



Distributed Make (dmake)

Sun[™] Studio 8

Sun Microsystems, Inc.
www.sun.com

Part No. 817-6097-10
March 2004, Revision A

Submit comments about this document at: <http://www.sun.com/hwdocs/feedback>

Copyright © 2004 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

U.S. Government Rights - Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements. Use is subject to license terms.

This distribution may include materials developed by third parties.

Parts of the product may be derived from Berkeley BSD systems, licensed from the University of California. UNIX is a registered trademark in the U.S. and in other countries, exclusively licensed through X/Open Company, Ltd.

Sun, Sun Microsystems, the Sun logo, Java, and JavaHelp are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the U.S. and other countries. Products bearing SPARC trademarks are based upon architecture developed by Sun Microsystems, Inc.

This product is covered and controlled by U.S. Export Control laws and may be subject to the export or import laws in other countries. Nuclear, missile, chemical biological weapons or nuclear maritime end uses or end users, whether direct or indirect, are strictly prohibited. Export or reexport to countries subject to U.S. embargo or to entities identified on U.S. export exclusion lists, including, but not limited to, the denied persons and specially designated nationals lists is strictly prohibited.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright © 2004 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, Etats-Unis. Tous droits réservés.

L'utilisation est soumise aux termes de la Licence.

Cette distribution peut comprendre des composants développés par des tierces parties.

Des parties de ce produit pourront être dérivées des systèmes Berkeley BSD licenciés par l'Université de Californie. UNIX est une marque déposée aux Etats-Unis et dans d'autres pays et licenciée exclusivement par X/Open Company, Ltd.

Sun, Sun Microsystems, le logo Sun, Java, et JavaHelp sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays. Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

Ce produit est soumis à la législation américaine en matière de contrôle des exportations et peut être soumis à la réglementation en vigueur dans d'autres pays dans le domaine des exportations et importations. Les utilisations, ou utilisateurs finaux, pour des armes nucléaires, des missiles, des armes biologiques et chimiques ou du nucléaire maritime, directement ou indirectement, sont strictement interdites. Les exportations ou réexportations vers les pays sous embargo américain, ou vers des entités figurant sur les listes d'exclusion d'exportation américaines, y compris, mais de manière non exhaustive, la liste de personnes qui font objet d'un ordre de ne pas participer, d'une façon directe ou indirecte, aux exportations des produits ou des services qui sont régis par la législation américaine en matière de contrôle des exportations et la liste de ressortissants spécifiquement désignés, sont rigoureusement interdites.

LA DOCUMENTATION EST FOURNIE "EN L'ÉTAT" ET TOUTES AUTRES CONDITIONS, DECLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES, DANS LA MESURE AUTORISEE PAR LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE A LA QUALITE MARCHANDE, A L'APTITUDE A UNE UTILISATION PARTICULIERE OU A L'ABSENCE DE CONTREFAÇON.



Adobe PostScript

Contents

Before You Begin 5

Typographic Conventions 5

Shell Prompts 6

Accessing Sun Studio Software and Man Pages 6

Accessing Compilers and Tools Documentation 9

Accessing Related Solaris Documentation 11

Resources for Developers 11

Contacting Sun Technical Support 12

Sending Your Comments 12

Distributed Make (dmake) 13

Synopsis 13

Description 13

Options 14

Usage 16

 Special-purpose Targets 16

 Controlling dmake Jobs 17

 The dmake Host 17

 User Commands 19

 The Build Server 20

Environment/Macros 21

Files 22

See Also 23

Before You Begin

Distributed Make (`dmake`) describes the Sun Studio 8 `dmake` command. `dmake` parses your makefiles and determines the targets that can be built concurrently, and distributes the build of those targets over a number of hosts set by you.

The information in this document supplements the `make` utility man page.

