Solaris Tunable Parameters Reference Manual

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 817–0404–10
January 2005

Adobe PostScript

041005@9495

Contents

Preface

The *Solaris Tunable Parameters Reference Manual* provides reference information about Solaris™ OS kernel and network tunable parameters. This manual does not provide tunable parameter information about the CDE, GNOME, or Java™ environments.

This manual contains information for both SPARC® based and x86 based systems.

Note – This Solaris release supports systems that use the SPARC and x86 families of processor architectures: UltraSPARC®, SPARC64, AMD64, Pentium, and Xeon EM64T. The supported systems appear in the *Solaris 10 Hardware Compatibility List* at http://www.sun.com/bigadmin/hcl. This document cites any implementation differences between the platform types.

In this document the term "x86" refers to 64-bit and 32-bit systems manufactured using processors compatible with the AMD64 or Intel Xeon/Pentium product families. For supported systems, see the *Solaris 10 Hardware Compatibility List*.

Who Should Use This Book

This book is intended for experienced Solaris system administrators who might need to change kernel tunable parameters in certain situations. For guidelines on changing Solaris tunable parameters, refer to "Tuning a Solaris System" on page 22.

How This Book Is Organized

The following table describes the chapters and appendixes in this book.

Chapter	Description
Chapter 1	An overview of tuning a Solaris system. Also provides a description of the format used in the book to describe the kernel tunables.
Chapter 2	A description of Solaris kernel tunables such as kernel memory, file system, process size, and paging parameters.
Chapter 3	A description of NFS tunables such as caching symbolic links, dynamic retransmission, and RPC security parameters.
Chapter 4	A description of TCP/IP tunables such as IP forwarding, source routing, and buffer-sizing parameters.
Chapter 5	A description of tunable parameters for the Network Cache and Accelerator (NCA).
Chapter 6	A description of parameters used to set default values of certain system facilities. Changes are made by modifying files in the `/etc/default` directory.
Appendix A	A history of parameters that have changed or are now obsolete.
Appendix B	A history of this manual's revisions including the current Solaris release.

Related Books

The following books provide background material that might be useful when you tune Solaris systems.

- *Configuration and Capacity Planning for Solaris Servers* by Brian L. Wong, Sun Microsystems Press, ISBN 0-13-349952–9
- *NFS Illustrated* by Brent Callaghan, Addison Wesley, ISBN 0-201-32570-5
- *Resource Management* by Richard McDougall, Adrian Cockcroft, Evert Hoogendoorn, Enrique Vargas, Tom Bialaski, Sun Microsystems Press, ISBN 0-13-025855-5

- *Sun Performance and Tuning: SPARC and Solaris* by Adrian Cockcroft, Sun Microsystems Press/PRT Prentice Hall, ISBN 0-13-149642-3

Other Resources for Solaris Tuning Information

This table describes other resources for Solaris tuning information.

Tuning Resource	For More Information
Performance tuning classes	http://suned.sun.com
Online performance tuning information	http://www.sun.com/sun-on-net/performance
Ordering performance tuning documentation by Sun Microsystems Press	http://www.sun.com/books/blueprints.series.html

Typographic Conventions

The following table describes the typographic changes that are used in this book.

TABLE P–1 Typographic Conventions

Typeface or Symbol	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`

`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`**
		`Password:`

AaBbCc123	Command-line placeholder: replace with a real name or value	To delete a file, type **`rm`** *filename*.

TABLE P–1 Typographic Conventions *(Continued)*

Typeface or Symbol	Meaning	Example
AaBbCc123	Book titles, new terms, or terms to be emphasized	Read Chapter 6 in *User's Guide*. These are called *class* options. You must be *root* to do this.

Shell Prompts in Command Examples

The following table shows the default system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

TABLE P–2 Shell Prompts

Shell	Prompt
C shell prompt	`machine_name%`
C shell superuser prompt	`machine_name#`
Bourne shell and Korn shell prompt	`$`
Bourne shell and Korn shell superuser prompt	`#`

CHAPTER 1

Overview of Solaris System Tuning

This section provides overview information about the format of the tuning information in this manual. This section also describes the different ways to tune a Solaris system.

What's New in Solaris System Tuning?

Default Stack Size

A new parameter, default_stksize, specifies the default stack size of all threads, kernel or user. The lwp_default_stksize parameter is still available, but it does not affect all kernel stacks. If default_stksize is set, it overrides lwp_default_stksize. For more information, see "default_stksize" on page 31.

System V IPC Configuration

In this Solaris release, all System V IPC facilities are either automatically configured or can be controlled by resource controls. Facilities that can be shared are memory, message queues, and semaphores.

Resource controls allow IPC settings to be made on a per-project or per-user basis on the local system or in a name service environment.

In previous Solaris releases, IPC facilities were controlled by kernel tunables. You had to modify the /etc/system file and reboot the system to change the default values for these facilities.

Because the IPC facilities are now controlled by resource controls, their configuration can be modified while the system is running.

Many applications that previously required system tuning to function might now run without tuning because of increased defaults and the automatic allocation of resources.

The following table identifies the now obsolete IPC tunables and their replacement resource controls.

Resource Control	Obsolete Tunable	Old Default Value	Maximum Value	New Default Value
process.max-msg-qbytes	msginfo_msgmnb	4096	ULONG_MAX	65536
process.max-msg-messages	msginfo_msgtql	40	UINT_MAX	8192
process.max-sem-ops	seminfo_semopm	10	INT_MAX	512
process.max-sem-nsems	seminfo_semmsl	25	SHRT_MAX	512
project.max-shm-memory	shminfo_shmmax	0x800000	UINT64_MAX	1/4 of physical memory
project.max-shm-ids	shminfo_shmmni	100	2^{24}	128
project.max-msg-ids	msginfo_msgmni	50	2^{24}	128
project.max-sem-ids	seminfo_semmni	10	2^{24}	128

Obsolete parameters can still be included in the /etc/system file on a Solaris system. If so, the parameters are used to initialize the default resource control values as in previous Solaris releases. For more information, see "Parameters That Are Obsolete or Have Been Removed" on page 189. However, using the obsolete parameters is not recommended.

The following related parameters have been removed. If these parameters are included in the /etc/system file on a Solaris system, the parameters are commented out.

```
semsys:seminfo_semmns                 semsys:seminfo_semvmx
```

semsys:seminfo_semmnu

semsys:seminfo_semume

semsys:seminfo_semmap

shmsys:shminfo_shmmin

msgsys:msginfo_msgseg

msgsys:msginfo_msgmax

semsys:seminfo_semaem

semsys:seminfo_semusz

shmsys:shminfo_shmseg

msgsys:msginfo_msgmap

msgsys:msginfo_msgssz

For the current list of available resource controls, see rctladm(1M). For information about configuring resource controls, see project(4).

NFSv4 Parameters

The following parameters for the NFSv4 protocol are included in this release:

- "nfs:nfs4_pathconf_disable_cache" on page 94
- "nfs:nfs4_cots_timeo" on page 97
- "nfs:nfs4_do_symlink_cache" on page 99
- "nfs:nfs4_dynamic" on page 101
- "nfs:nfs4_lookup_neg_cache" on page 103
- "nfs:nfs4_max_threads" on page 105
- "nfs:nfs4_nra" on page 107
- "nfs:nfs4_shrinkreaddir" on page 111
- "nfs:nfs4_bsize" on page 115
- "nfs:nfs4_async_clusters" on page 117
- "nfs:nfs4_max_transfer_size" on page 122

For information about NFSv4 parameters, see "NFS Module Parameters" on page 94.

New and Changed TCP/IP Parameters

The following IP parameters have been added in this Solaris release:

- "ip_squeue_worker_wait" on page 138
- "ip_squeue_write" on page 138
- "ip_squeue_fanout" on page 139
- "ipcl_conn_hash_size" on page 151

The following TCP parameters are new in this Solaris release:

- "tcp_rst_sent_rate_enabled" on page 150
- "tcp_rst_sent_rate" on page 150
- "tcp_mdt_max_pbufs" on page 151

The following TCP/IP parameters are obsolete in this Solaris release.

- ipc_tcp_conn_hash_size
- tcp_compression_enabled
- tcp_conn_hash_size
- ip_forwarding
- ip6_forwarding
- xxx_forwarding

IP Forwarding Changes

In this Solaris release, IP forwarding is enabled or disabled by using the routeadm command or the ifconfig commands instead of setting the following tunable parameters with the ndd command:

- ip_forwarding
- ip6_forwarding
- xxx_forwarding

Using the routeadm command and the ifconfig command instead of the ndd command to set IP forwarding provides the following advantages:

- All settings are persistent across reboots
- The new ifconfig router and -router commands can be placed in the /etc/hostname.*interface* files, along with other ifconfig commands that are run when the interface is initially configured.

To enable IPv4 or IPv6 packet forwarding on all interfaces of a system, you would use the following commands:

```
# routeadm -e ipv4-forwarding

# routeadm -e ipv6-forwarding
```

To disable IPv4 or IPv6 packet forwarding on all interfaces of a system, you would use the following commands:

```
# routeadm -d ipv4-forwarding

# routeadm -d ipv6-forwarding
```

In previous Solaris releases, you would enable IPv4 or IPv6 packet forwarding on all interfaces of a system as follows:

```
# ndd -set /dev/ip ip_forwarding 1

# ndd -set /dev/ip ip6_forwarding 1
```

In previous Solaris releases, you would disable IPv4 or IPv6 packet forwarding on all interfaces of a system as follows:

```
# ndd -set /dev/ip ip_forwarding 0
# ndd -set /dev/ip ip6_forwarding 0
```

If you want to enable IP forwarding on a specific IPv4 interface or IPv6 interface, you would use syntax similar to the following for your interface. The `bge0` interface is used an as example.

```
# ifconfig bge0 router
```

```
# ifconfig bge0 inet6 router
```

If you want to disable IP forwarding on a specific IPv4 interface or IPv6 interface, you would use syntax similar to the following for your interface. The `bge0` interface is used an as example.

```
# ifconfig bge0 -router
```

```
# ifconfig bge0 inet6 -router
```

Previously, IP forwarding was enabled on a specific interface as follows:

```
# ndd -set /dev/ip bge0:ip_forwarding 1
```

```
# ndd -set /dev/ip bge0:ip_forwarding 1
```

Previously, IP forwarding on a specific interface was disabled as follows:

```
# ndd -set /dev/ip ip_forwarding 0
```

```
# ndd -set /dev/ip ip6_forwarding 0
```

If you want any of the preceding `routeadm` settings to take effect on the running system, use the following command:

```
# routeadm -u
```

For more information, see `routeadm`(1M) and `ifconfig`(1M).

SPARC: Translation Storage Buffer (TSB) Parameters

New parameters for tuning Translation Storage Buffer (TSB) are included in this release. For information about TSB parameters, see "Sun-4u Specific Parameters" on page 85.

SCTP Tunable Parameters

Stream Control Transmission Protocol (SCTP), a reliable transport protocol that provides services similar to the services provided by TCP, is provided in this Solaris release. For more information about SCTP tunable parameters, see "SCTP Tunable Parameters" on page 157.

Tuning a Solaris System

The Solaris OS is a multi-threaded, scalable UNIX® operating system that runs on SPARC and x86 processors. It is self-adjusting to system load and demands minimal tuning. In some cases, however, tuning is necessary. This book provides details about the officially supported kernel tuning options available for the Solaris OS.

The Solaris kernel is composed of a core portion, which is always loaded, and a number of loadable modules that are loaded as references are made to them. Many variables referred to in the kernel portion of this guide are in the core portion. However, a few variables are located in loadable modules.

A key consideration in system tuning is that setting system parameters (or system variables) is often the least effective action that can be done to improve performance. Changing the behavior of the application is generally the most effective tuning aid available. Adding more physical memory and balancing disk I/O patterns are also useful. In a few rare cases, changing one of the variables described in this guide will have a substantial effect on system performance.

Remember that one system's `/etc/system` settings might not be applicable, either wholly or in part, to another system's environment. Carefully consider the values in the file with respect to the environment in which they will be applied. Make sure that you understand the behavior of a system before attempting to apply changes to the system variables that are described here.

Caution – The tunable parameters described in this book can and do change from release to release. A release is either a Solaris Update release or a new release such as Solaris 10. Publication of these tunable parameters does not preclude changes to the tunable parameters and their descriptions without notice.

Tuning Format of Tunable Parameters Descriptions

The format for the description of each tunable parameter is as follows:

- Parameter Name
- Description
- Data Type
- Default

- Range
- Units
- Dynamic?
- Validation
- Implicit
- When to Change
- Commitment Level
- Change History

Parameter Name

Is the exact name that is typed in the `/etc/system` file, or found in the `/etc/default/`*facility* file.

Most parameters names are of the form *parameter* where the parameter name does not contain a colon (`:`). These names refer to variables in the core portion of the kernel. If the name does contain a colon, the characters to the left of the colon reference the name of a loadable module. The name of the parameter within the module consists of the characters to the right of the colon. For example:

module_name`:`*variable*

Description

Briefly describes what the parameter does or controls.

Data Type

Indicates the signed or unsigned short integer or long integer with the following distinctions:

- On a system that runs a 32-bit kernel, a long integer is the same size as an integer.
- On a system that runs a 64-bit kernel, a long integer is twice the width in bits as an integer. For example, an unsigned integer = 32 bits, an unsigned long integer = 64 bits.

Units

(Optional) Describes the unit type.

Default

What the system uses as the default value.

Range

Specifies the possible range allowed by system validation or the bounds of the data type.

- MAXINT – A shorthand description for the maximum value of a signed integer (2,147,483,647)
- MAXUINT – A shorthand description for the maximum value of an unsigned integer (4,294,967,295)

Dynamic?

Yes, if the parameter can be changed on a running system with the `mdb` or `kmdb` debugger. No, if the parameter is a boot time initialization only.

Validation

Checks that the system applies to the value of the variable either as specified in the `/etc/system` file or the default value, as well as when the validation is applied.

Implicit
(Optional) Provides unstated constraints that might exist on the parameter, especially in relation to other parameters.

When to Change
Explains why someone might want to change this value. Includes error messages or return codes.

Commitment Level
Identifies the stability of the interface. Many of the parameters in this manual are still evolving and are classified as unstable. For more information, see `attributes`(5).

Change History
(Optional) Contains a link to the Change History appendix, if applicable.

Tuning the Solaris Kernel

The following table describes the different ways tunable parameters can be applied.

Apply Tunable Parameters in These Ways	For More Information
Modify the `/etc/system` file	"`/etc/system` File" on page 24
Use the kernel debugger (`kmdb`)	"`kmdb` Command" on page 26
Use the modular debugger (`mdb`)	"`mdb` Command" on page 26
Use the `ndd` command to set TCP/IP parameters	Chapter 4
Modify the `/etc/default` files	"Tuning NCA Parameters" on page 168

`/etc/system` File

The `/etc/system` file provides a static mechanism for adjusting the values of kernel parameters. Values specified in this file are read at boot time and are applied. Any changes that are made to the file are not applied to the operating system until the system is rebooted.

Prior to the Solaris 8 release, `/etc/system` entries that set the values of parameters were applied in two phases:

- The first phase obtains various bootstrap parameters (for example, `maxusers`) to initialize key system parameters.
- The second phase calculates the base configuration by using the bootstrap parameters, and all values specified in the `/etc/system` file are applied. In the case of the bootstrap parameters, reapplied values replace the values that are

calculated or reset in the initialization phase.

The second phase sometimes caused confusion to users and administrators by setting parameters to values that seem to be impermissible or by assigning values to parameters (for example, `max_nprocs`) that have a value overridden during the initial configuration.

Starting in the Solaris 8 release, one pass is made to set all the values before the configuration parameters are calculated.

Example—Setting a Parameter in `/etc/system`

The following `/etc/system` entry sets the number of read-ahead blocks that are read for file systems mounted using NFS version 2 software.

```
set nfs:nfs_nra=4
```

Recovering From an Incorrect Value

Make a copy of the `/etc/system` file before modifying it so that you can easily recover from incorrect value. For example:

```
# cp /etc/system /etc/system.good
```

If a value specified in the `/etc/system` file causes the system to become unbootable, you can recover with the following command:

```
ok boot -a
```

This command causes the system to ask for the name of various files used in the boot process. Press the Return key to accept the default values until the name of the `/etc/system` file is requested. When the `Name of system file [/etc/system]:` prompt is displayed, type the name of the good `/etc/system` file or `/dev/null`:

```
Name of system file [/etc/system]: /etc/system.good
```

If `/dev/null` is specified, this path causes the system to attempt to read from `/dev/null` for its configuration information. Because this file is empty, the system uses the default values. After the system is booted, the `/etc/system` file can be corrected.

For more information on system recovery, see *System Administration Guide: Basic Administration*.

kmdb Command

kmdb is a interactive kernel debugger with the same general syntax as mdb. An advantage of interactive kernel debugger is that you can set breakpoints. When a breakpoint is reached, you can examine data or step through the execution of kernel code.

kmdb can be loaded and unloaded on demand. You do not have to reboot the system to perform interactive kernel debugging, as was the case with kadb.

For more information, see kmdb(1).

mdb Command

Starting with the Solaris 8 release is the modular debugger, mdb, is unique among Solaris debuggers because it is easily extensible. A programming API is available that allows compilation of modules to perform desired tasks within the context of the debugger.

mdb also includes a number of desirable usability features, including command-line editing, command history, built-in output pager, syntax checking, and command pipelining. mdb is the recommended post-mortem debugger for the kernel.

For more information, see mdb(1).

Example—Using mdb to Change a Value

To change the value of the integer parameter maxusers from 495 to 512, do the following:

```
# mdb -kw
Loading modules: [ unix krtld genunix ip logindmux ptm nfs ipc lofs ]
> maxusers/D
maxusers:
maxusers:       495
> maxusers/W 200
maxusers:       0x1ef           =       0x200
> $q
```

Replace maxusers with the actual address of the item to be changed, as well as the value the parameter is to be set to.

For more information on using the modular debugger, see the *Solaris Modular Debugger Guide*.

When using either kmdb or mdb debugger, the module name prefix is not required. After a module is loaded, its symbols form a common name space with the core kernel symbols and any other previously loaded module symbols.

For example, ufs:ufs_WRITES would be accessed as ufs_WRITES in each debugger (assuming the UFS module is loaded). The ufs: prefix is required when set in the /etc/system file.

Special Solaris tune and var Structures

Solaris tunable parameters come in a variety of forms. The tune structure defined in the/usr/include/sys/tuneable.h file is the runtime representation of tune_t_fsflushr, tune_t_minarmem, and tune_t_flkrec. After the kernel is initialized, all references to these variables are found in the appropriate field of the tune structure.

Various documents (for example, previous versions of *Solaris System Administration Guide, Volume 2*) have stated that the proper way to set parameters in the tune structure is to use the syntax, tune:*field-name* where *field-name* is replaced by the actual parameter name listed above. This process silently fails. The proper way to set parameters for this structure at boot time is to initialize the special parameter that corresponds to the desired field name. The system initialization process then loads these values into the tune structure.

A second structure into which various tunable parameters are placed is the var structure named v. You can find the definition of a var structure in the /usr/include/sys/var.h file. The runtime representation of variables such as autoup and bufhwm is stored here.

Do not change either the tune or v structure on a running system. Changing any field in these structures on a running system might cause the system to panic.

Viewing Solaris System Configuration Information

Several tools are available to examine system configuration information. Some tools require superuser privilege. Other tools can be run by a non-privileged user. Every structure and data item can be examined with the kernel debugger by using mdb on a running system or by booting under kmdb.

For more information, see mdb(1) or kadb(1M).

sysdef Command

The sysdef command provides the values of System V IPC settings, STREAMS tunables, process resource limits, and portions of the tune and v structures. For example, the sysdef "Tunable Parameters" section from on a 512-Mbyte Sun™ Ultra™ 80 system is as follows:

```
10387456        maximum memory allowed in buffer cache (bufhwm)
    7930        maximum number of processes (v.v_proc)
      99        maximum global priority in sys class (MAXCLSYSPRI)
    7925        maximum processes per user id (v.v_maxup)
      30        auto update time limit in seconds (NAUTOUP)
      25        page stealing low water mark (GPGSLO)
       5        fsflush run rate (FSFLUSHR)
      25        minimum resident memory for avoiding deadlock (MINARMEM)
      25        minimum swapable memory for avoiding deadlock (MINASMEM)
```

For more information, see sysdef(1M).

kstat Utility

kstats are data structures maintained by various kernel subsystems and drivers. They provide a mechanism for exporting data from the kernel to user programs without requiring that the program read kernel memory or have superuser privilege. For more information, see kstat(1M) or kstat(3KSTAT).

Starting in the Solaris 8 release, the kstat command is available to enable the selection and display of kstats with a command-line interface. A Perl module, Kstat(3PERL), is also available to process kstat information.

Note – kstat data structures with system_pages name in the unix module do not report statistics for cachefree. cachefree is not supported, starting in the Solaris 9 release.

CHAPTER **2**

Solaris Kernel Tunable Parameters

This chapter describes most of the Solaris kernel tunable parameters.

Where to Find Tunable Parameter Information

Tunable Parameter	For Information
NFS tunable parameters	Chapter 3
Internet Protocol Suite tunable parameters	Chapter 4
Network Cache and Accelerator (NCA) tunable parameters	Chapter 5

General Kernel and Memory Parameters

This section describes general kernel parameters that are related to physical memory and stack configuration.

physmem

Description Modifies the system's configuration of the number of physical pages of memory after the Solaris OS and firmware are accounted for.

Data Type Unsigned long

Default Number of usable pages of physical memory available on the system, not counting the memory where the core kernel and data are stored

Range 1 to amount of physical memory on system

Units Pages

Dynamic? No

Validation None

When to Change Whenever you want to test the effect of running the system with less physical memory. Because this parameter does *not* take into account the memory used by the core kernel and data, as well as various other data structures allocated early in the startup process, the value of physmem should be less than the actual number of pages that represent the smaller amount of memory.

Commitment Level — Unstable

default_stksize

Description — Specifies the default stack size of all threads. No thread can be created with a stack size smaller than default_stksize. If default_stksize is set, it overrides lwp_default_stksize. See also "lwp_default_stksize" on page 32.

Data Type — Integer

Default

- 3 x PAGESIZE on SPARC systems
- 2 x PAGESIZE on x86 systems
- 5 x PAGESIZE on AMD64 systems

Range — Minimum is the default values:

- 3 x PAGESIZE on SPARC systems
- 2 x PAGESIZE on x86 systems
- 5 x PAGESIZE on AMD64 systems

Maximum is 32 times the default value.

Units — Bytes in multiples of the value returned by the getpagesize parameter. For more information, see getpagesize(3C).

Dynamic? — Yes. Affects threads created after the variable is changed.

Validation — Must be greater than or equal to 8192 and less than or equal to 262,144 (256 x 1024). Also must be a multiple of the system page size. If these conditions are not met, the following message is displayed:

```
Illegal stack size, Using N
```

The value of *N* is the default value of default_stksize.

When to Change — When the system panics because it has run out of stack space. The best solution for this problem is to determine why the system is running out of space and then make a correction.

Increasing the default stack size means that almost every kernel thread will have a larger stack, resulting in increased kernel memory consumption for no good reason. Generally, that space will be unused. The increased consumption means other resources that are competing for the same pool of memory will have the amount of space available to them reduced, possibly decreasing the system's ability to perform work. Among the side effects is a reduction in the number of threads that the kernel can create. This solution should be treated as no more than an interim workaround until the root cause is remedied.

Commitment Level Unstable

lwp_default_stksize

Description Specifies the default value of the stack size to be used when a kernel thread is created, and when the calling routine does not provide an explicit size to be used.

Data Type Integer

Default

- 8192 for x86 platforms
- 16,384 for SPARC platforms
- 20,480 for AMD64 platforms

Range Minimum is the default values:

- 3 x `PAGESIZE` on SPARC systems
- 2 x `PAGESIZE` on x86 systems
- 5 x `PAGESIZE` on AMD64 systems

Maximum is 32 times the default value.

Units Bytes in multiples of the value returned by the `getpagesize` parameter. For more information, see `getpagesize`(3C).

Dynamic? Yes. Affects threads created after the variable is changed.

Validation Must be greater than or equal to 8192 and less than or equal to 262,144 (256 x 1024). Also must be a multiple of the system page size. If these conditions are not met, the following message is displayed:

```
Illegal stack size, Using N
```

The value of *N* is the default value of `lwp_default_stksize`.

When to Change

When the system panics because it has run out of stack space. The best solution for this problem is to determine why the system is running out of space and then make a correction.

Increasing the default stack size means that almost every kernel thread will have a larger stack, resulting in increased kernel memory consumption for no good reason. Generally, that space will be unused. The increased consumption means other resources that are competing for the same pool of memory will have the amount of space available to them reduced, possibly decreasing the system's ability to perform work. Among the side effects is a reduction in the number of threads that the kernel can create. This solution should be treated as no more than an interim workaround until the root cause is remedied.

Commitment Level

Unstable

`logevent_max_q_sz`

Description

Maximum number of system events allowed to be queued and waiting for delivery to the `syseventd` daemon. Once the size of the system event queue reaches this limit, no other system events are allowed on the queue.

Data Type

Integer

Default

2000

Range

0 to MAXINT

Units

System events

Dynamic?

Yes

Validation

The system event framework checks this value every time a system event is generated by `ddi_log_sysevent` and `sysevent_post_event`.

For more information, see `ddi_log_sysevent`(9F) and `sysevent_post_event`(3SYSEVENT).

When to Change

When error log messages indicate that a system event failed to be logged, generated, or posted.

Commitment Level

Unstable

segkpsize

Description	Specifies the amount of kernel pageable memory available. This memory is used primarily for kernel thread stacks. Increasing this number allows either larger stacks for the same number of threads or more threads. This parameter can only be set on a system running a 64-bit kernel. A system running a 64-bit kernel uses a default stack size of 24 Kbytes.
Data Type	Unsigned long
Default	64-bit kernels, 2 Gbytes 32-bit kernels, 512 Mbytes
Range	64-bit kernels, 512 Mbytes to 24 Gbytes
Units	8-Kbyte pages
Dynamic?	No
Validation	Value is compared to minimum and maximum sizes (512 Mbytes and 24 Gbytes for 64-bit systems). If smaller than the minimum or larger than the maximum, it is reset to 2 Gbytes. A message to that effect is displayed. The actual size used in creation of the cache is the lesser of the value specified in segkpsize after the validation checking or 50 percent of physical memory.
When to Change	Required to support large numbers of processes on a system. The default size of 2 Gbytes, assuming at least 1 Gbyte of physical memory is present. This default size allows creation of 24-Kbyte stacks for more than 87,000 kernel threads. The size of a stack in a 64-bit kernel is the same, whether the process is a 32-bit process or a 64-bit process. If more than this number is needed, segkpsize can be increased, assuming sufficient physical memory exists.
Commitment Level	Unstable
Change History	For information, see "segkpsize (Solaris 9 12/02 Release)" on page 180.

noexec_user_stack

Description	Enables the stack to be marked as nonexecutable, which helps make buffer-overflow attacks more difficult.

A Solaris system running a 64-bit kernel makes the stacks of all 64-bit applications nonexecutable by default. Setting this parameter is necessary to make 32-bit applications nonexecutable on systems running 64-bit or 32-bit kernels.

Note – This parameter exists on all systems running the Solaris 2.6, 7, 8, 9, or 10 releases, but it is only effective on 64–bit SPARC and AMD64 architectures.

Data Type	Signed integer
Default	0 (disabled)
Range	0 (disabled) or 1 (enabled)
Units	Toggle (on/off)
Dynamic?	Yes. Does not affect currently running processes, only processes created after the value is set.
Validation	None
When to Change	Should be enabled at all times unless applications are deliberately placing executable code on the stack without using `mprotect` to make the stack executable. For more information, see `mprotect`(2).
Commitment Level	Unstable
Change History	For information, see "`noexec_user_stack` (Solaris 9 Releases)" on page 180.

`fsflush` and Related Parameters

This section describes `fsflush` and related tunables.

