

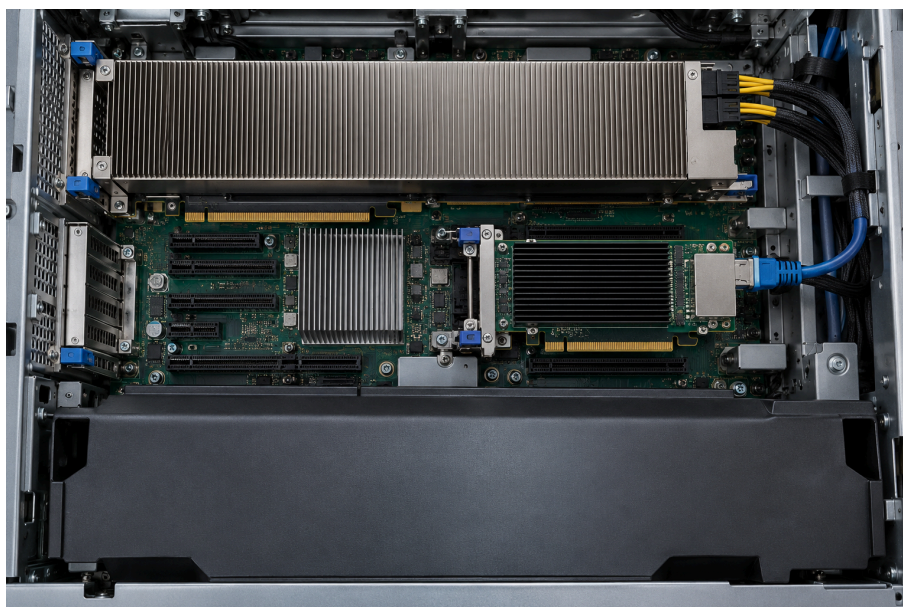


GPUDirect RDMA Validation: Topology and Memory Paths

Plan and build private AI compute with GPU Smith. We connect workload requirements with GPU hardware, networking, deployment and operating decisions so your infrastructure fits the work it must perform.

Published by GPU Smith · 2026-10-18 · gpudirect rdma validation · gpudirect rdma · nvidia-peermem · dma-buf · pcie topology · ib_write_bw · rdma troubleshooting · gpu networking · cluster acceptance

Original article: <https://gpusmith.com/articles/en/gpudirect-rdma-qualification-runbook>



In this article

1. Executive Summary
 2. Introduction and Background
 3. What a GPUDirect RDMA Pass Establishes
 4. Qualify the GPU-NIC PCIe Topology
 5. Review BAR Resources, IOMMU and ACS
 6. Select and Verify the Peer-Memory Mechanism
 7. Run Controlled GPU-Memory Transfer Tests
 8. Isolate Faults and Preserve the Evidence
 9. Data Analysis and Evidence
 10. Implications and Future Directions
 11. Conclusion
-

Executive Summary

GPUDirect RDMA validation should establish that a specific GPU and RDMA adapter can register and transfer the intended GPU allocation through a documented direct-access mechanism. NVIDIA's GPUDirect guide identifies a shared upstream PCIe root complex as a key requirement for its documented PCIe peer path. Eligibility therefore begins with device placement, not a network bandwidth test. Linux's peer-to-peer documentation independently explains the restrictions around forwarding between PCIe hierarchy domains. (docs.nvidia.com) (docs.kernel.org)

A loaded nvidia-peermem module is not the definition of a pass. GPU Operator **26.3** documents DMA-BUF and legacy peer-memory alternatives. Its DMA-BUF prerequisites include an open GPU kernel module, **CUDA 11.7 or later**, **Linux 5.12 or later**, and supported Turing-or-newer GPU categories. These are that deployment stack's prerequisites, not a universal compatibility guarantee. The driver README separately warns that the older peer provider and nvidia-peermem cannot operate concurrently. (docs.nvidia.com) (download.nvidia.com) Distribution packaging also matters: Ubuntu's kernel-team guidance describes its own transition associated with **kernel 6.8**. (discourse.ubuntu.com)

The recommended evidence sequence is **inventory, topology, resources and security policy, registration, controlled transfer, correctness coverage, then application correlation**. Inspect negotiated links rather than capability alone. Reconcile IOMMU and ACS requirements with OEM and security owners: Linux explicitly warns that disabling ACS redirect removes device isolation. (docs.altera.com) (docs.kernel.org) Use host-memory RDMA as a control, then select the intended GPU-memory mechanism explicitly. Current perftest documentation supports content validation, but the maintainer notes that checking may skip completed chunks when it cannot keep up. Retain coverage as well as the final status. (github.com)