Typographic Conventions

TABLE P-1 Typeface Conventions

Typeface	Meaning	Examples
AaBbCc123	The names of commands, files, and directories; on-screen computer output	Edit your <code>.login</code> file. Use <code>ls -a</code> to list all files. <code>% You have mail.</code>
AaBbCc123	What you type, when contrasted with on-screen computer output	<code>% su</code> Password:
<i>AaBbCc123</i>	Book titles, new words or terms, words to be emphasized	Read Chapter 6 in the <i>User's Guide</i> . These are called <i>class</i> options. You <i>must</i> be superuser to do this.
<code>AaBbCc123</code>	Command-line placeholder text; replace with a real name or value	To delete a file, type rm <i>filename</i> .

TABLE P-2 Code Conventions

Code Symbol	Meaning	Notation	Code Example
[]	Brackets contain arguments that are optional.	<code>O[n]</code>	<code>O4, O</code>
{ }	Braces contain a set of choices for a required option.	<code>d{y n}</code>	<code>dy</code>
	The “pipe” or “bar” symbol separates arguments, only one of which may be chosen.	<code>B{dynamic static}</code>	<code>Bstatic</code>
:	The colon, like the comma, is sometimes used to separate arguments.	<code>Rdir[:dir]</code>	<code>R/local/libs:/U/a</code>
...	The ellipsis indicates omission in a series.	<code>xinline=f1[,...fn]</code>	<code>xinline=alpha,dos</code>

Shell Prompts

Shell	Prompt
C shell	<i>machine-name%</i>
C shell superuser	<i>machine-name#</i>
Bourne shell and Korn shell	<code>\$</code>
Superuser for Bourne shell and Korn shell	<code>#</code>

Accessing Sun Studio Software and Man Pages

The compilers and tools and their man pages are not installed into the standard `/usr/bin/` and `/usr/share/man` directories. To access the compilers and tools, you must have your `PATH` environment variable set correctly (see [“Accessing the](#)

[Compilers and Tools” on page 7](#)). To access the man pages, you must have the your MANPATH environment variable set correctly (see [“Accessing the Man Pages” on page 8](#)).

For more information about the PATH variable, see the `csh(1)`, `sh(1)`, and `ksh(1)` man pages. For more information about the MANPATH variable, see the `man(1)` man page. For more information about setting your PATH variable and MANPATH variables to access this release, see the installation guide or your system administrator.

Note – The information in this section assumes that your Sun Studio compilers and tools are installed in the `/opt` directory. If your software is not installed in the `/opt` directory, ask your system administrator for the equivalent path on your system.

Accessing the Compilers and Tools

Use the steps below to determine whether you need to change your PATH variable to access the compilers and tools.

▼ To Determine Whether You Need to Set Your PATH Environment Variable

1. **Display the current value of the PATH variable by typing the following at a command prompt.**

```
% echo $PATH
```

2. **Review the output to find a string of paths that contain `/opt/SUNWspro/bin/`.**

If you find the path, your PATH variable is already set to access the compilers and tools. If you do not find the path, set your PATH environment variable by following the instructions in the next procedure.

▼ To Set Your PATH Environment Variable to Enable Access to the Compilers and Tools

1. **If you are using the C shell, edit your home `.cshrc` file. If you are using the Bourne shell or Korn shell, edit your home `.profile` file.**
2. **Add the following to your PATH environment variable. If you have Sun ONE Studio software or Forte Developer software installed, add the following path before the paths to those installations.**

```
/opt/SUNWspro/bin
```

Accessing the Man Pages

Use the following steps to determine whether you need to change your `MANPATH` variable to access the man pages.

▼ To Determine Whether You Need to Set Your `MANPATH` Environment Variable

1. Request the `dbx` man page by typing the following at a command prompt.

```
% man dbx
```

2. Review the output, if any.

If the `dbx(1)` man page cannot be found or if the man page displayed is not for the current version of the software installed, follow the instructions in the next procedure for setting your `MANPATH` environment variable.

▼ To Set Your `MANPATH` Environment Variable to Enable Access to the Man Pages

1. If you are using the C shell, edit your home `.cshrc` file. If you are using the Bourne shell or Korn shell, edit your home `.profile` file.
2. Add the following to your `MANPATH` environment variable.

```
/opt/SUNWspro/man
```

Accessing the Integrated Development Environment

The Sun Studio 8 integrated development environment (IDE) provides modules for creating, editing, building, debugging, and analyzing the performance of a C, C++, or Fortran application.

