CST Studio Suite® 2022

MPI Computing Guide

3DEXPERIENCE®

DASSAULT SYSTEMES | The 3DEXPERIENCE® Company

Contents

1 **Introduction** 4

2 **Supported Solvers and Features** 6

3 **MPI Processes and Roles** 7

4 **Interconnect** 9
- 4.1 Interconnect Topology 9
- 4.2 Network Protocols 12
- 4.3 Firewall Configuration 15

5 **Operating System Support** 16

6 **Installation and Configuration of MPI Computing** 17
- 6.1 Installation of CST Studio Suite® on Frontend Node and Compute Nodes 17
- 6.2 Installation of MPI Process Management Service (HYDRA) 18

7 **Update of CST Studio Suite® Installation** 23

8 **Shared Storage** 24

9 **Combined Hardware Acceleration and MPI Computing** 25

10 **Starting a Simulation Using MPI Computing** 26
- 10.1 Starting Simulations Interactively 26
- 10.2 Starting Simulations in Batch Mode 26
- 10.3 Use MPI Automatically for Huge Projects 30

11 **Job Scheduling System Integration** 32

12 **Assignment of Hardware Resources to Compute Nodes** 33
- 12.1 Full Computer Assignment (Compute Node=Physical Computer) 34
- 12.2 NUMA Node Assignment (Compute Node=NUMA Node) 34
- 12.3 CPU Core Assignment (Compute Node=CPU Core) 35

A **Setup of Passwordless SSH** 36
- A.1 Linux Compute Nodes 36
- A.2 From a Windows Frontend Node to Linux Compute Nodes 37

B **Setting Environment Variables** 39
- B.1 Setting System-Wide Environment Variables on Windows 39
- B.2 Setting System-Wide Environment Variables on Linux 39

C Setting Up and Testing an Infiniband Network **40**

D Network and Cluster Checks **41**

D.1 Check Name Resolution 41
D.2 Check Program File Access 41
D.3 Test MPI Communication 42
D.4 CST SystemCheck 43

1 Introduction

MPI[1] computing allows to split the computational workload for a simulation among computational resources[2]. These computational resources can either be tightly connected within a single computer such that they share access to the same RAM (shared memory environment) or can be distributed among different computers in a computer cluster[3] (distributed memory environment). Unlike Distributed Computing which can only assign independent computational tasks to a set of computational resources (e.g., simulations for different parameter sets in a parameter sweep), MPI computing can assign computational tasks which are not independent of each other (e.g., field computations in different areas of a 3D model) to computational resources, such that these resources work on the tasks in parallel.

Note: Setting up a computational environment for MPI Computing requires knowledge of operating systems and network configuration procedures. Thus, the target audience of this manual are not CST Studio Suite® end users but IT-professionals.

The set of computational resources to which a computational task is assigned paired with the MPI software functionality will be referred to as "Compute Nodes". From a software perspective, a Compute Node is a role (i.e., a set of functionality) which can be assigned to one or more physical computers in the context of a simulation using MPI computing. Throughout this document, it'll be made clear in the different sections whether the explanation refers to the set of functionality of a Compute Node from a software perspective or to the physical hardware (computational resources) which are assigned to the Compute Node. So please note that the computational resources assigned to a Compute Node can be a physical computer including all its computational resources, but it is also possible that just a set of CPU cores or a CPU device/socket is assigned to a Compute Node. Section 11 contains more information about the performance and licensing aspects of the assignment of physical hardware resources to Compute Nodes in an MPI simulation.

The split of the computational workload and data of a simulation allows the efficient handling of large models which require too much resources to be handled on a normal workstation or server system (see figure 1, the amount of data each compute node needs to handle is significantly smaller than the amount of data of the complete model).

Furthermore, the Compute Nodes work in parallel and, thus, a performance improvement can be achieved when increasing the number of Compute Nodes (assuming that

[1]The "Message Passing Interface" (MPI) describes a programming interface used for multi-process parallelization, which has become a de facto standard and because of its importance this type of multi-process parallel computing is often referred to as "MPI computing".

[2]The term "computational resource" refers to the hardware (CPU devices, RAM memory, Hardware Accelerator devices) of one or more computers.

[3]A computer cluster is a set of computers connected via a (fast) interconnection network and used as a single "supercomputer".

each Compute Node has the same hardware resources assigned) if the hardware is well chosen and the configuration is appropriate.

MPI Computing can also help to use computers efficiently, which have a shared memory architecture with strong NUMA[4] properties (e.g., quad- and octa-socket systems or machines of the SGI UltraViolet series).

Often the Compute Nodes are separated from the computational environment used to work interactively on a graphical desktop (e.g., the workstation of a user). The computational device/computer used to work interactively on a desktop environment will be referred to as "User Workstation" within the manual.

This manual will help to configure and test an existing cluster system or a shared memory environment for operation with CST Studio Suite®, such that simulations using MPI Computing can be performed with best possible performance.

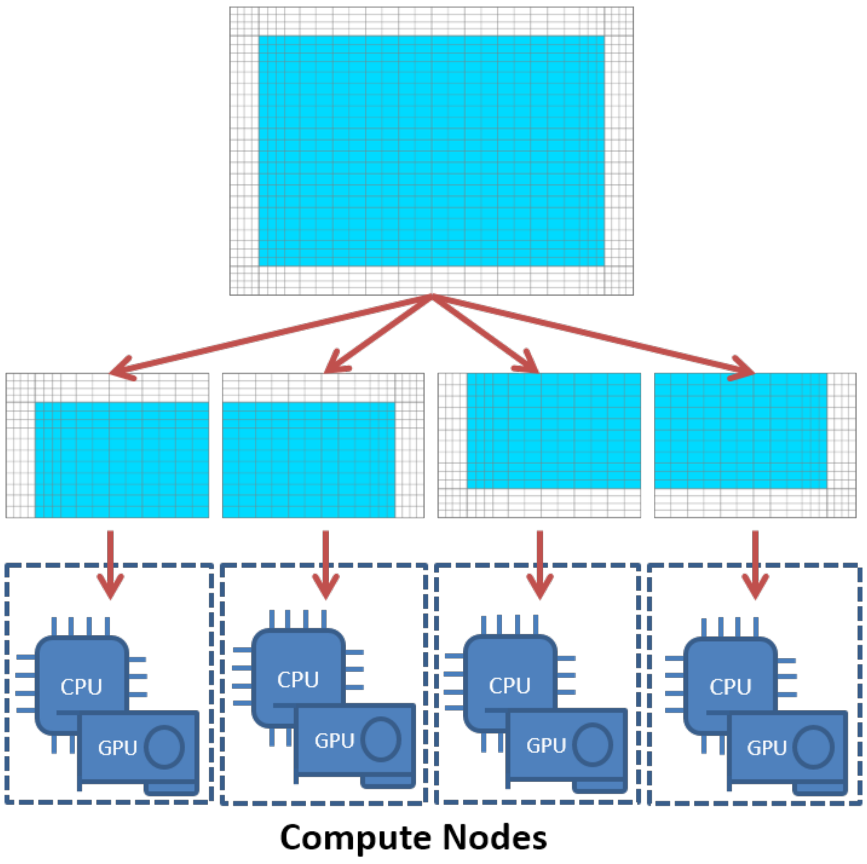


Figure 1: Assignment of computational workload and data of a simulation model to Compute Nodes.

[4]Non-Uniform Memory Access (NUMA) describes an architectural and performance property of the memory subsystem of a computer. On a computer with NUMA architecture the latency as well as the bandwidth when accessing system memory depends on the location of accessed memory region.

2 Supported Solvers and Features

Table 1 provides an overview of the solvers for which MPI Computing is supported. Solvers which are not listed in the table do not support MPI Computing.

Solver	Limitations
High Frequency: Time Domain Solver	Hexahedral TLM mesh is not supported. Periodic boundaries are not supported. Subgridding is not supported.
High Frequency: Frequency Domain Solver	Fast reduced order model sweep and variable order are not supported. Hexahedral mesh is not supported. Periodic boundaries and PML are not supported. Materials like ferrites or spatially varying materials are not supported. Plane wave excitation and near field sources are not supported. Protected projects are not supported.
High Frequency: Integral Equation Solver	MLFMM and direct MoM can use MPI Computing. ACA does not support MPI Computing.
Particles: Wakefield Solver	Periodic boundaries are not supported.

Table 1: Solvers of CST Studio Suite® supporting MPI Computing

3 MPI Processes and Roles

A simulation using MPI Computing always includes the processes as shown in figure 2. The arrows are showing bi-directional communication requirements between the different processes. Each group of processes (as indicated by the dashed frames) can run on a different computer (i.e., on computational resources which don't share a common memory address space) as long as the bi-directional communication links between the processes can be provided by the network topology and configuration (for more details see the section 4).

The terms "Frontend Node" and "Compute Node" should be understood in this context as roles (i.e., set of functionality) which are assigned to computational resources. Please note that the roles are not mutual exclusive, i.e., the Frontend Node role and the Compute Node role can be assigned to the same computational resources. However, for performance reasons, it's strongly recommended that the computational resources assigned to different Compute Nodes are mutually exclusive (see section 12 for more information about assigning hardware resources to Compute Nodes).

The **Compute Nodes** will run the compute and memory intensive parts of a simulation. Additionally, the processes running on the Compute Nodes need to exchange a large amount of data during a computation (indicated by the link labeled "MPI communication" in figure 2). This implies that there are great demands on the interconnect used to handle this data exchange with regards to bandwidth and latency. The **MPI Communication** can use different HPC interconnect technologies and various network stacks in order to improve performance. The other communication links are established using the Ethernet (TCP/IP) protocol (see also section 4 for more details).

The computational resources of a **Frontend Node** run the graphical CST Studio Suite® front-end, which starts the compute intensive processes on computational resources of the Compute Nodes. Please note that the CST Studio Suite® front-end always needs to start a MPI simulation as it orchestrates the data transfer to Compute Nodes and starts the pre-and post-processing tasks required by the simulation as well, i.e., a direct start of the solver processes without the CST Studio Suite® front-end is technically impossible. If computations must be started in a non-interactive batch mode, which is commonly the case on access restricted clusters managed by a job scheduling/resource management system, the front-end is still used to start the simulation (see section 10.2 for more information).

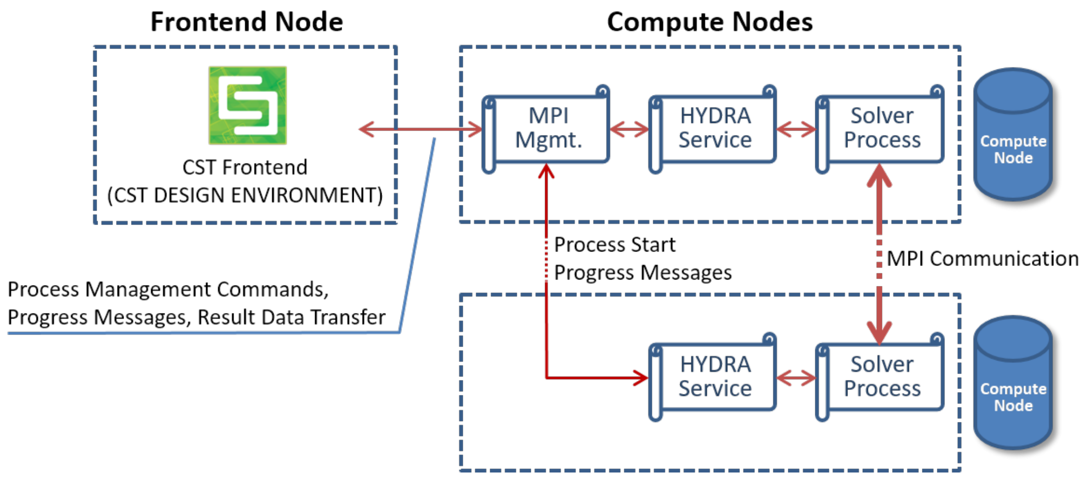


Figure 2: Processes taking part in an MPI simulation (simplified). "Frontend Node" and "Compute Node" should be understood as roles which can be assigned to hardware resources in this context. The roles are not mutually exclusive, i.e., the "Frontend Node" role and the "Compute Node" role can be assigned to the same hardware resources. The links between the different processes symbolize bi-directional data exchange between the processes.

