[NCCL](../index.html)

[2.25](https://docs.nvidia.com/deeplearning/sdk/nccl-archived/index.html)

* [Overview of NCCL](../overview.html)
* [Setup](../setup.html)
* [Using NCCL](../usage.html)
  * [Creating a Communicator](../usage/communicators.html)
    * [Creating a communicator with options](../usage/communicators.html#creating-a-communicator-with-options)
    * [Creating a communicator using multiple ncclUniqueIds](../usage/communicators.html#creating-a-communicator-using-multiple-nccluniqueids)
  * [Creating more communicators](../usage/communicators.html#creating-more-communicators)
    * [Using multiple NCCL communicators concurrently](../usage/communicators.html#using-multiple-nccl-communicators-concurrently)
  * [Finalizing a communicator](../usage/communicators.html#finalizing-a-communicator)
  * [Destroying a communicator](../usage/communicators.html#destroying-a-communicator)
    * [Error handling and communicator abort](../usage/communicators.html#error-handling-and-communicator-abort)
    * [Asynchronous errors and error handling](../usage/communicators.html#asynchronous-errors-and-error-handling)
  * [Fault Tolerance](../usage/communicators.html#fault-tolerance)
  * [Collective Operations](../usage/collectives.html)
    * [AllReduce](../usage/collectives.html#allreduce)
    * [Broadcast](../usage/collectives.html#broadcast)
    * [Reduce](../usage/collectives.html#reduce)

* [AllGather](../usage/collectives.html#allgather)
* [ReduceScatter](../usage/collectives.html#reducescatter)
* [Data Pointers](../usage/data.html)
* [CUDA Stream Semantics](../usage/streams.html)
* [Mixing Multiple Streams within the same ncclGroupStart/End() group](../usage/streams.html#mixing-multiple-streams-within-the-same-ncclgroupstart-end-group)
* [Group Calls](../usage/groups.html)
* [Management Of Multiple GPUs From One Thread](../usage/groups.html#management-of-multiple-gpus-from-one-thread)
* [Aggregated Operations (2.2 and later)](../usage/groups.html#aggregated-operations-2-2-and-later)
* [Nonblocking Group Operation](../usage/groups.html#nonblocking-group-operation)
* [Point-to-point communication](../usage/p2p.html)
* [Sendrecv](../usage/p2p.html#sendrecv)
* [One-to-all (scatter)](../usage/p2p.html#one-to-all-scatter)
* [All-to-one (gather)](../usage/p2p.html#all-to-one-gather)
* [All-to-all](../usage/p2p.html#all-to-all)
* [Neighbor exchange](../usage/p2p.html#neighbor-exchange)
* [Thread Safety](../usage/threadsafety.html)
* [In-place Operations](../usage/inplace.html)
* [Using NCCL with CUDA Graphs](../usage/cudagraph.html)
* [User Buffer Registration](../usage/bufferreg.html)
* [NVLink Sharp Buffer Registration](../usage/bufferreg.html#nvlink-sharp-buffer-registration)
* [IB Sharp Buffer Registration](../usage/bufferreg.html#ib-sharp-buffer-registration)
* [General Buffer Registration](../usage/bufferreg.html#general-buffer-registration)
* [Memory Allocator](../usage/bufferreg.html#memory-allocator)
* [NCCL API](../api.html)

