Abstract ........................................................................................................................................................................................... 2
NIC controllers ................................................................................................................................................................................... 2
Relevant drivers ................................................................................................................................................................................ 2
Applicable platforms ........................................................................................................................................................................ 2
HP Server Configuration .............................................................................................................................................................. 2
PCI Express requirements .............................................................................................................................................................. 2
NUMA architecture tuning ............................................................................................................................................................ 2
Memory configuration ................................................................................................................................................................. 2
Server power profile ................................................................................................................................................................. 3
Hyper-Threading ........................................................................................................................................................................ 3
Performance tuning for Linux ......................................................................................................................................................... 3
OS controlled processor C-states .................................................................................................................................................. 3
Managing daemons and services .................................................................................................................................................. 4
NIC driver parameters ................................................................................................................................................................. 4
IRQ management ............................................................................................................................................................................ 6
Performance tuning for Windows ................................................................................................................................................... 7
Accessing QLogic network adapter properties ......................................................................................................................... 7
Shutting down services ............................................................................................................................................................... 8
Abstract

The white paper lists tuning parameters and settings available in the Operating System (OS) and Network Interface Card (NIC) which can be tweaked to improve the performance for HP—NICs based on QLogic NetXtreme II technology. The user can make these adjustments to suit the workload being run.

We begin with the hardware configurations to eliminate any server induced bottlenecks for network adapter performance. Next, we list the Linux® OS and driver specifics that should be tweaked to get optimal NIC performance. The final section lists similar Windows® OS and NIC driver settings that need to be modified for performance improvements.

NIC controllers

The QLogic 10GbE and 20GbE NIC covered in this paper is based on the QLogic 57810S and QLogic 57840S controllers.

Relevant drivers

10 Gb Linux NIC Driver – bnx2x Version: 7.4.25-2 and later
10 Gb Windows NIC Driver – Component Version: 7.4.31.0 and later

Applicable platforms

Table 1. Applicable OS’s for the purposes of this white paper

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Applicable Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linux</td>
<td>RHEL 5.8, RHEL 5.9, RHEL 6.1, RHEL 6.2, RHEL 6.3, RHEL 6.4, RHEL 6.5, RHEL 7.0</td>
</tr>
<tr>
<td></td>
<td>SLES 11 SP1, SLES 11 SP2, SLES 11 SP3</td>
</tr>
<tr>
<td>Server</td>
<td>HP ProLiant Blades, Rack, and Tower Servers</td>
</tr>
</tbody>
</table>

Note

Performance variations may be observed across various families of Server hardware depending on the installed processor, memory configuration, the nature of the application and network traffic being run.

HP Server Configuration

This section lists generic Server hardware settings to ensure optimal configuration for network adapter performance.

PCI Express requirements

The HP QLogic 10Gb NICs are PCI Express (PCIe) 2.0 capable. A PCIe 2.0 x8 lane-width slot would be the minimum required for NIC placement. The HP QLogic 20Gb NICs are PCI Express (PCIe) 3.0 capable. A PCIe 2.0 x8 lane-width slot would be the minimum required for NIC placement but full 2x20G bandwidth would need PCIe 3.0 x8 slot.

NUMA architecture tuning

In Servers having multiple processors/NUMA nodes, the NIC performance is likely to be better if the NIC traffic is being handled by the NUMA node/cores closest to the PCIe slot the NIC is placed. Having remote NUMA nodes/cores process the network traffic induces additional latency as the traffic has to take extra hops across the QPI or HyperTransport links depending on the server platform. This may result in latency and throughput degradation for the NICs.

Memory configuration

In HP ProLiant servers, it is recommended to populate all memory channels with at least 1 DIMM to ensure maximum memory bandwidth and memory channel interleaving. It is also recommended to populate memory channels for every CPU installed. Distinct memory channels are identified by white colored slots. It is recommended to use identical DIMM’s (same rank, capacity, voltage, frequency) for all memory.
Server power profile

The ROM Based Setup Utility (RBSU) can be accessed by pressing F9 during server POST. For optimal performance, it is recommended to change the Server power profile from the default to “Maximum Performance.”

To set the maximum performance settings for the server, enable the following option:
Power Management Options -> HP Power Profile -> Maximum Performance

This setting disables processor C-states to prevent transition to low power states. It also sets the memory speed to be the maximum available and the processor frequencies to maximum supported.

Note
These performance settings could result in increased power consumption.