4 Interconnect

As mentioned in section 3, the performance of the data exchange/communication of the processes of different compute nodes, running potentially on different computers in a cluster, is crucial in order to achieve a good performance of the simulation.

4.1 Interconnect Topology

In this section it is discussed how the Compute Node and the Frontend Node role described in the previous section can be assigned to computational resources with regards to the network topology. This covers the requirements regarding the data exchange between the different software components as listed in figure 2.

Note: Please note that there are constraints regarding the assignment of the Frontend Node role if the cluster hardware resources are managed by a scheduling system (see section 11 for more information).

4.1.1 Basic Network Topology

The most basic network configuration for a working MPI Computing setup which spans across multiple computers is shown in figure 3. An Ethernet (TCP/IP) network connects

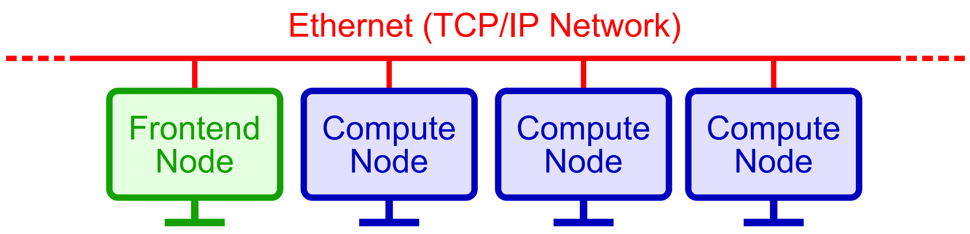


Figure 3: Basic Network Topology for MPI Computing.

the computational resources to which the Compute Node role is assigned.

MPI Computing is very sensitive to a correctly configured TCP/IP network. The network setup needs to fulfill the following minimum requirements:

1. An unique IPv4 address is assigned to each computer which is part of the setup.
2. Name resolution (i.e., mapping of computer names to IPv4 addresses) needs to be properly configured either using a DNS or a static configuration.
3. If any firewalls are active either on the computers themselves or as a separate network component, firewall rules which allow bi-directional data exchange between the computers taking part in the computation need to be applied (see section 4.3).

If in doubt whether the network is configured correctly, please refer to the section D to check the Ethernet network setup.

Note: As the Ethernet protocol introduces significant latency when exchanging data, it's strongly recommended to use an HPC interconnect (Infiniband or Omni-Path) between the Compute Nodes to achieve the best performance as described in the following sections.

4.1.2 High-Performance Interconnect Topologies

Although the basic network topology as described in the previous section allows for a working MPI Computing setup, it is recommended to use a dedicated high speed interconnect, i.e., Infiniband or Omni-Path, for the MPI Communication since this part is very sensitive to the network performance as mentioned in section 3. A configuration such as shown in figure 4 enables the processes running on the Compute Nodes to utilize the HPC interconnect instead of using the Ethernet network for the performance critical MPI Communication. It is also possible to assign the Frontend Node role to a

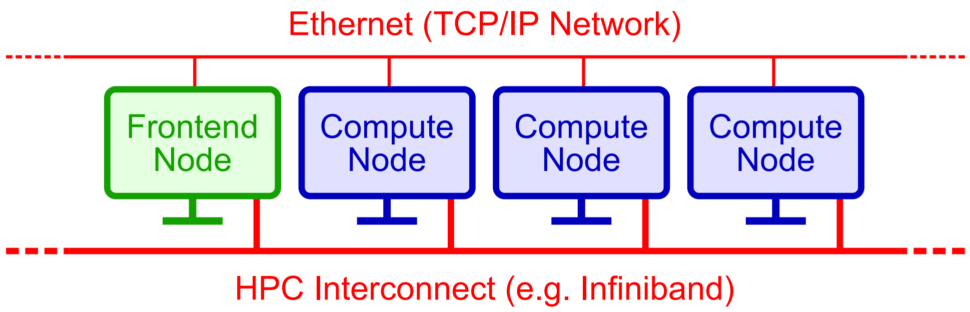


Figure 4: High Performance Interconnect Topology 1. All computational resources have access to a high-performance interconnect. Regardless of the HPC Interconnect a properly configured Ethernet network is required as well.

computational resource which has only a TCP/IP connection to the computational resources which have the Compute Node role assigned such as shown in Figure 5. This configuration is typical if the Frontend Node role is assigned to a user workstation for example which has typically no connection to the HPC interconnect of the cluster. Although this is possible it should be mentioned that significant data transfer can happen between the Frontend Node component and the Compute Nodes after a simulation has completed, because result data is transferred from the Compute Nodes to the Frontend Node component. This implies that the network link between the Frontend Node component and the Compute Nodes, although it is not so much utilized during the simulation itself, can become a bottleneck if a large amount of result data is generated by the Compute Nodes (e.g., because many 3D field monitors are defined for the simulation model). Thus, it should be ensured that the network link between the Frontend Node component and the Compute Node components offers a reasonable bandwidth unless a shared storage configuration is used to prevent the data transfer between Frontend Node and Compute Nodes completely (see section 8, to learn more about this possibility).

Note: While a Gigabit Ethernet connection is often sufficient to provide a decent bandwidth between the Frontend Node and the Compute Nodes, a WAN network link or a Fast Ethernet link is normally not sufficient.

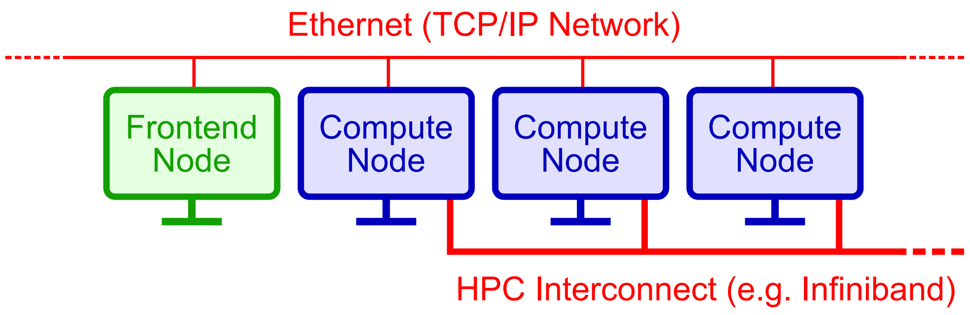


Figure 5: High Performance Interconnect Topology 2. The Frontend Node has only TCP/IP connectivity to the Compute Nodes.

4.1.3 Isolated Cluster Network

In some cluster configurations, the computational resources of the cluster are not directly accessible from the enterprise network, i.e., a cluster head node exists, which acts as a gateway to the computational resources of the cluster to which the Compute Node role is assigned (see figure 6). As explained in section 3, the Frontend

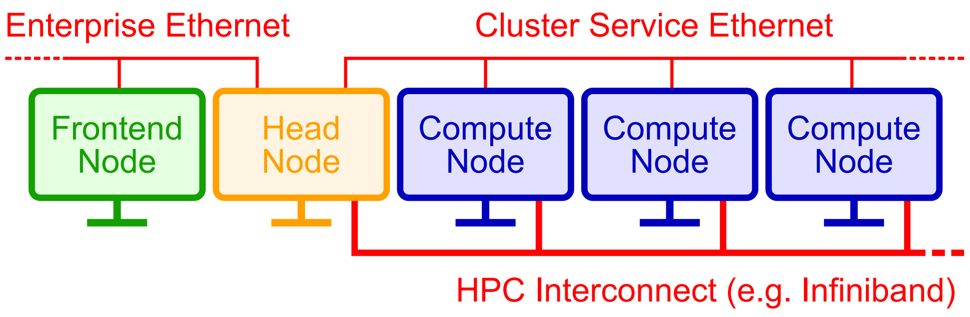


Figure 6: High Performance Interconnect Topology 3. The MPI Frontend Node has no direct network connection to the cluster compute nodes.

Node component and the Compute Node components use bi-directional network communication during a MPI simulation. Therefore, either routing between the different Ethernet subnets needs to be configured properly[5] on the cluster head node/gateway, or the Frontend Node role must be assigned to a computational resource which has a direct connection to the Ethernet network of the Compute Nodes.

This configuration (figure 6) will work if and only if bidirectional routing between the cluster service network and the enterprise network is configured such that the MPI

[5] A proper configuration in this context means that bi-directional routing between the cluster service network and the enterprise network is configured such that the Compute Nodes are visible on the enterprise network and the Frontend Node is visible for the Compute Nodes.

Compute Nodes are visible on the enterprise network and the MPI Frontend Node is visible for the cluster compute nodes. Otherwise, the Frontend Node role needs to be assigned to a computer with direct access to the TCP/IP network of the cluster.

4.2 Network Protocols

A network protocol is used to handle the data transfer from one process to another. The three different communication links between the MPI components as shown in figure 2 (Frontend Node to Compute Node, Service Communication, and MPI Communication between the Compute Nodes) support the network protocols as listed in table 2.

Communication Link	Supported Protocols
MPI Frontend Node <-> MPI Compute Node	Ethernet (TCP/IP)
Service Communication	Ethernet (TCP/IP)
MPI Communication	Ethernet (TCP/IP) Infiniband (OFA, DAPL) Omni-Path (OFI) Shared Memory (SHM)

Table 2: Supported network protocols

4.2.1 Configuration of the Network Protocol Used for MPI Communication

CST Studio Suite® tries to use the best (fastest) option available for the MPI Communication. However, the desired protocol can be specified manually as well. The automatic protocol selection can be overwritten by the environment variable `I_MPI_FABRICS`. The environment variable needs to be set on the computers which have either the Compute Node or the Frontend Node role assigned. The environment variable needs to be visible in the context of the MPI processes. If it is defined as a system wide environment variable this will be sufficient. The environment variable can have the values listed in table 3.