The IDE requires the Core Platform component of Sun Studio 8. You must set the `SPRO_NETBEANS_HOME` environment variable to the location where the Core Platform component is installed or mounted (*installation_directory/netbeans/3.5R*) if the Core Platform component is not installed or mounted to one of the following locations:

- The default installation directory `/opt/netbeans/3.5R`

- The same location as the Compilers and Tools component of the Sun Studio 8 (for example, the Compilers and Tools component installed in `/foo/SUNWspro` and the Core Platform component in `/foo/netbeans/3.5R`)

The command to start the IDE is `sunstudio`. For details on this command, see the `sunstudio(1)` man page.

Accessing Compilers and Tools Documentation

You can access the documentation at the following locations:

- The documentation is available from the documentation index that is installed with the software on your local system or network at `file:/opt/SUNWspro/docs/index.html`.

If your software is not installed in the `/opt` directory, ask your system administrator for the equivalent path on your system.

- Most manuals are available from the `docs.sun.comsm` web site. The following titles are available through your installed software only:
 - *Standard C++ Library Class Reference*
 - *Standard C++ Library User's Guide*
 - *Tools.h++ Class Library Reference*
 - *Tools.h++ User's Guide*
- The release notes are available from the `docs.sun.com` web site.
- Online help for all components of the IDE is available through the Help menu, as well as through Help buttons on many windows and dialogs, in the IDE.

The `docs.sun.com` web site (<http://docs.sun.com>) enables you to read, print, and buy Sun Microsystems manuals through the Internet. If you cannot find a manual, see the documentation index that is installed with the software on your local system or network.

Note – Sun is not responsible for the availability of third-party web sites mentioned in this document and does not endorse and is not responsible or liable for any content, advertising, products, or other materials on or available from such sites or resources. Sun will not be responsible or liable for any damage or loss caused or alleged to be caused by or in connection with use of or reliance on any such content, goods, or services available on or through any such sites or resources.

Documentation in Accessible Formats

The documentation is provided in accessible formats that are readable by assistive technologies for users with disabilities. You can find accessible versions of documentation as described in the following table. If your software is not installed in the `/opt` directory, ask your system administrator for the equivalent path on your system.

Type of Documentation	Format and Location of Accessible Version
Manuals (except third-party manuals)	HTML at http://docs.sun.com
Third-party manuals: <ul style="list-style-type: none">• <i>Standard C++ Library Class Reference</i>• <i>Standard C++ Library User's Guide</i>• <i>Tools.h++ Class Library Reference</i>• <i>Tools.h++ User's Guide</i>	HTML in the installed software through the documentation index at <code>file:/opt/SUNWspro/docs/index.html</code>
Readmes and man pages	HTML in the installed software through the documentation index at <code>file:/opt/SUNWspro/docs/index.html</code>
Online help	HTML available through the Help menu in the IDE
Release notes	HTML at http://docs.sun.com

Related Compilers and Tools Documentation

The following table describes related documentation that is available at `file:/opt/SUNWspro/docs/index.html` and <http://docs.sun.com>. If your software is not installed in the `/opt` directory, ask your system administrator for the equivalent path on your system.

Document Title	Description
<i>Numerical Computation Guide</i>	Describes issues regarding the numerical accuracy of floating-point computations.

Accessing Related Solaris Documentation

The following table describes related documentation that is available through the `docs.sun.com` web site.

Document Collection	Document Title	Description
Solaris Reference Manual Collection	See the titles of man page sections.	Provides information about the Solaris operating environment.
Solaris Software Developer Collection	<i>Linker and Libraries Guide</i>	Describes the operations of the Solaris link-editor and runtime linker.
Solaris Software Developer Collection	<i>Multithreaded Programming Guide</i>	Covers the POSIX and Solaris threads APIs, programming with synchronization objects, compiling multithreaded programs, and finding tools for multithreaded programs.

Resources for Developers

Visit <http://developers.sun.com/prodtech/cc> to find these frequently updated resources:

- Articles on programming techniques and best practices
- A knowledge base of short programming tips
- Documentation of compilers and tools components, as well as corrections to the documentation that is installed with your software
- Information on support levels
- User forums
- Downloadable code samples
- New technology previews

You can find additional resources for developers at <http://developers.sun.com>.

