Contents

Preface 13

1. Overview of Solaris System Tuning 19
   Tuning a Solaris System 19
     Tuning Format 20
   Tuning the Solaris Kernel 22
     /etc/system File 22
     adb 24
     kadb 24
     mdb 25
   Special Structures 25
   Viewing System Configuration Information 26
     sysdef 26
     kstats 27

2. Solaris Kernel Tunables 29
   General Parameters 30
     physmem 30
     lwp_default_stksize 31
   fsflush and Related Tunables 32
     fsflush 32
tune_t_fsflushr 33
autoup 33
dopageflush 35
doiflush 35

Process Sizing Tunables 36
maxusers 36
reserved_procs 38
pidmax 39
max_nprocs 39
maxuprc 41

Paging-Related Tunables 41
lotsfree 42
desfree 44
minfree 45
throttlefree 46
pageout_reserve 47
cachefree 49
priority_paging 49
pages_pp_maximum 50
tune_t_minarmem 51
fastscan 51
slowscan 52
min_percent_cpu 53
handspreadpages 53
pages_before_pager 54
maxpgio 55

Swapping-Related Variables 56
swapfs_reserve 56
swapfs_minfree 57
General Kernel Variables 58
noexec_user_stack 58
Kernel Memory Allocator 59
kmem_flags 59
General Driver 62
moddebug 62
General I/O 63
maxphys 63
rlim_fd_max 64
rlim_fd_cur 65
General File System 66
ncsize 66
rstchown 67
segkpsize 68
dnlc_dir_enable 69
dnlc_dir_min_size 70
dnlc_dir_max_size 71
UFS 71
bufhwm 71
ndquot 73
ufs_ninode 74
ufs:ufs_WRITES 76
ufs:ufs_LW and ufs:ufs_HW 77
TMPFS 78
tmpfs:tmpfs_maxkmem 78
tmpfs:tmpfs_minfree 79
Pseudo Terminals 79
pt_cnt 80
pt_pctofmem 81
pt_max_pty 82

Streams 82
nstrpush 82
strmsgsz 83
strctlisz 83

System V Message Queues 84
msgsys:msginfo_msgmax 85
msgsys:msginfo_msgmnb 85
msgsys:msginfo_msgmni 86
msgsys:msginfo_msgtql 86

System V Semaphores 87
semsys:seminfo_semmni 88
semsys:seminfo_semmns 88
semsys:seminfo_semmx 89
semsys:seminfo_semmx 89
semsys:seminfo_semmx 89
semsys:seminfo_semmx 89
semsys:seminfo_semopm 90
semsys:seminfo_semmnu 91
semsys:seminfo_semmme 92
semsys:seminfo_semaem 92

System V Shared Memory 93
shmsys:shminfo_shmmx 93
shmsys:shminfo_shmmin 94
shmsys:shminfo_shmmin 95
shmsys:shminfo_shmseg 95
segspt_minfree 96

Scheduling 97
rechoose_interval 97

Timers 98

    hires_tick 98

    timer_max 98

Sun4u Specific 99

    consistent_coloring 99

3. **NFS Tunable Parameters** 101

Tuning the NFS Environment 101

NFS Module Parameters 101

    nfs:nfs3_pathconf_disable_cache 102

    nfs:nfs_allow_preepoch_time 102

    nfs:nfs_cots_timeo 103

    nfs:nfs3_cots_timeo 104

    nfs:nfs_do_symlink_cache 105

    nfs:nfs3_do_symlink_cache 106

    nfs:nfs_dynamic 106

    nfs:nfs3_dynamic 107

    nfs:nfs_lookup_neg_cache 108

    nfs:nfs3_lookup_neg_cache 109

    nfs:nfs_max_threads 110

    nfs:nfs3_max_threads 111

    nfs:nfs_nra 112

    nfs:nfs3_nra 113

    nfs:nrnodenode 113

    nfs:nfs_shrinkreaddir 115

    nfs:nfs_write_error_interval 116

    nfs:nfs_write_error_to_cons_only 116

    nfs:nfs_disable_rddir_cache 117
nfssrv Module Parameters 125
nfssrv:nfs_portmon 125
nfssrv:rfs_write_async 126
nfssrv:nfsauth_ch_cache_max 127
nfssrv:exi_cache_time 128
rpcmod Module Parameters 129
rpcmod:clnt_max_conns 129
rpcmod:clnt_idle_timeout 130
rpcmod:svc_idle_timeout 130
rpcmod:svc_default_stksize 131
rpcmod:svc_default_max_same_xprt 132
rpcmod:maxdupreqs 132
rpcmod:cotsmaxdupreqs 134

4. TCP/IP Tunable Parameters 137

Overview of Tuning TCP/IP Parameters 137
TCP/IP Parameter Validation 138
Internet Request for Comments (RFCs) 138
IP Tunable Parameters 138
ip_icmp_err_interval and ip_icmp_err_burst 139
ip_forwarding and ip6_forwarding 139
xxx:ip_forwarding 140
ip_respond_to_echo_broadcast and
ip6_respond_to_echo_multicast 140

ip_send_redirects and ip6_send_redirects 141

ip_forward_src_routed and ip6_forward_src_routed 141

ip_addrs_per_if 142

ip_strict_dst_multihoming and
ip6_strict_dst_multihoming 142

TCP Tunable Parameters 144

tcp_deferred_ack_interval 144
tcp_deferred_acks_max 144
tcp_wscale_always 145
tcp_tstamp_always 145
tcp_xmit_hiwat 146
tcp_recv_hiwat 146
tcp_max_buf 147
tcp_cwnd_max 147
tcp_slow_start_initial 148
tcp_slow_start_after_idle 149
tcp_sack_permitted 149
tcp_rev_src_routes 150
tcp_time_wait_interval 150
tcp_conn_req_max_q 151
tcp_conn_req_max_q0 152
tcp_conn_req_min 152

TCP Parameters Set in the /etc/system File 153

TCP Parameters With Additional Cautions 155

UDP Tunable Parameters 159

udp_xmit_hiwat 159
udp_recv_hiwat 159
5. **System Facility Parameters** 163

System Default Parameters 163

- `cron` 164
- `devfsadm` 164
- `dhcpagent` 164
- `fs` 164
- `inetinit` 164
- `init` 164
- `kbd` 165
- `login` 165
- `nfslogd` 165
- `passwd` 165
- `power` 165
- `su` 165
- `sys-suspend` 165
- `tar` 166
- `utmpd` 166

A. **Tunable Parameter Change History** 167

Kernel Parameters 167

- Process Sizing Tunables 167
- Paging Related Tunables 169
- General Kernel Variables 171
- General I/O 172
- Pseudo Terminals 174
- Sun4u Specific 174

Parameters With No Functionality 175
Preface

Solaris Tunable Parameter Reference Manual provides reference information about Solaris kernel and network tunable parameters. This manual does not provide tunable parameter information about the CDE or Java environments.

It contains information for both SPARC™ based and IA based systems.

Note - The Solaris™ operating environment is supported on two types of hardware, or platforms—SPARC and IA. The Solaris operating environment supports 64-bit and 32-bit address spaces. The information in this document pertains to both platforms and address spaces unless specified in a special chapter, section, note, bullet, figure, table, example, or code example.

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. Refer to “Tuning a Solaris System” on page 19 for guidelines on changing Solaris tunable parameters.

How This Book Is Organized

The following table describes the chapters in this book.
Related Books

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

Other Resources for Solaris Tuning Information

This table describes other resources for Solaris tuning information.

| For ... | Go To ...
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Performance tuning classes</td>
<td><a href="http://suned.sun.com">http://suned.sun.com</a></td>
</tr>
<tr>
<td>Online performance tuning information</td>
<td><a href="http://www.sun.com/sun-on-net/performance">http://www.sun.com/sun-on-net/performance</a></td>
</tr>
</tbody>
</table>

Ordering Sun Documents

Fatbrain.com, an Internet professional bookstore, stocks select product documentation from Sun Microsystems, Inc.

For a list of documents and how to order them, visit the Sun Documentation Center on Fatbrain.com at http://www1.fatbrain.com/documentation/sun.

Accessing Sun Documentation Online

The docs.sun.com\textsuperscript{SM} Web site enables you to access Sun technical documentation online. You can browse the docs.sun.com archive or search for a specific book title or subject. The URL is http://docs.sun.com.
Typographic Conventions
The following table describes the typographic changes used in this book.

**TABLE P–1  Typographic Conventions**

<table>
<thead>
<tr>
<th>Typeface or Symbol</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>AaBbCc123</td>
<td>The names of commands, files, and directories; on-screen computer output</td>
<td>Edit your .login file. Use ls –a to list all files. machine_name% you have mail.</td>
</tr>
<tr>
<td>AaBbCc123</td>
<td>What you type, contrasted with on-screen computer output</td>
<td>machine_name% su</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Password:</td>
</tr>
<tr>
<td>AaBbCc123</td>
<td>Command-line placeholder: replace with a real name or value</td>
<td>To delete a file, type rm filename.</td>
</tr>
<tr>
<td>AaBbCc123</td>
<td>Book titles, new words, or terms, or words to be emphasized.</td>
<td>Read Chapter 6 in User’s Guide. These are called class options. You must be root to do this.</td>
</tr>
</tbody>
</table>

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>
<thead>
<tr>
<th>Shell</th>
<th>Prompt</th>
</tr>
</thead>
<tbody>
<tr>
<td>C shell prompt</td>
<td><code>machine_name%</code></td>
</tr>
<tr>
<td>C shell superuser prompt</td>
<td><code>machine_name#</code></td>
</tr>
<tr>
<td>Bourne shell and Korn shell prompt</td>
<td><code>$</code></td>
</tr>
<tr>
<td>Bourne shell and Korn shell superuser prompt</td>
<td><code>#</code></td>
</tr>
</tbody>
</table>
Overview of Solaris System Tuning

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

- “Tuning a Solaris System” on page 19
- “Tuning Format” on page 20
- “Tuning the Solaris Kernel” on page 22

Tuning a Solaris System

Solaris is a multi-threaded, scalable UNIX™ operating environment running on SPARC and Intel processors. It is self-adjusting to system load and demands minimal tuning. In some cases, however, tuning is necessary. This guide provides details about the officially supported kernel tuning options available for the Solaris environment.

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 of the variables referred to in the kernel portion of this guide are in the core portion, but a few are located in loadable modules.

A key consideration in system tuning is that setting various system variables is often the least effective thing 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.

Another thing to remember is that one system’s /etc/system settings might not be applicable, either wholly or in part, to another 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 described here.

**Caution** - The variables described here and their meanings can and do change from release to release. A release is either a Solaris Update release or a new version such as Solaris 8. Publication of these variables and their description does not preclude changes to the variables and descriptions without notice.

**Tuning Format**

The format for the description of each variable follows:

- **Variable-Name**
- **Description**
- **Data Type**
- **Default**
- **Units**
- **Range**
- **Dynamic?**
- **Validation**
- **Implicit**
- **When to Change**
- **Commitment Level**
- **Change History**

**Variable-Name**

*Variable-Name* is the exact name that would be typed in the `/etc/system` file, or found in the `/etc/default/facility` file.

Most names are of the form `variable` where the variable 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 variable within the module consists of the characters to the right of the colon. For example:

```
module_name:variable
```
<table>
<thead>
<tr>
<th>Description</th>
<th>This section briefly describes what the variable does or controls.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Type</td>
<td>Signed or unsigned short or long integer with the following distinctions:</td>
</tr>
<tr>
<td></td>
<td>- On a system running a 32-bit kernel, a long is the same size as an integer.</td>
</tr>
<tr>
<td></td>
<td>- On a system running a 64-bit kernel, a long is twice the width in bits as an integer. For example, an unsigned integer = 32 bits, an unsigned long = 64 bits.</td>
</tr>
<tr>
<td>Default</td>
<td>What the system uses as the default value.</td>
</tr>
<tr>
<td>Units</td>
<td>(Optional) Description of unit type.</td>
</tr>
<tr>
<td>Range</td>
<td>Possible range allowed by system validation or the bounds of the data type.</td>
</tr>
<tr>
<td></td>
<td>- MAXINT — A shorthand description for the maximum value of a signed integer (2,147,483,647).</td>
</tr>
<tr>
<td></td>
<td>- MAXUINT — A shorthand description for the maximum value of an unsigned integer (4,294,967,295).</td>
</tr>
<tr>
<td>Dynamic?</td>
<td>Yes, if it can be changed on a running system with the adb, mdb, or kadb debuggers. No, if it is a boot time initialization only.</td>
</tr>
<tr>
<td>Validation</td>
<td>Identifies checks the system applies to the value of the variable either as entered from the /etc/system file or the default value, as well as when the validation is applied.</td>
</tr>
<tr>
<td>Implicit</td>
<td>(Optional) Unstated constraints that might exist on the variable, especially in relation to other variables.</td>
</tr>
<tr>
<td>When to Change</td>
<td>Why someone might want to change this value including error messages or return codes.</td>
</tr>
<tr>
<td>Commitment Level</td>
<td>Identifies the stability of the interface. Many of the parameters in this manual are still evolving.</td>
</tr>
</tbody>
</table>
and are classified as unstable. See attributes(5) for more information.

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

---

**Tuning the Solaris Kernel**

The table below describes the different ways tuning parameters can be applied.

<table>
<thead>
<tr>
<th>Tuning Parameters Can Be Applied in These Ways ...</th>
<th>For More Information, See ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Modifying the <code>/etc/system</code> file</td>
<td>“/etc/system File” on page 22</td>
</tr>
<tr>
<td>Using the debugger (adb)</td>
<td>“adb” on page 24</td>
</tr>
<tr>
<td>Using the kernel debugger (kadb)</td>
<td>“kadb” on page 24</td>
</tr>
<tr>
<td>Using the modular debugger (mdb)</td>
<td>“mdb” on page 25</td>
</tr>
<tr>
<td>Using the <code>ndd</code> command to set TCP/IP parameters</td>
<td>Chapter 4</td>
</tr>
<tr>
<td>Modifying the <code>/etc/default</code> files</td>
<td>“System Default Parameters” on page 163</td>
</tr>
</tbody>
</table>

**/etc/system File**

The `/etc/system` file provides a static mechanism for adjusting the values of kernel variables. Values specified in this file are read at boot time and are applied. Any changes 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 system variables were applied in two phases:

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

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

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.