`fsflush`

The system daemon, `fsflush`, runs periodically to do three main tasks:

1. On every invocation, `fsflush` flushes dirty file system pages over a certain age to disk.

2. On every invocation, fsflush examines a portion of memory and causes modified pages to be written to their backing store. Pages are written if they are modified and if they do not meet one of the following conditions:

   - Pages are kernel page
   - Pages are free
   - Pages are locked
   - Pages are associated with a swap device
   - Pages are currently involved in an I/O operation

   The net effect is to flush pages from files that are mapped with mmap with write permission and that have actually been changed.

   Pages are flushed to backing store but left attached to the process using them. This will simplify page reclamation when the system runs low on memory by avoiding delay for writing the page to backing store before claiming it, if the page has not been modified since the flush.
3. fsflush writes file system metadata to disk. This write is done every *n*th invocation, where *n* is computed from various configuration variables. See "tune_t_fsflushr" on page 36 and "autoup" on page 37 for details.

The following features are configurable:

- Frequency of invocation (tune_t_fsflushr)
- Whether memory scanning is executed (dopageflush)
- Whether file system data flushing occurs (doiflush)
- The frequency with which file system data flushing occurs (autoup)

For most systems, memory scanning and file system metadata synchronizing are the dominant activities for fsflush. Depending on system usage, memory scanning can be of little use or consume too much CPU time.

tune_t_fsflushr

Description	Specifies the number of seconds between fsflush invocations
Data Type	Signed integer
Default	1
Range	1 to MAXINT
Units	Seconds
Dynamic?	No
Validation	If the value is less than or equal to zero, the value is reset to 1 and a warning message is displayed. This check is done only at boot time.

When to Change
See the `autoup` parameter.

Commitment Level
Unstable

autoup

Description
Along with `tune_t_flushr`, `autoup` controls the amount of memory examined for dirty pages in each invocation and frequency of file system synchronizing operations.

The value of `autoup` is also used to control whether a buffer is written out from the free list. Buffers marked with the `B_DELWRI` flag (which identifies file content pages that have changed) are written out whenever the buffer has been on the list for longer than *autoup* seconds. Increasing the value of `autoup` keeps the buffers in memory for a longer time.

Data Type
Signed integer

Default
30

Range
1 to MAXINT

Units
Seconds

Dynamic?
No

Validation
If `autoup` is less than or equal to zero, it is reset to 30 and a warning message is displayed. This check is done only at boot time.

Implicit
`autoup` should be an integer multiple of `tune_t_fsflushr`. At a minimum, `autoup` should be at least 6 times the value of `tune_t_fsflushr`. If not, excessive amounts of memory are scanned each time `fsflush` is invoked.

The total system pages multiplied by `tune_t_fsflushr` should be greater than or equal to `autoup` to cause memory to be checked if `dopageflush` is non-zero.

When to Change
Here are several potential situations for changing `autoup`, `tune_t_fsflushr`, or both:

- Systems with large amounts of memory – In this case, increasing `autoup` reduces the amount of memory scanned in each invocation of `fsflush`.
- Systems with minimal memory demand – Increasing both `autoup` and `tune_t_fsflushr` reduces the number of scans made. `autoup` should be increased also to maintain the current ratio of `autoup` / `tune_t_fsflushr`.

- Systems with large numbers of transient files (for example, mail servers or software build machines) – If large numbers of files are created and then deleted, `fsflush` might unnecessarily write data pages for those files to disk.

Commitment Level	Unstable

`dopageflush`

Description	Controls whether memory is examined for modified pages during `fsflush` invocations. In each invocation of `fsflush`, the number of memory pages in the system is determined. This number might have changed because of a dynamic reconfiguration operation. Each invocation scans by using this algorithm: total number of pages x `tune_t_fsflushr` / `autoup` pages
Data Type	Signed integer
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Units	Toggle (on/off)
Dynamic?	Yes
Validation	None
When to Change	If the system page scanner rarely runs, which is indicated by a value of 0 in the `sr` column of `vmstat` output.
Commitment Level	Unstable

`doiflush`

Description	Controls whether file system metadata syncs will be executed during `fsflush` invocations. This synchronization is done every *N*th invocation of `fsflush` where *N*`= (autoup / tune_t_fsflushr)`. Because this algorithm is integer division, if `tune_t_fsflushr` is greater than `autoup`, a synchronization is done on every invocation of `fsflush` because the code checks to see if its iteration counter is greater than or equal to *N*. Note that *N* is computed once on invocation of `fsflush`. Later changes to `tune_t_fsflushr` or `autoup` have no effect on the frequency of synchronization operations.

Data Type	Signed integer
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Units	Toggle (on/off)
Dynamic?	Yes
Validation	None
When to Change	When files are frequently modified over a period of time and the load caused by the flushing perturbs system behavior. Files whose existence, and therefore consistency of state, does not matter if the system reboots are better kept in a TMPFS file system (for example, `/tmp`). Inode traffic can be reduced on systems, starting in the Solaris 7 release, by using the `mount -noatime` option. This option eliminates inode updates when the file is accessed. For a system engaged in realtime processing, you might want to disable this option and use explicit application file synchronizing to achieve consistency.
Commitment Level	Unstable

Process-Sizing Parameters

Several parameters (or variables) are used to control the number of processes that are available on the system and the number of processes that an individual user can create. The foundation parameter is `maxusers`. This parameter drives the values assigned to `max_nprocs` and `maxuprc`.

`maxusers`

Description	Originally, `maxusers` defined the number of logged in users the system could support. When a kernel was generated, various tables were sized based on this setting. Current Solaris releases do much of its sizing based on the amount of memory on the system. Thus, much of the past use of `maxusers` has changed. A number of subsystems that are still derived from `maxusers`:

- The maximum number of processes on the system
- The number of quota structures held in the system
- The size of the directory name look-up cache (DNLC)

Data Type

Signed integer

Default

Lesser of the amount of memory in Mbytes or 2048

Range

1 to 2048, based on physical memory if not set in the /etc/system file

1 to 4096, if set in the /etc/system file

Units

Users

Dynamic?

No. After computation of dependent parameters is done, maxusers is never referenced again.

Validation

None

When to Change

When the default number of user processes derived by the system is too low. This situation is evident when the following message displays on the system console:

```
out of processes
```

You might also change this parameter when the default number of processes is too high, as in these situations:

- Database servers that have a lot of memory and relatively few running processes can save system memory when the default value of maxusers is reduced.
- If file servers have a lot of memory and few running processes, you might reduce this value. However, you should explicitly set the size of the DNLC. See "ncsize" on page 62.
- If compute servers have a lot of memory and few running processes, you might reduce this value.

Commitment Level

Unstable

reserved_procs

Description

Specifies the number of system process slots to be reserved in the process table for processes with a UID of root (0). For example, fsflush has a UID of root (0).

Data Type

Signed integer

Default

5

Range

5 to MAXINT

Units	Processes
Dynamic?	No. Not used after the initial parameter computation.
Validation	Starting in the Solaris 8 release, any `/etc/system` setting is honored.
Commitment Level	Unstable
When to Change	Consider increasing to 10 + the normal number of UID 0 (root) processes on system. This setting provides some cushion should it be necessary to obtain a root shell when the system is otherwise unable to create user-level processes.

`pidmax`

Description	Specifies the value of the largest possible process ID. Valid for Solaris 8 and later releases. `pidmax` sets the value for the `maxpid` variable. Once `maxpid` is set, `pidmax` is ignored. `maxpid` is used elsewhere in the kernel to determine the maximum process ID and for validation checking. Any attempts to set `maxpid` by adding an entry to the `/etc/system` file have no effect.
Data Type	Signed integer
Default	30,000
Range	266 to 999,999
Units	Processes
Dynamic?	No. Used only at boot time to set the value of `pidmax`.
Validation	Yes. Value is compared to the value of `reserved_procs` and 999,999. If less than `reserved_procs` or greater than 999,999, the value is set to 999,999.
Implicit	`max_nprocs` range checking ensures that `max_nprocs` is always less than or equal to this value.
When to Change	Required to enable support for more than 30,000 processes on a system.
Commitment Level	Unstable

`max_nprocs`

Description

Specifies the maximum number of processes that can be created on a system. Includes system processes and user processes. Any value specified in `/etc/system` is used in the computation of `maxuprc`.

This value is also used in determining the size of several other system data structures. Other data structures where this parameter plays a role are as follows:

- Determining the size of the directory name lookup cache (if `ncsize` is not specified)
- Allocating disk quota structures for UFS (if `ndquot` is not specified)
- Verifying that the amount of memory used by configured system V semaphores does not exceed system limits
- Configuring Hardware Address Translation resources for x86 platforms.

Data Type

Signed integer

Default

10 + (16 x `maxusers`)

Range

266 to value of `maxpid`

Dynamic?

No

Validation

Yes. The value is compared to `maxpid` and set to `maxpid` if it is larger. On x86 platforms, an additional check is made against a platform-specific value. `max_nprocs` is set to the smallest value in the triplet (`max_nprocs`, `maxpid`, platform value). Both SPARC and x86 platforms use 65,534 as the platform value.

When to Change

Changing this parameter is one of the steps necessary to enable support for more than 30,000 processes on a system.

Commitment Level

Unstable

Change History

For information, see "`max_nprocs` (Solaris 9 Releases)" on page 179.

`maxuprc`

Description

Specifies the maximum number of processes that can be created on a system by any one user.

Data Type

Signed integer

Default	`max_nprocs` - `reserved_procs`
Range	1 to `max_nprocs` - `reserved_procs`
Units	Processes
Dynamic?	No
Validation	Yes. This value is compared to `max_nprocs` - `reserved_procs` and set to the smaller of the two values.
When to Change	When you want to specify a hard limit for the number of processes a user can create that is less than the default value of however many processes the system can create. Attempting to exceed this limit generates the following warning messages on the console or in the messages file: `out of per-user processes for uid N`
Commitment Level	Unstable

Paging-Related Parameters

The Solaris OS uses a demand paged virtual memory system. As the system runs, pages are brought into memory as needed. When memory becomes occupied above a certain threshold and demand for memory continues, paging begins. Paging goes through several levels that are controlled by certain parameters.

The general paging algorithm is as follows:

- A memory deficit is noticed. The page scanner thread runs and begins to walk through memory. A two-step algorithm is employed:

  1. A page is marked as unused.
  2. If still unused after a time interval, the page is viewed as a subject for reclaim.

  If the page has been modified, a request is made to the pageout thread to schedule the page for I/O. Also, the page scanner continues looking at memory. Pageout causes the page to be written to the page's backing store and placed on the free list. When the page scanner scans memory, no distinction is made as to the origin of the page. The page might have come from a data file, or it might represent a page from an executable's text, data, or stack.
- As memory pressure on the system increases, the algorithm becomes more aggressive in the pages it will consider as candidates for reclamation and in how frequently the paging algorithm runs. (For more information, see "`fastscan`" on page 51 and "`slowscan`" on page 51.) As available memory falls between the

range lotsfree and minfree, the system linearly increases the amount of memory scanned in each invocation of the pageout thread from the value specified by slowscan to the value specified by fastscan. The system uses the desfree parameter to control a number of decisions about resource usage and behavior.

The system initially constrains itself to use no more than 4 percent of one CPU for pageout operations. As memory pressure increases, the amount of CPU time consumed in support of pageout operations linearly increases until a maximum of 80 percent of one CPU is consumed. The algorithm looks through some amount of memory between slowscan and fastscan, then stops when one of the following occurs:

- Enough pages have been found to satisfy the memory shortfall.
- The planned number of pages have been looked at.
- Too much time has elapsed.

If a memory shortfall is still present when pageout finishes its scan, another scan is scheduled for 1/4 second in the future.

The configuration mechanism of the paging subsystem was changed, starting in the Solaris 9 release. Instead of depending on a set of predefined values for fastscan, slowscan, and handspreadpages, the system determines the appropriate settings for these parameters at boot time. Setting any of these parameters in the /etc/system file can cause the system to use less than optimal values.

Caution – Remove all tuning of the VM system from the /etc/system file. Run with the default settings and determine if it is necessary to adjust any of these parameters. Do not set either cachefree or priority_paging. They have been removed, starting in the Solaris 9 release.

Beginning in the Solaris 7 5/99 release, dynamic reconfiguration (DR) for CPU and memory is supported. A system in a DR operation that involves the addition or deletion of memory recalculates values for the relevant parameters, unless the parameter has been explicitly set in /etc/system. In that case, the value specified in /etc/system is used, unless a constraint on the value of the variable has been violated. In this case, the value is reset.

lotsfree

Description Serves as the initial trigger for system paging to begin. When this threshold is crossed, the page scanner wakes up to begin looking for memory pages to reclaim.

Data Type Unsigned long

Default

The greater of 1/64th of physical memory or 512 Kbytes

Range

The minimum value is 512 Kbytes or 1/64th of physical memory, whichever is greater, expressed as pages using the page size returned by `getpagesize`. For more information, see`getpagesize`(3C).

The maximum value is the number of physical memory pages. The maximum value should be no more than 30 percent of physical memory. The system does not enforce this range, other than that described in the Validation section.

Units

Pages

Dynamic?

Yes, but dynamic changes are lost if a memory-based DR operation occurs.

Validation

If `lotsfree` is greater than the amount of physical memory, the value is reset to the default.

Implicit

The relationship of `lotsfree` being greater than `desfree`, which is greater than `minfree`, should be maintained at all times.

When to Change

When demand for pages is subject to sudden sharp spikes, the memory algorithm might be unable to keep up with demand. One workaround is to start reclaiming memory at an earlier time. This solution gives the paging system some additional margin.

A rule of thumb is to set this parameter to 2 times what the system needs to allocate in a few seconds. This parameter is workload dependent. A DBMS server can probably work fine with the default settings. However, you might need to adjust this parameter for a system doing heavy file system I/O.

For systems with relatively static workloads and large amounts of memory, lower this value. The minimum acceptable value is 512 Kbytes, expressed as pages using the page size returned by `getpagesize`.

Commitment Level

Unstable

desfree

Description

Specifies the preferred amount of memory to be free at all times on the system.

Data Type

Unsigned integer

Default

`lotsfree` / 2

Range

The minimum value is 256 Kbytes or 1/128th of physical memory, whichever is greater, expressed as pages using the page size returned by `getpagesize`.

The maximum value is the number of physical memory pages. The maximum value should be no more than 15 percent of physical memory. The system does not enforce this range other than that described in the Validation section.

Units

Pages

Dynamic?

Yes, unless dynamic reconfiguration operations that add or delete memory occur. At that point, the value is reset to the value provided in the `/etc/system` file or calculated from the new physical memory value.

Validation

If `desfree` is greater than `lotsfree`, `desfree` is set to `lotsfree` / 2. No message is displayed.

Implicit

The relationship of `lotsfree` being greater than `desfree`, which is greater than `minfree`, should be maintained at all times.

Side Effects

Several side effects can arise from increasing the value of this parameter. When the new value nears or exceeds the amount of available memory on the system, the following can occur:

- Asynchronous I/O requests are not processed, unless available memory exceeds `desfree`. Increasing the value of `desfree` can result in rejection of requests that otherwise would succeed.
- NFS asynchronous writes are executed as synchronous writes.
- The swapper is awakened earlier, and the behavior of the swapper is biased towards more aggressive actions.
- The system might not prefault as many executable pages into the system. This side effect results in applications potentially running slower than they otherwise would.

When to Change

For systems with relatively static workloads and large amounts of memory, lower this value. The minimum acceptable value is 256 Kbytes, expressed as pages using the page size returned by `getpagesize`.

Commitment Level

Unstable

`minfree`

Description

Specifies the minimum acceptable memory level. When memory drops below this number, the system biases allocations toward allocations necessary to successfully complete pageout operations or to swap processes completely out of memory. Either allocation denies or blocks other allocation requests.

Data Type

Unsigned integer

Default

`desfree` / 2

Range

The minimum value is 128 Kbytes or 1/256th of physical memory, whichever is greater, expressed as pages using the page size returned by `getpagesize`.

The maximum value is the number of physical memory pages. The maximum value should be no more than 7.5 percent of physical memory. The system does not enforce this range other than that described in the Validation section.

Units

Pages

Dynamic?

Yes, unless dynamic reconfiguration operations that add or delete memory occur. At that point, the value is reset to the value provided in the `/etc/system` file or calculated from the new physical memory value.

Validation

If `minfree` is greater than `desfree`, `minfree` is set to `desfree` / 2. No message is displayed.

Implicit

The relationship of `lotsfree` being greater than `desfree`, which is greater than `minfree`, should be maintained at all times.

When to Change

The default value is generally adequate. For systems with relatively static workloads and large amounts of memory, lower this value. The minimum acceptable value is 128 Kbytes, expressed as pages using the page size returned by `getpagesize`.

Commitment Level

Unstable

`throttlefree`

Description

Specifies the memory level at which blocking memory allocation requests are put to sleep, even if the memory is sufficient to satisfy the request.

Data Type

Unsigned integer

Default

`minfree`

Range

The minimum value is 128 Kbytes or 1/256th of physical memory, whichever is greater, expressed as pages using the page size returned by `getpagesize`.

The maximum value is the number of physical memory pages. The maximum value should be no more than 4 percent of physical memory. The system does not enforce this range other than that described in the Validation section.

Units

Pages

Dynamic?

Yes, unless dynamic reconfiguration operations that add or delete memory occur. At that point, the value is reset to the value provided in the `/etc/system` file or calculated from the new physical memory value.

Validation

If `throttlefree` is greater than `desfree`, `throttlefree` is set to `minfree`. No message is displayed.

Implicit

The relationship of `lotsfree` is greater than `desfree`, which is greater than `minfree`, should be maintained at all times.

When to Change

The default value is generally adequate. For systems with relatively static workloads and large amounts of memory, lower this value. The minimum acceptable value is 128 Kbytes, expressed as pages using the page size returned by `getpagesize`. For more information, see`getpagesize`(3C).

Commitment Level

Unstable

pageout_reserve

Description

Specifies the number of pages reserved for the exclusive use of the pageout or scheduler threads. When available memory is less than this value, nonblocking allocations are denied for any processes other than pageout or the scheduler. Pageout needs to have a small pool of memory for its use so it can allocate the data structures necessary to do the I/O for writing a page to its backing store. This variable was introduced in the Solaris 2.6 release to ensure that the system would be able to perform a pageout operation in the face of the most severe memory shortage.

Data Type

Unsigned integer

Default

`throttlefree` / 2

Range

The minimum value is 64 Kbytes or 1/512th of physical memory, whichever is greater, expressed as pages using the page size returned by `getpagesize(3C)`.

The maximum is the number of physical memory pages. The maximum value should be no more than 2 percent of physical memory. The system does not enforce this range, other than that described in the Validation section.

Units

Pages

Dynamic?

Yes, unless dynamic reconfiguration operations that add or delete memory occur. At that point, the value is reset to the value provided in the `/etc/system` file or calculated from the new physical memory value.

Validation

If `pageout_reserve` is greater than `throttlefree` / 2, `pageout_reserve` is set to `throttlefree` / 2. No message is displayed.

Implicit

The relationship of `lotsfree` being greater than `desfree`, which is greater than `minfree`, should be maintained at all times.

When to Change

The default value is generally adequate. For systems with relatively static workloads and large amounts of memory, lower this value. The minimum acceptable value is 64 Kbytes, expressed as pages using the page size returned by `getpagesize`.

Commitment Level

Unstable

pages_pp_maximum

Description

Defines the number of pages that must be unlocked. If a request to lock pages would force available memory below this value, that request is refused.

Data Type

Unsigned long

Default

The greater of (`tune_t_minarmem` + 100 and [4% of memory available at boot time + 4 Mbytes])

Range

Minimum value enforced by the system is `tune_t_minarmem` + 100. The system does not enforce a maximum value.

Units

Pages

Dynamic?

Yes, unless dynamic reconfiguration operations that add or delete memory occur. At that point, the value is reset to the value provided in the `/etc/system` file or was calculated from the new physical memory value.

Validation

If the value specified in the `/etc/system` file or the calculated default is less than `tune_t_minarmem` + 100, the value is reset to `tune_t_minarmem` + 100.

No message is displayed if the value from the `/etc/system` file is increased. Validation is done only at boot time and during dynamic reconfiguration operations that involve adding or deleting memory.

When to Change

When memory-locking requests fail or when attaching to a shared memory segment with the `SHARE_MMU` flag fails, yet the amount of memory available seems to be sufficient.

Excessively large values can cause memory locking requests (`mlock`, `mlockall`, and `memcntl`) to fail unnecessarily. For more information, see `mlock`(3C), `mlockall`(3C), and `memcntl`(2).

Commitment Level

Unstable

Change History

For information, see "`pages_pp_maximum` (Solaris Releases Prior to Solaris 9 Releases)" on page 180.

tune_t_minarmem

Description

Defines the minimum available resident (not swappable) memory to maintain necessary to avoid deadlock. Used to reserve a portion of memory for use by the core of the OS. Pages restricted in this way are not seen when the OS determines the maximum amount of memory available.

Data Type

Signed integer

Default

25

Range

1 to physical memory

Units

Pages

Dynamic?

No

Validation

None. Large values result in wasted physical memory.

When to Change

The default value is generally adequate. Consider increasing the default value if the system locks up and debugging information indicates that no memory was available.

Commitment Level	Unstable

fastscan

Description	Defines the maximum number of pages per second that the system looks at when memory pressure is highest.
Data Type	Signed integer
Default	The lesser of 64 Mbytes and 1/2 of physical memory.
Range	1 to one-half of physical memory
Units	Pages
Dynamic?	Yes, unless dynamic reconfiguration operations that add or delete memory occur. At that point, the value is reset to the value provided by `/etc/system` or calculated from the new physical memory value.
Validation	The maximum value is the lesser of 64 Mbytes and 1/2 of physical memory.
When to Change	When more aggressive scanning of memory is preferred during periods of memory shortfall, especially when the system is subject to periods of intense memory demand or when performing heavy file I/O.
Commitment Level	Unstable

slowscan

Description	Defines the minimum number of pages per second that the system looks at when attempting to reclaim memory.
Data Type	Signed integer
Default	The smaller of 1/20th of physical memory in pages and 100.
Range	1 to `fastscan` / 2
Units	Pages
Dynamic?	Yes, unless dynamic reconfiguration operations that add or delete memory occur. At that point, the value is reset to the value provided in the `/etc/system` file or calculated from the new physical memory value.
Validation	If `slowscan` is larger than `fastscan` / 2, `slowscan` is reset to `fastscan` / 2. No message is displayed.

When to Change	When more aggressive scanning of memory is preferred during periods of memory shortfall, especially when the system is subject to periods of intense memory demand.
Commitment Level	Unstable

min_percent_cpu

Description	Defines the minimum percentage of CPU that `pageout` can consume. This parameter is used as the starting point for determining the maximum amount of time that can be consumed by the page scanner.
Data Type	Signed integer
Default	4
Range	1 to 80
Units	Percentage
Dynamic?	Yes
Validation	None
When to Change	Increasing this value on systems with multiple CPUs and lots of memory, which are subject to intense periods of memory demand, enables the pager to spend more time attempting to find memory.
Commitment Level	Unstable

handspreadpages

Description	The Solaris OS uses a two-handed clock algorithm to look for pages that are candidates for reclaiming when memory is low. The first hand of the clock walks through memory marking pages as unused. The second hand walks through memory some distance after the first hand, checking to see if the page is still marked as unused. If so, the page is subject to being reclaimed. The distance between the first hand and the second hand is `handspreadpages`.
Data Type	Unsigned long
Default	`fastscan`
Range	1 to maximum number of physical memory pages on the system

Units	Pages
Dynamic?	Yes. This parameter requires that the kernel `reset_hands` parameter also be set to a non-zero value. Once the new value of `handspreadpages` has been recognized, `reset_hands` is set to zero.
Validation	The value is set to the lesser of either the amount of physical memory and the `handspreadpages` *value*.
When to Change	When you want to increase the amount of time that pages are potentially resident before being reclaimed. Increasing this value increases the separation between the hands, and therefore, the amount of time before a page can be reclaimed.
Commitment Level	Unstable

`pages_before_pager`

Description	Defines part of a system threshold that immediately frees pages after an I/O completes instead of storing the pages for possible reuse. The threshold is `lotsfree + pages_before_pager`. The NFS environment also uses this threshold to curtail its asynchronous activities as memory pressure mounts.
Data Type	Signed integer
Default	200
Range	1 to amount of physical memory
Units	Pages
Dynamic?	No
Validation	None
When to Change	You might change this parameter when the majority of I/O is done for pages that are truly read or written once and never referenced again. Setting this variable to a larger amount of memory keeps adding pages to the free list. You might also change this parameter when the system is subject to bursts of severe memory pressure. A larger value here helps maintain a larger cushion against the pressure.
Commitment Level	Unstable

maxpgio

Description	Defines the maximum number of page I/O requests that can be queued by the paging system. This number is divided by 4 to get the actual maximum number used by the paging system. This parameter is used to throttle the number of requests as well as to control process swapping.
Data Type	Signed integer
Default	40
Range	1 to 1024
Units	I/0s
Dynamic?	No
Validation	None
Implicit	The maximum number of I/O requests from the pager is limited by the size of a list of request buffers, which is currently sized at 256.
When to Change	When the system is subject to bursts of severe memory pressure. A larger value here helps to recover faster from the pressure if more than one swap device is configured or if the swap device is a striped device.
Commitment Level	Unstable

Swapping-Related Parameters

Swapping in the Solaris OS is accomplished by the swapfs pseudo file system. The combination of space on swap devices and physical memory is treated as the pool of space available to support the system for maintaining backing store for anonymous memory. The system attempts to allocate space from disk devices first, and then uses physical memory as backing store. When swapfs is forced to use system memory for backing store, limits are enforced to ensure that the system does not deadlock because of excessive consumption by swapfs.

swapfs_reserve

Description	Defines the amount of system memory that is reserved for use by system (UID = 0) processes.

Data Type	Unsigned long
Default	The smaller of 4 Mbytes and 1/16th of physical memory
Range	The minimum value is 4 Mbytes or 1/16th of physical memory, whichever is smaller, expressed as pages using the page size returned by `getpagesize`. The maximum value is the number of physical memory pages. The maximum value should be no more than 10 percent of physical memory. The system does not enforce this range, other than that described in the Validation section.
Units	Pages
Dynamic?	No
Validation	None
When to Change	Generally not necessary. Only change when recommended by a software provider, or when system processes are terminating because of an inability to obtain swap space. A much better solution is to add physical memory or additional swap devices to the system.
Commitment Level	Unstable

swapfs_minfree

Description	Defines the desired amount of physical memory to be kept free for the rest of the system. Attempts to reserve memory for use as swap space by any process that causes the system's perception of available memory to fall below this value are rejected. Pages reserved in this manner can only be used for locked-down allocations by the kernel or by user-level processes.
Data Type	Unsigned long
Default	The larger of 2 Mbytes and 1/8th of physical memory
Range	1 to amount of physical memory
Units	Pages
Dynamic?	No
Validation	None
When to Change	When processes are failing because of an inability to obtain swap space, yet the system has memory available.

Commitment Level Unstable

Kernel Memory Allocator

The Solaris kernel memory allocator distributes chunks of memory for use by clients inside the kernel. The allocator creates a number of caches of varying size for use by its clients. Clients can also request the allocator to create a cache for use by that client (for example, to allocate structures of a particular size). Statistics about each cache that the allocator manages can be seen by using the `kstat -c kmem_cache` command.

Occasionally, systems might panic because of memory corruption. The kernel memory allocator supports a debugging interface (a set of flags), that performs various integrity checks on the buffers. The kernel memory allocator also collects information on the allocators. The integrity checks provide the opportunity to detect errors closer to where they actually occurred. The collected information provides additional data for support people when they try to ascertain the reason for the panic.

Use of the flags incurs additional overhead and memory usage during system operations. The flags should only be used when a memory corruption problem is suspected.

kmem_flags

Description The Solaris kernel memory allocator has various debugging and test options that were extensively used during the internal development cycle of the Solaris OS. Starting in the Solaris 2.5 release, a subset of these options became available. They are controlled by the `kmem_flags` variable, which was set with a kernel debugger, and then rebooting the system. Because of issues with the timing of the instantiation of the kernel memory allocator and the parsing of the `/etc/system` file, it was not possible to set these flags in the `/etc/system` file until the Solaris 8 release.

Five supported flag settings are described here.