* [Communicator Creation and Management Functions](../api/comms.html)
  * [ncclGetLastError](../api/comms.html#ncclgetlasterror)
  * [ncclGetErrorString](../api/comms.html#ncclgeterrorstring)
  * [ncclGetVersion](../api/comms.html#ncclgetversion)
  * [ncclGetUniqueId](../api/comms.html#ncclgetuniqueid)
  * [ncclCommInitRank](../api/comms.html#ncclcomminitrank)
  * [ncclCommInitAll](../api/comms.html#ncclcomminitall)
  * [ncclCommInitRankConfig](../api/comms.html#ncclcomminitrankconfig)
  * [ncclCommInitRankScalable](../api/comms.html#ncclcomminitrankscalable)
  * [ncclCommSplit](../api/comms.html#ncclcommsplit)
  * [ncclCommFinalize](../api/comms.html#ncclcommfinalize)
  * [ncclCommDestroy](../api/comms.html#ncclcommdestroy)
  * [ncclCommAbort](../api/comms.html#ncclcommabort)
  * [ncclCommGetAsyncError](../api/comms.html#ncclcommgetasyncerror)
  * [ncclCommCount](../api/comms.html#ncclcommcount)
  * [ncclCommCuDevice](../api/comms.html#ncclcommcudevice)
  * [ncclCommUserRank](../api/comms.html#ncclcommuserrank)
  * [ncclCommRegister](../api/comms.html#ncclcommregister)
  * [ncclCommDeregister](../api/comms.html#ncclcommderegister)
  * [ncclMemAlloc](../api/comms.html#ncclmemalloc)
  * [ncclMemFree](../api/comms.html#ncclmemfree)
* [Collective Communication Functions](../api/colls.html)
  * [ncclAllReduce](../api/colls.html#ncclallreduce)
  * [ncclBroadcast](../api/colls.html#ncclbroadcast)
  * [ncclReduce](../api/colls.html#ncclreduce)
  * [ncclAllGather](../api/colls.html#ncclallgather)
  * [ncclReduceScatter](../api/colls.html#ncclreducescatter)

    * [Group Calls](../api/group.html)
      * [ncclGroupStart](../api/group.html#ncclgroupstart)
      * [ncclGroupEnd](../api/group.html#ncclgroupend)
      * [ncclGroupSimulateEnd](../api/group.html#ncclgroupsimulateend)
    * [Point To Point Communication Functions](../api/p2p.html)
      * [ncclSend](../api/p2p.html#ncclsend)
      * [ncclRecv](../api/p2p.html#ncclrecv)
    * [Types](../api/types.html)
      * [ncclComm_t](../api/types.html#ncclcomm-t)
      * [ncclResult_t](../api/types.html#ncclresult-t)
      * [ncclDataType_t](../api/types.html#ncclDatatype-t)
      * [ncclRedOp_t](../api/types.html#ncclredop-t)
      * [ncclScalarResidence_t](../api/types.html#ncclscalarresidence-t)
      * [ncclConfig_t](../api/types.html#ncclconfig-t)
      * [ncclSimInfo_t](../api/types.html#ncclsiminfo-t)
    * [User Defined Reduction Operators](../api/ops.html)
      * [ncclRedOpCreatePreMulSum](../api/ops.html#ncclredopcreatepremulsum)
      * [ncclRedOpDestroy](../api/ops.html#ncclredopdestroy)
  * [Migrating from NCCL 1 to NCCL 2](../nccl1.html)
    * [Initialization](../nccl1.html#initialization)
    * [Communication](../nccl1.html#communication)
    * [Counts](../nccl1.html#counts)
    * [In-place usage for AllGather and ReduceScatter](../nccl1.html#in-place-usage-for-allgather-and-reducescatter)
    * [AllGather arguments order](../nccl1.html#allgather-arguments-order)
    * [Datatypes](../nccl1.html#datatypes)
    * [Error codes](../nccl1.html#error-codes)

* [Examples](../examples.html)
* [Communicator Creation and Destruction Examples](../examples.html#communicator-creation-and-destruction-examples)
* [Example 1: Single Process, Single Thread, Multiple Devices](../examples.html#example-1-single-process-single-thread-multiple-devices)
* [Example 2: One Device per Process or Thread](../examples.html#example-2-one-device-per-process-or-thread)
* [Example 3: Multiple Devices per Thread](../examples.html#example-3-multiple-devices-per-thread)
* [Example 4: Multiple communicators per device](../examples.html#example-4-multiple-communicators-per-device)
* [Communication Examples](../examples.html#communication-examples)
* [Example 1: One Device per Process or Thread](../examples.html#example-1-one-device-per-process-or-thread)
* [Example 2: Multiple Devices per Thread](../examples.html#example-2-multiple-devices-per-thread)
* [NCCL and MPI](../mpi.html)
* [API](../mpi.html#api)
* [Using multiple devices per process](../mpi.html#using-multiple-devices-per-process)
* [ReduceScatter operation](../mpi.html#reducescatter-operation)
* [Send and Receive counts](../mpi.html#send-and-receive-counts)
* [Other collectives and point-to-point operations](../mpi.html#other-collectives-and-point-to-point-operations)
* [In-place operations](../mpi.html#in-place-operations)
* [Using NCCL within an MPI Program](../mpi.html#using-nccl-within-an-mpi-program)
* [MPI Progress](../mpi.html#mpi-progress)
* [Inter-GPU Communication with CUDA-aware