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

Recovering From an Incorrect Value

Make a copy of `/etc/system` before modifying it so you can easily recover from incorrect value:

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

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

```bash
ok boot -a
```

This command causes the system to ask for the name of various files used in the boot process. Press the carriage return 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, enter the name of the good `/etc/system` file or `/dev/null`:
Name of system file [/etc/system]: /etc/system.good

If /dev/null is entered, this path causes the system to attempt to read from /dev/null for its configuration information and because it 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, Volume 1.

adb

adb is a runtime debugger. Superuser can run adb with the -k option to see variables in the running kernel. If -w is specified with the -k option, superuser can change the in-memory values of the running kernel. Any changes made in this manner are lost when the system reboots.

Example—Using adb to Change a Value

To change the value of the integer variable maxusers from its current value to 0x200, do the following:

```
$ adb -kw
physmem f7c6
maxusers/D
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 variable is to be set to.

See adb(1) for information on using the adb command.

kadb

kadb is a bootable kernel debugger with the same general syntax as adb. See kadb(1M) for the exceptions. One advantage of kadb is that the user can set breakpoints and when the breakpoint is reached, examine data or step through the execution of kernel code.

If the system is booted with kadb -d, values for variables in the core kernel can be set, but values for loadable modules would have to be set when the module was actually loaded.

See “Debugging” in Writing Device Drivers for a brief tutorial on using the kadb command.
mdb

New to the Solaris 8 release is the modular debugger, `mdb(1)`, which is unique among available Solaris debuggers because it is easily extensible. Those who have attempted to create `adb` macros are aware of the pain involved in that task. A programming API is available that allows compilation of modules to perform desired tasks within the context of the debugger. `mdb` provides backward compatibility with both `adb(1)` and `crash(1M)`.

`mdb(1)` also includes a number of desirable usability features including command-line editing, command history, built-in output pager, syntax checking, and command pipelining. This is the recommended post-mortem debugger for the kernel.

Example—Using `mdb` to Change a Value

To change the value of the integer variable `maxusers` from 5 to 6, do the following:

```
# mdb -kw
Loading modules: [ unix krtld genunix ip logindmux ptm nfs ipc lofs ]
> maxusers/D
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 variable is to be set to.

See the Solaris Modular Debugger Guide for more information on using the modular debugger.

When using `adb`, `kadb`, and `mdb`, the module name prefix is not required because 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 of the debuggers (assuming the UFS module is loaded), but would require the `ufs:` prefix when set in the `/etc/system` file. Including the module name prefix using `adb` or `kadb` results in an undefined symbol message.

Special Structures

Solaris tuning variables come in a variety of forms. The `tune` structure defined in `/usr/include/sys/tuneable.h` is the runtime representation of
tune_t_gpgslo, tune_t_fsflushr, tune_t_minarmem, tune_t_minasmem,
and tune_t_flkrec. After the kernel is initialized, all references to values of 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 variables in the tune
structure is to use the syntax, tune:field-name where field name is replaced by the
actual variable name listed above. This process silently fails. The proper way to set
variables for this structure at boot time is to initialize the special variable
corresponding to the desired field name. The system initialization process then loads
these values into the tune structure.

A second structure into which various tuning parameters are placed is the var
structure named v. You can find the definition of a var struct in
/usr/include/sys/var.h. 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 of
the fields of these structures on a running system might cause the system to panic.

---

**Viewing System Configuration Information**

Several tools are available to examine system configuration. Some require root
privilege, others can be run by a non-privileged user. Every structure and data item
can be examined with the kernel debugger (adb on a running system, booting under
kadb, or mdb).

**sysdef**

The `sysdef(1M)` 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 Ultra™ 80 system is:

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>10387456</td>
<td>maximum memory allowed in buffer cache (bufhwm)</td>
</tr>
<tr>
<td>7930</td>
<td>maximum number of processes (v.v_proc)</td>
</tr>
<tr>
<td>99</td>
<td>maximum global priority in sys class (MAXCLSYSPRI)</td>
</tr>
<tr>
<td>7925</td>
<td>maximum processes per user id (v.v_maxup)</td>
</tr>
<tr>
<td>30</td>
<td>auto update time limit in seconds (NAUTOUP)</td>
</tr>
<tr>
<td>25</td>
<td>page stealing low water mark (GPGSLO)</td>
</tr>
</tbody>
</table>

(continued)
kstats

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 root privilege. See kstat(3KSTAT) for more information.

In the Solaris 8 release, a new command, kstat(1M), is available that enables selection and display of kstats with a command-line interface. A Perl module, kstat(3EXT), is also available to process kstat information.
CHAPTER 2

Solaris Kernel Tunables

This section describes most of the Solaris kernel tunables. For information on NFS tunables, see Chapter 4. For information on TCP/IP tunables, see Chapter 3.

- “General Parameters” on page 30
- “fsflush and Related Tunables” on page 32
- “Process Sizing Tunables” on page 36
- “Paging-Related Tunables” on page 41
- “Swapping-Related Variables” on page 56
- “General Kernel Variables” on page 58
- “Kernel Memory Allocator” on page 59
- “General Driver” on page 62
- “General I/O” on page 63
- “General File System” on page 66
- “UFS” on page 71
- “TMPFS” on page 78
- “Pseudo Terminals” on page 79
- “Streams” on page 82
- “System V Message Queues” on page 84
- “System V Semaphores” on page 87
- “System V Shared Memory” on page 93
- “Scheduling” on page 97
- “Timers” on page 98
- “Sun4u Specific” on page 99
General Parameters

This section describes general kernel parameters relating to physical memory and stack size.

**physmem**

**Description**

Modifies the system’s idea of the number of physical pages of memory after the 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 with less physical memory. Note that 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
**lwp_default_stksize**

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

**Data Type**  
Integer

**Default**  
8192 for all 32-bit SPARC and IA based platforms  
16,384 for 64-bit sun4u platforms

**Range**  
0 to 262,144

**Units**  
Bytes in multiples of the value returned by `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) and 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 described above.

**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 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, because that space will generally be unused. The increased consumption means that other resources 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 will be a reduction in the number of threads which the kernel can create. This solution should be treated as no more than an interim workaround until the root cause is remedied.
fsflush and Related Tunables

This section describes fsflush and related tunables.

fsflush

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

- On every invocation, fsflush ...
  1. Flushes dirty file system pages over a certain age to disk.
  2. Examines a portion of memory and causes modified pages to be written to their backing store. Pages are written if they are modified and do not meet one of the following conditions:
     - Kernel page
     - Free
     - Locked
     - Associated with a swap device
     - Currently involved in an I/O operation

  The net effect is to flush pages from files which are mmap(ed) with write permission and which 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. Writes file system metadata to disk. This write is done every nth invocation, where n is computed from various configuration variables. See "tune_t_fsflushr" on page 33 and "autoup" on page 33 for details.

  Frequency of invocation, whether the memory scanning is executed, whether the file system data flushing occurs, and the frequency with which it will occur are configurable.

  For most systems, memory scanning and file system metadata syncing 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: 5

Range: 1 to MAXINT

Units: Seconds

Dynamic?: No

Validation: If the value is less than or equal to zero, the value is reset to 5 and a warning message is displayed. This check is only done at boot time.

When to Change: See autoup below.

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 sync 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 (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 around for a longer time in memory.

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 only done at boot time.

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

(total system pages \( \times \) `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
There are several potential situations for changing `autoup` and or `tune_t_fsflushr`:

- 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 (it might have changed because of a dynamic reconfiguration operation). Each invocation scans (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, 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. Syncs are done every Nth invocation of *fsflush* where $N = \frac{\text{autoup}}{\text{tune_t_fsflushr}}$. Because this is an integer division, if `tune_t_fsflushr` is greater than `autoup`, a sync will be 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` will have no effect on the frequency of sync operations.

**Data Type**
Signed integer

Solaris Kernel Tunables 35
Process Sizing Tunables

Several 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 variable is `maxusers`, which drives the values assigned to `max_nprocs` and `maxuprc`.

`maxusers`

**Description**

Originally, `maxusers` defined the number of logged in users the system could support. Various tables were sized based on this setting when a kernel was generated. Now, the Solaris release does much of its sizing based on the amount of memory on the system, so much of...
the past use of `maxusers` has changed. There are still a number of subsystems that are 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 lookup cache (DNLC)

**Data Type**
Signed integer

**Default**
Lesser of the amount of memory in Mbytes and 2048

**Range**
1 to 2048

**Note** - Values greater than 1024 must be specified in `/etc/system`. If a value greater than 4096 is provided, it is set to 4096.

**Units**
Users

**Dynamic?**
No. After computation of dependent variables 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 seen by the following message that displays on the system console:

```
out of processes
```

When the default number of processes is too high:

- Database servers that have a lot of memory and relatively few running processes, can save system memory by reducing the default value of `maxusers`. 
- File servers that have a lot of memory and few running processes can reduce this value, but should explicitly set the size of the DNLC. (See “ncsize” on page 66.)

- Compute servers that have a lot of memory and few running processes can reduce this value.

**Commitment Level**  
Unstable

**Change History**  
See “maxusers (Solaris 7 Release)” on page 167 for more information.

### reserved_procs

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

**Data Type**  
Signed integer

**Default**  
5

**Range**  
5 to MAXINT

**Units**  
Processes

**Dynamic?**  
No. Not used after the initial parameter computation.

**Validation**  
In the Solaris 8 release, any /etc/system setting is honored.

**Commitment Level**  
Unstable

**When to Change**  
Consider increasing to 10 + normal number of UID 0 (root) processes on system. This setting provides some cushion should it be necessary to obtain a root shell during a time when the system is otherwise unable to create user-level processes.
### pidmax

**Description**
This parameter specifies value of 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 constraint checking.

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

**Commitment Level**
Unstable

### max_nprocs

**Description**
Maximum number of processes that can be created on a system. Includes system and user processes. Any value entered 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 variable plays a role are:

- 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 the sun4d, sun4m, and Intel platforms

**Data Type**
Signed integer

**Default**
\[10 + (16 \times \text{maxusers})\]

**Range**
266 to value of `maxpid`

**Dynamic?**
No

**Validation**
Compared to `maxpid` and set to `maxpid` if larger. On the sun4d and Intel platforms an additional check is made against a platform-specific value. `max_nprocs` is set to the smallest value in the triplet \((\text{max}_\text{nprocs}, \text{maxpid}, \text{platform value})\). Both 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**
See "`max_nprocs` (Pre-Solaris 8 Releases)" on page 168 for more information.
maxuprc

Description
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
Compared to max_nprocs - reserved_procs and set to the smaller of the two.

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 Tunables

The Solaris environment is 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 variables.

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.

Solaris Kernel Tunables  41
If the page has been modified, a request is made to the pageout thread to schedule the page for I/O and the 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 scanning memory, no distinction is made as to the origin of the page. It may 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. (See “fastscan” on page 51 and “slowscan” on page 52 for more information.) As available memory falls between the range lotsfree and minfree, the system will linearly increase 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 variable to control a number of decisions about resource usage and behavior.

The system also attempts to constrain itself to use not more than 4% of one CPU for pageout operations. The algorithm is to look through some amount of memory between slowscan and fastscan, and stop 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.

Caution - We recommend that all tuning of the VM system be removed from /etc/system. Run with the default settings and determine if it is necessary to adjust any of these parameters. Do not enable priority_paging or adjust cachefree. These are no longer needed, although still present in the kernel. Manipulating them will almost certainly result in performance degradation when the page scanner runs.

Beginning in the Solaris 7 5/99 release, dynamic reconfiguration (DR) for CPU and memory is supported. The behavior of the system in a DR operation involving the addition or deletion of memory is to recalculate 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 which case the value is reset.

lotsfree

Description 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.
<table>
<thead>
<tr>
<th>Data Type</th>
<th>Unsigned long</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>The greater of 1/64th of physical memory or 512 Kbytes</td>
</tr>
<tr>
<td>Range</td>
<td>The minimum value is 512 Kbytes or 1/64th of physical memory, whichever is greater, expressed as pages using the page size returned by <code>getpagesize(3C)</code>. The maximum is the number of physical memory pages. The maximum value should be no more than 30% of physical memory. The system does no enforcement of this range other than that described in the Validation section.</td>
</tr>
<tr>
<td>Units</td>
<td>Pages</td>
</tr>
<tr>
<td>Dynamic?</td>
<td>Yes, but dynamic changes are lost if a memory based DR operation occurs.</td>
</tr>
<tr>
<td>Validation</td>
<td>If <code>lotsfree</code> is greater than the amount of physical memory, the value is reset to the default.</td>
</tr>
<tr>
<td>Implicit</td>
<td>The relationship of <code>cachefree</code> is greater than or equal to <code>lotsfree</code>, which is greater than <code>desfree</code>, which is greater than <code>minfree</code>, must be maintained at all times.</td>
</tr>
<tr>
<td>When to Change</td>
<td>When demand for pages is subject to sudden sharp spikes, the memory algorithm might not be able to keep up with demand. One way to work around this problem 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, but a system doing heavy file system I/O might need to adjust this parameter. For systems with relatively static workloads and large amounts of memory, adjust this value downwards. The minimum acceptable value is 512 Kbytes expressed as pages using the page size returned by <code>getpagesize(3C)</code>.</td>
</tr>
</tbody>
</table>

Solaris Kernel Tunables 43
<table>
<thead>
<tr>
<th>Commitment Level</th>
<th>Unstable</th>
</tr>
</thead>
</table>

**desfree**

**Description**
Amount of memory desired 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(3C).

The maximum is the number of physical memory pages. The maximum value should be no more than 15% of physical memory. The system does no enforcement of 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 whatever was provided in the /etc/system file or was 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 cachefree is greater than or equal to lotsfree, which is 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 variable. When the new value nears or exceeds the amount of available memory on the system:

- 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 Version 3 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, adjust this value downwards. The minimum acceptable value is 256 Kbytes expressed as pages using the page size returned by `getpagesize(3C)`.