Flag	Setting	Description
AUDIT	0x1	The allocator maintains a log that contains recent history of its activity. The number of items logged depends on whether CONTENTS is also set. The log is a fixed size. When space is exhausted, earlier records are reclaimed.
TEST	0x2	The allocator writes a pattern into freed memory and checks that the pattern is unchanged when the buffer is next allocated. If some portion of the buffer is changed, then the memory was probably used by a client that had previously allocated and freed the buffer. If an overwrite is identified, the system panics.
REDZONE	0x4	The allocator provides extra memory at the end of the requested buffer and inserts a special pattern into that memory. When the buffer is freed, the pattern is checked to see if data was written past the end of the buffer. If an overwrite is identified, the kernel panics.
CONTENTS	0x8	The allocator logs up to 256 bytes of buffer contents when the buffer is freed. This flag requires that AUDIT also be set.
		The numeric value of these flags can be logically added together and set by the /etc/system file, starting in the Solaris 8 release, or for previous releases, by booting kadb and setting the flags before starting the kernel.
LITE	0x100	Does minimal integrity checking when a buffer is allocated and freed. When enabled, the allocator checks that the redzone has not been written into, that a freed buffer is not being freed again, and that the buffer being freed is the size that was allocated. This flag is available as of the Solaris 7 3/99 release. Do not combine this flag with any other flags.

Data Type	Signed integer
Default	0 (disabled)
Range	0 (disabled) or 1 - 15 or 256 (0x100)

Dynamic? Yes. Changes made during runtime only affect new kernel memory caches. After system initialization, the creation of new caches is rare.

Validation None

When to Change When memory corruption is suspected

Commitment Level Unstable

General Driver Parameter

moddebug

Description Used to cause messages about various steps in the module loading process to be displayed.

Data Type Signed integer

Default 0 (messages off)

Range Here are the most useful values:

- 0x80000000 – Prints `[un] loading...` message. For every module loaded, messages such as the following appear on the console and in the `/var/adm/messages` file:

  ```
  Nov 5 16:12:28 sys genunix: [ID 943528 kern.notice]
  load ’sched/TS_DPTBL’ id 9 loaded @ 0x10126438/
  0x10438dd8 size 132/2064
  Nov 5 16:12:28 sys genunix: [ID 131579 kern.notice]
  installing TS_DPTBL, module id 9.
  ```

- 0x40000000 – Prints detailed error messages. For every module loaded, messages such as the following appear on the console and in the `/var/adm/messages` file:

  ```
  Nov 5 16:16:50 sys krtld: [ID 284770 kern.notice]
  kobj_open: can’t open /platform/SUNW,Ultra-80/kernel/
  sched/TS_DPTBL
  Nov 5 16:16:50 sys krtld: [ID 284770 kern.notice]
  kobj_open: can’t open /platform/sun4u/kernel/sched/
  TS_DPTBL
  Nov 5 16:16:50 sys krtld: [ID 797908 kern.notice]
  kobj_open: ’/kernel/sch...
  Nov 5 16:16:50 sys krtld: [ID 605504 kern.notice]
  ```

```
descr = 0x2a
Nov 5 16:16:50 sys krtld: [ID 642728 kern.notice]
kobj_read_file: size=34,
Nov 5 16:16:50 sys krtld: [ID 217760 kern.notice]
offset=0
Nov 5 16:16:50 sys krtld: [ID 136382 kern.notice]
kobj_read: req 8192 bytes,
Nov 5 16:16:50 sys krtld: [ID 295989 kern.notice]
got 4224
Nov 5 16:16:50 sys krtld: [ID 426732 kern.notice]
read 1080 bytes
Nov 5 16:16:50 sys krtld: [ID 720464 kern.notice]
copying 34 bytes
Nov 5 16:16:50 sys krtld: [ID 234587 kern.notice]
count = 34
[33 lines elided]
Nov 5 16:16:50 sys genunix: [ID 943528 kern.notice]
load 'sched/TS_DPTBL' id 9 loaded @ 0x10126438/
0x10438dd8 size 132/2064
Nov 5 16:16:50 sys genunix: [ID 131579 kern.notice]
installing TS_DPTBL, module id 9.
Nov 5 16:16:50 sys genunix: [ID 324367 kern.notice]
init 'sched/TS_DPTBL' id 9 loaded @ 0x10126438/
0x10438dd8 size 132/2064
```

- 0x20000000 - Prints even more detailed messages. This value doesn't print any additional information beyond what the `0x40000000` flag does during system boot. However, this value does print additional information about releasing the module when the module is unloaded.

These values can be added together to set the final value.

Dynamic? Yes

Validation None

When to Change When a module is either not loading as expected, or the system seems to hang while loading modules. Note that when `0x40000000` is set, system boot is slowed down considerably by the number of messages written to the console.

Commitment Level Unstable

General I/O Parameters

maxphys

Description	Defines the maximum size of physical I/O requests. If a driver encounters a request larger than this size, the driver breaks the request into maxphys sized chunks. File systems can and do impose their own limit.
Data Type	Signed integer
Default	131,072 (Sun-4u) or 57,344 (x86). The sd driver uses the value of 1,048,576 if the drive supports wide transfers. The ssd driver uses 1,048,576 by default.
Range	Machine-specific page size to MAXINT
Units	Bytes
Dynamic?	Yes, but many file systems load this value into a per-mount point data structure when the file system is mounted. A number of drivers load the value at the time a device is attached to a driver-specific data structure.
Validation	None
When to Change	When doing I/O to and from raw devices in large chunks. Note that a DBMS doing OLTP operations issues large numbers of small I/Os. Changing maxphys does not result in any performance improvement in that case. You might also consider changing this parameter when doing I/O to and from a UFS file system where large amounts of data (greater than 64 Kbytes) are being read or written at any one time. The file system should be optimized to increase contiguity. For example, increase the size of the cylinder groups and decrease the number of inodes per cylinder group. UFS imposes an internal limit of 1 Mbyte on the maximum I/O size it transfers.
Commitment Level	Unstable

rlim_fd_max

Description

Specifies the “hard” limit on file descriptors that a single process might have open. Overriding this limit requires superuser privilege.

Data Type

Signed integer

Default

65,536

Range

1 to `MAXINT`

Units

File descriptors

Dynamic?

No

Validation

None

When to Change

When the maximum number of open files for a process is not enough. Other limitations in system facilities can mean that a larger number of file descriptors is not as useful as it might be. For example:

- A 32-bit program using standard I/O is limited to 256 file descriptors. A 64-bit program using standard I/O can use up to 2 billion descriptors. Specifically, standard I/O refers to the `stdio`(3C) functions in `libc`(3LIB).
- `select` is by default limited to 1024 descriptors per `fd_set`. For more information, see `select`(3C). Starting with the Solaris 7 release, 32-bit application code can be recompiled with a larger `fd_set` size (less than or equal to 65,536). A 64-bit application uses an `fd_set` size of 65,536, which cannot be changed.

An alternative to changing this on a system wide basis is to use the `plimit`(1) command. If a parent process has its limits changed by `plimit`, all children inherit the increased limit. This alternative is useful for daemons such as `inetd`.

Commitment Level

Unstable

Change History

For information, see “`rlim_fd_max` (Solaris 8 Release)” on page 180.

rlim_fd_cur

Description

Defines the “soft” limit on file descriptors that a single process can have open. A process might adjust its file descriptor limit to any value up to the “hard” limit defined by `rlim_fd_max`

	by using the `setrlimit()` call or by issuing the `limit` command in whatever shell it is running. You do not require superuser privilege to adjust the limit to any value less than or equal to the hard limit.
Data Type	Signed integer
Default	256
Range	1 to `MAXINT`
Units	File descriptors
Dynamic?	No
Validation	Compared to `rlim_fd_max`. If `rlim_fd_cur` is greater than `rlim_fd_max`, `rlim_fd_cur` is reset to `rlim_fd_max`.
When to Change	When the default number of open files for a process is not enough. Increasing this value means only that it might not be necessary for a program to use `setrlimit` to increase the maximum number of file descriptors available to it.
Commitment Level	Unstable

General File System Parameters

ncsize

Description	Defines the number of entries in the directory name look-up cache (DNLC). This parameter is used by UFS and NFS to cache elements of path names that have been resolved. Starting with the Solaris 8 6/00 release, the DNLC also caches negative look-up information, which means it caches a name not found in the cache.
Data Type	Signed integer
Default	4 x (`v.v_proc` + `maxusers`) + 320
Range	0 to `MAXINT`
Units	DNLC entries
Dynamic?	No

Validation

None. Larger values cause the time it takes to unmount a file system to increase as the cache must be flushed of entries for that file system during the unmount process.

When to Change

Prior to the Solaris 8 6/00 release, it was difficult to determine whether the cache was too small. You could make this inference by noting the number of entries returned by `kstat -n ncstats`. If the number seems high, given the system workload and file access pattern, this might be due to the size of the DNLC.

Starting with the Solaris 8 6/00 release, you can use the `kstat -n dnlcstats` command to determine when entries have been removed from the DNLC because it was too small. The sum of the `pick_heuristic` and the `pick_last` parameters represents otherwise valid entries that were reclaimed because the cache was too small.

Excessive values of `ncsize` have an immediate impact on the system because the system allocates a set of data structures for the DNLC based on the value of `ncsize`. A system running a 32-bit kernel allocates 36-byte structures for `ncsize`, while a system running a 64-bit kernel allocates 64-byte structures for `ncsize`. The value has a further effect on UFS and NFS, unless `ufs_ninode` and `nfs:nrnode` are explicitly set.

Commitment Level

Unstable

`rstchown`

Description

Indicates whether the POSIX semantics for the `chown` system call are in effect. POSIX semantics are as follows:

- A process cannot change the owner of a file, unless it is running with UID 0.
- A process cannot change the group ownership of a file to a group in which it is not currently a member, unless it is running as UID 0.

For more information, see `chown`(2).

Data Type

Signed integer

Default

1, indicating that POSIX semantics are used

Range

0 = POSIX semantics not in force or 1 = POSIX semantics used

Units

Toggle (on/off)

Dynamic?

Yes

Validation	None
When to Change	When POSIX semantics are not wanted. Note that turning off POSIX semantics opens the potential for various security holes. Doing so also opens the possibility of a user changing ownership of a file to another user and being unable to retrieve the file without intervention from the user or the system administrator.
Commitment Level	Obsolete

`dnlc_dir_enable`

Description	Enables large directory caching

Note – This parameter has no effect on NFS file systems.

Data Type	Unsigned integer
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes, but do not change this tunable dynamically. You can enable this parameter if it was originally disabled. Or, you can disable this parameter if it was originally enabled. However, enabling, disabling, and then enabling this parameter might lead to stale directory caches.
Validation	No
When to Change	Directory caching has no known problems. However, if problems occur, then set `dnlc_dir_enable` to 0 to disable caching.
Commitment Level	Unstable

`dnlc_dir_min_size`

Description	Specifies the minimum number of entries cached for one directory.

Note – This parameter has no effect on NFS file systems.

Data Type	Unsigned integer
Default	40
Range	0 to MAXUINT (no maximum)
Units	Entries
Dynamic?	Yes, this parameter can be changed at any time.
Validation	None
When to Change	If performance problems occur with caching small directories, then increase `dnlc_dir_min_size`. Note that individual file systems might have their own range limits for caching directories. For instance, UFS limits directories to a minimum of `ufs_min_dir_cache` bytes (approximately 1024 entries), assuming 16 bytes per entry.
Commitment Level	Unstable

dnlc_dir_max_size

Description	Specifies the maximum number of entries cached for one directory.

Note – This parameter has no effect on NFS file systems.

Data Type	Unsigned integer
Default	MAXUINT (no maximum)
Range	0 to MAXUINT
Dynamic?	Yes, this parameter can be changed at any time.
Validation	None
When to Change	If performance problems occur with large directories, then decrease `dnlc_dir_max_size`.
Commitment Level	Unstable

`segmap_percent`

Description	Defines the maximum amount of memory that is used for the fast-access file system cache. This pool of memory is subtracted from the free memory list.
Data Type	Unsigned integer
Default	12 percent of free memory at system startup time
Range	2 Mbytes to 100 percent of `physmem`
Units	% of physical memory
Dynamic?	No
Validation	None
When to Change	If heavy file system activity is expected, and sufficient free memory is available, you should increase the value of this parameter.
Commitment Level	Unstable

UFS Parameters

`bufhwm` and `bufhwm_pct`

Description	Defines the maximum amount of memory for caching I/O buffers. The buffers are used for writing file system metadata (superblocks, inodes, indirect blocks, and directories). Buffers are allocated as needed until the amount of memory (in Kbytes) to be allocated exceed `bufhwm`. At this point, metadata is purged from the buffer cache until enough buffers are reclaimed to satisfy the request. For historical reasons, `bufhwm` does not require the `ufs:` prefix.
Data Type	Signed integer
Default	2 percent of physical memory
Range	80 Kbytes to 20 percent of physical memory, or 2 TB, whichever is less. Consequently, `bufhwm_pct` can be between 1 and 20.

Units

bufhwm: Kbytes

bufhwm_pct: percent of physical memory

Dynamic?

No. bufhwm and bufhwm_pct are only evaluated at system initialization to compute hash bucket sizes. The limit in bytes calculated from these parameters is then stored in a data structure that adjusts this value as buffers are allocated and deallocated.

Attempting to adjust this value without following the locking protocol on a running system can lead to incorrect operation.

Modifying bufhwm or bufhwm_pct at runtime has no effect.

Validation

If bufhwm is less than its lower limit of 80 Kbytes or greater than its upper limit (the lesser of 20 percent of physical memory, 2 TB, or one quarter (1/4) of the maximum amount of kernel heap), it is reset to the upper limit. The following message appears on the system console and in the /var/adm/messages file if an invalid value is attempted:

```
"binit: bufhwm (value attempted) out of range
(range start..range end). Using N as default."
```

"Value attempted" refers to the value specified in the/etc/system file or by using a kernel debugger. *N* is the value computed by the system based on available system memory.

Likewise, if bufhwm_pct is set to a value that is outside the allowed range of 1 percent to 20 percent, it is reset to the default of 2 percent. And, the following message appears on the system console and in the /var/adm/messages file:

```
"binit: bufhwm_pct(value attempted) out of range(0..20).
       Using 2 as default."
```

If both bufhwm or bufhwm_pct are set to non-zero values, bufhwm takes precedence.

When to Change

Because buffers are only allocated as they are needed, the overhead from the default setting is the required allocation of control structures for the buffer hash headers. These structures consume 52 bytes per potential buffer on a 32-bit kernel and 96 bytes per potential buffer on a 64-bit kernel.

On a 512-Mbyte 64-bit kernel, the number of hash chains calculates to 10316 / 32 == 322, which scales up to next power of 2, 512. Therefore, the hash headers consume 512 x 96 bytes, or 48 Kbytes. The hash header allocations assume that buffers are 32 Kbytes.

The amount of memory, which has not been allocated in the buffer pool, can be found by looking at the bfreelist structure in the kernel with a kernel debugger. The field of interest in the structure is b_bufsize, which is the possible remaining memory in bytes. Looking at it with the buf macro by using the mdb command:

```
# mdb -kLoading modules: [ unix krtld genunix ip nfs ipc ]
> bfreelist::print "struct buf" b_bufsize
b_bufsize = 0x225800
```

The default value for bufhwm on this system, with 6 Gbytes of memory, is 122277. You cannot determine the number of header structures used because the actual buffer size requested is usually larger than 1 Kbyte. However, some space might be profitably reclaimed from control structure allocation for this system.

The same structure on a 512-Mbyte system shows that only 4 Kbytes of 10144 Kbytes has not been allocated. When the biostats kstat is examined with kstat -n biostats, it is determined that the system had a reasonable ratio of buffer_cache_hits to buffer_cache_lookups as well. As such, the default setting is reasonable for that system.

Commitment Level: Unstable

Change History: For information, see "bufhwm (Solaris 9 Releases)" on page 181.

ndquot

Description: Defines the number of quota structures for the UFS file system that should be allocated. Relevant only if quotas are enabled on one or more UFS file systems. Because of historical reasons, the ufs: prefix is not needed.

Data Type: Signed integer

Default: ((maxusers x 40) / 4) + max_nprocs

Range: 0 to MAXINT

Units: Quota structures

Dynamic?: No

Validation: None. Excessively large values hang the system.

When to Change: When the default number of quota structures is not enough. This situation is indicated by the following message displayed on the console or written in the message log:

```
dquot table full
```

Commitment Level Unstable

ufs_ninode

Description

Specifies the number of inodes to be held in memory. Inodes are cached globally for UFS, not on a per-file system basis.

A key parameter in this situation is `ufs_ninode`. This parameter is used to compute two key limits that affect the handling of inode caching. A high watermark of `ufs_ninode / 2` and a low watermark of `ufs_ninode / 4` are computed.

When the system is done with an inode, one of two things can happen:

- The file referred to by the inode is no longer on the system so the inode is deleted. After it is deleted, the space goes back into the inode cache for use by another inode (which is read from disk or created for a new file).
- The file still exists but is no longer referenced by a running process. The inode is then placed on the idle queue. Any referenced pages are still in memory.

When inodes are idled, the kernel defers the idling process to a later time. If a file system is a logging file system, the kernel also defers deletion of inodes. Two kernel threads handle this deferred processing. Each thread is responsible for one of the queues.

When the deferred processing is done, the system drops the inode onto either a delete queue or an idle queue, each of which has a thread that can run to process it. When the inode is placed on the queue, the queue occupancy is checked against the low watermark. If the queue occupancy exceeds the low watermark, the thread associated with the queue is awakened. After the queue is awakened, the thread runs through the queue and forces any pages associated with the inode out to disk and frees the inode. The thread stops when it has removed 50 percent of the inodes on the queue at the time it was awakened.

A second mechanism is in place if the idle thread is unable to keep up with the load. When the system needs to find a vnode, it goes through the `ufs_vget` routine. The *first* thing `vget` does is check the length of the idle queue. If the length is above the high watermark, then it takes two inodes off the idle queue and “idles” them (flushes pages and frees inodes). `vget` does this *before* it gets an inode for its own use.

The system does attempt to optimize by placing inodes with no in-core pages at the head of the idle list and inodes with pages at the end of the idle list. However, the system does no other ordering of the list. Inodes are always removed from the front of the idle queue.

The only time that inodes are removed from the queues as a whole is when a synchronization, unmount, or remount occur.

For historical reasons, this parameter does not require the `ufs:` prefix.

Data Type	Signed integer
Default	`ncsize`
Range	0 to MAXINT
Units	Inodes
Dynamic?	Yes
Validation	If `ufs_ninode` is less than or equal to zero, the value is set to `ncsize`.
When to Change	When the default number of inodes is not enough. If the `maxsize reached` field as reported by `kstat -n inode_cache` is larger than the `maxsize` field in the `kstat`, the value of `ufs_ninode` might be too small. Excessive inode idling can also be a problem. You can identify excessive inode idling by using `kstat -n inode_cache` to look at the `inode_cache` `kstat`. `Thread idles` are inodes idled by the background threads while `vget idles` are idles by the requesting process before using an inode.
Commitment Level	Unstable

ufs_WRITES

Description	If ufs_WRITES is non-zero, the number of bytes outstanding for writes on a file is checked. See ufs_HW to determine whether the write should be issued or deferred until only ufs_LW bytes are outstanding. The total number of bytes outstanding is tracked on a per-file basis so that if the limit is passed for one file, it won't affect writes to other files.
Data Type	Signed integer
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Units	Toggle (on/off)
Dynamic?	Yes
Validation	None
When to Change	When you want UFS write throttling turned off entirely. If sufficient I/O capacity does not exist, disabling this parameter can result in long service queues for disks.
Commitment Level	Unstable

ufs_LW and ufs_HW

Description	ufs_HW specifies the number of bytes outstanding on a single file barrier value. If the number of bytes outstanding is greater than this value and ufs_WRITES is set, then the write is deferred. The write is deferred by putting the thread issuing the write to sleep on a condition variable. ufs_LW is the barrier for the number of bytes outstanding on a single file below which the condition variable on which other sleeping processes are toggled. When a write completes and the number of bytes is less than ufs_LW, then the condition variable is toggled, which causes all threads waiting on the variable to awaken and try to issue their writes.
Data Type	Signed integer
Default	8 x 1024 x 1024 for ufs_LW and 16 x 1024 x 1024 for ufs_HW
Range	0 to MAXINT
Units	Bytes
Dynamic?	Yes

Validation	None
Implicit	`ufs_LW` and `ufs_HW` have meaning only if `ufs_WRITES` is not equal to zero. `ufs_HW` and `ufs_LW` should be changed together to avoid needless churning when processes awaken and find that either they cannot issue a write (when `ufs_LW` and `ufs_HW` are too close) or they might have waited longer than necessary (when `ufs_LW` and `ufs_HW` are too far apart).
When to Change	Consider changing these values when file systems consist of striped volumes. The aggregate bandwidth available can easily exceed the current value of `ufs_HW`. Unfortunately, this parameter is not a per-file system setting. You might also consider changing this parameter when `ufs_throttles` is a non-trivial number. Currently, `ufs_throttles` can only be accessed with a kernel debugger.
Commitment Level	Unstable

freebehind

Description	Enables the `freebehind` algorithm. When this algorithm is enabled, the system bypasses the file system cache on newly read blocks when sequential I/O is detected during times of heavy memory use.
Data Type	Boolean
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes
Validation	None
When to Change	The `freebehind` algorithm can occur too easily. If no significant sequential file system activity is expected, disabling `freebehind` makes sure that all files, no matter how large, will be candidates for retention in the file system page cache. For more fine-grained tuning, see `smallfile`.
Commitment Level	Unstable

`smallfile`

Description

Determines the size threshold of files larger than this value are candidates for no cache retention under the `freebehind` algorithm.

Large memory systems contain enough memory to cache thousands of 10-Mbyte files without making severe memory demands. However, this situation is highly application dependent.

The goal of the `smallfile` and `freebehind` parameters is to reuse cached information, without causing memory shortfalls by caching too much.

Data Type

Signed integer

Default

32,768

Range

0 to 2,147,483,647

Dynamic?

Yes

Validation

None

When to Change

Increase `smallfile` if an application does sequential reads on medium-sized files and can most likely benefit from buffering, and the system is not otherwise under pressure for free memory. Medium-sized files are 32 Kbytes to 2 Gbytes in size.

Commitment Level

Unstable

TMPFS Parameters

`tmpfs:tmpfs_maxkmem`

Description

Defines the maximum amount of kernel memory that TMPFS can use for its data structures (tmpnodes and directory entries).

Data Type

Unsigned long

Default

One page or 4 percent of physical memory, whichever is greater.

Range

Number of bytes in one page (8192 for sun4u systems, 4096 for all other systems) to 25 percent of the available kernel memory at the time TMPFS was first used.

Units

Bytes

Dynamic?

Yes

Validation

None

When to Change

Increase if the following message is displayed on the console or written in the messages file:

```
tmp_memalloc: tmpfs over memory limit
```

The current amount of memory used by TMPFS for its data structures is held in the `tmp_kmemspace` field. This field can be examined with a kernel debugger.

Commitment Level

Unstable

tmpfs:tmpfs_minfree

Description

Defines the minimum amount of swap space that TMPFS leaves for the rest of the system.

Data Type

Signed long

Default

256

Range

0 to maximum swap space size

Units

Pages

Dynamic?

Yes

Validation

None

When to Change

To maintain a reasonable amount of swap space on systems with large amounts of TMPFS usage, you can increase this number. The limit has been reached when the console or messages file displays the following message:

```
fs-name: File system full, swap space limit exceeded
```

Commitment Level

Unstable

Change History

For information, see “`tmpfs:tmpfs_minfree` (Solaris 8 Releases)” on page 180.

Pseudo Terminals

Pseudo terminals, ptys, are used for two purposes in Solaris software:

- Supporting remote logins by using the telnet, rlogin, or rsh commands
- Providing the interface through which the X Window system creates command interpreter windows

The default number of pseudo-terminals is sufficient for a desktop workstation. So, tuning focuses on the number of ptys available for remote logins.

Previous versions of Solaris required that steps be taken to explicitly configure the system for the preferred number of ptys. Starting with the Solaris 8 release, a new mechanism removes the necessity for tuning in most cases. The default number of ptys is now based on the amount of memory on the system. This default should be changed only to restrict or increase the number of users who can log in to the system.

Three related variables are used in the configuration process:

- pt_cnt – Default maximum number of ptys.
- pt_pctofmem – Percentage of kernel memory that can be dedicated to pty support structures. A value of zero means that no remote users can log in to the system.
- pt_max_pty – Hard maximum for number of ptys.

pt_cnt has a default value of zero, which tells the system to limit logins based on the amount of memory specified in pct_pctofmem, unless pt_max_pty is set. If pt_cnt is non-zero, ptys are allocated until this limit is reached. When that threshold is crossed, the system looks at pt_max_pty. If pt_max_pty has a non-zero value, it is compared to pt_cnt. The pty allocation is allowed if pt_cnt is less than pt_max_pty. If pt_max_pty is zero, pt_cnt is compared to the number of ptys supported based on pt_pctofmem. If pt_cnt is less than this value, the pty allocation is allowed. Note that the limit based on pt_pctofmem only comes into play if both pt_cnt and ptms_ptymax have default values of zero.

To put a hard limit on ptys that is different than the maximum derived from pt_pctofmem, set pt_cnt and ptms_ptymax in /etc/system to the preferred number of ptys. The setting of ptms_pctofmem is not relevant in this case.

To dedicate a different percentage of system memory to pty support and let the operating system manage the explicit limits, do the following:

- Do not set pt_cnt or ptms_ptymax in /etc/system.
- Set pt_pctofmem in /etc/system to the preferred percentage. For example, set pt_pctofmem=10 for a 10 percent setting.

Note that the memory is not actually allocated until it is used in support of a pty. Once memory is allocated, it remains allocated.

pt_cnt

Description	The number of available /dev/pts entries is dynamic up to a limit determined by the amount of physical memory available on the system. pt_cnt is one of three variables that determines the minimum number of logins that the system can accommodate. The default maximum number of /dev/pts devices the system can support is determined at boot time by computing the number of pty structures that can fit in a percentage of system memory (see pt_pctofmem). If pt_cnt is zero, the system allocates up to that maximum. If pt_cnt is non-zero, the system allocates to the greater of pt_cnt and the default maximum.
Data Type	Unsigned integer
Default	0
Range	0 to maxpid
Units	Logins/windows
Dynamic?	No
Validation	None
When to Change	When you want to explicitly control the number of users who can remotely log in to the system.
Commitment Level	Unstable

pt_pctofmem

Description	Specifies the maximum percentage of physical memory that can be consumed by data structures to support /dev/pts entries. A system running a 64-bit kernel consumes 176 bytes per /dev/pts entry. A system running a 32-bit kernel consumes 112 bytes per /dev/pts entry.
Data Type	Unsigned integer
Default	5
Range	0 to 100
Units	Percentage

Dynamic?	No
Validation	None
When to Change	When you want to either restrict or increase the number of users who can log in to the system. A value of zero means that no remote users can log in to the system.
Commitment Level	Unstable

pt_max_pty

Description	Defines the maximum number of ptys the system offers
Data Type	Unsigned integer
Default	0 (Uses system-defined maximum)
Range	0 to MAXUINT
Units	Logins/windows
Dynamic?	Yes
Validation	None
Implicit	Should be greater than or equal to pt_cnt.Value is not checked until the number of ptys allocated exceeds the value of pt_cnt.
When to Change	When you want to place an absolute ceiling on the number of logins supported, even if the system could handle more based on its current configuration values.
Commitment Level	Unstable

STREAMS Parameters

nstrpush

Description	Specifies the number of modules that can be inserted into (pushed onto) a STREAM.
Data Type	Signed integer

Default	9
Range	9 to 16
Units	Modules
Dynamic?	Yes
Validation	None
When to Change	At the direction of your software vendor. No messages are displayed when a STREAM exceeds its permitted push count. A value of `EINVAL` is returned to the program that attempted the push.
Commitment Level	Unstable

`strmsgsz`

Description	Specifies the maximum number of bytes that a single system call can pass to a STREAM to be placed in the data part of a message. Any `write` exceeding this size is broken into multiple messages. For more information, see `write`(2).
Data Type	Signed integer
Default	65,536
Range	0 to 262,144
Units	Bytes
Dynamic?	Yes
Validation	None
When to Change	When `putmsg` calls return `ERANGE`. For more information, see `putmsg`(2).
Commitment Level	Unstable

`strctlsz`

Description	Specifies the maximum number of bytes that a single system call can pass to a STREAM to be placed in the control part of a message
Data Type	Signed integer
Default	1024

Range	0 to MAXINT
Units	Bytes
Dynamic?	Yes
Validation	None
When to Change	At the direction of your software vendor. `putmsg`(2) calls return `ERANGE` if they attempt to exceed this limit.
Commitment Level	Unstable

System V Message Queues

For new information about System V message queues in the Solaris 10 release, see "System V IPC Configuration" on page 17.