There is **no measured throughput or application speedup claim** in this report. Its PCIe calculations provide context: an x16 link at **32 GT/s** with **128b/130b encoding** has an encoding-adjusted ceiling of **63.02 GB/s per direction**, before protocol overhead. This is not a GPUDirect acceptance threshold. (pcisig.com)

(www.intel.com) Finally, a CUDA-aware application can still stage traffic through host memory. A host qualification pass must be followed by evidence of the application's selected path and representative workload measurements. (openucx.readthedocs.io)

Introduction and Background

Healthy network links do not establish a direct path between a graphics processing unit (GPU) and a network interface controller (NIC). **GPUDirect remote direct memory access (RDMA)** provides GPU-to-peer Peripheral Component Interconnect Express (PCIe) access, with shared-root-complex constraints in NVIDIA's documented path. Device placement is therefore part of host qualification. (docs.nvidia.com)

This report addresses cluster and network engineers deciding whether a particular GPU, host channel adapter (HCA), and software stack are eligible for that path. It distinguishes a host-side connectivity decision from acceptance of **RDMA over Converged Ethernet (RoCE)** or InfiniBand fabric behavior. It also separates qualification from process placement and application optimization. Linux's PCI peer-to-peer documentation explains why this matters: traffic can remain below a switch, while forwarding between separate hierarchy domains is generally blocked unless hardware is known to support it. (docs.kernel.org)

GPU Smith's independent advisory perspective here is to make the acceptance decision reproducible: retain the selected devices, exact software, observed registration mechanism, test output, and limits of the conclusion. The proposed workflow and acceptance categories are engineering recommendations. They are not a vendor certification or a report of tests performed on a supplied cluster.

The evidence reviewed for this report includes vendor documentation, Linux and RDMA implementation documentation, distribution guidance, and maintained test utilities. Commands below are **templates requiring local substitution**, not executed measurements. No host inventory, logs, or benchmark results were supplied. A completed report should therefore distinguish documented eligibility from locally demonstrated behavior, especially because a CUDA-aware library can accept GPU buffers while staging traffic through host memory. (docs.open-mpi.org)

What a GPUDirect RDMA Pass Establishes

Direct transfer describes where payload data moves. It does not imply that the CPU disappears from setup, registration, synchronization, or completion handling. A useful validation record identifies the allocation, export or peer-memory mechanism, RDMA registration, selected devices, and completed operation. The verbs API makes registration a distinct step: ordinary memory registration and DMA-BUF registration have different interfaces, and registration can fail independently of allocation. (man7.org)

Use separate acceptance statements rather than an undifferentiated "RDMA passed" label:

- **Eligibility:** The documented hardware, topology, driver, and operating-system requirements match the intended deployment.
- **Device access:** The test identity can open the intended GPU and RDMA device.
- **Registration:** The selected GPU allocation successfully becomes an RDMA memory region.
- **Transport:** The selected endpoints complete the intended operation using that region.

- **Correctness:** A supported checker verifies received contents, with its coverage recorded.
- **Performance:** Separately collected measurements meet a locally approved baseline for the same configuration.

These categories are a proposed reporting structure. Their separation follows the implementation boundary between GPU allocation and export, verbs registration, and transfer validation. Perftest's CUDA implementation explicitly distinguishes allocating GPU memory from requesting a DMA-BUF handle. Its resource code separately reports memory-region registration errors. (github.com)

DMA-BUF, Linux's shared direct-memory-access buffer framework, and **nvidia-peermem** are alternative integration paths in NVIDIA's GPU Operator documentation. Absence of the latter is therefore not an automatic failure. The older `nv_peer_mem` project documents comparable driver-integrated functionality beginning with **R470**, which helps explain why historical runbooks use different module names. (docs.nvidia.com) (github.com)

Conversely, a completed application collective is insufficient direct-path evidence. **Unified Communication X (UCX)** explicitly supports either direct GPU operations or copies through host memory. An application can therefore be functional without demonstrating the mechanism being qualified. The recommended acceptance wording is narrow: "the recorded GPU-memory registration and transfer configuration passed the recorded checks," followed by the actual evidence and exclusions. (openucx.readthedocs.io)