Hyper-Threading

The current generation of Intel® Sandy-bridge processors come with support for Hyper-Threading. It allows for sharing of a physical core between two logical cores.

It is recommended to turn ON Hyper-Threading for network applications that are multi-threaded. However, for applications that are latency dependent or CPU intensive, better performance can be achieved by disabling Hyper-Threading.

Performance tuning for Linux

In Linux, there are various OS network parameters that can be tweaked to improve IPv4 and IPv6 based network performance.

OS controlled processor C-states

The newer Linux kernels have moved to the ‘intel_idle’ instead of the ‘acpi_cpubfreq’ driver previously. The ‘intel_idle’ is a hardware driver not depending on the ACPI information for driving the HW states of the CPUs. As a result, the OS can override the processor related power settings done via RBSU.

To prevent enabling C-states, the following kernel boot parameter can be used:
intel_idle.max_cstate=0 and processor.max_cstate=1

To make this change permanent across server reboots, the boot parameter can be appended to the bootloader configuration file.
title Red Hat Enterprise Linux (2.6.32-358.el6.x86_64)
root (hd0,0)
kernal /vmlinuz-2.6.32-358.el6.x86_64 ro root=/dev/mapper/VolGroup-lv_root
...intel_idle.max_cstate=0 processor.max_cstate=1

Additionally, the ‘intel_idle’ driver can be blacklisted in /etc/modprobe.d/blacklist
Managing daemons and services

Processor frequency scaling
Newer flavors of the Linux kernel have a user-space frequency scaling program named 'cpuspeed'. Depending upon the workload, this service allows the OS to control the CPU clock speeds. To ensure maximum available CPU frequency is available to handle the network workload, it is recommended to set the scaling mode to "performance" or disabling the service altogether.

Set Scaling Mode:
```
echo performance > /sys/devices/system/cpu/cpu##/cpufreq/scaling_governor
```

Disable Service:
```
service cpuspeed stop
```

Firewall management
If the environment in which the servers will be deployed allow, it is recommended to disable the Firewall service in Linux.

To disable for IPv4 and IPv6
```
/etc/init.d/iptables stop
/etc/init.d/ip6tables stop
```

Verify:
```
iptables -L
ip6tables -L
```

NIC driver parameters

Adjusting number of queues
The 10GbE driver for QLogic NICs in Linux allows the flexibility for the number of IRQs to be created for a particular NIC port.

Step 1: Relevant driver parameter for setting IRQs
```
# modinfo bnx2x
...
parm: num_queues: Set number of queues (default is as a number of CPUs) (uint)
```

Step 2: Unloading and reloading the driver with num_queues set to 2
```
# modprobe –r bnx2x
# modprobe bnx2x num_queues=2
```

Step 3: Verifying two queues being created eth4-fp-0 and eth4-fp-1
```
# cat /proc/interrupts | grep eth4
84: 772 0 0 0 0 0 0 0 IR-PCI-MSI-edge eth4
86: 1 0 0 0 0 0 0 0 IR-PCI-MSI-edge eth4-fp-0
87: 1 0 0 0 0 0 0 0 IR-PCI-MSI-edge eth4-fp-1
```

Note
The number of queues available per port/function changes in a Virtual Connect or SRIOV based environment.
Interrupt moderation/coalescing
The rate with which the network adapter generates interrupts to the processor can be controlled by adjusting its Interrupt Coalescing parameters. This can be achieved via ‘ethtool’, an inbox Linux tool which is used for querying and modifying the settings of an Ethernet device.

Checking interrupt coalescing parameters

```
# ethtool -c eth4
Coalesce parameters for eth4:
Adaptive RX: off  TX: off
  rx-usecs: 24  
  rx-frames: 0
  rx-usecs-irq: 0
  rx-frames-irq: 0
  
  tx-usecs: 48  
  tx-frames: 0
  tx-usecs-irq: 0
  tx-frames-irq: 0
```

Setting interrupt coalescing parameters

```
ethtool –C eth4 rx-usecs 0 tx-usecs 0 rx-frames 1 tx-frames 1
```

Note
Disabling interrupt moderation/coalescing is recommended for improved latency. However, this is likely to result in increased CPU utilization.

