# Tunable Kernel Parameters

## HP-UX Release 11i

### Update 1.1

Customer Order Number: ONLINE ONLY

HEWLETT®
PACKARD

**Manufacturing Part Number: TKP-90202**
**U0202**

Printed in United States of America

# Legal Notices

The information in this document is subject to change without notice.

*Hewlett-Packard makes no warranty of any kind with regard to this manual, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.* Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Please read the enclosed Hewlett-Packard Software Product License Agreement and Limited Warranty before operating this product. Rights in the software are offered only on the condition that the customer accepts all terms and conditions of the License Agreement.

Operating the product indicates your acceptance of these terms and conditions. If you do not agree to the License Agreement, you may return the unused product for a full refund.

A copy of the specific warranty terms applicable to your Hewlett-Packard product and replacement parts can be obtained from your local Sales and Service Office.

## Warranty

A copy of the specific warranty terms applicable to your Hewlett-Packard product and replacement parts can be obtained from your local Sales and Service Office.

## Restricted Rights Legend

Use, duplication or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 for DOD agencies, and subparagraphs (c) (1) and (c) (2) of the Commercial Computer Software Restricted Rights clause at FAR 52.227-19 for other agencies.

HEWLETT-PACKARD COMPANY
3000 Hanover Street
Palo Alto, California 94304 U.S.A.

Use of this document and any supporting software media (CD-ROMs, flexible disk, and tape cartridges) supplied for this pack is restricted to this product only. Additional copies of the programs may be made for security and back-up purposes only. Resale of the programs, in their present form or with alterations, is expressly prohibited.

## Copyright Notices

### Trademark Notices

## Revision History

This guide’s printing date and part number indicate its current edition. The printing date changes when a new edition is printed. (Minor corrections and updates which are incorporated at reprint do not cause the date to change.) The part number changes when extensive technical changes are incorporated.

New editions of this manual incorporate all material updated since the previous edition.

| | |
|---|---|
| Edition 1 | December 2001. Part number TKP-91201. Online only. HP-UX Release 11i. |
| Update 1.1 | February 2002. Part number TKP-90202. Online only. HP-UX Release 11i. |

# Conventions

We use the following typographical conventions.

*audit* (5) An HP-UX manpage. On docs.hp.com and on the Instant Information CD, it may be a hot link to the manpage itself.

For example, *audit* is the name and *5* is the section in the *HP-UX Reference*. From the HP-UX command line, you can enter “`man audit`” or “`man 5 audit`” to view the manpage. See *man* (1).

*Book Title* The title of a book. On the web and on the Instant Information CD, it may be a hot link to the book itself.

**KeyCap** The name of a keyboard key. Note that **Return** and **Enter** both refer to the same key.

*Emphasis* Text that is emphasized.

**Emphasis** Text that is strongly emphasized or a heading.

**Term** The defined use of an important word or phrase.

`ComputerOut` Text displayed by the computer.

**`UserInput`** Commands and other text that you type.

`Command` A command name or qualified command phrase.

*`Variable`* The name of a variable that you may replace in a command or function or information in a display that represents several possible values.

`[ ]` The contents are optional in formats and command descriptions. If the contents are a list separated by |, you may choose one of the items.

`{ }` The contents are required in formats and command descriptions. If the contents are a list separated by |, you must choose one of the items.

... The preceding element may be repeated an arbitrary number of times.

| Separates items in a list of choices.

# Contents

## 1. Overview

## 2. Accounting Subsystem

## 3. Asynchronous I/O Subsystem

## 4. FibreChannel Subsystem

## 5. File System Subsystem

## 6. Interprocess Communication (IPC) Subsystem

## 7. Kernel Panic Dump Subsystem

## 8. Memory Paging Subsystem

## 9. Process Management Subsystem

## 10. Spinlock Pool

## 11. Streams Subsystem

## 12. Miscellaneous Parameters

## 13. Obsolete Parameters

## A. Tunable Kernel Parameters

## B. Tunable Reference (Manpages)

# 1 Overview

## Tunables by Subsystem

The tunable kernel parameters are divided into the following subsystems and groups.

# Tunable Kernel Parameters

Tunables (or configurable) kernel parameters are kernel variables that allow the operating system to be configured to fit specific system needs, resulting in better performance and/or more efficient allocation of resources. The ideal value for each parameter is often determined by the system's particular hardware configuration, the specific mix of applications the system runs, and the trustworthiness of system users; factors that vary widely from system to system.

HP attempts to provide reasonable default parameter settings, but it may be necessary or beneficial to modify these settings to better suit the needs of your particular system's users. Refer to the individual manpage to obtain detailed information about any configurable kernel parameter.

Individual parameters usually pertain to a specific subsystem; some are independent, but others are interrelated or interact with each other. The parameter descriptions are grouped according to subsystem in “Tunables by Subsystem” on page 10.

**IMPORTANT**

All HP-UX tunable kernel parameters are release-specific. They may be added, removed, or have their meanings changed in future releases of HP-UX.

The tunable parameters described in this manual are specific to HP-UX release 11i. Though most of the names are the same as those in the previous releases, the behavior of specific tunables may differ significantly from previous releases.

As technological advances expand system size and capabilities, it is not uncommon for maximum and/or default values of certain parameters to change between releases or to vary between 32-bit and 64-bit processors. The information provided in this manual is believed to be accurate as of the time it was written. However, system changes may occasionally result in differences between this manual and what is actually present on your system.

**CAUTION**

Changing kernel parameters to improper or inappropriate values or combinations of values can cause data loss, system panics, and other (possibly very obscure and/or difficult to diagnose) operating anomalies.

- *Read the documentation*. Before altering the value of any configurable kernel parameter, be sure you know the implications of making the change.
- *Keep records*. Write down the parameter name and its old value as well as the new value in case you need to restore the old value.
- *Check ranges*. Never set any system parameter to a value outside the allowable range for the parameter. (SAM refuses to store values outside the allowable range.)
- *Be aware of side effects*. Many parameters interact, and their values must be selected in a balanced way.

# How to Specify Configurable Parameter Values

All configurable kernel parameters must be specified using an integer value or a formula consisting of a valid integer expression. All can be specified as an integer, and most can be specified using a formula; but only a minority are usually specified using formula values.

## Entering Values

To enter an integer or formula value for a configurable parameter:

1. Select **Modify Configurable Parameter** from the **Actions** pull-down menu at the top of the **Kernel Configuration: Configurable Parameters** window.
2. Select **Specify New Formula/Value** in the **Modify Configurable Parameter** dialog box that appears, then click on the box labeled **Formula/Value**.
3. Type the formula or value to be used in the box using the backspace key to move left and erase existing characters and typing new characters as desired.
4. Use the **OK** button to exit with the new value, or **Cancel** to exit without changing the original value.

## Integer Values

To specify an integer value, type the correct value in the **Formula/Value** box in the **Modify Configurable Parameter** window.

To specify an integer value, select the **Formula/Value** box in the **Modify Configurable Parameter** window, then type the correct value in the box.

## Formula Values

When using formulas to specify a parameter value, the formula must be an integer expression. In other words, every element in the expression must be an integer value as defined for the C programming language.

Configurable parameter names are usually lowercase but the values assigned to them are specified in formulas as the same name with initial capitals. For example, the value assigned to `npty` (by a C-language `#define` statement) is `Npty`. Thus, a typical formula for `nfile` in the kernel description file resembles:

```
nfile ((16*(Nproc+16+MaxUsers)/10)+32+2*Npty)
```

where `Nproc` represents the defined value of `nproc`, `MaxUsers` represents the defined value of `maxusers`, and `Npty` represents the defined value for `npty`. There are a few isolated exceptions to the uppercase/lowercase rule on certain systems, but they are few.

White space (spaces and tabs) are not allowed in the kernel description file. If any are present in the formula value typed into the value box, SAM removes the white-space characters where necessary.

## Undocumented Kernel Parameters

Some of the configurable parameters that appear in the kernel master file are not documented, and some are not known to or are not supported by SAM for any of several possible reasons:

- The parameter is obsolete. It is no longer used in current HP-UX releases, but might appear in an existing kernel configuration. If SAM encounters an obsolete parameter in the current kernel configuration, it does not display it in the list of configurable parameters that can be changed. It also removes that parameter when creating the new configuration file used to build the pending kernel.
- The parameter is not supported by SAM. As with obsolete parameters, it is not displayed in the list of configurable parameters, but it is retained in the new kernel configuration file to ensure that no malfunctions are introduced due to a missing parameter.
- The parameter is assigned a value by kernel configuration software, which is frequently based on external factors. The assigned value might be used when calculating values for one or more other parameters.
- The parameter is for HP factory or support use only. No change from the default value should be made unless specifically directed otherwise by official HP support personnel.
- The parameter supports obsolete or obsolescent software. For information about how to select a nondefault value, consult the documentation furnished with the software that the parameter supports.

# 2 Accounting Subsystem

## Overview of Accounting Parameters

When activated by the system administrator, the process accounting subsystem maintains log files for storing information about system and user processes. These files can become large, reducing available space for other files on the file system where they reside.

If process accounting is running, the system suspends process accounting whenever the available space on the file system where the accounting files reside falls below a certain threshold. The threshold is defined as the signed, arithmetic sum of the `acctsuspend` kernel parameter value at system boot time and the `minfree` file system parameter value that was defined during file system creation. (See “How to Determine Value of minfree” on page 19.) When process accounting is suspended, the system issues the message:

```
Accounting suspended
```

and accounting remains suspended until enough free space becomes available to resume. The resume threshold value is defined by the signed, arithmetic sum of the `acctresume` kernel parameter and the `minfree` file system parameter. To prevent suspend-resume conflicts, `acctresume` must have a value greater than `acctsuspend`.

When sufficient file system space becomes available and accounting resumes, the system issues the message:

```
Accounting resumed
```

For more information about selecting values for `acctsuspend` and `acctresume`, see “Specifying a Threshold Value” on page 18.

### Suspend Accounting Threshold

The suspend accounting threshold can be set to any value from zero to 100 percent of the usable file system size. Any value for `acctsuspend` which, when added to `minfree`, produces a zero or negative result, sets the suspend threshold at 0 percent, allowing accounting files to overwrite the entire reserved minimum free space without being suspended.

### Resume Accounting Threshold

The resume accounting threshold also can be set to any value from zero to 100 percent of the usable file system size. If the signed, arithmetic sum of the `acctresume` value at system boot time plus `minfree` is 100 or greater, accounting cannot resume until 100 percent of the entire usable file system is available. This means that accounting must remain suspended until the file system is completely empty (unlikely), the system is rebooted, or a startup command is executed from the keyboard (see *acctsh* (1M)). In a more practical sense, any value that results in a resume threshold higher than maximum normal free space on the file system will prevent process accounting from resuming.