MPI](../mpi.html#inter-gpu-communication-with-cuda-aware-mpi)

* [Environment Variables](../env.html)

  * [System configuration](../env.html#system-configuration)

    * [NCCL_SOCKET_IFNAME](../env.html#nccl-socket-ifname)

      * [Values accepted](../env.html#values-accepted)

    * [NCCL_SOCKET_FAMILY](../env.html#nccl-socket-family)

      * [Values accepted](../env.html#id2)

    * [NCCL_SOCKET_RETRY_CNT](../env.html#nccl-socket-retry-cnt)

      * [Values accepted](../env.html#id3)

    * [NCCL_SOCKET_RETRY_SLEEP_MSEC](../env.html#nccl-socket-retry-sleep-msec)

      * [Values accepted](../env.html#id4)

    * [NCCL_SOCKET_NTHREADS](../env.html#nccl-socket-nthreads)

      * [Values accepted](../env.html#id5)

    * [NCCL_NSOCKS_PERTHREAD](../env.html#nccl-nsocks-perthread)

      * [Values accepted](../env.html#id6)

    * [NCCL_CROSS_NIC](../env.html#nccl-cross-nic)

      * [Values accepted](../env.html#id7)

    * [NCCL_IB_HCA](../env.html#nccl-ib-hca)

      * [Values accepted](../env.html#id8)

    * [NCCL_IB_TIMEOUT](../env.html#nccl-ib-timeout)

      * [Values accepted](../env.html#id9)

    * [NCCL_IB_RETRY_CNT](../env.html#nccl-ib-retry-cnt)

      * [Values accepted](../env.html#id10)

    * [NCCL_IB_GID_INDEX](../env.html#nccl-ib-gid-index)

      * [Values accepted](../env.html#id11)

    * [NCCL_IB_ADDR_FAMILY](../env.html#nccl-ib-addr-family)

      * [Values accepted](../env.html#id12)

* [NCCL_IB_ADDR_RANGE](../env.html#nccl-ib-addr-range)
  * [Values accepted](../env.html#id13)
* [NCCL_IB_ROCE_VERSION_NUM](../env.html#nccl-ib-roce-version-num)
  * [Values accepted](../env.html#id14)
* [NCCL_IB_SL](../env.html#nccl-ib-sl)
  * [Values accepted](../env.html#id15)
* [NCCL_IB_TC](../env.html#nccl-ib-tc)
  * [Values accepted](../env.html#id16)
* [NCCL_IB_FIFO_TC](../env.html#nccl-ib-fifo-tc)
  * [Values accepted](../env.html#id17)
* [NCCL_IB_RETURN_ASYNC_EVENTS](../env.html#nccl-ib-return-async-events)
  * [Values accepted](../env.html#id18)
* [NCCL_OOB_NET_ENABLE](../env.html#nccl-oob-net-enable)
  * [Values accepted](../env.html#id19)
* [NCCL_OOB_NET_IFNAME](../env.html#nccl-oob-net-ifname)
  * [Values accepted](../env.html#id20)
* [NCCL_UID_STAGGER_THRESHOLD](../env.html#nccl-uid-stagger-threshold)
  * [Values accepted](../env.html#id21)
* [NCCL_UID_STAGGER_RATE](../env.html#nccl-uid-stagger-rate)
  * [Values accepted](../env.html#id22)
* [NCCL_NET](../env.html#nccl-net)
  * [Values accepted](../env.html#id23)
* [NCCL_NET_PLUGIN](../env.html#nccl-net-plugin)
  * [Values accepted](../env.html#id24)
* [NCCL_TUNER_PLUGIN](../env.html#nccl-tuner-plugin)
  * [Values accepted](../env.html#id25)
* [NCCL_PROFILER_PLUGIN](../env.html#nccl-profiler-plugin)

