ChatGPT

InfiniBand Technical Guide for Network Engineers

Fundamentals of InfiniBand

Overview: InfiniBand (IB) is a high-speed, low-latency interconnect standard used primarily in high-performance computing (HPC) clusters and data centers [1]. It uses a **switched fabric architecture**, meaning devices connect through InfiniBand switches in a fabric (as opposed to a shared bus). The fabric is *channel-based* and highly scalable, supporting tens of thousands of nodes in a single subnet [2] [3]. InfiniBand was designed for both **inter-node and intra-node** communication, and can connect servers to servers or servers to storage with unified protocols and high reliability. Key built-in features include **quality of service (QoS)**, failover, and lossless packet delivery via credit-based flow control.

Components of an InfiniBand fabric: multiple computing nodes connect to an InfiniBand switch fabric via Host Channel Adapters (HCAs), enabling high-throughput, low-latency communication [1] [4].

Architecture and Protocols: InfiniBand defines its own network stack and programming interface (the **verbs** API). Applications typically use InfiniBand via RDMA verbs – for example, performing **Remote Direct Memory Access (RDMA)** reads/writes or message send/receive operations. Unlike TCP/IP, InfiniBand's communication is message-oriented and can bypass the CPU for data transfer, reading/writing directly to memory buffers (zero-copy). Communications are carried over **Queue Pairs (QPs)** consisting of a send queue and receive queue on each end. The InfiniBand transport layer supports several transport types, notably: Reliable Connection (RC) for guaranteed in-order delivery (commonly used for MPI and RDMA operations), Unreliable Datagram (UD) for connectionless messaging (used for things like the Subnet Manager protocol), as well as Reliable Datagram and Unreliable Connection in specific cases. These transports give flexibility in how applications communicate (e.g. RC for heavy two-way traffic with RDMA, UD for multicast or discovery traffic).

Host Channel Adapter (HCA): An HCA is the InfiniBand interface on a compute node – analogous to a NIC in Ethernet. It connects the server's PCIe bus to the InfiniBand network and offloads IB transport functions. Each HCA executes transport-layer tasks, manages RDMA operations, and exposes the verbs API to software [4]. Modern HCAs (like NVIDIA Mellanox ConnectX series) have advanced capabilities: they handle packet sequencing, acknowledgments, and even advanced offloads like tag matching (for MPI messages) and in-network computing. HCAs typically provide multiple ports (often 2) for redundancy or extra bandwidth.

Switches and Topologies: InfiniBand switches forward packets within the IB fabric. They operate at the link layer (similar to Ethernet switches) and use local identifiers (LIDs) for addressing on the fabric. Common network topologies include **fat-tree (Clos)** networks for high bisection bandwidth – many HPC clusters deploy InfiniBand in multi-tier fat-tree topologies to ensure low-diameter, high-throughput connectivity between any two nodes. Other topologies like mesh or torus can be used but are less common with InfiniBand; the fat-tree/Clos is prevalent for its scalability and support for full bandwidth communication. InfiniBand fabrics require a **Subnet Manager (SM)** process which runs on one of the switches or a designated server – the SM assigns addresses (LIDs) to each port and establishes routing paths in the fabric. Only one SM is active per fabric (with an optional standby for failover) to control the topology [5]

[6]. Once the SM configures the network, switches forward traffic based on LID routing tables. InfiniBand routing is credit-based and lossless – a sender only sends when the receiver (or switch) has buffer credits, preventing congestion drops.

Link Speeds and Data Rates: InfiniBand has generational link speeds often referred to by names like SDR, DDR, QDR, etc. Each link is typically a 4-lane (4×) connection (though 1× and 8× links exist in some cases). The table below summarizes the major InfiniBand speed generations and their throughput and latency characteristics:

InfiniBand Generation	Signaling Rate per lane	4× Link Throughput (raw)	Effective Data Throughput (4×)	Typical One-Way Latency
SDR (Single Data Rate)	2.5 Gb/s	10 Gb/s	~8 Gb/s [7] [8]	~5 µs [9]
DDR (Double Data Rate)	5 Gb/s	20 Gb/s	~16 Gb/s [7] [8]	~2.5 µs [9]
QDR (Quad Data Rate)	10 Gb/s	40 Gb/s	~32 Gb/s [7] [8]	~1.3 µs [9]
FDR (Fourteen Data Rate)	14.0625 Gb/s	~56 Gb/s	~54.5 Gb/s [8]	~0.7 µs [9]
EDR (Enhanced Data Rate)	25 Gb/s	100 Gb/s	~97 Gb/s [8]	~0.5 µs [9]
HDR (High Data Rate)	50 Gb/s	200 Gb/s	~194 Gb/s [8]	< 0.5 µs [9]
NDR (Next Data Rate)	100 Gb/s	400 Gb/s	~388 Gb/s (estimated)	~0.3–0.4 µs (estimated)
XDR (eXtreme Data Rate) *Upcoming*	200 Gb/s (PAM4)	800 Gb/s	~776 Gb/s (estimated)	~0.3 µs (expected)

Table: **InfiniBand Speed Roadmap –** Each new generation doubles bandwidth. Note the effective throughput is slightly lower than raw due to encoding overhead (SDR/DDR/QDR used 8b/10b encoding giving ~80% efficiency, while FDR and beyond use 64/66 encoding ≈97% efficiency) [7]. InfiniBand latency has decreased with faster signaling and improved hardware, now well below a microsecond at HDR/NDR speeds. For comparison, even 10Gb Ethernet is around 7–8 µs latency, an order of magnitude higher than modern InfiniBand [9].