## Accounting Parameter Summary

Two configurable kernel parameters are related to accounting:

| | | |
|---|---|---|
| Accounting | `acctresume` | Resume accounting when free space on the file system where accounting log files reside rises above `acctresume` plus `minfree` percent of total usable file system size. See *acctsuspend* (5). |
| Accounting | `acctsuspend` | Suspend accounting when free space on the file system where accounting log files reside drops below `acctsuspend` plus `minfree()` percent of total usable file system size. See *acctsuspend* (5). |

### Additional Information

- "Overview of Accounting Parameters" on page 16
- "Specifying a Threshold Value" on page 18
- "How to Specify Configurable Parameter Values" on page 12

# Specifying a Threshold Value

acctsuspend and acctresume are integers that each specify a percentage of total usable space on the file system where the accounting log files reside. This percentage is added to the current value of minfree to determine what percentage of total file system space must be available for process accounting to continue operating.

Threshold values are determined by adding the specified value for acctsuspend or acctresume to the current value of minfree. If the value of either parameter is less than zero, that value reduces the free space requirement, allowing the accounting files to invade the protected minimum free space area reserved by minfree. If the sum of acctsuspend and minfree is zero or negative, accounting can continue until the entire file system is full, rendering the value of acctresume meaningless.

## Calculating Threshold Values

The value chosen for acctsuspend or acctresume represents the difference between the corresponding suspend-accounting or resume-accounting threshold value and the current value of minfree for the file system where accounting log files reside.

- Nonnegative values for acctsuspend and acctresume are added to the current value of minfree to determine the suspend or resume threshold, respectively, as a percentage of total usable file system size. Accounting cannot invade free-space area.
- Negative values for acctsuspend and acctresume are subtracted from the current value of minfree to determine the suspend or resume threshold as a percentage of total usable file system size. Accounting suspension or resumption, respectively, occurs inside the reserved minimum free-space area.
- A suspend-accounting threshold value that is zero or negative allows accounting to continue until the file system overflows.
- A resume threshold value that is 100 or greater, or that is larger than the maximum expectable free space on the file system at any time, prevents accounting from resuming until the system is rebooted or accounting is manually restarted using the startup command (see *acctsh* (1M)*)*.

## Examples:

Suspend when less than 15% of total file system space is available, resume when available space reaches 18%:

minfree = 10
acctsuspend = 5
acctresume = 8

Suspend when less than 2% of total file system space is available, resume when available space reaches minfree:

minfree = 5
acctsuspend = -3 (minus 3)
acctresume = 0

Suspend when available file-system space shrinks to minfree; resume when available space increases to minfree plus 3% of total file system:

minfree = 5
acctsuspend = 0
acctresume = 3

Suspend when file system is completely full, do not resume (resume when available space reaches 106% of total file system which cannot happen):

minfree = 5
acctsuspend = -5 (minus 5)
acctresume = 101

## How to Determine Value of minfree

`acctsuspend` and `acctresume` specify accounting thresholds based on the current value of the `minfree` parameter for the file system where the accounting files reside. `minfree` is specified when the file system is being created, and determines what percentage of the total file system space is reserved and cannot be used by ordinary users. The value can range from zero to 100.

The default value for `minfree` on HP-UX file systems is 10 percent. To determine the current value of `minfree`, use the `df` command with the `-t` option (see *df* (1M)). The value is determined by the current value of `fs_minfree` in the file system data structure (see *fs* (4)).

For example, to find the current value of `minfree` on the mounted file system `/users`, use the command:

```
df -t /users
```

## Additional Information

- “Overview of Accounting Parameters” on page 16
- “Accounting Parameter Summary” on page 17
- “How to Specify Configurable Parameter Values” on page 12

# 3 Asynchronous I/O Subsystem

## Asynchronous I/O Parameter Summary

The following kernel parameters are used for managing asynchronous I/O operations. The first four (aio_*) are related to POSIX asynchronous I/O operations; the last (max_async_ports) pertains to open ports between processes and the asynchronous disk I/O driver.

See also “How to Specify Configurable Parameter Values” on page 12.

| | | |
|---|---|---|
| Asynchronous I/O | aio_listio_max | Maximum number of POSIX asynchronous I/O operations allowed in a single lio_listio() call. See *aio_listio_max* (5). |
| Asynchronous I/O | aio_max_ops | System-wide maximum number of POSIX asynchronous I/O operations allowed at one time. See *aio_max_ops* (5). |
| Asynchronous I/O | aio_physmem_pct | Maximum percentage of total system memory that can be locked for use in POSIX asynchronous I/O operations. See *aio_physmem_pct* (5). |
| Asynchronous I/O | aio_prio_delta_max | Maximum priority offset (slowdown factor) allowed in a POSIX asynchronous I/O control block (aiocb). See *aio_prio_delta_max* (5). |
| Asynchronous I/O | max_async_ports | System-wide maximum number of ports to the asynchronous disk I/O driver that processes can have open at any given time. See *max_async_ports* (5). |

# 4 FibreChannel Subsystem

## FibreChannel Parameter Summary

Two kernel configuration parameters pertain to the FibreChannel SCSI subsystem and communication between the system processor and any peripheral devices that interact with it using FiberChannel Protocol (FCP). These parameters are used for adjusting the default amount and type of memory allocated for supporting concurrent FCP read, write, and/or control requests.

See also “How to Specify Configurable Parameter Values” on page 12.

| | | |
|---|---|---|
| FibreChannel | `fcp_large_config` | Define a Tachyon host bus adapter configuration as large or small. See *fcp_large_config* (5). |
| FibreChannel | `max_fcp_reqs` | Maximum number of concurrent FCP requests that are allowed on an FCP adapter. The default value of 512 requests can be changed by specifying a different value for this parameter. The optimal limit on concurrent requests depends on a number of factors such as configuration, device characteristics, I/O load, host memory, and other values that FCP software cannot easily determine. See *max_fcp_reqs* (5). |

# 5 File System Subsystem

# File System Parameter Summary

Configurable parameters related to file system performance fall into the following categories.

See also “How to Specify Configurable Parameter Values” on page 12.

## File System Buffer Parameter Summary

Allocate system physical memory resources for static and dynamic file system buffer cache space. For more information, see “File System Buffer Parameters” on page 29.

| | | |
|---|---|---|
| File System: Buffer | `bufpages` | Number of 4 KB pages in file system static buffer cache. See *bufpages* (5). |
| File System: Buffer | `dbc_max_pct` | Maximum percentage of memory for dynamic buffer cache. See *dbc_max_pct* (5). |
| File System: Buffer | `dbc_min_pct` | Minimum percentage of memory for dynamic buffer cache. See *dbc_min_pct* (5). |
| File System: Buffer | `disksort_seconds` | Maximum wait time for disk requests. NO MANPAGE. |
| File System: Buffer | `nbuf` | System-wide number of static file system buffer and cache buffer headers. See *nbuf* (5). |

## Journaled File Systems (VxFS) Parameter Summary

Allocate space for the Directory Name Lookup Cache (DNLC) associated with VxFS file system inodes.

| | | |
|---|---|---|
| File System: Journaled | `vxfs_max_ra_kbytes` | Maximum amount of read-ahead data, in KB, that the kernel may have outstanding for a single VxFS file system. See *vxfs_max_ra_kbytes* (5). |
| File System: Journaled | `vxfs_ra_per_disk` | Maximum amount of VxFS file system read-ahead per disk, in KB. See *vxfs_ra_per_disk* (5). |
| File System: Journaled | `vx_fancyra_enabled` | Enable or disable VXFS file system read-ahead. NO MANPAGE. |
| File System: Journaled | `vx_ncsize` | Memory space reserved for VxFS directory path name cache. See *vx_ncsize* (5). |

## Logical Volume Manager (LVM) Parameter Summary

Control kernel interaction with the Logical Volume Manager. For more information, see “Logical Volume Manager (LVM) Operation” on page 32.

| | | |
|---|---|---|
| File System: LVM | `maxvgs` | Maximum number of volume groups configured by the Logical Volume Manager on the system. See *maxvgs* (5). |

## Open or Locked Files Parameter Summary

Set number of files that can be open or locked simultaneously. For more information, see "Open or Locked Files Parameters" on page 31.

| | | |
|---|---|---|
| File System: Open/Lock | `maxfiles` | Soft limit on how many files a single process can have opened or locked at any given time. See *maxfiles* (5). |
| File System: Open/Lock | `maxfiles_lim` | Hard limit on how many files a single process can have opened or locked at any given time. See *maxfiles_lim* (5). |
| File System: Open/Lock | `ncsize` | Inode space needed for directory name lookup cache (DNLC). NO MANPAGE. |
| File System: Open/Lock | `nfile` | Maximum number of files that can be open simultaneously on the system at any given time. See *nfile* (5). |
| File System: Open/Lock | `nflocks` | Maximum combined number of file locks that are available system-wide to all processes at one time. See *nflocks* (5). |
| File System: Open/Lock | `ninode` | Maximum number of open inodes that can be in memory. See *ninode* (5). |

## Disk Read-Ahead Parameter Summary

Control the amount of memory available for disk read-ahead operations.

| | | |
|---|---|---|
| File System: Read | `hfs_max_ra_blocks` | The maximum number of read-ahead blocks that the kernel may have outstanding for a single HFS file system. See *hfs_max_ra_blocks* (5). |
| File System: Read | `hfs_max_revra_blocks` | The maximum number of reverse read-ahead blocks that the kernel may have outstanding for a single HFS file system. See *hfs_max_revra_blocks* (5). |
| File System: Read | `hfs_ra_per_disk` | The amount of HFS file system read-ahead per disk drive, in KB. See *hfs_ra_per_disk* (5). |
| File System: Read | `hfs_revra_per_disk` | The amount of memory (in KB) for HFS reverse read-ahead operations, per disk drive. See *hfs_revra_per_disk* (5). |
| File System: Read | `hp_hfs_mtra_enabled` | Enable or disable HFS multithreaded read-ahead. NO MANPAGE. |
| File System: Read | `vxfs_max_ra_kbytes` | Maximum amount of read-ahead data, in KB, that the kernel may have outstanding for a single VxFS file system. See *vxfs_max_ra_kbytes* (5). |
| File System: Read | `vxfs_ra_per_disk` | Maximum amount of VxFS file system read-ahead per disk, in KB. See *vxfs_ra_per_disk* (5). |

## SCSI Devices Parameter Summary

Control SCSI device access.

| | | |
|---|---|---|
| File System: SCSI | `scsi_maxphys` | Maximum record size for the SCSI I/O subsystem, in bytes. See *scsi_maxphys* (5). |
| File System: SCSI | `scsi_max_qdepth` | Maximum number of SCSI commands queued up for SCSI devices. See *scsi_max_qdepth* (5). |
| File System: SCSI | `st_ats_enabled` | Flag whether to reserve a tape device on open. See *st_ats_enabled* (5). |
| File System: SCSI | `st_fail_overruns` | SCSI tape read resulting in data overrun causes failure. See *st_fail_overruns* (5). |
| File System: SCSI | `st_large_recs` | Enable large record support for SCSI tape. See *st_large_recs* (5). |

## Asynchronous Writes Parameter Summary

Control asynchronous writes to file system.

| | | |
|---|---|---|
| File System: Write | `fs_async` | Enable/disable asynchronous writes of file system data structures to disk. See *fs_async* (5). |

# File System Buffer Parameters

The system allocates a portion of system memory for use in block-mode file operations (such as exec() and mount() system calls and inode reading). Buffer space is reserved in increments of 4096-byte pages, but buffers can be much larger than 4096 bytes, requiring as many as 16 or more pages per buffer, depending on hardware device and configuration characteristics.