    * [Values accepted](../env.html#id26)
  * [NCCL_IGNORE_CPU_AFFINITY](../env.html#nccl-ignore-cpu-affinity)
    * [Values accepted](../env.html#id27)
  * [NCCL_CONF_FILE](../env.html#nccl-conf-file)
    * [Values accepted](../env.html#id28)
  * [NCCL_DEBUG](../env.html#nccl-debug)
    * [Values accepted](../env.html#id30)
  * [NCCL_DEBUG_FILE](../env.html#nccl-debug-file)
    * [Values accepted](../env.html#id31)
  * [NCCL_DEBUG_SUBSYS](../env.html#nccl-debug-subsys)
    * [Values accepted](../env.html#id32)
  * [NCCL_COLLNET_ENABLE](../env.html#nccl-collnet-enable)
    * [Value accepted](../env.html#value-accepted)
  * [NCCL_COLLNET_NODE_THRESHOLD](../env.html#nccl-collnet-node-threshold)
    * [Value accepted](../env.html#id33)
  * [NCCL_TOPO_FILE](../env.html#nccl-topo-file)
    * [Value accepted](../env.html#id34)
  * [NCCL_TOPO_DUMP_FILE](../env.html#nccl-topo-dump-file)
    * [Value accepted](../env.html#id35)
  * [NCCL_SET_THREAD_NAME](../env.html#nccl-set-thread-name)
    * [Value accepted](../env.html#id36)
* [Debugging](../env.html#debugging)
  * [NCCL_P2P_DISABLE](../env.html#nccl-p2p-disable)
    * [Values accepted](../env.html#id37)
  * [NCCL_P2P_LEVEL](../env.html#nccl-p2p-level)
    * [Values accepted](../env.html#id38)
    * [Integer Values (Legacy)](../env.html#integer-values-legacy)

* [NCCL_P2P_DIRECT_DISABLE](../env.html#nccl-p2p-direct-disable)
  * [Values accepted](../env.html#id39)
* [NCCL_SHM_DISABLE](../env.html#nccl-shm-disable)
  * [Values accepted](../env.html#id40)
* [NCCL_BUFFSIZE](../env.html#nccl-buffsize)
  * [Values accepted](../env.html#id41)
* [NCCL_NTHREADS](../env.html#nccl-nthreads)
  * [Values accepted](../env.html#id42)
* [NCCL_MAX_NCHANNELS](../env.html#nccl-max-nchannels)
  * [Values accepted](../env.html#id43)
* [NCCL_MIN_NCHANNELS](../env.html#nccl-min-nchannels)
  * [Values accepted](../env.html#id44)
* [NCCL_CHECKS_DISABLE](../env.html#nccl-checks-disable)
  * [Values accepted](../env.html#id45)
* [NCCL_CHECK_POINTERS](../env.html#nccl-check-pointers)
  * [Values accepted](../env.html#id46)
* [NCCL_LAUNCH_MODE](../env.html#nccl-launch-mode)
  * [Values accepted](../env.html#id47)
* [NCCL_IB_DISABLE](../env.html#nccl-ib-disable)
  * [Values accepted](../env.html#id48)
* [NCCL_IB_AR_THRESHOLD](../env.html#nccl-ib-ar-threshold)
  * [Values accepted](../env.html#id49)
* [NCCL_IB_QPS_PER_CONNECTION](../env.html#nccl-ib-qps-per-connection)
  * [Values accepted](../env.html#id50)
* [NCCL_IB_SPLIT_DATA_ON_QPS](../env.html#nccl-ib-split-data-on-qps)
  * [Values accepted](../env.html#id51)
* [NCCL_IB_CUDA_SUPPORT](../env.html#nccl-ib-cuda-support)