**Commitment Level**

Unstable

**minfree**

**Description**

Minimum acceptable memory level. When memory drops below this number, the system biases allocations toward those necessary to successfully complete pageout operations or to swap processes completely out of memory, and either 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(3C)`.

The maximum is the number of physical memory pages. The maximum value should be no more than 7.5% of physical memory. The system does
no enforcement of 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 whatever was provided in the `/etc/system` file or was 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 `cachefree` is greater than or equal to `lotsfree`, which 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, adjust this value downwards. The minimum acceptable value is 128 Kbytes expressed as pages using the page size returned by `getpagesize(3C)`.

**Commitment Level**

Unstable

**throttlefree**

**Description**

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(3C)`.

The maximum is the number of physical memory pages. The maximum value should be no more
than 4% of physical memory. The system does no enforcement of 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 whatever was provided in the `/etc/system` file or was 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 `cachefree` is greater than or equal to `lotsfree`, which 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, adjust this value downwards. The minimum acceptable value is 128 Kbytes expressed as pages using the page size returned by `getpagesize(3C)`.

**Commitment Level**

Unstable

---

**pageout_reserve**

**Description**

Number of pages reserved for the exclusive use of the pageout or scheduler threads. When available memory is less than this value, non-blocking 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.
<table>
<thead>
<tr>
<th><strong>Data Type</strong></th>
<th>Unsigned integer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>throttlefree / 2</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>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% of physical memory. The system does no enforcement of this range other than that described in the Validation section.</td>
</tr>
<tr>
<td><strong>Units</strong></td>
<td>Pages</td>
</tr>
<tr>
<td><strong>Dynamic?</strong></td>
<td>Yes, unless dynamic reconfigurion operations that add or delete memory occur. At that point, the value is reset to whatever was provided in the /etc/system file or was calculated from the new physical memory value.</td>
</tr>
<tr>
<td><strong>Validation</strong></td>
<td>If pageout_reserve is greater than throttlefree / 2, pageout_reserve is set to throttlefree / 2. No message is displayed.</td>
</tr>
<tr>
<td><strong>Implicit</strong></td>
<td>The relationship of cachefree is greater than or equal to lotsfree, which is greater than desfree, which is greater than minfree, which is greater than or equal to throttlefree, should be maintained at all times.</td>
</tr>
<tr>
<td><strong>When to Change</strong></td>
<td>The default value is generally adequate. For systems with relatively static workloads and large amounts of memory, adjust this value downwards. The minimum acceptable value is 64 Kbytes expressed as pages using the page size returned by getpagesize(3C).</td>
</tr>
<tr>
<td><strong>Commitment Level</strong></td>
<td>Unstable</td>
</tr>
</tbody>
</table>
**cachefree**

**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_page 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

**Change History**  
See “cachefree (Solaris 2.6 and Solaris 7 Releases)” on page 170 for more information.

---

**priority_page**

**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_page settings in the /etc/system file.

---

**Commitment Level**  
Obsolete
### Change History

See “priority_paging (Solaris 2.6 and 7 Releases)” on page 170 for more information.

### pages_pp_maximum

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Defines the number of pages that the system requires be unlocked. If a request to lock pages would force available memory below this value, that request is refused.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Type</strong></td>
<td>Unsigned long</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>Maximum of the triplet (200, tune_t_minarmem + 100, [10% of memory available at boot time])</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>Default value to no more than 20% of physical memory. The systems does no enforcement of this range other than that described in the Validation section.</td>
</tr>
<tr>
<td><strong>Units</strong></td>
<td>Pages</td>
</tr>
<tr>
<td><strong>Dynamic?</strong></td>
<td>Yes, unless dynamic reconfiguration operations that add or delete memory occur. At that point, the value is reset to whatever was provided in the /etc/system file or was calculated.</td>
</tr>
<tr>
<td><strong>Validation</strong></td>
<td>Maximum of the quadruplet (200, tune_t_minarmem + 100, [10% of memory available], and the value from /etc/system). No message is displayed if the value from /etc/system is increased. Done only at boot time.</td>
</tr>
<tr>
<td><strong>When to Change</strong></td>
<td>When memory locking requests or attaching to a shared memory segment with the SHARE_MMU flag fails, yet the amount of memory available seems to be sufficient. Keeping 10% of memory free on a 32-Gbyte system might be excessive. Excessively large values can cause memory locking requests to fail unnecessarily.</td>
</tr>
<tr>
<td><strong>Commitment Level</strong></td>
<td>Unstable</td>
</tr>
</tbody>
</table>

---

50  
### tune_t_minarmem

**Description**
The minimum available resident (not swappable) memory to maintain in order to avoid deadlock. Used to reserve a portion of memory for use by the core of the operating system. 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 it if the system locks up and debugging information indicates the problem was because no memory was available.

**Commitment Level**
Unstable

### fastscan

**Description**
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
<table>
<thead>
<tr>
<th>Dynamic?</th>
<th>Yes, unless dynamic reconfiguration operations that add or delete memory occur. At that point, the value is reset to whatever was provided by /etc/system or was calculated from the new physical memory value.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation</td>
<td>Maximum value is the lesser of 64 Mbytes and 1/2 of physical memory.</td>
</tr>
<tr>
<td>When to Change</td>
<td>When more aggressive scanning of memory is desired during periods of memory shortfall, especially if the system is subject to periods of intense memory demand or when performing heavy file I/O.</td>
</tr>
<tr>
<td>Commitment Level</td>
<td>Unstable</td>
</tr>
</tbody>
</table>

**slowscan**

<table>
<thead>
<tr>
<th>Description</th>
<th>Minimum number of pages per second that the system looks at when attempting to reclaim memory.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Type</td>
<td>Signed integer</td>
</tr>
<tr>
<td>Default</td>
<td>The smaller of 1/20th of physical memory in pages and 100.</td>
</tr>
<tr>
<td>Range</td>
<td>1 to fastscan / 2</td>
</tr>
<tr>
<td>Units</td>
<td>Pages</td>
</tr>
<tr>
<td>Dynamic?</td>
<td>Yes, unless dynamic reconfiguration operations that add or delete memory occur. At that point, the value is reset to whatever was provided in the /etc/system file or was calculated from the new physical memory value.</td>
</tr>
<tr>
<td>Validation</td>
<td>If slowscan is larger than fastscan / 2, slowscan is reset to fastscan / 2. No message is displayed.</td>
</tr>
<tr>
<td>When to Change</td>
<td>When more aggressive scanning of memory is desired during periods of memory shortfall.</td>
</tr>
</tbody>
</table>
especially if the system is subject to periods of intense memory demand.

Commitment Level

Unstable

min_percent_cpu

Description
Minimum percentage of CPU that pageout can consume. This variable 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 environment 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
reclaim. The distance between the front hand and the back hand is handspreadpages.

<table>
<thead>
<tr>
<th><strong>Data Type</strong></th>
<th>Unsigned long</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>fastscan</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>1 to maximum number of physical memory pages on the system</td>
</tr>
<tr>
<td><strong>Units</strong></td>
<td>Pages</td>
</tr>
<tr>
<td><strong>Dynamic?</strong></td>
<td>Yes. This parameter requires that the kernel variable reset_hands also be set to a non-zero value. Once the new value of handspreadpages has been recognized, reset_hands is set to zero.</td>
</tr>
<tr>
<td><strong>Validation</strong></td>
<td>Set to lesser of the amount of physical memory and the handspreadpages value</td>
</tr>
<tr>
<td><strong>When to Change</strong></td>
<td>When you want the amount of time that pages are potentially resident before reclaim is increased. Increasing this value increases the separation between the hands, and therefore, the amount of time before a page can be reclaimed.</td>
</tr>
<tr>
<td><strong>Commitment Level</strong></td>
<td>Unstable</td>
</tr>
</tbody>
</table>

**pages_before_pager**

**Description**
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.

<table>
<thead>
<tr>
<th><strong>Data Type</strong></th>
<th>Signed integer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>200</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>1 to amount of physical memory</td>
</tr>
<tr>
<td><strong>Units</strong></td>
<td>Pages</td>
</tr>
</tbody>
</table>
Dynamic? No

Validation None

When to Change 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.

When the system is subject to bursts of severe memory pressure. A larger value here helps to keep a bigger cushion against the pressure.

Commitment Level Unstable

maxpgio

Description 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 used by the paging system. It 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 the swap device is a striped device.

Commitment Level Unstable

Swapping-Related Variables

Swapping in the Solaris environment 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.

\texttt{swapfs\_reserve}

\textbf{Description}  
Amount of system memory that is reserved for use by system (UID = 0) processes.

\textbf{Data Type}  
Unsigned long

\textbf{Default}  
The smaller of 4 Mbytes and 1/16th of physical memory

\textbf{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 \texttt{getpagesize(3C)}.

The maximum is the number of physical memory pages. The maximum value should be no more than 10% of physical memory. The system does no enforcement of this range other than that described in the Validation section.

\textbf{Units}  
Pages

\textbf{Dynamic?}  
No

\textbf{Validation}  
None
### When to Change

Generally not necessary. Only change on recommendation of 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**

Amount of physical memory that is desired 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
General Kernel Variables

noexec_user_stack

Description
Enables the stack to be marked as non-executable. This helps in making buffer-overflow attacks more difficult.

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

Note - This variable exists on all systems running Solaris 2.6 or later, but it is only effective on sun4u, sun4m, and sun4d architectures.

Data Type
Signed integer

Default
0 (disabled)

Range
0 (disabled), 1 (enabled)

Units
Toggle (on/off)

Dynamic?
Yes. Does not affect currently running processes—only those 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(2) to make the stack executable.

Commitment Level
Unstable

Change History
See “noexec_user_stack (Solaris 2.6 and Solaris 7 Releases)” on page 171 for more information.
Kernel Memory Allocator

The Solaris kernel memory allocator distributes chunks of memory for use by entities 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 of the caches that the allocator manages can be seen with the `kstat -c kmem_cache` command. Specialized caches can be viewed with the `crash(1M)` command, using the `kmastat` operator.

Occasionally, systems might panic because of memory corruption. The kernel memory allocator supports a debugging interface that performs various integrity checks on the buffers as well as collecting information on the allocators. The integrity checks provide the opportunity to detect errors closer to where they actually occurred, and 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 environment. Prior to the Solaris 2.5 release, these options were not usable in released Solaris versions. Starting with the Solaris 2.5 release, a subset of these options are available and they are controlled by the `kmem_flags` variable, which was set by booting `kadb`, and then setting the variable before starting the kernel. Because of issues with the timing of the instantiation of the kernel memory allocator and the parsing of `/etc/system`, 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.
<table>
<thead>
<tr>
<th>Flag</th>
<th>Setting</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUDIT</td>
<td>0x1</td>
<td>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 and when space is exhausted, earlier records are reclaimed.</td>
</tr>
<tr>
<td>TEST</td>
<td>0x2</td>
<td>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, this indicates that the memory was probably used by an entity that had previously allocated and freed the buffer. If an overwrite is seen, the system panics.</td>
</tr>
<tr>
<td>REDZONE</td>
<td>0x4</td>
<td>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 seen, the kernel panics.</td>
</tr>
<tr>
<td>Flag</td>
<td>Setting</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>CONTENTS 0x8</td>
<td>The allocator logs up to 256 bytes of buffer contents when the buffer is freed. Requires that AUDIT also be set. The numeric value of these flags can be logically added (OR'ed) together and set by the /etc/system file in the Solaris 8 release, or for previous releases, by booting kadb and setting the flags before starting the kernel.</td>
<td></td>
</tr>
<tr>
<td>LITE 0x100</td>
<td>Does minimal sanity 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.</td>
<td></td>
</tr>
</tbody>
</table>

### 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

**moddebug**

**Description**
Variable that you can set to values that cause messages about various steps in the module loading process to be displayed.

