ParaStation MPI

ParaStation MPI 5.0

Software Product Detailed Description

Product: ParaStation MPI 5.0

Date: August 2010
Document number: PSMPI2-1.0-4en

This software product description documents the functionality provided by the ParaStation MPI 5.0 as well as the system prerequisites required for installation and operation, licensing scheme and other useful information.

Overview

ParaStation MPI as part of the *ParaStationV5* cluster suite provides robust, flexible and scalable communication and management functions for Linux-based compute clusters. Beside parallel applications based on the Message Passing Interface specification, version 2 (MPI2), serial applications are supported, too.

Technical features

ParaStation MPI is divided into the packages

- process management (psmgmt),
- communication (pscom),
- MPI (psmpi2, psmpi2-mt),
- documentation (psdoc, psmpi2-doc).

The 'process management' and 'communication' modules are mandatory for operation.

Process management

The process management module is responsible for starting, monitoring, and terminating processes and entire applications on the cluster. For this purpose, a daemon process *(psid)* is executing on each compute node. All these daemons talk among each other using UDP over an administration network, which might be different to the MPI network used for application data.

Load distribution, process placement: When starting a parallelized application, the current node list is evaluated for available nodes and cores, utilization, number of processes executing and possible user restrictions. Based on this current information a temporary node list is generated, in which the individual processes of a parallel application are started. The distribution of processes is based on adjustable criteria. If less cores than processes are available, the processes are distributed to the temporary node list in round robin fashion, provided overbooking of cores is enabled.

The criteria listed for distribution can be influences by various environment variables or options to the *mpiexec* command.

Within ParaStation MPI, nodes can be explicitly defined as starter nodes. On these nodes, which are typically front-end computers, parallel applications can be started, but no compute processes are placed on it. In the same manner nodes can be configured as pure compute nodes, starting of applications is not possible on such nodes.

The maximum number of processes per node is also configurable. By default, it's the number of cores available on a node. In addition, overbooking of nodes may be enabled.

Parts of the compute cluster can be exclusively reserved for a user or user group. Only this user or user group member is allowed to start processes on the reserved nodes. All other users are mapped to the remaining nodes.

Parallel applications: Applications that were parallelized by calls to the ParaStation MPI library are distributed to the available nodes according to the pattern described earlier. The start can be executed by means of calling the supplied *mpiexec* command. The application data is transferred via the networks and protocols supported by ParaStation MPI, refer to section 'Supported networks and communication protocols'.

Applications that are parallelized by means of 3rd party MPI libraries can be started via ParaStation MPI, too. The distribution of processes is performed as described above. Communication is carried out via the

corresponding mechanism implemented in the particular MPI library, e. g. TCP/IP. At present, the following 3rd party MPI libraries are supported:

- MPIch (with ch_p4),
- MPIch-GM,
- MPIch2,
- Mvapich2,
- Intel MPI.

The applications based on MPI1 compatible libraries are started by means of corresponding mpirun commands, e. g. *mpirun_chp4* for applications which were linked by MPIch with ch_p4, or *mpirun_chgm* for applications linked with MPIch-GM. Other MPI versions implementing the MPI2 standard and supporting the PMI interface, like MPIch2, MVapich2 or Intel MPI, may be started by the *mpiexec* command supplied with ParaStation MPI.

Serial applications: Applications that are not parallelized by means of MPI, can be started via ParaStation MPI, too. By calling *mpiexec,* an application is started on a suitable node according to the criteria mentioned above. The application itself can create new processes and threads on this node. Process pinning and memory binding is enforced.

Process monitoring, process termination: Each process created by ParaStation MPI on a compute node is permanently monitored. If one of these processes is terminated, e. g. due to a program error, or if a node is no longer available or accessible, all processes associated with this application on the respective nodes are terminated. All previously allocated resources will be freed. A corresponding return value is passed back to the calling process.

Executing processes can be listed at any time by means of the *psiadmin* command. The individual processes are identified by a cluster-wide, unique process ID. The system administrator or owner (starter) of an application can reliably terminate the application by means of the *kill* function of the *psiadmin* command at any time.

I/O forwarding: The default input (file descriptor 0), default output (1) and default error output (2) of each process started is linked back to the calling process via the management network. ParaStation MPI ensures that all outputs of all processes are collected centrally by a so-called logger process on the starter node, independent of the node on which the processes are executing. Subsequently, the logger process forwards the outputs.

By default, all input on file descriptor 0 of the logger process is sent to the first process, within MPI this process is assigned rank 0. The input can be assigned to another process by means of an environment variable.