System V message queues provide a message-passing interface that enables the exchange of messages by queues created in the kernel. Interfaces are provided in the Solaris environment to enqueue and dequeue messages. Messages can have a type associated with them. Enqueueing places messages at the end of a queue. Dequeuing removes the first message of a specific type from the queue or the first message if no type is specified.

For information on tuning these system resources, see Chapter 6, "Resource Controls (Overview)," in *System Administration Guide: N1 Grid Containers, Resource Management, and Solaris Zones*.

System V Semaphores

For new information about System V semaphores in the Solaris 10 release, see "System V IPC Configuration" on page 17.

System V semaphores provide counting semaphores in the Solaris OS. A *semaphore* is a counter used to provide access to a shared data object for multiple processes. In addition to the standard set and release operations for semaphores, System V semaphores can have values that are incremented and decremented as needed (for example, to represent the number of resources available). System V semaphores also provide the ability to do operations on a group of semaphores simultaneously as well as to have the system undo the last operation by a process if the process dies.

Semaphores are created in sets.

The module is dynamically loaded on first reference. Parameters provided to the subsystem are validated at that time and all data structures (including the semaphores) are created. Values for parameters are, accordingly, not changeable at runtime because increases in values would lead to data corruption. Entries in the /etc/system file must contain the semsys: prefix.

System V semaphores are different from the POSIX semaphore facility.

semsys:seminfo_semmni

Description	Specifies the maximum number of semaphore identifiers.
Data Type	Signed integer
Default	10
Range	1 to 65,535
Dynamic?	No
Validation	Compared to SEMA_INDEX_MAX (currently 65,535) and reset to that value if larger. A warning message is written to the console, messages file, or both.
When to Change	When the default number of sets is not enough. Generally changed at the recommendation of software vendors. No error messages are displayed when an attempt is made to create more sets than are currently configured. Instead, the application receives a return code of ENOSPC from a semget call. For more information, see semget(2).
Commitment Level	Unstable

semsys:seminfo_semmsl

Description	Specifies the maximum number of System V semaphores per semaphore identifier.
Data Type	Signed integer
Default	25
Range	1 to MAXINT
Dynamic?	No

Validation

The amount of space that could possibly be consumed by the semaphores and their supporting data structures is compared to 25 percent of the kernel memory available at the time the module is first loaded. If the memory threshold is exceeded, the module refuses to load and the semaphore facility is not available.

When to Change

When the default value is not enough. Generally changed at the recommendation of software vendors. No error messages are displayed when an attempt is made to create more semaphores in a set than are currently configured. The application sees a return code of `EINVAL` from a `semget`(2) call.

Commitment Level

Unstable

semsys:seminfo_semopm

Description

Specifies the maximum number of System V semaphore operations per `semop` call. This parameter refers to the number of `sembufs` in the `sops` array that is provided to the `semop()` system call. For more information, see `semop`(2).

Data Type

Signed integer

Default

10

Range

1 to MAXINT

Dynamic?

No

Validation

The amount of space that could possibly be consumed by the semaphores and their supporting data structures is compared to 25 percent of the kernel memory available at the time the module is first loaded. If the memory threshold is exceeded, the module refuses to load and the semaphore facility is not available.

When to Change

When the default value is not enough. Generally changed at the recommendation of software vendors. No error messages are displayed when an attempt is made to perform more semaphore operations in a single `semop` call than are currently allowed. Instead, the application receives a return code of `E2BIG` from a `semop()` call.

Commitment Level

Unstable

System V Shared Memory

For new information about System V shared memory in the Solaris 10 release, see "System V IPC Configuration" on page 17.

System V shared memory allows the creation of a segment by a process. Cooperating processes can attach to the memory segment (subject to access permissions on the segment) and gain access to the data contained in the segment. This capability is implemented as a loadable module. Entries in the `/etc/system` file must contain the `shmsys`: prefix. Starting with the Solaris 7 release, the `keyserv` daemon uses System V shared memory.

A special kind of shared memory known as *intimate shared memory* (ISM) is used by DBMS vendors to maximize performance. When a shared memory segment is made into an ISM segment, the memory for the segment is locked. This feature enables a faster I/O path to be followed and improves memory usage. A number of kernel resources describing the segment are then shared between all processes that attach to the segment in ISM mode.

The module is dynamically loaded on first reference. Parameters provided to the subsystem are validated at that time.

System V shared memory is different from the POSIX shared memory facility.

segspt_minfree

Description	Identifies pages of system memory that cannot be allocated for ISM shared memory.
Data Type	Unsigned long
Default	5 percent of available system memory when the first ISM segment is created
Range	0 to 50 percent of physical memory
Units	Pages
Dynamic?	Yes
Validation	None. Values that are too small can cause the system to hang or performance to severely degrade when memory is consumed with ISM segments.

When to Change	On database servers with large amounts of physical memory using ISM, the value of this parameter can be decreased. If ISM segments are not used, this parameter has no effect. A maximum value of 128 Mbytes (0x4000) is almost certainly sufficient on large memory machines.
Commitment Level	Unstable

Scheduling

`rechoose_interval`

Description	Specifies the number of clock ticks before a process is deemed to have lost all affinity for the last CPU it ran on. After this interval expires, any CPU is considered a candidate for scheduling a thread. This parameter is relevant only for threads in the timesharing class. Real-time threads are scheduled on the first available CPU.
Data Type	Signed integer
Default	3
Range	0 to MAXINT
Dynamic?	Yes
Validation	None
When to Change	When caches are large, or when the system is running a critical process or a set of processes that seem to suffer from excessive cache misses not caused by data access patterns. Consider using the processor set capabilities available as of the Solaris 2.6 release or processor binding before changing this parameter. For more information, see `psrset`(1M) or `pbind`(1M).
Commitment Level	Unstable

Timers

hires_tick

Description	When set, this parameter causes the Solaris OS to use a system clock rate of 1000 instead of the default value of 100.
Data Type	Signed integer
Default	0
Range	0 (disabled) or 1 (enabled)
Dynamic?	No. Causes new system timing variable to be set at boot time. Not referenced after boot.
Validation	None
When to Change	When you want timeouts with a resolution of less than 10 milliseconds, and greater than or equal to 1 millisecond.
Commitment Level	Unstable

timer_max

Description	Specifies the number of POSIX™ timers available.
Data Type	Signed integer
Default	32
Range	0 to MAXINT
Dynamic?	No. Increasing the value can cause a system crash.
Validation	None
When to Change	When the default number of timers offered by the system is inadequate. Applications receive an `EAGAIN` error when executing `timer_create` system calls.
Commitment Level	Unstable

Sun-4u Specific Parameters

consistent_coloring

Description

Starting with the Solaris 2.6 release, the ability to use different page placement policies on the UltraSPARC® (sun4u) platform was introduced. A page placement policy attempts to allocate physical page addresses to maximize the use of the L2 cache. Whatever algorithm is chosen as the default algorithm, that algorithm can potentially provide less optimal results than another algorithm for a particular application set. This parameter changes the placement algorithm selected for all processes on the system.

Based on the size of the L2 cache, memory is divided into bins. The page placement code allocates a page from a bin when a page fault first occurs on an unmapped page. The page chosen depends on which of the three possible algorithms are used:

- Page coloring – Various bits of the virtual address are used to determine the bin from which the page is selected. This is the default algorithm in the Solaris 8 release. consistent_coloring is set to zero to use this algorithm. No per-process history exists for this algorithm.
- Virtual addr=physical address – Consecutive pages in the program selects pages from consecutive bins. consistent_coloring is set to 1 to use this algorithm. No per-process history exists for this algorithm.
- Bin-hopping – Consecutive pages in the program generally allocate pages from every other bin, but the algorithm occasionally skips more bins. consistent_coloring is set to 2 to use this algorithm. Each process starts at a randomly selected bin, and a per-process memory of the last bin allocated is kept.

Dynamic?

Yes

Validation

None. Values larger than 2 cause a number of WARNING: AS_2_BIN: bad consistent coloring value messages to appear on the console. The system hangs immediately thereafter. A power-cycle is required to recover.

When to Change

When the primary workload of the system is a set of long-running high-performance computing (HPC) applications. Changing this value might provide better performance. File servers, database servers, and systems with a number of active processes (for example, compile or time sharing servers) do not benefit from changes.

Commitment Level

Unstable

`tsb_alloc_hiwater_factor`

Description

Initializes `tsb_alloc_hiwater` to impose an upper limit on the amount of physical memory that can be allocated for translation storage buffers (TSBs) as follows:

`tsb_alloc_hiwater` = physical memory (bytes) / `tsb_alloc_hiwater_factor`

When the memory that is allocated to TSBs is equal to the value of `tsb_alloc_hiwater`, the TSB memory allocation algorithm attempts to reclaim TSB memory as pages are unmapped.

Exercise caution when using this factor to increase the value of `tsb_alloc_hiwater`. To prevent system hangs, the resulting high water value must be considerably lower than the value of `swapfs_minfree` and `segspt_minfree`.

Data Type

Integer

Default

32

Range

1 to MAXINIT

Note that a factor of 1 makes all physical memory available for allocation to TSBs, which could cause the system to hang. A factor that is too high will not leave memory available for allocation to TSBs, decreasing system performance.

Dynamic?

Yes

Validation

None

When to Change

Change the value of this parameter if the system has many processes that attach to very large shared memory segments. Under most circumstances, tuning of this variable is not necessary.

Commitment Level

Unstable

default_tsb_size

Description Selects size of the initial translation storage buffers (TSBs) allocated to all processes.

Data Type Integer

Default Default is 0 (8 Kbytes)

Range Possible values are:

Value	Description
0	8 Kbytes
1	16 Kbytes
3	32 Kbytes
4	128 Kbytes
5	256 Kbytes
6	512 Kbytes
7	1 Mbyte

Dynamic? Yes

Validation None

When to Change Generally, you do not need to change this value. However, doing so might provide some advantages if the majority of processes on the system have a larger than average working set, or if resident set size (RSS) sizing is disabled.

Commitment Level Unstable

enable_tsb_rss_sizing

Description Enables a resident set size (RSS) based TSB sizing heuristic.

Data Type Boolean

Default 1 (TSBs can be resized)

Range 0 (TSBs can be resized) or 1 (TSBs remain at `tsb_default_size`)

Dynamic? Yes

Validation Yes

When to Change	Do not change this tunable.
Commitment Level	Unstable

tsb_rss_factor

Description	Controls the RSS to TSB span ratio of the RSS sizing heuristic. This factor divided by 512 yields the percentage of the TSB span which must be resident in memory before the TSB is considered as a candidate for resizing.
Data Type	Integer
Default	384, resulting in a value of 75%, since some virtual addresses are expected to map to the same slot in the TSB.
Range	0 to 512
Dynamic?	Yes
Validation	None
When to Change	If an application with a small address space is seeing TSB misses due to virtual address conflicts in the TSB, you might consider decreasing this value toward 0. For example, changing `tsb_rss_factor` to 50% instead of 75% might help eliminate virtual address conflicts in the TSB in some cases, but will use more kernel memory, particularly on a heavily loaded system.
Commitment Level	Unstable

Solaris Volume Manager Parameters

md_mirror:md_resync_bufsz

Description	Sets the size of the buffer used for resynchronizing RAID 1 volumes (mirrors) as the number of 512-byte blocks in the buffer. Setting larger values can increase resynchronization speed.
Data Type	Integer

Default	The default value is 128, which is acceptable for small systems. Larger systems could use higher values to increase mirror resynchronization speed.
Range	128 to 2048
Units	Blocks (512 bytes)
Dynamic?	No
Validation	None
When to Change	If you use Solaris Volume Manager RAID 1 volumes (mirrors), and you want to increase the speed of mirror resynchronizations. Assuming that you have adequate memory for overall system performance, you can increase this value without causing other performance problems. If you need to increase the speed of mirror resynchronizations, increase the value of this parameter incrementally (using 128-block increments) until performance is satisfactory. On fairly large or relatively new systems, a value of 2048 seems to be optimal. High values on older systems might hang the system.
Commitment Level	Unstable

md:mirrored_root_flag

Description	Overrides Solaris Volume Manager requirements for replica quorum and forces Solaris Volume Manager to start if any valid state database replicas are available. The default value is disabled, which requires that a majority of all replicas are available and synchronized before Solaris Volume Manager will start.
Data Type	Boolean values
Default	0 (disabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	No
Validation	None
When to Change	Use of this parameter is not supported.

Some people using Solaris Volume Manager accept the risk of enabling this parameter if all three of the following conditions apply:

- When root (/) or other system-critical file systems are mirrored
- Only two disks or controllers are available
- An unattended reboot of the system is required

If this parameter is enabled, the system might boot with a stale replica that inaccurately represents the system state (including which mirror sides are good or in Maintenance state). This situation could result in data corruption or system corruption.

Change this parameter only if system availability is more important than data consistency and integrity. Closely monitor the system for any failures. You can mitigate the risk by keeping the number of failed, Maintenance, or hot-swapped volumes as low as possible.

For more information about state database replicas, see Chapter 6, “State Database (Overview),” in *Solaris Volume Manager Administration Guide*.

Commitment Level Unstable

Network Driver Parameters

`intr_blank_time` and `intr_blank_packets`

Description These parameters affect on-board network throughput and latency on SPARC systems.

If interrupt blanking is disabled, packets are processed by the driver as soon as they arrive, resulting in higher network throughput and lower latency, but with higher CPU utilization. With interrupt blanking disabled, processor utilization can be as high as 80–90 percent in some high-load web server environments.

If interrupt blanking is enabled, packets are processed when the interrupt is issued. Enabling interrupt blanking can result in reduced processor utilization and network throughput, but higher network latency.

Both parameters should be set at the same time. You can set these parameters by using the ndd command as follows:

```
# ndd -set /dev/eri intr_blank_time 0
# ndd -set /dev/eri intr_blank_packets 0
```

You can add them to the /etc/system file as follows:

```
set eri:intr_blank_time 0
set eri:intr_blank_packets 0
```

Default

Both parameters are enabled on SPARC systems with an eri driver.

Both parameters are disabled on SPARC systems with an hme driver.

Range

0 (disabled) or 1 (enabled)

Dynamic?

Yes

Validation

None

When to Change

The value of the interrupt blanking parameter is a trade-off between network throughput and processor utilization. If higher processor utilization is acceptable for achieving higher network throughput, then disable interrupt blanking. If lower processor utilization is preferred and higher network latency is the penalty, then enable interrupt blanking.

Commitment Level

Unstable

CHAPTER 3

NFS Tunable Parameters

This section describes the NFS tunable parameters.

- "Tuning the NFS Environment" on page 93
- "NFS Module Parameters" on page 94
- "nfssrv Module Parameters" on page 124
- "rpcmod Module Parameters" on page 127

Where to Find Tunable Parameter Information

Tunable Parameter	For Information
Solaris kernel tunables	Chapter 2
Internet Protocol Suite tunable parameters	Chapter 4
Network Cache and Accelerator (NCA) tunable parameters	Chapter 5

Tuning the NFS Environment

You can define NFS parameters in the /etc/system file, which is read during the boot process. Each parameter includes the name of its associated kernel module. For more information, see "Tuning a Solaris System" on page 22.

Caution – The names of the parameters, the modules that they reside in, and the default values can change between releases. Check the documentation for the version of the active SunOS release before making changes or applying values from previous releases.

NFS Module Parameters

This section describes parameters related to the NFS kernel module.

nfs:nfs3_pathconf_disable_cache

Description	Controls the caching of `pathconf` information for NFS Version 3 mounted file systems.
Data Type	Integer (32-bit)
Default	0 (caching enabled)
Range	0 (caching enabled) or 1 (caching disabled)
Units	Boolean values
Dynamic?	Yes
Validation	None
When to Change	The `pathconf` information is cached on a per file basis. However, if the server can change the information for a specific file dynamically, use this parameter to disable caching. There is no mechanism for the client to validate its cache entry.
Commitment Level	Unstable

nfs:nfs4_pathconf_disable_cache

Description	Controls the caching of `pathconf` information for NFS Version 4 mounted file systems.
Data Type	Integer (32-bit)
Default	0 (caching enabled)
Range	0 (caching enabled) or 1 (caching disabled)

Units	Boolean values
Dynamic?	Yes
Validation	None
When to Change	The `pathconf` information is cached on a per file basis. However, if the server can change the information for a specific file dynamically, use this parameter to disable caching. There is no mechanism for the client to validate its cache entry.
Commitment Level	Unstable

`nfs:nfs_allow_preepoch_time`

Description	Controls whether files with incorrect or *negative* time stamps should be made visible on the client. Historically, neither the NFS client nor the NFS server would do any range checking on the file times being returned. The over-the-wire timestamp values are unsigned and 32-bits long. So, all values have been legal. However, on a system running a 32-bit Solaris kernel, the timestamp values are signed and 32-bits long. Thus, it would be possible to have a timestamp representation that appeared to be prior to January 1, 1970, or *pre-epoch*. The problem on a system running a 64-bit Solaris kernel is slightly different. The timestamp values on the 64-bit Solaris kernel are signed and 64-bits long. It is impossible to determine whether a time field represents a full 32-bit time or a negative time, that is, a time prior to January 1, 1970. It is impossible to determine whether to sign extend a time value when converting from 32 bits to 64 bits. The time value should be sign extended if the time value is truly a negative number. However, the time value should not be sign extended if it does truly represent a full 32-bit time value. This problem is resolved by simply disallowing full 32-bit time values.
Data Type	Integer (32-bit)
Default	0 (32-bit time stamps disabled)
Range	0 (32-bit time stamps disabled) or 1 (32-bit time stamps enabled)
Units	Boolean values

Dynamic?	Yes
Validation	None
When to Change	Even during normal operation, it is possible for the timestamp values on some files to be set very far in the future or very far in the past. If access to these files is preferred using NFS mounted file systems, set this parameter to 1 to allow the timestamp values to be passed through unchecked.
Commitment Level	Unstable

nfs:nfs_cots_timeo

Description	Controls the default RPC timeout for NFS version 2 mounted file systems using connection-oriented transports such as TCP for the transport protocol.
Data Type	Signed integer (32-bit)
Default	600 (60 seconds)
Range	0 to 2^{31} - 1
Units	10th of seconds
Dynamic?	Yes, but the RPC timeout for a file system is set when the file system is mounted. To affect a particular file system, unmount and mount the file system after changing this parameter.
Validation	None
When to Change	TCP does a good job ensuring requests and responses are delivered appropriately. However, if the round-trip times are very large in a particularly slow network, the NFS version 2 client might time out prematurely. Increase this parameter to prevent the client from timing out incorrectly. The range of values is very large, so increasing this value too much might result in situations where a retransmission is not detected for long periods of time.
Commitment Level	Unstable

nfs:nfs3_cots_timeo

Description	Controls the default RPC timeout for NFS version 3 mounted file systems using connection-oriented transports such as TCP for the transport protocol.

Data Type	Signed integer (32-bit)
Default	600 (60 seconds)
Range	0 to 2^{31} - 1
Units	10th of seconds
Dynamic?	Yes, but the RPC timeout for a file system is set when the file system is mounted. To affect a particular file system, unmount and mount the file system after changing this parameter.
Validation	None
When to Change	TCP does a good job ensuring requests and responses are delivered appropriately. However, if the round-trip times are very large in a particularly slow network, the NFS version 3 client might time out prematurely. Increase this parameter to prevent the client from timing out incorrectly. The range of values is very large, so increasing this value too much might result in situations where a retransmission is not detected for long periods of time.
Commitment Level	Unstable

nfs:nfs4_cots_timeo

Description	Controls the default RPC timeout for NFS version 4 mounted file systems using connection-oriented transports such as TCP for the transport protocol. The NFS Version 4 protocol specification disallows retransmission over the same TCP connection. Thus, this parameter primarily controls how quickly the client responds to certain events, such as detecting a forced unmount operation or detecting how quickly the server fails over to a new server.
Data Type	Signed integer (32-bit)
Default	600 (60 seconds)
Range	0 to 2^{31} - 1
Units	10th of seconds
Dynamic?	Yes, but this parameter is set when the file system is mounted. To affect a particular file system, unmount and mount the file system after changing this parameter.
Validation	None

When to Change	TCP does a good job ensuring requests and responses are delivered appropriately. However, if the round-trip times are very large in a particularly slow network, the NFS version 4 client might time out prematurely. Increase this parameter to prevent the client from timing out incorrectly. The range of values is very large, so increasing this value too much might result in situations where a retransmission is not detected for long periods of time.
Commitment Level	Unstable

nfs:nfs_do_symlink_cache

Description	Controls whether the contents of symbolic link files are cached for NFS version 2 mounted file systems.
Data Type	Integer (32–bit)
Default	1 (caching enabled)
Range	0 (caching disabled) or 1 (caching enabled)
Units	Boolean values
Dynamic?	Yes
Validation	None
When to Change	If a server changes the contents of a symbolic link file without updating the modification timestamp on the file or if the granularity of the timestamp is too large, then changes to the contents of the symbolic link file might not be visible on the client for extended periods. In this case, use this parameter to disable the caching of symbolic link contents. Doing so makes the changes immediately visible to applications running on the client.
Commitment Level	Unstable

nfs:nfs3_do_symlink_cache

Description	Controls whether the contents of symbolic link files are cached for NFS version 3 mounted file systems.
Data Type	Integer (32-bit)
Default	1 (caching enabled)

Range	0 (caching disabled) or 1 (caching enabled)
Units	Boolean values
Dynamic?	Yes
Validation	None
When to Change	If a server changes the contents of a symbolic link file without updating the modification timestamp on the file or if the granularity of the timestamp is too large, then changes to the contents of the symbolic link file might not be visible on the client for extended periods. In this case, use this parameter to disable the caching of symbolic link contents. Doing so makes the changes immediately visible to applications running on the client.
Commitment Level	Unstable

nfs:nfs4_do_symlink_cache

Description	Controls whether the contents of symbolic link files are cached for NFS version 4 mounted file systems.
Data Type	Integer (32-bit)
Default	1 (caching enabled)
Range	0 (caching disabled) or 1 (caching enabled)
Units	Boolean values
Dynamic?	Yes
Validation	None
When to Change	If a server changes the contents of a symbolic link file without updating the modification timestamp on the file or if the granularity of the timestamp is too large, then changes to the contents of the symbolic link file might not be visible on the client for extended periods. In this case, use this parameter to disable the caching of symbolic link contents. Doing so makes the changes immediately visible to applications running on the client.
Commitment Level	Unstable

nfs:nfs_dynamic

Description	Controls whether a feature known as *dynamic retransmission* is enabled for NFS version 2 mounted file systems using connectionless transports such as UDP. This feature attempts to reduce retransmissions by monitoring server response times and then adjusting RPC timeouts and read- and write- transfer sizes.
Data Type	Integer (32-bit)
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes, but this parameter is set per file system at mount time. To affect a particular file system, unmount and mount the file system after changing this parameter.
Validation	None
When to Change	Do not change this parameter.
Commitment Level	Unstable

nfs:nfs3_dynamic

Description	Controls whether a feature known as *dynamic retransmission* is enabled for NFS version 3 mounted file systems using connectionless transports such as UDP. This feature attempts to reduce retransmissions by monitoring server response times and then adjusting RPC timeouts and read- and write- transfer sizes.
Data Type	Integer (32-bit)
Default	0 (disabled)
Range	0 (disabled) or 1 (enabled)
Units	Boolean values
Dynamic?	Yes, but this parameter is set per file system at mount time. To affect a particular file system, unmount and mount the file system after changing this parameter.
Validation	None
When to Change	Do not change this parameter.
Commitment Level	Unstable

nfs:nfs4_dynamic

Description	Controls whether a feature known as *dynamic retransmission* is enabled for NFS version 4 mounted file systems using connectionless transports such as UDP. This feature attempts to reduce retransmissions by monitoring server response times and then adjusting RPC timeouts and read- and write- transfer sizes.
Data Type	Integer (32-bit)
Default	0 (disabled)
Range	0 (disabled) or 1 (enabled)
Units	Boolean values
Dynamic?	Yes, but this parameter is set per file system at mount time. To affect a particular file system, unmount and mount the file system after changing this parameter.
Validation	None
When to Change	Do not change this parameter.
Commitment Level	Unstable

nfs:nfs_lookup_neg_cache

Description	Controls whether a negative name cache is used for NFS version 2 mounted file systems. This negative name cache records file names that were looked up, but not found. The cache is used to avoid over-the-network look-up requests made for file names that are already known to not exist.
Data Type	Integer (32-bit)
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Units	Boolean values
Dynamic?	Yes
Validation	None
When to Change	For the cache to perform correctly, negative entries must be strictly verified before they are used. This consistency mechanism is relaxed slightly for read-only mounted file systems. It is assumed that the file system on the server is not

	changing or is changing very slowly, and that it is okay for such changes to propagate slowly to the client. The consistency mechanism becomes the normal attribute cache mechanism in this case. If file systems are mounted read-only on the client, but are expected to change on the server and these changes need to be seen immediately by the client, use this parameter to disable the negative cache. If you disable the `nfs:nfs_disable_rddir_cache` parameter, you should probably also disable this parameter. For more information, see "`nfs:nfs_disable_rddir_cache`" on page 113.
Commitment Level	Unstable

nfs:nfs3_lookup_neg_cache

Description	Controls whether a negative name cache is used for NFS version 3 mounted file systems. This negative name cache records file names that were looked up, but were not found. The cache is used to avoid over-the-network look-up requests made for file names that are already known to not exist.
Data Type	Integer (32-bit)
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Units	Boolean values
Dynamic?	Yes
Validation	None
When to Change	For the cache to perform correctly, negative entries must be strictly verified before they are used. This consistency mechanism is relaxed slightly for read-only mounted file systems. It is assumed that the file system on the server is not changing or is changing very slowly, and that it is okay for such changes to propagate slowly to the client. The consistency mechanism becomes the normal attribute cache mechanism in this case. If file systems are mounted read-only on the client, but are expected to change on the server and these changes need to be seen immediately by the client, use this parameter to disable the negative cache.

	If you disable the `nfs:nfs_disable_rddir_cache` parameter, you should probably also disable this parameter. For more information, see “`nfs:nfs_disable_rddir_cache`” on page 113.
Commitment Level	Unstable

nfs:nfs4_lookup_neg_cache

Description	Controls whether a negative name cache is used for NFS version 4 mounted file systems. This negative name cache records file names that were looked up, but were not found. The cache is used to avoid over-the-network look-up requests made for file names that are already known to not exist.
Data Type	Integer (32-bit)
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Units	Boolean values
Dynamic?	Yes
Validation	None
When to Change	For the cache to perform correctly, negative entries must be strictly verified before they are used. This consistency mechanism is relaxed slightly for read-only mounted file systems. It is assumed that the file system on the server is not changing or is changing very slowly, and that it is okay for such changes to propagate slowly to the client. The consistency mechanism becomes the normal attribute cache mechanism in this case. If file systems are mounted read-only on the client, but are expected to change on the server and these changes need to be seen immediately by the client, use this parameter to disable the negative cache. If you disable the `nfs:nfs_disable_rddir_cache` parameter, you should probably also disable this parameter. For more information, see “`nfs:nfs_disable_rddir_cache`” on page 113.
Commitment Level	Unstable

nfs:nfs_max_threads

Description

Controls the number of kernel threads that perform asynchronous I/O for the NFS version 2 client. Because NFS is based on RPC and RPC is inherently synchronous, separate execution contexts are required to perform NFS operations that are asynchronous from the calling thread.

The operations that can be executed asynchronously are read for read-ahead, readdir for readdir read-ahead, write for putpage and pageio operations, commit, and inactive for cleanup operations that the client performs when it stops using a file.

Data Type

Integer (16-bit)

Default

8

Range

0 to 2^{15} - 1

Units

Threads

Dynamic?

Yes, but this parameter is set per file system at mount time. To affect a particular file system, unmount and mount the file system after changing this parameter.