Qualify the GPU-NIC PCIe Topology

Begin with stable identifiers. Record each GPU's universally unique identifier (UUID) and complete PCI domain, bus, device, and function address, abbreviated **BDF**. NVIDIA recommends UUID or PCI bus ID for consistent identification because index ordering can change across reboots. The `lspci -D` option preserves PCI domain numbers; `-t` shows the bus tree and `-vv` exposes detailed device information. (docs.nvidia.com) (www.man7.org)

Use a read-only inventory first. Discover supported query fields from the installed binary rather than copying a query from an unrelated driver release:

```
nvidia-smi --help-query-gpu
nvidia-smi --query-gpu=index,uuid,pci.bus_id,name --format=csv
nvidia-smi topo -m
lspci -D -t
lspci -D -vv
ibv_devices
ibv_devinfo -d mlx5_0
```

The NVIDIA manual documents selective properties and CSV output. Red Hat's core RDMA procedure documents adapter enumeration and selected-device inspection. Here, `mlx5_0` is a replaceable local device name, not a required name or a statement about the intended adapter. (docs.nvidia.com) (docs.redhat.com)

Trace the GPU and NIC through intervening switches and bridges to their upstream root ports. A shared non-uniform memory access (NUMA) node is useful locality evidence, but this runbook does not accept that label as a substitute for drawing the PCIe path. HPE's architecture guidance illustrates internal PCIe switches connecting GPUs, NICs, and processors; an actual server's slot map must establish its own arrangement. (arubanetworking.hpe.com)

Table 1 is the proposed eligibility crosswalk. Complete it for each tested pair, retaining the underlying output rather than only a colored status.

Boundary	Evidence to retain	Decision rule
GPU identity	UUID, complete BDF, model	Match the allocation reported by the test.
RDMA identity	Adapter, port, firmware, associated NIC BDF	Confirm the intended local device. (docs.redhat.com)
Shared hierarchy	Annotated PCIe tree and OEM slot map	Investigate unsupported hierarchy crossings. (docs.kernel.org)
Link state	Capability and negotiated speed and width	Compare LnkCap with LnkSta. (docs.altera.com)
BAR resources	GPU aperture total, used, and free	Retain resource evidence before registration.
Translation and routing	Effective IOMMU and ACS policy	Use approved platform-specific settings. (docs.kernel.org)
Memory integration	DMA-BUF or peer-memory mechanism	Match the chosen stack's requirements.
Test outcome	Allocation, registration, transfer, checker output	Assign separate results to each stage. (man7.org)

The table prevents a locality label from standing in for eligibility. It also makes changes reviewable: a slot move changes the topology record, while a driver change changes the software record. Neither should inherit the other's evidence automatically.

Finally, distinguish **link capability from negotiated operation**. Altera's PCIe guide identifies LnkCap as advertised capability and LnkSta as negotiated speed and width. Inspect relevant upstream links as well as endpoints; a capability listing alone does not establish the operating path's available bandwidth. (docs.altera.com)

Review BAR Resources, IOMMU and ACS

A **base address register (BAR)** describes a PCI resource window. NVIDIA's management documentation describes BAR1 as mapping GPU device memory for access by CPUs or other PCIe devices and exposes total, used, and free values. Record these values around registration attempts. Do not interpret a GPU's installed memory capacity as proof that the same amount is available through the relevant aperture. (docs.nvidia.com)

NVIDIA's documented PCIe peer path requires disabled or identity/pass-through **input-output memory management unit (IOMMU)** translation because peers must share physical addresses. Reconcile that compatibility constraint with supported platform security policy before considering a change. (docs.nvidia.com)

Access Control Services (ACS) affect PCIe routing and isolation. Linux explicitly warns that disabling ACS redirect removes isolation between devices and can enlarge IOMMU groups. Its VFIO documentation also explains that topology and interconnect properties influence isolation granularity. An apparent connectivity improvement does not, by itself, establish an acceptable isolation design. (docs.kernel.org)

Before changing policy, use this review sequence:

- **Map the affected path:** Identify the exact endpoints, bridges, and switch ports.
- **Identify the deployment model:** Record bare-metal ownership, virtualization, and tenant boundaries.
- **Read model-specific guidance:** Use the server's supported configuration and firmware documentation.
- **Approve the exception:** Have the platform and security owners accept any isolation change.
- **Limit the change:** Record only the devices and settings actually affected.
- **Prepare rollback:** Preserve the original state and a recovery method before applying it.