  * [Values accepted](../env.html#id52)
* [NCCL_IB_PCI_RELAXED_ORDERING](../env.html#nccl-ib-pci-relaxed-ordering)
  * [Values accepted](../env.html#id53)
* [NCCL_IB_ADAPTIVE_ROUTING](../env.html#nccl-ib-adaptive-routing)
  * [Values accepted](../env.html#id54)
* [NCCL_IB_ECE_ENABLE](../env.html#nccl-ib-ece-enable)
  * [Values accepted](../env.html#id55)
* [NCCL_MEM_SYNC_DOMAIN](../env.html#nccl-mem-sync-domain)
  * [Values accepted](../env.html#id56)
* [NCCL_CUMEM_ENABLE](../env.html#nccl-cumem-enable)
  * [Values accepted](../env.html#id57)
* [NCCL_CUMEM_HOST_ENABLE](../env.html#nccl-cumem-host-enable)
  * [Values accepted](../env.html#id58)
* [NCCL_NET_GDR_LEVEL (formerly NCCL_IB_GDR_LEVEL)](../env.html#nccl-net-gdr-level-formerly-nccl-ib-gdr-level)
  * [Values accepted](../env.html#id59)
  * [Integer Values (Legacy)](../env.html#id60)
* [NCCL_NET_GDR_READ](../env.html#nccl-net-gdr-read)
  * [Values accepted](../env.html#id61)
* [NCCL_NET_SHARED_BUFFERS](../env.html#nccl-net-shared-buffers)
  * [Value accepted](../env.html#id62)
* [NCCL_NET_SHARED_COMMS](../env.html#nccl-net-shared-comms)
  * [Value accepted](../env.html#id63)
* [NCCL_SINGLE_RING_THRESHOLD](../env.html#nccl-single-ring-threshold)
  * [Values accepted](../env.html#id64)
* [NCCL_LL_THRESHOLD](../env.html#nccl-ll-threshold)
  * [Values accepted](../env.html#id65)

* [NCCL_TREE_THRESHOLD](../env.html#nccl-tree-threshold)
  * [Values accepted](../env.html#id66)
* [NCCL_ALGO](../env.html#nccl-algo)
  * [Values accepted](../env.html#id67)
* [NCCL_PROTO](../env.html#nccl-proto)
  * [Values accepted](../env.html#id68)
* [NCCL_NVB_DISABLE](../env.html#nccl-nvb-disable)
  * [Value accepted](../env.html#id69)
* [NCCL_PXN_DISABLE](../env.html#nccl-pxn-disable)
  * [Value accepted](../env.html#id70)
* [NCCL_P2P_PXN_LEVEL](../env.html#nccl-p2p-pxn-level)
  * [Value accepted](../env.html#id71)
* [NCCL_RUNTIME_CONNECT](../env.html#nccl-runtime-connect)
  * [Value accepted](../env.html#id72)
* [NCCL_GRAPH_REGISTER](../env.html#nccl-graph-register)
  * [Value accepted](../env.html#id74)
* [NCCL_LOCAL_REGISTER](../env.html#nccl-local-register)
  * [Value accepted](../env.html#id75)
* [NCCL_LEGACY_CUDA_REGISTER](../env.html#nccl-legacy-cuda-register)
  * [Value accepted](../env.html#id76)
* [NCCL_SET_STACK_SIZE](../env.html#nccl-set-stack-size)
  * [Value accepted](../env.html#id77)
* [NCCL_GRAPH_MIXING_SUPPORT](../env.html#nccl-graph-mixing-support)
  * [Value accepted](../env.html#id79)
* [NCCL_DMABUF_ENABLE](../env.html#nccl-dmabuf-enable)
  * [Value accepted](../env.html#id80)
* [NCCL_P2P_NET_CHUNKSIZE](../env.html#nccl-p2p-net-chunksize)

