SANDIA REPORT

SAND2003-4072
Unlimited Release
Printed November 2003

ASCI Red for Dummies - A Recipe Book for Easy Use of the ASCI Red Platform

Paula L. McAllister, Allen G. Sault (ed.), Suzanne M. Kelly, Gerald F. Quinlan, and Joel D. Miller

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation,
a Lockheed Martin Company, for the United States Department of Energy's
National Nuclear Security Administration under Contract DE-AC04-94AL85000.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865)576-8401
Facsimile: (865)576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.doe.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800)553-6847
Facsimile: (703)605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2003-4072
Unlimited Release
Printed November 2003

ASCI Red for Dummies
A Recipe Book for Easy Use of the ASCI Red Platform

Paula L. McAllister
Scientific Computing

Allen G. Sault, editor
Mission Analysis & Simulation

Suzanne M. Kelly and Gerald F. Quinlan
Scalable Systems Integration

Joel D. Miller
Visualization & data

Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-0807

Adapted from *ACSI White for Dummies* by

Jim Ang (ed.), Martha Ernest, Kathie Hiebert-Dodd, Curtis Janssen, Jeff Jortner, and Judy Sturtevant

Abstract

It has been recognized that documentation for new customers of ASCI Red, aka janus or the Intel Teraflops at Sandia National Laboratories, has been sadly lacking. This document has been prepared by a team of subject matter experts to fill that void and to provide a starting point for providing a similar document for ASCI Red Storm in the future. This document is intended for SNL users who need to jumpstart their use of Janus and Janus-s.

Acknowledgements

The authors wish to thank the following individuals for their contributions to this project:

The ASCI Red Operations team (Frank M. Jaramillo, Jason J. Repik, Victor G. Kuhns, Paul E. Sanchez, and Sean R. Taylor, all of Compaq Federal) for verifying the accuracy of the technical operational content of this paper, and Sue P. Goudy (Scalable Systems Integration) for testing user readability of this document. The authors also wish to thank John Noe (Manager of Scientific Computing) and Doug Doerfler (Manager of Scalable Systems Integration) for enthusiastically supporting and encouraging this work, and the staff of both the Scientific Computing Department and the Scalable Systems Integration Department for their contributions.

Contents

Scope

The primary audience for this document is Sandia's code team members who need to get up to speed on how to use the ASCI Red platform. The authors made a conscious effort to limit information to only those essential facts and recommendations that help new users get started. This document is based on a similar document prepared for ASCI White, with additional information gleaned from SNL ASCI Red documentation; comments and FAQs from users; and input from the ASCI Red help, software, and system teams. Our goal is to provide new SNL ASCI Red users a concise, easy to understand guide. This recipe book is not intended to provide all of the details that users may need or want to learn. Extensive documentation, tutorials and the latest information are available in the ASCI Red Users Guide

(http://www.sandia.gov/ASCI/Red/UserGuide.htm) and the Getting Started with ASCI Red page (http://www.sandia.gov/ASCI/Red/Start.htm).

Prerequisites

In order to use ASCI Red, there are a few prerequisites:

1. A Sandia Kerberos password (often the same one you use to read your mail). For classified computing, a separate classified Kerberos password is required.

2. An approved unclassified or classified account for ASCI Red.

3. The ssh (secure shell) software: Version 2 for unclassified and at least Version 1 for classified.

4. A rudimentary knowledge of UNIX and one of its shells.

For help with item 1 and 2, see "Establishing Accounts" on page 9 of this document, or the Web-based Computer Account Request System (WebCARS) account-request web page https://workflow.sandia.gov/webcars/webcars.html. For help with item 3, ssh, see https://secureweb.sandia.gov/.../dce.sandia.gov/fs/web/public/html/dfs/asci_tools.html. There are various sources for item 4, including classes (see http://www.learningtree.com), numerous books (try searching http://www.amazon.com/), and on-line tutorials (use your favorite search engine).

A Very Rapid Jump Start

If you want to get going as quickly as possible, try the following brief tutorial. If this information does not meet your needs, the remainder of the document provides more detailed information.

Short Tutorial

1. Log on to the front-end compile server.

   If you are using F-Secure ssh from a Windows system, start F-Secure and specify sasn100 (or sasn101 for classified) as the host. Supply your username and when prompted, your Kerberos password (or classified Kerberos password for sasn101).

   If you are using a UNIX workstation, type the following at your prompt (shown as "%" below) and supply your Kerberos password (or classified Kerberos password for sasn101) when prompted:

   % ssh sasn100 *(replace sasn100 with sasn101 for classified)*

2. The first time you log in, you must set up your C shell initialization script:

```
% cp local.cshrc .cshrc

% source .cshrc
```

3. Create and compile a C hello_world program in your home directory:

```
% echo 'int main () { printf("Hello world\n"); }' > hello.c

% cicc -o hello hello.c
```

4. Logon to ASCI Red:

```
% ssh janus                                    (or janus-s for classified)
```

5. Run your hello program interactively on two compute nodes. Yod is the program that loads your program onto compute nodes:

```
% yod -sz 2 hello          (the text string "Hello world" will be returned twice)
```

6. Run the hello program on three compute nodes from a batch job. First, create a shell script of commands to be run, then submit the script:

```
% echo "yod -sz 3 hello" > sample

% qsub -lP 3 -lT 00:00:30 -q snl.day sample      (note the request
                                                   number returned)
```

7. Check on the batch job:

```
% qstat                                  (repeat until your job no longer appears)
```

8. View your results

```
% cat sample.o<request#>   (replace <request#> with number noted in step 6)
```

   Ignore the error message "Warning: no access to tty". It is just a warning that batch jobs should not try to set terminal characteristics in a batch job. This comes from your .cshrc trying to set your erase key.