**Data Type**
Signed integer

**Default**
0 (messages off)

**Range**
The most useful values are:
- 0x80000000 - Prints [un]loading... message. For every module loaded, messages such as the following would 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 would 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-1/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,
```

(continued)
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 doesn't print any additional information beyond what the detailed error message flag does during system boot, but it 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 print detailed messages is set, system boot is slowed down considerably by the number of messages written to the console.

Commitment Level Unstable

General I/O

maxphys

Description Maximum size of physical I/O requests. If a driver sees a request larger than this size, the driver breaks the request into maxphys size chunks. File systems can and do impose their own limit.

Data Type Signed integer
### **Default**

126,976 (sun4m and sun4d), 131,072 (sun4u), 57,344 (Intel). 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 into 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.

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. Note that 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**

"Hard" limit on file descriptors that a single process might have open. To override this limit requires superuser privilege.

**Data Type**

Signed integer

**Default**

1024
<table>
<thead>
<tr>
<th><strong>Range</strong></th>
<th>1 to MAXINT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Units</strong></td>
<td>File descriptors</td>
</tr>
<tr>
<td><strong>Dynamic?</strong></td>
<td>No</td>
</tr>
<tr>
<td><strong>Validation</strong></td>
<td>None</td>
</tr>
</tbody>
</table>

**When to Change**

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

- 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.

- `select(3C)` is by default limited to 1024 descriptors per `fd_set`. 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 sees 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 is useful for daemons such as `inetd`.

**Commitment Level**

Unstable

### `rlim_fd_cur`

**Description**

"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 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 and 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 is possibly not necessary for a program to use setrlimit(2) to increase the maximum number of file descriptors available to it.
Commitment Level Unstable
Change History See “rlim_fd_cur (Solaris 7 Release and Earlier)” on page 172 for more information.

General File System
ncsize

Description 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 lookup 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 is difficult to determine whether the cache is too small. It is possible to infer this by noting the number of enters returned by `kstat -n ncstats`. If the number seems high given the system workload and file access pattern, this may be due to the size of the DNLC.

Starting with the Solaris 8 6/00 release, `kstat -n dnlcstats`, is available for you 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` represents otherwise valid entries which were reclaimed because the cache was too small.

Note that excessive values of `ncsize` have an immediate impact on the system since 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 also has a further affect on UFS and NFS unless `ufs_inode` and `nfs:nfs_rnode` are explicitly set.

### Commitment Level
Unstable

### rstchown

#### Description
Indicates whether the POSIX semantics for the `chown(2)` system call are in effect. POSIX semantics are:

- 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.

**Data Type**
Signed integer

**Default**
1, indicating that POSIX semantics are used

**Range**
0 = POSIX semantics not in force, 1 = POSIX semantics used

**Units**
Toggle (on/off)

**Dynamic?**
Yes

**Validation**
None

**When to Change**
When POSIX semantics are not desired. Note that turning off POSIX semantics opens the potential for various security holes. It also opens the possibility of a user changing ownership of a file to another user and being unable to retrieve the file back without intervention from the user or the system administrator.

**Commitment Level**
Obsolete

### segkpsize

**Description**
Specify 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 systems running 64-bit kernels. Systems running 64-bit kernels use 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 - 24 Gbytes
32-bit kernels, 512 Mbytes

Units
Mbytes

Dynamic?
No

Validation
Value is compared to minimum and maximum sizes (512 Mbytes and 24 Gbytes for 64-bit systems) and if smaller than the minimum or larger than the maximum, it is reset to 2 Gbytes and 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 constraints checking and 50% of physical memory.

When to Change
This is one of the steps necessary 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, 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
See “segkpsize (Solaris 7 and Earlier Releases)” on page 173 for more information.

dnlc_dir_enable

Description
Enables large directory caching.

Data Type
Unsigned integer

Default
1 (enabled)

Range
0 (disabled), 1 (enabled)

Dynamic?
Yes, but do not change this tunable dynamically. It is possible to enable it if originally disabled, or
to disable it if 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, but if problems occur, then set dnlc_dir_enable to 0 to disable caching.

**Commitment Level**

Unstable

---

**dnlc_dir_min_size**

**Description**

Minimum number of entries before caching for one directory.

**Data Type**

Unsigned integer

**Default**

40

**Range**

0 to MAXUINT (no maximum)

**Units**

Dynamic?

Yes, it can be changed at any time.

**Validation**

No

**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**
Maximum number of entries cached for one directory.

**Data Type**
Unsigned integer

**Default**
MAXUINT (no maximum)

**Range**
0 to MAXUINT

**Dynamic?**
Yes, it can be changed at any time.

**Validation**
No

**When to Change**
If performance problems occur with large directories, then decrease `dnlc_dir_max_size`.

**Commitment Level**
Unstable

### UFS

#### bufhwm

**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 kadb -d. 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:
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

ndquot

Description

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

Number of inodes to be held in memory. Inodes are cached globally (for UFS), not on a per-file system basis.

A key variable 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 water mark of ufs_ninode / 4 are computed.

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

1. 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).

2. 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 do this. Each thread is responsible for one of the queues.

When the deferred processing is done, the system drops the inode onto either a delete or 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 it is in excess of the low watermark, the thread associated with the queue is awakened. After it 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% 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 pops two inodes off the idle queue and "idles" them (flushes pages and frees inodes). It 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, but it 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 sync, unmount, or remount occur.

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

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Signed integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>ncszie</td>
</tr>
<tr>
<td>Range</td>
<td>0 to MAXINT</td>
</tr>
<tr>
<td>Units</td>
<td>Inodes</td>
</tr>
<tr>
<td>Dynamic?</td>
<td>Yes</td>
</tr>
<tr>
<td>Validation</td>
<td>If ufs_ninode is less than or equal to zero, the value is set to ncszie.</td>
</tr>
</tbody>
</table>
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` may be too small. Excessive inode idling (described previously) can also be a problem.

This situation can be identified 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:ufs_WRITES

Description
If `ufs_WRITES` is non-zero, the number of bytes outstanding for writes on a file is checked. See `ufs_HW` subsequently to determine whether the write should be issued or should be 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), 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:ufs_LW and ufs:ufs_HW

**Description**

ufs_HW is 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**

256 x 1024 for ufs_LW and 384 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 awake and find that they either cannot issue a write (when ufs_LW and ufs_HW are too close) or when 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 is not a per-file system setting.
When *ufs_throttles* is a non-trivial number, *ufs_throttles* can currently be accessed only with a kernel debugger.

| Commitment Level | Unstable |

---

**TMPFS**

**tmpfs:** *tmpfs_maxkmem*

<table>
<thead>
<tr>
<th>Description</th>
<th>Maximum amount of kernel memory that TMPFS can use for its data structures (tmpnodes and directory entries).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Type</td>
<td>Unsigned long</td>
</tr>
<tr>
<td>Default</td>
<td></td>
</tr>
<tr>
<td>Range</td>
<td>Number of bytes in one page (8192 for UltraSPARC™ systems, 4096 for all others) to 25% of the available kernel memory at the time TMPFS was first used.</td>
</tr>
<tr>
<td>Units</td>
<td>Bytes</td>
</tr>
<tr>
<td>Dynamic?</td>
<td>Yes</td>
</tr>
<tr>
<td>Validation</td>
<td>None</td>
</tr>
<tr>
<td>When to Change</td>
<td>Increase if the following message is displayed on the console or written in the messages file.</td>
</tr>
</tbody>
</table>

```
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, which can be examined with a kernel debugger.

| Commitment Level | Unstable |
**tmpfs:tmpfs_minfree**

**Description**  
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**  
Bytes

**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 system messages file displays the following message.

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

**Commitment Level**  
Unstable

---

**Pseudo Terminals**

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

- 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 desired 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 and should be changed only to increase the number or to decrease the default value.
Three related variables are used in the configuration process:

- **pt_cnt** - Default maximum number of pty's
- **pt_pctofmem** - Percentage of kernel memory that can be dedicated to pty support structures
- **pt_max_pty** - Hard maximum for number of pty's

**pt_cnt** has a default value of zero, which tells the system to limit logins based on the amount of memory specified in **pt_pctofmem**, unless **pt_max_pty** is set. If **pt_cnt** is non-zero, pty's are allocated until this limit. When that threshold is crossed, the system looks at **pt_max_pty**. If that has a non-zero value, it is compared to **pt_cnt** and 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 pty's 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 their default values of zero.

To put a hard limit on pty's that is different than the maximum derived from **pt_pctofmem**, set **pt_cnt** and **ptms_ptymax** in /etc/system to the number of pty's desired. 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 desired percentage. For example, set **pt_pctofmem=10** for a 10% 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 /dev/pts entries available 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** next). 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.
<table>
<thead>
<tr>
<th><strong>Data Type</strong></th>
<th>Unsigned integer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>0 to maxpid</td>
</tr>
<tr>
<td><strong>Units</strong></td>
<td>logins/windows</td>
</tr>
<tr>
<td><strong>Dynamic?</strong></td>
<td>No</td>
</tr>
<tr>
<td><strong>Validation</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>When to Change</strong></td>
<td>When you want to explicitly control the number of users that can remotely log in to the system.</td>
</tr>
<tr>
<td><strong>Commitment Level</strong></td>
<td>Unstable</td>
</tr>
<tr>
<td><strong>Change History</strong></td>
<td>See “pt_cnt (Solaris 7 and Earlier Releases)” on page 174 for more information.</td>
</tr>
</tbody>
</table>

**pt_pctofmem**

**Description**
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.

<table>
<thead>
<tr>
<th><strong>Data Type</strong></th>
<th>Unsigned integer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>5</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>0 to 100</td>
</tr>
<tr>
<td><strong>Units</strong></td>
<td>Percentage</td>
</tr>
<tr>
<td><strong>Dynamic?</strong></td>
<td>No</td>
</tr>
<tr>
<td><strong>Validation</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>When to Change</strong></td>
<td>When you want to either restrict or increase the number of users that can log in to the system. A value of zero means that no remote users can log in to the system.</td>
</tr>
</tbody>
</table>
**Commitment Level**
Unstable

**pt_max_pty**

**Description**
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**

**nstrpush**

**Description**
Number of modules that can be inserted into (pushed onto) into a stream.

**Data Type**
Signed integer

**Default**
9

**Range**
9 to 16
<table>
<thead>
<tr>
<th>Units</th>
<th>Modules</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamic?</td>
<td>Yes</td>
</tr>
<tr>
<td>Validation</td>
<td>None</td>
</tr>
<tr>
<td>When to Change</td>
<td>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.</td>
</tr>
<tr>
<td>Commitment Level</td>
<td>Unstable</td>
</tr>
</tbody>
</table>

**strmsgsz**

<table>
<thead>
<tr>
<th>Description</th>
<th>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(2) exceeding this size is broken into multiple messages.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Type</td>
<td>Signed integer</td>
</tr>
<tr>
<td>Default</td>
<td>65,536</td>
</tr>
<tr>
<td>Range</td>
<td>0 to 262,144</td>
</tr>
<tr>
<td>Units</td>
<td>Bytes</td>
</tr>
<tr>
<td>Dynamic?</td>
<td>Yes</td>
</tr>
<tr>
<td>Validation</td>
<td>None</td>
</tr>
<tr>
<td>When to Change</td>
<td>When putmsg(2) calls return ERANGE.</td>
</tr>
<tr>
<td>Commitment Level</td>
<td>Unstable</td>
</tr>
</tbody>
</table>

**strctlksz**

<table>
<thead>
<tr>
<th>Description</th>
<th>Maximum number of bytes that a single system call can pass to a STREAM to be placed in the control part of a message.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Type</td>
<td>Signed integer</td>
</tr>
</tbody>
</table>
System V Message Queues

System V message queues provide a message-passing interface that enables 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.

The module is dynamically loaded on first reference. Parameters provided to the subsystem are validated at that time. Entries in the /etc/system file must contain the msgsys: prefix.

This facility is different from the POSIX 1003.1b message queue facility.

The Solaris 8 release modified the use of some of the parameters for this facility. The msgsys:msginfo_msgssz, msgsys:msginfo_msgmap, and msgsys:msginfo_msgseg parameters are now obsolete. The variables have been left in place to avoid error messages. Any values applied are ignored.

The maximum number of messages the facility can handle at any one point in time is now entirely defined by msgsys:msginfo_msgtql. An array of message headers sized to the value specified in this variable is allocated and initialized as a free list. When an attempt is made to send a message, the free list is examined and if a header is available, a buffer is allocated from kernel memory to handle the message data. The data is copied into the buffer and the message is placed in the destination queue. When the message is read, the buffer is freed and the header placed on the free list.

Previous Solaris versions would limit the number of messages either by setting msgsys:msginfo_msgtql or by limiting the number of memory segments and the

<table>
<thead>
<tr>
<th>Default</th>
<th>1024</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range</td>
<td>0 to MAXINT</td>
</tr>
<tr>
<td>Units</td>
<td>Bytes</td>
</tr>
<tr>
<td>Dynamic?</td>
<td>Yes</td>
</tr>
<tr>
<td>Validation</td>
<td>None</td>
</tr>
<tr>
<td>When to Change</td>
<td>At the direction of your software vendor. putmsg(2) calls return ERANGE if they attempt to exceed this limit.</td>
</tr>
<tr>
<td>Commitment Level</td>
<td>Unstable</td>
</tr>
</tbody>
</table>
size of the segments that were allocated to a message buffer pool. When the module is first loaded, it allocates a number of data structures needed to manage messages. The total space allocated for these structures must not exceed 25% of available kernel memory, or the attempt to load fails and the following message is displayed.

```
msgsys: can't load module, too much memory requested
```

Unlike previous Solaris versions, a message buffer pool is not allocated as part of set up and is no longer considered in the 25% of memory check.

### msgsys: msginfo_msgmax

<table>
<thead>
<tr>
<th>Description</th>
<th>Maximum size of System V message.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Type</td>
<td>Unsigned long</td>
</tr>
<tr>
<td>Default</td>
<td>2048</td>
</tr>
<tr>
<td>Range</td>
<td>0 to amount of physical memory</td>
</tr>
<tr>
<td>Units</td>
<td>Bytes</td>
</tr>
<tr>
<td>Dynamic?</td>
<td>No. Loaded into msgmax field of msginfo structure.</td>
</tr>
<tr>
<td>Validation</td>
<td>None</td>
</tr>
<tr>
<td>When to Change</td>
<td>When msgsnd(2) calls return with error of EINVAL or at the recommendation of a software vendor.</td>
</tr>
<tr>
<td>Commitment Level</td>
<td>Unstable</td>
</tr>
</tbody>
</table>

### msgsys: msginfo_msgmnb

<table>
<thead>
<tr>
<th>Description</th>
<th>Maximum number of bytes that can be on any one message queue.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Type</td>
<td>Unsigned long</td>
</tr>
<tr>
<td>Default</td>
<td>4096</td>
</tr>
<tr>
<td>Range</td>
<td>0 to amount of physical memory</td>
</tr>
<tr>
<td>Units</td>
<td>Bytes</td>
</tr>
<tr>
<td>Dynamic?</td>
<td>No. Loaded into msgmnb field of msginfo structure.</td>
</tr>
<tr>
<td>Validation</td>
<td>None</td>
</tr>
<tr>
<td>When to Change</td>
<td>When msgsnd() calls block or return with an error of EGAIN, or at the recommendation of a software vendor.</td>
</tr>
<tr>
<td>Commitment Level</td>
<td>Unstable</td>
</tr>
</tbody>
</table>

**msgsys:msginfo_msgmni**

| 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_msgsqmax**

| Description | Maximum number of messages that can be created. If a msgsnd call attempts to exceed this |

---

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

---

System V Semaphores

System V semaphores provide counting semaphores in the Solaris environment. 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). The ability is offered to do operations on a group of semaphores simultaneously as well as to have the system undo the last operation by a process if it 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.