Contacting Sun Technical Support

If you have technical questions about this product that are not answered in this document, go to:

<http://www.sun.com/service/contacting>

Sending Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. Email your comments to Sun at this address:

`docfeedback@sun.com`

Please include the part number (817-6097-10) of your document in the subject line of your email.

Distributed Make (dmake)

Synopsis

```
dmake [-c dmake_rcfile] [-g dmake_group] [-j dmake_max_jobs]
      [-m serial | parallel | distributed] [-o dmake_odir]
```

Description

The information in this document supplements the `make` utility man page.

Distributed Make (`dmake`) parses your makefiles and determines which target can be built concurrently, and distributes the build of those targets over a number of hosts set by you.

If you use the standard `make` utility, the transition to `dmake` requires little if any alteration to your makefiles. `dmake` is a superset of the `make` utility. With nested makes, if a top-level makefile calls "`make`", you need to use `$ (MAKE)`.

You execute `dmake` on a "dmake host" and distribute "jobs" to "build servers."

You can also distribute jobs to the `dmake` host, in which case the `dmake` host also becomes a build server. `dmake` distributes jobs based on makefile targets that `dmake` determines (based on your makefiles) can be built concurrently.

You can use any machine as a build server as long as it meets the following requirements:

- From the `dmake` host (the machine you are using) you must be able to use `rsh`, without being prompted for a password, to remotely execute commands on the build server.

For example:

```
demo% rsh <machine_name> which dmake  
/opt/SUNWspro/bin/dmake
```

The `rsh` must be "clean", with no additional output.

- The bin directory in which the `dmake` software is installed must be accessible from the build server. By default, `dmake` assumes that the logical path to the `dmake` executables on the build server is the same as the `dmake` host. This assumption can be overridden by specifying a path name as an attribute of the host entry in the runtime configuration file.

From the `dmake` host you can control which build servers are used and how many `dmake` jobs are allotted to each build server.

The number of `dmake` jobs that can run on a given build server can also be limited on that server.

Options

`-c dmake_rcfile`

Specifies an alternate runtime configuration file.

The default runtime configuration file is `$(HOME) / .dmake.rc`.

`-g dmake_group`

Specifies the name of the build server group to which jobs are distributed.

You define server groups in the runtime configuration file.

The default server group is the first group in the runtime configuration file.

`-j dmake_max_jobs`

Specifies the maximum total number of jobs that are distributed to the specified group of build servers in the runtime configuration file.

The default maximum number of jobs is the sum of all the specified jobs in a build server group.

The jobs are subtracted from, or added to, hosts by 1 in the order they appear in the runtime configuration file.

For example, if all jobs specified in the runtime configuration file total 8:

```
host earth { jobs = 3 }
host mars  { jobs = 5 }
```

and `dmake_max_jobs` is specified as 11, `dmake` adds three more jobs to the current total maximum number of jobs (which is eight) as follows:

```
host earth { jobs = 5 }
host mars  { jobs = 6 }
```

Also, if `dmake_max_jobs` is specified as 4, `dmake` subtracts 4 jobs (from the original eight) as follows:

```
host earth { jobs = 1 }
host mars  { jobs = 3 }
```

`-m {serial | parallel | distributed}`

Specify one of the following key words:

- `serial`: Causes `dmake` to behave like the standard serial version of `make`.
- `parallel`: Causes `dmake` to distribute jobs to only the `dmake` host.
- `distributed`: Causes `dmake` to behave in fully distributed mode. This is the `dmake` default.

`-o dmake_odir`

Specifies a common physical directory that `dmake` can write temporary output files to and read temporary output files from. The directory used is `$(HOME) / .dmake` and this or whichever directory is specified, must be visible to all build servers.

Use this option only if the `$(HOME)` directory on your local host and the `$(HOME)` directory on all of your remote hosts are not the same physical `$(HOME)` directory.

For example, a root user would use this option.

Note – Note: These options and the environment variables and makefile macros described later in this man page modify the same behavior.