9. Check for errors:

```
% cat sample.e<request#>   (replace <request#> with value returned in step 6)
```

10. You're done! Type `exit` to log off janus[-s] and then type `exit` again to log off the compile server.

Platform Description

The table on the next page summarizes the machines that comprise the ASCI Red platform, with additional information given following the table. More detailed descriptions of ASCI Red can be found at http://www.cs.sandia.gov/ISUG97/papers/Mattson/OVERVIEW.html and http://www.sandia.gov/ASCI/Red/RedFacts.htm, though users are cautioned that due to upgrades

and configuration changes, some of the specific information found in these links may no longer be valid.

Machine Name	Classification	Primary Usage	Nodes*, memory
janus	Unclassified	Large Batch Jobs and Code Development	Big mode: batch – 3204, 256M interactive – 140, 256M Small mode: batch – 1028, 256M interactive – 140, 256M
sasn100	Unclassified	Unclassified Compile Server	2, 4G
janus-s	Secure	Large Classified Batch Jobs	Big mode: batch – 3326, 256M interactive – 16, 256M Small mode: batch – 1150, 256M interactive – 16, 256M
sasn101	Secure	Classified Compile Server	2, 4G

*** Each node on janus and janus-s contains two processors**

Janus and Janus-s

The Red Platform consists of an unclassified side named janus and a classified side named janus-s, with a center section that can be attached to either janus or janus-s. Janus consists of 1168 compute nodes, janus-s consists of 1166 compute nodes, and the center section consists of 2176 compute nodes. When a side is attached to the center section it is referred to as being in "big" mode. Otherwise, the side is referred to as being in "small" mode. Each compute node contains two 333 MHz Intel Pentium® processors and 256 Mbytes of RAM. Users decide at the time of job submission whether or not to use both processors. When both processors are used, the memory per processor is reduced to 128 Mbytes. If only a single processor is used per node, the full 256 Mbytes of RAM is available to that processor. A small number of compute nodes are reserved for interactive use, and are intended to facilitate code development, debugging, and testing of input files prior to submission of large batch jobs. Because most code development is performed on the unclassified side, a larger number of interactive nodes are available on janus. In addition to the compute nodes, each machine also has 26 service nodes and 36 I/O nodes (37 on janus-s).

At the time of this writing, the center section is moved between janus and janus-s every two weeks unless priority usage of the machine mandates a change in the schedule. Users can find the schedule for the center section, as well as other important events, by typing (the "%" is the command prompt):

```
% news janus-dedicate
```

on sasn100. The schedule is also available from the WebEvents calendar at https://www.prod.sandia.gov/cgi-bin/cals-SCS/webevent.cgi, for users with a Sandia Kerberos

password. After logging on to the WebEvents site, select the appropriate calendar (TF-Janus classified, or TF-janus unclassified) to see all scheduled events on janus and janus-s.

Compile Servers

Compile servers are available for both the unclassified and classified sides of janus. The unclassified compile server is named sasn100, while the classified compile server is named sasn101. Both of these machines are two-processor 900 Mhz Sun Fire workstations. User home directories on sasn100 (sasn101) are NFS (Network File System) mounted to janus (janus-s) so that users do not have to move executables to janus or janus-s after compilation.

How to Get Started

User Support Points of Contact

All requests for user support or information regarding ASCI Red should be directed to janus-help@sandia.gov. Requests for user support or information about the compile servers should be directed to tflan-help@sandia.gov. In addition, there is a Frequently-Asked Questions (FAQ) page (http://www.sandia.gov/ASCI/Red/usage/faq.html) that users are encouraged to consult before requesting additional help. Additional FAQ files can be found in the /usr/local/FAQ directory on sasn100.

Establishing Unclassified Accounts

SNL users desiring to use the unclassified ASCI Red platform need accounts on both janus and sasn100. Accounts are obtained by accessing the unclassified-account request form from the WebCars application (https://workflow.sandia.gov/webcars/webcars.html), filling in the appropriate information for an Intel Teraflops – SRN account, and submitting the request. Note that a valid project/task number is required to obtain an account, though currently no charges will be made against the account. Once a request is approved, the necessary accounts will be automatically created on both janus and sasn100 and the user will be notified by e-mail. Users are encouraged to request a restricted Sandia mass storage system (rsmss) account when requesting an account on janus. The rsmss account will provide users with a long-term data storage option. (See the “I/O on Janus and Janus-s” and “File Transfer to and from Janus Platforms” subsections below for information on transferring files.) Only persons with a valid Kerberos password can request accounts. Users who do not have a valid Kerberos password, such as new employees and users from educational institutions, must have another person submit the request on their behalf.

Establishing Classified Accounts

SNL users desiring to use the classified ASCI Red platform need accounts on both janus-s and sasn101, as well as janus and sasn100. Sandia Classified Network (SCN) accounts are obtained by accessing the classified-account request form from the WebCars application on the SRN (https://workflow.sandia.gov/webcars/webcars.html), filling in the appropriate information for an Intel Teraflops – SCN account, and submitting the request. Classified users must also have a classified Kerberos password, which also can be obtained using the WebCars application. Note that a valid project/task number is required to obtain an account, though currently no charges will be made against the account. Once a request is approved, the necessary accounts will be automatically created on both janus-s and sasn101 and the requestor notified by e-mail. Janus-s users must also have accounts on janus and sasn100 to access the calendars, FAQs, etc. located on sasn100. Users are encouraged to request a classified Sandia mass storage system (smss) account when requesting an account on janus-s. The smss account will provide users with a classified long term data storage option. (See the “I/O on Janus and Janus-s” and “File Transfer

to and from Janus Platforms" sections below for information on transferring files.) Only persons with a valid Kerberos password can request accounts. Users who do not yet have a valid Kerberos password must have another person submit the request on their behalf.