Cables and Transceivers: InfiniBand typically uses high-density connectors (like QSFP family). For short runs (up to a few meters), **passive copper** cables (DACs) are common – e.g. 4× links over copper up to ~3–5 m at HDR speeds [10]. Longer connections use **active optical cables (AOCs)** or transceivers with fiber, supporting tens to hundreds of meters (Omnipath up to 100m+ for 100G, HDR up to ~100m with optics) [11]. There are also **active copper** cables with signal conditioning for intermediate lengths. For extremely long distances, IB can be extended via devices like NVIDIA Mellanox **MetroX**, which extend IB over fiber up to 80 km (introducing ~5 μs extra latency per km) [12] . Notably, any two InfiniBand-capable systems can be directly connected with a single cable (point-to-point) if needed – no Ethernet-style crossover is required, as the link is auto-sensing [13] .

Key Hardware Components: Aside from HCAs and switches, InfiniBand supports routers and gateways: - *InfiniBand Router:* Used to connect separate IB subnets, forwarding traffic between them (this is rarely used in small clusters, but can scale out very large networks by partitioning into subnets). - *Target Channel Adapter (TCA):* Similar concept to HCA but for storage or I/O devices. It allows a storage system or specialized device to connect to the IB fabric directly [14] . - *Gateway/Bridge:* Devices that translate InfiniBand to other protocols (for example, a gateway might connect an IB fabric to an Ethernet network or Fibre Channel SAN). For instance, IP over IB (IPoIB) is a protocol for carrying IP packets on InfiniBand, often used to integrate IB networks with existing IP-based infrastructure.

In summary, InfiniBand's fundamentals can be thought of as a complete network stack designed for speed: purpose-built adapters (HCAs), a lossless switched fabric, and an efficient RDMA-capable transport protocol. These enable the ultra-high throughput and low latency that make IB popular in performance-critical environments [1] .

Practical InfiniBand Commands and Troubleshooting

Managing and troubleshooting an InfiniBand network involves a set of specialized command-line tools (primarily from the Mellanox/NVIDIA OFED software stack). Below are common IB utilities and their uses:

- `ibstat` **:** Displays the status of InfiniBand HCAs and ports on a server. This shows information like the HCA model, firmware version, and each port's state, link width, and speed. For example, running `ibstat` will indicate if a port is **ACTIVE** (linked up) or down, and at what speed (e.g. "LinkUp - 100Gb/s (4X EDR)") [15] . Use this to verify that the host sees the IB adapter and that the link is up at the expected rate.

- `ibping` **:** Tests basic connectivity between two InfiniBand nodes, analogous to an IP ping. It sends a packet from one node's HCA to a target destination (specified by GUID, LID, or GID address). For example, `ibping <destination>` will send ping packets to the target and report if responses are received [16] [17] . This helps confirm that two servers can reach each other over the IB fabric (end-to-end check at the IB level).

- `ibdiagnet` **:** A comprehensive diagnostic tool that scans the entire InfiniBand fabric for errors and configuration issues [18] . Running `ibdiagnet` (with appropriate options) will query all fabric components via the subnet manager, gathering data on node links, error counters, firmware versions, topology consistency, etc. It can report problems like mis-cabled links, missing route entries, and performance issues. This is often the go-to for a *health check* of the IB network – for example, after installation or if experiencing issues, an admin can run `ibdiagnet` to identify any ports with lots of errors or switches with misconfigured settings.

- `perfquery` **:** Retrieves performance counters from InfiniBand ports (either on an HCA or switch). This utility queries statistics like packet transmit/receive counts, symbol errors, link error recovery counts, etc., which are vital for troubleshooting physical link problems. For instance, `perfquery -C <ca_name> -P <port>` can show if a port is seeing symbol errors or retranmissions. High error counters may indicate a bad cable or optical module. `perfquery` also has options to reset counters (so you can zero them out and then later check if they are increasing, to isolate an intermittent issue) [19] [20]. Use this to pinpoint whether packet loss or errors are occurring at the link layer.

- **Other useful tools:**

- `ibnetdiscover`**:** Generates a map of the InfiniBand fabric topology. It lists all the nodes (HCAs, switches) and how they connect (which port to which port) [21] [22]. This is useful to visualize or verify the cabling of the cluster.
- `ibroute` **/** `ibtracert`**:** Traces the route between two end nodes in the fabric. `ibtracert <LID1> <LID2>` will step through each hop from source to destination, useful for understanding path routing or finding at which hop a failure occurs [23] [24].
- **Subnet Manager commands (OpenSM):** If using OpenSM (the subnet manager), commands like `smquery` or logs from the SM can be examined when diagnosing fabric initialization issues (e.g., if a node isn't getting a LID or if there are partition (P_Key) configuration problems).