Two methods for allocating buffer space are supported: static and dynamic. The obsolescent static method allocates buffer space and buffer header structures at system boot time. The preferred dynamic buffer cache method allocates buffer space and supporting data structures as they are needed, using predefined minimum and maximum values to establish overall buffer cache space allocation limits.

## Dynamic Buffer Cache

Most system administrators prefer to specify what percentage or range of percentages of available system memory can be allocated for buffer use, letting the system allocate memory for buffers as needed within the specified limits. Two kernel parameters, dbc_min_pct and dbc_max_pct, control the lower and upper limit, respectively, as a percentage of system memory. How many pages of memory are allocated for buffer cache use at any given time is determined by system needs, but the two parameters ensure that allocated memory never drops below dbc_min_pct and cannot exceed dbc_max_pct percent of total system memory. Administrators of multiple systems usually prefer this method because it provides an easy way to choose buffer space limits that are directly related to how much memory is actually installed in each machine, allowing common or similar kernel configurations throughout the network.

**NOTE** To enable dynamic buffer caching, the kernel parameters nbuf and bufpages must both be set to zero.

## Static Buffer Allocation

For administrators who choose not to use dynamic buffer caching, the two kernel parameters, nbuf and bufpages, control static buffer allocation. If bufpages is nonzero, it specifies the fixed number of 4096-byte pages that are to be allocated for the file system buffer cache. nbuf is provided for backward compatibility purposes. If set to a nonzero value, nbuf specifies the maximum number of buffer headers that can exist in the buffer header array. However, the preferred approach is to set nbuf to zero, in which case, one header is created for each two bufpages allocated.

## Recommended Procedure

Set bufpages and nbuf to zero, and set dbc_min_pct and dbc_max_pct to the desired upper and lower limits as a percentage of total system memory. This activates dynamic buffer cache allocation.

## Alternate Procedure

To allocate a fixed amount of memory for static buffer cache, set `bufpages` to the desired number of 4 KB pages, and set `nbuf` to zero, which allocates space for `bufpages/2` buffer headers in the header array. `dbc_min_pct` and `dbc_max_pct` are ignored.

## Additional Information

- “File System Buffer Parameter Summary” on page  26

# Open or Locked Files Parameters

## Open and Locked Files

Individual processes can open one or more files for reading or writing, and it is not uncommon for a process to have many files open at the same time; particularly in large database applications, for example.

Furthermore, mail, database, and other applications often require simultaneous access to a given file by two or more processes. To prevent file or data corruption, a process that is altering the contents of a file must be able to lock the file against conflicting uses until it is safe to release control of the file.

Open and locked files require memory and other system resources. These resources must be balanced against other system needs to maintain optimum overall system performance. Use `nflocks` to limit the combined total number of file locks that are available system-wide to all processes at any given time.

## Limits

Two parameters, `maxfiles` and `maxfiles_lim`, respectively, govern the soft and hard limits on the number of files a process can open simultaneously. `nfile` governs the maximum number of files that can be open on the entire system at any given time.

## Additional Information

- “Open or Locked Files Parameter Summary” on page 27

# Logical Volume Manager (LVM) Operation

Logical Volume Manager (LVM) is a subsystem for managing file systems and disk storage space that are structured into logical volumes rather than being restricted to the beginning and end points of a physical disk. Logical volumes can be smaller than the disk or disk array on which they reside, or they can include all or part of several disks or disk arrays. Logical volume boundaries are not required to coincide with the boundaries of physical disks when multiple disks or arrays are used.

Managing logical volumes is done by the Logical Volume Manager, not the kernel. However, the kernel contains data structures for each volume group on the system, and the space reserved for LVM data structures must be sufficient to support the number of volume groups that exist on the system. This is done with the `maxvgs` kernel configuration parameter.

## Volume and Group Boundaries

Logical volume groups consist of one or more logical volumes. Logical volume boundaries within a volume group can be configured anywhere on a given disk. However, a single disk device cannot be shared between volume groups. Disk arrays configured as RAID (redundant array of independent disks) arrays for data protection are treated as a single disk device and cannot be shared between volume groups. Individual disks in any array that is not configured as a RAID array are treated as individual devices, and individual devices can be assigned to any volume group as desired by the administrator.

## Additional Information

- “Logical Volume Manager (LVM) Parameter Summary” on page 26

# 6 Interprocess Communication (IPC) Subsystem

## System V Interprocess Communication Mechanisms

The HP-UX operating system uses shared memory to provide three mechanisms for communicating between cooperating programs and processes:

*Messages* Message data is stored in a given part of shared memory to be retrieved by receiving programs. For more information about how shared memory space for messages is allocated and managed, see "Overview of Message Queue Operations" on page 35.

*Semaphores* Shared storage locations that contain up/down counters for signaling the current status of cooperating processes. For more information about how shared memory space for semaphores is allocated and managed, see "Overview of Semaphore Operations" on page 37.

*Shared Memory* Reserved data storage area shared by two or more processes by means of identically-defined data structures in each cooperating process. For more information about how shared memory space for shared data structures is allocated and managed, see "Overview of Shared Memory Operation" on page 38.

## Overview of Message Queue Operations

See “Message Parameter Summary” on page 39 for the list of parameters.

### Messages

Messages are small collections of data (400 bytes, for example) that can be passed between cooperating programs through a message queue. Messages within a queue can be of different types, and any process with proper permissions can receive the messages. A receiving process can retrieve the first message, the first message of a given type, or the first message of a group of types. See *msgop* (2) for more information.

### Message Queues

Message queues are implemented as linked lists of data stored in shared memory. The message queue itself contains a series of data structures, one for each message, each of which identifies the address, type, and size of the message plus a pointer to the next message in the queue.

To allocate a queue, a program uses the `msgget()` system call (see *msgget* (2)). Messages are placed in the queue by `msgsnd()` system calls and retrieved by `msgrcv()` (see msgop(2)). Other operations related to managing a given message queue are performed by the `msgctl()` system call (see *msgctl* (2)).

For information about using messages in programs, refer to an advanced UNIX programming text such as *Advanced UNIX Programming* by Marc J. Rochkind, Prentice-Hall, Inc., ISBN 0-13-011800-1.

### Message and Message Queue Management

Message queues and the message header array are located in (swappable) shared memory space. Other data structure arrays necessary for managing them are located in the (nonswappable) kernel space.

Kernel configuration parameters for messages control:

- Maximum number of message queues on system,
- Maximum message queue size,
- Maximum message length,
- Maximum total combined length of messages in a queue,
- Maximum number of messages per queue,
- Maximum number of simultaneous messages system-wide,
- Maximum size of message header list,
- Maximum size of free-space map for locating new messages.

IPC messages require the following memory space allocations:

- Space in the kernel area for message identifiers.
- Shared memory space for message queues.
- Shared memory space for message headers.

#### Message Queues

*Queue Size* Each message queue is created with enough space for `msgmnb` (message queue number of bytes) bytes of messages. Each message contains one or more message segments of `msgssz` (message segment size) bytes each.

*Message Size* To discourage malicious or poorly written programs from consuming excessive message space, individual messages cannot contain more than `msgmax` (message maximum) bytes per message. Each message is stored in the queue as a series of one or more segments containing `msgssz` bytes per segment. The number of segments used for a message is the smallest integer that, when multiplied by `msgssz`, is greater than the total number of bytes in the message.

*Queue Space* The total space consumed by all messages in any given queue cannot exceed `msgmnb` bytes (message-queue maximum number of bytes. This value must not be less than `msgmax` bytes.

*Total Messages* The maximum number of messages that can exist in a queue at any given time depends on the length of individual messages and the constraints of queue size and other factors listed above. However, a system-wide limit is imposed. The total number of messages that can reside on the system at any one time in all queues cannot exceed `msgtql` (see *Message Header Array* in “System Globals” below).

### System Globals

In addition to the constraints on individual message queues and messages listed in “Message Queues” above, the kernel imposes additional constraints on IPC message management

*Message Queue Identifiers*

Each message queue has an associated message queue identifier stored in the nonswappable kernel area. `msgmni` (message maximum number of identifiers) limits the total number of message queues allowed on the system at any given time.

*Message Header Array*

Each message has an associated message header which is stored in an array in the swappable shared memory area. `msgtql` (message total quantity limit) defines the total number of messages that can be present system-wide at any given time.

*Free Space Management*

As messages are sent and received, space to contain those messages is allocated then released, making the space available for other messages. The kernel maintains a resource map which is used for identifying free space to allocate for new messages. The size of this map is controlled by the `msgmap` parameter.

# Overview of Semaphore Operations

System V IPC semaphores are used mainly to keep processes properly synchronized to prevent collisions when accessing shared data structures. Their use in software tends to be complex, so many programmers use alternate means of control where practical.

See “Semaphore Parameter Summary” on page 39 for the list of parameters.

## Semaphore Operations

The `semget()` system call allocates an array containing a specified number of semaphores (see *semget* (2)). Assigning an array with only one semaphore in the set is usually the most sensible for keeping programs reasonably simple. Subsequent semaphore operations are performed atomically on the entire array; individual semaphores in the array are manipulated by an array of semaphore operations (see *semop* (2)). `semctl()` is used to ascertain or change a semaphore's permissions, change time, or owner (see *semctl* (2)).

Semaphores are typically incremented by a process to block other processes while it is performing a critical operation or using a shared resource. When finished, it decrements the value, allowing blocked processes to then access the resource. Semaphores can be configured as binary semaphores which have only two values: 0 and 1, or they can serve as general semaphores (or counters) where one process increments the semaphore and one or more cooperating processes decrement it. To prevent undetectable overflow conditions, the kernel imposes a maximum value limit beyond which semaphores cannot be incremented. This limit, defined by the `semvmx` kernel parameter, must not exceed the maximum value of 65535. Semaphores are not allowed to have negative (less than zero) values.

## Semaphore Undo Operations

It may occasionally be necessary — when errors occur, a process must abort, a process dies, etc. — to change one or more semaphores to a new or previous value. This is called undoing a semaphore. Since the value of any semaphore when such conditions occur is unpredictable, the system enforces a limit on how much the value of a semaphore can change any undo operation. This limit is defined by the `semaem` kernel parameter.

For more information about System V IPC semaphore operation, consult an advanced UNIX programming text such as *Advanced UNIX Programming* by Marc J. Rochkind, Prentice-Hall, Inc., ISBN 0-13-011800-1.

## Semaphore Limits

Configurable kernel parameters are available to limit the number of sets of semaphores that can exist simultaneously on the system and the total number of individual semaphores available to users that can exist simultaneously on the system. A fixed limit on the number of semaphores that can exist in a set (500) is not configurable.

## Overview of Shared Memory Operation

See “Shared Memory Parameter Summary” on page 40 for the list of parameters.

### Shared Memory

Shared memory is reserved memory space for storing data structures and data being shared between or among cooperating processes. Sharing a common memory space eliminates the need for copying or moving data to a separate location before it can be used by other processes, reducing processor time and overhead as well as memory consumption.