4.2.2 Selection of TCP/IP Network for MPI Communication

If the TCP/IP network protocol is used for MPI Communication (i.e., if the `I_MPI_FABRICS` environment variable is set to `tcp`, see section 4.2.1), there might be more than one TCP/IP network which is accessible to the Compute Nodes. See figure 4 as an example for a network configuration that allows the use of the TCP/IP network protocol on two different networks (on the Ethernet network and on the HPC Interconnect). The `I_MPI_TCP_NETMASK` environment variable can be used to bind the MPI

Value	Network Protocol Description
shm	The Shared Memory (SHM) protocol can only be used in a shared memory environment, i.e., for MPI Communication between Compute Nodes which have access to the same RAM memory. This is the case for processes running on the same physical computer or for special shared memory architectures like the SGI UltraViolet system where a common memory address space is emulated on kernel level. *Available on Linux and Windows.*
dapl	Direct Access Programming Library (DAPL) is an HPC network protocol which can be used e.g. with Infiniband hardware. It provides low latency and high network bandwidth and can be recommended for MPI Communication. *Available on Linux and Windows.*
tcp	TCP/IP (Ethernet) network protocol is the least favorable option as it introduces significant latency and often provides a low bandwidth. *Available on Linux and Windows.*
tmi	Tag Matching Interface (TMI), a network protocol which can be used e.g. with Intel True Scale Fabric and Intel Omni-Path hardware. *Available on Linux only.*
ofa	The network protocol provided by the OpenFabrics Alliance (OFA) is an HPC network protocol which can be used with Infiniband hardware. It provides low latency and high network bandwidth and can be recommended for MPI Communication. *Available on Linux only.*
ofi	OpenFabrics Interface (OFI) is an HPC network protocol which can be used e.g. with Intel True Scale Fabric and Intel Omni-Path hardware. It provides low latency and high network bandwidth and can be recommended for MPI Communication. *Available on Linux only.*

Table 3: Possible values for the environment variable `I_MPI_FABRICS`.

Communication to a specific TCP/IP network or a specific network interface as shown in table 4. The environment variable needs to be set on the computers which have either the Compute Node or the Frontend Node role assigned. The environment variable needs to be visible in the context of the MPI processes. If it is defined as a system wide environment variable this will be sufficient.

Value	Description
`ib`	Use an "IP over IB" interface (only if an Infiniband network is available).
`ib<n>`	Use a specific "IP over IB" interface (only if an Infiniband network is available). Example: `ib0`
`eth`	Use any available Ethernet network interface. This is the default value.
`<netmask>`	The TCP/IP netmask of the subnet which should be used for MPI Communication. Example: `192.168.0.0/16`

Table 4: Possible values of the `I_MPI_TCP_NETMASK` environment variable.

4.2.3 Selection of TCP/IP Network for MPI Frontend Node/MPI Compute Node Data Exchange

The data exchange between the Frontend Node component and the Compute Node components uses the TCP/IP network protocol. As there might be more than one TCP/IP network links which can be chosen for this data exchange, a TCP/IP netmask can be defined in order to select the desired network for this data exchange. The `CST_MPI_PREFERRED_SUBNET` environment variable can be used to bind the data exchange between the Frontend Node and the Compute Nodes to a specific TCP/IP network. The environment variable needs to be set on the machines which have either the Compute Node or the Frontend Node role assigned. The environment variable needs to be visible in the context of the MPI processes. If it is defined as a system wide environment variable this will be sufficient. See figure 4 as an example for a network configuration that allows the use of two TCP/IP networks.

Example: `CST_MPI_PREFERRED_SUBNET=192.168.0.0/24`

4.3 Firewall Configuration

As the Frontend Node component and the Compute Node components need to exchange data over the network, a proper firewall configuration is required in order to allow this data exchange. Although the simplest possible way to prevent all firewall related issues is to switch off the firewall on the computers which have the Frontend Node and/or the Compute Node role assigned, this might not be possible on some systems because of security restrictions.

The commands required to configure the firewall properly depend on the specific firewall implementation and, thus, this manual will only list the required ports which need to be opened to allow TCP/IP data exchange.

The following port ranges are used by the MPI system and need to be opened within the network which handles the MPI Communication between the Compute Nodes as well as for the connection between Frontend Node and Compute Nodes: 8678, 20000 ... 20500, 30000 ... 65535.

Note: On Windows it is not necessary to open any ports manually for the local Windows firewall on Frontend Node and Compute Nodes as the MPI installation will automatically add exceptions to the Windows firewall to allow the executables taking part in the simulation to exchange data.

5 Operating System Support

This section contains information about the operating system support for the Frontend Node and the Compute Node components. This section contains only high level information, i.e., it will not be discussed whether a certain version of Linux or Windows is supported, but only which operating system (OS) family (Linux or Windows) is supported for Frontend Node and Compute Nodes.

All Compute Nodes need to share the same OS family (Windows or Linux) and, ideally, the same version and patch level of the OS. The Frontend Node can run a different operating system than the Compute Nodes. Supported configurations are shown in table 5.

	Frontend Node	**Compute Node**
Configuration 1	Windows	Windows
Configuration 2	Windows	Linux
Configuration 3	Linux	Linux

Table 5: Supported configurations for the OS family of Frontend Node and Compute Node.

General information about supported operating systems for CST Studio Suite® is available on the CST website.

6 Installation and Configuration of MPI Computing

Preparing a computational environment such that CST Studio Suite® can be used with MPI computing requires a few installation and configuration steps. This section contains information about the required steps to enable MPI computing for all three configurations listed in table 5.

6.1 Installation of CST Studio Suite® on Frontend Node and Compute Nodes

The first step of an installation is to make the program files of CST Studio Suite® available on the Frontend Node and all Compute Nodes. Depending on the operating system of the Frontend Node and Compute Nodes, either the Linux version or the Windows version of CST Studio Suite® should be installed. The installation of CST Studio Suite® can be an installation which resides on a local disk, i.e., the installation needs to be performed multiple times such that the CST Studio Suite® program files are available on the local disk of every computer which has either the Frontend Node role or the Compute Node role in the setup, or all computers can share a single installation. However, especially in a cluster environment a shared installation is recommended as it simplifies the installation and update process. The following two sections describe both setup procedures. In any case it is recommended to use the same installation location on all computers which have the compute node role.

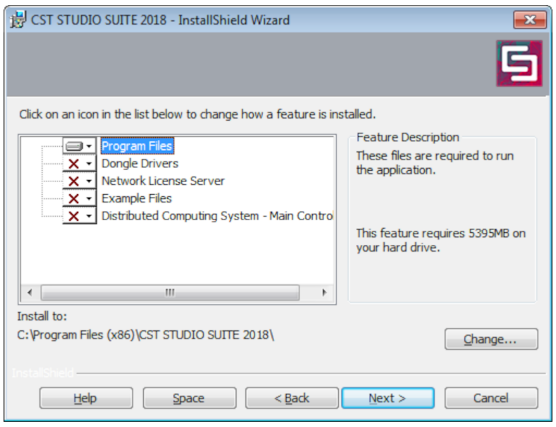
6.1.1 Local Installation of CST Studio Suite®

When using a local installation, make sure that at least the feature set shown in figure 7 is installed on computers which have the Frontend Node role or the Compute Node role. Installing any of the other features available in the installation programs is possible and won't be in conflict with the MPI computing setup.

6.1.2 Shared Installation of CST Studio Suite®

On computer cluster systems an installation shared by all computers which are part of the cluster is often preferred over a local installation as described in section 6.1.1 as it simplifies the installation and update procedure. Setting up a shared installation requires a storage device shared by all computers which have the Compute Node role. This can be either a dedicated NAS or, in simplest case, a network share provided by one of the computers in the cluster (e.g., Windows share, NFS share). The CST Studio Suite® installer needs to be executed just once to install the feature set shown in figure 7 on the shared storage device. Computers accessing the shared installation require read and execute permissions on the installation folder.

On all computers which access a shared installation of CST Studio Suite® on Windows, a special installation program (`install-cluster-node.bat`) found in the "ClusterUtil-


CST STUDIO SUITE 2018 - InstallShield Wizard

Click on an icon in the list below to change how a feature is installed.

- Program Files
- ✗ Dongle Drivers
- ✗ Network License Server
- ✗ Example Files
- ✗ Distributed Computing System - Main Contro

Feature Description

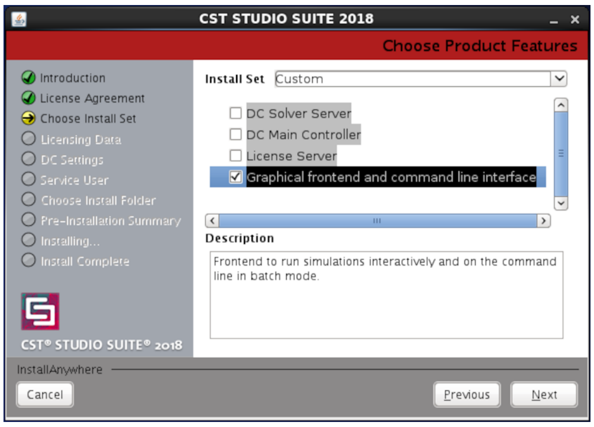
These files are required to run the application.

This feature requires 5395MB on your hard drive.

Install to:

C:\Program Files (x86)\CST STUDIO SUITE 2018\

(a) Windows installer.


CST STUDIO SUITE 2018

Choose Product Features

Install Set: Custom

- ☐ DC Solver Server
- ☐ DC Main Controller
- ☐ License Server
- ☑ Graphical frontend and command line interface

Description

Frontend to run simulations interactively and on the command line in batch mode.

(b) Linux installer.

Figure 7: (a) Shows the required feature set on Windows. (b) Shows the required feature set on Linux.

ities" folder of the CST Studio Suite® installation needs to be executed once in order to create some required registry settings and install prerequisite packages which are required by the program files of CST Studio Suite®. This installer can be started in non-interactive mode. Please refer to the `readme-setup-cluster-node.txt` file found in the "ClusterUtilities" folder which contains information about the installation procedure and the command line interface of the installer.

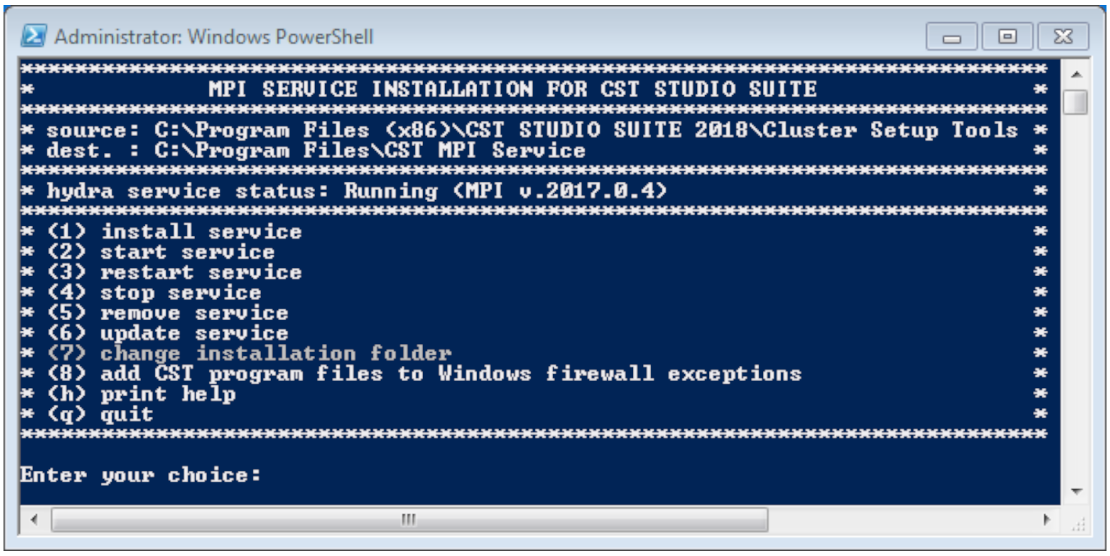
On computers which access a shared installation of CST Studio Suite® on Linux, no additional steps are required. However, it is recommended to start the `cst_system_check` tool found in the installation folder at least once on each computer using the shared installation to make sure that all system packages and tools required to use the CST Studio Suite® program files are available.