Basic Troubleshooting Workflow: Troubleshooting InfiniBand is often tackled layer-by-layer [25] [26]: 1. **Physical Layer:** Start by checking physical connections. Ensure all cables are firmly seated; reseat or replace any suspect cables. Verify that port LEDs on HCAs and switches show link. Because IB can use passive copper for short links, cable quality and length limitations should be considered (e.g. a long copper cable might not sustain HDR speeds – swapping to fiber might resolve errors). If you have access to InfiniBand cable testers or loopback connectors, use them to test ports exhibiting issues [27] [28].

1. **Link Layer:** Use `ibstat` to confirm the HCA ports are active and at the expected speed. If a port is down or at a lower width/speed than expected (e.g., a 4X HDR link only coming up as 1X or at SDR rates), that could indicate a cabling or compatibility issue. Check for errors: run `ibdiagnet` or `perfquery` to see if the port or its switch port is logging symbol errors, FEC errors, etc. **Clear counters** (using `perfquery` or `ibclearerrors`) and see if they increment rapidly, which would confirm a physical layer problem. Often, high error counts on a port suggest a bad cable or optical transceiver. Also verify firmware and driver versions – outdated firmware can sometimes cause link instability [29].

2. **Network Layer:** If links are up but communication fails between nodes, ensure the Subnet Manager is running and has configured all endpoints. Use `ibnetdiscover` to verify the network topology and that all nodes appear in the fabric map [21]. If a node is missing, it may not have an active link or might be configured for a different subnet (check the subnet prefix/GID). Check the **LID assignments** (each node should have a unique LID from the SM). If you suspect routing issues, `ibtracert` between two nodes can show if the route stops at a certain switch. All switches should be running in the same subnet (one active SM). Misconfigurations in partitioning (P_Keys) can also prevent two nodes from communicating; ensure they share any required P_Key for communication (if partitions are used).

3. **Transport/Application Layer:** Use `ibping` to test basic connectivity between two hosts' HCAs [16]. If `ibping` works but higher-level MPI or IPoIB traffic does not, the issue might be with those layers (e.g., IPoIB configuration or MPI setup). For MPI applications, ensure that the MPI is built with

InfiniBand support (e.g., via OFED or using libfabric/UCX as a transport). At this stage, also consider performance troubleshooting: use tools like `ib_send_bw` or `ib_send_lat` (part of perftest utilities) to measure point-to-point bandwidth and latency between nodes. These can help determine if you're getting expected performance or if there's an unseen issue (for example, an application running slow because one IB link is degraded and causing a bottleneck).

Common Issues: Some frequent InfiniBand issues include mis-cabling (ports wired incorrectly leading to a non-optimized topology or loops), missing Subnet Manager (no SM means the fabric won't initialize – no LIDs assigned), firmware mismatches or outdated versions, and buffer credit starvation (if an older switch with limited buffering gets saturated, although IB is lossless, it can throttle traffic heavily). In large fabrics, **credit loops or congestion** can occur – features like Congestion Control or adaptive routing (if supported by switch/HCA) might need tuning in such cases. Always ensure all nodes and switches are running supported firmware/driver combinations, as IB is sensitive to consistency across the fabric.

In summary, InfiniBand troubleshooting starts with verifying the physical link health, then fabric configuration, and finally performance tuning. The toolset (`ibstat`, `ibping`, `ibdiagnet`, `perfquery`, etc.) is your toolbox to observe each layer of the IB stack and pinpoint problems quickly.

InfiniBand in High-Performance Computing (HPC)

InfiniBand has a long history in supercomputing and HPC clusters. It became popular due to its ability to deliver **extremely low latency and high throughput**, which are critical for tightly-coupled parallel applications. In an HPC cluster, dozens to tens of thousands of nodes may need to exchange messages for algorithms like numerical simulations, weather modeling, or scientific computing – IB's design minimizes the communication overhead in these scenarios.

Use in Top Supercomputers: InfiniBand is widely used in the TOP500 list of supercomputers. As of late 2024, over half of the Top500 systems use InfiniBand interconnect, including some of the largest AI supercomputers [30]. In fact, IB or its RDMA-over-Ethernet cousin (RoCE) was reported to power 365 out of the top 500 supercomputers (73%), with pure InfiniBand in 254 systems (over 50% of the list) [30]. This trend underscores that for many HPC installations, InfiniBand is the preferred network to achieve the required performance. Clusters like **Leonardo** in Europe (which uses Quad-rail HDR100 InfiniBand) demonstrate how multiple IB links per node are employed to scale bandwidth [31]. InfiniBand's dominance in HPC grew through the 2010s and continues with HDR (200 Gbps) and NDR (400 Gbps) technologies being adopted in new installations.

MPI and Communication Patterns: The Message Passing Interface (MPI) is the standard library for distributed computing in HPC. InfiniBand provides an excellent transport for MPI because it supports RDMA and one-sided operations, which MPI can use to implement efficient send/receive and publish/subscribe semantics. For example, when one MPI rank sends a message to another, the underlying IB HCA can directly place that data into the target node's memory (RDMA write) or generate a remote completion event, all with minimal CPU involvement. Common communication patterns in HPC include **collective operations** (like broadcast, reduce, all-reduce, gather/scatter, etc.) and **point-to-point** messaging. InfiniBand HCAs and switches have evolved to accelerate these patterns. One notable feature is **NVIDIA SHARP (Scalable Hierarchical Aggregation and Reduction Protocol)**, originally implemented in Mellanox InfiniBand switches. SHARP offloads collective reduction operations (like MPI All-Reduce for summing arrays) into the network fabric, so the switch can combine data from multiple nodes and send out a single reduced result, greatly speeding up operations like distributed averaging of gradients or values [32]. This in-network computing approach reduces the number of messages and latency for collectives, boosting scaling efficiency for large node counts.