Logging on to Janus and Sasn100

The Kerberos software allows you to perform your DCE/Kerberos authentication on a local machine to obtain a forwardable authorization credential to access and use janus and sasn100. Cross–cell users from other laboratories must have this credential before using ssh (secure shell on Unix, or F-Secure ssh on Windows) to connect to janus. Version 2 of ssh is required for janus, while ssh version 1 or 2 is required for janus-s. Sandians can obtain both Kerberos and ssh software from https://secureweb.sandia.gov/.../dce.sandia.gov/fs/web/public/html/dfs/asci_tools.html. Others can obtain ssh from http://www.f-secure.com/products/ssh. To log in to janus from a Unix machine running Kerberos software, start with the following commands, where "%" is the command-line prompt:

```
% /usr/local/bin/kinit -f

% ssh janus.sandia.gov
```

The kinit command will ask you to enter your Kerberos password and will generate a time stamped Kerberos authentication ticket that is automatically forwarded with ssh. This will allow you to log on to janus or sasn100 using ssh without retyping your Kerberos password, until the ticket expires. Alternatively, if you do not have Kerberos on your local Unix machine you can simply type:

```
% ssh janus.sandia.gov
```

and you will be prompted for your janus password during the login process. Note that from a machine connected to either the SON (Sandia Open Network) or the SRN the machine name "janus.sandia.gov" can be truncated to "janus".

To log in from a Windows system you must obtain the F-Secure ssh client from one of the web sites listed above. Once the F-Secure software is running, click on the "quick connect" button, type in janus.sandia.gov as the host name (or janus if your local machine is connected to the SON or SRN), supply your user name, and click the connect button. When prompted enter your Kerberos password.

Logins to sasn100 are achieved using an analogous procedure, except that the machine name "sasn100.sandia.gov" is substituted for "janus.sandia.gov" (or in truncated form, "sasn100" for "janus").

Logging on to Janus-s and Sasn101

Logins to janus-s or sasn101 must be performed from a machine connected to the SCN. From such a machine, the login procedure is identical to that for janus and sasn100, except that the machine names are janus-s and sasn101 and you must use your classified Kerberos password.

Allocation and Access to Resources

Batch Nodes

The batch nodes of ASCI Red are allocated through the NQS (Network Queuing System). There are no quotas for use of the ASCI Red machine, but janus limits the number of jobs an individual user may have queued or running at any given time. This limit depends on whether janus is in big or small mode, and on the number of nodes and the computation time requested. In normal use, janus also limits the maximum CPU time that a job can run to 24 hours. Janus-s operates with fewer restrictions; users can submit a greater number of jobs for longer times than on janus. In addition, users with mission- and time-critical needs can request priority use of the machine by contacting janus-managers@sandia.gov. Approved priority requests are handled by giving the user access to special priority queues or by dedicating all or part of the machine to a specific project for a given time. Details of current queuing and priority-use policies are provided in /usr/local/FAQ/NQS_general_info on sasn100.

Interactive Partition

The interactive partition on janus is intended for code development, debugging, and testing on a small number of nodes. Production jobs should be run using the batch queuing system. There are 140 total nodes in the interactive partition on janus. On janus-s, the interactive partition is substantially smaller since relatively little code development is performed in the classified environment. The use of the interactive partition is governed by “good citizen” rules which can be found in /usr/local/FAQ/janus_good_citizen on sasn100. Users are strongly encouraged to read and abide by the rules in this document.

Storage

Each user’s account on janus and janus-s is provided with a limited amount of “home” disk space. The specific user’s home file space is /Net/usr/home/<userID> on janus and janus-s and /usr/home/<userID> on sasn100 and sasn101. The physical drive containing the home directories is mounted on sasn100 (sasn101), and NFS mounted to janus (janus-s). This file space is backed up regularly. However, the size of home disk storage is restricted to 500 megabytes per user, so users are discouraged from storing data in their home directories. Users with long-term data storage needs, or, for that matter, users with any data of importance, should copy their valuable files to Sandia’s High Performance Storage System (rsmss for unclassified, smss for classified). Several tools are available for moving files to and from janus and janus-s; these tools are described below in the “I/O on Janus and Janus-s” and “File Transfer to and from Janus Platforms “ subsections. Note that program loads from home directories are very slow and may cause problems such as load aborts and timeouts. Executable programs should be copied to one of the UFS (Unix File System) disks and loaded from there.

There are two UFS disk systems available for temporary file storage: /ufs and /scratch. The /ufs and /scratch file systems contain arrays of ten to fifteen disks of around 30 GB size per disk. Unlike the home disks, these disks are not backed up and are subject to purging if necessary.

Much larger temporary storage capacity is available on the GPFS (General Parallel File System) disks. Janus and janus-s each have a total of approximately 5 terabytes (TB) of GPFS storage space in their /pfs_grande file systems.

- janus: the /pfs_grande file system is currently set up with 18 2-way parallel file systems (/pfs_grande/tmp_1 – tmp_18) with a storage capacity of 180 gigabytes (GB) apiece, and 3 12-way parallel file systems (/pfs_grande/multi/tmp_1 – tmp_3), each of which has a capacity of 540 GB.