Their order of precedence is defined as:

- Command-line options
- Makefile macros
- Environment variables
- dmake default

Usage

Special-purpose Targets

dmake allows targets to be built concurrently on a number of build servers. Concurrent processing can greatly reduce the time required to build a large system or project. dmake supplies the special makefile targets for controlling concurrency and timing.

- .PARALLEL
- .NO_PARALLEL
- .LOCAL
- .WAIT

.NO_PARALLEL

Use this target to indicate which targets are to be processed serially.

.PARALLEL

Use this target to indicate which targets are to be processed in parallel.

.LOCAL

Use this target to indicate which targets are to be processed serially on the local host.

.WAIT

When you specify this target in a dependency list, dmake waits until the dependencies that precede it are finished before processing those that follow, even when processing is parallel.

Makefiles that you write using these targets remain compatible with the standard version of make distributed with Solaris 1.x and Solaris 2.x. Standard make accepts these targets without error (and without action).

Controlling dmake Jobs

The distribution of dmake jobs is controlled in two ways:

1. A dmake user on a dmake host can specify the machines they want to use as build servers and the number of jobs they want to distribute to each build server.
2. The owner (a user that can alter the `/etc/opt/SPROdmake/dmake.conf` file) on a build server can control the maximum total number of dmake jobs that can be distributed to that build server.

The dmake Host

When dmake begins execution it searches for a runtime configuration file to know where to distribute jobs.

Generally, this file is located in your home directory on the dmake host and is named `dmakerc`.

dmake searches for the runtime configuration file in the following locations and in the following order:

1. The path name you specify on the command line using the `-c` option
2. The path name you specify using the `DMAKE_RCFILE` makefile macro
3. The path name you specify using the `DMAKE_RCFILE` environment variable
4. In `$(HOME)/.dmakerc`

If a runtime configuration file is not found, dmake switches to parallel mode and distributes two jobs (the default) to the dmake host. You can change this using the `-j` option, or `DMAKE_MAX_JOBS`.

The runtime configuration file may contain a list of build servers and the number of jobs you want distributed to each build server.

The following is a sample of a simple runtime configuration file:

```
# My machine. This entry causes dmake to distribute to it
falcon { jobs = 1 }
hawk
eagle { jobs = 3 }
# Manager's machine. She's usually at meetings
heron { jobs = 4 }
avocet
```

The entries: falcon, hawk, eagle, heron, and avocet are listed as build servers.

You can specify the number of jobs you want distributed to each build server. The default number of jobs is two.

Any line that begins with the "#" character is interpreted as a comment.

Note – This list of build servers includes falcon which is also the dmake host. The dmake host can also be specified as a build server. If you do not include it in the runtime configuration file, no dmake jobs are distributed to it.

You can also construct groups of build servers in the runtime configuration file.

This provides you with the flexibility of easily switching between different groups of build servers as circumstances warrant. For instance you may define a different group of build servers for builds under different operating systems, or on groups of build servers that have special software installed on them. The build servers must be all the same architecture and have the same SunOS installed.

The following runtime configuration file contains groups:

```
earth                { jobs = 2 }
mars                 { jobs = 3 }

group lab1 {
    host falcon{ jobs = 3 }
    host hawk
    host eagle{ jobs = 3 }
}

group lab2 {
    host heron
    host avocet{ jobs = 3 }
    host stilt{ jobs = 2 }
}

group labs {
    group lab1
    group lab2
}

group sunos5.x {
    group labs
    host jupiter
    host venus{ jobs = 2 }
    host pluto { jobs = 3 }
}
```

User Commands

Formal groups are specified by the "group" directive and lists of their constituents are delimited by braces ({}).

Build servers that are constituents of groups are specified by the optional "host" directive.

Groups can be constituents of other groups.

Individual build servers can be listed in runtime configuration files that also contain groups of build servers. In this case `dmake` treats these build servers as constituents of the unnamed group.

dmake distributes jobs to a single group of hosts specified by the following list and in precedence from 1 to 4.

1. The group specified on the command-line as an argument to the `-g` option.
2. The group specified by the `DMAKE_GROUP` makefile macro.
3. The group specified by the `DMAKE_GROUP` environment variable.
4. The first formal group listed in the runtime configuration file.