Performance Considerations: HPC workloads often involve fine-grained communication, where latency (measured in microseconds or even nanoseconds) is as important as raw bandwidth. InfiniBand's ultra-low latency ensures that operations like synchronization (barriers, locks) and small message exchanges (tens of bytes) incur minimal delay. Additionally, IB's high bandwidth allows massive data exchanges (like exchanging boundary data in a domain decomposition, or doing large-scale FFT transpositions) to occur quickly. Modern InfiniBand (HDR/NDR) can deliver on the order of 100–200 Gb/s *per link* of effective throughput, and nodes can have multiple links. For instance, an HPC node might use *dual-rail* InfiniBand (two HCAs or two ports bonded) to double the available bandwidth for very communication-heavy codes.

Another advantage is InfiniBand's low CPU overhead. By using RDMA and offloading transport management to the HCA, the CPU is freed from handling interrupts for every packet. This is crucial in HPC where CPU cycles are precious for computations. With Ethernet, even with optimizations, the kernel and CPU typically handle more of the networking stack, whereas IB offloads much of that work to the NIC hardware.

Deployment in HPC Clusters: Typically, HPC clusters using InfiniBand will have a set of **leaf switches** connecting groups of nodes (for example, a 40-port switch might connect 36 servers and uplink to spine switches). These then connect to **spine switches** (higher-radix switches) to interconnect all leaf switches in a fat-tree topology. The goal is often to provide **full bisection bandwidth**, meaning any node can communicate with any other at full wire speed concurrently – something InfiniBand is well-suited for, though it requires careful network architecture and enough switch bandwidth. Many clusters choose InfiniBand not just for performance, but also for its deterministic latency under load (since it's lossless and has features like congestion control to manage hot spots).

In summary, InfiniBand's role in HPC is to enable fast message passing and scaling. It has a proven track record here: from small 8-node GPU clusters to leadership-class supercomputers with thousands of nodes, IB provides the communication backbone. HPC engineers value the ability to run communication-intensive codes with minimal penalty, which InfiniBand delivers through RDMA, low latency, and robust hardware offloads.

InfiniBand in Modern Machine Learning Clusters

In recent years, machine learning (ML) and AI training workloads have pushed the limits of network infrastructure, and InfiniBand has become a key player in many AI supercomputers and training clusters. The requirements of ML workloads (especially deep learning training) align well with InfiniBand's strengths: high bandwidth, low latency, and efficient scaling for large distributed jobs.

Data-Parallel Training: In large-scale training of neural networks (for example, training a transformer model on dozens or hundreds of GPUs), a common pattern is *data-parallelism*. Each GPU processes a portion of the input data (a mini-batch), computes gradients, and then all GPUs need to exchange and aggregate these gradients before the next training iteration. This exchange is typically an **All-Reduce** operation (summing gradients from all workers and distributing the result back to all). InfiniBand networks, often through libraries like **NCCL (NVIDIA Collective Communications Library)** or MPI, perform these All-

Reduce operations efficiently. Using RDMA and optimized collectives, clusters with InfiniBand achieve much faster gradient aggregation than they would over traditional Ethernet. This directly translates to faster model training times.

GPUDirect RDMA: NVIDIA GPUs coupled with InfiniBand HCAs support a feature called GPUDirect RDMA. This allows the HCA to directly read from or write to GPU memory over the PCIe bus, bypassing the CPU. In an ML context, when GPU 0 on Node A wants to send a tensor to GPU 0 on Node B, GPUDirect RDMA allows the data to flow directly from the source GPU's memory, through the InfiniBand HCA, across the network, and into the destination GPU's memory – without intermediate copies to host memory. This dramatically lowers latency and CPU overhead for GPU-to-GPU communication. It essentially lets InfiniBand serve as a **cluster-wide GPU interconnect**, which is exactly what large-scale ML training needs. All major distributed ML frameworks (TensorFlow, PyTorch, etc.) leverage this capability under the hood (via NCCL or MPI) when available.

Collective Offloads and In-Network Computing: As mentioned in the HPC section, technologies like NVIDIA SHARP can benefit ML as well. Large AI clusters (e.g. NVIDIA DGX SuperPODs) enable SHARP on their InfiniBand switches to offload the All-Reduce for gradients. Instead of each GPU sending gradients to every other GPU, each GPU sends to its local switch, which performs the addition at the switch level and then forwards a reduced result. This hierarchy speeds up communication in *N*-node training by reducing collective operation complexity from many-to-many into a tree of reductions. The result is better scaling efficiency – adding more GPUs yields nearly linear speedup until network bandwidth becomes the limiter.

Latency and Synchronized Training: While training is mostly about throughput (bandwidth for large tensor all-reduces), low latency is still valuable in scenarios like reinforcement learning or asynchronous parameter updates. InfiniBand's low latency ensures that synchronization barriers and short control messages (for coordination between processes) incur minimal delay, keeping GPUs fed with data.