This facility is different from the POSIX 1003.1b semaphore facility.
semsys:seminfo_sem MSI

Description
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 and or system messages file.

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. The application sees a return code of ENOSPC from a semget(2) call.

Commitment Level
Unstable

semsys:seminfo_semmns

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_semvmx

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_semmsl

Description

Maximum number of System V semaphores per semaphore identifier.

Data Type

Signed integer
### semsys:seminfo_semopm

**Description**
Maximum number of System V semaphore operations per `semop(2)` call. This parameter refers to the number of `sembufs` in the `sops` array that is provided to the `semop()` system call.

**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 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. The application sees a return code of E2BIG from a `semop()` call.

**Commitment Level**

Unstable

---

**semsys:seminfo_semmnu**

**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
### Changes From Previous Release

See “semsys:seminfo_semmnu (Solaris 8 Update 3)” on page 180 for more information.

---

**semsys:seminfo_semume**

**Description**

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_semaem**

**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

System V Shared Memory

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 enables a faster I/O path to be followed and improves memory usage because a number of kernel resources describing the segment are now shared between all processes attaching to the segment in ISM mode.

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

This facility is different from the POSIX 1003.1b shared memory facility.

shmsys:shminfo_shmmax

Description

Maximum size of system V shared memory segment that can be created. This parameter is an upper limit that is checked before the system sees
if it actually has the physical resources to create the requested memory segment.

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Unsigned long</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>1,048,576</td>
</tr>
<tr>
<td>Range</td>
<td>0 - MAXINT on 32-bit systems, MAXINT64 on 64-bit systems</td>
</tr>
<tr>
<td>Units</td>
<td>Bytes</td>
</tr>
<tr>
<td>Dynamic?</td>
<td>No. Loaded into shmmax field of shminfo structure.</td>
</tr>
<tr>
<td>Validation</td>
<td>None</td>
</tr>
<tr>
<td>When to Change</td>
<td>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.</td>
</tr>
<tr>
<td>Commitment Level</td>
<td>Unstable</td>
</tr>
</tbody>
</table>

**shmsys:shminfo_shmmin**

<table>
<thead>
<tr>
<th>Description</th>
<th>Minimum size of system V shared memory segment that can be created.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Type</td>
<td>Unsigned long</td>
</tr>
<tr>
<td>Default</td>
<td>1</td>
</tr>
<tr>
<td>Range</td>
<td>0 to amount of physical memory</td>
</tr>
<tr>
<td>Units</td>
<td>Bytes</td>
</tr>
<tr>
<td>When to Change</td>
<td>Not recommended. System programs such as powerd might fail if this value is too large. Programs attempting to create a section smaller than the value of shminfo_shmmin will see an EINVAL error when attempting to create the segment and generally, will exit.</td>
</tr>
</tbody>
</table>
Commitment Level | Unstable
---|---
Changes From Previous Release | See “shmsys:shminfo_shmmni (Solaris 8 Update 3)” on page 179 for more information.

### shmsys:shminfo_shmmni

<table>
<thead>
<tr>
<th>Description</th>
<th>System wide limit on number of shared memory segments that can be created.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Type</td>
<td>Signed integer</td>
</tr>
<tr>
<td>Default</td>
<td>100</td>
</tr>
<tr>
<td>Range</td>
<td>0 to MAXINT</td>
</tr>
<tr>
<td>Dynamic?</td>
<td>No. Loaded into shmmni field of shminfo structure.</td>
</tr>
<tr>
<td>Validation</td>
<td>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.</td>
</tr>
<tr>
<td>When to Change</td>
<td>When the system limits are too low. Generally changed on the recommendation of software vendors.</td>
</tr>
<tr>
<td>Commitment Level</td>
<td>Unstable</td>
</tr>
</tbody>
</table>

### shmsys:shminfo_shmseg

<table>
<thead>
<tr>
<th>Description</th>
<th>Limit on the number of shared memory segments that any one process can create.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Type</td>
<td>Signed short</td>
</tr>
<tr>
<td>Default</td>
<td>6</td>
</tr>
<tr>
<td>Range</td>
<td>0 to 32,767</td>
</tr>
</tbody>
</table>
Dynamic? No. Loaded into shmseg 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

segspt_minfree

Description Pages of system memory that cannot be allocated for ISM shared memory.

Data Type Unsigned long

Default 5% of available system memory when first ISM segment is created.

Range 0 to 32,767

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, this parameter can be tuned downward. 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.
Scheduling

rechoose_interval

**Description**
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 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 (psrset(1M)) capabilities available as of the Solaris 2.6 release or processor binding (pbind(1M)) before changing this parameter.
Timers

**hires_tick**

**Description**  
Variable that when set causes the Solaris environment 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**  
Number of POSIX timers available.

**Data Type**  
Signed integer

**Default**  
32

**Range**  
0 to MAXINT

**Dynamic?**  
No. Increasing value can cause a system crash.

**Validation**  
None

**When to Change**  
When default number of timers offered by system is inadequate. Applications see an **EGAIN** error when executing **timer_create** system calls.
Sun4u Specific

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 variable 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.

<table>
<thead>
<tr>
<th>Dynamic?</th>
<th>Yes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Validation</td>
<td>None. Values larger than 2 cause a number of WARNING: AS_2_BIN: bad consistent coloring value messages to appear on the console and the system hangs immediately thereafter. A power-cycle is required to recover.</td>
</tr>
<tr>
<td>When to Change</td>
<td>When the primary workload of the system is a set of long-running high-performance computing (HPC) application(s). 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) will not benefit from changes.</td>
</tr>
<tr>
<td>Commitment Level</td>
<td>Unstable</td>
</tr>
</tbody>
</table>
NFS Tunable Parameters

This section describes the NFS tunable parameters. For information on kernel tunables, see Chapter 2. For information on TCP/IP tunables, see Chapter 4.

- "NFS Module Parameters" on page 101
- "nfssrv Module Parameters" on page 125
- "rpcmod Module Parameters" on page 129

Tuning the NFS Environment

You can define these parameters in the /etc/system file, which is read during the boot process. Each parameter can be identified by the name of the kernel module that it is in and a parameter name that identifies it. See "Tuning a Solaris System" on page 19 for more information.

Note - The names of the symbols, 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 relating to the NFS kernel module.
nfs:nfs3_pathconf_disable_cache

Description
Controls the caching of pathconf(2) information for NFS Version 3 mounted file systems.

Data Type
Integer (32-bit)

Default
0 (caching enabled)

Range
0 (caching enabled), 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, then use this parameter to disable caching because there is no mechanism for the client to validate its cache entry.

Stability Level
Evolving

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 by using these attributes. The over-the-wire time stamp values are unsigned and 32-bits long, so all values have been legal.

However, on a system running a 32-bit Solaris release, the time stamp values are signed and 32-bits long. Thus, it would be possible to have a time stamp representation that appeared to be prior to January 1, 1970, or pre-epoch.
The problem on a system running a 64–bit Solaris release is slightly different. The time stamp values on the 64–bit Solaris release 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, one 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, but 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), 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 time stamp values on some files to be set very far in the future or very far in the past. If access to these files is desired using NFS mounted file systems, then set this parameter to 1 to allow the time stamp values to be passed through unchecked.

### Stability Level
Evolving

### 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 to be too large might result in real situations where a retransmission was required to not be detected for long periods of time.
Stability Level: Evolving

**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 to be too large might result in real situations where a retransmission was required to not be detected for long periods of time.

Stability Level  Evolving

**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), 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 time stamp on the file or if the granularity of the time stamp 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, thus making the changes visible to applications running on the client immediately.

**Stability Level**  
Evolving

### nfs:nfs3_do_symmlink_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), 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 time stamp on the file or if the granularity of the time stamp 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, thus making the changes visible to applications running on the client immediately.

**Stability Level**  
Evolving

### 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), 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**  
In a situation where server response or network load varies rapidly, the dynamic retransmission support might incorrectly increase RPC timeouts or reduce read and write transfer sizes unnecessarily. Disabling this functionality might result in increased throughput, but possibly, also increasing the visibility of the spikes due to server response or network load.

**Stability Level**  
Evolving

### 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), 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**
In a situation where server response or network load varies rapidly, the dynamic retransmission support might incorrectly increase RPC timeouts or reduce read and write transfer sizes unnecessarily. Disabling this functionality might result in increased throughput, but possibly, also increasing the visibility of the spikes due to server response or network load.

**Stability Level**
Evolving

**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 filenames that were looked up, but not found. The cache is used to avoid over the network lookup requests made for filenames that are already known to not exist.

**Data Type**
Integer (32-bit)

**Default**
1 (enabled)

**Range**
0 (disabled), 1 (enabled)

**Units**
Boolean values

**Dynamic?**
Yes

**Validation**
None

**When to Change**
In order 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 by assuming 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, then use this parameter to disable the negative cache.

**Stability Level**

Evolving

---

**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 filenames that were looked up, but were not found. The cache is used to avoid over-the-network lookup requests made for filenames that are already known to not exist.

**Data Type**

Integer (32-bit)

**Default**

1 (enabled)

**Range**

0 (disabled), 1 (enabled)

**Units**

Boolean values

**Dynamic?**

Yes

**Validation**

None

**When to Change**

In order 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 by assuming 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, then use this parameter to disable the negative cache.

### Stability Level
Evolving

### nfs:nfs_max_threads

**Description**
Controls the number of kernel threads that perform asynchronous I/O for the NFS version 2 client. Since 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 which can be executed asynchronously are read for read-ahead, readdir for readdir read-ahead, and write for putpage and pageio requests.

**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**
Change this parameter 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 to more effectively utilize the available network bandwidth and client and server resources.

**Stability Level**

Unstable

---

**nfs:nfs3_max_threads**

**Description**

Controls the number of kernel threads that perform asynchronous I/O for the NFS version 3 client. Since 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**

Change this parameter 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 to more effectively...
utilize the available network bandwidth and the client and server resources.

**Stability 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 8192 bytes of file data.

**Data Type**

Integer (32-bit)

**Default**

4

**Range**

0 to $2^{31} - 1$

**Units**

Read-ahead requests

**Dynamic?**

Yes

**Validation**

None

**When to Change**

Change this parameter 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 to more effectively utilize the available network bandwidth and the client and server resources.

**Stability 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 32,768 bytes of file data.

Data Type  Integer (32-bit)

Default  4

Range  0 to $2^{31} - 1$

Units  Read-ahead requests

Dynamic?  Yes

Validation  None

When to Change  Change this parameter 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 to more effectively utilize the available network bandwidth and the client and server resources.

Stability Level  Unstable

nfs:nrnode

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

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

The NFS client attempts to keep a minimum number of \textit{rnodes} around to attempt to avoid destroying cached data and metadata. When an \textit{rnode} is reused or freed, the cached data and metadata must be destroyed.

<table>
<thead>
<tr>
<th><strong>Data Type</strong></th>
<th>Integer (32-bit)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Default</strong></td>
<td>The default setting of this parameter is 0, which means that the value of \textit{nrnode} should be set to the value of the \textit{ncsize} parameter. Actually, any non-positive value of \textit{nrnode} results in \textit{nrnode} being set to the value of \textit{ncsize}.</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>1 to $2^{31} - 1$</td>
</tr>
<tr>
<td><strong>Units</strong></td>
<td>\textit{rnodes}</td>
</tr>
<tr>
<td><strong>Dynamic?</strong></td>
<td>No. This value can only be changed by adding or changing the parameter in the /etc/system file, and then rebooting the system.</td>
</tr>
<tr>
<td><strong>Validation</strong></td>
<td>The system enforces a maximum value such that the \textit{rnode} cache can only consume 25% of available memory.</td>
</tr>
<tr>
<td><strong>When to Change</strong></td>
<td>Since \textit{rnodes} are created and destroyed dynamically, the system tends to settle upon a \textit{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, it might be helpful to set the value of \textit{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, it might be useful to set the value of \textit{nrnode} to be a small number so that system memory can cache file data instead of \textit{rnodes}. Alternately, if the client is accessing many small files, it might be helpful to set the value of \textit{nrnode} large enough to optimize the performance.</td>
</tr>
</tbody>
</table>
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 instructs the client to
only cache 1 rnode, which means that it is
reused frequently.

Stability Level Evolving

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 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 minimum of either the size passed in
by using the getdents(2) system call or by
using NFS_MAXDATA, which is 8192 bytes.

Data Type Integer (32-bit)

Default 0 (disabled)

Range 0 (disabled), 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 are seen when trying to
read directories. Enabling this parameter might
cause a slight performance drop for applications
that read directories.
### nfs:nfs_write_error_interval

**Description**
Controls the time duration in between logging ENOSPC and EDQUOT write errors seen by the NFS client. It affects both NFS version 2 and 3 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.

**Stability Level**
Evolving

### 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. It affects messages for both NFS version 2 and 3 clients.

**Data Type**
Integer (32-bit)
Default: 0 (system console and syslog)
Range: 0 (system console and syslog), 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(1M) daemon. When this parameter is enabled, messages are printed on the system console only and are not copied to the syslog messages file.

**Stability Level**
Evolving

---

**nfs:nfs_disable_rddir_cache**

**Description**
Controls the use of a cache to hold responses from NFS version 2 readdir and NFS Version 3 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), 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 both NFS version 2 and 3 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.

**Stability Level**

Evolving

**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 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 \texttt{nfs3\_max\_transfer\_size} parameter. If larger transfers are desired, increase both parameters. If smaller transfers are desired, then just reducing this parameter should suffice.

**Stability Level**

Unstable

\texttt{nfs\_nfs\_async\_clusters}

**Description**

Controls the mix of asynchronous requests that are generated by the NFS version 2 client. There are four types of asynchronous requests, 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 operation type in favor of another.