Validation

None

When to Change

To increase or reduce the number of simultaneous I/O operations that are outstanding at any given time. For example, for a very low bandwidth network, you might want to decrease this value so that the NFS client does not overload the network. Alternately, if the network is very high bandwidth, and the client and server have sufficient resources, you might want to increase this value. Doing so can more effectively utilize the available network bandwidth, and the client and server resources.

Commitment Level

Unstable

nfs:nfs3_max_threads

Description

Controls the number of kernel threads that perform asynchronous I/O for the NFS version 3 client. Because NFS is based on RPC and RPC is inherently synchronous, separate execution contexts are required to perform NFS operations that are asynchronous from the calling thread.

	The operations that can be executed asynchronously are read for read-ahead, readdir for readdir read-ahead, write for putpage and pageio requests, and commit.
Data Type	Integer (16-bit)
Default	8
Range	0 to 2^{15} - 1
Units	Threads
Dynamic?	Yes, but this parameter is set per file system at mount time. To affect a particular file system, unmount and mount the file system after changing this parameter.
Validation	None
When to Change	To increase or reduce the number of simultaneous I/O operations that are outstanding at any given time. For example, for a very low bandwidth network, you might want to decrease this value so that the NFS client does not overload the network. Alternately, if the network is very high bandwidth, and the client and server have sufficient resources, you might want to increase this value. Doing so can more effectively utilize the available network bandwidth, and the client and server resources.
Commitment Level	Unstable

nfs:nfs4_max_threads

Description	Controls the number of kernel threads that perform asynchronous I/O for the NFS version 4 client. Because NFS is based on RPC and RPC is inherently synchronous, separate execution contexts are required to perform NFS operations that are asynchronous from the calling thread. The operations that can be executed asynchronously are read for read-ahead, write-behind, directory read-ahead, and cleanup operations that the client performs when it stops using a file.
Data Type	Integer (16-bit)
Default	8
Range	0 to 2^{15} - 1
Units	Threads

Dynamic?	Yes, but this parameter is set per file system at mount time. To affect a particular file system, unmount and mount the file system after changing this parameter.
Validation	None
When to Change	To increase or reduce the number of simultaneous I/O operations that are outstanding at any given time. For example, for a very low bandwidth network, you might want to decrease this value so that the NFS client does not overload the network. Alternately, if the network is very high bandwidth, and the client and server have sufficient resources, you might want to increase this value. Doing so can more effectively utilize the available network bandwidth, and the client and server resources.
Commitment Level	Unstable

nfs:nfs_nra

Description	Controls the number of read-ahead operations that are queued by the NFS version 2 client when sequential access to a file is discovered. These read-ahead operations increase concurrency and read throughput. Each read-ahead request is generally for one logical block of file data.
Data Type	Integer (32-bit)
Default	4
Range	0 to 2^{31} - 1
Units	Logical blocks. (See "nfs:nfs_bsize" on page 113.)
Dynamic?	Yes
Validation	None
When to Change	To increase or reduce the number of read-ahead requests that are outstanding for a specific file at any given time. For example, for a very low bandwidth network or on a low memory client, you might want to decrease this value so that the NFS client does not overload the network or the system memory. Alternately, if the network is very high bandwidth, and the client and server have sufficient resources, you might want to increase this value. Doing so can more effectively utilize the available network bandwidth, and the client and server resources.
Commitment Level	Unstable

nfs:nfs3_nra

Description — Controls the number of read-ahead operations that are queued by the NFS version 3 client when sequential access to a file is discovered. These read-ahead operations increase concurrency and read throughput. Each read-ahead request is generally for one logical block of file data.

Data Type — Integer (32-bit)

Default — 1

Range — 0 to 2^{31} - 1

Units — Logical blocks. (See "nfs:nfs3_bsize" on page 114.)

Dynamic? — Yes

Validation — None

When to Change — To increase or reduce the number of read-ahead requests that are outstanding for a specific file at any given time. For example, for a very low bandwidth network or on a low memory client, you might want to decrease this value so that the NFS client does not overload the network or the system memory. Alternately, if the network is very high bandwidth and the client and server have sufficient resources, you might want to increase this value. Doing so can more effectively utilize the available network bandwidth, and the client and server resources.

Commitment Level — Unstable

nfs:nfs4_nra

Description — Controls the number of read-ahead operations that are queued by the NFS version 4 client when sequential access to a file is discovered. These read-ahead operations increase concurrency and read throughput. Each read-ahead request is generally for one logical block of file data.

Data Type — Integer (32-bit)

Default — 4

Range — 0 to 2^{31} - 1

Units — Logical blocks. (See "nfs:nfs4_bsize" on page 115.)

Dynamic? — Yes

Validation — None

When to Change

To increase or reduce the number of read-ahead requests that are outstanding for a specific file at any given time. For example, for a very low bandwidth network or on a low memory client, you might want to decrease this value so that the NFS client does not overload the network or the system memory. Alternately, if the network is very high bandwidth, and the client and server have sufficient resources, you might want to increase this value. Doing so can more effectively utilize the available network bandwidth, and the client and server resources.

Commitment Level

Unstable

`nfs:nrnode`

Description

Controls the size of the `rnode` cache on the NFS client.

The `rnode`, used by both NFS version 2, 3, and 4 clients, is the central data structure that describes a file on the NFS client. The `rnode` contains the file handle that identifies the file on the server. The `rnode` also contains pointers to various caches used by the NFS client to avoid network calls to the server. Each `rnode` has a one-to-one association with a `vnode`. The `vnode` caches file data.

The NFS client attempts to maintain a minimum number of `rnodes` to attempt to avoid destroying cached data and metadata. When an `rnode` is reused or freed, the cached data and metadata must be destroyed.

Data Type

Integer (32-bit)

Default

The default setting of this parameter is 0, which means that the value of `nrnode` should be set to the value of the `ncsize` parameter. Actually, any non positive value of `nrnode` results in `nrnode` being set to the value of `ncsize`.

Range

1 to 2^{31} - 1

Units

`rnodes`

Dynamic?

No. This value can only be changed by adding or changing the parameter in the `/etc/system` file, and then rebooting the system.

Validation

The system enforces a maximum value such that the `rnode` cache can only consume 25 percent of available memory.

When to Change

Because rnodes are created and destroyed dynamically, the system tends to settle upon a *nrnode*-size cache, automatically adjusting the size of the cache as memory pressure on the system increases or as more files are simultaneously accessed. However, in certain situations, you could set the value of nrnode if the mix of files being accessed can be predicted in advance. For example, if the NFS client is accessing a few very large files, you could set the value of nrnode to a small number so that system memory can cache file data instead of rnodes. Alternately, if the client is accessing many small files, you could increase the value of nrnode to optimize for storing file metadata to reduce the number of network calls for metadata.

Although it is not recommended, the rnode cache can be effectively disabled by setting the value of nrnode to 1. This value instructs the client to only cache 1 rnode, which means that it is reused frequently.

Commitment Level

Unstable

Change History

For information, see "nfs:nrnode (Solaris 9 8/03)" on page 183.

nfs:nfs_shrinkreaddir

Description

Some older NFS servers might incorrectly handle NFS version 2 READDIR requests for more than 1024 bytes of directory information. This problem is due to a bug in the server implementation. However, this parameter contains a workaround in the NFS version 2 client.

When this parameter is enabled, the client does not generate a READDIR request for larger than 1024 bytes of directory information. If this parameter is disabled, then the over-the-wire size is set to the lesser of either the size passed in by using the getdents system call or by using NFS_MAXDATA, which is 8192 bytes. For more information, see getdents(2).

Data Type	Integer (32-bit)
Default	0 (disabled)
Range	0 (disabled) or 1 (enabled)
Units	Boolean values
Dynamic?	Yes

Validation	None
When to Change	Examine the value of this parameter if an older NFS version 2 only server is used and interoperability problems occur when the server tries to read directories. Enabling this parameter might cause a slight decrease in performance for applications that read directories.
Commitment Level	Unstable

nfs:nfs3_shrinkreaddir

Description	Some older NFS servers might incorrectly handle NFS version 3 `READDIR` requests for more than 1024 bytes of directory information. This problem is due to a bug in the server implementation. However, this parameter contains a workaround in the NFS version 3 client. When this parameter is enabled, the client does not generate a `READDIR` request for larger than 1024 bytes of directory information. If this parameter is disabled, then the over-the-wire size is set to the minimum of either the size passed in by using the `getdents` system call or by using `MAXBSIZE`, which is 8192 bytes. For more information, see `getdents`(2).
Data Type	Integer (32-bit)
Default	0 (disabled)
Range	0 (disabled) or 1 (enabled)
Units	Boolean values
Dynamic?	Yes
Validation	None
When to Change	Examine the value of this parameter if an older NFS version 3 only server is used and interoperability problems occur when the server tries to read directories. Enabling this parameter might cause a slight decrease in performance for applications that read directories.
Commitment Level	Unstable

`nfs:nfs4_shrinkreaddir`

Description	Some NFS servers might incorrectly handle NFS version 4 `READDIR` requests for more than 1024 bytes of directory information. This problem is due to a bug in the server implementation. However, this parameter contains a workaround in the NFS version 4 client. When this parameter is enabled, the client does not generate a `READDIR` request for larger than 1024 bytes of directory information. If this parameter is disabled, then the over-the-wire size is set to the lesser of either the size passed in by using the `getdents` system call or by using `MAXBSIZE`, which is 8192 bytes. For more information, see `getdents`(2).
Data Type	Integer (32-bit)
Default	0 (disabled)
Range	0 (disabled) or 1 (enabled)
Units	Boolean values
Dynamic?	Yes
Validation	None
When to Change	Examine the value of this parameter if an NFS version 4 only server is used and interoperability problems occur when the server tries to read directories. Enabling this parameter might cause a slight performance drop for applications that read directories.
Commitment Level	Unstable

`nfs:nfs_write_error_interval`

Description	Controls the time duration in between logging `ENOSPC` and `EDQUOT` write errors received by the NFS client. This parameter affects NFS version 2, 3, and 4 clients.
Data Type	Long integer (32 bits on 32-bit platforms and 64 bits on 64-bit platforms)
Default	5 seconds
Range	0 to 2^{31} - 1 on 32-bit platforms 0 to 2^{63} - 1 on 64-bit platforms
Units	Seconds

Dynamic?	Yes
Validation	None
When to Change	Increase or decrease the value of this parameter in response to the volume of messages being logged by the client. Typically, you might want to increase the value of this parameter to decrease the number of out of space messages being printed when a full file system on a server is being actively used.
Commitment Level	Unstable
Change History	For information, see "nfs:nfs_write_error_interval (Solaris 9 8/03)" on page 183.

nfs:nfs_write_error_to_cons_only

Description	Controls whether NFS write errors are logged to the system console and syslog or to the system console only. This parameter affects messages for NFS version 2, 3, and 4 clients.
Data Type	Integer (32-bit)
Default	0 (system console and syslog)
Range	0 (system console and syslog) or 1 (system console)
Units	Boolean values
Dynamic?	Yes
Validation	None
When to Change	Examine the value of this parameter to avoid filling up the file system containing the messages logged by the syslogd daemon. When this parameter is enabled, messages are printed on the system console only and are not copied to the syslog messages file.
Commitment Level	Unstable
Change History	For information, see "nfs:nfs_write_error_to_cons_only (Solaris 9 8/03)" on page 183.

nfs:nfs_disable_rddir_cache

Description

Controls the use of a cache to hold responses from READDIR and READDIRPLUS requests. This cache avoids over-the-wire calls to the server to retrieve directory information.

Data Type

Integer (32-bit)

Default

0 (caching enabled)

Range

0 (caching enabled) or 1 (caching disabled)

Units

Boolean values

Dynamic?

Yes

Validation

None

When to Change

Examine the value of this parameter if interoperability problems develop due to a server that does not update the modification time on a directory when a file or directory is created in it or removed from it. The symptoms are that new names do not appear in directory listings after they have been added to the directory or that old names do not disappear after they have been removed from the directory.

This parameter controls the caching for NFS version 2, 3, and 4 mounted file systems. This parameter applies to all NFS mounted file systems, so caching cannot be disabled or enabled on a per file system basis.

If you disable this parameter, you should also disable the following parameters to to prevent bad entries in the DNLC negative cache:

- "nfs:nfs_lookup_neg_cache" on page 101
- "nfs:nfs3_lookup_neg_cache" on page 102
- "nfs:nfs4_lookup_neg_cache" on page 103

Commitment Level

Unstable

Change History

For information, see "nfs:nfs_disable_rddir_cache (Solaris 9 8/03)" on page 183.

nfs:nfs_bsize

Description

Controls the logical block size used by the NFS version 2 client. This block size represents the amount of data that the client attempts to read from or write to the server when it needs to do an I/O.

Data Type Unsigned integer (32-bit)

Default 8192 bytes

Range 0 to 2^{31} - 1

Units Bytes

Dynamic? Yes, but the block size for a file system is set when the file system is mounted. To affect a particular file system, unmount and mount the file system after changing this parameter.

Validation None. Setting this parameter too low or too high might cause the system to malfunction. Do not set this parameter to anything less than `PAGESIZE` for the specific platform. Do not set this parameter too high because it might cause the system to hang while waiting for memory allocations to be granted.

When to Change Do not change this parameter.

Commitment Level Unstable

`nfs:nfs3_bsize`

Description Controls the logical block size used by the NFS version 3 client. This block size represents the amount of data that the client attempts to read from or write to the server when it needs to do an I/O.

Data Type Unsigned integer (32-bit)

Default 32,768 (32 Kbytes)

Range 0 to 2^{31} - 1

Units Bytes

Dynamic? Yes, but the block size for a file system is set when the file system is mounted. To affect a particular file system, unmount and mount the file system after changing this parameter.

Validation None. Setting this parameter too low or too high might cause the system to malfunction. Do not set this parameter to anything less than `PAGESIZE` for the specific platform. Do not set this parameter too high because it might cause the system to hang while waiting for memory allocations to be granted.

When to Change Examine the value of this parameter when attempting to change the maximum data transfer size. Change this parameter in conjunction with the `nfs:nfs3_max_transfer_size` parameter. If larger transfers are preferred, increase both parameters. If smaller transfers are preferred, then just reducing this parameter should suffice.

Commitment Level
Unstable

`nfs:nfs4_bsize`

Description
Controls the logical block size used by the NFS version 4 client. This block size represents the amount of data that the client attempts to read from or write to the server when it needs to do an I/O.

Data Type
Unsigned integer (32-bit)

Default
32,768 (32 Kbytes)

Range
0 to 2^{31} - 1

Units
Bytes

Dynamic?
Yes, but the block size for a file system is set when the file system is mounted. To affect a particular file system, unmount and mount the file system after changing this parameter.

Validation
None. Setting this parameter too low or too high might cause the system to malfunction. Do not set this parameter to anything less than `PAGESIZE` for the specific platform. Do not set this parameter too high because it might cause the system to hang while waiting for memory allocations to be granted.

When to Change
Examine the value of this parameter when attempting to change the maximum data transfer size. Change this parameter in conjunction with the `nfs:nfs4_max_transfer_size` parameter. If larger transfers are preferred, increase both parameters. If smaller transfers are preferred, then just reducing this parameter should suffice.

Commitment Level
Unstable

nfs:nfs_async_clusters

Description

Controls the mix of asynchronous requests that are generated by the NFS version 2 client. The four types of asynchronous requests are read-ahead, putpage, pageio, and readdir-ahead. The client attempts to round-robin between these different request types to attempt to be fair and not starve one request type in favor of another.

However, the functionality in some NFS version 2 servers such as write gathering depends upon certain behaviors of existing NFS Version 2 clients. In particular, this functionality depends upon the client sending out multiple WRITE requests at about the same time. If one request is taken out of the queue at a time, the client would be defeating this server functionality designed to enhance performance for the client.

Thus, use this parameter to control the number of requests of each request type that are sent out before changing types.

Data Type

Unsigned integer (32-bit)

Default

1

Range

0 to 2^{31} - 1

Units

Asynchronous requests

Dynamic?

Yes, but the cluster setting for a file system is set when the file system is mounted. To affect a particular file system, unmount and mount the file system after changing this parameter.

Validation

None. However, setting the value of this parameter to 0 causes all of the queued requests of a particular request type to be processed before moving on to the next type. This effectively disables the fairness portion of the algorithm.

When to Change

To increase the number of each type of asynchronous request that is generated before switching to the next type. Doing so might help with server functionality that depends upon clusters of requests coming from the client.

Commitment Level

Unstable

nfs:nfs3_async_clusters

Description

Controls the mix of asynchronous requests that are generated by the NFS version 3 client. The five types of asynchronous requests are read-ahead, putpage, pageio, readdir-ahead, and commit. The client attempts to round-robin between these different request types to attempt to be fair and not starve one request type in favor of another.

However, the functionality in some NFS version 3 servers such as write gathering depends upon certain behaviors of existing NFS version 3 clients. In particular, this functionality depends upon the client sending out multiple WRITE requests at about the same time. If one request is taken out of the queue at a time, the client would be defeating this server functionality designed to enhance performance for the client.

Thus, use this parameter to control the number of requests of each request type that are sent out before changing types.

Data Type	Unsigned integer (32-bit)
Default	1
Range	0 to 2^{31} - 1
Units	Asynchronous requests
Dynamic?	Yes, but the cluster setting for a file system is set when the file system is mounted. To affect a particular file system, unmount and mount the file system after changing this parameter.
Validation	None. However, setting the value of this parameter to 0 causes all of the queued requests of a particular request type to be processed before moving on to the next type. This value effectively disables the fairness portion of the algorithm.
When to Change	To increase the number of each type of asynchronous operation that is generated before switching to the next type. Doing so might help with server functionality that depends upon clusters of operations coming from the client.
Commitment Level	Unstable

nfs:nfs4_async_clusters

Description	Controls the mix of asynchronous requests that are generated by the NFS version 4 client. The six types of asynchronous requests are read-ahead, putpage, pageio, readdir-ahead, commit, and inactive. The client attempts to round-robin between these different request types to attempt to be fair and not starve one request type in favor of another.

However, the functionality in some NFS version 4 servers such as write gathering depends upon certain behaviors of existing NFS version 4 clients. In particular, this functionality depends upon the client sending out multiple WRITE requests at about the same time. If one request is taken out of the queue at a time, the client would be defeating this server functionality designed to enhance performance for the client.

Thus, use this parameter to control the number of requests of each request type that are sent out before changing types.

Data Type	Unsigned integer (32-bit)
Default	1
Range	0 to 2^{31} - 1
Units	Asynchronous requests
Dynamic?	Yes, but the cluster setting for a file system is set when the file system is mounted. To affect a particular file system, unmount and mount the file system after changing this parameter.
Validation	None. However, setting the value of this parameter to 0 causes all of the queued requests of a particular request type to be processed before moving on to the next type. This effectively disables the fairness portion of the algorithm.
When to Change	To increase the number of each type of asynchronous request that is generated before switching to the next type. Doing so might help with server functionality that depends upon clusters of requests coming from the client.
Commitment Level	Unstable

nfs:nfs_async_timeout

Description	Controls the duration of time that threads, which execute asynchronous I/O requests, sleep with nothing to do before exiting. When there are no more requests to execute, each thread goes to sleep. If no new requests come in before this timer expires, the thread wakes up and exits. If a request does arrive, a thread is woken up to execute requests until there are none again. Then, the thread goes back to sleep waiting for another request to arrive, or for the timer to expire.
Data Type	Integer (32-bit)
Default	6000 (1 minute expressed as 60 sec * 100Hz)
Range	0 to 2^{31} - 1

Units

Hz. (Typically, the clock runs at 100Hz.)

Dynamic?

Yes

Validation

None. However, setting this parameter to a non positive value causes these threads exit as soon as there are no requests in the queue for them to process.

When to Change

If the behavior of applications in the system is known precisely and the rate of asynchronous I/O requests can be predicted, it might be possible to tune this parameter to optimize performance slightly in either of the following ways:

- By making the threads expire more quickly, thus freeing up kernel resources more quickly
- By making the threads expire more slowly, thus avoiding thread create and destroy overhead

Commitment Level

Unstable

`nfs:nacache`

Description

Tunes the number of hash queues that access the file access cache on the NFS client. The file access cache stores file access rights that users have with respect to files that they are trying to access. The cache itself is dynamically allocated. However, the hash queues used to index into the cache are statically allocated. The algorithm assumes that there is one access cache entry per active file and four of these access cache entries per hash bucket. Thus, by default, the value of this parameter is set to the value of the `nrnode` parameter.

Data Type

Integer (32-bit)

Default

The default setting of this parameter is 0. This value means that the value of `nacache` should be set to the value of the `nrnode` parameter.

Range

1 to 2^{31} - 1

Units

Access cache entries

Dynamic?

No. This value can only be changed by adding or changing the parameter in the `/etc/system` file, and then rebooting system.

Validation

None. However, setting this parameter to a negative value will probably cause the system to try to allocate a very large set of hash queues. While trying to do so, the system is likely to hang.

When to Change

Examine the value of this parameter if the basic assumption of one access cache entry per file would be violated. This violation could occur for systems in a timesharing mode where multiple users are accessing the same file at about the same time. In this case, it might be helpful to increase the expected size of the access cache so that the hashed access to the cache stays efficient.

Commitment Level

Unstable

nfs:nfs3_jukebox_delay

Description

Controls the duration of time that the NFS version 3 client waits to transmit a new request after receiving the NFS3ERR_JUKEBOX error from a previous request. The NFS3ERR_JUKEBOX error is generally returned from the server when the file is temporarily unavailable for some reason. This error is generally associated with hierarchical storage, and CD or tape jukeboxes.

Data Type

Long integer (32 bits on 32-bit platforms and 64 bits on 64-bit platforms)

Default

1000 (10 seconds expressed as 10 sec * 100Hz)

Range

0 to 2^{31} - 1 on 32-bit platforms

0 to 2^{63} - 1 on 64-bit platforms

Units

Hz. (Typically, the clock runs at 100Hz.)

Dynamic?

Yes

Validation

None

When to Change

Examine the value of this parameter and perhaps adjust it to match the behaviors exhibited by the server. Increase this value if the delays in making the file available are long in order to reduce network overhead due to repeated retransmissions. Decrease this value to reduce the delay in discovering that the file has become available.

Commitment Level

Unstable

nfs:nfs3_max_transfer_size

Description

Controls the maximum size of the data portion of an NFS version 3 READ, WRITE, READDIR, or READDIRPLUS request. This parameter controls both the maximum size of the request that the server returns as well as the maximum size of the request that the client generates.

Data Type

Integer (32-bit)

Default

1,048,576 (1 Mbyte)

Range

0 to 2^{31} - 1

Units

Bytes

Dynamic?

Yes, but this parameter is set per file system at mount time. To affect a particular file system, unmount and mount the file system after changing this parameter.

Validation

None. However, setting the maximum transfer size on the server to 0 is likely to cause clients to malfunction or just decide not to attempt to talk to the server.

There is also a limit on the maximum transfer size when using NFS over the UDP transport. UDP has a hard limit of 64 Kbytes per datagram. This 64 Kbytes must include the RPC header as well as other NFS information, in addition to the data portion of the request. Setting the limit too high might result in errors from UDP and communication problems between the client and the server.

When to Change

To tune the size of data transmitted over the network. In general, the `nfs:nfs3_bsize` parameter should also be updated to reflect changes in this parameter.

For example, when you attempt to increase the transfer size beyond 32 Kbytes, update `nfs:nfs3_bsize` to reflect the increased value. Otherwise, no change in the over-the-wire request size is observed. For more information, see "`nfs:nfs3_bsize`" on page 114.

If you want to use a smaller transfer size than the default transfer size, use the `mount` command's `-wsize` or `-rsize` option on a per-file system basis.

Commitment Level

Unstable

Change History

For information, see "`nfs:nfs3_max_transfer_size` (Solaris 9 8/03)" on page 183.

nfs:nfs4_max_transfer_size

Description	Controls the maximum size of the data portion of an NFS version 4 READ, WRITE, READDIR, or READDIRPLUS request. This parameter controls both the maximum size of the request that the server returns as well as the maximum size of the request that the client generates.
Data Type	Integer (32-bit)
Default	32, 768 (32 Kbytes)
Range	0 to 2^{31} - 1
Units	Bytes
Dynamic?	Yes, but this parameter is set per file system at mount time. To affect a particular file system, unmount and mount the file system after changing this parameter.
Validation	None. However, setting the maximum transfer size on the server to 0 is likely to cause clients to malfunction or just decide not to attempt to talk to the server. There is also a limit on the maximum transfer size when using NFS over the UDP transport. For more information on the maximum for UDP, see "nfs:nfs3_max_transfer_size" on page 121.
When to Change	To tune the size of data transmitted over the network. In general, the nfs:nfs4_bsize parameter should also be updated to reflect changes in this parameter. For example, when you attempt to increase the transfer size beyond 32 Kbytes, update nfs:nfs4_bsize to reflect the increased value. Otherwise, no change in the over-the-wire request size is observed. For more information, see "nfs:nfs4_bsize" on page 115. If you want to use a smaller transfer size than the default transfer size, use the mount command's -wsize or -rsize option on a per-file system basis.
Commitment Level	Unstable

nfs:nfs3_max_transfer_size_clts

Description	Controls the maximum size of the data portion of an NFS version 3 `READ`, `WRITE`, `READDIR`, or `READDIRPLUS` request over UDP. This parameter controls both the maximum size of the request that the server returns as well as the maximum size of the request that the client generates.
Data Type	Integer (32-bit)
Default	32, 768 (32 Kbytes)
Range	0 to 2^{31} - 1
Units	Bytes
Dynamic?	Yes, but this parameter is set per file system at mount time. To affect a particular file system, unmount and mount the file system after changing this parameter.
Validation	None. However, setting the maximum transfer size on the server to 0 is likely to cause clients to malfunction or just decide not to attempt to talk to the server.
When to Change	Do not change this parameter.
Commitment Level	Unstable

nfs:nfs3_max_transfer_size_cots

Description	Controls the maximum size of the data portion of an NFS version 3 `READ`, `WRITE`, `READDIR`, or `READDIRPLUS` request over TCP. This parameter controls both the maximum size of the request that the server returns as well as the maximum size of the request that the client generates.
Data Type	Integer (32-bit)
Default	1048576 bytes
Range	0 to 2^{31} - 1
Units	Bytes
Dynamic?	Yes, but this parameter is set per file system at mount time. To affect a particular file system, unmount and mount the file system after changing this parameter.
Validation	None. However, setting the maximum transfer size on the server to 0 is likely to cause clients to malfunction or just decide not to attempt to talk to the server.

When to Change	Do not change this parameter unless transfer sizes larger than 1 Mbyte are preferred.
Commitment Level	Unstable

`nfssrv` Module Parameters

This section describes NFS parameters for the `nfssrv` module.