### Shared Memory Access and Use

Shared memory management is similar in many respects to messages and semaphores. A process requests a shared memory segment allocation by means of the `shmget()` system call, specifying the segment size in one of the function parameters (see *shmget* (2)). One or more processes can then attach to the allocated segment by using the `shmat()` system call and detach when finished with the `shmdt()` system call (see *shmop* (2)). `shmctl()` is used to obtain information about the segment, and to remove the segment when it is no longer needed (see *shmctl* (2)).

Semaphores can be used to prevent shared memory read/write access collisions, but the more common method is for one process to populate (write to) a given shared memory segment and other processes read from the segment (write once, read many). In such arrangements, when a subsequent write operation is to be performed, the writing process allocates a new segment, and cooperating processes attach to that new segment when ready to use it.

Shared memory is allocated in swappable shared memory space. Data structures for managing shared memory are located in the kernel.

# IPC Parameter Summary

See also "How to Specify Configurable Parameter Values" on page 12.

## Message Parameter Summary

The following parameters control allocation of space for messages.

See also "Overview of Message Queue Operations" on page 35.

| | | |
|---|---|---|
| IPC: Message | `mesg` | Enable or disable IPC messages at system boot time. See *mesg* (5). |
| IPC: Message | `msgmap` | Size of free-space resource map for allocating shared memory space for messages. See *msgmap* (5). |
| IPC: Message | `msgmax` | System-wide maximum size (in bytes) for individual messages. See *msgmax* (5). |
| IPC: Message | `msgmnb` | Maximum combined size (in bytes) of all messages that can be queued simultaneously in a message queue. See *msgmnb* (5). |
| IPC: Message | `msgmni` | Maximum number of message queues allowed on the system at any given time. See *msgmni* (5). |
| IPC: Message | `msgseg` | Maximum number of message segments that can exist on the system. See *msgseg* (5). |
| IPC: Message | `msgssz` | Message segment size in bytes. See *msgssz* (5). |
| IPC: Message | `msgtql` | Maximum number of messages that can exist on the system at any given time. See *msgtql* (5). |

## Semaphore Parameter Summary

The following kernel parameters control System V IPC semaphore operating limits.

See also "Overview of Semaphore Operations" on page 37.

| | | |
|---|---|---|
| IPC: Semaphore | `sema` | Enable or disable IPC semaphores at system boot time. See *sema* (5). |
| IPC: Semaphore | `semaem` | Maximum value by which a semaphore can be changed in a semaphore "undo" operation. See *semaem* (5). |
| IPC: Semaphore | `semmap` | Size of free-semaphores resource map for allocating requested sets of semaphores. See *semmap* (5). |
| IPC: Semaphore | `semmni` | Maximum number of sets of IPC semaphores allowed on the system at any one time. See *semmni* (5). |
| IPC: Semaphore | `semmns` | Maximum number of individual IPC semaphores available to system users, system-wide. See *semmns* (5). |

| | | |
|---|---|---|
| IPC: Semaphore | `semmnu` | Maximum number of processes that can have undo operations pending on any given IPC semaphore on the system. See *semmnu* (5). |
| IPC: Semaphore | `semmsl` | Maximum number of individual System V IPC semaphores per semaphore identifier. See *semmsl* (5). |
| IPC: Semaphore | `semume` | Maximum number of IPC semaphores that a given process can have undo operations pending on. See *semume* (5). |
| IPC: Semaphore | `semvmx` | Maximum value any given IPC semaphore is allowed to reach (prevents undetected overflow conditions). See *semvmx* (5). |

## Shared Memory Parameter Summary

The following parameters control allocation of space for shared memory.

See also “Overview of Shared Memory Operation” on page 38.

| | | |
|---|---|---|
| IPC: Share | `core_addshmem_read` | Flag to include readable shared memory in a process core dump. See *core_addshmem_read* (5). |
| IPC: Share | `core_addshmem_write` | Flag to include read/write shared memory in a process core dump. See *core_addshmem_write* (5). |
| IPC: Share | `shmem` | Enable or disable shared memory at system boot time. See *shmem* (5). |
| IPC: Share | `shmmax` | Maximum allowable shared memory segment size (in bytes). See *shmmax* (5). |
| IPC: Share | `shmmni` | Maximum number of shared memory segments allowed on the system at any given time. See *shmmni* (5). |
| IPC: Share | `shmseg` | Maximum number of shared memory segments that can be attached simultaneously to any given process. See *shmseg* (5). |

# 7 Kernel Panic Dump Subsystem

# Kernel Panic Dump Parameter Summary

The following parameters affect dump operations when a kernel panic occurs and parts of system memory are dumped to disk.

See also “How to Specify Configurable Parameter Values” on page 12.

| | | |
|---|---|---|
| Kernel Panic Dump | `alwaysdump` | Select which classes of system memory pages are to be dumped if a kernel panic occurs. See *alwaysdump* (5). |
| Kernel Panic Dump | `dontdump` | Select which classes of system memory pages are not to be dumped if a kernel panic occurs. See *dontdump* (5). |

# 8 Memory Paging Subsystem

# Overview of Memory Paging Parameters

Configurable kernel parameters for memory paging enforce operating rules and limits related to virtual memory (swap space). They fall into the following categories:

*Total System Swap*

Maximum swap space that can be allocated, system-wide. Parameters include `maxswapchunks` and `swchunk`.

*Device Swap*

Swap space allocated on hard disk devices. Parameters include: `nswapdev`.

*File System Swap*

Swap space allocated on mounted file systems. Parameters include: `allocate_fs_swapmap` and `nswapfs`.

*Pseudo-Swap*

Use of installed RAM as pseudo-swap, allowing virtual memory space allocation beyond the limit of swap space on disk devices. Parameters include: `swapmem_on`.

*Variable Page Sizes*

The size of virtual memory pages can be altered to make swap operations more efficient in particular applications. Parameters include: `vps_ceiling`, `vps_chatr_ceiling`, and `vps_pagesize`.

## Total System Swap

Two configurable kernel parameters determine the total amount of swap space that can exist on the entire system, including device swap, file system swap, and remote NFS swap. `maxswapchunks` specifies how many swap chunks can be created and/or allocated, system-wide. `swchunk` determines the size of each chunk that is created on successive devices or file systems. Selecting an appropriate value for this parameter requires extensive knowledge of kernel operation and system internals. *Without such knowledge, do not change* `swchunk` *to a nondefault value.*

## Device Swap

When devices are connected to the system and configured, part of the disk can be reserved for device swap. The only configurable kernel parameter related to device swap is `nswapdev`, which specifies how many devices have been configured with space allocated for device swap. Data structure storage space is reserved in the kernel for `nswapdev` devices.

- If the value of `nswapdev` is greater than the actual number of swap devices, the small amount of memory space (<50 bytes per device) allocated for data structures to support nonexistent devices is wasted and cannot be used for other purposes.
- If `nswapdev` is less than the number of available swap devices, only `nswapdev` devices can be accessed because no data structure storage space is available for supporting the remaining devices.

### File System Swap

In addition to swap space on individual devices, swap space can also be created in existing mounted file systems. This file system swap is somewhat slower than device swap because it must be handled by reading from and writing to open files rather than transferring data directly between the operating system and the disk device. `nswapfs` is the file system swap counterpart to `nswapdev` and defines how many locally mounted file systems can be configured system-wide to support file system swap. Like `nswapdev`, `nswapfs` should be set to match the actual number of file systems that are normally mounted and intended to be used for file system swap (about `nswapfs` times 300 bytes of kernel space is required for data structure storage). Only hard disk read/write file systems can be used for file system swap.

A second configurable parameter, `allocate_fs_swapmap` can be set to enable or disable the allocation of swap space at the time `swapon()` is called rather than waiting to allocate space using `malloc()`. Preallocating space ensures the unconditional availability of file system swap space when `malloc()` is called (otherwise a file-system-full error could occur in some circumstances). Preallocation of file system swap space when `swapon()` is called is commonly used when high availability is important, but it does prevent other processes from using any resources that are not being use by the process that reserved them.

### Pseudo-Swap

Memory allocation is normally based on the availability of virtual memory (swap space available on disks and file systems). Total available (swappable) memory is the same as total swap space. However, on large systems with massive amounts of installed RAM this can lead to inefficiencies.

Consider a system, for example, that contains 200 MB of RAM, and has 1 GB of swap on the root disk. It is inappropriate to limit such a system running in single-user mode with only the root disk mounted to only 1 GB of memory space when the kernel occupies only 10% or less of the available system RAM. By allowing the use of pseudo-swap, any unused RAM can also be allocated for use by the swap system, allowing larger, more demanding processes to run. On a workstation with 100 or 200 MB of swap on the root device and 16 MB of RAM, the advantage of this capability is much less significant.

`swapmem_on` is used to enable or disable the allocation of pseudo-swap space in system RAM.

### Additional Information

- "Memory Paging Parameter Summary" on page 46

## Memory Paging Parameter Summary

| | | |
|---|---|---|
| Memory Paging | `allocate_fs_swapmap` | Enable or disable preallocation of file system swap space when `swapon()` is called as opposed to allocating swap space when `malloc()` is called. Enabling allocation reduces risk of insufficient swap space and is used primarily where high availability is important. See *allocate_fs_swapmap* (5). |
| Memory Paging | `maxswapchunks` | This parameter, multiplied by `swchunk` times the system block size value (`DEV_BSIZE`), determines the combined maximum amount of device swap and file system swap space that can be configured, system-wide. See *maxswapchunks* (5). |
| Memory Paging | `max_mem_window` | Maximum number of group-private 32-bit shared memory windows. See *max_mem_window* (5). |
| Memory Paging | `nswapdev` | Maximum number of devices, system-wide, that can be used for device swap. Set to match actual system configuration. See *nswapdev* (5). |
| Memory Paging | `nswapfs` | Maximum number of mounted file systems, system-wide, that can be used for file system swap. Set to match actual system configuration. See *nswapfs* (5). |
| Memory Paging | `remote_nfs_swap` | Enable or disable swap to mounted remote NFS file system. Used on cluster clients for swapping to NFS-mounted server file systems. See *remote_nfs_swap* (5). |
| Memory Paging | `swapmem_on` | Enable or disable pseudo-swap allocation. This allows systems with large installed memory to allocate memory space as well as disk swap space for virtual memory use instead of restricting availability to defined disk swap area. See *swapmem_on* (5). |
| Memory Paging | `swchunk` | Amount of space allocated for each chunk of swap area. Chunks are allocated from device to device by the kernel. Changing this parameter requires extensive knowledge of system internals. *Without such knowledge, do not change this parameter from the normal default value.* See *swchunk* (5). |

## Variable Page Size Parameter Summary

| | | |
|---|---|---|
| Memory Paging: Size | `vps_ceiling` | Maximum system-selected page size (in KB) if the user does not specify a page size. See *vps_ceiling* (5). |
| Memory Paging: Size | `vps_chatr_ceiling` | Maximum page size a user can specify with the `chatr` command in a program. See *vps_chatr_ceiling* (5). |

| | | |
|---|---|---|
| Memory Paging: Size | `vps_pagesize` | Minimum user page size (in KB) if no page size is specified using `chatr`. See *vps_pagesize* (5). |

## Additional Information

- “Overview of Memory Paging Parameters” on page 44
- “How to Specify Configurable Parameter Values” on page 12

# 9 Process Management Subsystem

# Overview of Process Management Parameters

Process management includes:

- Managing the number of processes on the system and processes per user to keep system resources effectively distributed among users for optimal overall system operation.
- Managing allocation of CPU time to competing processes at equal and different priority levels.
- Allocation of virtual memory between processes, protecting the system and competing users against unreasonable demands of abusive or run-away processes.