      * [Values accepted](../env.html#id81)
    * [NCCL_P2P_LL_THRESHOLD](../env.html#nccl-p2p-ll-threshold)
      * [Values accepted](../env.html#id82)
    * [NCCL_ALLOC_P2P_NET_LL_BUFFERS](../env.html#nccl-alloc-p2p-net-ll-buffers)
      * [Values accepted](../env.html#id83)
    * [NCCL_COMM_BLOCKING](../env.html#nccl-comm-blocking)
      * [Values accepted](../env.html#id84)
    * [NCCL_CGA_CLUSTER_SIZE](../env.html#nccl-cga-cluster-size)
      * [Values accepted](../env.html#id85)
    * [NCCL_MAX_CTAS](../env.html#nccl-max-ctas)
      * [Values accepted](../env.html#id86)
    * [NCCL_MIN_CTAS](../env.html#nccl-min-ctas)
      * [Values accepted](../env.html#id87)
    * [NCCL_NVLS_ENABLE](../env.html#nccl-nvls-enable)
      * [Values accepted](../env.html#id88)
    * [NCCL_IB_MERGE_NICS](../env.html#nccl-ib-merge-nics)
      * [Values accepted](../env.html#id89)
    * [NCCL_MNNVL_ENABLE](../env.html#nccl-mnnvl-enable)
      * [Values accepted](../env.html#id90)
    * [NCCL_RAS_ENABLE](../env.html#nccl-ras-enable)
      * [Values accepted](../env.html#id91)
    * [NCCL_RAS_ADDR](../env.html#nccl-ras-addr)
      * [Values accepted](../env.html#id92)
    * [NCCL_RAS_TIMEOUT_FACTOR](../env.html#nccl-ras-timeout-factor)
      * [Values accepted](../env.html#id93)
* [Troubleshooting](../troubleshooting.html)
  * [Errors](../troubleshooting.html#errors)

* [RAS](../troubleshooting.html#ras)
* RAS
* Principle of Operation
* RAS Queries
* Sample Output
* [GPU Direct](../troubleshooting.html#gpu-direct)
* [GPU-to-GPU communication](../troubleshooting.html#gpu-to-gpu-communication)
* [GPU-to-NIC communication](../troubleshooting.html#gpu-to-nic-communication)
* [PCI Access Control Services (ACS)](../troubleshooting.html#pci-access-control-services-acs)
* [Topology detection](../troubleshooting.html#topology-detection)
* [Shared memory](../troubleshooting.html#shared-memory)
* [Docker](../troubleshooting.html#docker)
* [Systemd](../troubleshooting.html#systemd)
* [Networking issues](../troubleshooting.html#networking-issues)
* [IP Network Interfaces](../troubleshooting.html#ip-network-interfaces)
* [IP Ports](../troubleshooting.html#ip-ports)
* [InfiniBand](../troubleshooting.html#infiniband)
* [RDMA over Converged Ethernet (RoCE)](../troubleshooting.html#rdma-over-converged-ethernet-roce)

__[NCCL](../index.html)

* * *

RASÂ¶

Since NCCL 2.24, the reliability, availability, and serviceability (RAS) subsystem can be used to query the health of NCCL jobs during execution. This can help with the diagnosis and debugging of crashes and hangs. RAS is a low-overhead infrastructure that NCCL users and developers can use while the application is running. It provides a global view of the state of the running application and can aide in the detection of outliers such as unresponsive processes. With that information, users can then narrow down on the suspected root cause(s) through other techniques such as interactive debugging, system log analysis, etc.

Principle of OperationÂ¶

RAS is built into NCCL and launches during NCCL initialization. It consists of a set of threads (one per process) that establish connections with each other, forming a network that the RAS threads then use to exchange information and monitor each otherâ€™s health. In a typical configuration, the RAS network traffic (which uses plain TCP/IP sockets on top of the bootstrap/out-of-band network interface that NCCL uses during initialization) should not compete with the main NCCL traffic (which utilizes RDMA networking). RAS is lightweight and should not interfere with the main NCCL job; as such, it is enabled by default (but see [NCCL_RAS_ENABLE](../env.html#env-nccl-ras-enable)).

The RAS threads communicate with each other about any changes to the job configuration; they also exchange regular keep-alive messages. If a NCCL process crashes or hangs, the RAS threads running on other NCCL processes learn about it through the RAS network connections to that process being shut down or becoming unresponsive.