6.2 Installation of MPI Process Management Service (HYDRA)

On computers which have the Compute Node role and which are running the Windows operating system, a system service called HYDRA needs to be installed (i.e., this step can be skipped on Compute Nodes running the Linux operating system). This system service is required to handle the start of processes in context of the MPI system. Please execute the script `install-mpi-service.bat` found in the folder "ClusterUtilities" of the CST Studio Suite® installation and select the first option (install service) in the list (see figure 8). The service can be removed using option 5 ("remove service"). It will not be removed automatically by the CST Studio Suite® uninstall program.

6.2.1 User Authentication

MPI computing involves the start of processes on different computers in a network. All processes need to be assigned to a certain user account when started. This determines



Administrator: Windows PowerShell

```
*****************************************************************************
*              MPI SERVICE INSTALLATION FOR CST STUDIO SUITE                *
*****************************************************************************
* source: C:\Program Files (x86)\CST STUDIO SUITE 2018\Cluster Setup Tools *
* dest. : C:\Program Files\CST MPI Service                                  *
*****************************************************************************
* hydra service status: Running (MPI v.2017.0.4)                            *
*****************************************************************************
* (1) install service                                                       *
* (2) start service                                                         *
* (3) restart service                                                       *
* (4) stop service                                                          *
* (5) remove service                                                        *
* (6) update service                                                        *
* (7) change installation folder                                            *
* (8) add CST program files to Windows firewall exceptions                  *
* (h) print help                                                            *
* (q) quit                                                                  *
*****************************************************************************

Enter your choice:
```

Figure 8: Options of the installation program of the HYDRA service.

for example the file access permissions of the process or the access to other system resources. Depending on the operating system used for Frontend Node and Compute Nodes there are different possibilities to perform user authentication and, thus, to configure the security context of a process.

6.2.2 Pure Linux Environment

If both the Frontend Node and the Compute Nodes are running Linux, the MPI process startup mechanism requires passwordless remote shell access to all compute nodes taking part in the simulation. The environment variable `I_MPI_HYDRA_BOOTSTRAP` can be used to specify the remote shell tool. It can have the values shown in table 6.

6.2.3 Mixed Windows/Linux Environment

If the Frontend Node is running a Windows operating system and the Compute Nodes are running a Linux operating system, the passwordless remote shell setup as described in section 6.2.2 needs to be in place on the Compute Nodes. Please note that only ssh or rsh can be used in this case and it is not possible to start the simulation using a scheduling system (see section 11).

In addition to the setup of passwordless remote shell access between the Compute Nodes, passwordless ssh access from the Frontend Node to the Compute Nodes needs to be configured. The CST Studio Suite® installation contains ssh client programs which will be used for this purpose. Please refer to section A.1 for the configuration of the passwordless ssh login from a Windows Frontend Node to Linux Compute Nodes.

Value	Description
ssh	Use ssh to establish connection to Compute Nodes. Passwordless ssh login needs to be configured properly for the user account used to start the simulation (see section A.1). This is the default value.
rsh	Use rsh to establish connection to Compute Nodes. Passwordless rsh login needs to be configured properly for the user account used to start the simulation.
pdsh	Use pdsh to establish connection to Compute Nodes.
pbsdsh	Use the pbsdsh command to establish connection to Compute Nodes. The pbsdsh tool is provided by the Torque scheduling system. This option is only valid if the MPI simulation is started in context of the Torque scheduling system (see section 11).
slurm	Use the srun command to establish connection to Compute Nodes. The srun tool is provided by the SLURM scheduling system. This option is only valid if the MPI simulation is started in context of the SLURM scheduling system (see section 11).
lsf	Use the blaunch command to establish connection to Compute Nodes. The blaunch tool is provided by the LSF scheduling system. This option is only valid if the MPI simulation is started in context of the LSF scheduling system (see section 11).
sge	Use the qrsh command to establish connection to compute nodes. The qrsh tool is provided by the Univa Grid Engine (UGE) scheduling system. This option is only valid if the MPI simulation is started in context of the UGE scheduling system (see section 11)

Table 6: Values of the `I_MPI_HYDRA_BOOTSTRAP` environment variable which allows to select the remote shell tool used to establish connections to Compute Nodes.

6.2.4 Pure Windows Environment

If both, the Frontend Node and the Compute Nodes are running Windows, the MPI process startup mechanism uses a Windows service, the so-called HYDRA service, to access all compute nodes taking part in the simulation (see section 6.2). The environment variable `I_MPI_AUTH_METHOD` can be used to select the authentication mechanism used when starting processes on the Compute Nodes. It can have the values shown in table 7.

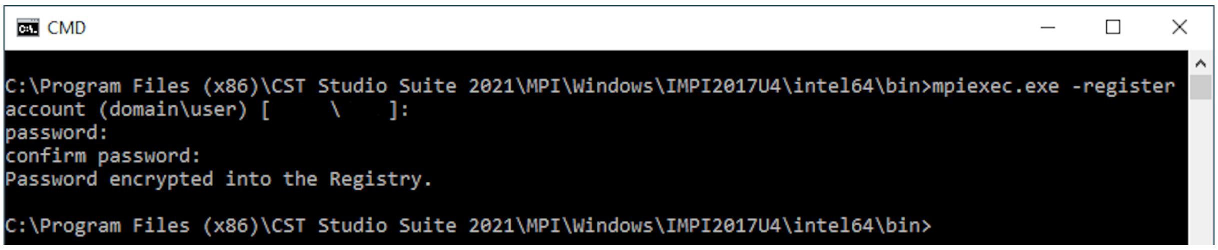
Value	Description
password	Store username and password used to start processes on remote nodes in the encrypted part of the Windows registry. This is the default value.
delegate	This option requires that the Frontend Node and the Compute Nodes are part of the same Active Directory Domain and that delegation has been activated on the domain controller.
impersonate	Use the limited domain-based authorization. This option requires that the Frontend Node and the Compute Nodes are part of the same Active Directory Domain. This option will NOT allow access to any network drive or mapped network drive for Compute Nodes.

Table 7: Values of the `I_MPI_AUTH_METHOD` environment variable which allows to select the user authentication method on Windows.

6.2.5 Password Authentication Method on Windows

In order to use the password authentication method, the password of a user account which exists and has the same credentials on all computers with the Compute Node role needs to be registered on the Frontend Node. The credentials are stored in the encrypted part of the Windows registry. Currently, the registration can be done only with the command line tool `mpiexec`. Starting "`mpiexec`" with the "`-register`" option will prompt for the user account and password which is then stored to the registry (see figure 9).

It's recommended to register domain accounts as this ensures that the user credentials are identical on all Compute Nodes. However, local accounts created on all computers taking part in a simulation with identical credentials can also be used. In this case the domain prefix of this account (which is the name of the local machine in case of a local account) needs to be removed when registering the account. For example, when registering the local account "mpi-user" the "`mpiexec -register`" command on a computer



```
C:\Program Files (x86)\CST Studio Suite 2021\MPI\Windows\IMPI2017U4\intel64\bin>mpiexec.exe -register
account (domain\user) [     \    ]:
password:
confirm password:
Password encrypted into the Registry.

C:\Program Files (x86)\CST Studio Suite 2021\MPI\Windows\IMPI2017U4\intel64\bin>
```

Figure 9: Registration of user credentials on Windows for the password authentication method.

with the name "comp001" it will suggest to register the account "comp001\mpi-user". Please don't use this suggested name but instead enter "mpi-user" without any prefix.

Note: The registered credentials are stored in the user registry, i.e., the registered credentials are only accessible by the user who did the registration. Any other user needs to perform the registration step as well.

7 Update of CST Studio Suite® Installation

The update procedure of CST Studio Suite® installations which have been configured to be used for MPI computing according to this manual can be updated as any normal CST Studio Suite® installation. It's important to know however, that all CST Studio Suite® installations taking part in a simulation using MPI computing must have the same version, i.e., if some Compute Nodes have service pack X installed while others have service pack Y installed, the setup won't work. If a shared installation is used (see section 6.1.2), only this shared installation needs to be updated. Otherwise, a service pack needs to be applied to all local installations on Compute Nodes and the Frontend Node taking part in a simulation.

Often the CST Studio Suite® front-end which triggers the download of service packs from the internet is never opened interactively on systems used for MPI computing, especially if they allow only batch mode operation under control of a job scheduling system. Thus, service packs need to be downloaded manually from the download section of the CST website. The downloaded service pack file (file extension ".sup") can be applied to a CST installation using the CST Update Manager which can be started interactively from the "Applications" menu of the operating system or in non-interactive mode on the command line. To install a service pack named "`<service-pack>.sup`" on the command line, the "`AutoUpdate`" program found in the CST Studio Suite® installation folder can be started with the following command line:

```
AutoUpdate --install "<service-pack>.sup"
```

8 Shared Storage

Certain computational environments provide access to an HPC storage device which computers having the front end node or the compute node role can access (see Figure 10). This can range from a simple NFS share to a dedicated, distributed HPC filesystem attached to the computational resources by a separate storage area network (SAN). A shared storage allows to reduce the data transfer from the compute nodes to the front end node to a minimum. Instead of transferring data between the different components explicitly, CST Studio Suite® uses the knowledge that files created on a compute node are visible on the front end node. However, during a simulation run significant I/O workload will occur on the shared storage, i.e., when using this approach it should be tested, whether the storage device can handle the I/O workload sufficiently well.

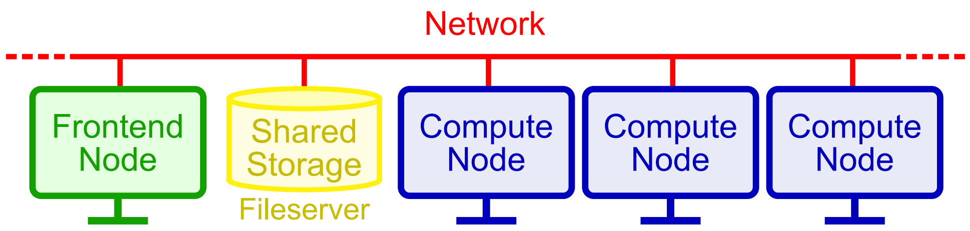


Figure 10: Shared storage configuration. The shared storage device must be accessible from the front end node as well as from the compute nodes.

Currently, the option to switch off the network transfer of result data and rely on a shared storage for data exchange can be activated by a command line option. This is a command line option of the CST Studio Suite® front end, i.e., `CST DESIGN ENVIRONMENT.exe` on Windows and `cst_design_environment` on Linux:

```
-shared-dir
```

The shared directory feature is available if front end and compute nodes are running the same operating system, Linux or Windows. In this case, CST Studio Suite® will assume that the model file is located in a directory through the same path on front end node and all compute nodes.