- janus-s: /pfs_grande currently consists of 18 2-way parallel 280 GB file systems (pfs_grande/tmp_1 – tmp_18), and 1 36-way parallel file system (/pfs_grande/multi/tmp_1) with a maximum capacity of 5 TB. However, the 36-way file system shares disk space with the 2-way file systems, so the total disk capacity is limited to 5 TB. The amount of disk space divided between the various file systems is subject to change depending upon specific job usage requirements.

The /pfs_grande directories are the recommended location for all parallel I/O. A detailed analysis of the parallel file systems can be obtained from the link "Getting I/O performance on ASCI RED report" at http://www.sandia.gov/ASCI/Red/usage, while a brief overview is available from FAQ 37 at http://www.sandia.gov/ASCI/Red/usage/faq.html#ioperformance.

Again, the /pfs_grande, /ufs and /scratch file systems are for temporary storage only. These file systems are not backed up and are subject to purging if the need arises. Users should create their own directories on these disks using their <userID> and keep all of their files in their directories, not at the top level (*e.g.*, /scratch/tmp_2/<userID>/myProjectFiles, rather than /scratch/tmp_2/myProjectFiles). Users are responsible both for backing up their own data they write to these file systems and for deleting unneeded files. General file system policies for janus can be found in /usr/local/FAQ/janus_good_citizen on sasn100.

Running Jobs on Janus and Janus-s

The yod Command

The basic command for running both interactive and batch jobs is "yod." A minimal invocation of the yod command is as follows:

```
% yod –sz <size> <executable name>
```

where size is the number of processors requested for the job. If yod is entered at the command line the job will run in the interactive partition. In order to use the batch nodes, yod must be run from within an NQS script (see "Running Batch Jobs using NQS" subsection below). Nodes in the interactive partition are allocated on a first-come, first-serve basis, while batch nodes are allocated by the NQS queuing software. If a sufficient number of nodes are not available for interactive allocation, yod will fail and return an error message. Use of the interactive nodes is governed by the good citizen rules listed in /usr/local/FAQ/janus_good_citizen on sasn100. All users are expected to abide by these rules voluntarily in order to ensure equitable access to janus resources.

Many other options are available with yod in addition to the –sz option. A complete list of options can be found on the yod man page on janus (enter man yod at the command prompt), while a description of the most commonly used options along with some usage examples are given in FAQ #25 at http://www.sandia.gov/ASCI/Red/usage/faq.html#runcode. This information is reproduced here for a few of the most basic options:

Usage: `yod [-stack <size>] [-proc|-p <0|1|2|3>] [-sz|-size <size>] [-fyod <num_fyods>] <file> <args>`

Options:

`-stack <size>` --- Reserve the size in bytes for the stack. The default is 2M. If other CPUs on the same node are used for computation (see the –proc option), the stack is divided evenly among the CPUs. Note that megabytes can be abbreviated with "M", *e.g.*, 2000000 bytes may be expressed as 2M.

`-proc <mode>` --- The default mode is 0 which uses only one of the Pentium II microprocessors on each node. If mode is set to 1, the second processor is turned on and used as a message coprocessor. Mode 2 allows programs to use the cop() and cop2() functions to execute code simultaneously on all processors. Mode 3 ("virtual node" mode) allows the program to use the second processor of each physical node by subdividing it into 2 virtual nodes. In virtual node mode, each virtual node has access to only 128 MB of memory.

`-p <mode>` --- Same as the –proc option

`-size <size>` --- The number of processors that should be allocated, specified as an integer. The default size in interactive mode is 1, while in batch mode it is the entire allocation specified on an NQS submission.

`-sz <size>` --- Same as the -size option.

`-fyod <num_fyods>` --- Run a specific number of f(ile)yods in the service partition for I/O. This option is best reserved for fine tuning I/O intensive applications. See FAQ 36 at http://www.sandia.gov/ASCI/Red/usage/faq.html#fyodinfo for an introduction to fyods.

`<file> <args>` --- file is the executable to be run, and args are any arguments required by file.

Interactive Job Examples Using "yod"

Example 1: run the code hello_world on five nodes.

```
% /cougar/bin/yod -sz 5 hello_world
```

Example 2: run the code do_it on 128 processors in virtual node mode (2 processors per node, 64 nodes) and use input_file as the code's first command line argument.

```
% /cougar/bin/yod -sz 128 -p 3 do_it input_file
```

Terminating Interactive Jobs

Interactive jobs may be killed by using the following commands.

```
% ps          (shows Process ID numbers, or PIDs, of your currently running jobs)

% kill -2 <PID>          (multiple jobs can be killed by including a list of PIDs)
```

Running Batch Jobs Using NQS

Janus and janus-s use NQS to control the submission of batch jobs. Jobs are submitted to NQS using the qsub command. Both janus and janus-s are configured with multiple queues, and the current policies regarding the use of these queues can be found in either /usr/local/FAQ/NQS_general_info on sasn100 or http://www.sandia.gov/ASCI/Red/usage/NQS_general_info.html. Upon creation of an account users will be given access to one or more of these queues depending on the user's project affiliation. Most Sandia employees will be given access to the snl queues, while users from educational institutions will be given access to either the edu or snl queues, depending on the nature of their work.

The basic use of the qsub command is summarized below. More complete details can be found at http://www.sandia.gov/ASCI/Red/usage/nqs/qsub.html or in the qsub man pages on janus and janus-s (enter man qsub at the command prompt). A short tutorial on submitting batch NQS jobs is available at http://www.sandia.gov/ASCI/Red/usage/nqs/index.html.

