IP OVER INFINIBAND

The ib_ipoib driver is an implementation of the IP over InfiniBand protocol as specified by RFC 4391 and 4392, issued by the IETF ipoib working group. It is a "native" implementation in the sense of setting the interface type to ARPHRD_INFINIBAND and the hardware address length to 20 (earlier proprietary implementations masqueraded to the kernel as ethernet interfaces).

Partitions and P_Keys

When the IPoIB driver is loaded, it creates one interface for each port using the P_Key at index 0. To create an interface with a different P_Key, write the desired P_Key into the main interface's /sys/class/net/<intf name>/create_child file. For example:

```
echo 0x8001 > /sys/class/net/ib0/create_child
```

This will create an interface named ib0.8001 with P_Key 0x8001. To remove a subinterface, use the "delete_child" file:

```
echo 0x8001 > /sys/class/net/ib0/delete_child
```

The P_Key for any interface is given by the "pkey" file, and the main interface for a subinterface is in "parent."

Datagram vs Connected modes

The IPoIB driver supports two modes of operation: datagram and connected. The mode is set and read through an interface's /sys/class/net/<intf name>/mode file.

In datagram mode, the IB UD (Unreliable Datagram) transport is used and so the interface MTU has is equal to the IB L2 MTU minus the IPoIB encapsulation header (4 bytes). For example, in a typical IB fabric with a 2K MTU, the IPoIB MTU will be 2048 - 4 = 2044 bytes.

In connected mode, the IB RC (Reliable Connected) transport is used. Connected mode takes advantage of the connected nature of the IB transport and allows an MTU up to the maximal IP packet size of 64K, which reduces the number of IP packets needed for handling large UDP datagrams, TCP segments, etc and increases the performance for large messages.

In connected mode, the interface's UD QP is still used for multicast and communication with peers that don't support connected mode. In this case, RX emulation of ICMP PMTU packets is used to cause the networking stack to use the smaller UD MTU for these neighbours.

Stateless offloads

If the IB HW supports IPoIB stateless offloads, IPoIB advertises TCP/IP checksum and/or Large Send (LSO) offloading capability to the network stack.

Large Receive (LRO) offloading is also implemented and may be turned

on/off using ethtool calls. Currently LRO is supported only for checksum offload capable devices.

Stateless offloads are supported only in datagram mode.

Interrupt moderation

If the underlying IB device supports CQ event moderation, one can use ethtool to set interrupt mitigation parameters and thus reduce the overhead incurred by handling interrupts. The main code path of IPoIB doesn't use events for TX completion signaling so only RX moderation is supported.

Debugging Information

By compiling the IPoIB driver with CONFIG_INFINIBAND_IPOIB_DEBUG set to 'y', tracing messages are compiled into the driver. They are turned on by setting the module parameters debug_level and mcast_debug_level to 1. These parameters can be controlled at runtime through files in /sys/module/ib_ipoib/.

CONFIG_INFINIBAND_IPOIB_DEBUG also enables files in the debugfs virtual filesystem. By mounting this filesystem, for example with

```
mount -t debugfs none /sys/kernel/debug
```

it is possible to get statistics about multicast groups from the files /sys/kernel/debug/ipoib/ib0_mcg and so on.

The performance impact of this option is negligible, so it is safe to enable this option with debug_level set to 0 for normal operation.

CONFIG_INFINIBAND_IPOIB_DEBUG_DATA enables even more debug output in the data path when data_debug_level is set to 1. However, even with the output disabled, enabling this configuration option will affect performance, because it adds tests to the fast path.

References

Transmission of IP over InfiniBand (IPoIB) (RFC 4391)
http://ietf.org/rfc/rfc4391.txt
IP over InfiniBand (IPoIB) Architecture (RFC 4392)
http://ietf.org/rfc/rfc4392.txt
IP over InfiniBand: Connected Mode (RFC 4755)
http://ietf.org/rfc/rfc4755.txt