Use in AI Supercomputers: Many of the world's AI-focused supercomputers or cloud ML clusters use InfiniBand. For example, Meta's AI Research SuperCluster (RSC) and Microsoft's AI infrastructure (for OpenAI models) have leveraged InfiniBand networks to tie together hundreds to thousands of GPUs. The **convergence of HPC and AI** is evident – the InfiniBand Trade Association noted that RDMA networks (InfiniBand and RoCE) are increasingly adopted for cutting-edge AI workloads due to the bandwidth and latency demands [33]. In the November 2024 Top500 list, multiple top AI systems were listed using either InfiniBand or RoCE, underlining that Ethernet alone often cannot meet the strict performance required without RDMA [30] [33].

Example – NVIDIA DGX Systems: NVIDIA's DGX nodes (each containing multiple GPUs) typically include InfiniBand adapters for clustering. A DGX A100, for instance, has 8 GPUs connected by NVSwitch (for intra-node communication) and often dual-rail Mellanox HDR InfiniBand ports for inter-node. When building a DGX SuperPOD (a cluster of many DGX nodes), these InfiniBand ports link all the nodes in a Clos network. The result is a seamless GPU farm where any GPU can efficiently talk to any other GPU, either via NVSwitch (if within the same node) or via InfiniBand (across nodes). This design was used to set records in MLPerf benchmarks – demonstrating near-linear scaling of training throughput from 8 GPUs to 16, 32, or more, largely thanks to InfiniBand's communication efficiency.

In summary, InfiniBand has become as vital to large-scale ML as it has been to traditional HPC. It addresses the communication-heavy nature of distributed training by providing the needed bandwidth (hundreds of

Gbps per node), low latency, and advanced features like GPUDirect RDMA and switch-based collectives. For any organization building an AI cluster with dozens+ of accelerators, InfiniBand is often chosen to ensure the network isn't the bottleneck in training jobs.

NVIDIA-Specific Implementations and NVLink vs InfiniBand

Since acquiring Mellanox in 2020, NVIDIA has been heavily investing in InfiniBand technology and related networking. NVIDIA's networking portfolio includes the **ConnectX** series of HCAs/NICs and **Quantum** series of InfiniBand switches, as well as **BlueField DPU** (which combines a ConnectX NIC with an embedded CPU for offloads). These provide some NVIDIA-specific enhancements:

- **ConnectX Architecture:** NVIDIA ConnectX HCAs (e.g., ConnectX-6, ConnectX-7) support both InfiniBand and Ethernet (including RoCE) on the same hardware. They offer advanced offloads like NVMEoF (for storage over IB), GPUDirect, and in-network computing. For example, the ConnectX-7 is an NDR 400Gb/s InfiniBand adapter [34] that not only pushes 400 Gbps throughput with low latency, but also includes engines for collective offload (SHARP) and enhanced tag matching for MPI. The ConnectX line and upcoming **NVIDIA Spectrum-4** switches (for Ethernet) and **Quantum-2/ Quantum-3** switches (for InfiniBand) indicate NVIDIA's strategy to lead in both Ethernet and InfiniBand high-performance networking.

- **Mellanox Switches (Quantum series):** The NVIDIA Quantum-2 InfiniBand switches (e.g., QM9700) support up to 64 ports of NDR (400G) in a 1U chassis [35], which is immense density. These switches also incorporate features like adaptive routing (to dynamically route around hot spots in the network) and the SHARP technology for in-network reductions. They form the building blocks of large IB fabrics used in supercomputers and AI clusters.

- **BlueField DPUs:** NVIDIA's BlueField-2 and BlueField-3 DPUs embed an InfiniBand/Ethernet NIC and ARM CPU cores. In an InfiniBand context, a BlueField DPU can act as a **smart HCA**, offloading even more work from the host. For instance, it could run the Subnet Manager or handle data-plane tasks like virtualization, storage acceleration, or security (isolation via IB partitions), all without burdening the main server CPU. This is more of an architectural option for cloud or specialized HPC deployments where isolating the network stack on a DPU is beneficial.

Now, a crucial point of confusion to clarify is **NVLink vs InfiniBand** – both are associated with NVIDIA:

What is NVLink? NVLink is NVIDIA's high-speed interconnect for GPUs. It is a *chip-to-chip* link primarily used for connecting GPUs to each other (and sometimes GPUs to CPUs in certain systems). NVLink operates at the board or chassis level, not as a fabric across a data center. It provides extremely high bandwidth between GPUs – for example, NVLink 4 (used in Hopper H100 GPUs) provides on the order of 900 GB/s *per GPU* in a fully connected 8-GPU system via NVSwitch [36]. NVLink is a **coherence-capable** link (GPUs can access each other's memory over NVLink almost as if it's local memory), which is different from InfiniBand's RDMA (explicit memory operations).

NVLink vs InfiniBand – Do they overlap? NVLink does **not use InfiniBand** technology; it is a separate interface and protocol entirely. NVLink is used for *intra-node* or *intra-system* communication, whereas InfiniBand is used for **inter-node** communication. In simpler terms, inside a single server with multiple

GPUs, NVLink (and NVSwitch) create a high-speed network among GPUs; but to communicate from one server to another, you use InfiniBand (or Ethernet). NVLink cannot directly extend between two separate servers in a standard setup.