The distinction between general constraints and model-specific guidance is material. Dell's **PowerEdge XE9680L** manual explicitly calls for X2 APIC and operating-system ACS changes for GPUDirect RDMA performance. Separate Dell guidance links ACS state to Virtualization Technology on its discussed platforms. These are scoped OEM instructions, not portable commands for arbitrary servers. (www.dell.com) (infohub.delltechnologies.com)

Retain **PCIe Advanced Error Reporting (AER)** evidence before and after tests. Linux documents captured-error messages and exposed counters. Use the relevant devices' counter changes and timestamps as supporting evidence; absence of a logged error does not replace successful registration, content checking, or topology qualification. (docs.kernel.org)

Select and Verify the Peer-Memory Mechanism

Freeze the software inventory before modifying packages: kernel, GPU driver and module flavor, CUDA libraries, NIC driver and firmware, RDMA userspace, test binary, and deployment image. Record whether network drivers come from the distribution or an **OpenFabrics Enterprise Distribution (OFED)** package. The purpose is to identify one coherent configuration that can be reproduced and rolled back.

Table 2 summarizes documented integration paths **reviewed on October 18, 2026**. The numerical prerequisites are scoped to the named documentation, rather than presented as universal requirements.

Documented path	Documented prerequisites or behavior	Qualification consequence
GPU Operator 26.3, DMA-BUF	Open GPU kernel module; CUDA 11.7 or later; Linux 5.12 or later; supported Turing-or-newer GPU categories; OFED optional. (docs.nvidia.com)	Verify the complete supported combination and actual export/registration.
GPU Operator 26.3, legacy peer memory	Supported GPU driver; MLNX_OFED or DOCA-OFED required in its table. (docs.nvidia.com)	Verify <code>nvidia-peermem</code> against the selected network-driver stack.
NVIDIA driver README 575.51.02	Old <code>nv_peer_mem</code> and <code>nvidia-peermem</code> cannot operate together; installation order can require rebuilding the GPU driver against MOFED APIs. (download.nvidia.com)	Remove ambiguity about which provider is active.

Documented path	Documented prerequisites or behavior	Qualification consequence
Ubuntu kernel-team migration guidance	Its April 23, 2024 post scopes a transition toward DMA-BUF to kernel 6.8. (discourse.ubuntu.com)	Check distribution policy alongside upstream capability.

These rows describe different administrative scopes. A distribution's packaging transition, an Operator deployment matrix, and a driver README answer different questions. Their version numbers should remain attached to their sources. A locally installed package may also expose fewer features than a current upstream manual.

For a legacy configuration, NVIDIA documents `modprobe nvidia-peermem` and warns about the older provider running concurrently. Its README instructs reinstalling the GPU driver when installation before MOFED prevented compilation against the required RDMA APIs. Apply the instructions for the selected release during an approved maintenance change, then retain module-load output and kernel diagnostics.

(download.nvidia.com)

Check access under the **actual job identity**:

- **RDMA enumeration:** Inspect the chosen adapter and port using the distribution's supported utilities. (docs.redhat.com)
- **Device permissions:** Verify access to the intended `/dev/infiniband/uverbsN` character device and its assigned group or mode. (docs.kernel.org)
- **Locked memory:** Inspect effective `ulimit -l`; Red Hat documents login-time application of its pinning limits. (docs.redhat.com)
- **Container devices:** Expose only required devices through the runtime's supported device controls. (docs.docker.com)
- **Container limits:** Set and verify the intended `memlock` limit; Docker documents its separate soft/hard limit controls. (docs.docker.com)
- **Scheduler allocation:** Confirm both GPU and RDMA resources; Kubernetes documents distinct device-plugin implementations. (kubernetes.io)

Docker documents `IPC_LOCK` as an addable capability for memory-lock operations. That does not establish that every RDMA container needs it. Check the runtime's effective limits and required operations before adding capabilities. A privileged launch should not become the acceptance procedure merely because it masks an access problem. (docs.docker.com)

Run Controlled GPU-Memory Transfer Tests

Choose a peer with a known working host-memory RDMA configuration and use a dedicated test window. Record the selected GPU, adapter, port, global identifier (GID), operation, payload size, queue configuration, and binary version on each endpoint. Keep the same fabric path while changing memory type where possible. This makes the comparison interpretable without claiming that it isolates every possible host variable.