## Process Management

See also “Process Parameter Summary” on page 52.

Process requirements on individual systems can vary widely. For example, it is not uncommon for a modest workstation running a CDE or Motif environment to have over 100 simultaneous processes supporting a single system user. On the other hand, a large, multiuser system could have 1000 simple ASCII user terminals connected to it, each running only two or three specialized processes. Obviously the process limits imposed on the system by the kernel must be quite different for these two common examples.

Two configurable parameters apply specifically to managing system processes, `nproc` and `maxuprc`.

Select a value for `nproc` that is sufficient to provide enough processes for every user at any given time when the maximum normal number of users are logged in.

Select a value for `maxuprc` that is adequate to meet the normal needs of all system users, but low enough to prevent a run-away program from spawning too many processes (thus preventing or restricting new process availability to other users), and to protect normal users from malicious system abuse by any other user.

## Kernel Threads

See also “Kernel Threads Parameter Summary” on page 53.

On large systems with multiple processors, parts of processes can sometimes be split into threads and run simultaneously on separate processors. Two kernel parameters help manage the consumption of system resources by threaded processes, `max_thread_proc` and `nkthread`.

## CPU Timesharing Management

See also “Timesharing Parameter Summary” on page 52.

The kernel checks frequently for other processes requesting CPU time. Checks for CPU requests from higher-priority processes are made every 10 milliseconds. When two or more processes at the same priority level (such as from two different system users) are competing for CPU time, CPU time is “sliced” into segments, defined by `timeslice`, and passed from process to process in a round-robin fashion, preventing a single process from monopolizing the CPU until it blocks or terminates.

## Memory Allocation Management

See also “Memory Space Parameter Summary” on page 52.

PA-RISC hardware supports the allocation of up to 2 GB of virtual memory to a single process. This memory is divided between the text space where programs are stored, data space (globals, statics, locals to `main()`, strings, etc.), dynamic storage (space allocated by `malloc()`, stack, registers, etc.), and approximately 200 MB reserved for other purposes. Most processes never need that much room, and furthermore, the 2 GB limit exceeds the installed disk space on many systems.

These configurable kernel parameters place protective limits on allocation of process space: `maxtsiz`, `maxtsiz_64bit`, `maxdsiz`, `maxdsiz_64bit`, `maxssiz`, and `maxssiz_64bit`.

Different users on different systems have widely divergent process space needs, making it impractical or frustrating to enforce a fixed limit on text or data storage, for example, for every user on every system. For example, some users on a research or development system may have very modest text segment needs yet require very large data segment space (a small program operating on a very large array, for example), while other users on the same system may need little data space yet have rather large text (program storage) requirements.

On the other hand, a production system supporting hundreds of users may require moderate data storage space per user, yet the aggregate total users on the system may require massive swap space, making it appropriate to use relatively small space limits to protect other users from malfunctioning or run-away iterative processes started by a single user.

### Selecting Limits

Selecting limits that meet the needs of all users on the system while also protecting all users from run-away programs or abuses by others requires a thorough understanding of individual users' program needs. When selecting values, be aware that the limits chosen for `maxtsiz`, `maxdsiz`, and `maxssiz` are safety nets to prevent monopolization of virtual memory resources to the detriment of other users. They are not intended to restrict individual users' access to needed space.

`maxtsiz`, `maxdsiz`, and `maxssiz` are usually set to values whose combined totals can exceed available swap space. This is appropriate because most user program requirements are significantly less than the limit values imposed by these parameters, and selecting values that protect users from occasional malfunctions rather than using smaller values that tend to ration space among users improves overall system performance for everyone on the system.

## Process Management Parameter Summary

See also “How to Specify Configurable Parameter Values” on page 12.

### Timesharing Parameter Summary

See also “CPU Timesharing Management” on page 50.

| | | |
|---|---|---|
| ProcessMgmt: CPU | `timeslice` | Maximum time a process can use the CPU until it is made available to the next process having the same process execution priority. This feature also prevents runaway processes from causing system lock-up. See *timeslice* (5). |

### Memory Space Parameter Summary

See also “Memory Allocation Management” on page 51.

| | | |
|---|---|---|
| ProcessMgmt: Memory | `maxdsiz` | Maximum process data storage segment space that can be used for statics and strings, as well as dynamic data space allocated by `sbrk()` and `malloc()` (32-bit processes). See *maxdsiz* (5). |
| ProcessMgmt: Memory | `maxdsiz_64bit` | Maximum process data storage segment space that can be used for statics and strings, as well as dynamic data space allocated by `sbrk()` and `malloc()` (64-bit processes). See *maxdsiz* (5). |
| ProcessMgmt: Memory | `maxssiz` | Maximum dynamic storage segment (DSS) space used for stack space (32-bit processes). See *maxssiz* (5). |
| ProcessMgmt: Memory | `maxssiz_64bit` | Maximum dynamic storage segment (DSS) space used for stack space (64-bit processes). See *maxssiz* (5). |
| ProcessMgmt: Memory | `maxtsiz` | Maximum allowable process text segment size, used by unchanging executable-code (32-bit processes). See *maxtsiz* (5). |
| ProcessMgmt: Memory | `maxtsiz_64bit` | Maximum allowable process text segment size, used by unchanging executable-code (64-bit processes). See *maxtsiz* (5). |

### Process Parameter Summary

See also “Process Management” on page 50.

| | | |
|---|---|---|
| ProcessMgmt: Process | `executable_stack` | Allows or denies program execution on the stack. See *executable_stack* (5). |

| | | |
|---|---|---|
| ProcessMgmt: Process | `maxuprc` | Maximum number of processes that any single user can have running at the same time, including login shells, user interface processes, running programs and child processes, I/O processes, etc. If a user is using multiple, simultaneous logins under the same login name (user ID) as is common in X Window, CDE, or Motif environments, all processes are combined, even though they may belong to separate process groups. Processes that detach from their parent process group, where that is possible, are not counted after they detach (line printer spooler jobs, certain specialized applications, etc.). See *maxuprc* (5). |
| ProcessMgmt: Process | `nproc` | Defines the maximum number of processes that can be running simultaneously on the entire system, including remote execution processes initiated by other systems via remsh or other networking commands. See *nproc* (5). |

## Kernel Threads Parameter Summary

See also “Kernel Threads” on page 50.

| | | |
|---|---|---|
| ProcessMgmt: Threads | `max_thread_proc` | Maximum number of threads that any single process can create and have running at the same time. See *max_thread_proc* (5). |
| ProcessMgmt: Threads | `nkthread` | Maximum number of kernel threads allowed on the system at the same time. See *nkthread* (5). |

# 10 Spinlock Pool

## Spinlock Pool Parameters

The parameters related to spinlock pools for multiprocessor computers are used similarly and are documented together here. Each parameter allocates the specified number of spinlocks for the corresponding system resource:

These parameters are for use by advanced users only who have a thorough understanding of how spinlocks are used by multiple processors and how the number of spinlocks needed are related to system size and complexity. Do not change these from their default value unless you understand the consequences of any changes. In general, these values should not be altered without the advice of HP support engineers who are thoroughly familiar with their use.

Setting these parameters to inappropriate values can result in severe performance problems in multiprocessor systems.

### Acceptable Values

All of these parameters have the same minimum and maximum values. Only the defaults are different as indicated:

| | |
|---|---|
| Minimum | 64 |
| Maximum | 4096 |
| Default | 64 (`ftable_hash_locks`, `io_ports_hash_locks`) |
| Default | 128 (`bufcache_hash_locks`, `pfdat_hash_locks`, `region_hash_locks`, `sysv_hash_locks`, `vnode_hash_locks`, `vnode_cd_hash_locks`) |
| Default | 256 (`chanq_hash_locks`) |

Specify a value that is an integer exponent of 2. If you specify any other value, SAM or the kernel itself will change the parameter value to the next larger integer exponent of two (for example, specifying 100 results in the value of 128. For more information, see “How to Specify Configurable Parameter Values” on page 12.

### Description

In simple terms, spinlocks are a mechanism used in multiple-processor systems to control the interaction of processors that must be held off while waiting for another processor to finish a task so the results can be passed to the waiting processor. Spinlocks control access to file system vnodes, I/O ports, buffer cache, and various other resources.

Earlier HP-UX versions allocated a fixed number of spinlocks for all resources, but beginning with HP-UX 11.0, spinlocks can be allocated for each resource type to accommodate very large and complex systems.

In general, if the system is encountering lock contention problems that are associated with one of these hashed pools, first identify the resource spinlock pool that is associated with the contention, then increase the spinlock pool parameter for that resource.

As stated above, these parameters are for use by experienced, knowledgeable system administrators only. They should not be altered unless you are quite certain that what you are doing is the correct thing to do.

## Spinlock Pool Parameter Summary

See also “Spinlock Pool Parameters” on page 56 and “Tunables by Subsystem” on page 10.

| | | |
|---|---|---|
| Spinlock Pool | `bufcache_hash_locks` | Buffer-cache spinlock pool. NO MANPAGE. |
| Spinlock Pool | `chanq_hash_locks` | Channel queue spinlock pool. See *chanq_hash_locks* (5). |
| Spinlock Pool | `dnlc_hash_locks` | Number of locks for directory cache synchronization. NO MANPAGE. |
| Spinlock Pool | `ftable_hash_locks` | File table spinlock pool. NO MANPAGE. |
| Spinlock Pool | `hdlpreg_hash_locks` | Set the size of the pregion spinlock pool. See *hdlpreg_hash_locks* (5). |
| Spinlock Pool | `io_ports_hash_locks` | I/O port spinlock pool. NO MANPAGE. |
| Spinlock Pool | `pfdat_hash_locks` | Pfdat spinlock pool. See *pfdat_hash_locks* (5). |
| Spinlock Pool | `region_hash_locks` | Process-region spinlock pool. See *region_hash_locks* (5). |
| Spinlock Pool | `sysv_hash_locks` | System V interprocess communication spinlock pool. See *sysv_hash_locks* (5). |
| Spinlock Pool | `vas_hash_locks` | Sets the size of the vas spinlock pool. See *vas_hash_locks* (5). |
| Spinlock Pool | `vnode_cd_hash_locks` | Vnode clean/dirty spinlock pool. NO MANPAGE. |
| Spinlock Pool | `vnode_hash_locks` | Vnode spinlock pool. NO MANPAGE. |

# 11 Streams Subsystem

## Overview of Streams Kernel Parameters

A stream is an I/O pipeline through which serial data passes between the HP-UX operating system and a kernel driver associated with a raw (character-mode) device file and corresponding device such as a terminal or pty. One or more streams modules can be “plugged” or inserted into the stream to perform various functions such as data encryption or compression, character or message translation, attaching information to data packets, adding protocol data to the message, etc.