If the starting process reads its default input from a (pseudo) terminal (pty), the I/O is routed via pseudo terminals for the processes started by ParaStation MPI.

Process pinning, memory binding: Within multi-core and multi-CPU systems, the ParaStation MPI process management allows pinning of each compute process to a particular core of a CPU using a configurable mapping table. In addition, memory binding may be enabled on SMP systems to automatically allocate the “nearest” memory available.

Hybrid applications parallelized with MPI and OpenMP are also supported by the process management. While starting up a process, more then one core may be assigned and subsequent created threads will be allowed to migrate to these cores. SMT is also supported.

Signals: Except of SIGKILL and SIGSTOP, all signals to the logger process are forwarded to all processes executing in a distributed manner. When terminating an interactive application by using SIGTERM, e.g. typing ^C, all processes belonging to this application are terminated on all nodes.

User administration: For application start-ups within the cluster, ParaStation MPI uses only user IDs. The user names are only resolved on the starter node. Therefore, users must be typically known on the front-end node, (complex) user management on the compute nodes is not necessary.

Job queuing and batch systems: ParaStation MPI includes a queuing facility to queue up application start-up requests, which currently can not be accomplished due to resource constrains. The only resources currently taken into account are cores. All start-up requests are queued in a first-come-first-serve manner.

If a more elaborated functionality is required, ParaStation MPI supports the batch queuing systems LSF, PBS-PRO and Torque. Corresponding environment variables are analyzed. Other queuing systems could be typically integrated using prologue scripts.

Parallel debugging: Using an option to the *mpiexec* command, each individual process of the job started with ParaStation MPI may be run under the control of the *gdb* debugger. All those processes may be controlled by the user using an easy-to-use, command-line driven interface, which allows debugging commands to be sent to all or a group of processes at once or to an individual process.

MPI library

The MPI library is based on MPIch2, version 1.2.1p1 Except for functions to spawn additional processes, all other functions defined within the MPI2 reference document including asynchronous send/receives are implemented. The abstract device interface layer (ADI2) of ParaStation MPI uses the *pscom* library and thus supports all interconnects and protocols described within the section `Supported networks and communication protocols´.

Small data buffers are directly transmitted to the receiving process, for a larger data volume, a rendezvous procedure is used. The buffer size at which the rendezvous starts can be influenced by an environment variable. Libraries with wrappers functions for different languages (*libfmpich*, *libpmpichcxx*, etc.) are contained.

ParaStation MPI supports application threads, which have been parallelized by means of the Linux *pthread* library. It must be noted, however, that the MPI operations provided by the *psmpi2* package are not thread-safe, e.g. several threads cannot call MPI functions simultaneously (MPI_THREAD_SERIALIZED). The MPI functions in the package *psmpi2-mt* are fully thread safe (MPI_THREAD_MULTIPLE). Both packages may be installed in parallel.

In addition, applications parallelized with OpenMP are supported, too.

All debugging and analysis tools provided by MPIch2 are available.

Supported networks and communication protocols

At present, ParaStation MPI supports communication using the *pscom* library via the following networks or communication paths:

- Fast or Gigabit Ethernet using TCP/IP,
- Fast or Gigabit Ethernet using an optimized ParaStation protocol *p4sock*,
- Shared memory for local communication within a node, especially for SMP systems,
- Infiniband using vapi kernel interface,
- Infiniband or 10GB Ethernet using the DAPL interface,
- Infiniband using unreliable datagram connections,
- Myrinet using the gm kernel driver,
- Quadrics QsNet^{II}.

The appropriate network is automatically selected: shared memory for communication within SMP nodes, Infiniband, Myrinet or optimized ParaStation protocol for communication using Ethernet. If a protocol is not available on application start-up, another available protocol will be selected automatically.

The communication networks and protocols to use can be pre-defined upon job start. If several Ethernet-based networks are available, an environment variable may be used to select the interface for transmitting the data and therefore select the network.

Via the underlying protocol(s), the *pscom* library ensures secure and reliable communication between all participating processes. Errors are automatically detected and eliminated, e. g. through repeated

transmission of data. The fragmentation/reassembly and flow control implemented allows the transmission of data buffers of any size.

TCP bypass

A TCP bypass module, based on the optimized ParaStation protocol for Ethernet (*p4sock*), is part of ParaStation MPI. Thus, data sent and received by an application using TCP sockets can be transparently rerouted to the optimized ParaStation protocol. This typically improves performance and lowers system load.

Kernel modules

The optimized ParaStation protocol *p4sock* for Ethernet and the TCP bypass are implemented as Linux kernel modules. If configured, these modules are automatically loaded at start-up of the *psid*.