For MPI simulations with mixed operating systems, that means Windows front end and Linux compute nodes, the shared directory cannot be used and will be deactivated if the corresponding option is set. Instead, standard upload and download mechanics are used.

Note: Simulations may cause an I/O pattern which is challenging for a storage device in terms of the number of files created as well as in terms of the total amount of data created. Thus, performance may be significantly deteriorated when using a shared storage system which is not designed to handle HPC I/O workloads. **Additionally, the shared storage needs to allow file locking.**

9 Combined Hardware Acceleration and MPI Computing

If Hardware Acceleration should be used to speed up the computations of the Compute Nodes, please refer to the GPU Computing Guide, respectively, to learn more about the configuration of the accelerator hardware and for any special configuration required if processes accessing the accelerator hardware are using MPI computing.

10 Starting a Simulation Using MPI Computing

Once the configuration steps described in the previous sections of this documents have been completed on the computers with Frontend Node role and Compute Node role, a simulation using MPI computing can be started. This section explains how to start a simulation using MPI computing either interactively from the CST Studio Suite® front-end or in batch mode using appropriate command line options. If the simulations need to be started in a computational environment managed by a job scheduling system this section can be skipped. Please proceed to section A.1 in this case.

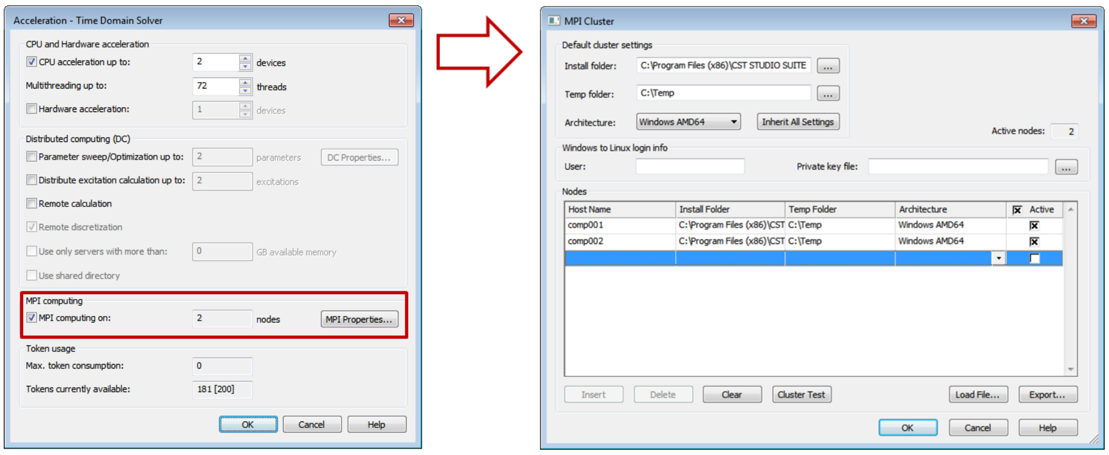
10.1 Starting Simulations Interactively

To activate MPI computing for a simulation please activate the "MPI Computing" checkbox in the acceleration dialog of the solver. Please note that the "MPI computing" checkbox might be disabled if Distributed Computing options are activated. Currently, Distributed Computing and MPI computing are mutually exclusive, i.e., the Distributed Computing checkboxes need to be disabled before MPI computing can be enabled. Click on "MPI Properties" to open the MPI computing dialog (see figure 11). If all computers which have the Compute Node role are identical regarding the folder where the CST Studio Suite® installation can be accessed ("Install Folder" setting) and the place where temporary files are stored during the computation ("Temp folder" setting), these settings can be entered in the "Default cluster settings" section. Select the operating system family the computers with Compute Node role are running in the "Architecture" drop down list. The "Windows to Linux login info" section is only required if the computer with Frontend Node role is running a Windows operating system and the computers with Compute Node role are running the Linux operating system. In this case the user account and the private key file used for login to the Compute Nodes must be specified (see section 6.2.3). In the "Nodes" section the computers which have the Compute Node role assigned need to be specified with their host name. Computer names can be specified multiple times to achieve a certain placement of processes on available hardware resources (see section 12 for more information about the mapping of Compute Nodes to hardware resources). The settings in the dialog can be stored as file using the "Export..." button. This allows to reuse the settings for other model files as the exported file can be imported again by clicking on the "Load File..." button or as predefined machine file for a batch run.

After all settings have been filled in it is recommended to test the MPI computing setup by clicking on the "Cluster Test" button. This will perform a check of the setup and will report any error found during this test.

10.2 Starting Simulations in Batch Mode

The options in the dialog shown in figure 11 can also be specified using a so-called "machinefile", i.e., a text file which contains all options in a well-defined format.


Acceleration - Time Domain Solver

CPU and Hardware acceleration
- [✓] CPU acceleration up to: 2 devices
- Multithreading up to: 72 threads
- [ ] Hardware acceleration: 1 devices

Distributed computing (DC)
- [ ] Parameter sweep/Optimization up to: 2 parameters
- [ ] Distribute excitation calculation up to: 2 excitations
- [ ] Remote calculation
- [✓] Remote discretization
- [ ] Use only servers with more than: 0 GB available memory
- [ ] Use shared directory

MPI computing
- [✓] MPI computing on: 2 nodes

Token usage
- Max. token consumption: 0
- Tokens currently available: 181 [200]

MPI Cluster

Default cluster settings
- Install folder: C:\Program Files (x86)\CST STUDIO SUITE
- Temp folder: C:\Temp
- Architecture: Windows AMD64

Active nodes: 2

Windows to Linux login info
- User:
- Private key file:

Nodes

Host Name	Install Folder	Temp Folder	Architecture	Active
comp001	C:\Program Files (x86)\CST	C:\Temp	Windows AMD64	☒
comp002	C:\Program Files (x86)\CST	C:\Temp	Windows AMD64	☒
				☐

Figure 11: MPI computing dialog showing typical settings for a pure Windows environment.

This can be helpful if a simulation needs to be started in an environment without graphical capabilities (e.g., a Linux system without an active X-server). Below a template for a "machinefile" can be found. Please note that lines which are starting with a hashtag are comments and are not parsed, unless they contain the keywords "CSTMachineFile", "DefaultInstallationFolder", "DefaultTempFolder", "DefaultMachineArchitecture", "GlobalMPICmdLineAddOn", "DefaultLoginUser", or "DefaultLoginPrivateKeyfile" directly after the hashtag (whitespace between the hashtag and the keywords is allowed). Lines which don't start with a hashtag are interpreted as computer names (names of the computers having the Compute Node role assigned). Lines which contain any of the above keywords must be placed before the first line containing a computer name. In the template below, all lines which are interpreted by CST Studio Suite® as they either contain a keyword or are not starting with a hashtag are written in bold font.

```
# CSTMachineFile 20130527
# Template for machine configuration in CST format.
# The very first line in the CST machine file is a version number,
# derived from a date. The format of this first line is:
# # CSTMachineFile YYYYMMDD
# It's recommended to put always the version number corresponding
# to the date when CST STUDIO SUITE was first installed in the header
# of the machine file, e.g., if CST STUDIO SUITE was first installed on
# 14th Nov. 2013 the setting
# # CSTMachineFile 20131114
# should be used.
# This way it can be ensured that the machinefiles will be parsed correctly
# even when the machinefile format changes in a future version of
# CST STUDIO SUITE.
#
# The default settings are specified here:
# DefaultInstallationFolder "<INSTALLATION_FOLDER>"
# DefaultTempFolder "<TEMP_FOLDER>"
# DefaultMachineArchitecture <ARCHITECTURE>
# GlobalMPICmdLineAddOn "<ARGUMENTS>"
#
# For mixed Windows/Linux setup
# DefaultLoginUser <LINUX_USER>
# DefaultLoginPrivateKeyfile "<PATH_TO_KEY_FILE>"
#
# Then each machine node is specified as follows:
# HostName # <MPIEXEC_CMD_LINE_ARGS> ; <TEMP_FOLDER> ; <INSTALLATION_FOLDER> ; <AR-
CHITECTURE> ; <ActiveFlag>
#
# Architecture can have following values:
# Windows AMD64, Linux AMD64
#
# The <ActiveFlag> can have the following values: Active, Inactive
#
# The following list provides an example for a valid list of computers.
# comp001 uses its own settings.
comp001 # "-env MY_ENV_VAR 1" ; "/tmp" ; "/opt/cst/CST_STUDIO_SUITE_2022" ; Linux AMD64
; Active
# comp002 uses the default settings
comp002 # ; ; ; ; Active
```

Please note that only the lines listing the computer names are mandatory. If the other lines are not specified, CST Studio Suite® will use default values as specified in table 8. For most cases it is sufficient to just specify the computer names of the Compute Nodes and stay with the default values as listed in table 8.

Keyword	Default Value
DefaultInstallationFolder	The CST Studio Suite® installation folder on the Frontend Node. The assumption is that all Compute Nodes can access the CST Studio Suite® installation at the same location as on the Frontend Node.
DefaultTempFolder	Default system temporary directory (normally defined by the global environment variables `TMPDIR` or `TEMPDIR`)
DefaultMachineArchitecture	All Compute Nodes are supposed to have the same operating system as the Frontend Node.
GlobalMPICmdLineAddOn	Empty
DefaultLoginUser	Empty
DefaultLoginPrivateKeyfile	Empty

Table 8: Default values for MPI settings.

Example 1: A machinefile with two entries which uses the default settings as listed in table 8:[6]

```
comp001
comp002
```

Example 2: A machinefile which sets a certain temp folder (identical for all Compute Nodes):

```
# CSTMachineFile 20130527
# DefaultTempFolder "/tmp/cst-mpi"
comp001
comp002
```

Example 3: Specify a temp folder for each Compute Node separately:

```
# CSTMachineFile 20130527
comp001 # ; "/tmp/cst-mpi" ; ; ;
comp002 # ; "/scratch/cst-mpi" ; ; ;
```

Once a valid machinefile has been created, the simulation can be started from the command line using the following command line options (in addition to the options starting the selected solver and specifying the CST model file):

```
-withmpi -machinefile="<PATH_TO_MACHINEFILE>"
```

Example: Starting the transient solver of CST Microwave Studio® for the file "modelfile.cst" with MPI Computing, assuming that the CST Studio Suite® installation folder is in the executable search path (`PATH` environment variable) of the system:

Windows:

```
"CST DESIGN ENVIRONMENT.exe" -withmpi -machinefile="<MACHINEFILE>" -m -r modelfile.cst
```

Linux:

```
cst_design_environment -withmpi -machinefile="<MACHINEFILE>" -m -r modelfile.cst
```

10.3 Use MPI Automatically for Huge Projects

Huge FIT-TD simulations with a mesh cell size greater than two billions (2^{31}) will not succeed if they run on a single computer without MPI. The reason is that the FIT-TD solver cannot handle these huge numbers. However, there is solution to overcome this problem, the feature is called "Use MPI automatically for huge projects".

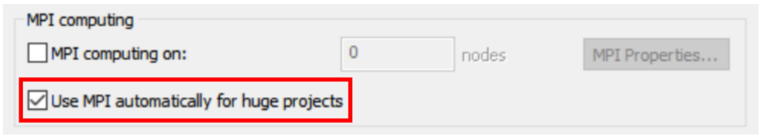
Regular (non-MPI) simulations: If automatic MPI is enabled and CST Studio Suite detects that the mesh cell limit of two billions is exceeded, local MPI will be turned on automatically. As many local MPI simulations as needed will be created, so that no MPI domain solver will exceed the 2 billion limit.