Check the installed utility first. Perftest documents automatic CUDA-header discovery starting with **release 25.07**; older build instructions may differ. Its maintained manual documents `--use_cuda`, PCI-bus-ID selection, and `--use_cuda_dmabuf`. Inspect the actual binary's help rather than assuming a distribution build includes every upstream feature. (github.com)

The following **command template** uses locally chosen test parameters, not an acceptance benchmark. Substitute endpoint-local adapter, port, GID index, and GPU ordinal. Both sides must agree on test semantics; their local device names need not be identical. Perftest documents server/client pairing and these option meanings. (manpages.opensuse.org)

```
# Server: host-memory control
ib_write_bw -d mlx5_0 -i 1 -x 3 -s 65536 -n 10000 --report_gbits

# Client: replace SERVER_IP with the server address
ib_write_bw -d mlx5_0 -i 1 -x 3 -s 65536 -n 10000 --report_gbits SERVER_IP
```

Repeat the pair with `--use_cuda=0` for the selected GPU allocation. For a supported DMA-BUF configuration, also append `--use_cuda_dmabuf` before the client's server address. For legacy peer-memory testing, omit that DMA-BUF flag and verify the intended module path. These commands select the mechanism explicitly; they do not demonstrate that the local stack supports it. (github.com)

```
#Server GPU-memory DMA-BUF transfer, if supported
ib_write_bw -d mlx5_0 -i 1 -x 3 -s 65536 -n 10000 --report_gbits --use_cuda=0 --use_cuda_dmabuf

#Client, using its own locally selected devices
ib_write_bw -d mlx5_0 -i 1 -x 3 -s 65536 -n 10000 --report_gbits --use_cuda=0 --use_cuda_dmabuf SERVER_IP
```

Run the sequence as distinct observations:

- **Host control:** Establish whether ordinary host-memory RDMA completes.
- **GPU allocation:** Capture the selected GPU and successful allocation.
- **Memory registration:** Retain export and registration messages, including failure codes.
- **GPU transfer:** Complete the same operation using the selected GPU allocation.
- **Correctness:** Run a supported content checker separately.
- **Direction:** Exchange endpoint roles and qualify the reverse configuration.
- **Repeatability:** Repeat the frozen configuration before declaring a local baseline.

Oracle's tutorial shows CUDA-enabled test output containing GPU selection, PCIe information, and allocation. Those lines are useful allocation evidence. Registration still needs its own result: the verbs API separately returns a memory-region pointer or failure, and perftest reports registration errors with diagnostics. (docs.oracle.com) (man7.org) (github.com)

For builds supporting `--data_validation`, the current manual requires reliable-connection bandwidth WRITE/READ tests and matching transmit depths of at least **32**; `-a` is incompatible. Run correctness and throughput separately because checking adds overhead. Record mismatches, skipped work, and coverage:

the maintainer documents chunks completing faster than the checker can verify. A successful status without coverage context is incomplete evidence. (manpages.opensuse.org) (github.com)

Completion is not CUDA visibility. Follow the allocation's documented synchronization protocol before dependent GPU work; NVIDIA warns that concurrent kernels can observe inconsistent DMA writes. (docs.nvidia.com)

Isolate Faults and Preserve the Evidence

Use the earliest failed boundary to direct the next diagnostic step. The following is a **proposed fault tree**, not a claim that one symptom uniquely identifies one cause:

```
GPU/RDMA device unavailable
  -> identity, allocation, permissions, scheduler resources
GPU allocation succeeds; registration fails
  -> export capability, peer provider, driver/API compatibility, BAR resources
Host-memory transfer fails
  -> RDMA device, endpoint parameters, connection setup, fabric
Host-memory passes; GPU-memory fails
  -> prioritize GPU registration and the selected host peer path
Transfer completes; content checking fails or lacks coverage
  -> correctness, synchronization, checker configuration, resource contention
Controlled transport passes; application is unsatisfactory
  -> library path selection, collectives, affinity, workload behavior
```

Registration failure is a particularly useful boundary. Perftest's resource implementation distinguishes a memory-region error and an explicit unsupported DMA-BUF diagnostic. Preserve the exact error rather than collapsing every registration failure into a generic network fault. Successful CUDA allocation has already answered a different question. (github.com)