NQS job submission using qsub is typically performed with a command line similar to the following:

```
% qsub [flags] [script file]
```

where flags is a list of options, and the script file contains commands required to run the job, including the yod command(s) that actually submit the job(s). At a minimum, users must specify the –lP, -lT, and –q options , where -lP <size> requests the number of nodes, -lT <time> requests that the processors be reserved for a length of time in seconds (or optionally formatted as [[hours**:**] minutes**:**] seconds[**.**milliseconds]), and –q <queue> specifies the queue to which the job is being submitted. These options (-lP, -lT, and -q) can either be specified on the command line or embedded in the script file. Additional qsub options can be found in the qsub man pages or at http://www.sandia.gov/ASCI/Red/usage/nqs/qsub.html.

Here is an example of a working qsub submission line:

```
% qsub -q snl -lP 100 -lT 24:00:00 run3
```

The `-lP`, `-lT` and `-q` options must be specified on the `qsub` line. Additional qsub options can either be specified on the command line or in the script file as embedded default flags. Embedded default flags must be listed at the start of the script file before any other commands. Each line setting an embedded flag must have "# " as the first non-whitespace character, and the second non-whitespace character must either be "Q" immediately followed by "SUB", or the @ character immediately followed by "$". The next character must be "-" followed by the name of the flag (*e.g.*, lT, lP, a) and any parameters required by the flag. The first line that does not follow this syntax will terminate the recognition of embedded flags. If an embedded default flag also is specified on the command line, the value on the command line will take precedence.

Here is an example of the use of embedded flags within a script file. Note that comments can also be inserted between the embedded flag specifications.

```
#
#  Batch request script example:
#
#  @$-a "11:30pm EDT" -lt "21:10, 20:00"
#                # Run request after 11:30 EDT by default, and
#                # set a maximum per-process CPU time limit of
#                # 21 minutes and 10 seconds. Send a warning
#                # signal when any process of the running batch
#                # request consumes more than 20 minutes of CPU
#                # time.
#
#  QSUB-lT 1:45:00
#                # Set a maximum per-request CPU time limit of
#                # one hour and 45 minutes. (The implementation
#                # of CPU time limits is completely dependent
#                # upon the NQS implementation at the execution
#                # machine.)
#
#  QSUB-mb -me
#                # Send mail at beginning and end of request
#                # execution.
#
#  @$-q batch1   # Queue request to queue: batch1 by default
```

```
#
#  @$            # No more embedded flags.
#
make all
```

Following the embedded flag specifications, the script should contain additional commands necessary to run the job, including one or more yod commands. The qsub script will run in the user's login shell unless an alternative shell is specified at the beginning of the script. Thus, any commands or control structures that are allowed within the shell may be included in the script file.

Within an NQS script file, the –sz parameter may be omitted from the yod command. In this case, the sz parameter will take the value specified by the qsub –lP parameter, except in –p 3 mode where the sz parameter will be twice the -lP value. If the yod -sz parameter specifies fewer nodes than specified by the qsub –lP parameter, then the yod –sz parameter will determine the number of processors allocated to the job. This capability allows users to run more than one yod command simultaneously within an NQS script, dividing the nodes requested by the qsub –lP parameter among the various jobs.

Users may also specify consecutive yod commands with different numbers of nodes in the event that preprocessing and post-processing of data is required, but the pre-processing and post-processing programs do not require as many nodes as the main program. If a yod –sz command requests more processors than specified by the qsub –lP parameter, an error will be generated and the script will exit without running the program.

When a yod command specifies the –proc 3 option within an NQS script, the number of processors requested by the yod –sz parameter may be twice that specified by the qsub –lP parameter. This is allowed since qsub –lP <size> requests a total number of nodes having two processors apiece, while yod –sz <procs> requests the number of processors (or virtual nodes). Thus, if n nodes are allocated by NQS, 2n processors or virtual nodes are available.

A set of built-in environment variables are available within NQS scripts. One of the most important is QSUB_WORKDIR, which contains the absolute path name of the directory from which the job is submitted. By default, qsub scripts start in the user's home directory. By including the line

```
cd $QSUB_WORKDIR
```

users can force the NQS script to look for files in the directory from which the job was submitted. Additional built-in variables include QSUB_HOME, QSUB_SHELL, QSUB_PATH, QSUB_USER, QSUB_LOGNAME, QSUB_MAIL, and QSUB_TZ, which correspond to the shell-environment variables with the same names, minus the QSUB_ prefix. (Enter the command env in a unix shell window to list the values of the current shell-environment variables.)

Standard output and standard error from batch jobs will be directed to the files <job_name>.o<request#> and <job_name>.e<request#>, respectively, where <job_name> is the first seven letters of the name of the NQS script, and <request#> is the value returned by qsub upon submission. The request# can also be obtained from the first column returned by the qstat command (see next section).

Monitoring Batch Jobs

The qstat command is the main method by which users can monitor the status of NQS jobs. By default qstat only shows jobs submitted by the user. Adding the –a option shows all jobs, while the –v option generates verbose output in a 132 column format. Consult the qstat man page or http://www.sandia.gov/ASCI/Red/usage/nqs/qstat.html for further details.

Additional job status information can be obtained from “showmesh” and “qbyhand”. The showmesh command generates a graphic showing all nodes on the machine and the jobs (yods) to which each node is assigned, as well as free nodes, failed nodes, and several other types of nodes. The numbers of free interactive and NQS nodes are also displayed. Alternatively, /usr/community/bin/shmesh generates a more compact version of the same information. The qbyhand command provides information that allows users to estimate the earliest time at which a submitted job might be loaded.

Finally, the qwall utility outputs the time until the next wall. Walls are set to allow the queues to be cleared prior to configuration changes or preventative maintenance, and to prevent jobs from running if they will be interrupted by these events. In effect, qwall returns the maximum time that an NQS request may specify and still start before the next wall. Walls are only used on janus, not janus-s.