MPI simulations: In a first step, the number of MPI domains will be doubled to reduce the mesh cells per MPI domain solver. This means, two MPI simulations will run on every MPI node. If that should not be sufficient, and the mesh cells still exceed the limit of two billions, the number of MPI domains will be doubled, again.

License: Automatic MPI does not involve additional costs as the number of needed acceleration tokes depends on the used hardware only. And in case of local MPI, the used hardware remains the same, for regular and for MPI simulations.

Automatic MPI can be enabled by command line option "-autompi" or in the acceleration dialog box, see figure 12. For regular simulations, machine file information is not needed as the MPI simulation will run only locally.

```
$  cst_design_environment -autompi -m -r model.cst
```


MPI computing

☐ MPI computing on: 0 nodes MPI Properties...

☑ Use MPI automatically for huge projects

Figure 12: Setting in acceleration dialog box to enable automatic MPI.

On Linux, there is no MPI setup needed to get automatic MPI to work. Just enable the feature and it will work automatically.

On Windows, MPI has to be set up like for a standard MPI simulation, see section 6.2. That means, the hydra service has to be installed and the MPI user has to be registered.

Due to internal data structures, the upper mesh cell limit is not exactly two billions. In fact, it is dependent on the model and mesh cell settings. To have a solution for simulations in which the upper limit is lower than two billions, it can be set by an OS environment variable.

```
$  export CST_2BILLION_LIMIT=1000000000
```

For job scheduling systems, automatic MPI can be disabled in the cluster integration scripts as it is enabled by default. The setting is called CST_AUTO_MPI and can be found in the file cst_settings.

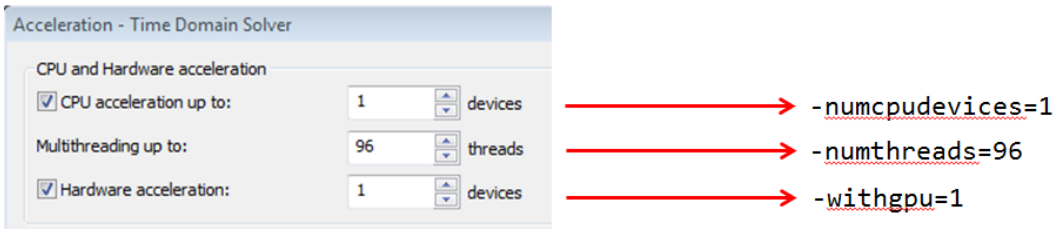
11 Job Scheduling System Integration

Systems used for MPI computing are often high-end computational resources shared by many applications and by many different users. Therefore, such systems are usually access controlled environments, i.e., simulations can't be started interactively but must run in batch mode on computational resources which have been selected automatically by a job scheduling system like LSF, SLURM, Univa Grid Engine, Torque, PBSPro, etc. The topic of submitting CST Studio Suite® simulations to a job scheduling systems is covered in detail by the Cluster Integration Guide available in the CST download area.

12 Assignment of Hardware Resources to Compute Nodes

The assignment of hardware resources to Compute Nodes is a crucial step to achieve optimal performance for a simulation using MPI computing. Hardware resources in this context are CPUs (i.e., CPU cores and CPU devices/sockets) and hardware accelerators (GPU devices). Please note that the algorithm assigning the computational workload and simulation data to Compute Nodes is based on the assumption that the hardware resources of each Compute Node are identical in terms of hardware equipment and performance. Thus, it is strongly recommended to assign identical hardware resources to all Compute Nodes to get the best possible performance.

The hardware resources specified either in the acceleration dialog or on the command line are meant per Compute Node, i.e., if each compute node is supposed to use a single CPU device/socket plus a single hardware accelerator device (e.g., a GPU), one CPU device and one GPU should be specified regardless of the number of Compute Nodes used for the simulation. The settings in the acceleration dialog for this case are shown in figure 13. All three settings can also be specified on the command line of a simulation started in batch mode. The corresponding command line options are shown in figure 13 as well. CST Studio Suite® will assign hardware resources found


Acceleration - Time Domain Solver

CPU and Hardware acceleration

[✓] CPU acceleration up to: 1 devices → `-numcpudevices=1`

Multithreading up to: 96 threads → `-numthreads=96`

[✓] Hardware acceleration: 1 devices → `-withgpu=1`

Figure 13: Settings in the acceleration dialog to assign one CPU device/socket and one hardware accelerator device to each Compute Node.

on the computers specified either in the MPI computing dialog or in the machinefile to Compute Nodes based on the following rules.

1. If the name of a computer taking part in the simulation is listed only once in the MPI computing dialog or in the machinefile, CST Studio Suite® will assign a single Compute Node to the computer. The computational resources assigned to this Compute Node (i.e., the resources CST Studio Suite® utilizes during the simulation) is the minimum of the available resources and the resources requested in the acceleration dialog or on the command line.
2. If the name of a computer taking part in the simulation is listed N times in the MPI computing dialog, CST Studio Suite® will assign N Compute Nodes to the computer. The computational resources of the computer will be split among the

Compute Nodes and CST Studio Suite® will try to ensure that no computational resources are assigned to more than one Compute Node.

To achieve optimal performance, it is strongly recommended to use one of the following settings.

12.1 Full Computer Assignment (Compute Node=Physical Computer)

Compute Node gets assigned all computational resources available on the computers taking part in the simulation. The settings in the acceleration dialog or on the command line should allow to use all hardware resources and the name of each computer should be listed only once in the MPI dialog or in the machinefile.

If simulations are started using a job scheduling system this means that always full machines (not only a set of CPU cores) should be requested for exclusive access.

Example: Hardware properties: Two CPU devices with eight cores each. The computer has two GPUs. The user setting in the acceleration dialog allows to use up to two CPU devices/sockets, up to 96 threads and up to 2 accelerator devices per Compute Node. The computer name is listed only once in the MPI dialog or in the machine file.

$\rightarrow$ The Compute Node will get access to all hardware resources of the physical computer (Compute Node=physical computer). However, not more than 16 threads will be started for the compute intensive part of the simulation (even though the user setting allows to start 96 threads) as the hardware itself doesn't offer more than 16 physical CPU cores.

12.2 NUMA Node Assignment (Compute Node=NUMA Node)

Modern systems often have more than one CPU device/socket. For performance reasons it's often advantageous to assign the resources of one NUMA Node (usually this is one CPU device/socket) to each Compute Node as this can improve the performance as compared to the assignment as described in section 12.1. To achieve such an assignment, the computer with several NUMA nodes should be listed multiple times in the MPI dialog or in the machinefile, respectively. E.g., if a computer has four NUMA Nodes, it should be listed four times. CST Studio Suite® will automatically assign the resources to the Compute Nodes such that each compute node will get the CPU resources of one NUMA Node.

Note: If accelerator devices are connected to the computer it should be ensured that the accelerator devices can be assigned to the Compute Nodes in a reasonable way such that if hardware acceleration is enabled for the simulation, i.e., the number of hardware accelerator devices divided by the number of Compute Nodes assigned to the computer should be an integer number.

Example: Hardware properties: Four CPU devices/sockets with eight cores each, i.e., the computer has four NUMA Nodes. The computer has four GPUs. The user setting in the acceleration dialog allows to use up to two CPU devices/sockets, up to 96 threads, and up to 1 accelerator devices per Compute Node. The computer name is listed four times in the MPI dialog or in the machine file.

$\rightarrow$ Each of the Compute Nodes will get access to one NUMA Node (CPU device/socket) of the physical computer (Compute Node=NUMA Node). CST Studio Suite® ensures that each of the Compute Nodes gets access to the resources of one NUMA Node such that the different Compute Nodes on the same physical computer will not compete for resources with each other. However, not more than 8 threads will be started for the compute intensive part of the simulation on each Compute Node (even though the user setting allows to start 96 threads) as the hardware itself doesn't offer more than 8 physical CPU cores per NUMA Node.

12.3 CPU Core Assignment (Compute Node=CPU Core)

In some environments it might be desired to assign only a single CPU core to each Compute Node. However, this type of resource assignment is the least favorable option for the following reasons. The CST Studio Suite® licensing scheme is based on CPU devices/sockets, not on CPU cores, i.e., even if only a single CPU core of a CPU device is used for a simulation, the CPU device will be counted for the licensing and acceleration tokens will be checked out accordingly (see CST Licensing Guide for more information). On the other hand, the licensing scheme will allow the usage of all CPU cores on a CPU device without additional cost, i.e., using all CPU cores of a CPU device will deliver the best price/performance. Additionally, this type of assignment will usually not work well when hardware acceleration is used to accelerate the computations of the Compute Nodes, as the assignment of hardware accelerator devices to Compute Nodes can't be unique anymore as either a hardware accelerator device is shared between different Compute Nodes, or the number of CPU cores which can be used on a computer is limited by the number of hardware accelerator devices (assuming that the number of accelerator devices is usually smaller than the number of CPU cores on a computer).

Example: Hardware properties: Four CPU devices/sockets with eight cores each, i.e., the computer has four NUMA Nodes. The user setting in the acceleration dialog allows to use up to one CPU device/socket, and up to 96 threads per Compute Node. The computer name is listed 32 times in the MPI dialog or in the machine file.

$\rightarrow$ Each of the Compute Nodes will get access to one CPU core of the physical computer (Compute Node=CPU core). CST Studio Suite® ensures that each of the Compute Nodes gets access to the resources of one CPU core such that the different Compute Nodes on the same physical computer will not compete for resources with each other. However, not more than 1 thread will be started for the compute intensive part of the simulation on each Compute Node (even though the user setting allows to start 96 threads) as only one CPU core can be assigned to each Compute Node without causing a resource conflict.

A Setup of Passwordless SSH

The host key storage is sensitive to the hostname, i.e., connecting to "comp1" and accepting the host key won't enable to connect to "comp1.company.com" even though both names might refer to the same computer. Thus, it is important to specify the hostnames during the login procedure exactly in the way they are specified later on in the acceleration dialog or in the machinefile, respectively.

A.1 Linux Compute Nodes

1. Create a pair of a private and a public key for ssh using the following command:

   ```
   $ ssh-keygen -t rsa
   ```

   Do not use a passphrase if the `ssh-keygen` program asks for it. The `ssh-keygen` program will create two files named `id_rsa` and `id_rsa.pub`, respectively in a directory named `.ssh` in the home directory of the logged-in user.