`nfssrv:nfs_portmon`

Description	Controls some security checking that the NFS server attempts to do to enforce integrity on the part of its clients. The NFS server can check whether the source port from which a request was sent was a *reserved port*. A reserved port has a number less than 1024. For BSD-based systems, these ports are reserved for processes being run by root. This security checking can prevent users from writing their own RPC-based applications that defeat the access checking that the NFS client uses.
Data Type	Integer (32-bit)
Default	0 (security checking disabled)
Range	0 (security checking disabled) or 1 (security checking enabled)
Units	Boolean values
Dynamic?	Yes
Validation	None
When to Change	Use this parameter to prevent malicious users from gaining access to files by using the NFS server that they would not ordinarily have access to. However, the *reserved port* notion is not universally supported. Thus, the security aspects of the check are very weak. Also, not all NFS client implementations bind their transport endpoints to a port number in the reserved range. Thus, interoperability problems might result if the security checking is enabled.
Commitment Level	Unstable

nfssrv:rfs_write_async

Description	Controls the behavior of the NFS version 2 server when it processes WRITE requests. The NFS version 2 protocol mandates that all modified data and metadata associated with the WRITE request reside on stable storage before the server can respond to the client. NFS version 2 WRITE requests are limited to 8192 bytes of data. Thus, each WRITE request might cause multiple small writes to the storage subsystem. This can cause a performance problem. One method to accelerate NFS version 2 WRITE requests is to take advantage of a client behavior. Clients tend to send WRITE requests in batches. The server can take advantage of this behavior by clustering together the different WRITE requests into a single request to the underlying file system. Thus, the data to be written to the storage subsystem can be written in fewer, larger requests. This method can significantly increase the throughput for WRITE requests.
Data Type	Integer (32-bit)
Default	1 (clustering enabled)
Range	0 (clustering disabled) or 1 (clustering enabled)
Units	Boolean values
Dynamic?	Yes
Validation	None
When to Change	Some very small NFS clients, particularly PC clients, might not batch WRITE requests. Thus, the behavior required from the clients might not exist. In addition, the clustering in the NFS version 2 server might just add overhead and slow down performance instead of increasing it.
Commitment Level	Unstable

nfssrv:nfsauth_ch_cache_max

Description	Controls the size of the cache of client handles that contact the NFS authentication server. This server authenticates NFS clients to determine whether they are allowed access to the file handle that they are trying to use.
Data Type	Integer (32-bit)
Default	16

Range	0 to 2^{31} - 1
Units	Client handles
Dynamic?	Yes
Validation	None
When to Change	This cache is not dynamic, so attempts to allocate a client handle when all are busy will fail. This failure results in requests being dropped by the NFS server because they could not be authenticated. Most often, this result is not a problem because the NFS client just times out and retransmits the request. However, for soft-mounted file systems on the client, the client might time out, not retry the request, and then return an error to the application. This situation might be avoided if you ensure that the size of the cache on the server is large enough to handle the load.
Commitment Level	Unstable

`nfssrv:exi_cache_time`

Description	Controls the duration of time that entries are held in the NFS authentication cache before being purged due to memory pressure in the system.
Data Type	Long integer (32 bits on 32-bit platforms and 64 bits on 64-bit platforms)
Default	3600 seconds (1 hour)
Range	0 to 2^{31} - 1 on 32-bit platforms 0 to 2^{63} - 1 on 64-bit platforms
Units	Seconds
Dynamic?	Yes
Validation	None
When to Change	The size of the NFS authentication cache can be adjusted by varying the minimum age of entries that can get purged from the cache. The size of the cache should be controlled so that it is not allowed to grow too large, thus using system resources that are not allowed to be released due to this aging process.
Commitment Level	Unstable

rpcmod Module Parameters

This section describes NFS parameters for the rpcmod module.

rpcmod:clnt_max_conns

Description	Controls the number of TCP connections that the NFS client uses when communicating with each NFS server. The kernel RPC is constructed so that it can multiplex RPCs over a single connection. However, multiple connections can be used, if preferred.
Data Type	Integer (32-bit)
Default	1
Range	1 to 2^{31} - 1
Units	Connections
Dynamic?	Yes
Validation	None
When to Change	In general, one connection is sufficient to achieve full network bandwidth. However, if TCP cannot utilize the bandwidth offered by the network in a single stream, then multiple connections might increase the throughput between the client and the server. Increasing the number of connections doesn't come without consequences. Increasing the number of connections also increases kernel resource usage needed to keep track of each connection.
Commitment Level	Unstable

rpcmod:clnt_idle_timeout

Description	Controls the duration of time on the client that a connection between the client and server is allowed to remain idle before being closed.
Data Type	Long integer (32 bits on 32-bit platforms and 64 bits on 64-bit platforms)

Default	300,000 milliseconds (5 minutes)
Range	0 to 2^{31} - 1 on 32-bit platforms 0 to 2^{63} - 1 on 64-bit platforms
Units	Milliseconds
Dynamic?	Yes
Validation	None
When to Change	Use this parameter to change the time that idle connections are allowed to exist on the client before being closed. You might might want to close connections at a faster rate to avoid consuming system resources.
Commitment Level	Unstable

rpcmod:svc_idle_timeout

Description	Controls the duration of time on the server that a connection between the client and server is allowed to remain idle before being closed.
Data Type	Long integer (32 bits on 32-bit platforms and 64 bits on 64-bit platforms)
Default	360,000 milliseconds (6 minutes)
Range	0 to 2^{31} - 1 on 32-bit platforms 0 to 2^{63} - 1 on 64-bit platforms
Units	Milliseconds
Dynamic?	Yes
Validation	None
When to Change	Use this parameter to change the time that idle connections are allowed to exist on the server before being closed. You might want to close connections at a faster rate to avoid consuming system resources.
Commitment Level	Unstable

rpcmod:svc_default_stksize

Description	Sets the size of the kernel stack for kernel RPC service threads.

Data Type	Integer (32-bit)
Default	The default value is 0. This value means that the stack size is set to the system default.
Range	0 to 2^{31} - 1
Units	Bytes
Dynamic?	Yes, for all new threads that are allocated. The stack size is set when the thread is created. Therefore, changes to this parameter do not affect existing threads but are applied to all new threads that are allocated.
Validation	None
When to Change	Very deep call depths can cause the stack to overflow and cause red zone faults. The combination of a fairly deep call depth for the transport, coupled with a deep call depth for the local file system, can cause NFS service threads to overflow their stacks. Set this parameter to a multiple of the hardware `pagesize` on the platform.
Commitment Level	Unstable

`rpcmod:svc_default_max_same_xprt`

Description	Controls the maximum number of requests that are processed for each transport endpoint before switching transport endpoints. The kernel RPC works by having a pool of service threads and a pool of transport endpoints. Any one of the service threads can process requests from any one of the transport endpoints. For performance, multiple requests on each transport endpoint are consumed before switching to a different transport endpoint. This approach offers performance benefits while avoiding starvation.
Data Type	Integer (32-bit)
Default	8
Range	0 to 2^{31} - 1
Units	Requests

Dynamic? Yes, but the maximum number of requests to process before switching transport endpoints is set when the transport endpoint is configured into the kernel RPC subsystem. Changes to this parameter only affect new transport endpoints, not existing transport endpoints.

Validation None

When to Change Tune this parameter so that services can take advantage of client behaviors such as the clustering that accelerate NFS version 2 `WRITE` requests. Increasing this parameter might result in the server being better able to take advantage of client behaviors.

Commitment Level Unstable

rpcmod:maxdupreqs

Description Controls the size of the duplicate request cache that detects RPC- level retransmissions on connectionless transports. This cache is indexed by the client network address and the RPC procedure number, program number, version number, and transaction ID. This cache avoids processing retransmitted requests that might not be idempotent.

Data Type Integer (32-bit)

Default 1024

Range 1 to 2^{31} - 1

Units Requests

Dynamic? The cache is dynamically sized, but the hash queues that provide fast access to the cache are statically sized. Making the cache very large might result in long search times to find entries in the cache.

Do not set the value of this parameter to 0. This value prevents the NFS server from handling non idempotent requests.

Validation None

When to Change Examine the value of this parameter if false failures are encountered by NFS clients. For example, if an attempt to create a directory fails, but the directory is actually created, perhaps that retransmitted `MKDIR` request was not detected by the server.

The size of the cache should match the load on the server. The cache records non idempotent requests and so only needs to track a portion of the total requests. The cache does need to hold the information long enough to be able to detect a retransmission by the client. Typically, the client timeout for connectionless transports is relatively short, starting around 1 second and increasing to about 20 seconds.

Commitment Level	Unstable

rpcmod:cotsmaxdupreqs

Description	Controls the size of the duplicate request cache that detects RPC- level retransmissions on connection-oriented transports. This cache is indexed by the client network address and the RPC procedure number, program number, version number, and transaction ID. This cache avoids processing retransmitted requests that might not be idempotent.
Data Type	Integer (32–bit)
Default	1024
Range	1 to 2^{31} - 1
Units	Requests
Dynamic?	Yes
Validation	The cache is dynamically sized, but the hash queues that provide fast access to the cache are statically sized. Making the cache very large might result in long search times to find entries in the cache. Do not set the value of this parameter to 0. It prevents the NFS server from handling non-idempotent requests.
When to Change	Examine the value of this parameter if false failures are encountered by NFS clients. For example, if an attempt to create a directory fails, but the directory is actually created, it is possible that a retransmitted `MKDIR` request was not detected by the server.

The size of the cache should match the load on the server. The cache records non-idempotent requests and so only needs to track a portion of the total requests. It does need to hold the information long enough to be able to detect a retransmission on the part of the client. Typically, the client timeout for connection oriented transports is very long, about 1 minute. Thus, entries need to stay in the cache for fairly long times.

Commitment Level	Unstable

CHAPTER 4

Internet Protocol Suite Tunable Parameters

This chapter describes various Internet Protocol suite parameters, such as TCP, IP, UDP, and SCTP.

Where to Find Tunable Parameter Information

Tunable Parameter	For Information
Solaris kernel tunables	Chapter 2
NFS tunable parameters	Chapter 3
Network Cache and Accelerator (NCA) tunable parameters	Chapter 5

Overview of Tuning IP Suite Parameters

For new information about IP forwarding, see "New and Changed TCP/IP Parameters" on page 19.

You can set all of the tuning parameters described in this chapter by using the ndd command. One exception is "ipcl_conn_hash_size" on page 151. This parameter can only be set in the /etc/system file.

For example, use the following syntax to set TCP/IP parameters by using the ndd command:

```
# ndd -set driver parameter
```

For more information, see ndd(1M).

For more information on creating a startup script, see "Run Control Scripts" in *System Administration Guide: Basic Administration*.

IP Suite Parameter Validation

All parameters described in this section are checked to verify that they fall in the parameter range. The parameter's range is provided with the description for each parameter.

Internet Request for Comments (RFCs)

Internet protocol and standard specifications are described in RFC documents. You can get copies of RFCs from ftp://ftp.rfc-editor.org/in-notes. Browse RFC topics by viewing the rfc-index.txt file at this site.

IP Tunable Parameters

ip_icmp_err_interval and ip_icmp_err_burst

Description	Controls the rate of IP in generating IPv4 or IPv6 ICMP error messages. IP generates only up to ip_icmp_err_burst IPv4 or IPv6 ICMP error messages in any ip_icmp_err_interval. The ip_icmp_err_interval parameter protects IP from denial of service attacks. Setting this parameter to 0 disables rate limiting. It does not disable the generation of error messages.
Default	100 milliseconds for ip_icmp_err_interval 10 error messages for ip_icmp_err_burst
Range	0 – 99,999 milliseconds for ip_icmp_err_interval 1 – 99,999 error messages for ip_icmp_err_burst
Dynamic?	Yes
When to Change	If you need a higher error message generation rate for diagnostic purposes.
Commitment Level	Unstable

ip_respond_to_echo_broadcast and ip6_respond_to_echo_multicast

Description	Controls whether IPv4 or IPv6 responds to a broadcast ICMPv4 echo request or a multicast ICMPv6 echo request.
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes
When to Change	If you do not want this behavior for security reasons, disable it.

Commitment Level	Unstable

ip_send_redirects and ip6_send_redirects

Description	Controls whether IPv4 or IPv6 sends out ICMPv4 or ICMPv6 redirect messages.
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes
When to Change	If you do not want this behavior for security reasons, disable it.
Commitment Level	Unstable

ip_forward_src_routed and ip6_forward_src_routed

Description	Controls whether IPv4 or IPv6 forwards packets with source IPv4 routing options or IPv6 routing headers.
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes
When to Change	If you do not want this behavior for security reasons, disable it.
Commitment Level	Unstable

ip_addrs_per_if

Description	Defines the maximum number of logical interfaces associated with a real interface.
Default	256
Range	1 to 8192
Dynamic?	Yes

When to Change

Do not change the value. If more logical interfaces are required, you might consider increasing the value. However, recognize that this change might have a negative impact on IP's performance.

Commitment Level

Unstable

ip_strict_dst_multihoming and ip6_strict_dst_multihoming

Description

Determines whether a packet arriving on a non forwarding interface can be accepted for an IP address that is not explicitly configured on that interface. If ip_forwarding is enabled, or *xxx*:ip_forwarding for the appropriate interfaces is enabled, then this parameter is ignored, because the packet is actually forwarded.

Refer to RFC 1122, 3.3.4.2.

Default

0 (loose multihoming)

Range

0 = Off (loose multihoming)

1 = On (strict multihoming)

Dynamic?

Yes

When to Change

If a machine has interfaces that cross strict networking domains (for example, a firewall or a VPN node), set this parameter to 1.

Commitment Level

Unstable

ip_multidata_outbound

Description

Enables the network stack to send more than one packet at one time to the network device driver during transmission.

Enabling this parameter reduces the per-packet processing costs by improving host CPU utilization, network throughput, or both.

The multidata transmit (MDT) feature is only effective for device drivers that support this feature.

See also "tcp_mdt_max_pbufs" on page 151.

Default	1 (Enabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes
When to Change	If you do not want this parameter enabled for debugging purposes or for any other reasons, disable it.
Commitment Level	Unstable
Change History	For information, see "`ip_multidata_outbound` (Solaris 9 8/03)" on page 184.

`ip_squeue_worker_wait`

Description	Governs the maximum delay in waking up a worker thread to process TCP/IP packets that are enqueued on an squeue. An *squeue* is a serialization queue that is used by the TCP/IP kernel code to process TCP/IP packets.
Default	10 milliseconds
Range	0 – 50 milliseconds
Dynamic?	Yes
When to Change	Consider tuning this parameter if latency is an issue, and network traffic is light. For example, if the machine serves mostly interactive network traffic. The default value usually works best on a network file server, a web server, or any server that has substantial network traffic.
Commitment Level	Unstable

`ip_squeue_write`

Description	Governs the behavior of squeue operations while processing TCP/IP packets coming from the application. A value of 1 causes the running thread to process the current packet only. A value of 2 causes the running thread to process all packets that are queued on the squeue.
Default	2
Range	0 to 2
Dynamic?	Yes

When to Change

Consider changing this parameter to 1 to improve network performance for certain applications in certain situations. For example, when the number of CPUs exceed the number of active NICs, change this parameter to 1.

Otherwise, do not change this parameter.

Commitment Level

Unstable

ip_squeue_fanout

Description

Determines the mode of associating TCP/IP connections with squeues

A value of 0 associates a new TCP/IP connection with the CPU that creates the connection. A value of 1 associates the connection with a random CPU, effectively distributing the load across all CPUs and all squeues in the system.

Default

0

Range

0 or 1

Dynamic?

Yes

When to Change

Consider setting this parameter to 1 to spread the load across all CPUs in certain situations. For example, when the number of CPUs exceed the number of NICs, and one CPU is not capable of handling the network load of a single NIC, change this parameter to 1.

Commitment Level

Unstable

IP Tunable Parameters With Additional Cautions

Changing the following parameters is not recommended.

ip_ire_pathmtu_interval

Description

Specifies the interval in milliseconds when IP flushes the path maximum transfer unit (PMTU) discovery information, and tries to rediscover PMTU.

Refer to RFC 1191 on PMTU discovery.

Default

10 minutes

Range	5 seconds to 277 hours
Dynamic?	Yes
When to Change	Do not change this value.
Commitment Level	Unstable

ip_icmp_return_data_bytes and ip6_icmp_return_data_bytes

Description	When IPv4 or IPv6 sends an ICMPv4 or ICMPv6 error message, it includes the IP header of the packet that caused the error message. This parameter controls how many extra bytes of the packet beyond the IPv4 or IPv6 header are included in the ICMPv4 or ICMPv6 error message.
Default	64 bytes
Range	8 to 65,536 bytes
Dynamic?	Yes
When to Change	Do not change the value. Including more information in an ICMP error message might help in diagnosing network problems. If this feature is needed, increase the value.
Commitment Level	Unstable

TCP Tunable Parameters

tcp_deferred_ack_interval

Description	Specifies the time-out value for the TCP-delayed acknowledgment (ACK) timer for hosts that are not directly connected. Refer to RFC 1122, 4.2.3.2.
Default	100 milliseconds
Range	1 millisecond to 1 minute
Dynamic?	Yes

When to Change

Do not increase this value to more than 500 milliseconds.

Increase the value under the following circumstances:

- Slow network links (less than 57.6 Kbps) with greater than 512 bytes maximum segment size (MSS)
- The interval for receiving more than one TCP segment is short

Commitment Level

Unstable

tcp_local_dack_interval

Description

Specifies the time-out value for TCP-delayed acknowledgment (ACK) timer for hosts that are directly connected.

Refer to RFC 1122, 4.2.3.2.

Default

50 milliseconds

Range

1 millisecond to 1 minute

Dynamic?

Yes

When to Change

Do not increase this value to more than 500 milliseconds.

Increase the value under the following circumstances:

- Slow network links (less than 57.6 Kbps) with greater than 512 bytes maximum segment size (MSS)
- The interval for receiving more than one TCP segment is short

Commitment Level

Unstable

tcp_deferred_acks_max

Description

Specifies the maximum number of TCP segments received from remote destinations (not directly connected) before an acknowledgment (ACK) is generated. TCP segments are measured in units of maximum segment size (MSS) for individual connections. If set to 0 or 1, no ACKs are delayed, assuming all segments are 1 MSS long. The actual number is dynamically calculated for each connection. The value is the default maximum.

Default

2

Range

0 to 16

Dynamic?	Yes
When to Change	Do not change the value. In some circumstances, when the network traffic becomes very bursty because of the delayed ACK effect, decrease the value. Do not decrease this value below 2.
Commitment Level	Unstable

tcp_local_dacks_max

Description	Specifies the maximum number of TCP segments received from directly connected destinations before an acknowledgment (ACK) is generated. TCP segments are measured in units of maximum segment size (MSS) for individual connections. If set to 0 or 1, it means no ACKs are delayed, assuming all segments are 1 MSS long. The actual number is dynamically calculated for each connection. The value is the default maximum.
Default	8
Range	0 to 16
Dynamic?	Yes
When to Change	Do not change the value. In some circumstances, when the network traffic becomes very bursty because of the delayed ACK effect, decrease the value. Do not decrease this value below 2.
Commitment Level	Unstable

tcp_wscale_always

Description	If set to 1, TCP always sends a SYN segment with the window scale option, even if the option value is 0. Note that if TCP receives a SYN segment with the window scale option, even if the parameter is set to 0, TCP responds with a SYN segment with the window scale option. In addition, the option value is set according to the receive window size. Refer to RFC 1323 for the window scale option.
Default	0 (disabled)
Range	0 (disabled) or 1 (enabled)

Dynamic?	Yes
When to Change	If you want the window scale option in a high-speed network configuration, enable this parameter.
Commitment Level	Unstable

tcp_tstamp_always

Description	If set to 1, TCP always sends a SYN segment with the timestamp option. Note that if TCP receives a SYN segment with the timestamp option, TCP responds with a SYN segment with the timestamp option even if the parameter is set to 0.
Default	0 (disabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes
When to Change	If getting an accurate measurement of round-trip time (RTT) and TCP sequence number wraparound is a problem, enable this parameter. Refer to RFC 1323 for more reasons to enable this option.
Commitment Level	Unstable

tcp_xmit_hiwat

Description	Defines the default send window size in bytes. Refer to "Per-Route Metrics" on page 166 for a discussion of setting a different value on a per-route basis. See also "tcp_max_buf" on page 144.
Default	49,152
Range	4096 to 1,073,741,824
Dynamic?	Yes
When to Change	An application can use setsockopt(3XNET) SO_SNDBUF to change the individual connection's send buffer.
Commitment Level	Unstable

tcp_recv_hiwat

Description	Defines the default receive window size in bytes. Refer to "Per-Route Metrics" on page 166 for a discussion of setting a different value on a per-route basis. See also "tcp_max_buf" on page 144 and "tcp_recv_hiwat_minmss" on page 154.
Default	49,152
Range	2048 to 1,073,741,824
Dynamic?	Yes
When to Change	An application can use setsockopt(3XNET) SO_RCVBUF to change the individual connection's receive buffer.
Commitment Level	Unstable

tcp_max_buf

Description	Defines the maximum buffer size in bytes. This parameter controls how large the send and receive buffers are set to by an application that uses setsockopt(3XNET).
Default	1,048,576
Range	8192 to 1,073,741,824
Dynamic?	Yes
When to Change	If TCP connections are being made in a high-speed network environment, increase the value to match the network link speed.
Commitment Level	Unstable

tcp_cwnd_max

Description	Defines the maximum value of the TCP congestion window (cwnd) in bytes. For more information on the TCP congestion window, refer to RFC 1122 and RFC 2581.
Default	1,048,576
Range	128 to 1,073,741,824
Dynamic?	Yes

When to Change	Even if an application uses `setsockopt`(3XNET) to change the window size to a value higher than `tcp_cwnd_max`, the actual window used can never grow beyond `tcp_cwnd_max`. Thus, `tcp_max_buf` should be greater than `tcp_cwnd_max`.
Commitment Level	Unstable

`tcp_slow_start_initial`

Description	Defines the maximum initial congestion window (cwnd) size in the maximum segment size (MSS) of a TCP connection.
	Refer to RFC 2414 on how the initial congestion window size is calculated.
Default	4
Range	1 to 4
Dynamic?	Yes
When to Change	Do not change the value.
	If the initial cwnd size causes network congestion under special circumstances, decrease the value.
Commitment Level	Unstable

`tcp_slow_start_after_idle`

Description	The congestion window size in the maximum segment size (MSS) of a TCP connection after it has been idled (no segment received) for a period of one retransmission timeout (RTO).
	Refer to RFC 2414 on how the initial congestion window size is calculated.
Default	4
Range	1 to 16,384
Dynamic?	Yes
When to Change	For more information, see "`tcp_slow_start_initial`" on page 145.
Commitment Level	Unstable

tcp_sack_permitted

Description	If set to 2, TCP always sends a SYN segment with the selective acknowledgment (SACK) permitted option. If TCP receives a SYN segment with a SACK-permitted option and this parameter is set to 1, TCP responds with a SACK-permitted option. If the parameter is set to 0, TCP does not send a SACK-permitted option, regardless of whether the incoming segment contains the SACK permitted option. Refer to RFC 2018 for information on the SACK option.
Default	2 (active enabled)
Range	0 (disabled), 1 (passive enabled), or 2 (active enabled)
Dynamic?	Yes
When to Change	SACK processing can improve TCP retransmission performance so it should be actively enabled. Sometimes, the other side can be confused with the SACK option actively enabled. If this confusion occurs, set the value to 1 so that SACK processing is enabled only when incoming connections allow SACK processing.
Commitment Level	Unstable

tcp_rev_src_routes

Description	If set to 0, TCP does not reverse the IP source routing option for incoming connections for security reasons. If set to 1, TCP does the normal reverse source routing.
Default	0 (disabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes
When to Change	If IP source routing is needed for diagnostic purposes, enable it.
Commitment Level	Unstable

tcp_time_wait_interval

Description	Specifies the time in milliseconds that a TCP connection stays in TIME-WAIT state.

For more information, refer to RFC 1122, 4.2.2.13.

Default

60,000 (60 seconds)

Range

1 second to 10 minutes

Dynamic?

Yes

When to Change

Do not set the value lower than 60 seconds.

For information on changing this parameter, refer to RFC 1122, 4.2.2.13.

Commitment Level

Unstable

tcp_ecn_permitted

Description

Controls Explicit Congestion Notification (ECN) support.

If this parameter is set to 0, TCP does not negotiate with a peer that supports the ECN mechanism.

If this parameter is set to 1 when initiating a connection, TCP does not tell a peer that it supports ECN mechanism.

However, TCP tells a peer that it supports ECN mechanism when accepting a new incoming connection request if the peer indicates that it supports ECN mechanism in the SYN segment.

If this parameter is set to 2, in addition to negotiating with a peer on the ECN mechanism when accepting connections, TCP indicates in the outgoing SYN segment that it supports the ECN mechanism when TCP makes active outgoing connections.

Refer to RFC 3168 for information on ECN.

Default

1 (passive enabled)

Range

0 (disabled), 1 (passive enabled), or 2 (active enabled)

Dynamic?

Yes

When to Change

ECN can help TCP better handle congestion control. However, there are existing TCP implementations, firewalls, NATs, and other network devices that are confused by this mechanism. These devices do not comply to the IETF standard.

Because of these devices, the default value of this parameter is set to 1. In rare cases, passive enabling can still cause problems. Set the parameter to 0 only if absolutely necessary.

Commitment Level — Unstable

tcp_conn_req_max_q

Description — Specifies the default maximum number of pending TCP connections for a TCP listener waiting to be accepted by accept(3SOCKET). See also "tcp_conn_req_max_q0" on page 149.

Default — 128

Range — 1 to 4,294,967,296

Dynamic? — Yes

When to Change — For applications such as web servers that might receive several connection requests, the default value might be increased to match the incoming rate.

Do not increase the parameter to a very large value. The pending TCP connections can consume excessive memory. Also, if an application cannot handle that many connection requests fast enough because the number of pending TCP connections is too large, new incoming requests might be denied.

Note that increasing tcp_conn_req_max_q does not mean that applications can have that many pending TCP connections. Applications can use listen(3SOCKET) to change the maximum number of pending TCP connections for each socket. This parameter is the maximum an application can use listen() to set the number to. Thus, even if this parameter is set to a very large value, the actual maximum number for a socket might be much less than tcp_conn_req_max_q, depending on the value used in listen().

Commitment Level — Unstable

Change History — For information, see "*xxx*:ip_forwarding (Solaris 9 Releases)" on page 187.

tcp_conn_req_max_q0

Description	Specifies the default maximum number of incomplete (three-way handshake not yet finished) pending TCP connections for a TCP listener. For more information on TCP three-way handshake, refer to RFC 793. See also "tcp_conn_req_max_q" on page 148.
Default	1024
Range	0 to 4,294,967,296
Dynamic?	Yes
When to Change	For applications such as web servers that might receive excessive connection requests, you can increase the default value to match the incoming rate. The following explains the relationship between tcp_conn_req_max_q0 and the maximum number of pending connections for each socket. When a connection request is received, TCP first checks if the number of pending TCP connections (three-way handshake is done) waiting to be accepted exceeds the maximum (*N*) for the listener. If the connections are excessive, the request is denied. If the number of connections is allowable, then TCP checks if the number of incomplete pending TCP connections exceeds the sum of *N* and tcp_conn_req_max_q0. If it does not, the request is accepted. Otherwise, the oldest incomplete pending TCP request is dropped.
Commitment Level	Unstable
Change History	For information, see "*xxx*:ip_forwarding (Solaris 9 Releases)" on page 187.

tcp_conn_req_min

Description	Specifies the default minimum value for the maximum number of pending TCP connection requests for a listener waiting to be accepted. This is the lowest maximum value of listen(3SOCKET) that an application can use.
Default	1
Range	1 to 1024
Dynamic?	Yes

When to Change	This parameter can be a solution for applications that use `listen`(3SOCKET) to set the maximum number of pending TCP connections to a value too low. Increase the value to match the incoming connection request rate.
Commitment Level	Unstable

tcp_rst_sent_rate_enabled

Description	If this parameter is set to 1, the maximum rate of sending a RST segment is controlled by the `ndd` parameter, `tcp_rst_sent_rate`. If this parameter is set to 0, no rate control when sending a RST segment is available.
Default	1(enabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes
When to Change	This tunable helps defend against denial of service attacks on TCP by limiting the rate by which a RST segment is sent out. The only time this rate control should be disabled is when strict conformance to RFC 793 is required.
Commitment Level	Unstable

tcp_rst_sent_rate

Description	Sets the maximum number of RST segments that TCP can send out per second.
Default	40
Range	0 to 4,294,967,296
Dynamic?	Yes
When to Change	In a TCP environment, there might be a legitimate reason to generate more RSTs than the default value allows. In this case, increase the default value of this parameter.
Commitment Level	Unstable

tcp_mdt_max_pbufs

Description	Specifies the number of payload buffers that can be carried by a single M_MULTIDATA message that is generated by TCP. See also "ip_multidata_outbound" on page 137.
Default	16
Range	1 to 16
Dynamic?	Yes
When to Change	Decreasing this parameter might aid in debugging device driver development by limiting the amount of payload buffers per M_MULTIDATA message that is generated by TCP.
Commitment Level	Unstable

TCP/IP Parameters Set in the /etc/system File

The ipcl_conn_hash_size parameter can be set only in the /etc/system file. After the file is modified, reboot the system.