For application correlation, inspect **Message Passing Interface (MPI)** and UCX build/runtime capability first. Open MPI documents `ucx_info -v`, `mpi_info`, and its runtime CUDA-support query. UCX additionally documents `ucx_info -d` for available transports and devices. Capability discovery establishes availability, not use by every message. (docs.open-mpi.org) (openucx.readthedocs.io)

Then capture the actual operation. UCX documents `UCX_LOG_LEVEL=info` and `UCX_PROTO_INFO=y` for transport/protocol information. For the **NVIDIA Collective Communications Library (NCCL)**, retain debug logs and their subsystem context; NVIDIA cautions that formatting varies with version. Correlate those records with the previously qualified GPU and NIC identities. (openucx.readthedocs.io) (docs.nvidia.com)

A temporary application-level negative control can strengthen interpretation. NCCL documents `NCCL_NET_GDR_LEVEL=LOC` as disabling GPU Direct RDMA by distance policy, but its `C2C` setting can override that policy. On releases supporting it, account for `NCCL_NET_GDR_C2C=0` and confirm behavior in logs. Using `NCCL_IB_DISABLE` is a different question: setting it to 1 disables the IB/RoCE transport itself. (docs.nvidia.com)

Package a qualification record containing:

- **Identity:** Host, GPU UUID/BDF, NIC BDF, adapter, port, and test identity.

- **Configuration:** Firmware, kernel, modules, userspace libraries, container image, and test build.
- **Topology:** Annotated bridges, negotiated links, and effective routing/translation policy.
- **Stage results:** Allocation, export, registration, transfer, and correctness coverage.
- **Measurements:** Units, direction, payload, concurrency, repetitions, and local baseline.
- **Rollback:** Original settings, approved changes, restoration commands, and retest scope.

Data Analysis and Evidence

No measured bandwidth is reported here. The quantitative analysis is an **encoding-adjusted PCIe ceiling**, intended to check units and impossible expectations. PCI-SIG documents the successive per-lane signaling rates; Intel's PIPE specification documents **128b/130b encoding at 8, 16, and 32 GT/s**. These are interface properties, not GPUDirect benchmark results. ([pcisig.com](https://www.pcisig.com)) (www.intel.com)

For a one-direction link, calculate:

```
GB/s ceiling = signaling rate in GT/s × lanes × 128/130 ÷ 8
Gbit/s ceiling = signaling rate in GT/s × lanes × 128/130
```

The calculation converts signaling into encoded data capacity. It excludes PCIe packet overhead and other system limits. Intel's processor documentation independently associates PCIe Gen4 with **16 GT/s** and Gen5 with **32 GT/s**, using the same encoding. ([edc.intel.com](https://www.edc.intel.com))

Table 3 applies that formula to an **x16** link. Values are rounded calculations using decimal GB/s, not a throughput target.

PCIe generation	Rate per lane	Calculated GB/s, one direction	Calculated Gbit/s, one direction
Gen3	8 GT/s	15.75	126.03 (pcisig.com) (www.intel.com)
Gen4	16 GT/s	31.51	252.06 (edc.intel.com) (www.intel.com)
Gen5	32 GT/s	63.02	504.12 (edc.intel.com) (www.intel.com)

The table is derived from the documented signaling and encoding inputs. It supplies no acceptance percentage: a real path includes devices, links, transaction overhead, and the chosen operation. Apply the formula to negotiated width and speed, not merely advertised capability. ([pcisig.com](https://www.pcisig.com)) (www.intel.com) (docs.altera.com)

Keep **raw signaling**, **encoding-adjusted capacity**, and **measured payload throughput** as separate fields. A bidirectional sum is also different from a one-direction result. This distinction avoids treating a large advertised aggregate as the amount one transfer should achieve. The local evidence sheet should state direction and reporting units before comparing runs.

Perftest's `--report_gbits` switches reporting to Gbit/sec rather than its documented MiB/sec default. Preserve the selected reporting mode in the command record. A unit conversion can reconcile two reports without explaining their underlying performance; it should never be used to hide a changed test configuration. (manpages.opensuse.org)

For repeated local measurements, retain raw output and report the sample count, median, range, warm-up policy, and whether content checking was enabled. These are proposed reporting choices, not vendor-defined pass thresholds. Define any acceptance band in advance for the frozen configuration. Investigate changes against that baseline before changing congestion controls or collective parameters.