2. Append the content of `id_rsa.pub` to the file `~/.ssh/authorized_keys` (the file doesn't exist by default), e.g., by using the command

   ```
   $ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
   ```

   The `authorized_keys` file needs to be protected by changing the file access permissions. Please execute the following command to do this:

   ```
   $ chmod 600 ~/.ssh/authorized_keys
   ```

3. If the Compute Nodes do not share the same home directory, the SSH setup needs to be transferred to all Compute Nodes using the command

   ```
   $ ssh-copy-id -i ~/.ssh/id_rsa.pub <compute-node-name>
   ```

   Example: A computer cluster consists of four computers named "comp1", "comp2", "comp3", and "comp4". The computers don't share their home directories. The following series of commands need to be executed in order to prepare all computers for passwordless SSH access:

   ```
   $ ssh-copy-id -i ~/.ssh/id_rsa.pub comp1
   $ ssh-copy-id -i ~/.ssh/id_rsa.pub comp2
   $ ssh-copy-id -i ~/.ssh/id_rsa.pub comp3
   $ ssh-copy-id -i ~/.ssh/id_rsa.pub comp4
   ```

4. Make sure that the configuration file of the SSH server daemon is configured to accept login with a public key not protected by a password. Although it is the default behavior of the SSH server to allow login with a public key without using a password, it might be worth checking that the SSH server configuration file (`/etc/ssh/sshd_config`) doesn't contain the line

   ```
   PubkeyAuthentication no
   ```

which prevents SSH connections with a public key if the key is not protected by a password, at least if SSH still asks for a password when trying to connect from one Compute Node to another.

5. It should be possible to log in to each Compute Node now without being asked for a password. Please make sure that `$ ssh` won't ask whether it should accept a host key during the login procedure as no interactive command prompts will be displayed during the MPI process start and may cause the MPI system to hang. In order to prevent the interactive prompt regarding a host key, either log in to all Compute Nodes using `$ ssh` at least once and accept the host key or configure the ssh client to accept any host key presented, i.e., insert the line

   ```
   StrictHostKeyChecking no
   ```

   into the SSH client configuration file (`/etc/ssh/ssh_config`).

   Please note that in case of Compute Nodes which do not share home directories, it's either required to apply the "`StrictHostKeyChecking no`" setting on all Compute Nodes, or log in from one node to all other Compute Nodes, accept the presented host keys, and then copy the file `~/.ssh/known_hosts` to each Compute Node.

6. To check the setup, the command

   ```
   $ ssh <compute-node-name> hostname
   ```

   can be used. This command should print the hostname of the specified compute node without asking for a password and without presenting any interactive prompt.

A.2 From a Windows Frontend Node to Linux Compute Nodes

This section explains how to configure a passwordless ssh login from a Windows Frontend Node to Compute Nodes running Linux, i.e., this is required for "Configuration 2" in table 5. Please note that this configuration requires that "`ssh`" is also used for user authentication mechanism between the nodes (`I_MPI_HYDRA_BOOTSTRAP`, see section 6.2.2).

In order to start any processes on Compute Nodes the Frontend Node needs passwordless SSH access to all of the Compute Nodes. Make sure that the passwordless remote shell setup between the Compute Nodes as described in section A.1 is already in place before starting the configuration described in this section.

1. Transfer the SSH private key file (`~/.ssh/id_rsa`) from a Compute Node to the Frontend Node. This can be done using the `pscp.exe` program found in the CST Studio Suite® installation folder. `pscp.exe` is a command line program, i.e., a command prompt needs to be started in order to use it.

   Example: If a computer which has the Compute Node role has the hostname "comp1", and the private key file has been created for the user account "mpi-user", the private key file can be transferred using the following command line (assuming that the current directory is the CST Studio Suite® installation directory):

```
pscp.exe mpi-user@comp1:~/.ssh/id_rsa.
```

This will copy the private key file from "comp1" to the CST Studio Suite® installation directory.

Note: The private key file will allow passwordless access to remote computers. It's recommended to store the file in a location on the file system which is only accessible by users of the remote computational resources (e.g., in the "Documents" folder of the logged in user account). This is presuming it's considered as a security risk in the computational environment to store the key file at a central place (like the CST Studio Suite® installation) and potentially share it with other users having access to the computer which has the Frontend Node role.

2. The `id_rsa` file which has been transferred from the Linux side needs to be converted to a different format such that it can be used on Windows. Open the program "`puttygen.exe`" found in the CST Studio Suite® installation directory by double-clicking. Load the `id_rsa` file into the Puttygen program and store it as a "Putty private key file" with file extension "`.ppk`".

3. In the next step, the host keys of all Linux computers, which have the Compute Node role assigned, need to be imported. This requires to establish a remote interactive SSH login using the private key file created in the previous step. Start the program `putty.exe` found in the installation folder of CST Studio Suite® and specify the hostname of one of the Linux computers which has the Compute Node role assigned and specify the username of the remote user account as well as the private key file created in the previous step. Then click on "Open". Select to accept the host key presented by the remote computer. The interactive command prompt which will open, can be closed. There must be no prompt asking for a password. This procedure needs to be repeated for every remote computer which has the Compute Node role assigned.

B Setting Environment Variables

Many system specific settings of the MPI system are configured using environment variables. It is recommended to set these environment variables globally as system environment variables to be sure that the setting is applied to all simulations using MPI. This section describes how to define system wide environment variables.

B.1 Setting System-Wide Environment Variables on Windows

On Windows the "Control Panel" can be used to set environment variables. After opening the "Control Panel" please search for and select "Edit the system environment variables". Then select "Environment Variables..." on the "Advanced" tab of the dialog box which will open. Please note that the new setting will be applied only after pressing "OK" in the dialog box. Please restart any instance of CST Studio Suite® after setting/changing any of the environment variables described in this document to be sure that the new setting is respected by any simulation.

Setting system wide environment variables requires administrative permissions. If the environment variables cannot be defined system wide because of security restrictions, they can be defined for certain user accounts. To define environment variables in the context of a certain user, they can be entered in the "User variables" list of the dialog box.

B.2 Setting System-Wide Environment Variables on Linux

Depending on the Linux distribution used, environment variables can be set by entering them in the file `/etc/environment`, if this file exists. The entries have the form "<var>=<value>" with one entry per line, e.g., `I_MPI_FABRICS=ofa`. Alternatively, the variables can be defined in a shell script which is placed in the `/etc/profile.d` directory. In this case the environment variables need to be defined in the syntax of the system default shell (usually the bash shell) and they must be "exported" such that they show up in the environment of all processes started on the system, e.g., a script containing the single line "`export I_MPI_FABRICS=ofa`" will set the environment variable "`I_MPI_FABRICS`" globally.

Please log out and log in again after setting/changing any of the environment variables described in this document to be sure that the new setting is respected by any simulation.

Changing the `/etc/environment` file and placing a file in `/etc/profile.d` requires root permissions. If the variables can't be set system wide because of security restrictions, they can be defined either in the file `~/.bashrc` of the logged-in user (or equivalent files of the default shell if bash is not the default shell) or in the `cst_settings` file of the script collection (see section 11).

C Setting Up and Testing an Infiniband Network

Setting up Infiniband hardware is beyond the scope of this manual. IT-professionals like system administrators should install and configure the Infiniband software. Please verify that Infiniband communication works properly before running MPI simulations based on that network standard.

Note: If the MPI run fails on Linux Compute Nodes with the error message

"RLIMIT_MEMLOCK too small",

the locked memory size must be increased. Please follow the steps listed below to do so.

1. Open the file `/etc/security/limits.conf` with a text editor as root.
2. Add the following lines to remove the locked memory size limitation for all users:

```
* hard memlock unlimited
* soft memlock unlimited
```

3. Save the file and reboot the system.

D Network and Cluster Checks

If problems occur during the setup of MPI computing when following the instructions of this manual, please perform the steps described in this section, collect the output of all commands listed such that you can provide this information to CST support.

D.1 Check Name Resolution

MPI computing requires a proper name resolution mechanism, i.e., mapping of computer names to IPv4 addresses. This can either be a static mechanism (host file on each computer taking part in the simulation) or a setup using a DNS server. To check whether name resolution works and whether a network connection physically exists between two computers the following command can be used:[7]

```
ping <computer-name>
```

where "`<computer-name>`" should be replaced by the name of the computer to which the connection should be tested, e.g., "ping comp1" will test the connection to a computer with the name "comp1". A reply from the remote computer should be displayed as result.

If a DNS server is used for name resolution, the following commands can be used to check whether the DNS server sends the correct IPv4 address in reply to a computer name:

```
nslookup <computer-name>
```

A reverse name lookup can be checked with the following command:

```
nslookup <IPv4-address>
```

D.2 Check Program File Access

The CST Studio Suite® installation folder must be accessible on all computers taking part in the simulation. To check whether a process started on computer "comp1" can access the CST Studio Suite® installation on computer "comp2", the following command can be used.[8]

On Windows:

```
mpiexec.exe -host comp2 -n 1 "<CST-INSTALL>\AMD64\CSTHardwareInfo_AMD64.exe"
```

The `mpiexec` program can be found in:
"`<CST-INSTALLPATH>\MPI\Windows\IMPI2017U4\intel64\bin`".

[7]The command must be entered in a terminal window, i.e., cmd or Powershell on Windows and any shell on Linux, respectively.

[8]This won't work for configuration 2 as listed in table 5 when started from the Frontend Node. If testing this configuration, please log in to one of the Linux computers via remote shell and execute the command listed for the Linux operating system.

On Linux:

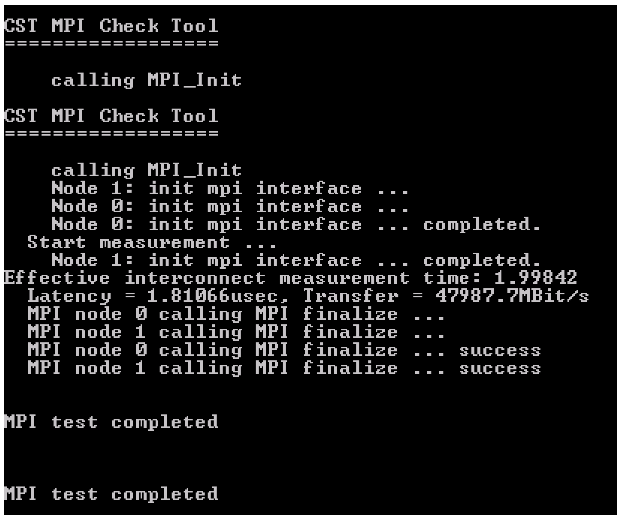
```
mpiexec.hydra -host comp2 -n 1 "<CST-INSTALL>/LinuxAMD64/CSTHardwareInfo_AMD64"
```

The `mpiexec.hydra` program can be found in:
"`<CST-INSTALLPATH>\MPI\Linux\IMPI2017U3\intel64\bin`".

The `<CST-INSTALLPATH>` refers to the directory where the CST Studio Suite® installation can be accessed on "comp2".

D.3 Test MPI Communication

The MPI communication as well as bandwidth and latency of the interconnect selected for the MPI data exchange can be tested. To check whether a Compute Node on computer "comp1" can exchange data with a Compute Node on computer "comp2", the following command can be used.[9]



```
CST MPI Check Tool
==================

    calling MPI_Init

CST MPI Check Tool
==================

    calling MPI_Init
    Node 1: init mpi interface ...
    Node 0: init mpi interface ...
    Node 0: init mpi interface ... completed.
  Start measurement ...
    Node 1: init mpi interface ... completed.
Effective interconnect measurement time: 1.99842
  Latency = 1.81066usec, Transfer = 47987.7MBit/s
  MPI node 0 calling MPI finalize ...
  MPI node 1 calling MPI finalize ...
  MPI node 0 calling MPI finalize ... success
  MPI node 1 calling MPI finalize ... success


MPI test completed


MPI test completed
```

Figure 14: Output of CSTMPIPerformanceTest if the MPI communication works correctly.

