true
```

2. Check SET Switch:

```
Get-VMSwitchTeam -name SETswitch | fl
```

3. Set load balancing to dynamic:

```
Set-VMSwitchTeam -name SETswitch -LoadBalancingAlgorithm Dynamic
```

NOTE

Set Switch is configured in HyperVPort load balancing mode by default. Dynamic loading balancing is required to distribute traffic evenly across available PFs and optimize performance.

4. Check SET Switch:

```
Get-VMSwitchTeam -name SETswitch | fl
```

5. Attach the switch to the VM and enable SR-IOV on the network adapter:
    - Configure the VM settings:
        a. Select the Add Hardware option.
        b. Select the Network Adapter.
        c. Click on the Add button.
        d. In the Add Hardware window, select Network Adapter.
        e. Click the **Add** button.
        f. On the Network Adapter configuration page, under Hardware Acceleration, choose the option to Enable SR-IOV.

NOTE

Once SR-IOV is enabled for the virtual machine, it allows the virtual machine to directly access the physical network adapter and utilize the SR-IOV capabilities for improved network performance.