Why NVLink for intra-node and InfiniBand for inter-node? The choice comes down to distance and specialization: - NVLink is designed for extremely high bandwidth over very short distances (within a chassis or between GPUs on the same motherboard). It uses parallel chip-to-chip links and is tightly integrated into GPU architecture. This makes it fantastic for GPU-to-GPU communication *within* a machine – offering much higher bandwidth than even PCIe. For example, NVLink 5 in the latest GPUs supports 1.8 TB/s per GPU (bidirectional) [37] [38], which dwarfs even 400 Gb/s (50 GB/s) InfiniBand on a per-node basis. - However, NVLink is not a network protocol for long cables or switches in the traditional sense (at least historically). It's not designed to go through optical transceivers or across hundreds of meters – InfiniBand (and Ethernet) excel there. InfiniBand can maintain signal integrity and manage flow control over longer distances and through multiple switch hops, which NVLink cannot do without significant changes.

So, inside a single AI server (say 8 GPUs), NVIDIA uses NVLink/NVSwitch to make those 8 GPUs operate almost like one giant GPU with fast memory sharing. But to connect multiple such servers, InfiniBand is used because it can handle the networking across racks with the required reach and switching.

NVLink and NVSwitch advancements: It's worth noting that NVIDIA is *extending* NVLink beyond the node in new ways. The introduction of the **NVLink Switch (NVSwitch external to servers)** means that multi-server clusters can have all-to-all GPU communication **at NVLink speeds** within a rack or across racks [39]. For example, the NVIDIA NVLink Switch (part of NVIDIA's HGX/GB series systems) allows up to 256 or even 576 GPUs to be connected in a non-blocking network at 1.8 TB/s per port [40] [41]. This effectively creates an even larger "virtual super-GPU" across nodes. However, these NVLink switches are purpose-built for GPU clustering and are not general-purpose network switches – they only interconnect GPUs (and through NVLink bridges, possibly GPUs in different servers like the new GB300 NVL systems). **InfiniBand remains the more general solution for inter-node comms** and is widely used outside of purely NVIDIA GPU-to-GPU links. NVLink networks also tend to be confined to within a single AI cluster or superpod, whereas InfiniBand can connect heterogeneous servers, storage, and GPU nodes all in one fabric using standard protocols.

In summary, NVLink is an *intra*-server (or now, rack-scale) GPU interconnect offering unparalleled bandwidth for GPU memory sharing, and it operates independently of InfiniBand. InfiniBand is the *inter*-server network that connects entire systems together. In NVIDIA's portfolio, **NVLink is for inside the box (or specialized multi-box GPU networks) and InfiniBand is for between the boxes**. They complement each other: NVLink for maximum speed between GPUs, InfiniBand for scaling out across many servers. Notably, both have their place in modern AI systems – high-end deployments use NVLink/NVSwitch for nodes and InfiniBand between nodes to minimize bottlenecks at every level.

(And to answer a common question: NVLink does not "run over" InfiniBand – it's separate. NVLink doesn't carry IP or IB traffic. Conversely, GPUs in different servers typically must use InfiniBand or Ethernet to communicate, unless one has access to the specialized NVLink Switch infrastructure which is still emerging.)

Comparison with Other Interconnects

Finally, it's useful to compare InfiniBand with other networking and interconnect technologies that a network engineer might encounter: namely Ethernet (including RoCE for RDMA over Ethernet), NVIDIA's NVLink, and Intel's Omni-Path. Each has its niche, with some overlap in use cases.

InfiniBand vs. Ethernet (and RoCE)

Throughput and Latency: Traditional Ethernet has historically lagged a bit behind InfiniBand in raw speed and latency, though the gap has narrowed. For example, 100 Gb Ethernet became common a couple of years after 100 Gb InfiniBand (EDR) was out. Today, high-end Ethernet is available at 200 Gb/s and 400 Gb/s, similar to HDR/NDR IB speeds. However, out-of-the-box Ethernet latency (even at 100 Gb+) is higher – often a few microseconds or more one-way – whereas InfiniBand can be sub-microsecond [9]. Ethernet also usually involves the kernel network stack, adding software latency, unless specialized techniques are used.

RDMA and RoCE: To get RDMA capabilities on Ethernet, the solution is RoCE (RDMA over Converged Ethernet) or iWARP. RoCE uses the InfiniBand verbs programming interface but encapsulates the packets in Ethernet frames (either requiring a lossless network for RoCE v1 or using RoCE v2 with UDP encapsulation plus PFC/ECN to handle congestion). Modern Mellanox/NVIDIA NICs support RoCE natively, so the same adapter can do either InfiniBand or Ethernet+RoCE. The **advantage of RoCE** is that it brings RDMA's low CPU overhead and zero-copy transfers to Ethernet networks – useful for distributed storage (e.g., NVMe-oF, iSER) and some HPC/AI applications on Ethernet fabrics [33]. The **challenge** is that Ethernet was designed as a best-effort, packet-drop network. To make RoCE reliable and low-latency, administrators must configure quality of service: enabling PFC (Priority Flow Control) to prevent drops, or use ECN (Explicit Congestion Notification) to manage congestion instead of packet loss. Misconfiguration can lead to issues like deadlock (with PFC) or high latencies (if congestion isn't managed).