In addition, modified versions of the e1000 (e1000.ko) and bcm5700 (bcm5700.ko) network drivers are part of ParaStation MPI. In conjunction with the respective glue-modules (e1000_glue.ko or bcm5700_glue.ko), even better results are achieved using Gigabit Ethernet. The source code of these modified drivers comes with the *pscom* package.

User programs

ParaStation MPI comprises, beside others, the following commands and programs:

Program:	*Description:*
psid	ParaStation daemon
psiadmin	central command-line-based administration command
mpiexec	start program for serial and parallel applications
mpirun_chp4, mpirun_chgm, mpirun_openib, mpirun_ipath	start program for applications linked with MPIch, MPIch-gm, MVapich or Pathscale MPI.
psh	parallel shell
pscp	parallel copy tool
test_nodes	diagnostic tool for communication layer

Supported compilers and languages:

The following table lists the compilers and versions supported by ParaStation MPI for Linux x86_64:

	Compiler:			
Language:	*GNU*	*Pathscale*	*Intel*	*Portland Group*
C	gcc	pathcc	icc	pgcc
C++	gcc	pathCC	icc	pgcc
Fortran77	f77	pathf90	ifort	
Fortran90	gfortran	pathf90	ifort	

Versions for different compilers may be installed and used in parallel. All typical compiler frontend scripts like *mpicc*, *mpif77*, *mpif90* and *mpicxx* are available for all different compiler flavors.

Versions for Linux IA32, IA64 and ppc[1] are also available.

[1]Depending on the underlying communication hardware and protocol, not all networks and communication protocols are supported on every system architecture.

Scalability and Limitations

Within ParaStation MPI there are no limitations with respect to the number of processes per application and the number of simultaneously executing processes per compute node. The decisive factors for these are the available system resources such as, for example, the available memory and configured limits, like maximum number of processes per node.

ParaStation MPI is tested with jobs running on 3200 nodes in parallel and sizes up to 25,600 processes per job. No limiting factors are foreseeable within ParaStation MPI.

Documentation

Documentation is available as separate, installable packages called *psdoc* and *psmpi2-doc*. Documentation contained includes:

- ParaStation MPI Users Guide (English, PDF and HTML format)
- ParaStation MPI Administrators Guide (English, PDF and HTML format)
- Manual pages with descriptions of ParaStation MPI and MPIch commands, functions, and environment variables: (English, PDF, HTML and groff format)

Installation prerequisites

To install and operate ParaStation MPI, the following prerequisites must be met:

Supported hardware architectures and operating systems:

ParaStation MPI is supported on all major Linux distributions and hardware platforms:

Architecture:	*Linux distributions:*
Intel IA32 and AMD Athlon	openSuSE, SLES, RHES, Fedora, CentOS
Intel IA64 (Itanium)	SLES, Redhat AS,
AMD x86_64 (Opteron)	openSuSE, SLES, RHES, Fedora, CentOS
IBM Power5	SLES 9

Supported are single-processor nodes as well as SMP nodes. SMT is also supported.

Network: For management tasks, a UDP connection must be established between all compute nodes. The operating system must support the installed network hardware, the corresponding low level drivers are also used by ParaStation MPI.

Management data and application data are separated and may be transmitted via different networks. For a list of supported data networks refer to section ‘Supported networks and communication protocols’.

Free disk space: The installation of all ParaStation MPI packages requires approximately 100MB free disk space per node. Administrator privileges are required for installation. The software can be exported from a file sever via NFS or Lustre to all cluster nodes.

Media

All ParaStation MPI modules are available in binary format as pre-compiled RPM packages as well as source RPM packages ready for compilation and installation on the installation system.

The packages are available in the download area of ParTec Cluster Competence Center’s web server www.par-tec.com.

License

ParaStation MPI is not freeware, although it is available in source form! Details can be found in the ParaStation license agreement on www.par-tec.com

Support

After signing a support contract, support for all packages is granted for the agreed period of time. The maximum response time is one working day. Support is performed by telephone, email, and/or remote login. Onsite support at the installation site is not included.

The support comprises all *ParaStationV5* components as well as the open source software utilized (if applicable). Other open source software tools that have been provided free of charge are only supported if resources are available, a general claim cannot be advanced on the basis of this support agreement.

Scope of delivery

ParaStation MPI comprises the following components:

- Software packages in the download area of ParTec Cluster Competence Center's web site,
- Documentation (ParaStation MPI Users' Guide and ParaStation MPI Administrator's Guide) in electronic format,
- Support as agreed in the support contract.

Copyright

Further information

For further information about ParaStation MPI visit http://www.par-tec.com or send an email to sales@par-tec.com.