The following entry sets the ipcl_conn_hash_size parameter:

```
set ip:ipcl_conn_hash_sizes=value
```

ipcl_conn_hash_size

Description	Controls the size of the connection hash table used by IP. The default value of 0 means that the system automatically sizes an appropriate value for this parameter at boot time, depending on the available memory.
Data Type	Unsigned integer
Default	0
Range	0 to 82,500
Dynamic?	No. The parameter can only be changed at boot time.
When to Change	If the system consistently has tens of thousands of TCP connections, the value can be increased accordingly. Increasing the hash table size means that more memory is wired down, thereby reducing available memory to user applications.
Commitment Level	Unstable

TCP Parameters With Additional Cautions

Changing the following parameters is not recommended.

tcp_ip_abort_interval

Description	Specifies the default total retransmission timeout value for a TCP connection. For a given TCP connection, if TCP has been retransmitting for `tcp_ip_abort_interval` period of time and it has not received any acknowledgment from the other endpoint during this period, TCP closes this connection. For TCP retransmission timeout (RTO) calculation, refer to RFC 1122, 4.2.3. See also "`tcp_rexmit_interval_max`" on page 153.
Default	8 minutes
Range	500 milliseconds to 1193 hours
Dynamic?	Yes
When to Change	Do not change this value. See "`tcp_rexmit_interval_max`" on page 153 for exceptions.
Commitment Level	Unstable

tcp_rexmit_interval_initial

Description	Specifies the default initial retransmission timeout (RTO) value for a TCP connection. Refer to "Per-Route Metrics" on page 166 for a discussion of setting a different value on a per-route basis.
Default	3 seconds
Range	1 millisecond to 20 seconds
Dynamic?	Yes
When to Change	Do not change this value. Lowering the value can result in unnecessary retransmissions.
Commitment Level	Unstable

tcp_rexmit_interval_max

Description Defines the default maximum retransmission timeout value (RTO). The calculated RTO for all TCP connections cannot exceed this value. See also "tcp_ip_abort_interval" on page 152.

Default 60 seconds

Range 1 millisecond to 2 hours

Dynamic? Yes

When to Change Do not change the value in a normal network environment.

If, in some special circumstances, the round-trip time (RTT) for a connection is about 10 seconds, you can increase this value. If you change this value, you should also change the tcp_ip_abort_interval parameter. Change the value of tcp_ip_abort_interval to at least four times the value of tcp_rexmit_interval_max.

Commitment Level Unstable

tcp_rexmit_interval_min

Description Specifies the default minimum retransmission time out (RTO) value. The calculated RTO for all TCP connections cannot be lower than this value. See also "tcp_rexmit_interval_max" on page 153.

Default 400 milliseconds

Range 1 millisecond to 20 seconds

Dynamic? Yes

When to Change Do not change the value in a normal network environment.

TCP's RTO calculation should cope with most RTT fluctuations. If, in some very special circumstances, the round-trip time (RTT) for a connection is about 10 seconds, increase this value. If you change this value, you should change the tcp_rexmit_interval_max parameter. Change the value of tcp_rexmit_interval_max to at least eight times the value of tcp_rexmit_interval_min.

Commitment Level Unstable

tcp_rexmit_interval_extra

Description	Specifies a constant added to the calculated retransmission time out value (RTO).
Default	0 milliseconds
Range	0 to 2 hours
Dynamic?	Yes
When to Change	Do not change the value. When the RTO calculation fails to obtain a good value for a connection, you can change this value to avoid unnecessary retransmissions.
Commitment Level	Unstable

tcp_tstamp_if_wscale

Description	If this parameter is set to 1, and the window scale option is enabled for a connection, TCP also enables the `timestamp` option for that connection.
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes
When to Change	Do not change this value. In general, when TCP is used in high-speed network, protection against sequence number wraparound is essential. Thus, you need the `timestamp` option.
Commitment Level	Unstable

tcp_recv_hiwat_minmss

Description	Controls the default minimum receive window size. The minimum is `tcp_recv_hiwat_minmss` times the size of maximum segment size (MSS) of a connection.
Default	4
Range	1 to 65,536
Dynamic?	Yes

When to Change	Do not change the value. If changing it is necessary, do not change the value lower than 4.
Commitment Level	Unstable

UDP Tunable Parameters

udp_xmit_hiwat

Description	Defines the default maximum UDP socket datagram size. For more information, see "udp_max_buf" on page 156.
Default	8192 bytes
Range	4096 to 65,536 bytes
Dynamic?	Yes
When to Change	Note that an application can use setsockopt(3XNET) SO_SNDBUF to change the size for an individual socket. In general, you do not need to change the default value.
Commitment Level	Unstable

udp_recv_hiwat

Description	Defines the default maximum UDP socket receive buffer size. For more information, see "udp_max_buf" on page 156.
Default	8192 bytes
Range	4096 to 65,536 bytes
Dynamic?	Yes
When to Change	Note that an application can use setsockopt(3XNET) SO_RCVBUF to change the size for an individual socket. In general, you do not need to change the default value.
Commitment Level	Unstable

UDP Parameters With Additional Cautions

Changing the following parameters is not recommended.

udp_max_buf

Description
: Controls how large send and receive buffers can be for a UDP socket.

Default
: 262,144 bytes

Range
: 65,536 to 1,073,741,824 bytes

Dynamic?
: Yes

When to Change
: Do not change the value. If this parameter is set to a very large value, UDP socket applications can consume too much memory.

Commitment Level
: Unstable

IPQoS Tunable Parameter

ip_policy_mask

Description
: Enables or disables IPQoS processing in any of the following callout positions: forward outbound, forward inbound, local outbound, and local inbound. This parameter is a bitmask as follows:

Not Used	Not Used	Not Used	Not Used	Forward Outbound	Forward Inbound	Local Outbound	Local Inbound
X	X	X	X	0	0	0	0

A `1` in any of the position masks or disables IPQoS processing in that particular callout position. For example, a value of `0x01` disables IPQoS processing for all the local inbound packets.

Default
: The default value is 0, meaning that IPQoS processing is enabled in all the callout positions.

Range	0 (0x00) to 15 (0x0F). A value of 15 indicates that IPQoS processing is disabled in all the callout positions.
Dynamic?	Yes
When to Change	If you want to enable or disable IPQoS processing in any of the callout positions.
Commitment Level	Unstable

SCTP Tunable Parameters

`sctp_max_init_retr`

Description	Controls the maximum number of attempts an SCTP endpoint should make at resending an INIT chunk. The SCTP endpoint can use the SCTP initiation structure to override this value.
Default	8
Range	0 to 128
Dynamic?	Yes
When to Change	The number of INIT retransmissons depend on "`sctp_pa_max_retr`" on page 157. Ideally, `sctp_max_init_retr` should be less than or equal to `sctp_pa_max_retr`.
Commitment Level	Unstable

`sctp_pa_max_retr`

Description	Controls the maximum number of retransmissions (over all paths) for an SCTP association. The SCTP association is aborted when this number is exceeded.
Default	10
Range	1 to 128
Dynamic?	Yes

When to Change	The maximum number of retransmissions over all paths depend on the number of paths and the maximum number of retransmisson over each path. Ideally, `sctp_pa_max_retr` should be set to the sum of "`sctp_pp_max_retr`" on page 158 over all available paths. For example, if there are 3 paths to the destination and the maximum number of retransmissions over each of the 3 paths is 5, then `sctp_pa_max_retr` should be set to less than or equal to 15. (See the Note in Section 8.2, RFC 2960.)
Commitment Level	Unstable

sctp_pp_max_retr

Description	Controls the maximum number of retransmissions over a specific path. When this number is exceeded for a path, the path (destination) is considered unreachable.
Default	5
Range	1 to 128
Dynamic?	Yes
When to Change	Do not change this value to less than 5.
Commitment Level	Unstable

sctp_cwnd_max

Description	Controls the maximum value of the congestion window for an SCTP association.
Default	1,048,576
Range	128 to 1,073,741,824
Dynamic?	Yes
When to Change	Even if an application uses `setsockopt`(3XNET) to change the window size to a value higher than `sctp_cwnd_max`, the actual window used can never grow beyond `sctp_cwnd_max`. Thus, "`sctp_max_buf`" on page 162 should be greater than `sctp_cwnd_max`.
Commitment Level	Unstable

sctp_ipv4_ttl

Description — Controls the time to live (TTL) value in the IP version 4 header for the outbound IP version 4 packets on an SCTP association.

Default — 64

Range — 1 to 255

Dynamic? — Yes

When to Change — Generally, you do not need to change this value. Consider increasing this parameter if the path to the destination is likely to span more than 64 hops.

Commitment Level — Unstable

sctp_heartbeat_interval

Description — Computes the interval between HEARTBEAT chunks to an idle destination, that is allowed to heartbeat.

An SCTP endpoint periodically sends an HEARTBEAT chunk to monitor the reachability of the idle destinations transport addresses of its peer.

Default — 30 seconds

Range — 0 to 86,400 seconds

Dynamic? — Yes

When to Change — Refer to RFC 2960, section 8.3.

Commitment Level — Unstable

sctp_new_secret_interval

Description — Determines when a new secret needs to be generated. The generated secret is used to compute the MAC for a cookie.

Default — 2 minutes

Range — 0 to 1,440 minutes

Dynamic? — Yes

When to Change — Refer to RFC 2960, section 5.1.3.

Commitment Level — Unstable

sctp_initial_mtu

Description	Determines the initial maximum send size for an SCTP packet including the length of the IP header.
Default	1500 bytes
Range	68 to 65,535
Dynamic?	Yes
When to Change	Increase this parameter if the underlying link supports frame sizes that are greater than 1500 bytes.
Commitment Level	Unstable

sctp_deferred_ack_interval

Description	Sets the time-out value for SCTP delayed acknowledgment (ACK) timer in milliseconds.
Default	100 milliseconds
Range	1 to 60,000 milliseconds
Dynamic?	Yes
When to Change	Refer to RFC 2960, section 6.2.
Commitment Level	Unstable

sctp_ignore_path_mtu

Description	Enables or disables path MTU discovery.
Default	0 (disabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes
When to Change	Enable this parameter if you want to ignore MTU changes along the path. However, doing so might result in IP fragmentation if the path MTU decreases.
Commitment Level	Unstable

sctp_initial_ssthresh

Description Sets the initial slow start threshold for a destination address of the peer.

Default 102,400

Range 1024 to 4,294,967,295

Dynamic? Yes

When to Change Refer to RFC 2960, section 7.2.1.

Commitment Level Unstable

sctp_xmit_hiwat

Description Sets the default send window size in bytes. See also "sctp_max_buf" on page 162.

Default 102,400

Range 8,192 to 1,073,741,824

Dynamic? Yes

When to Change An application can use setsockopt(3SOCKET) SO_SNDBUF to change the individual association's send buffer.

Commitment Level Unstable

sctp_xmit_lowat

Description Controls the lower limit on the send window size.

Default 8,192

Range 8,192 to 1,073,741,824

Dynamic? Yes

When to Change Generally, you do not need to change this value. This parameter sets the minimum size required in the send buffer for the socket to be marked writable. If required, consider changing this parameter in accordance with "sctp_xmit_hiwat" on page 161.

Commitment Level Unstable

sctp_recv_hiwat

Description	Controls the default receive window size in bytes. See also "sctp_max_buf" on page 162.
Default	102,400
Range	8,192 to 1,073,741,824
Dynamic?	Yes
When to Change	An application can use setsockopt(3SOCKET) SO_RCVBUF to change the individual association's receive buffer.
Commitment Level	Unstable

sctp_max_buf

Description	Controls the maximum buffer size in bytes. It controls how large the send and receive buffers are set to by an application that uses setsockopt(3SOCKET).
Default	1,048,576
Range	8,192 to 1,073,741,824
Dynamic?	Yes
When to Change	Increase the value of this parameter to match the network link speed if associations are being made in a high-speed network environment.
Commitment Level	Unstable

sctp_ipv6_hoplimit

Description	Sets the value of the hop limit in the IP version 6 header for the outbound IP version 6 packets on an SCTP association.
Default	60
Range	0 to 255
Dynamic?	Yes
When to Change	Generally, you do not need to change this value. Consider increasing this parameter if the path to the destination is likely to span more than 60 hops.
Commitment Level	Unstable

sctp_rto_min

Description	Sets the lower bound for the retransmission timeout (RTO) in milliseconds for all the destination addresses of the peer.
Default	1,000
Range	500 to 60,000
Dynamic?	Yes
When to Change	Refer to RFC 2960, section 6.3.1.
Commitment Level	Unstable

sctp_rto_max

Description	Controls the upper bound for the retransmission timeout (RTO) in milliseconds for all the destination addresses of the peer.
Default	60,000
Range	1,000 to 60,000,000
Dynamic?	Yes
When to Change	Refer to RFC 2960, section 6.3.1.
Commitment Level	Unstable

sctp_rto_initial

Description	Controls the initial retransmission timeout (RTO) in milliseconds for all the destination addresses of the peer.
Default	3,000
Range	1,000 to 60,000,000
Dynamic?	Yes
When to Change	Refer to RFC 2960, section 6.3.1.
Commitment Level	Unstable

sctp_cookie_life

Description	Sets the lifespan of a cookie in milliseconds.

Default 60,000
Range 10 to 60,000,000

Dynamic? Yes

When to Change Generally, you do not need to change this value. This parameter might be changed in accordance with "sctp_rto_max" on page 163.

Commitment Level Unstable

sctp_max_in_streams

Description Controls the maximum number of inbound streams permitted for an SCTP association.

Default 32
Range 1 to 65,535

Dynamic? Yes

When to Change Refer to RFC 2960, section 5.1.1.

Commitment Level Unstable

sctp_initial_out_streams

Description Controls the maximum number of outbound streams permitted for an SCTP association.

Default 32
Range 1 to 65,535

Dynamic? Yes

When to Change Refer to RFC 2960, section 5.1.1.

Commitment Level Unstable

sctp_shutack_wait_bound

Description Controls the maximum time, in milliseconds, to wait for a SHUTDOWN ACK after having sent a SHUTDOWN chunk.

Default 60,000
Range 0 to 300,000

Dynamic? Yes

When to Change	Generally, you do not need to change this value. This parameter might be changed in accordance with "sctp_rto_max" on page 163.
Commitment Level	Unstable

sctp_maxburst

Description	Sets the limit on the number of segments to be sent in a burst.
Default	4
Range	2 to 8
Dynamic?	Yes
When to Change	You do not need to change this parameter. You might change it for testing purposes.
Commitment Level	Unstable

sctp_addip_enabled

Description	Enables or disables SCTP dynamic address reconfiguration.
Default	0 (disabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes
When to Change	The parameter can be enabled if dynamic address reconfiguration is needed. Due to security implications, enable this parameter only for testing purposes.
Commitment Level	Unstable

sctp_prsctp_enabled

Description	Enables or disables the partial reliability extension (RFC 3758) to SCTP.
Default	1 (enabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes
When to Change	Disable this parameter if partial reliability is not supported in your SCTP environment.

Commitment Level Unstable

Per-Route Metrics

Starting in the Solaris 8 release, you can use per-route metrics to associate some properties with IPv4 and IPv6 routing table entries.

For example, a system has two different network interfaces, a fast Ethernet interface and a gigabit Ethernet interface. The system default `tcp_recv_hiwat` is 24,576 bytes. This default is sufficient for the fast Ethernet interface, but may not be sufficient for the gigabit Ethernet interface.

Instead of increasing the system's default for `tcp_recv_hiwat`, you can associate a different default TCP receive window size to the gigabit Ethernet interface routing entry. By making this association, all TCP connections going through the route will have the increased receive window size.

For example, the following is in the routing table (`netstat -rn`), assuming IPv4:

```
192.123.123.0        192.123.123.4        U        1      4  hme0
192.123.124.0        192.123.124.4        U        1      4  ge0
default              192.123.123.1        UG       1      8
```

In this example, do the following:

```
# route change -net 192.123.124.0 -recvpipe x
```

Then, all connections going to the `192.123.124.0` network, which is on the `ge0` link, use the receive buffer size *x*, instead of the default `24567` receive window size.

If the destination is in the `a.b.c.d` network, and no specific routing entry exists for that network, you can add a prefix route to that network and change the metric. For example:

```
# route add -net a.b.c.d 192.123.123.1 -netmask w.x.y.z
# route change -net a.b.c.d -recvpipe y
```

Note that the prefix route's gateway is the default router. Then, all connections going to that network use the receive buffer size *y*. If you have more than one interface, use the `-ifp` argument to specify which interface to use. This way, you can control which interface to use for specific destinations. To verify the metric, use the `route`(1M) `get` command.

CHAPTER 5

Network Cache and Accelerator Tunable Parameters

This chapter describes some of the Network Cache and Accelerator (NCA) tunable parameters.

- "nca:nca_conn_hash_size" on page 168
- "nca:nca_conn_req_max_q" on page 168
- "nca:nca_conn_req_max_q0" on page 169
- "nca:nca_ppmax" on page 169
- "nca:nca_vpmax" on page 170
- "sq_max_size" on page 171
- "ge:ge_intr_mode" on page 171

Where to Find Tunable Parameters Information

Tunable Parameter	For Information
Solaris kernel tunables	Chapter 2
NFS tunable parameters	Chapter 3
Internet Protocol Suite tunable parameters	Chapter 4

Tuning NCA Parameters

Setting these parameters is appropriate on a system that is a dedicated web server. These parameters allocate more memory for caching pages. You can set all of the tuning parameters described in this chapter in the `/etc/system` file.

For information on adding tunable parameters to the `/etc/system` file, see "Tuning the Solaris Kernel" on page 24.

`nca:nca_conn_hash_size`

Description	Controls the hash table size in the NCA module for all TCP connections, adjusted to the nearest prime number.
Default	383 hash table entries
Range	0 to 201,326,557
Dynamic?	No
When to Change	When the NCA's TCP hash table is too small to keep track of the incoming TCP connections. This situation causes many TCP connections to be grouped together in the same hashtable entry. This situation is indicated when NCA is receiving many TCP connections, and system performance decreases.
Commitment Level	Unstable

`nca:nca_conn_req_max_q`

Description	Defines the maximum number of pending TCP connections for NCA to listen on.
Default	256 connections
Range	0 to 4,294,967,295
Dynamic?	No
When to Change	When NCA closes a connection immediately after it is established because it already has too many established TCP connections. If NCA is receiving many TCP connections and can handle a larger load, but is refusing any more connections, increase this parameter. Doing so allows NCA to handle more simultaneous TCP connections.
Commitment Level	Unstable

nca:nca_conn_req_max_q0

Description	Defines the maximum number of incomplete (three-way handshake not yet finished) pending TCP connections for NCA to listen on.
Default	1024 connections
Range	0 to 4,294,967,295
Dynamic?	No
When to Change	When NCA refuses to accept any more TCP connections because it already has too many pending TCP connections. If NCA is receiving many TCP connections and can handle a larger load, but is refusing any more connections, increase this parameter. Doing so allows NCA to handle more simultaneous TCP connections.
Commitment Level	Unstable

nca:nca_ppmax

Description	Specifies the maximum amount of physical memory (in pages) used by NCA for caching the pages. This value should not be more than 75 percent of total memory.
Default	25 percent of physical memory
Range	1 percent to maximum amount of physical memory
Dynamic?	No
When to Change	When using NCA on a system with more than 512 Mbytes of memory. If a system has a lot of physical memory that is not being used, increase this parameter. Then, NCA will efficiently use this memory to cache new objects. As a result, system performance will increase. This parameter should be increased in conjunction with `nca_vpmax`, unless you have a system with more physical memory than virtual memory (a 32-bit kernel that has greater than 4 Gbytes memory). Use `pagesize`(1) to determine your system's page size.
Commitment Level	Unstable

nca:nca_vpmax

Description — Specifies the maximum amount of virtual memory (in pages) used by NCA for caching pages. This value should not be more than 75 percent of the total memory.

Default — 25 percent of virtual memory

Range — 1 percent to maximum amount of virtual memory

Dynamic? — No

When to Change — When using NCA on a system with more than 512 Mbytes of memory. If a system has a lot of virtual memory that is not being used, increase this parameter. Then, NCA will efficiently use this memory to cache new objects. As a result, system performance will increase.

This parameter should be increased in conjunction with `nca_ppmax`. Set this parameter about the same value as `nca_vpmax`, unless you have a system with more physical memory than virtual memory.

Commitment Level — Unstable

General System Tuning for the NCA

In addition to setting the NCA parameters, you can do some general system tuning to benefit NCA performance. If you are using gigabit Ethernet (`ge` driver), you should set the interface in interrupt mode for better results.

For example, a system with 4 Gbytes of memory that is booted under 64–bit kernel should have the following parameters set in the `/etc/system` file. Use `pagesize` to determine your system's page size.

```
set sq_max_size=0
set ge:ge_intr_mode=1
set nca:nca_conn_hash_size=82500
set nca:nca_conn_req_max_q=100000
set nca:nca_conn_req_max_q0=100000
set nca:nca_ppmax=393216
set nca:nca_vpmax=393216
```

sq_max_size

Description
: Sets the depth of the syncq (number of messages) before a destination STREAMS queue generates a QFULL message.

Default
: 2 messages

Range
: 0 (unlimited) to MAXINT

Dynamic?
: No

When to Change
: When NCA is running on a system with a lot of memory, increase this parameter to allow drivers to queue more packets of data. If a server is under heavy load, increase this parameter so that modules and drivers can process more data without dropping packets or getting backlogged.

Note – sq_max_size=0 is intended only for benchmarks or testing environments.

Solaris 8 and later releases – Do not set this parameter to 0 on production systems. If you need to change this parameter, gradually increase this value, and monitor the system.

Solaris 7 and earlier releases – Do not set this parameter to 0 on production systems. If you need to change this parameter, gradually increase this value to a maximum of 100, and monitor the system.

Commitment Level
: Unstable

Change History
: For information, see "sq_max_size (Solaris 9 12/02 Release)" on page 188.

ge:ge_intr_mode

Description
: Enables the ge driver to send packets directly to the upper communication layers rather than queue the packets

Default
: 0 (queue packets to upper layers)

Range
: 0 (enable) or 1 (disable)

Dynamic?
: No

When to Change
: When NCA is enabled, set this parameter to 1 so that the packet is delivered to NCA in interrupt mode for faster processing.

Commitment Level	Unstable

CHAPTER 6

System Facility Parameters

This chapter describes most of the parameters default values for various system facilities.

System Default Parameters

The functioning of various system facilities is governed by a set of values that are read by each facility on startup. The values stored in a file for each facility are located in the `/etc/default` directory. Not every system facility has a file located in this directory.

autofs

This facility enables you to configure `autofs` parameters such as automatic timeout, displaying or logging status messages, browsing autofs mount points, and tracing. For details, see `autofs`(4).

cron

This facility enables you to disable or enable `cron` logging.

devfsadm

This file is not currently used.

dhcpagent

Client usage of DHCP is provided by the `dhcpagent` daemon. When `ifconfig` identifies an interface that has been configured to receive its network configuration from DHCP, it starts the client daemon to manage that interface.

For more information, see the `/etc/default/dhcpagent` information in the FILES section of `dhcpagent`(1M).

fs

File system administrative commands have a generic and file system-specific portion. If the file system type is not explicitly specified with the `-F` option, a default is applied. The value is specified in this file. For more information, see the Description section of `default_fs`(4).

ftp

This facility enables you to set the `ls` command behavior to the RFC 959 `NLST` command. The default `ls` behavior is the same as in the previous Solaris release.

For details, see ftp(4).

inetinit

This facility enables you to configure TCP sequence numbers and to enable or disable support for 6to4 relay routers.

init

For details, see the /etc/default/init information in the FILES section of init(1M).

All values in the file are placed in the environment of the shell that init invokes in response to a single user boot request. The init process also passes these values to any commands that it starts or restarts from the /etc/inittab file.

keyserv

For details, see the /etc/default/keyserv information in the FILES section of keyserv(1M).

kbd

For details, see the Extended Description section of kbd(1).

login

For details, see the /etc/default/login information in the FILES section of login(1).

mpathd

This facility enables you to set in.mpathd configuration parameters.

For details, see in.mpathd(1M).

nfs

This facility enables you to set NFS daemon configuration parameters.

For details, see nfs(4).

nfslogd

For details, see the Description section of nfslogd(1M).

nss

This facility enables you to configure initgroups(3C) lookup parameters.

For details, see nss(4).

passwd

For details, see the /etc/default/passwd information in the FILES section of passwd(1).

power

For details, see the /etc/default/power information in the FILES section of pmconfig(1M).

rpc.nisd

For details, see the /etc/default/rpc.nisd information in the FILES section of rpc.nisd(1M).

su

For details, see the /etc/default/su information in the FILES section of su(1M).

syslog

For details, see the /etc/default/syslogd information in the FILES section of syslogd(1M).

sys-suspend

For details, see the /etc/default/sys-suspend information in the FILES section of sys-suspend(1M).

tar

For a description of the -f function modifier, see tar(1).

If the TAPE environment variable is not present and the value of one of the arguments is a number and -f is not specified, the number matching the archive*N* string is looked up in the /etc/default/tar file. The value of the archive*N* string is used as the output device with the blocking and size specifications from the file.

For example:

```
% tar -c 2 /tmp/*
```

This command writes the output to the device specified as archive2 in the /etc/default/tar file.

utmpd

The utmpd daemon monitors /var/adm/utmpx (and /var/adm/utmp in earlier Solaris versions) to ensure that utmp entries inserted by non-root processes by pututxline(3C) are cleaned up on process termination.

Two entries in /etc/default/utmpd are supported:

- SCAN_PERIOD – The number of seconds that utmpd sleeps between checks of /proc to see if monitored processes are still alive. The default is 300.
- MAX_FDS – The maximum number of processes that utmpd attempts to monitor. The default value is 4096 and should never need to be changed.

yppasswdd

This facility enables you to configure whether a user can successfully set a login shell to a restricted shell when using the passwd -r nis -e command.

For details, see rpc.yppasswdd(1M).

APPENDIX **A**

Tunable Parameters Change History

This chapter describes the change history of specific tunable parameters. If a parameter is in this section, it has changed from a previous release. Parameters whose functionality has been removed are listed also.

Kernel Parameters

Process-Sizing Tunables

`max_nprocs` (Solaris 9 Releases)

The Solaris 10 description section was updated by removing the text "sun4m."

General I/O

`maxphys` (Solaris 9 Releases)

The text "126,976 (sun4m)" was removed from the Solaris 10 `maxphys` default section.

`rlim_fd_max` (Solaris 8 Release)

In the Solaris 8 version, the default is 1024. In later Solaris releases, the default is 65,536.

General Kernel and Memory Parameters

`noexec_user_stack` (Solaris 9 Releases)

The Solaris 10 description section was updated by removing the text "and sun4m" and adding the text "64–bit SPARC and AMD64."

`lwp_default_stksize` (Solaris 9 Releases)

The Solaris 10 description section was updated by adding default and maximum values for AMD64.

`noexec_user_stack` (Solaris 2.6, 7, and 8 Releases)

The Solaris 9 description section was updated by removing the text "and sun4d."

`segkpsize` (Solaris 9 12/02 Release)

In previous Solaris 9 releases, units were incorrectly identified as Mbytes instead of 8-Kbyte pages. In addition, the following text is removed from the range and default descriptions in the Solaris 10 release because this parameter is only available on systems running 64-bit kernels:

32-bit kernels, 512 Mbytes

Paging-Related Tunables

`tmpfs:tmpfs_minfree` (Solaris 8 Releases)

In the Solaris 8 versions, the units was incorrectly described as "Bytes," instead of "Pages."

`pages_pp_maximum` (Solaris Releases Prior to Solaris 9 Releases)

In the Solaris 8 versions, the default description is as follows:

Maximum of the triplet (200, `tune_t_minarmem` + 100, [10% of memory available at boot time])

General File System

`ncsize` (Solaris 9 and Previous Releases)

In the Solaris 9 release and previous releases, the `nfs:nrnode` parameter was incorrectly identified as `nfs:nfs_rnode` in the when to change description.

UFS

`bufhwm` (Solaris 9 Releases)

This parameter information changed significantly in the Solaris 10 release. Please see "`bufhwm` and `bufhwm_pct`" on page 66 for more information.

Description

Maximum amount of memory for caching I/O buffers. The buffers are used for writing file system metadata (superblocks, inodes, indirect blocks, and directories). Buffers are allocated as needed until the amount to be allocated would exceed `bufhwm`. At this point, enough buffers are reclaimed to satisfy the request.

For historical reasons, this parameter does not require the `ufs:` prefix.

Data Type

Signed integer

Default

2% of physical memory

Range

80 Kbytes to 20% of physical memory

Units

Kbytes

Dynamic?