Ecosystem and Ubiquity: Ethernet is everywhere – it's the standard for general data center traffic. This means Ethernet hardware is mass-produced and often cheaper (port for port) than InfiniBand. Network engineers and tools are more commonly oriented around Ethernet and TCP/IP. For many applications that are not latency sensitive or don't need RDMA, Ethernet is perfectly fine (web services, cloud VMs, storage traffic, etc.). InfiniBand tends to appear where maximum performance is needed and applications can justify a specialized network (HPC, AI clusters, high-frequency trading, etc.). There is also the consideration of skillset: many enterprise IT teams are more familiar with Ethernet networking (routing, switching, VLANs, etc.) than InfiniBand's fabric (Subnets, LIDs, etc.).

When to use which: If ultra-low latency, high message rate, and efficient RDMA are required (HPC MPI traffic, AI training, distributed GPU memory sharing), InfiniBand is often the better choice. It's basically built for that purpose and typically "just works" for HPC provided the fabric is set up (because credit-based flow control and such are automatic). If the use case is general-purpose data center traffic, or where RDMA is nice-to-have but not critical, Ethernet (potentially with RoCE) might be chosen due to its versatility and cost. In practice, we also see **hybrid approaches**: for example, some clusters use Ethernet for storage or service traffic and InfiniBand for MPI traffic (keep two networks). Also, emerging 800G Ethernet and beyond will continue to push performance, but IB is also moving to 800G (XDR) and retains the latency/offload edge as a purpose-built HPC network.

InfiniBand vs. NVLink

This was covered in detail in the NVIDIA section. To recap briefly: NVLink is a **GPU interconnect** within a node (or multi-node system with specialized NVLink switches). It's not a general networking tech for all devices. NVLink provides extreme bandwidth (hundreds of GB/s) but only works for NVIDIA GPUs (and some CPU-GPU combos like NVLink between IBM POWER CPUs and NVIDIA GPUs in the past). It also maintains cache coherence and memory sharing at a hardware level between GPUs. InfiniBand, on the other hand, connects independent systems and can work across CPUs, GPUs, storage, etc., but at lower bandwidth and with message-based (not fully coherent) communication.

Pros/Cons: NVLink's pro is sheer speed and tight coupling (GPUs communicate as if in one system). Its con is limited scope (distance and device support). InfiniBand's pro is general applicability and long-distance capability (you can have a kilometer-long InfiniBand cable with optical transceivers; you cannot do that with NVLink). InfiniBand's con might be that it cannot match NVLink's bandwidth or the coherence feature (each CPU/GPU is still distinct, sharing data via RDMA rather than truly unified memory). In designing a system, you typically **use both**: NVLink inside nodes for multi-GPU servers, InfiniBand between those servers. If one tried to use only NVLink everywhere, you would be restricted to basically one giant specialized machine (which NVIDIA is in fact exploring with their NVLink Switch systems, but those are exotic and dedicated to GPU clusters). For general multi-purpose clusters, InfiniBand is a more flexible choice.

InfiniBand vs. Omni-Path

Omni-Path was Intel's alternative to InfiniBand. Omni-Path Architecture (OPA) grew out of Intel's acquisition of QLogic's InfiniBand technology and was first released as **OPA 100** (100 Gbps) around 2015–2016. It competed with EDR (100 Gbps) InfiniBand at the time. Omni-Path had similar goals: high throughput, low latency, and features like end-to-end reliability and QoS for HPC. It used a lot of the same concepts (Verbs API support, queue pairs, etc.) so that MPI libraries could run over either IB or OPA.

Key differences: Omni-Path's first gen used a **48-port switch ASIC** (while InfiniBand EDR typically had 36-port switches). This allowed building flatter topologies with fewer switch tiers for a given cluster size. OPA also did not rely on an external subnet manager in the same way – it had a somewhat different management approach (though conceptually, it still had to configure the fabric). Latency and performance of Omni-Path were comparable to same-generation InfiniBand; some reports showed OPA slightly higher latency at the time, others showed similar. One claimed advantage of Omni-Path was congestion management – Intel designed features to handle hot spot traffic and to avoid head-of-line blocking in large fabrics (a notorious issue in large networks). However, InfiniBand also evolved congestion control, so it's a moving target.

Status and usage: Omni-Path saw adoption in a number of supercomputers (especially where there were Intel partnerships). However, after Intel cancelled Omni-Path development around 2019 (choosing to focus elsewhere), a company called **Cornelis Networks** took over OPA technology. Cornelis has been continuing with Omni-Path Gen2 (sometimes called Omni-Path Express), aiming for 200 Gbps and beyond. As of 2024, Cornelis has announced deployments of Omni-Path in some new HPC systems and claims competitive price/performance vs InfiniBand [42]. Still, Omni-Path is a niche player now compared to InfiniBand.

Pros/Cons: Historically, Omni-Path's pro was potentially lower cost at scale (Intel could package it with their CPUs/servers) and integration in Intel ecosystems. It also had good performance, and because it was less common, some HPC centers found less contention in supply chains. The cons included being locked to a single vendor (at the time, Intel, now Cornelis) and a smaller ecosystem for support. InfiniBand, backed by Mellanox (NVIDIA), had many generations of proven hardware, a larger user base, and faster introduction of new speeds (200G, 400G came on IB first). Today, choosing Omni-Path would likely only make sense if there's a specific need or existing investment – most new installations lean towards InfiniBand or high-speed Ethernet.