RAS QueriesÂ¶

The RAS threads also listen for client connections on `localhost`, port `28028` (these defaults can be changed using [NCCL_RAS_ADDR](../env.html#env-nccl-ras-addr)). The `ncclras` binary client can be used to connect to that socket and query the RAS subsystem for the current job status, which is then printed to standard output. The client accepts the `-h` and `-p` arguments to specify the host name and port, `-v` to produce a more verbose output in case of problems, and `-t` to specify a different timeout (`5` seconds by default; 0 disables the timeout).

As the client communication protocol is fully text-based, standard networking tools such as telnet or netcat can be used instead of the `ncclras` binary. The relevant commands include `STATUS`, `VERBOSE STATUS` (equivalent to the `ncclras` clientâ€™s `-v` argument), and `TIMEOUT <seconds>` (equivalent to `-t`); e.g., `echo verbose status | nc localhost 28028`.

Irrespective of how the query is submitted, the receiving RAS thread sends back the job summary information as well as the summary information about all the NCCL communicators; the latter is collected from all the jobâ€™s processes so, for jobs experiencing problems or ones that are particularly large, the response may take several seconds to generate. In case any issues were

encountered, additional information is provided.

Sample OutputÂ¶

This section contains excerpts of the RAS status output. Please note that the exact format and scope of the information being made available is expected to evolve; the excerpts are provided for illustrative purposes only.

Hereâ€™s an example output from a job that is progressing normally:

```
  Job summary
  ===========

   Nodes  Processes        GPUs  Processes     GPUs
  (total)   per node  per process    (total)  (total)
       4         8           1         32       32
```

Weâ€™ve got a job consisting of 32 GPUs (1 GPU per process) running on 4 nodes.

```
  Communicators... (0.00s)
  =============
```

Group #	Comms in group	Nodes per comm	Ranks per node	Ranks per comm	Ranks in group	Status	Errors
0	8	4	1	4	32	RUNNING	OK

The GPUs are split into 8 communicators, 1 GPU per node. RAS attempts to make the summary output as short as possible by grouping together objects having the same size and other important properties.

For jobs that are actively communicating during the RAS query, the following output can sometimes be observed:

Group #	Comms in group	Nodes per comm	Ranks per node	Ranks per comm	Ranks in group	Status	Errors
0	1	4	8	32	32	RUNNING	MISMATCH

The output indicates that there is an inconsistency in the information provided by different communicator ranks. Additional information is printed underneath (in this case itâ€™s in the Warnings section, indicating a potentially lower severity):

Warnings

========

#0-0 (27a079b828ff1a75) MISMATCH

Communicator ranks have different collective operation counts

26 ranks have launched up to operation 6650

6 ranks have launched up to operation 6649

Rank 0 -- GPU 0 managed by process 483072 on node 172.16.64.210

Rank 2 -- GPU 2 managed by process 483074 on node 172.16.64.210

Rank 3 -- GPU 3 managed by process 483075 on node 172.16.64.210

Rank 4 -- GPU 4 managed by process 483076 on node 172.16.64.210

Rank 5 -- GPU 5 managed by process 483077 on node 172.16.64.210

Rank 7 -- GPU 7 managed by process 483079 on node 172.16.64.210

Communicators are referred to using the `#<x>-<y>` identifiers, where `<x>` is the group number from the summary output and `<y>` is the communicator number within the group, both starting with 0 (in this example there is only one (32-GPU) communicator so, unsurprisingly, the identifier is `#0-0`). The identifier is followed by a communicator hash, which is a value that can be found in NCCLâ€™s regular debug output as well, and the rank information. RAS groups together the ranks with the same relevant property (the count of issued collective operations in this case). If a group constitutes an outlier, RAS prints additional information about each group member. By default this is done if the group size is at most 25% of the total _and_ the group has no more than 10 members; enabling verbose output relaxes this to under 50% of the total and lifts the group size limit.