The names of groups and hosts specified in the runtime configuration file may be enclosed in double quotes. This is to allow more flexibility with respect to the character sequences that may appear as part of the group and host names. For example, if the name of the group starts with a digit it should be double-quoted:

```
group "123_sparc"
```

As mentioned above, the `bin` directory in which the dmake software is installed must be accessible from the build server. By default, dmake assumes that the logical path to the dmake executables on the build server is the same as the dmake host. This assumption can be overridden by specifying a path name as an attribute of the host entry in the runtime configuration file. For example:

```
group sparc-cluster {  
    host wren    { jobs = 10 , path = "/export/SUNWspro/bin" }  
    host stimpay { path = "/opt/SUNWspro/bin" }  
}
```

The Build Server

The `/etc/opt/SPROdmake/dmake.conf` file is located in the file system of build servers.

Use this file to specify the following:

- Required: The maximum total number of dmake jobs (from all users) that can run concurrently on that build server.
- Optional: The `/usr/bin/` priority under which all dmake jobs are to be run.

The following is a sample of a `dmake.conf` file:

```
max_jobs: 8  
nice_prio: 5
```

This file sets the maximum number of `dmake` jobs permitted to run on that build server (from all `dmake` users) to be eight. You can change the priority of the jobs to be run by using the `nice_prio` command. See `nice(1)`.

Note – If the `/etc/opt/SPROdmake/dmake.conf` file does not exist on a build server, no `dmake` jobs will be allowed to run on that server.

Environment/Macros

The following can be defined as either environment variables or makefile macros:

`DMAKE_RCFILE`

Defines an alternate runtime configuration file. The default runtime configuration file is `$(HOME)/.dmakerc`.

`DMAKE_GROUP`

Defines the name of the build server group to which jobs are distributed. Server groups are defined in the runtime configuration file. The default server group is the first group in the runtime configuration file.

`DMAKE_MAX_JOBS`

Defines the maximum total number of jobs that are distributed to the specified group of build servers in the runtime configuration file. The default maximum number of jobs is the sum of all the specified jobs in a build server group. The jobs are subtracted from, or added to, hosts by 1 in the order they appear in the runtime configuration file.

See the `-j` option in this man page for an example.

DMAKE_ADJUST_MAX_JOBS

May contain one of the following key words:

- YES: Allows dmake to adjust the limit of parallel jobs according to the current loading of the system. If the system is not overloaded, dmake will use the limit defined by the user. If the system is overloaded, dmake will set the "current" limit to less than the limit defined by the user.

If this variable is not set, dmake will adjust the limit of parallel jobs according to the current loading of the system. This is the dmake default.

- NO: Causes dmake to switch off the autoadjustment mechanism.

DMAKE_MODE

May contain one of the following key words:

- serial: Causes dmake to behave like the standard serial version of make.
- parallel: Causes dmake to distribute jobs to only the dmake host.
- distributed: Causes dmake to behave in fully distributed mode. This is the dmake default.

DMAKE_ODIR

Defines a common physical directory that dmake can write temporary output files to and read temporary output files from.

Use this environment variable, or macro, only if the `$(HOME)` directory on your local host and the `$(HOME)` directory on all of your remote hosts are not the same physical `$(HOME)` directory.

For example, a root user would use this option.

Files

- `$(HOME)/.dmakerc`: The default runtime configuration file. Contains the names of build servers and groups of build servers.
- `/etc/opt/SPROdmake/dmake.conf`: Located on build servers, this file is used to specify the maximum total number of jobs that can be distributed to it by all dmake users. It is also used to specify the `/usr/bin/nice` priority all dmake jobs are to be run under.

See Also

Access the complete Sun Studio 8 documentation set by pointing an HTML browser at `/opt/SUNWspro/docs/index.html`

If the compilers are not installed in the default `/opt` directory, ask your system administrator for the equivalent path on your system.

Consult the following man pages for additional information.

- `make(1)`
- `rsh(1)`
- `hosts(4)`
- `hosts.equiv(4)`
- `attributes(5)`
- `largefile(5)`