Each module in the stream performs a specific task on or associated with the data being transmitted through the pipe, and multiple modules can be pushed onto a given stream. The configurable kernel parameters for streams are used to manage the resources allocated for or consumed by character-mode I/O streams and to help prevent inappropriate pushing or other streams behaviors.

For more information about streams and streams programming, refer to the *STREAMS/UX for the HP 9000 Reference Manual*.

Note that streams I/O pipes are not the same as conventional HP-UX and UNIX file system pipelines.

## Streams Parameter Summary

The following configurable kernel parameters are used for managing resources used by character-mode I/O streams.

See also:

- “Overview of Streams Kernel Parameters” on page 60
- “How to Specify Configurable Parameter Values” on page 12
- *STREAMS/UX for the HP 9000 Reference Manual*

| | | |
|---|---|---|
| Streams | NSTREVENT | Maximum number of outstanding streams bufcalls that are allowed to exist at any given time on the system. This number should be equal to or greater than the maximum bufcalls that can be generated by the combined total modules pushed onto any given stream, and serves to limit run-away bufcalls. See *nstrevent* (5). |
| Streams | nstrpty | System-wide maximum number of streams-based pseudo-ttys that are allowed on the system. See *npty* (5). |
| Streams | NSTRPUSH | Maximum number of streams modules that are allowed to exist in any single stream at any one time on the system. This provides a mechanism for preventing a software defect from attempting to push too many modules onto a stream, but it is not intended as adequate protection against malicious use of streams. See *nstrpush* (5). |
| Streams | NSTRSCHED | Maximum number of streams scheduler daemons that are allowed to run at any given time on the system. This value is related to the number of processors installed in the system. See *nstrsched* (5). |
| Streams | STRCTLSZ | Maximum number of control bytes allowed in the control portion of any streams message on the system. See *strctlsz* (5). |
| Streams | streampipes | Force all pipes to be streams-based. See *streampipes* (5). |
| Streams | STRMSGSZ | Maximum number of bytes that can be placed in the data portion of any streams message on the system. See *strmsgsz* (5). |

# 12 Miscellaneous Parameters

## Miscellaneous Parameter Summary

The following miscellaneous configurable kernel parameters affect various, usually unrelated, HP-UX subsystems.

See also “How to Specify Configurable Parameter Values” on page 12.

### CD-ROM Parameter Summary

| | | |
|---|---|---|
| Miscellaneous: CD | `ncdnode` | Maximum number of entries in the vnode table and therefore the maximum number of open CD-ROM file system nodes that can be in memory. See *ncdnode* (5). |

### System Clock Parameter Summary

| | | |
|---|---|---|
| Miscellaneous: Clock | `dst` | Enable/disable daylight-savings time.See *timezone* (5). |
| Miscellaneous: Clock | `timezone` | The offset between the local time zone and Coordinated Universal Time (UTC), often called Greenwich Mean Time or GMT. See *timezone* (5). |

### Disk I/O Parameter Summary

| | | |
|---|---|---|
| Miscellaneous: Disk I/O | `default_disk_ir` | Immediate reporting for disk writes; whether a `write()` returns immediately after the data is placed in the disk's write buffer or waits until the data is physically stored on the disk media. See *default_disk_ir* (5). |
| Miscellaneous: Disk I/O | `o_sync_is_o_dsync` | Specifies whether an `open()` or `fcntl()` with the `O_SYNC` flag set can be converted to the same call with the `O_DSYNC` flag instead. This controls whether the function can return before updating the file access. NO MANPAGE. |

### Intrusion Detection System (IDS/9000)

| | | |
|---|---|---|
| Miscellaneous: IDS | `enable_idds` | Flag to enable the IDDS daemon, which gathers data for IDS/9000. See *enable_idds* (5). |

### Fast Symbolic Link Traversal Parameter Summary

| | | |
|---|---|---|
| Miscellaneous: Links | `create_fastlinks` | Create fast symbolic links using a newer, more efficient format to improve access speed by reducing disk block accesses during path name look-up sequences. See *create_fastlinks* (5). |

### Reserved System Memory Parameter Summary

| | | |
|---|---|---|
| Miscellaneous: Memory | `eqmemsize` | Number of pages of memory to be reserved for equivalently mapped memory, used mostly for DMA transfers. See *eqmemsize* (5). |
| Miscellaneous: Memory | `nsysmap` | Number of entries in the kernel dynamic memory virtual address space resource map (32-bit processes). See *nsysmap* (5). |
| Miscellaneous: Memory | `nsysmap64` | Number of entries in the kernel dynamic memory virtual address space resource map (64-bit processes). See *nsysmap* (5). |
| Miscellaneous: Memory | `unlockable_mem` | Amount of system memory to be reserved for system overhead and virtual memory management, that cannot be locked by user processes. See *unlockable_mem* (5). |

### Network Parameter Summary

| | | |
|---|---|---|
| Miscellaneous: Network | `tcphashsz` | TCP hash table size, in bytes. NO MANPAGE. |

### Queued Signals Parameter Summary

| | | |
|---|---|---|
| Miscellaneous: Queue | `ksi_alloc_max` | Maximum number of system-wide queued signals that can be allocated. See *ksi_alloc_max* (5). |
| Miscellaneous: Queue | `ksi_send_max` | Maximum number of queued signals that a process can send and have pending at one or more receivers. See *ksi_send_max* (5). |

### Real-Time Priority Parameter Summary

| | | |
|---|---|---|
| Miscellaneous: Schedule | `rtsched_numpri` | Number of distinct real-time interrupt scheduling priority levels are available on the system. See *rtsched_numpri* (5). |

### Terminal Parameter Summary

| | | |
|---|---|---|
| Miscellaneous: Terminal | `nclist` | Maximum number of cblocks available for data transfers through tty and pty devices. See *nclist* (5). |
| Miscellaneous: Terminal | `npty` | Maximum number of pseudo-tty entries allowed on the system at any one time. See *acctsuspend* (5). |
| Miscellaneous: Terminal | `nstrpty` | System-wide maximum number of streams-based pseudo-ttys that are allowed on the system. See *npty* (5). |
| Miscellaneous: Terminal | `nstrtel` | Number of telnet session device files that are available on the system. See *nstrtel* (5). |

| | | |
|---|---|---|
| Miscellaneous: Terminal | `scroll_lines` | Defines the number of lines that can be scrolled on the internal terminal emulator (ITE) system console. See *scroll_lines* (5). |

### Kernel Timeout Parameter Summary

| | | |
|---|---|---|
| Miscellaneous: Timeout | `ncallout` | Maximum number of timeouts that can be scheduled by the kernel at any one time. See *ncallout* (5). |

### Maximum Users Parameter Summary

| | | |
|---|---|---|
| Miscellaneous: Users | `maxusers` | Maximum number of users expected to be logged in on the system at one time; used by other system parameters to allocate system resources. See *maxusers* (5). |

### Web Server Parameter Summary

| | | |
|---|---|---|
| Miscellaneous: Web | `sendfile_max` | The amount of buffer cache that can be used by the `sendfile()` system call on HP-UX web servers. See *sendfile_max* (5). |

# 13 Obsolete Parameters

## Obsolete Parameter Summary

The following parameters are obsolete in HP-UX 11i. They are included here because they are still defined in the operating system. In general, they have no effect on the operating system.

| | | |
|---|---|---|
| Obsolete | `clicreservedmem` | OBSOLETE. See *clicreservedmem* (5). |
| Obsolete | `ndilbuffers` | OBSOLETE. See *ndilbuffers* (5). |

# A Tunable Kernel Parameters

# Table of Tunables - Tunable Order

**Table A-1 Table of Tunables - Tunable Order**

| Category | Tunable | Description |
|---|---|---|
| Accounting | `acctresume` | Resume accounting when free space on the file system where accounting log files reside rises above `acctresume` plus `minfree` percent of total usable file system size. See *acctsuspend* (5). |
| Accounting | `acctsuspend` | Suspend accounting when free space on the file system where accounting log files reside drops below `acctsuspend` plus `minfree()` percent of total usable file system size. See *acctsuspend* (5). |
| Asynchronous I/O | `aio_listio_max` | Maximum number of POSIX asynchronous I/O operations allowed in a single `lio_listio()` call. See *aio_listio_max* (5). |
| Asynchronous I/O | `aio_max_ops` | System-wide maximum number of POSIX asynchronous I/O operations allowed at one time. See *aio_max_ops* (5). |
| Asynchronous I/O | `aio_physmem_pct` | Maximum percentage of total system memory that can be locked for use in POSIX asynchronous I/O operations. See *aio_physmem_pct* (5). |
| Asynchronous I/O | `aio_prio_delta_max` | Maximum priority offset (slowdown factor) allowed in a POSIX asynchronous I/O control block (`aiocb`). See *aio_prio_delta_max* (5). |
| Memory Paging | `allocate_fs_swapmap` | Enable or disable preallocation of file system swap space when `swapon()` is called as opposed to allocating swap space when `malloc()` is called. Enabling allocation reduces risk of insufficient swap space and is used primarily where high-availabilility is important. See *allocate_fs_swapmap* (5). |
| Kernel Panic Dump | `alwaysdump` | Select which classes of system memory pages are to be dumped if a kernel panic occurs. See *alwaysdump* (5). |
| Spinlock Pool | `bufcache_hash_locks` | Buffer-cache spinlock pool. NO MANPAGE. |
| File System: Buffer | `bufpages` | Number of 4 KB pages in file system static buffer cache. See *bufpages* (5). |
| Spinlock Pool | `chanq_hash_locks` | Channel queue spinlock pool. See *chanq_hash_locks* (5). |
| Obsolete | `clicreservedmem` | OBSOLETE. See *clicreservedmem* (5). |
| IPC: Share | `core_addshmem_read` | Flag to include readable shared memory in a process core dump. See *core_addshmem_read* (5). |
| IPC: Share | `core_addshmem_write` | Flag to include read/write shared memory in a process core dump. See *core_addshmem_write* (5). |