The particular case above should not be a cause for concern, as long as the counts increase across repeated queries. NCCL collectives, being optimized for speed, can easily outpace the RAS collective queries, especially if the size of the collectives is fairly small. An application may also exhibit work imbalance, with certain ranks routinely arriving to the collective operations later than others â€“ an experience with a particular workload is needed to determine whatâ€™s normal and whatâ€™s not. However, if the output does not change across subsequent RAS queries, it may indicate that the communicator is â€œstuckâ€• for some reason, which could warrant an investigation.

Similar effects can sometimes be observed during communicator initialization or tear-down:

```
  Group     Comms     Nodes     Ranks     Ranks     Ranks    Status  Errors
    #  in group  per comm  per node  per comm  in group
    0         1         4       1-2        32        32  FINALIZE  MISMATCH
    1         7         4         1         4        28   RUNNING      OK
    2         1         4         1         4         4      INIT      OK

  [...]

  #0-0 (9e17999afaa87dbb) MISMATCH
    Communicator ranks have different status
    26 ranks have status UNKNOWN
```

4 ranks have status RUNNING

Rank 0 -- GPU 0 managed by process 507285 on node 172.16.64.210

Rank 8 -- GPU 0 managed by process 1598388 on node 172.16.64.212

Rank 16 -- GPU 0 managed by process 3500071 on node 172.16.64.213

Rank 24 -- GPU 0 managed by process 2405067 on node 172.16.64.222

2 ranks have status FINALIZE

Rank 4 -- GPU 4 managed by process 507289 on node 172.16.64.210

Rank 20 -- GPU 4 managed by process 3500075 on node 172.16.64.213

The above snapshot depicts a transitional situation as the initial, 32-GPU communicator is being replaced by eight 4-GPU communicators (one of which is still initializing, so it is listed separately (group `#2`) from the already initialized seven (group `#1`)). The 32-GPU communicator (`#0-0`) is being torn down, with two ranks in the middle of ncclCommFinalize, four ranks that have _not_ called ncclCommFinalize yet, and the remaining 26 ranks â€œunknownâ€• â€“ meaning that they didnâ€™t provide any information about that communicator when RAS was collecting data, simply because their call to ncclCommFinalize has already completed so they are in fact no longer that communicatorâ€™s members. Again, as long as the situation is resolved when the query is repeated, it can be ignored.

Hereâ€™s an excerpt from an invocation right after artificially creating a problem with one of the job processes:

```
Communicators... (2.05s)
=============

Group     Comms     Nodes     Ranks     Ranks     Ranks    Status  Errors
   #  in group  per comm  per node  per comm  in group
   0         1         4       7-8        32        32   RUNNING  INCOMPLETE

Errors
======

INCOMPLETE
  Missing communicator data from 1 job process
  Process 3487984 on node 172.16.64.213 managing GPU 5

#0-0 (cf264af53edbe986) INCOMPLETE
  Missing communicator data from 1 rank
  The missing rank: 21

Warnings
========

TIMEOUT
  Encountered 2 communication timeouts while gathering communicator data
```

In this case the summary takes a few seconds to generate because RAS waits for the data from the process experiencing problems (the process is unresponsive

â€“ it was stopped â€“ but RAS doesnâ€™t know it yet). Repeated queries should be much faster because once RAS determines that a process is unresponsive, it reconfigures the RAS network to route around it.

RAS will attempt to reestablish communication with the unresponsive process; if itâ€™s unable to do so for 60 seconds, it will declare the process dead (permanently):

```
Errors
======

DEAD
  1 job process is considered dead (unreachable via the RAS network)
  Process 3487984 on node 172.16.64.213 managing GPU 5

#0-0 (cf264af53edbe986) INCOMPLETE
  Missing communicator data from 1 rank
  The missing rank: 21
```

RAS will simply stop attempting to communicate with such processes over the RAS network anymore, leaving it up to the user to determine if any additional action is warranted.

[Previous](../troubleshooting.html "Troubleshooting")

* * *

Built with [Sphinx](http://sphinx-doc.org/) using a

[theme](https://github.com/rtfd/sphinx_rtd_theme) provided by [Read the

Docs](https://readthedocs.org).