On Windows:

```
mpiexec.exe -rr -hosts comp1,comp2 -n 2 \
"<CST-INSTALL>\AMD64\CSTMPIPerformanceTest_AMD64.exe"
```

On Linux:

```
mpiexec.hydra -rr -hosts comp1,comp2 -n 2 \
"<CST-INSTALL>/LinuxAMD64/CSTMPIPerformanceTest_AMD64"
```

An output similar to figure 14 should be shown.[10] If the test is not successful, try to

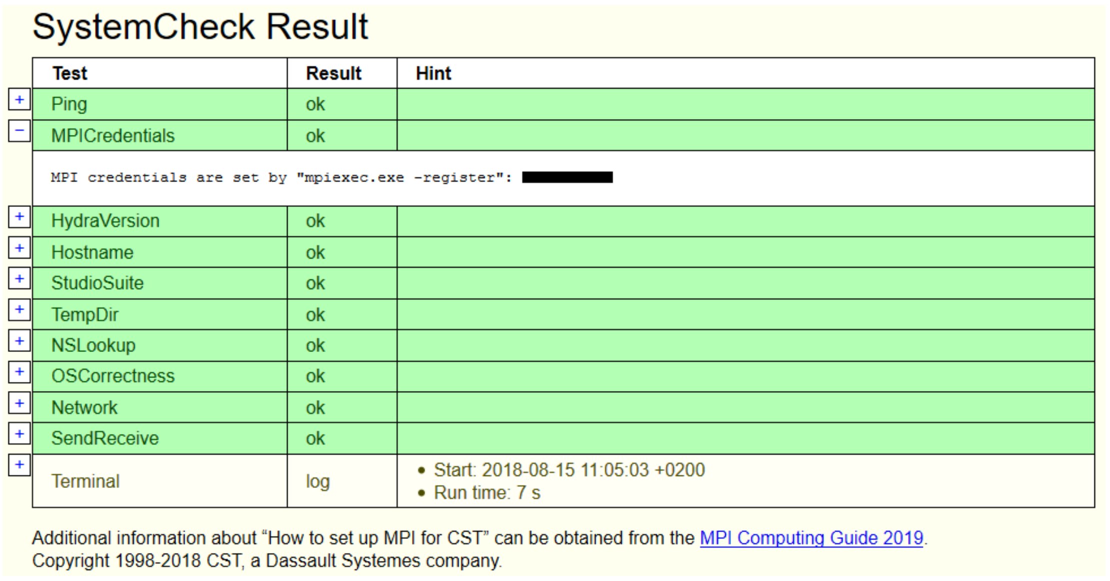
[9]This won't work for configuration 2 as listed in table 5 when started from the Frontend Node. If testing this configuration, please log in to one of the Linux computers via remote shell and execute the command listed for the Linux operating system.

[10]The example output was taken from an Infiniband network configuration. In general, an HPC interconnect should show a latency of a few μs at maximum and at least a bandwidth of 10Gb/s.

configure the network protocol and the network interface used for the MPI communication according to the system setup (see section 4.2).

D.4 CST SystemCheck

CST SystemCheck is an application that checks the MPI Cluster setup by executing several tests, on Linux and Windows. This tool is meant to support users configuring a cluster properly for CST Studio Suite®. For example, it is checked if user-defined



SystemCheck Result

Test	Result	Hint
Ping	ok	
MPICredentials	ok	

```
MPI credentials are set by "mpiexec.exe -register": [redacted]
```

Test	Result	Hint
HydraVersion	ok	
Hostname	ok	
StudioSuite	ok	
TempDir	ok	
NSLookup	ok	
OSCorrectness	ok	
Network	ok	
SendReceive	ok	
Terminal	log	• Start: 2018-08-15 11:05:03 +0200 • Run time: 7 s

Additional information about "How to set up MPI for CST" can be obtained from the MPI Computing Guide 2019.
Copyright 1998-2018 CST, a Dassault Systemes company.

Figure 15: HTML result of a SystemCheck run.

temporary directories exist and if they are writable. The results are available in form of an HTML file (figure 15). In case of a not successful test, some hints will be shown to fix the issue/error in the clusters setup.

Before every MPI solver run, SystemCheck is started automatically. Additionally, there are two different ways to start SystemCheck manually, in the front end and on the command line. These three execution methods will be discussed in detail in the following chapters (D.4.1 – D.4.3).

D.4.1 Automatic Start Before MPI Solver Run

SystemCheck is executed before every MPI solver run to detect issues in an early stage of the simulation. Front end and compute notes are always checked. In case a test fails, the simulation will not start and the cluster has to be adjusted to fulfill the requirements. A click on the link displayed in the message window (figure 16) opens the result file. In case of a failure, the execution can last a while as some tests can hang and are killed only after a certain timeout.

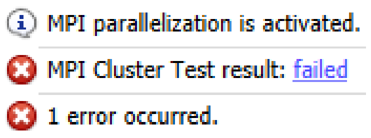
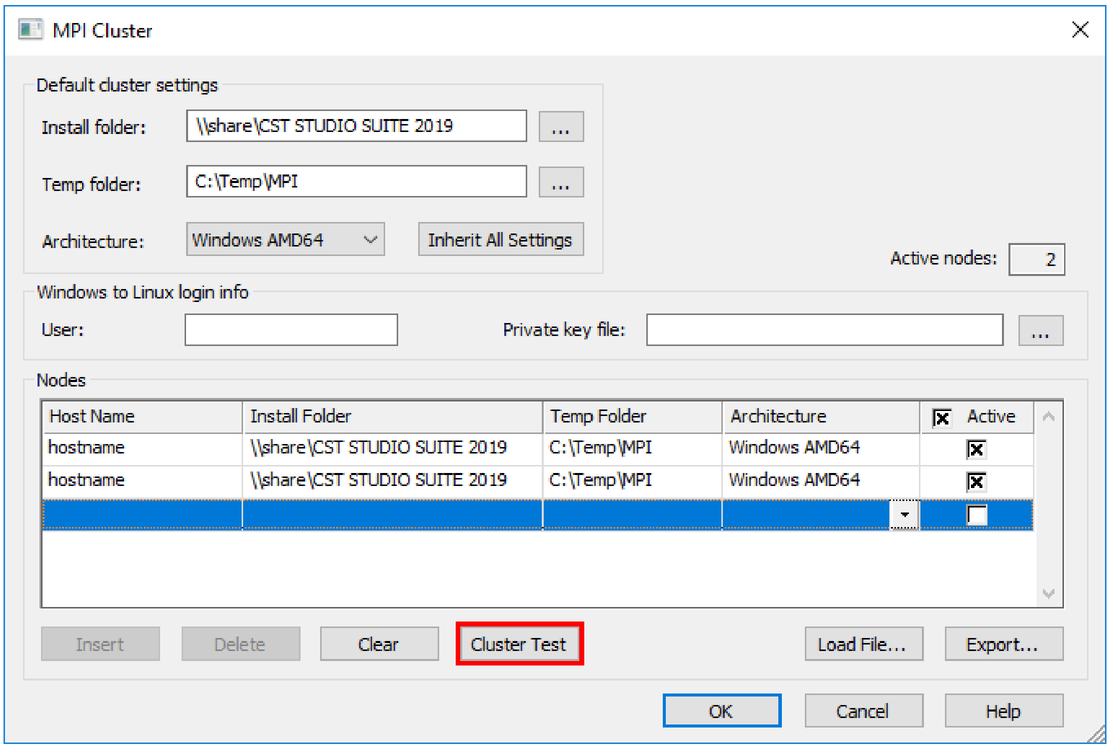

MPI parallelization is activated.
MPI Cluster Test result: failed
1 error occurred.

Figure 16: SystemCheck failure in CST Studio Suite® before simulation start.

D.4.2 Manual Start in the Front End

The manual start supports the user when setting up a simulation project. Using this feature, the cluster can already be checked while configuring the setup. To start SystemCheck in the front end, open the dialog box MPI Cluster (figure 17). This is the dialog box, in that the desired MPI cluster can be defined. Click on the button Cluster


MPI Cluster

Default cluster settings
Install folder: \\share\CST STUDIO SUITE 2019
Temp folder: C:\Temp\MPI
Architecture: Windows AMD64
Inherit All Settings
Active nodes: 2

Windows to Linux login info
User:
Private key file:

Nodes

Host Name	Install Folder	Temp Folder	Architecture	☒ Active
hostname	\\share\CST STUDIO SUITE 2019	C:\Temp\MPI	Windows AMD64	☒
hostname	\\share\CST STUDIO SUITE 2019	C:\Temp\MPI	Windows AMD64	☒
				☐

Figure 17: MPI cluster dialog box of CST Studio Suite®. Start SystemCheck by clicking on the button *Cluster Test*.

Test to start SystemCheck. When the test is finished, an HTML result file (figure 15) will open automatically.

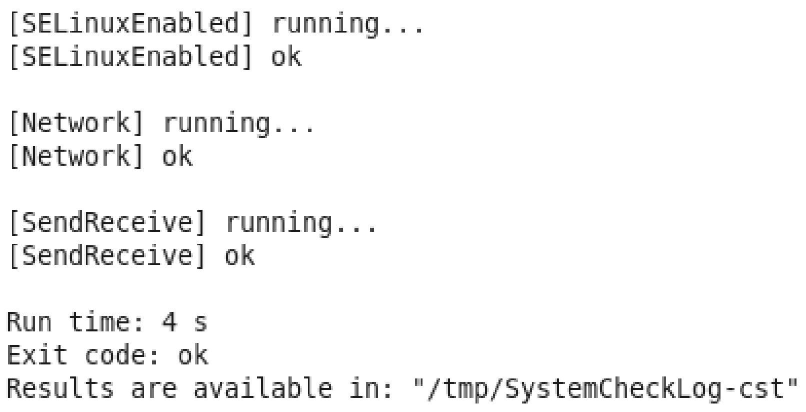
D.4.3 Manual Start on the Command Line

Alternatively, SystemCheck can be started on the command line. This can be very helpful if a cluster is configured using a terminal window. As part of the CST Studio Suite® installation, SystemCheck is located in the directory AMD64 on Windows or

LinuxAMD64 on Linux. To start SystemCheck, a machine file has to be defined. The easiest way to generate a machine file is filling in the dialog box MPI Cluster (figure 17) and then to export the configuration to a text file (section 10.1). This exported file can be used as machine file for SystemCheck. It is also possible to generate the machine file manually as described in one of the previous chapters (section 10.2). SystemCheck can be started by the following command:

```
$ ./SystemCheck_AMD64 -m /home/user/mpi_machines.txt
```

Some essential output will be displayed by SystemCheck in the terminal (figure 18). More detailed results can be obtained from the HTML result file.



```
[SELinuxEnabled] running...
[SELinuxEnabled] ok

[Network] running...
[Network] ok

[SendReceive] running...
[SendReceive] ok

Run time: 4 s
Exit code: ok
Results are available in: "/tmp/SystemCheckLog-cst"
```

Figure 18: Terminal output of SystemCheck on Linux.

The configuration of SystemCheck can be adjusted by command line options. Calling SystemCheck with --help will display all available options:

```
$ ./SystemCheck_AMD64 --help
```

In principle the behavior of SystemCheck on Linux and Windows is identical. But the specific test set that will check the cluster is dependent on the operating system.