**Table A-1 Table of Tunables - Tunable Order**

| Category | Tunable | Description |
|---|---|---|
| Miscellaneous: Links | `create_fastlinks` | Create fast symbolic links using a newer, more efficient format to improve access speed by reducing disk block accesses during path name look-up sequences. See *create_fastlinks* (5). |
| File System: Buffer | `dbc_max_pct` | Maximum percentage of memory for dynamic buffer cache. See *dbc_max_pct* (5). |
| File System: Buffer | `dbc_min_pct` | Minimum percentage of memory for dynamic buffer cache. See *dbc_min_pct* (5). |
| Miscellaneous: Disk I/O | `default_disk_ir` | Immediate reporting for disk writes; whether a `write()` returns immediately after the data is placed in the disk's write buffer or waits until the data is physically stored on the disk media. See *default_disk_ir* (5). |
| File System: Buffer | `disksort_seconds` | Maximum wait time for disk requests. NO MANPAGE. |
| Spinlock Pool | `dnlc_hash_locks` | Number of locks for directory cache synchronization. NO MANPAGE. |
| Kernel Panic Dump | `dontdump` | Select which classes of system memory pages are not to be dumped if a kernel panic occurs. See *dontdump* (5). |
| Miscellaneous: Clock | `dst` | Enable/disable daylight-savings time.See *timezone* (5). |
| Miscellaneous: IDS | `enable_idds` | Flag to enable the IDDS daemon, which gathers data for IDS/9000. See *enable_idds* (5). |
| Miscellaneous: Memory | `eqmemsize` | Number of pages of memory to be reserved for equivalently mapped memory, used mostly for DMA transfers. See *eqmemsize* (5). |
| ProcessMgmt: Process | `executable_stack` | Allows or denies program execution on the stack. See *executable_stack* (5). |
| FibreChannel | `fcp_large_config` | Define a Tachyon host bus adapter configuration as large or small. See *fcp_large_config* (5). |
| File System: Write | `fs_async` | Enable/disable asynchronous writes of file system data structures to disk. See *fs_async* (5). |
| Spinlock Pool | `ftable_hash_locks` | File table spinlock pool. NO MANPAGE. |
| Spinlock Pool | `hdlpreg_hash_locks` | Set the size of the pregion spinlock pool. See *hdlpreg_hash_locks* (5). |
| File System: Read | `hfs_max_ra_blocks` | The maximum number of read-ahead blocks that the kernel may have outstanding for a single HFS file system. See *hfs_max_ra_blocks* (5). |

**Table A-1 Table of Tunables - Tunable Order**

| Category | Tunable | Description |
|---|---|---|
| File System: Read | `hfs_max_revra_blocks` | The maximum number of reverse read-ahead blocks that the kernel may have outstanding for a single HFS file system. See *hfs_max_revra_blocks* (5). |
| File System: Read | `hfs_ra_per_disk` | The amount of HFS file system read-ahead per disk drive, in KB. See *hfs_ra_per_disk* (5). |
| File System: Read | `hfs_revra_per_disk` | The amount of memory (in KB) for HFS reverse read-ahead operations, per disk drive. See *hfs_revra_per_disk* (5). |
| File System: Read | `hp_hfs_mtra_enabled` | Enable or disable HFS multithreaded read-ahead. NO MANPAGE. |
| Spinlock Pool | `io_ports_hash_locks` | I/O port spinlock pool. NO MANPAGE. |
| Miscellaneous: Queue | `ksi_alloc_max` | Maximum number of system-wide queued signals that can be allocated. See *ksi_alloc_max* (5). |
| Miscellaneous: Queue | `ksi_send_max` | Maximum number of queued signals that a process can send and have pending at one or more receivers. See *ksi_send_max* (5). |
| ProcessMgmt: Memory | `maxdsiz` | Maximum process data storage segment space that can be used for statics and strings, as well as dynamic data space allocated by `sbrk()` and `malloc()` (32-bit processes). See *maxdsiz* (5). |
| ProcessMgmt: Memory | `maxdsiz_64bit` | Maximum process data storage segment space that can be used for statics and strings, as well as dynamic data space allocated by `sbrk()` and `malloc()` (64-bit processes). See *maxdsiz* (5). |
| File System: Open/Lock | `maxfiles` | Soft limit on how many files a single process can have opened or locked at any given time. See *maxfiles* (5). |
| File System: Open/Lock | `maxfiles_lim` | Hard limit on how many files a single process can have opened or locked at any given time. See *maxfiles_lim* (5). |
| ProcessMgmt: Memory | `maxssiz` | Maximum dynamic storage segment (DSS) space used for stack space (32-bit processes). See *maxssiz* (5). |
| ProcessMgmt: Memory | `maxssiz_64bit` | Maximum dynamic storage segment (DSS) space used for stack space (64-bit processes). See *maxssiz* (5). |
| Memory Paging | `maxswapchunks` | This parameter, multiplied by `swchunk` times the system block size value (`DEV_BSIZE`), determines the combined maximum amount of device swap and file system swap space that can be configured, system-wide. See *maxswapchunks* (5). |

**Table A-1 Table of Tunables - Tunable Order**

| Category | Tunable | Description |
|---|---|---|
| ProcessMgmt: Memory | `maxtsiz` | Maximum allowable process text segment size, used by unchanging executable-code (32-bit processes). See *maxtsiz* (5). |
| ProcessMgmt: Memory | `maxtsiz_64bit` | Maximum allowable process text segment size, used by unchanging executable-code (64-bit processes). See *maxtsiz* (5). |
| ProcessMgmt: Process | `maxuprc` | Maximum number of processes that any single user can have running at the same time, including login shells, user interface processes, running programs and child processes, I/O processes, etc. If a user is using multiple, simultaneous logins under the same login name (user ID) as is common in X Window, CDE, or Motif environments, all processes are combined, even though they may belong to separate process groups. Processes that detach from their parent process group, where that is possible, are not counted after they detach (line printer spooler jobs, certain specialized applications, etc.). See *maxuprc* (5). |
| Miscellaneous: Users | `maxusers` | Maximum number of users expected to be logged in on the system at one time; used by other system parameters to allocate system resources. See *maxusers* (5). |
| File System: LVM | `maxvgs` | Maximum number of volume groups configured by the Logical Volume Manager on the system. See *maxvgs* (5). |
| Asynchronous I/O | `max_async_ports` | System-wide maximum number of ports to the asynchronous disk I/O driver that processes can have open at any given time. See *max_async_ports* (5). |
| FibreChannel | `max_fcp_reqs` | Maximum number of concurrent FCP requests that are allowed on an FCP adapter. The default value of 512 requests can be changed by specifying a different value for this parameter. The optimal limit on concurrent requests depends on a number of factors such as configuration, device characteristics, I/O load, host memory, and other values that FCP software cannot easily determine. See *max_fcp_reqs* (5). |
| Memory Paging | `max_mem_window` | Maximum number of group-private 32-bit shared memory windows. See *max_mem_window* (5). |
| ProcessMgmt: Threads | `max_thread_proc` | Maximum number of threads that any single process can create and have running at the same time. See *max_thread_proc* (5). |
| IPC: Message | `mesg` | Enable or disable IPC messages at system boot time. See *mesg* (5). |
| IPC: Message | `msgmap` | Size of free-space resource map for allocating shared memory space for messages. See *msgmap* (5). |

Table A-1 Table of Tunables - Tunable Order

| Category | Tunable | Description |
|---|---|---|
| IPC: Message | msgmax | System-wide maximum size (in bytes) for individual messages. See *msgmax* (5). |
| IPC: Message | msgmnb | Maximum combined size (in bytes) of all messages that can be queued simultaneously in a message queue. See *msgmnb* (5). |
| IPC: Message | msgmni | Maximum number of message queues allowed on the system at any given time. See *msgmni* (5). |
| IPC: Message | msgseg | Maximum number of message segments that can exist on the system. See *msgseg* (5). |
| IPC: Message | msgssz | Message segment size in bytes. See *msgssz* (5). |
| IPC: Message | msgtql | Maximum number of messages that can exist on the system at any given time. See *msgtql* (5). |
| File System: Buffer | nbuf | System-wide number of static file system buffer and cache buffer headers. See *nbuf* (5). |
| Miscellaneous: Timeout | ncallout | Maximum number of time-outs that can be scheduled by the kernel at any one time. See *ncallout* (5). |
| Miscellaneous: CD | ncdnode | Maximum number of entries in the vnode table and therefore the maximum number of open CD-ROM file system nodes that can be in memory. See *ncdnode* (5). |
| Miscellaneous: Terminal | nclist | Maximum number of cblocks available for data transfers through tty and pty devices. See *nclist* (5). |
| File System: Open/Lock | ncsize | Inode space needed for directory name lookup cache (DNLC). NO MANPAGE. |
| Obsolete | ndilbuffers | OBSOLETE. See *ndilbuffers* (5). |
| File System: Open/Lock | nfile | Maximum number of files that can be open simultaneously on the system at any given time. See *nfile* (5). |
| File System: Open/Lock | nflocks | Maximum combined number of file locks that are available system-wide to all processes at one time. See *nflocks* (5). |
| File System: Open/Lock | ninode | Maximum number of open inodes that can be in memory. See *ninode* (5). |
| ProcessMgmt: Threads | nkthread | Maximum number of kernel threads allowed on the system at the same time. See *nkthread* (5). |
| ProcessMgmt: Process | nproc | Defines the maximum number of processes that can be running simultaneously on the entire system, including remote execution processes initiated by other systems via remsh or other networking commands. See *nproc* (5). |

**Table A-1 Table of Tunables - Tunable Order**

| Category | Tunable | Description |
|---|---|---|
| Miscellaneous: Terminal | `npty` | Maximum number of pseudo-tty entries allowed on the system at any one time. See *acctsuspend* (5). |
| Streams | `NSTREVENT` | Maximum number of outstanding streams bufcalls that are allowed to exist at any given time on the system. This number should be equal to or greater than the maximum bufcalls that can be generated by the combined total modules pushed onto any given stream, and serves to limit run-away bufcalls. See *nstrevent* (5). |
| Miscellaneous: Terminal | `nstrpty` | System-wide maximum number of streams-based pseudo-ttys that are allowed on the system. See *npty* (5). |
| Streams | `nstrpty` | System-wide maximum number of streams-based pseudo-ttys that are allowed on the system. See *npty* (5). |
| Streams | `NSTRPUSH` | Maximum number of streams modules that are allowed to exist in any single stream at any one time on the system. This provides a mechanism for preventing a software defect from attempting to push too many modules onto a stream, but it is not intended as adequate protection against malicious use of streams. See *nstrpush* (5). |
| Streams | `NSTRSCHED` | Maximum number of streams scheduler daemons that are allowed to run at any given time on the system. This value is related to the number of processors installed in the system. See *nstrsched* (5). |
| Miscellaneous: Terminal | `nstrtel` | Number of telnet session device files that are available on the system. See *nstrtel* (5). |
| Memory Paging | `nswapdev` | Maximum number of devices, system-wide, that can be used for device swap. Set to match actual system configuration. See *nswapdev* (5). |
| Memory Paging | `nswapfs` | Maximum number of mounted file systems, system-wide, that can be used for file system swap. Set to match actual system configuration. See *nswapfs* (5). |
| Miscellaneous: Memory | `nsysmap` | Number of entries in the kernel dynamic memory virtual address space resource map (32-bit processes). See *nsysmap* (5). |
| Miscellaneous: Memory | `nsysmap64` | Number of entries in the kernel dynamic memory virtual address space resource map (64-bit processes). See *nsysmap* (5). |
| Miscellaneous: Disk I/O | `o_sync_is_o_dsync` | Specifies whether an `open()` or `fcntl()` with the `O_SYNC` flag set can be converted to the same call with the `O_DSYNC` flag instead. This controls whether the function can return before updating the file access. NO MANPAGE. |
| Spinlock Pool | `pfdat_hash_locks` | Pfdat spinlock pool. See *pfdat_hash_locks* (5). |