The command qstat is located in /usr/bin on janus and janus-s, while showmesh is located in /cougar/bin. Both of these directories are part of the standard path provided to users with the default standard .csrhc and .profile files. The qbyhand and qwall commands are located in /usr/community/bin, and users must either specify the full pathname or add this directory to their PATH shell-environment variable in order to use these commands.

Users may also monitor the job status through the Grid Services Monitoring Pages. These pages are available only on the SRN for janus and the SCN for janus-s. From the Sandia home page, select “Grid Services” from the pull down menu under “Engineering & Manufacturing”, and select the “Monitoring” link on the left side of the page. The “Grid Utilization” link then leads to a tabular and graphical display of machine and job status. A User’s Guide is available from the monitoring page that users should consult to learn how to use the tool.

Deleting batch jobs

The qdel utility allows users to terminate batch jobs that are running or queued. In order to use qdel, users must first determine the NQS request number, given in the first column of the qstat output. Jobs that are queued but not running can be deleted by entering:

```
% qdel <request#>
```

Running jobs can be killed by the command

```
% qdel -k <request#>
```

which sends a SIGINT (interrupt job) signal, and, if the job is not terminated in 45 seconds, a SIGKILL (kill job) signal to the running process

I/O on Janus and Janus-s

As described in the “Storage” subheading under “Allocation and Access to Resources” above, both janus and janus-s have parallel I/O capability as well as standard serial I/O capability. Users can avail themselves of the parallel capability by simply reading from and writing to the /pfs_grande directory where the parallel file systems are mounted. As a general rule, when reading or writing less than 256 kilobytes per operation or when using ASCII format, best performance is achieved by writing to one of the UFS file systems (/ufs or /scratch). For operations involving more than 256 KB or binary file formats, users should use the parallel /pfs_grande files systems. Only the /pfs_grande file systems can handle file sizes greater than 2 GB. Users should avoid reading from and writing to their home directories or any directory starting with /Net/ as these directories are NFS mounted from sasn100 (sasn101 for janus-s) and performance will be extremely poor.

Optimizing I/O performance requires a full understanding of the parallel I/O system. Users with demanding I/O needs should carefully consult http://www.sandia.gov/ASCI/Red/usage/ioreport.ps and http://www.sandia.gov/ASCI/Red/usage/pres_io/index.htm for more information on this topic. Note that since the publication of these resources, the parallel file system has been moved from /pfs to /pfs_grande.

Red Simulation-Development Environment

The Getting Started with ASCI Red page (http://www.sandia.gov/ASCI/Red/Start.htm) offers much more information on SDE tools than this document can contain. A few tools are offered here to get you started.

Paths and Environment

Default standard local.cshrc, local.login, and local.profile files are placed in the user's home directory upon creation of an account. Users are expected to copy these files to .cshrc, .login, and .profile, respectively, and modify them to suit their own needs. These files are permanently stored in /etc/skel on sasn100 and sasn101, in case users need new copies to recover from mistakenly modifying or deleting their own copies, or simply want the most recent versions. An example .cshrc file can also be found at http://www.sandia.gov/ASCI/Red/usage/cshrc.html, though the local.cshrc file is likely to be more up-to-date. The default files will set the proper paths and environment variables to allow compilation on sasn100 and sasn101, and program execution on janus and janus-s. More information on setting up the cross-compiler environment can be found at http://www.sandia.gov/ASCI/Red/usage/cross.html.

The default shell given to users is the C-shell. Users with a preference for another default shell should contact janus-help@sandia.gov to request a change.

Shared Areas for Projects/Users

In the janus and janus-s environments there are file structures to accommodate shared user/project files. Sandia application-code executables can be found in subdirectories under /Net/projects. This directory is cross-mounted from the /projects directory on sasn100 (sasn101 on the classified side). Other applications that users may wish to make available to the general user community may be placed in /usr/community on any of the machines. The /usr/community directories are not shared between janus (janus-s) and sasn100 (sasn101), nor of course between the unclassified and classified networks, so if you wish to use a utility on both unclassified platforms, for example, the relevant files must be placed on both janus and sasn100. To request the creation of a shared area in either /Net/projects or /usr/community on sasn100 or sasn101, contact tflan-help@sandia.gov. To request creation of a shared area in /usr/community on janus or janus-s, contact janus-help@sandia.gov.

Group Accounts

To request creation of a new group, or to add someone to an existing group, users must first use the MetaGroups utility. This utility can be found on Sandia's SRN home pages (http://www-irn.sandia.gov/) in the pull down menu under "Engineering and Manufacturing". For classified groups, the MetaGroup utility can be accessed from the SCN home page (https://www-isn.sandia.gov/) by choosing the "NTK Groups" button near the top of the page and then choosing the "MetaGroups Utility" link. Only a Sandia Manager can request creation of classified groups. After the requested group is created, the group owner must notify janus-help and tflan-help to include the groups on janus (janus-s) and sasn100 (sasn101).

Compilers

All compilation of codes should be performed on sasn100, sasn101, or another machine that is configured as a cross-compiler for janus. On both sasn100 and sasn101 the compilers are located in /usr/local/intel/tflop/current/tflops/bin.solaris. Although compilation can be performed on janus and janus-s, this practice is discouraged as these machines are intended for running and debugging codes. In the rare event that users need to compile on janus or janus-s (*e.g.*, because a compile server is down), the compilers can be found in /cougar/bin. If users employ the provided .cshrc file, the path variable will include the proper directories for all compilers, regardless of whether the user is on sasn100, sasn101, janus or janus-s.

The primary compilers are listed in the table below.