However, 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 approximately 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 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 type to be processed before moving on to the next type. This effectively disables the fairness portion of the algorithm.

When to Change

Change this parameter to increase the number of each type of asynchronous operation that is generated before switching to the next type. This might help with server functionality that depends upon clusters of operations coming from the client.

Stability Level

Unstable

nfs:nfs3_async_clusters

Description

Controls the mix of asynchronous requests that are generated by the NFS version 3 client. There are five types of asynchronous requests, 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 operation type in favor of another.

However, 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 approximately 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 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 type to be processed before moving on to the next type. This effectively disables the fairness portion of the algorithm.

When to Change: Change this parameter to increase the number of each type of asynchronous operation that is generated before switching to the next type. This might help with server functionality that depends upon clusters of operations coming from the client.

Stability 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, and then 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 has the affect of having 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,
- Or, by making them expire more slowly, thus avoiding thread create and destroy overhead.

Stability Level

Evolving

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, but the hash queues used to index into it 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, which 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, and then hang while trying to do so.

When to Change  Examine the value of this parameter if the basic assumption of one access cache entry per file would be violated. This might be true for systems in a time sharing 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.

Stability Level  Evolving

**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 error, NFS3ERR_JUKEBOX, from a previous request. The error, NFS3ERR_JUKEBOX, is generally returned from the server when the file is temporarily unavailable for some reason. These situations are 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. The value should be increased if the delays in making the file available are long in order to reduce network overhead due to repeated retransmissions. The value can also be decreased to reduce the delay in discovering that the file has become available.

Stability Level
Evolving

\texttt{nfs:nfs3\_max\_transfer\_size}

Description
Controls the maximum size of the data portion of an NFS version 3 \texttt{READ}, \texttt{WRITE}, \texttt{READDIR}, or \texttt{READDIRPLUS} request. This parameter controls both the maximum size of request that the server returns as well as the maximum size of a 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

Validation
None. Although setting the maximum transfer size on the server to 0 will probably either 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 large might result in errors from UDP and...
communication problems between the client and the server.

**When to Change**

Change this parameter to tune the size of data being passed over the network. In general, the `nfs3_bsize` parameter should also be updated to reflect changes in this parameter. For example, when attempting to reduce the default over-the-wire transfer size to 8 kbytes, the value of both the `nfs3_max_transfer_size` and `nfs3_bsize` parameters should be changed to 8192 to avoid using multiple operations, each reading or writing 8 kbytes. Alternately, when attempting to increase the transfer size beyond 32 kbytes, then `nfs3_bsize` should also be updated to reflect the increased value, otherwise no change in the over-the-wire request size is seen.

**Stability 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 can do to attempt to enforce integrity on the part of its clients. It can check to see whether the source port from which a request was sent was a reserved port. This is a port whose number is less than 1024. For BSD based systems, these ports are reserved to processes being run by root. This checking can prevent users from writing their own RPC-based applications to defeat the access checking that the NFS client uses.

**Data Type**

Integer (32-bit)

**Default**

0 (checking disabled)
Range
0 (checking disabled), 1 (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, so interoperability problems might result if the checking is enabled.

Stability Level
Evolving

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 trick to accelerate NFS version 2 WRITEs is to take advantage of a client behavior. Clients tend to send out 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 can increase the throughput for WRITE requests tremendously.
Data Type: Integer (32-bit)
Default: 1 (clustering enabled)
Range: 0 (clustering disabled), 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, and the clustering in the NFS version 2 server might just add overhead and slow down performance instead of increasing it.

Stability Level: Evolving

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 results in requests being dropped by the NFS server because they could not be authenticated.
Most of the time, this 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 might have been avoided by ensuring that the size of the cache on the server is large enough to handle the load.

**Stability 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.

**Stability Level**
- Evolving
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, but multiple connections can be used if desired.

Data Type
Integer (32-bit)

Default
1

Range
1 to $2^{31} - 1$

Units
Connections

Dynamic?
Yes

Validation
None

When to Change
In general, 1 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 for free though. The price for increasing the number of connections is increased kernel resource usage to keep track of each of the connections.

Stability Level
Evolving
**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, if desired. You might want to close connections at a faster rate to avoid consuming system resources.

**Stability Level**
Evolving

---

**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, if desired. Close connections at a faster rate to avoid consuming system resources, if desired.
Stability Level Evolving

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 is 0, which means set the stack size to the system default.
Range 0 to $2^{31}$ - 1
Units Bytes
Dynamic? 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 Possibly, 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.
Stability Level Evolving

NFS Tunable Parameters 131
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 ones.

Validation
None

When to Change
Tune this number so that services can take advantage of client behaviors such as the clustering that accelerate NFS version 2 WRITE requests. It is possible that increasing this parameter results in the server being better able to take advantage of client behaviors.

Stability Level
Evolving

rpcmod:maxdupreqs

Description
Controls the size of the duplicate request cache that detect 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 the transaction ID. This cache avoids processing of retransmitted requests that might be non-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. It prevents the NFS server from handling non-idempotent requests.

**Validation**  
None

**When to Change**  
Examine the value of this parameter if false failures are being seen 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 connectionless transports is relatively short, starting at about 1 second and increasing to about 20 seconds.

**Stability 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 the transaction ID. This cache avoids processing of retransmitted requests that might be non-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 being seen 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.
<table>
<thead>
<tr>
<th>Stability Level</th>
<th>Unstable</th>
</tr>
</thead>
</table>

NFS Tunable Parameters 135
TCP/IP Tunable Parameters

This section describes the TCP/IP tunable parameters. For information on kernel tunables, see Chapter 2. For information on NFS tunables, see Chapter 3.

- “IP Tunable Parameters” on page 138
- “TCP Tunable Parameters” on page 144
- “UDP Tunable Parameters” on page 159
- “Per-Route Metrics” on page 160

Overview of Tuning TCP/IP Parameters

You can set all of the tuning parameters described in this chapter with the ndd command, except for the following two parameters that can only be set in the /etc/system file:

- “tcp_conn_hash_size” on page 153
- “ipc_tcp_conn_hash_size” on page 154

Use the following syntax to set TCP/IP parameters with the ndd command.

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

For example, the following ndd command disables IP forwarding.
† nedd -set /dev/ip ip_forwarding 0

See nedd(1M) for more information.

To set a TCP/IP parameter across system reboots, include the appropriate nedd command in a system startup script. Use the following guidelines to create a system startup script to include nedd commands:

- Create a script in the /etc/init.d directory and create links to it in the /etc/rc2.d, /etc/rc1.d, and /etc/rcS.d directories.
- The script should run between the existing S69inet and S72inetsvc scripts.
- Name the script with the S70 or S71 prefix. Scripts with the same prefix are run in some sequential way so it doesn’t matter if there is more than one script with the same prefix.
- See the README file in the /etc/init.d directory for more information on naming run control scripts.

See “Run Control Scripts” in System Administration Guide, Volume 1 for more information on creating a startup script.

TCP/IP Parameter Validation

All of the TCP/IP parameters described in this section are checked to verify they fall in the parameter range, which is provided in each tunable section, except for the two parameters that can be set only in the /etc/system file described above. See the validation section for “tcp_conn_hash_size” on page 153 and “ipc_tcp_conn_hash_size” on page 154 for more information.

Internet Request for Comments (RFCs)

Internet protocol and standard specifications are described in RFC documents. You can get copies of RFCs by using anonymous ftp to the sri-nic.arpa machine. Browse RFC topics by viewing the rfc-index.txt file at this site.

IP Tunable Parameters

This section describes some of the IP tunable parameters.
**ip_icmp_err_interval** and **ip_icmp_err_burst**

**Description**
Control 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. This parameter protects IP from denial of service attacks. Set ip_icmp_err_interval to 0 to disable IP to generate IPv4 or IPv6 ICMP error messages.

**Default**
100 milliseconds for ip_icmp_err_interval
10 for ip_icmp_err_burst

**Range**
0 - 99,999 milliseconds for
ip_icmp_err_interval
1 - 99,999 for ip_icmp_err_burst

**Dynamic?**
Yes

**When to Change**
Change the parameter values if you need a higher error message generation rate for diagnostic purposes.

**Commitment Level**
Unstable

---

**ip_forwarding** and **ip6_forwarding**

**Description**
Control whether IP does IPv4 or IPv6 forwarding between interfaces. See also xxx:ip_forwarding below.

**Default**
0 (disabled)

**Range**
0 (disabled), 1 (enabled)

**Dynamic?**
Yes

**When to Change**
If IP forwarding is needed, enable it.

**Commitment Level**
Unstable
**xxx:ip_forwarding**

**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`. Their corresponding parameter names are:

- `hme0:ip_forwarding`
- `hme1:ip_forwarding`

**Default**

0 (disabled)

**Range**

0 (disabled), 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

---

**ip_respond_to_echo_broadcast and ip6_respond_to_echo_multicast**

**Description**

Control whether IPv4 or IPv6 responds to broadcast ICMPv4 echo request or multicast ICMPv6 echo request.

**Default**

1 (enabled)

**Range**

0 (disabled), 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**  
Control whether IPv4 or IPv6 sends out ICMPv4 or ICMPv6 redirect messages. See also “ip_forwarding and ip6_forwarding” on page 139.

**Default**  
1 (enabled)

**Range**  
0 (disabled), 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**  
Control whether IPv4 or IPv6 forwards packets with source IPv4 routing options or IPv6 routing headers. See also “ip_forwarding and ip6_forwarding” on page 139.

**Default**  
1 (enabled)

**Range**  
0 (disabled), 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: 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, increase the value, but 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: Determine 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 3.3.2.4.

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 variable to 1.

Commitment Level: Unstable
IP Tunable Parameters With Additional Cautions

Changing the following parameters is not recommended unless there are extenuating circumstances that are described with each parameter.

**ip_ire_pathmtu_interval**

**Description**
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 causes the error message. This parameter controls how many extra bytes of the packet beyond the IPv4 or IPv6 header to be 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.
## TCP Tunable Parameters

### tcp_deferred_ack_interval

<table>
<thead>
<tr>
<th>Description</th>
<th>The time-out value for TCP delayed acknowledgment (ACK) timer in milliseconds. Refer to RFC 1122, 4.2.3.2.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>100 milliseconds</td>
</tr>
<tr>
<td>Range</td>
<td>1 millisecond to 1 minute</td>
</tr>
<tr>
<td>Dynamic?</td>
<td>Yes</td>
</tr>
<tr>
<td>When to Change</td>
<td>Do not increase this value to more than 500 milliseconds. If in some circumstances, slow network links (less than 57.6 Kbps) with greater than 512 bytes maximum segment size (MSS) when the interval is short for receiving more than one TCP segment, increase the value.</td>
</tr>
</tbody>
</table>

### tcp_deferred_acks_max

<table>
<thead>
<tr>
<th>Description</th>
<th>The maximum number of TCP segments (in units of maximum segment size MSS for individual connections) received before an acknowledgment (ACK) is generated. If set to 0 or 1, it means no delayed ACKs, assuming all segments are 1 MSS long. Note that for remote destinations (not directly connected), the maximum number is fixed to 2, no matter what this parameter is set to. The actual number is dynamically calculated for each connection. The value is the default maximum.</th>
</tr>
</thead>
</table>

Commitment Level: Unstable
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Range</th>
<th>Dynamic?</th>
<th>When to Change</th>
<th>Commitment Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>tcp_wscale_always</td>
<td>8</td>
<td>0 to 16</td>
<td>Yes</td>
<td>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.</td>
<td>Unstable</td>
</tr>
</tbody>
</table>

**tcp_wscale_always**

**Description**
If set to 1, TCP always sends 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, and 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), 1 (enabled)

**Dynamic?**
Yes

**When to Change**
If you want the window scale option in a high-speed network configuration, enable it.

**Commitment Level**
Unstable

**tcp_tstamp_always**

**Description**
If set to 1, TCP always sends 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), 1 (enabled)
Dynamic? Yes
When to Change In summary, if an accurate measurement of round trip time (RTT) and TCP sequence number wraparound is a problem, enable it.
Refer to RFC 1323 for more reasons to enable this option.
Commitment Level Unstable

tcp_xmit_hiwat

Description The default send window size in bytes. Refer to the following discussion of per-route metrics for setting a different value on a per route basis. See “tcp_max_buf” on page 147 also.
Default 16,384 bytes
Range 4096 to 1,073,741,824
Dynamic? Yes
When to Change Note that this is the default value. An application can use setsockopt(3SOCKET) SO_SNDBUF to change the individual connection’s send buffer.
Commitment Level Unstable

tcp_recv_hiwat

Description The default receive window size in bytes. Refer to the following discussion of per-route metrics for setting a different value on a per-route basis.
See “tcp_recv_hiwat_minmss” on page 158 and “tcp_max_buf” on page 147 also.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Range</th>
<th>Dynamic?</th>
<th>When to Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>24,576</td>
<td>2048 to 1,073,741,824</td>
<td>Yes</td>
<td>Note that this is the default value. An application can use setsockopt(3SOCKET) SO_RCVBUF to change the individual connection’s receive buffer.</td>
<td></td>
</tr>
<tr>
<td>Commitment Level</td>
<td>Unstable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tcp_max_buf</td>
<td>The maximum buffer size in bytes. It controls how large the send and receive buffers are set to by an application using setsockopt(3SOCKET).</td>
<td>1,048,576</td>
<td>8192 to 1,073,741,824</td>
<td>Yes</td>
<td>If TCP connections are being made in a high-speed network environment, increase the value to match the network link speed.</td>
</tr>
<tr>
<td>Commitment Level</td>
<td>Unstable</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tcp_cwnd_max</td>
<td>The maximum value of TCP congestion window (cwnd) in bytes. Refer to RFC 1122 and RFC 2581 for more information on TCP congestion window.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Default: 1,048,576

Range: 128 to 1,073,741,824

Dynamic?: Yes

When to Change: This is the maximum value a TCP cwnd can grow to. Note that even if an application uses setsockopt(3SOCKET) to change the window size to a value higher than \texttt{tcp\_cwnd\_max}, the actual window used can never grow beyond \texttt{tcp\_cwnd\_max}. Thus, \texttt{tcp\_max\_buf} should be greater than \texttt{tcp\_cwnd\_max} in general.