Finally, report **content coverage separately from bandwidth**. Perftest's maintainer explicitly notes skipped or stale validation work under contention. The appropriate evidence includes how much was checked and any mismatches, rather than only a throughput number or final process exit status. (github.com)

Implications and Future Directions

The useful qualification unit is a **versioned GPU-NIC-host configuration**, not a GPU model alone. A change to slot placement, firmware, kernel, GPU module, RDMA driver, userspace, container exposure, or application library creates a reason to review the affected boundary. The proposed process is to rerun the smallest relevant checks, then refresh the complete record before wider deployment.

Distribution policy illustrates why current documentation matters. Ubuntu's kernel-team migration notice describes a transition associated with its kernel packaging, and documents signed open-driver variants. That is evidence about Ubuntu's stated deployment path, not a reason to infer identical behavior across distributions or every GPU generation. (discourse.ubuntu.com)

For containers and schedulers, retain the same evidence boundary:

- **Host qualification:** Preserve a controlled result outside the workload container.
- **Container qualification:** Repeat under the intended image, identity, device mappings, and limits.
- **Scheduler qualification:** Retain allocated GPU and RDMA resource identities.
- **Application qualification:** Verify the selected transport on representative operations.
- **Change control:** Remove temporary diagnostic overrides and preserve the final production configuration.

Kubernetes documents vendor drivers and device plugins for GPU scheduling, while its device-plugin framework separately covers high-performance NICs and InfiniBand adapters. The engineering inference is that successful GPU scheduling alone does not demonstrate RDMA-device allocation or a usable peer-memory path. Verify both resources in the launched workload. (kubernetes.io)

The same caution applies to CUDA-aware communication. Open MPI documents transport- and version-dependent CUDA support, and a documented host-staging fallback exists. After a host qualification pass, application logs and workload measurements remain necessary to determine whether that application uses the path effectively. No connectivity result in this report establishes an application speedup. (docs.open-mpi.org)

Conclusion

GPUDirect RDMA validation should produce a bounded, reproducible claim. The record should identify the GPU and RDMA device, show their PCIe path, document acceptable translation and routing policy, establish the selected memory-integration mechanism, and preserve successful registration and transfer

evidence. Correctness checking and local performance measurement deserve separate results.

The proposed sequence is deliberately layered. First inventory stable identifiers and software. Then inspect the topology and resources. Reconcile IOMMU and ACS requirements with platform support and isolation policy. Verify the selected DMA-BUF or peer-memory path under the intended user and container context. Run the host-memory control, GPU-memory transfer, supported content check, and reverse configuration before interpreting an application collective.

A pass should remain attached to the tested configuration and operation. It should not be expanded into a universal compatibility claim, a fabric certification, or an assumed application improvement. A failure should likewise remain attached to the earliest failed boundary until evidence narrows its cause.

The practical deliverable is the evidence bundle: inventory, topology, configuration, commands, diagnostics, correctness coverage, measurements, and rollback information. With that record, an engineer can explain precisely what worked, what remains unverified, and which next check will reduce uncertainty. That is the point at which fabric tuning and application optimization become better-informed engineering work.

Source links

Links cited in this article, in order of first appearance. Full addresses are provided for readers using extracted text.

1. <https://docs.nvidia.com/cuda/gpudirect-rdma>
2. <https://docs.kernel.org/driver-api/pci/p2pdma.html>
3. <https://docs.nvidia.com/datacenter/cloud-native/gpu-operator/26.3/gpu-operator-rdma.html>
4. https://download.nvidia.com/XFree86/Linux-x86_64/575.51.02/README/nvidia-peermem.html
5. <https://discourse.ubuntu.com/t/nvidia-gpudirect-over-infiniband-migration-paths/44425>
6. <https://docs.altera.com/r/docs/790711/24.3.1/axi-streaming-ip-for-pci-express-user-guide/operating-system-tools-and-utilities>
7. <https://docs.kernel.org/admin-guide/kernel-parameters.html>
8. <https://github.com/linux-rdma/perftest/blob/master/README>
9. <https://pcisig.com/what-bit-rates-does-pcie-50-specification-support-and-how-does-it-compare-prior-pcie-generations>
10. <https://www.intel.com/content/dam/www/public/us/en/documents/white-papers/phy-interface-pci-express-sata-usb30-architectures-3.1.pdf>
11. <https://openucx.readthedocs.io/en/master/faq.html>
12. <https://docs.open-mpi.org/en/v5.0.5/tuning-apps/networking/cuda.html>
13. https://man7.org/linux/man-pages/man3/ibv_reg_mr.3.html
14. https://github.com/linux-rdma/perftest/blob/master/src/cuda_memory.c
15. https://github.com/Mellanox/nv_peer_memory
16. <https://docs.nvidia.com/deploy/nvidia-smi/index.html>
17. <https://www.man7.org/linux/man-pages/man8/lspci.8@@pciutils.html>
18. https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/9/html/configuring_infiniband_and_rdma_networks/configuring-the-core-rdma-subsystem_configuring-infiniband-and-rdma-networks
19. <https://arubanetworking.hpe.com/techdocs/VSG/docs/040-dc-design/esp-dc-design-025-lossless-ethernet>