No. Value is used to compute hash bucket sizes and is then stored into a data structure that adjusts the value in the field as buffers are allocated and deallocated. Attempting to adjust this value without following the locking protocol on a running system can lead to incorrect operation.

Validation

If `bufhwm` is less than 80 Kbytes or greater than the lesser of 20% of physical memory or twice the current amount of kernel heap, it is reset to the lesser of 20% of physical memory or twice the current amount of kernel heap. The following message appears on the system console and in the `/var/adm/messages` file.

`"binit: bufhwm out of range (value attempted). Using` *N*`."`

Value attempted refers to the value entered in `/etc/system` or by using the `kadb -d` command. *N* is the value computed by the system based on available system memory.

When to Change

Since buffers are only allocated as they are needed, the overhead from the default setting is the allocation of a number of control structures to handle the maximum possible number of buffers. These structures consume 52 bytes per potential buffer on a 32–bit kernel and 104 bytes per potential buffer on a 64–bit kernel. On a 512 Mbyte 64–bit kernel this consumes 104*10144 bytes, or 1 Mbyte. The header allocations assumes buffers are 1 Kbyte in size, although in most cases, the buffer size is larger.

The amount of memory, which has not been allocated in the buffer pool, can be found by looking at the `bfreelist` structure in the kernel with a kernel debugger. The field of interest in the structure is `bufsize`, which is the possible remaining memory in bytes. Looking at it with the `buf` macro by using `mdb`:

```
# mdb -kLoading modules: [ unix krtld genunix ip nfs ipc ]
> bfreelist$<buf
bfreelist:
[ elided ]
bfreelist + 0x78:    bufsize            [ elided ]
                           75734016
```

`bufhwm` on this system, with 6 Gbytes of memory, is 122277. It is not directly possible to determine the number of header structures used since the actual buffer size requested is usually larger than 1 Kbyte. However, some space might be profitably reclaimed from control structure allocation for this system.

The same structure on the 512 Mbyte system shows that only 4 Kbytes of 10144 Kbytes has not been allocated. When the `biostats` `kstat` is examined with `kstat -n biostats`, it is seen that the system had a reasonable ratio of `buffer_cache_hits` to `buffer_cache_lookups` as well. This indicates that the default setting is reasonable for that system.

Commitment Level

Unstable

NFS Parameters

nfs:nrnode (Solaris 9 8/03)

The Solaris 10 description was updated to include the text “NFS version 4 client.”

nfs:nfs_write_error_interval (Solaris 9 8/03)

The Solaris 10 description was updated to include the text “NFS version 4 client.”

nfs:nfs_write_error_to_cons_only (Solaris 9 8/03)

The Solaris 10 description was updated to include the text “NFS version 4 client.”

nfs:nfs_disable_rddir_cache (Solaris 9 8/03)

The Solaris 10 when to change text was updated to include the text “NFS version 4 client.”

nfs:nfs3_max_transfer_size (Solaris 9 8/03)

The Solaris 10 default description was updated to 1,048,576 (1 Mbyte) from 32, 768 (32 Kbytes).

TCP/IP Tunable Parameters

ip_multidata_outbound (Solaris 9 8/03)

This parameter information changed significantly in releases after the Solaris 9 8/03 release. Please see "ip_multidata_outbound" on page 137 for more information.

Description

This parameter enables the network stack to send more than one packet at one time to the network device driver during transmission.

Enabling this parameter reduces the per-packet processing costs by improving the host CPU utilization and/or network throughput.

The multidata transmit (MDT) feature is only effective for device drivers that support this feature.

The following parameter must be enabled in the /etc/system file to use the MDT parameter:

```
set ip:ip_use_dl_cap = 0x1
```

Default

Disabled

Range

0 (disabled), 1 (enabled)

Dynamic?

Yes

When to Change

This feature can be enabled at any time to allow for improved system performance with the following cautions:

- Enabling this feature might change the appearance of any packets between the IP layer and the DLPI provider. So, any third-party STREAMS module that is dynamically inserted between the IP layer and the DLPI provider by using ifconfig's modinsert feature, which doesn't understand the MDT STREAMS data type, might not work.

  Modules that are inserted between the IP and the DLPI provider with the autopush(1m) mechanism might not work as well.

- Keep this feature disabled when a STREAMS module is not MDT aware. For example, the public domain utilities such as ipfilter, Checkpoint Firewall-1, and so on, are not MDT aware.

Commitment Level Unstable

`tcp_conn_hash_size` (Solaris 9 Releases)

This parameter was removed in the Solaris 10 release.

Description Controls the hash table size in the TCP module for all TCP connections.

Data Type Signed integer

Default 512

Range 512 to 1,073,741,824

Implicit The value should be a power of 2.

Dynamic? No. The parameter can only be changed at boot time.

Validation If you set the parameter to a value that is not a power of 2, it is rounded up to the nearest power of 2.

When to Change If the system consistently has tens of thousands of TCP connections, increase the value accordingly. With the default value, TCP performs well up to a few thousand active connections. Note that increasing the hash table size means more memory consumption so set an appropriate value to avoid wasting memory unnecessarily.

Commitment Level Unstable

`ipc_tcp_conn_hash_size` (Solaris 9 Releases)

This parameter was removed in the Solaris 10 release.

Description Controls the hash table size in an IP module for all active (in ESTABLISHED state) TCP connections.

Data Type Unsigned integer

Default 512

Range 512 to 2,147,483,648

Implicit It should be a power of two.

Dynamic?	No. This parameter can only be changed at boot time.
Validation	If you set the parameter to a value that is not a power of 2, it is rounded up to the nearest power of two.
When to Change	If the system consistently has tens of thousands of active TCP connections, increase the value accordingly. With the default value, the system performs well up to a few thousand active connections. Note that increasing the hash table size means more memory consumption so set an appropriate value to avoid wasting memory unnecessarily.
Commitment Level	Unstable

`tcp_compression_enabled` (Solaris 9 Releases)

This parameter was removed in the Solaris 10 release.

Description	If set to 1, protocol control blocks of TCP connections in TIME-WAIT state are compressed to reduce memory usage. If set to 0, no compression is done. See "`tcp_time_wait_interval`" on page 146 also.
Default	1 (enabled)
Range	0 (disabled), 1 (enabled)
Dynamic?	Yes
When to Change	Do not turn off the compression mechanism.
Commitment Level	Unstable

`ip_forwarding` and `ip6_forwarding` (Solaris 9 Releases)

These parameters are obsolete in the Solaris 10 release.

Description	Controls whether IP does IPv4 or IPv6 forwarding between interfaces. See also "*xxx*`:ip_forwarding` (Solaris 9 Releases)" on page 187.
Default	0 (disabled)
Range	0 (disabled) or 1 (enabled)
Dynamic?	Yes
When to Change	If IP forwarding is needed, enable it.

Commitment Level Unstable

xxx:ip_forwarding (Solaris 9 Releases)

This parameter is obsolete in the Solaris 10 release.

Description Enables IPv4 forwarding for a particular *xxx* interface. The exact name of the parameter is *interface-name*:ip_forwarding. For example, two interfaces are hme0 and hme1. Here are their corresponding parameter names:

hme0:ip_forwarding and hme1:ip_forwarding

Default 0 (disabled)

Range 0 (disabled) or 1 (enabled)

Dynamic? Yes

When to Change If you need IPv4 forwarding, use this parameter to enable forwarding on a per-interface basis.

Commitment Level Unstable

tcp_conn_req_max_q0 (Solaris 8 Release)

The when to change text was revised in later Solaris releases from this:

When to Change For applications, such as web servers that might receive excessive connection requests, you can increase the default value to match the incoming rate.

The following explains the relationship between tcp_conn_req_max_q0 and the maximum number of pending connections for each socket.

When a connection request is received, TCP first checks if the number (*N*) of pending TCP connections (three-way handshake is done) waiting to be accepted exceeds the maximum for the listener. If the connections are excessive, the request is denied. If the number of connections is allowable, then TCP checks if the number of incomplete pending TCP connections exceeds the sum of *N* and tcp_conn_req_max_q0. If it does not, the request is accepted. Otherwise, the oldest incomplete pending TCP request is dropped.

to this:

When to Change

For applications, such as web servers that might receive excessive connection requests, you can increase the default value to match the incoming rate.

The following explains the relationship between tcp_conn_req_max_q0 and the maximum number of pending connections for each socket.

When a connection request is received, TCP first checks if the number of pending TCP connections (three-way handshake is done) waiting to be accepted exceeds the maximum (*N*) for the listener. If the connections are excessive, the request is denied. If the number of connections is allowable, then TCP checks if the number of incomplete pending TCP connections exceeds the sum of N and tcp_conn_req_max_q0. If it does not, the request is accepted. Otherwise, the oldest incomplete pending TCP request is dropped.

Network Cache and Accelerator (NCA) Tunable Parameters

sq_max_size (Solaris 9 12/02 Release)

This parameter information changed significantly in later Solaris releases. Please see "sq_max_size" on page 171 for more information.

Description

The depth of the syncq (number of messages) before a destination streams queue generates a QFULL message.

Default

2 messages

Range

1 to 0 (unlimited)

Dynamic?

No

When to Change

When NCA is running on a system with a lot of memory, increase this parameter to allow drivers to queue more packets of data. If a server is under heavy load, increase this parameter so modules and drivers may process more data without dropping packets or getting backlogged.

Commitment Level Unstable

Parameters That Are Obsolete or Have Been Removed

The following section describes parameters that are obsolete or have been removed from more recent Solaris releases.

Paging-Related Tunables

cachefree (Solaris 8 Releases)

Obsolete in Solaris 9 and later releases.

Description The Solaris 8 release changes the way file system pages are cached. These changes subsume the priority paging capability.

Note – Remove both cachefree and priority_paging settings in the /etc/system file.

The caching changes remove most of the pressure on the virtual memory system resulting from file system activity. Several statistics exhibit new behavior:

- Page reclaims are higher because pages are now explicitly added to the free list after I/O completes.
- Free memory is now higher because the free memory count now includes a large component of the file cache.
- Scan rates are drastically reduced.

Commitment Level Obsolete

priority_paging (Solaris 8 Releases)

Obsolete in Solaris 9 and later releases.

Description	This variable sets cachefree to 2 times lotsfree. The Solaris 8 release changes the way file system pages are cached. These changes subsume the priority paging capability. **Note –** Remove both cachefree and priority_paging settings in the /etc/system file.
Commitment Level	Obsolete

tune_t_gpgslo (Solaris 7 Releases)

Description	Obsolete in the Solaris 8 release. Variable left in place for compatibility reasons.

tune_t_minasmem (Solaris 7 Releases)

Description	Obsolete in the Solaris 8 release. Variable left in place for compatibility reasons.

System V Message Parameters

msgsys:msginfo_msgmni (Solaris 9 Releases)

Obsolete in the Solaris 10 release.

Description	Maximum number of message queues that can be created.
Data Type	Signed integer
Default	50
Range	0 to MAXINT
Dynamic?	No. Loaded into msgmni field of msginfo structure.
Validation	None
When to Change	When msgget(2) calls return with an error of ENOSPC or at the recommendation of a software vendor.
Commitment Level	Unstable

msgsys:msginfo_msgtql (Solaris 9 Releases)

Obsolete in the Solaris 10 release.

Description	Maximum number of messages that can be created. If a msgsnd call attempts to exceed this limit, the request is deferred until a message header is available. Or, if the request has set the IPC_NOWAIT flag, the request fails with the error EAGAIN.
Data Type	Signed integer
Default	40
Range	0 to MAXINT
Dynamic?	No. Loaded into msgtql field of msginfo structure.
Validation	None
When to Change	When msgsnd() calls block or return with error of EGAIN, or at the recommendation of a software vendor.
Commitment Level	Unstable

msgsys:msginfo_msgmnb (Solaris 9 Releases)

Obsolete in the Solaris 10 release.

Description	Maximum number of bytes that can be on any one message queue.
Data Type	Unsigned long
Default	4096
Range	0 to amount of physical memory
Units	Bytes
Dynamic?	No. Loaded into msgmnb field of msginfo structure.
Validation	None
When to Change	When msgsnd() calls block or return with an error of EGAIN, or at the recommendation of a software vendor.
Commitment Level	Unstable

msgsys:msginfo_msgssz (Solaris 9 Releases)

Removed in the Solaris 10 release.

Description	Specifies size of chunks system uses to manage space for message buffers.
Data Type	Signed integer
Default	40
Range	0 to MAXINT
Dynamic?	No. Loaded into `msgtql` field of `msginfo`structure.
Validation	The space consumed by the maximum number of data structures that would be created to support the messages and queues is compared to 25% of the available kernel memory at the time the module is loaded. If the number is too big, the message queue module refuses to load and the facility is unavailable. This computation does include the space that might be consumed by the messages. This situation occurs only when the module is first loaded.
When to Change	When the default value is not enough. Generally changed at the recommendation of software vendors.
Commitment Level	Obsolete

`msgsys:msginfo_msgmap` (Solaris 9 Releases)

Removed in the Solaris 10 release.

Description	Number of messages the system supports.
Data Type	Signed integer
Default	100
Range	0 to MAXINT
Dynamic?	No
Validation	The space consumed by the maximum number of data structures that would be created to support the messages and queues is compared to 25% of the available kernel memory at the time the module is loaded. If the number is too big, the message queue module refuses to load and the facility is unavailable. This computation does include the space that might be consumed by the messages. This situation occurs only when the module is first loaded.
When to Change	When the default value is not enough. Generally changed at the recommendation of software vendors.
Commitment Level	Obsolete

`msgsys:msginfo_msgseg` (Solaris 9 Releases)

Removed in the Solaris 10 release.

Description	Number of `msginfo_msgssz` segments the system uses as a pool for available message memory. Total memory available for messages is `msginfo_msgseg * msginfo_msgssz`.
Data Type	Signed short
Default	1024
Range	0 to 32,767
Dynamic?	No
Validation	The space consumed by the maximum number of data structures that would be created to support the messages and queues is compared to 25% of the available kernel memory at the time the module is loaded. If the number is too big, the message queue module refuses to load and the facility is unavailable. This computation does not include the space that might be consumed by the messages. This situation occurs only when the module is first loaded.
When to Change	When the default value is not enough. Generally changed at the recommendation of software vendors.
Commitment Level	Obsolete

`msgsys:msginfo_msgmax` (Solaris 9 Releases)

Removed in the Solaris 10 release.

Description	Maximum size of System V message.
Data Type	Unsigned long
Default	2048
Range	0 to amount of physical memory
Units	Bytes
Dynamic?	No. Loaded into `msgmax` field of `msginfo` structure.
Validation	None
When to Change	When `msgsnd`(2) calls return with error of `EINVAL` or at the recommendation of a software vendor.
Commitment Level	Unstable

System V Semaphore Parameters

semsys:seminfo_semmns (Solaris 9 Releases)

Removed in the Solaris 10 release.

Description	Maximum number of System V semaphores on the system.
Data Type	Signed integer
Default	60
Range	1 to MAXINT
Dynamic?	No
Validation	The amount of space that could possibly be consumed by the semaphores and their supporting data structures is compared to 25% of the kernel memory available at the time the module is first loaded. If the memory threshold is exceeded, the module refuses to load and the semaphore facility is not available.
When to Change	When the default number of semaphores is not enough. Generally changed at the recommendation of software vendors. No error messages are displayed when an attempt is made to create more semaphores than are currently configured. The application sees a return code of `ENOSPC` from a `semget`(2) call.
Commitment Level	Unstable

semsys:seminfo_semmnu (Solaris 9 Releases)

Removed in the Solaris 10 release.

Description	Total number of undo structures supported by the System V semaphore system.
Data Type	Signed integer
Default	30
Range	1 to MAXINT
Dynamic?	No

Validation — The amount of space that could possibly be consumed by the semaphores and their supporting data structures is compared to 25% of the kernel memory available at the time the module is first loaded. If the memory threshold is exceeded, the module refuses to load and the semaphore facility is not available.

When to Change — When the default value is not enough. Generally changed at the recommendation of software vendors. No error message is displayed when an attempt is made to perform more undo operations than are currently configured. The application sees a return value of `ENOSPC` from a `semop`(2) call when the system runs out of undo structures.

Commitment Level — Unstable

`semsys:seminfo_semume` (Solaris 9 Releases)

Description — Removed in the Solaris 10 release.

Maximum number of System V semaphore undo structures that can be used by any one process.

Data Type — Signed integer

Default — 10

Range — 1 to MAXINT

Dynamic? — No

Validation — The amount of space that could possibly be consumed by the semaphores and their supporting data structures is compared to 25% of the kernel memory available at the time the module is first loaded. If the memory threshold is exceeded, the module refuses to load and the semaphore facility is not available.

When to Change — When the default value is not enough. Generally changed at the recommendation of software vendors. No error messages are displayed when an attempt is made to perform more undo operations than are currently configured. The application sees a return code of `EINVAL` from a `semop`(2) call.

Commitment Level — Unstable

`semsys:seminfo_semvmx` (Solaris 9 Releases)

Removed in the Solaris 10 release.

Description	Maximum value a semaphore can be set to.
Data Type	Unsigned short
Default	32,767
Range	1 to 65,535
Dynamic?	No
Validation	None
When to Change	When the default value is not enough. Generally changed at the recommendation of software vendors. No error messages are displayed when the maximum value is exceeded. The application sees a return code of `ERANGE` from a `semop`(2) call.
Commitment Level	Unstable

`semsys:seminfo_semaem` (Solaris 9 Releases)

Removed in the Solaris 10 release.

Description	Maximum value that a semaphore's value in an undo structure can be set to.
Data Type	Unsigned short
Default	16,384
Range	1 to 65,535
Dynamic?	No
Validation	None
When to Change	When the default value is not enough. Generally changed at the recommendation of software vendors. No error messages are displayed when an attempt is made to perform more undo operations than are currently configured. The application sees a return code of `EINVAL` from a `semop`(2) call.
Commitment Level	Unstable

`semsys:seminfo_semmap` (Solaris 7 Releases)

Obsolete in the Solaris 8 release. Variable is present in kernel for compatibility reasons but is no longer used.

`semsys:seminfo_semusz` (Solaris 7 Releases)

Obsolete in the Solaris 8 release. Any values entered are ignored.

System V Shared Memory

`shmsys:shminfo_shmmni` (Solaris 9 Releases)

Obsolete in the Solaris 10 release.

Description	System wide limit on number of shared memory segments that can be created.
Data Type	Signed integer
Default	100
Range	0 to MAXINT
Dynamic?	No. Loaded into `shmmni` field of `shminfo` structure.
Validation	The amount of space consumed by the maximum possible number of data structures to support System V shared memory is checked against 25% of the currently available kernel memory at the time the module is loaded. If the memory consumed is too large, the attempt to load the module fails.
When to Change	When the system limits are too low. Generally changed on the recommendation of software vendors.
Commitment Level	Unstable

`shmsys:shminfo_shmmax` (Solaris 9 Releases)

Obsolete in the Solaris 10 release.

Description	Maximum size of system V shared memory segment that can be created. This parameter is an upper limit that is checked before the application sees if it actually has the physical resources to create the requested memory segment. Attempts to create a shared memory section whose size is zero or whose size is larger than the specified value will fail with an `EINVAL` error.

This parameter specifies only the largest value the operating system can accept for the size of a shared memory segment. Whether the segment can be created depends entirely on the amount of swap space available on the system and, for a 32-bit process, whether there is enough space available in the process's address space for the segment to be attached.

Data Type	Unsigned long
Default	8,388,608
Range	0 - MAXUINT32 on 32-bit systems, 0 – MAXUINT64 on 64-bit systems
Units	Bytes
Dynamic?	No. Loaded into `shmmax` field of `shminfo` structure.
Validation	None
When to Change	When the default value is too low. Generally changed at the recommendation of software vendors, but unless the size of a shared memory segment needs to be constrained, setting this parameter to the maximum possible value has no side effects.
Commitment Level	Unstable

`shmsys:shminfo_shmmin` (Solaris 8 Release)

Obsolete in the Solaris 9 release. Variable is present in kernel for compatibility reasons but is no longer used.

`shmsys:shminfo_shmseg` (Solaris 8 Release)

Obsolete in the Solaris 9 release. Variable is present in kernel for compatibility reasons but is no longer used.

NFS Module Parameters

`nfs:nfs_32_time_ok` (Solaris 7)

Obsolete in the Solaris 8 release.

APPENDIX B

Revision History for This Manual

This section describes the revision history for this manual.

- "Current Version: Solaris 10 Release" on page 199
- "New or Changed Parameters in the Solaris 10 Release" on page 199
- "New or Changed Parameters in the Solaris 9 Releases" on page 203
- "New Parameters in the Solaris 8 Release" on page 205

Current Version: Solaris 10 Release

The current version of this manual applies to the Solaris 10 release.

New or Changed Parameters in the Solaris 10 Release

Solaris Kernel Tunable Parameters

The following sections describe new, changed, or obsolete kernel tunables.

General Kernel and Memory Parameters

The parameter, "`default_stksize`" on page 31, is new in the Solaris 10 release.

The “lwp_default_stksize” on page 32 and “noexec_user_stack” on page 34 parameters are changed in this release.

UFS

The following parameters are modified in the Solaris 10 release:

- “bufhwm and bufhwm_pct” on page 66
- “ncsize” on page 62

General File System

The following parameters are newly documented in the Solaris 10 release:

- “freebehind” on page 72
- “segmap_percent” on page 66
- “smallfile” on page 73

System V Message Queues

The following parameters have been removed in the Solaris 10 release:

- msgsys:msginfo_msgmap
- msgsys:msginfo_msgmax
- msgsys:msginfo_msgseg
- msgsys:msginfo_msgssz

The following parameters are obsolete as of the Solaris 10 release:

- msgsys:msginfo_msgmnb
- msgsys:msginfo_msgmni
- msgsys:msginfo_msgtql

System V Semaphores

The following parameters have been removed in the Solaris 10 release:

- semsys:seminfo_semmaem
- semsys:seminfo_semmap
- semsys:seminfo_semmns
- semsys:seminfo_semmnu
- semsys:seminfo_semvmx
- semsys:seminfo_semume
- semsys:seminfo_semusz

System V Shared Memory

The following parameters have been removed from the Solaris 10 release:

- shmsys:shminfo_shmmin
- shmsys:shminfo_shmseg

The following parameters are obsolete.

- shmsys:shminfo_shmmni
- shmsys:shminfo_shmmax

TSB Parameters

The following TSB parameters are new in the Solaris 10 release:

- "tsb_alloc_hiwater_factor" on page 86
- "default_tsb_size" on page 87
- "enable_tsb_rss_sizing" on page 87
- "tsb_rss_factor" on page 88

NFS Parameters

The stability level of all NFS parameters was changed from "Evolving" to "Unstable."

The following NFSv4 parameters are new in the Solaris 10 release:

- "nfs:nfs4_async_clusters" on page 117
- "nfs:nfs4_bsize" on page 115
- "nfs:nfs4_cots_timeo" on page 97
- "nfs:nfs4_do_symlink_cache" on page 99
- "nfs:nfs4_dynamic" on page 101
- "nfs:nfs4_lookup_neg_cache" on page 103
- "nfs:nfs4_max_threads" on page 105
- "nfs:nfs4_max_transfer_size" on page 122
- "nfs:nfs4_nra" on page 107
- "nfs:nfs4_pathconf_disable_cache" on page 94
- "nfs:nfs4_shrinkreaddir" on page 111

The following NFS parameters are new or changed in the Solaris 10 release:

- "nfs:nfs_nra" on page 106
- "nfs:nfs3_nra" on page 107
- "nfs:nfs3_shrinkreaddir" on page 110

The following NFS parameters were previously provided in error and have been removed:

- nfsserv:nfs_shrinkreaddir

- nfsserv:nfs3_shrinkreaddir

TCP/IP Parameters

The following IP parameters are new in the Solaris 10 release:

- "ip_squeue_worker_wait" on page 138
- "ip_squeue_write" on page 138
- "ip_squeue_fanout" on page 139
- "ipcl_conn_hash_size" on page 151

The following TCP parameters are new in this Solaris release:

- "tcp_rst_sent_rate_enabled" on page 150
- "tcp_rst_sent_rate" on page 150
- "tcp_mdt_max_pbufs" on page 151

The following TCP/IP parameters are obsolete in the Solaris 10 release:

- tcp_conn_hash_size
- ipc_tcp_conn_hash_size
- tcp_compression_enabled
- ip_forwarding
- ip6_forwarding
- xxx_forwarding

SCTP Tunable Parameters

The following SCTP parameters are new in the Solaris 10 release:

- "sctp_max_init_retr" on page 157
- "sctp_pa_max_retr" on page 157
- "sctp_pp_max_retr" on page 158
- "sctp_cwnd_max" on page 158
- "sctp_ipv4_ttl" on page 159
- "sctp_heartbeat_interval" on page 159
- "sctp_new_secret_interval" on page 159
- "sctp_initial_mtu" on page 160
- "sctp_deferred_ack_interval" on page 160
- "sctp_ignore_path_mtu" on page 160
- "sctp_initial_ssthresh" on page 161
- "sctp_xmit_hiwat" on page 161
- "sctp_xmit_lowat" on page 161
- "sctp_recv_hiwat" on page 162
- "sctp_max_buf" on page 162
- "sctp_ipv6_hoplimit" on page 162
- "sctp_rto_min" on page 163

- "sctp_rto_max" on page 163
- "sctp_rto_initial" on page 163
- "sctp_cookie_life" on page 163
- "sctp_max_in_streams" on page 164
- "sctp_initial_out_streams" on page 164
- "sctp_shutack_wait_bound" on page 164
- "sctp_maxburst" on page 165
- "sctp_addip_enabled" on page 165
- "sctp_prsctp_enabled" on page 165

System Facility Parameters

The following system facilities are new in the Solaris 10 release:

- "autofs" on page 174
- "ftp" on page 174
- "nfs" on page 175

The inetd system facility is obsolete in the Solaris 10 release.

Removal of sun4m Support

The sun4m platform is not supported in the Solaris 10 release. The following parameters were modified to reflect the removal of sun4m support:

- max_nprocs
- maxphys
- noexec_user_stack

New or Changed Parameters in the Solaris 9 Releases

The following sections describe new or changed parameters in the Solaris 9 releases.

ip_policy_mask

This parameter is new in the Solaris 9 12/02 release. For information, see "ip_policy_mask" on page 156.

Removal of sun4d Support

The sun4d platform is not supported in the Solaris 9 release. The following parameters were modified to reflect the removal of sun4d support:

- max_nprocs
- maxphys
- noexec_user_stack

Unsupported or Obsolete Parameters

priority_paging and cachefree are Not Supported

The priority_paging and cachefree tunable parameters are not supported in the Solaris 9 release. They have been replaced with an enhanced file system caching architecture that implements paging policies similar to priority paging, but are always enabled. Attempts to set these parameters in the /etc/system file result in boot-time warnings such as:

```
sorry, variable 'priority_paging' is not defined in the 'kernel'
sorry, variable 'cachefree' is not defined in the 'kernel'
```

The SUNWcsr packages that contain the /etc/system file have been modified so that the inclusion of the priority_paging or cachefree tunable parameters are prohibited. If you upgrade to the Solaris 9 release or add the SUNWcsr packages and your /etc/system file includes the priority_paging or cachefree parameters, the following occurs:

1. This message is displayed if the priority_paging or cachefree parameters are set in the /etc/system file:

   ```
   /etc/system has been modified since it contains references to priority
   paging tunables. Please review the changed file.
   ```

2. Comments are inserted in the /etc/system file before any line that sets priority_paging or cachefree. For example, if priority_paging is set to 1, the following lines are inserted before the line with the priority_paging value:

```
*NOTE: As of Solaris 9, priority paging is unnecessary and has been removed.
* Since references to priority paging-related tunables will now result in
* boot-time warnings, the assignment below has been commented out. For more
* details,see the Solaris 9 Release Notes,or the "Solaris Tunable Parameters
* Reference Manual".
```

System V Shared Memory

The following parameters are obsolete:

- shmsys:shminfo_shmmin
- shmsys:shminfo_shmseg

New Parameters in the Solaris 8 Release

logevent_max_q_sz

This parameter is new in the Solaris 8 1/01 release. For information, see "logevent_max_q_sz" on page 33.

Index