Commitment Level: Unstable

\texttt{tcp\_slow\_start\_initial}

Description: The maximum initial congestion window (cwnd) size in MSS of a TCP connection.

Refer to RFC 2414 on how initial congestion window size is calculated.

When to Change: This is the maximum value a TCP cwnd can grow to. Note that even if an application uses setsockopt(3SOCKET) to change the window size to a value higher than \texttt{tcp\_cwnd\_max}, the actual window used can never grow beyond \texttt{tcp\_cwnd\_max}. Thus, \texttt{tcp\_max\_buf} should be greater than \texttt{tcp\_cwnd\_max} in general.

Commitment Level: Unstable

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 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 for the calculation.

Default
4

Range
1 to 16,384

Dynamic?
Yes

When to Change
See “tcp_slow_start_initial” on page 148 for more information.

Commitment Level
Unstable

tcp_sack_permitted

Description
If set to 2, TCP always sends 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 or not.

Refer to RFC 2018 for information on the SACK option.

Default
2 (active enabled)

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

Dynamic?
Yes

When to Change
SACK processing can improve TCP retransmission performance so it should be actively enabled. If, in some circumstances, the other side can be confused with the SACK option
actively enabled, 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), 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
The time in milliseconds a TCP connection stays
in TIME-WAIT state.

Refer to RFC 1122, 4.2.2.13 for more information.

Default
4 minutes

Range
1 second to 10 minutes

Dynamic?
Yes

When to Change
On a busy web server, there can be too many
TCP connections in TIME-WAIT state, consuming
too much memory. In this situation, you can
decrease the value for performance reasons. Do
not set the value lower than 60 seconds.

Refer to RFC 1122, 4.2.2.13 for more information.
Commitment Level: Unstable

**tcp_conn_req_max_q**

**Description**

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_q” on page 152.

**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. And if an application is not fast enough to handle that many connection requests in a timely fashion 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. This means that 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
**tcp_conn_req_max_q0**

**Description**
The default maximum number of incomplete (three-way handshake not yet finished) pending TCP connections for a TCP listener.

Refer to RFC 793 for more information on TCP three-way handshake. See also “tcp_conn_req_max_q” on page 151.

**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 \((N)\) 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.

**Commitment Level**
Unstable

**tcp_conn_req_min**

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

**When to Change**
This 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 Parameters Set in the `/etc/system` File**
These parameters can be set only in the `/etc/system` file. After the file is modified, reboot the system.

The following entry sets `tcp_conn_hash_size`:

```
set tcp:tcp_conn_hash_size=1024
```

**`tcp_conn_hash_size`**

- **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.

<table>
<thead>
<tr>
<th>Commitment Level</th>
<th>Unstable</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ipc_tcp_conn_hash_size</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Controls the hash table size in an IP module for all active (in ESTABLISHED state) TCP connections.</td>
</tr>
<tr>
<td><strong>Data Type</strong></td>
<td>Unsigned integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>512</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>512 to 2,147,483,648</td>
</tr>
<tr>
<td><strong>Implicit</strong></td>
<td>It should be a power of two.</td>
</tr>
<tr>
<td><strong>Dynamic?</strong></td>
<td>No. This parameter can only be changed at boot time.</td>
</tr>
<tr>
<td><strong>Validation</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>When to Change</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>Commitment Level</strong></td>
<td>Unstable</td>
</tr>
</tbody>
</table>

154 Solaris Tunable Parameters Reference Manual • January 2001
**TCP Parameters With Additional Cautions**

Changing the following parameters is not recommended unless there are extenuating circumstances that are described with each parameter.

**tcp_ip_abort_interval**

**Description**  
The default total retransmission timeout value for a TCP connection in milliseconds. 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 156.

**Default**  
8 minutes

**Range**  
500 millisecond to 1193 hours

**Dynamic?**  
Yes

**When to Change**  
Do not change this value. See “tcp_rexmit_interval_max” on page 156 for exceptions.

**Commitment Level**  
Unstable

**tcp_rexmit_interval_initial**

**Description**  
The default initial retransmission timeout (RTO) value for a TCP connection in milliseconds. Refer to the following discussion of per route metrics for 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  The default maximum retransmission timeout value (RTO) in milliseconds. The calculated RTO for all TCP connections cannot exceed this value. See also "tcp_ip_abort_interval" on page 155.

Default  60 seconds

Range  1 millisecond to 20 seconds

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 in the order of 10 seconds, you can change the value to a higher value. If you change this value, you should also change the tcp_ip_abort_interval parameter to match it. 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  The default minimum retransmission timeout (RTO) value in milliseconds. The calculated RTO for all TCP connections cannot be lower than this value. See also "tcp_rexmit_interval_max" on page 156.

Default  400 milliseconds

Range  1 millisecond to 20 seconds
TCP/IP Tunable Parameters

<table>
<thead>
<tr>
<th>Dynamic?</th>
<th>Yes</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>When to Change</strong></td>
<td>Do not change the value in a normal network environment. TCP’s RTO calculation should be able to cope with most RTT fluctuations. If in some very special circumstances such that the round trip time (RTT) for a connection is in the order of 10 seconds, change to a higher value. If you change this value, you should change the <code>tcp_rexmit_interval_max</code> parameter to match it. You should change the value of <code>tcp_rexmit_interval_max</code> to at least eight times the value of <code>tcp_rexmit_interval_min</code>.</td>
</tr>
<tr>
<td><strong>Commitment Level</strong></td>
<td>Unstable</td>
</tr>
</tbody>
</table>

**tcp_rexmit_interval_extra**

| Description | A constant added to the calculated retransmission time-out value (RTO) in milliseconds. |
| 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 in some circumstances, 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. |

TCP/IP Tunable Parameters 157
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Range</th>
<th>Dynamic?</th>
<th>When to Change</th>
<th>Commitment Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>tcp_recv_hiwat_minmss</td>
<td>4</td>
<td>1 to 65,536</td>
<td>Yes</td>
<td>Do not change the value. If changing it is necessary, do not change the value lower than 4.</td>
<td>Unstable</td>
</tr>
<tr>
<td>Description</td>
<td>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.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>tcp_compression_enabled</td>
<td>1 (enabled)</td>
<td>0 (disabled), 1 (enabled)</td>
<td>Yes</td>
<td>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.</td>
<td>Unstable</td>
</tr>
<tr>
<td>Description</td>
<td>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 150 also.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
UDP Tunable Parameters

This section describes some of the UDP tunable parameters.

udp_xmit_hiwat

**Description**
The default maximum UDP socket datagram size in bytes. See “udp_max_buf” on page 160 for more information.

**Default**
8192 bytes

**Range**
4096 to 65,536

**Dynamic?**
Yes

**When to Change**
Note that an application can use `setsockopt(3SOCKET) 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**
The default maximum UDP socket receive buffer size in bytes. See “udp_max_buf” on page 160 for more information.

**Default**
8192 bytes

**Range**
4096 to 65,536

**Dynamic?**
Yes
When to Change

Note that an application can use `setsockopt(3SOCKET) 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 unless there are extenuating circumstances that are described with each parameter.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Range</th>
<th>Dynamic?</th>
<th>When to Change</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>udp_max_buf</code></td>
<td>Controls how large send and receive buffers (in bytes) can be for a UDP socket.</td>
<td>262,144 bytes</td>
<td>65,536 to 1,073,741,824</td>
<td>Yes</td>
<td>Do not change the value. If this parameter is set to a very large value, UDP socket applications can consume too much memory.</td>
</tr>
</tbody>
</table>

Commitment Level

Unstable

Per-Route Metrics

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

For example, a system has two different network interfaces, fast ethernet interface and 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 `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.

Assuming IPv4, the following is in the routing table (\texttt{netstat -rn}).

\begin{verbatim}
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
\end{verbatim}

Do the following:

\begin{verbatim}
# route change -net 192.123.124.0 -recvpipe x
\end{verbatim}

This means 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 there is no specific routing entry for that network, you can add a prefix route to that network and change the metric. For example:

\begin{verbatim}
# 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
\end{verbatim}

Note that the prefix route's gateway is the default router. Then all connections going to that network use receive buffer size $y$. If you have more than one interface, use the \texttt{-ifp} argument to specify which interface to use. This way, you can control which interface to use for specific destinations. Use the \texttt{route(IM)} \texttt{get} command to verify the metric.
System Facility Parameters

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

- “cron” on page 164
- “devfsadm” on page 164
- “dhcpcagent” on page 164
- “fs” on page 164
- “inetinit” on page 164
- “init” on page 164
- “kbd” on page 165
- “login” on page 165
- “nfslogd” on page 165
- “passwd” on page 165
- “power” on page 165
- “su” on page 165
- “sys-suspend” on page 165
- “tar” on page 166
- “utmpd” on page 166

System Default Parameters

The functioning of various system facilities is governed by a set of values that are read by the 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.

cron
See cron(1M), the "Setting cron Defaults" section for more information.

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.
See dhcpagent(1M), the "/etc/default/dhcpagent" section for more information.

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. See default_fs(4) for more information.

inetinit
Used by the /etc/rc2.d/S69inet script to control the sequence numbers used by TCP.

init
See init(1M), the "/etc/default/init" section for more information.
The CMASK variable referred to in the file is not documented in the man page. CMASK is the umask that init uses and that every process inherits from the init process. If not set, init uses the default umask it obtains from the kernel. The init process
always attempt to apply a umask of 022 before creating any files, regardless of the setting of CMASK. 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.

**kbd**
See kbd(1), the "Extended Description" section for more information.

**login**
See login(1), "/etc/default/login" in the FILES section for more information.

**nfslogd**
See nfslogd(1M), the "Description" section for more information.

**passwd**
See passwd(1), "/etc/default/passwd" in the FILES section for more information.

**power**
See pmconfig(1M), "/etc/default/power" in the FILES section for more information.

**su**
See su(1M), "/etc/default/su" in the FILES section for more information.

**sys-suspend**
See sys-suspend(1M), "/etc/default/sys-suspend" in the FILES section for more information.
tar

See tar(1) for description of the −f function modifier.

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 archiveN string is looked up in the /etc/default/tar file. The value of the archiveN string is used as the output device with the blocking and size specifications from the file.

For example:

   % tar -c 2 /tmp/*

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 versions of Solaris) 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.
APPENDIX A

Tunable Parameter Change History

This section describes the change history of specific parameters. Parameters whose functionality has been removed are listed also.

- “Process Sizing Tunables” on page 167
- “Paging Related Tunables” on page 169
- “General Kernel Variables” on page 171
- “General I/O” on page 172
- “Pseudo Terminals” on page 174
- “Sun4u Specific” on page 174
- “Parameters With No Functionality” on page 175

Kernel Parameters

Process Sizing Tunables

maxusers (Solaris 7 Release)

<table>
<thead>
<tr>
<th>Description</th>
<th>The maxusers parameter drives max_nprocs and maxuprc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Type</td>
<td>Signed integer</td>
</tr>
<tr>
<td>Default</td>
<td>Lesser of the amount of memory in Mbytes and 1024</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>1 to 2048</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------</td>
</tr>
</tbody>
</table>

**Note** - Values greater than 1024 must be specified in `/etc/system`. If a value greater than 2048 is provided, calculations clamps the value at 2048, but later processing sets the value to the provided value.

<table>
<thead>
<tr>
<th><strong>Units</strong></th>
<th>Users</th>
</tr>
</thead>
</table>

**Dynamic?**  No. After computation of dependent variables is done, `maxusers` is never referenced again.

**Validation** None

**When to Change** If the default number of user processes derived by the system is insufficient. This insufficiency is seen by the following messages on the system console or messages file.

```
out of processes
```

**Commitment Level** Unstable

---

max_nprocs (Pre-Solaris 8 Releases)

**Description** Maximum number of processes that can be created on a system. Includes system and user processes. Prior to the Solaris 8 release, the value was determined by computation and then used in the setting of `maxuprc`.

This value is also used in determining the size of several other system data structures. For releases prior to Solaris 8, if a value is provided in `/etc/system` it is used rather than the computed value. Other data structures where this variable plays a role are:

- 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 the sun4d, sun4m, and Intel platforms

**Data Type**
Signed integer

**Default**
$10 + (16 \times \text{maxusers})$

**Range**
266 to value of pidmax

**Dynamic?**
No. \(\text{max_nprocs}\) is assigned to the \(v\_\text{proc}\) element of the \(v\) structure after the initial parameter calculation is completed. Changing \(v\_v\_\text{proc}\) on a running system almost certainly results in a system crash or silent data corruption.

**Validation**
Compared to \(\text{maxpid}\) and set to \(\text{maxpid}\), if larger. On the sun4d and Intel platforms, an additional check is made against a platform-specific value. \(\text{max_nprocs}\) is set to the smallest value in the triplet \((\text{max_nprocs}, \text{maxpid}, \text{platform value})\). Both platforms use 65,534 as the platform value.

**When to Change**
Starting with the Solaris 8 release, this value can be changed to enable more than 30,000 processes on a system. Changing this parameter is one of the steps necessary to enable support for more than 30,000 processes on a system.

**Commitment Level**
Unstable

**Paging Related Tunables**
In certain revisions of the Solaris 2.6 kernel patch (105181-10 for SPARC platforms and 105182-09 for Intel platforms) and in the Solaris 7 release, a new parameter is introduced: **priority paging**. A new starting point for pageout thread activity (\(\text{cachefree}\)) is also used. When available memory is between \(\text{cachefree}\) and \(\text{lotsfree}\), priority paging modifies the page-checking algorithm to skip the page, if it came from an executable (text, stack, or data). After memory falls below \(\text{lotsfree}\), every page is considered equally. The facility is not enabled by default,
but can be enabled by either setting `cachefree` to a value greater than `lotsfree` or by setting the `priority_paging` variable to a non-zero value, which sets `cachefree` to 2 times `lotsfree`.