20. https://www.dell.com/support/manuals/en-uk/poweredge-xe9680l/pexe9680l_ism_pub/processor-settings?guid=guid-638e046f-9d70-4238-a560-624f9b0fd057&lang=en-us
21. <https://infohub.delltechnologies.com/nl-nl/p/resolving-acis-disable-on-dell-poweredge-platforms>
22. <https://docs.kernel.org/PCI/pcieaer-howto.html>
23. https://docs.kernel.org/infiniband/user_verbs.html
24. https://docs.redhat.com/en/documentation/red_hat_enterprise_linux/9/html/configuring_infiniband_and_rdma_networks/increasing-the-amount-of-memory-that-users-are-allowed-to-pin-in-the-system_configuring-infiniband-and-rdma-networks
25. <https://docs.docker.com/engine/containers/run>
26. <https://docs.docker.com/reference/cli/docker/container/run>
27. <https://kubernetes.io/docs/concepts/extend-kubernetes/compute-storage-net/device-plugins>
28. https://manpages.opensuse.org/Leap-16.0/perftest/ib_write_bw.1.en.html
29. <https://docs.oracle.com/en/learn/gpudirect-rdma-ib-write-bw>
30. https://github.com/linux-rdma/perftest/raw/refs/heads/master/src/perftest_resources.c
31. <https://manpages.opensuse.org/Tumbleweed/perftest/perftest.1.en.html>
32. <https://docs.open-mpi.org/en/main/tuning-apps/accelerators/cuda.html>
33. <https://docs.nvidia.com/deeplearning/nccl/user-guide/docs/troubleshooting/logging.html>
34. <https://docs.nvidia.com/deeplearning/nccl/user-guide/docs/env.html>
35. <https://edc.intel.com/content/www/us/en/design/products/platforms/details/raptor-lake-s/13th-generation-core-processors-datasheet-volume-1-of-2/002/pci-express-architecture>
36. <https://kubernetes.io/docs/tasks/manage-gpus/scheduling-gpus>

THE PUBLISHER

About GPU Smith

GPU Smith provides independent engineering and research for private AI infrastructure. We help technical buyers and operators reason about GPU workload sizing, hardware selection, procurement, deployment and inference operations, from the compute system to the networking, power and cooling requirements around it.

Start with the workload

Useful infrastructure decisions begin with the models, data, throughput, latency and operating constraints a team actually has. GPU Smith helps connect those requirements with system design and deployment choices. The goal is a privately controlled AI environment whose capacity and operational demands are understood before a purchasing decision.

Hardware and supplier research

Our public reference library covers GPUs, complete systems and networking components. The server-vendor directory and manufacturing research help teams investigate suppliers, compare options and follow sources behind technical claims. We distinguish manufacturer specifications from measured benchmarks and advertised capabilities from tested configurations. Reference pages are research resources, not a live inventory listing or a binding equipment quote.

Deployment and operations

GPU Smith's engineering scope includes infrastructure integration, networking, power, cooling and the practical operation of inference workloads. Our published articles explain the tradeoffs behind deployment, procurement and ongoing operations so teams can ask better questions and document decisions.

Work with GPU Smith

Explore the [hardware library](#), [GPU references](#), [networking references](#) and [vendor research](#). [Contact GPU Smith](#) with your workloads and deployment constraints to discuss a project and confirm current scope, pricing and availability.

Inclusion of a manufacturer or product in our research does not imply a partnership, certification or endorsement.

Website: <https://gpusmith.com>

This article is educational. Verify consequential medical, legal, financial, regulatory and technical decisions with appropriate professionals and the cited primary sources. Company services are described separately from the article's research findings.