InfiniBand vs. Fibre Channel (and others)

While not explicitly asked, a note: InfiniBand can also be contrasted with **Fibre Channel (FC)**, which is used for storage area networks. FC has very low latency and high reliability too, but it's strictly for storage (block storage protocols). InfiniBand with NVMe-oF can actually serve similar purposes with even higher performance, so IB sometimes displaces proprietary FC in high-end storage connectivity (or Ethernet with NVMe/TCP or RoCE does, depending on environment). There are also other proprietary interconnects like Cray's **Aries** or **Slingshot** used in certain Cray supercomputers, and newer ones like **CXL** (Compute Express Link) emerging for within-rack memory pooling (which is more of a cache-coherent fabric over PCIe). Each of these targets specific niches.

In summary: InfiniBand's main competition in the cluster networking space is advanced Ethernet (with RoCE) and to some extent the remnants of Omni-Path. Ethernet/RoCE offers more integration with standard networks and is often "good enough" for many uses, but InfiniBand still leads in maximum performance for latency and collective throughput in HPC/AI contexts. NVLink is not a general cluster network, but rather a complement for GPU-heavy nodes. Omni-Path shares IB's goals but is currently less prevalent. As an engineer, the choice boils down to the use case: - Use **InfiniBand** when you need the absolute lowest latency and highest message rate (HPC MPI, large-scale AI training) and are willing to manage a separate network fabric for it. - Use **Ethernet/RoCE** when you want RDMA capabilities but also need integration with standard Ethernet networks or have mixed workloads (and ensure to configure lossless behavior for RoCE traffic). - Use **NVLink** (internally) when building multi-GPU systems – NVLink isn't something you "choose" at cluster level; it comes as part of NVIDIA's GPU systems to accelerate intra-node comms. - **Omni-Path** might be considered if you have existing Omni-Path infrastructure or specific vendor support, but otherwise, InfiniBand currently has more momentum and roadmap (with NDR 400G and beyond to XDR 800G already in view [43]).

Each interconnect has its pros and cons, but InfiniBand remains a cornerstone of both HPC and cutting-edge AI networks due to its balance of throughput, latency, and advanced networking features tailored for distributed computing. By understanding these differences, a network engineer can select the right tool for the job – or often, deploy a combination (e.g. IB for MPI/AI traffic and Ethernet for everything else) to meet all requirements.

Sources: The information above is drawn from InfiniBand official documentation and community resources, including performance specifications [8] [9], industry reports on Top500 supercomputers [30], and vendor insights from NVIDIA/Mellanox and others on technologies like NVLink [39] and RoCE [33]. These references reflect the state-of-the-art as of 2024–2025 in high-speed interconnects.

---

1 InfiniBand - Wikipedia
https://en.wikipedia.org/wiki/InfiniBand

2 3 What is InfiniBand?
https://www.techtarget.com/searchstorage/definition/InfiniBand

4 5 6 7 8 9 10 11 12 13 14 InfiniBand - ArchWiki
https://wiki.archlinux.org/title/InfiniBand

15 16 17 18 21 22 23 24 25 26 27 28 29 Troubleshooting InfiniBand Networks: A Detailed Guide - DEV Community
https://dev.to/mbayoun95/troubleshooting-infiniband-networks-a-detailed-guide-1dk6

19 perfquery(8): query InfiniBand port counters - Linux man page
https://linux.die.net/man/8/perfquery

20 InfiniBand Commands - NVIDIA Docs
https://docs.nvidia.com/networking/display/ufmenterpriseapplianceswv192/InfiniBand+Commands

30 33 43 InfiniBand and RoCE Advances Further in the TOP500 November 2024 List - InfiniBand Trade Association
https://www.infinibandta.org/infiniband-and-roce-advances-further-in-the-top500-november-2024-list/

31 TOP500 List - November 2023
https://www.top500.org/lists/top500/list/2023/11/

32 36 40 How Nvidia Blackwell Systems Attack 1 Trillion Parameter AI Models
https://www.nextplatform.com/2024/03/19/how-nvidia-blackwell-systems-attack-1-trillion-parameter-ai-models/

34 NVIDIA ConnectX-7 NDR 400G InfiniBand Adapter Card | Dell USA
https://www.dell.com/en-us/shop/nvidia-connectx-7-ndr-400g-infiniband-adapter-card/apd/ac478006/computer-chassis-components

35 [PDF] QM9700 and QM9790 1U NDR 400Gbps InfiniBand Switch Systems ...
https://docs.nvidia.com/networking/display/qm9700-and-qm9790-1u-ndr-400gbps-infiniband-switch-systems-user-manual.pdf

37 38 39 41 NVLink & NVSwitch: Fastest HPC Data Center Platform | NVIDIA
https://www.nvidia.com/en-us/data-center/nvlink/

42 Cornelis Networks Omni-Path Delivers 53% Better Price ...
https://www.cornelisnetworks.com/stories/cornelis-networks-omni-path-delivers-53-better-price-performance-than-nvidia-infiniband-hdr-when-running-seissol-on-amd-epyc-processors