**priority_paging (Solaris 2.6 and 7 Releases)**

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Enables priority paging feature. When set, this variable sets <code>cachefree</code> to 2 times <code>lotsfree</code>, thereby enabling priority paging.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Type</strong></td>
<td>Signed integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>0 (priority paging disabled unless <code>cachefree</code> set separately) or 1 (enabled)</td>
</tr>
<tr>
<td><strong>Units</strong></td>
<td>Toggle (on/off)</td>
</tr>
<tr>
<td><strong>Dynamic?</strong></td>
<td>No. Sets the value of <code>cachefree</code> at boot time only. Runtime enabling can be achieved by setting <code>cachefree</code> with <code>adb</code> while the system is running.</td>
</tr>
<tr>
<td><strong>Validation</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>When to Change</strong></td>
<td>Should always be enabled unless the system is tight on memory, and does excessive I/O where the contents of the files are needed in the future.</td>
</tr>
<tr>
<td><strong>Commitment Level</strong></td>
<td>Obsolete</td>
</tr>
</tbody>
</table>

**cachefree (Solaris 2.6 and Solaris 7 Releases)**

<table>
<thead>
<tr>
<th><strong>Description</strong></th>
<th>Enables priority paging feature, provided <code>cachefree</code> is greater than <code>lotsfree</code>. This variable is available for systems running the Solaris 2.6 release, with at a minimum, revision 10 of patch 105181 installed, and for systems running the Solaris 7 release. By default, this feature (cachefree equals lotsfree) is disabled.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Type</strong></td>
<td>Unsigned long</td>
</tr>
</tbody>
</table>
Default Value of lotsfree unless priority_paging is set, which means cachefree is 2 times lotsfree

Range lotsfree to physical memory on system

Units Pages

Dynamic? Yes

Validation If less than lotsfree, it is reset to the value of lotsfree.

When to Change Should always be enabled unless the system is tight on memory, and does excessive I/O where the contents of the files are needed in the future.

Commitment Level Obsolete

General Kernel Variables

noexec_user_stack (Solaris 2.6 and Solaris 7 Releases)

Description Introduced in the Solaris 2.6 release to allow the stack to be marked as non-executable. This helps make buffer-overflow attacks more difficult.

In the Solaris 2.6 release, the value does not affect threaded applications. All 64-bit Solaris applications effectively make all stacks non-executable irrespective of the setting of this variable.

Note - This variable exists on all systems running Solaris 2.6 or later, but it is only effective on sun4u, sun4m, and sun4d architectures.

Data Type Signed integer

Default 0 (disabled)

Range 0 (disabled), 1 (enabled)

Units Toggle (on/off)
### Dynamic?
Yes. Does not affect currently running processes—only those 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(2)` to make the stack executable.

### Commitment Level
Unstable

## General I/O

### rlim_fd_cur (Solaris 7 Release and Earlier)

**Description**
"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 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**
64

**Range**
1 to MAXINT

**Units**
File descriptors

**Dynamic?**
No. Loaded into `rlimits` structure.

**Validation**
Compared to `rlim_fd_max` and 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 is possibly not necessary for a
program to use `setrlimit(2)` to increase the maximum number of file descriptors available to it.

Commitment Level Unstable

**segkpsize (Solaris 7 and Earlier Releases)**

**Description** Specify 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 64-bit kernels. 64-bit kernels use a default stack size of 24 Kbytes.

Available for the Solaris 7 release with patch 106541-04 or the Solaris 7 5/99 and Solaris 8 releases.

**Data Type** Unsigned long

**Default** 64-bit kernels, 2 Gbytes
32-bit kernels, 512 Mbytes

**Range** 64-bit kernels, 512 Mbytes - 24 Gbytes
32-bit kernels, 512 Mbytes

**Units** Mbytes

**Dynamic?** No

**Validation** None

**When to Change** Increase when more threads are desired.

Commitment Level Unstable

**Tunable Parameter Change History** 173
Pseudo Terminals

**pt_cnt (Solaris 7 and Earlier Releases)**

**Description**
Number of /dev/pts (the pseudo terminal devices used by telnet or rlogin for network logins) entries to create on a reconfiguration boot. This parameter effectively limits the number of users that can simultaneously be logged in across the net to the value of pt_cnt. You must do a reconfiguration boot (boot -r) after making the change to the /etc/system file for the additional device nodes to be created.

**Data Type**
Signed integer

**Default**
48

**Range**
0 to maxpid

**Units**
logins/windows

**Dynamic?**
No

**Validation**
None. Excessively large values hang the system.

**When to Change**
When the desired number of users cannot log in to the system.

**Commitment Level**
Unstable

Sun4u Specific

**enable_grp_ism (Solaris 2.6 Release)**

**Description**
Enables a shared memory Translation Setaside Buffer (TSB) capability for System V Shared Memory that has been attached with the SHARE_MMU flag set. This parameter is available in, at minimum, patch 105181-05 for the Solaris 2.6 release. Starting with the Solaris 7 release, the
parameter name has been removed, but the system implements this parameter by default.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Type</strong></td>
<td>Signed integer</td>
</tr>
<tr>
<td><strong>Default</strong></td>
<td>0</td>
</tr>
<tr>
<td><strong>Range</strong></td>
<td>0 (disabled) or 1 (enabled)</td>
</tr>
<tr>
<td><strong>Dynamic?</strong></td>
<td>No</td>
</tr>
<tr>
<td><strong>Validation</strong></td>
<td>None</td>
</tr>
<tr>
<td><strong>When to Change</strong></td>
<td>Turn on when using System V Shared Memory attached with the SHARE_MMU flag set.</td>
</tr>
<tr>
<td><strong>Commitment Level</strong></td>
<td>Unstable</td>
</tr>
</tbody>
</table>

---

### Parameters With No Functionality

The following section describes parameters whose functionality has been removed, but the parameter might still be available for compatibility reasons. These parameters are ignored if they are set.

#### Paging-Related Tunables

**tune_t_gpgslo**

**Description**

Obsolete. Variable left in place for compatibility reasons.

**tune_t_minasmem**

**Description**

Obsolete. Variable left in place for compatibility reasons.
System V Message Parameters

msgsys:msginfo_msgssz

Description
Specifications size of chunks system uses to manage space for message buffers. Obsolete since the Solaris 8 release.

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

Description
Number of messages the system supports. Obsolete since the Solaris 8 release.

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 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_msgseg

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. Obsolete as of the Solaris 8 release.

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

System V Semaphore Parameters

semsys:seminfo_semmap
Obsolete. Variable is present in kernel for compatibility reasons but is no longer used.

semsys:seminfo_semusz
Obsolete. Any values entered are ignored.

NFS Module Parameters

nfs:nfs_32_time_ok
Obsolete as of the Solaris 8 release.

nfs:nfs_acl_cache
Obsolete as of the Solaris 2.6 release.
Revision History for this Manual

This section describes the revision history for this manual.

Current Version—Solaris 8 01/01 Release

The current version of this manual applies to the Solaris 8 01/01 release.

Changes From Previous Release

shmsys:shminfo_shmmin (Solaris 8 Update 3)

Changed the following section:

<table>
<thead>
<tr>
<th>When to Change</th>
<th>No known reason.</th>
</tr>
</thead>
<tbody>
<tr>
<td>To:</td>
<td>Not recommended. System programs such as powerd might fail if this value is too large. Programs attempting to create a section smaller than the value of shminfo_shmmin will see an EINVAL error when attempting to create the segment and generally, will exit.</td>
</tr>
</tbody>
</table>
See “shmsys:shminfo_shmmin” on page 94 for more information.

**semsys:seminfo_semmnu (Solaris 8 Update 3)**

Added this parameter because it was left out inadvertently.

See “semsys:seminfo_semmnu” on page 91 for more information.
## Index

### A
- autoup 33

### B
- bufhwm 71

### C
- cachefree 49, 170
- consistent_coloring 99
- CRON 164

### D
- desfree 44
- dhcpagent 164
- dnlc_dir_enable 69
- dnlc_dir_max_size 71
- dnlc_dir_min_size 70
- doiflush 35
- dopageflush 35

### E
- enable_grp_ism 175

### F
- fastscan 51
- fs 164
- fsflush 32

### H
- handspreadpages 54
- hires_tick 98

### I
- inetinit 164
- init 164
- ip6_forwarding 139
- ip6_forward_src_routed 141
- ip6_icmp_return_data_bytes 143
- ip6_respond_to_echo_multicast 140
- ip6_send_redirects 141
- ip6_strict_dst_multihoming 142
- ipc_tcp_conn_hash_size 154
- ip_addrs_per_if 142
- ip_forwarding 139
- ip_forward_src_routed 141
- ip_icmp_err_burst 139
- ip_icmp_err_interval 139
- ip_icmp_return_data_bytes 143
- ip_ire_pathmtu_interval 143
- ip_respond_to_echo_broadcast 140
- ip_send_redirects 141
- ip_strict_dst_multihoming 142

### K
- kbd 165
- kmem_flags 59
L
login 165
lotsfree 43
lwp_default_stksize 31

M
maxpgio 55
maxphys 63
maxpid 39
maxuprc 41
maxusers 37, 167
max_nprocs 40, 168
minfree 45
min_percent_cpu 53
moddebug 62
msgsys:msginfo_msgmap 176
msgsys:msginfo_msgmax 85
msgsys:msginfo_msgmnb 85
msgsys:msginfo_msgmni 86
msgsys:msginfo_msgseg 177
msgsys:msginfo_msgssz 176
msgsys:msginfo_msgtql 87

N
ncsize 66
ndd 137
ndquot 73
nfs:nacache 122
nfs:nfs3_async_clusters 120
nfs:nfs3_bsize 118
nfs:nfs3_cots_timeo 104
nfs:nfs3_do_symlink_cache 106
nfs:nfs3_dynamic 107
nfs:nfs3_jukebox_delay 123
nfs:nfs3_lookup_neg_cache 109
nfs:nfs3_max_threads 111
nfs:nfs3_max_transfer_size 124
nfs:nfs3_nra 113
nfs:nfs3_pathconf_disable_cache 102
nfs:nfs_allow_preepoch_time 103
nfs:nfs_async_clusters 119
nfs:nfs_async_timeout 121
nfs:nfs_cots_timeo 104
nfs:nfs_disable_rddir_cache 117
nfs:nfs_do_symlink_cache 105
nfs:nfs_dynamic 107
nfs:nfs_lookup_neg_cache 108
nfs:nfs_nra 112
nfs:nfs_shrinkreaddir 115
nfs:nfs_write_error_interval 116
nfs:nfs_write_error_to_cons_only 116
nfs:nrmode 114
nfssrv:exi_cache_time 128
nfssrv:nfsauth_ch_cache_max 127
nfssrv:nfs_portmon 125
nfssrv:nfs_write_async 127
nfs_32_time_ok 178
nfs_acl_cache 178
nfs_max_threads 110
noexec_user_stack 58, 171
nstrpush 82

P
pageout_reserve 48
pages_before_pager 54
pages_pp_maximum 50
passwd 165
physmem 30
pidmax 39
power 165
priority_paging 49, 170
pt_cnt 81, 174
pt_max_pty 82
pt_pctofmem 81

R
rechoose_interval 97
reserved_procs 38
rlim_fd_cur 65, 172
rlim_fd_max 64
rpcmod:clnt_idle_timeout 130
rpcmod:clnt_max_conns 129
rpcmod:csmaxdupreqs 134
rpcmod:maxdupreqs 133
rpcmod:svc_default_max_same_xprt 132
rpcmod:svc_default_stksize 131
rpcmod:svc_idle_timeout 130
rstchown 67

S
segkpsize 173
segspt_minfree 96
semsys:seminfo_semaem 92
semsys:seminfo_semmmap 178
semsys:seminfo_semmnu 88
semsys:seminfo_semmnu 91
semsys:seminfo_semmnu 89
semsys:seminfo_semopm 90
semsys:seminfo_semume 92
semsys:seminfo_semusz 178
semsys:seminfo_semvmx 89
shmsys:shminfo_shmmax 94
shmsys:shminfo_shmmin 94
shmsys:shminfo_shmmni 95
shmsys:shminfo_shmseg 95
slowscan 52
strmsgsz 83
su 165
swapfs_minfree 57
swapfs_reserve 56
sys-suspend 165

T

tar 166
tcp_compression_enabled 158
tcp_conn_hash_size 153
tcp_conn_req_max_q 151
tcp_conn_req_max_q0 152
tcp_conn_req_min 153
tcp_cwnd_max 147
tcp_deferred_acks_max 144
tcp_deferred_ack_interval 144
tcp_ip_abort_interval 155
tcp_max_buf 147
tcp_recv_hiwat 147
tcp_recv_hiwat_minmss 158
tcp_rev_src_routes 150
tcp_rexmit_interval_extra 157
tcp_rexmit_interval_initial 155
tcp_rexmit_interval_max 156
tcp_rexmit_interval_min 156
tcp_sack_permitted 149
tcp_slow_start_after_idle 149
tcp_slow_start_initial 148
tcp_time_wait_interval 150
tcp_tstamp_always 146
tcp_tstamp_if_wscale 158
tcp_wsacle_always 145
tcp_xmit_hiwat 146
throttlefree 46
timer_max 98
tmpfs_maxkmem 78
tmpfs_minfree 79
tune_t_fsflushr 33
tune_t_gpgslo 175
tune_t_minarmem 51
tune_t_minasmem 176

U

udp_max_buf 160
udp_recv_hiwat 159
udp_xmit_hiwat 159
ufs:ufs_HW 77
ufs:ufs_LW 77
ufs:ufs_WRITES 76
ufs_ninode 74
utmpd 166

X

xxx:ip_forwarding 140