Language (Vendor)	Service Partition Compilers	Compute Partition Compilers
C (PGI)	icc	cicc
C++ (PGI)	iCC	ciCC
Fortran 77 (PGI)	if77	cif77
Fortran 90 (PGI)	if90	cif90

The compilers starting with "c" generate executables for the compute partition (batch or interactive compute nodes). These executables can only be run with yod (see "Running Jobs on Janus and Janus-s" section above). The service partition compilers generate serial code for use on the service nodes; however, this practice is strongly discouraged as the service nodes are not

intended to be used for computation. Instead, users should perform intensive serial calculations on other platforms designed for that purpose. Additional information on the compile environment is available from FAQ 20 at http://www.sandia.gov/ASCI/Red/UserGuide.htm. Manuals for the PGI compilers can be found at http://www.pgroup.com/ppro_docs/, and several example compilation sessions are provided from the "Getting Started" link at http://www.sandia.gov/ASCI/Red/UserGuide.htm. Users needing to compile programs to be run on sasn100 or sasn101 should use the native Sun compilers located in /opt/SUNWspro/bin.

Several versions of each compiler are available on sasn100. Older versions of each compiler can be accessed by changing "current" to "old" in the TFLOPS_XDEV environment variable. Prior to operating system (OS) upgrades, new versions of the compilers can be accessed by changing "current" to "new" in the TFLOPS_XDEV environment variable. This feature allows users to compile executables that will work with a new OS version prior to the upgrade. Once an upgrade is performed, both the new and current compilers are identical.

Compilers are not guaranteed to conform to current ANSI standards and it is unlikely that ANSI-compliant compilers will be obtained during the remaining lifetime of ASCI Red. Users are cautioned that codes that work on other machines may need to be revised in order to work properly on ASCI Red.

Common Compiler Options

As usual, the -O and -g options can be given, respectively, for optimization and generating debugging information. Both options can be given at the same time, but this can lead to misleading information from debugging tools.

Common Linker Options

The most common linker option is –lmpi, to link to the production MPI library. The production MPI library conforms to version 1 of the MPI standard and is based on Release 1.0.12 of Argonne's MPICH software.

Large File Support

Files greater than 2 GB must reside on /pfs_grande file systems.

Compiling 64-bit Programs

Janus and janus-s are 32-bit machines and do not support 64-bit applications.

Libraries

Information on math libraries and other special libraries (*e.g.*, LAPACK, PBLAS, ScaLAPACK) can be found at http://www.sandia.gov/ASCI/Red/usage/pres_mathlibs/index.htm.

Finding General Bugs

Both a command line debugger (debug) and a GUI debugger (xdebug) are available on janus and janus-s. Users are encouraged to perform debugging on janus rather than janus-s if at all possible. Janus-s has a limited number of interactive nodes and therefore is only appropriate for debugging the simplest problems. Tutorials on using the debugging tools, including example programs, can be found at http://www.sandia.gov/ASCI/Red/usage/tutorial/index.html.

TotalView is not available on janus or janus-s.

Processor-Performance Measurement—Execution-Time Profiling

Execution time profiling is available on janus in two forms: automated and manual. Automated profiling is simple to use but lacks the flexibility of manual profiling. To perform automated instrumentation of a code, simply compile the code with the –Mprof=func or –Mprof=lines option. The former provides function level profiling while the latter profiles at the ”basic block” level. The code is then run as usual, and profiling output files are put, by default, in the subdirectory pmon.out of the current directory. The user can then use the profile command to generate a readable profiling report. An introduction to automated profiling, with examples, is available at http://www.sandia.gov/ASCI/Red/usage/pres_profile/index.htm. In addition, the man page for the profile command provides a detailed description of usage and output.

Manual instrumentation is more complex, requiring modification of the user's code using function calls from the perfmon library. Users interested in this capability should consult FAQ 52 from the ASCI Red users guide (http://www.sandia.gov/ASCI/Red/UserGuide.htm) or http://www.sandia.gov/ASCI/Red/usage/tutorial/perfmon/index.html for a tutorial and examples.

Processor-Performance Measurement—Hardware-Performance Counters

The Intel Pentium® processors provide many hardware (HW) performance counters that can be accessed by users. A list of the available counters can be found at http://www.sandia.gov/ASCI/Red/usage/perfeva.htm. In order to access and use the HW counters, users must set the PROFILE_COUNTERS environment variable to the name of one or two HW counters as follows:

% setenv PROFILE_COUNTERS <counter1>

or

```
% setenv PROFILE_COUNTERS <counter1>,<counter2>
```

The user must then compile the program using the –Mprof=func or –Mprof=lines option as described in the previous subsection. After running the program, the user can enter the command

```
% profile -p
```

and the output will include data for the HW counters specified by PROFILE_COUNTERS. An example profiling session using HW counters is available at http://www.sandia.gov/ASCI/Red/usage/pres_profile/sld013.htm.

Hardware counters can also be accessed by manually instrumenting code with the genperf routines. Interested users should consult http://www.sandia.gov/ASCI/Red/usage/genperf.html for details.

General Performance Optimization and Troubleshooting

Additional information on optimizing and troubleshooting on janus and janus-s can be found at http://www.sandia.gov/ASCI/Red/usage/optimize.html. A postscript version of the same information is available at http://www.sandia.gov/ASCI/Red/usage/optimize.ps.

Pre & Post Processing Data

Pre-processing of data should only be performed on janus if the pre-processing program runs in parallel. Serial pre-processing should be performed on one of the SGI visualization servers

(edison for classified; tesla, atlantis, or discovery for unclassified), or with your local LAN resources. The pre-processed data should then be moved to janus (janus-s) prior to analysis. Users are discouraged from writing qsub scripts that gang pre- and post-processing together with the running of analysis codes, unless the pre- and post-processing programs use a similar number of nodes as the analysis code or the pre- and post-processing program execution times are very short. This policy is intended to minimize idle nodes that are allocated to an NQS job but are not in use during extended pre-and post-processing.