Monitoring NIC statistics
In order to monitor NIC statistics such as bytes sent/received per port, traffic distribution across IRQs, CRC errors, pause frames, etc. the following command can be used:

```
ethtool -S eth4
```
**IRQ management**

**Irqbalance**
The 'irqbalance' service is used to distribute hardware interrupts (IRQs) across cores on a multi-core system. In the case where the IRQs need to be managed manually, the 'irqbalance' service should be disabled.

```
/etc/init.d/irqbalance stop
```

OR

```
/etc/init.d/irq_balancer stop
```

---

**Warning**
Disabling 'irqbalance' may affect the performance of other hardware/IO devices that depend on the service.

---

**IRQ affinity**
In cases that may require mapping IRQs to specific core, the following commands can be used. Please note, irqbalance must be disabled while setting affinities for IRQs.

**Read IRQ vector values**

```
# cat /proc/interrupts | grep eth4
```

<table>
<thead>
<tr>
<th>IRQ Vector</th>
<th>Value</th>
<th>Description</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>084</td>
<td>772</td>
<td>0 0 0 0 0 0 0 0</td>
<td>IR-PCI-MSI-edge eth4</td>
</tr>
<tr>
<td>086</td>
<td>1</td>
<td>0 0 0 0 0 0 0 0</td>
<td>IR-PCI-MSI-edge eth4-fp-0</td>
</tr>
<tr>
<td>087</td>
<td>1</td>
<td>0 0 0 0 0 0 0 0</td>
<td>IR-PCI-MSI-edge eth4-fp-1</td>
</tr>
</tbody>
</table>

**Assign bit mask to IRQ vectors**

```
# echo 1 > /proc/irq/86/smp_affinity
# echo 2 > /proc/irq/87/smp_affinity
```

To generalize, echo [Hexadecimal CPU mask] > /proc/irq/<IRQ VECTOR>/smp_affinity

---

**Note**
It is recommended to use IRQ affinities along with 'taskset' to affinitize applications to the same set of NUMA nodes as the IRQs. More information can be found using the command 'man taskset.'
Performance tuning for Windows

Accessing QLogic network adapter properties
The adapter parameters to be tuned can be accessed in the following way:
Start -> Control Panel -> Network and Sharing Center -> Change adapter settings

Once there, right click on the desired QLogic 10GbE or 20GbE network adapters and select properties. Under Properties, select the Advanced tab.

Properties:
Interrupt moderation
It specifies a time interval or a packet count before the NIC can generate an interrupt to the processor. It is enabled by default. For achieving better latency times, it is recommended to disable interrupt moderation.

Receive Side Scaling (RSS)
RSS allows spreading the incoming/receiving traffic across multiple CPUs, resulting in parallel and efficient processing of network traffic. To enable this feature:
1. Set value for Receive Side Scaling: Enabled
2. At Command Prompt: netsh int tcp set global rss=enabled

Note
RSS only utilizes the physical cores. In an Intel processor based system, the Hyper-Threaded cores are not used for RSS.

Maximum number of RSS queues
This property allows selecting the number of queues that are available to process network receive traffic. Depending on the use case, the value can be adjusted accordingly (Auto, 2, 4, 8, 16).

Transmit and receive CPU affinity
The QLogic Windows driver allows the user to set CPU affinities for single receive and transmit queue on a per-port basis.
1. RSS and Virtual Machine Queues (VMQs) must be disabled in advanced properties.
2. Disable RSS at OS level: netsh int tcp set global rss=disabled
3. Assign Receive CPU affinity (between 0 – 63)
4. Assign Transmit CPU affinity (between 0 – 63)
5. Repeat, depending on use case, for another NIC port

Transmit and receive buffers
QLogic 10GbE or 20GbE adapters allow for manual configuration of transmit and receive buffers. Depending on the network workload (transmit intensive or receive intensive) the buffer values can be changed to reduce the number of dropped packets and improve performance overall. Increasing the buffer size may affect latency times.

TCP Offload Engine
Depending whether the running application supports TCP Offload Engine (TOE), this feature can be set accordingly. It offloads the TCP connections to the NIC, thereby reducing CPU utilization significantly.
1. Enable TOE at OS: netsh int tcp set global chimney=enable
2. NIC property: TCP connection offload (IPv4) or TCP connection offload (IPv6) set to enabled
**Shutting down services**

It is recommended to turn off the following services:

1. Windows Firewall
2. Base Filtering Engine

**Resources**

Search for additional details using the QLogic 57810S and 57840S controllers at [qlogic.com/go/hp](http://qlogic.com/go/hp).

**Learn more at**

[hp.com/go/proliantnics](http://hp.com/go/proliantnics)