At the present time there are two recommendations for post-processing and visualizing data generated on janus and janus-s:

1. Manipulate the data into manageable (but large) data sets on janus and move these data sets to edison (classified) or to tesla, atlantis, or discovery (unclassified) using pftp, sftp or scp (see "File Transfer to and from Janus Platforms" below). Visualization can then be performed on these SGI machines. Alternatively, users can move data to their local visualization server.

2. Transfer all data to one of the SGI machines or to your local visualization server before any post processing.

Note that the Wide Area Network (WAN) is tuned for larger file sizes. Each file transferred to a machine causes a file to be opened or created and opened and there is overhead associated with each of these file operations. On the smss, this overhead can be prohibitive when moving thousands of small files. To manage this overhead, we recommend the use of post-processing tools such as Nemesis, which includes subprograms nem_slice, nem_spread, and nem_join. For example, nem_join may be used to "join" a large data set with many files into a small number of large files. Subsets may be "joined" based on lists and/or ranges of time indices (time steps), lists of nodal variables, and lists of elemental variables. More information on the Nemesis toolset is available at: http://www.jal.sandia.gov/SEACAS/Documentation/SEACAS.html.

File Transfer to and from Janus Platforms

The preferred methods for file movement to and from janus, janus-s, sasn100 and sasn101 are via the sftp, pftp, or scp commands. The sftp command is a secure version of the familiar ftp command, and is provided as part of the secure shell software. The use of sftp is virtually identical to the use of ftp from the user's perspective. Both janus and the remote machine to or from which files are being transferred must be running ssh for sftp to succeed. Pftp is a secure parallel version of sftp. Pftp automatically breaks up the file transfer into multiple streams (and automatically reassembles them back to the original form at the destination machine) and can greatly reduce transfer times. Parallel ftp scripts have been made available in /usr/local/bin for many hosts (scripts of the form pftp2*host* – ie, pftp2edison). Thus, where available, pftp is the recommended method for transferring large files. The scp command is a secure version of the familiar rcp command.

In the event that transfer must be made to or from a remote machine that is not running secure shell software, users can use the normal ftp or rcp commands with the understanding that any information that is transferred by either of these methods may not be entirely secure. While no man pages exist for scp, pftp, or sftp, the man pages for rcp and ftp are highly useful. Users unfamiliar with these commands should consult one of the many general Unix reference books or web tutorials on Unix.

Transfer of data to and from Windows machines can be done with the F-Secure file transfer utility that is part of the F-Secure ssh client. Once a Windows user is logged in to the desired janus platform using the F-Secure ssh client, a secure file transfer window can be opened by clicking on the "New file transfer window" icon. Files can be transferred to and from the janus platform by the normal drag and drop or copy and paste methods used by Windows.

For long term, reliable archiving of data, users are encouraged to use the Sandia Mass Storage System (smss). All four janus platforms (janus, janus-s, sasn100, and sasn101) provide an smssftp utility that enables connection and file transfer to the corresponding mass storage system. Once connected, file transfer is performed analogously to the sftp and/or pftp utilities. More information on Sandia's mass storage systems can be found at http://sc.sandia.gov/systems/site_map.shtml by clicking on the relevant smss link on the left side of the page (this web site can only be accessed from the SRN). FAQ 59 at http://www.sandia.gov/ASCI/Red/usage/faq.html provides specific guidance for use of smss with the janus platforms.

If files are intended to be placed on a /Net file system on janus (such as /Net/usr/home or /Net/scratch - cross-mounted from sasn100 or sasn101), then the effective way to transfer the files is to transfer directly to the appropriate file system on the application server. The /Net file systems are nfs-mounted from the application servers, sasn100 and sasn101.

Distribution for Asci Red for Dummies:

No.	MS	Org.	
1	0807	9328	Amdahl, Robert R.
1	0813	9311	Cahoon, Robert M.
1	0807	9328	Collins, William P.
1	0807	9328	Davidson, William M.
1	0807	9328	Davis, Michael E.
1	1110	9224	Doerfler, Douglas W.
1	1110	9224	Goudy, Sue P.
1	0801	9300	Hale, Arthur L.
1	0807	9328	Jaramillo, Frank M.
1	0807	9328	Jennings, Barbara J.
1	0806	9322	Jones, P. Carol Romero
2	1109	9224	Kelly, Suzanne M.
1	0807	9328	Korbin, John P.
1	0807	9328	Kuhns, Victor G.
1	0807	9328	Martinez, Michael A.
1	0801	9320	Mason, W. Franklin
5	0807	9328	McAllister, Paula L.
1	0807	9328	Meyer, Harold E.
2	0847	9326	Miller, Joel D.
1	0807	9328	Noe, John P.
1	0822	9326	Pavlakos, Constantine
2	1109	9224	Quinlan, Gerald F.
3	0807	9328	Rajan, Mahesh
1	0807	9328	Repik, Jason J.
1	0807	9328	Sanchez, Paul E.
2	0670	6524	Sault, Allen G.
1	0807	9328	Shirley David N.
1	0801	9330	Sjulin, Michael R.
1	0807	9328	Smith, Rosanne M.
1	0807	9328	Taylor, Sean R.
1	0805	9329	Swartz, William D.
1	0823	9324	Zepper, John D.
1	9018	8945-1	Central Technical Files
2	0899	9616	Technical Library