**Table A-1 Table of Tunables - Tunable Order**

| Category | Tunable | Description |
|---|---|---|
| Spinlock Pool | `region_hash_locks` | Process-region spinlock pool. See *region_hash_locks* (5). |
| Memory Paging | `remote_nfs_swap` | Enable or disable swap to mounted remote NFS file system. Used on cluster clients for swapping to NFS-mounted server file systems. See *remote_nfs_swap* (5). |
| Miscellaneous: Schedule | `rtsched_numpri` | Number of distinct real-time interrupt scheduling priority levels are available on the system. See *rtsched_numpri* (5). |
| Miscellaneous: Terminal | `scroll_lines` | Defines the number of lines that can be scrolled on the internal terminal emulator (ITE) system console. See *scroll_lines* (5). |
| File System: SCSI | `scsi_maxphys` | Maximum record size for the SCSI I/O subsystem, in bytes. See *scsi_maxphys* (5). |
| File System: SCSI | `scsi_max_qdepth` | Maximum number of SCSI commands queued up for SCSI devices. See *scsi_max_qdepth* (5). |
| IPC: Semaphore | `sema` | Enable or disable IPC semaphores at system boot time. See *sema* (5). |
| IPC: Semaphore | `semaem` | Maximum value by which a semaphore can be changed in a semaphore “undo” operation. See *semaem* (5). |
| IPC: Semaphore | `semmap` | Size of free-semaphores resource map for allocating requested sets of semaphores. See *semmap* (5). |
| IPC: Semaphore | `semmni` | Maximum number of sets of IPC semaphores allowed on the system at any one time. See *semmni* (5). |
| IPC: Semaphore | `semmns` | Maximum number of individual IPC semaphores available to system users, system-wide. See *semmns* (5). |
| IPC: Semaphore | `semmnu` | Maximum number of processes that can have undo operations pending on any given IPC semaphore on the system. See *semmnu* (5). |
| IPC: Semaphore | `semmsl` | Maximum number of individual System V IPC semaphores per semaphore identifier. See *semmsl* (5). |
| IPC: Semaphore | `semume` | Maximum number of IPC semaphores that a given process can have undo operations pending on. See *semume* (5). |
| IPC: Semaphore | `semvmx` | Maximum value any given IPC semaphore is allowed to reach (prevents undetected overflow conditions). See *semvmx* (5). |
| Miscellaneous: Web | `sendfile_max` | The amount of buffer cache that can be used by the `sendfile()` system call on HP-UX web servers. See *sendfile_max* (5). |

**Table A-1 Table of Tunables - Tunable Order**

| Category | Tunable | Description |
|---|---|---|
| IPC: Share | `shmem` | Enable or disable shared memory at system boot time. See *shmem* (5). |
| IPC: Share | `shmmax` | Maximum allowable shared memory segment size (in bytes). See *shmmax* (5). |
| IPC: Share | `shmmni` | Maximum number of shared memory segments allowed on the system at any given time. See *shmmni* (5). |
| IPC: Share | `shmseg` | Maximum number of shared memory segments that can be attached simultaneously to any given process. See *shmseg* (5). |
| Streams | `STRCTLSZ` | Maximum number of control bytes allowed in the control portion of any streams message on the system. See *strctlsz* (5). |
| Streams | `streampipes` | Force all pipes to be streams-based. See *streampipes* (5). |
| Streams | `STRMSGSZ` | Maximum number of bytes that can be placed in the data portion of any streams message on the system. See *strmsgsz* (5). |
| File System: SCSI | `st_ats_enabled` | Flag whether to reserve a tape device on open. See *st_ats_enabled* (5). |
| File System: SCSI | `st_fail_overruns` | SCSI tape read resulting in data overrun causes failure. See *st_fail_overruns* (5). |
| File System: SCSI | `st_large_recs` | Enable large record support for SCSI tape. See *st_large_recs* (5). |
| Memory Paging | `swapmem_on` | Enable or disable pseudo-swap allocation. This allows systems with large installed memory to allocate memory space as well as disk swap space for virtual memory use instead of restricting availability to defined disk swap area. See *swapmem_on* (5). |
| Memory Paging | `swchunk` | Amount of space allocated for each chunk of swap area. Chunks are allocated from device to device by the kernel. Changing this parameter requires extensive knowledge of system internals. *Without such knowledge, do not change this parameter from the normal default value.* See *swchunk* (5). |
| Spinlock Pool | `sysv_hash_locks` | System V interprocess communication spinlock pool. See *sysv_hash_locks* (5). |
| Miscellaneous: Network | `tcphashsz` | TCP hash table size, in bytes. NO MANPAGE. |

**Table A-1 Table of Tunables - Tunable Order**

| Category | Tunable | Description |
|---|---|---|
| ProcessMgmt: CPU | `timeslice` | Maximum time a process can use the CPU until it is made available to the next process having the same process execution priority. This feature also prevents runaway processes from causing system lock-up. See *timeslice* (5). |
| Miscellaneous: Clock | `timezone` | The offset between the local time zone and Coordinated Universal Time (UTC), often called Greenwich Mean Time or GMT. See *timezone* (5). |
| Miscellaneous: Memory | `unlockable_mem` | Amount of system memory to be reserved for system overhead and virtual memory management, that cannot be locked by user processes. See *unlockable_mem* (5). |
| Spinlock Pool | `vas_hash_locks` | Sets the size of the vas spinlock pool. See *vas_hash_locks* (5). |
| Spinlock Pool | `vnode_cd_hash_locks` | Vnode clean/dirty spinlock pool. NO MANPAGE. |
| Spinlock Pool | `vnode_hash_locks` | Vnode spinlock pool. NO MANPAGE. |
| Memory Paging: Size | `vps_ceiling` | Maximum system-selected page size (in KB) if the user does not specify a page size. See *vps_ceiling* (5). |
| Memory Paging: Size | `vps_chatr_ceiling` | Maximum page size a user can specify with the `chatr` command in a program. See *vps_chatr_ceiling* (5). |
| Memory Paging: Size | `vps_pagesize` | Minimum user page size (in KB) if no page size is specified using `chatr`. See *vps_pagesize* (5). |
| File System: Journaled | `vxfs_max_ra_kbytes` | Maximum amount of read-ahead data, in KB, that the kernel may have outstanding for a single VxFS file system. See *vxfs_max_ra_kbytes* (5). |
| File System: Read | `vxfs_max_ra_kbytes` | Maximum amount of read-ahead data, in KB, that the kernel may have outstanding for a single VxFS file system. See *vxfs_max_ra_kbytes* (5). |
| File System: Journaled | `vxfs_ra_per_disk` | Maximum amount of VxFS file system read-ahead per disk, in KB. See *vxfs_ra_per_disk* (5). |
| File System: Read | `vxfs_ra_per_disk` | Maximum amount of VxFS file system read-ahead per disk, in KB. See *vxfs_ra_per_disk* (5). |
| File System: Journaled | `vx_fancyra_enabled` | Enable or disable VXFS file system read-ahead. NO MANPAGE. |
| File System: Journaled | `vx_ncsize` | Memory space reserved for VxFS directory path name cache. See *vx_ncsize* (5). |

# B Tunable Reference (Manpages)

## Manpage Layout

The manpages for tunable kernel parameters have a layout that is different from other manpages. The special headings and subheadings are described below.

NAME This section contains the name or names of the tunable and a short description. Some manpages describe more than one tunable. The names are the names of the tunables as they would appear in a *sam* (1M) display or the output of *kmtune* (1M).

VALUES This section contains subsections that describe the values that can be applied.

Default Value The value that is set on most systems when they are shipped. For most tunables, there should be no need to change the value from the default.

Allowed values The range of values that can be used for the tunable. Using a value that is outside the range specified can produce unexpected results.

Fail-safe value The value that is recommended for use when using the system in maintenance mode. Set this value when you are trying to recover from errors that have been caused by erroneous tuning of the system. If you set the tunable to this value, the system will boot at least in single-user mode without trouble caused by tunables set to improper values.

Recommended Values Recommended settings for the tunable.

DESCRIPTION This section describes what the tunable does and the information that you need to set the value of the tunable in a manner that is customized to the environment.

Under this section several questions that are relevant to the tunable are answered. The questions include:

- Who Is Expected to Change This Tunable?
- Restrictions on Changing
- When Should the Value of This Tunable Be Raised?
- What are the Side Effects of Raising the Value?
- When Should the Value of This Tunable Be Lowered?
- What are the Side Effects of Lowering the Value?
- What Other Tunable Values Should Be Changed at the Same Time?

## Tunable Manpages

The tunable manpages in this appendix are currently available only in this manual. That is, they are not installed on HP-UX systems. Similar information is available through the help system of the HP System Administration Manager (SAM); see *sam* (1M).

# Table of Contents

# Section 5

# Tunable Kernel Parameters

Table of Contents

# Section 5

# Tunable Kernel Parameters

# Table of Contents

## Section 5: Miscellaneous Topics: Tunable Kernel Parameters

## Table of Contents

## Index

# Notes

# Section 5

# Miscellaneous Topics

# Tunable Kernel Parameters

# Section 5

# Miscellaneous Topics

# Tunable Kernel Parameters

## NAME

acctsuspend, acctresume - suspend and resume accounting when available disk space reaches threshold

## VALUES

### Failsafe

Default.

### Default

acctsuspend: **2**<br>
acctresume: **4**

### Allowed values

acctresume: **0 - 101**<br>
acctsuspend: **0 - acctresume-1**

### Recommended values

acctsuspend: **2 - 6**<br>
acctresume: **4 - 10** (But more than acctsuspend)

## DESCRIPTION

The **acctsuspend** and **acctresume** tunables control when auditing stops and resumes due to disk space constraints. When free disk space on the file system being used by audit reaches the **acctsuspend** percentage, auditing is suspended until such time as the free disk space reaches the **acctresume** percentage.

### Who Is Expected to Change This Tunable?

Anyone using auditing.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

Increasing either variable should be considered when it is necessary to maintain a higher percentage of free space on the audited filesystem.

### What Are the Side Effects of Raising the Value of This Tunable?

The higher either value is, the less auditing data that is captured. The further the values are separated, the greater the amount of potentially lost audit data.

### When Should the Value of This Tunable Be Lowered?

If additional disk space is needed for auditing data, and it cannot be obtained by moving files off of the filesystem, then the value of **acctsuspend** should be lowered.

### What Are the Side Effects of Lowering the Value of This Tunable?

Filesystem performance (writing auditing records) decreases as the filesystem fills up. In turn, this will decrease the overall performance of audited processes.

### What Other Tunables Should Be Changed at the Same Time?

When changing either of these tunables, both should be considered.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**acctsuspend** and **acctresume** were developed by HP.

## SEE ALSO

accton(1M).

## NAME

aio_listio_max - the maximum number of POSIX asynchronous I/O operations that can be specified in a listio() call

## VALUES

### Minimum

**`2`**

### Maximum

**`0x10000`**

### Default

**`256`**

Specify an integer value.

## DESCRIPTION

This parameter places a limit on the system resources that can be consumed if a large number of POSIX asynchronous I/O operations are requested in a single **`listio()`**call. The value should be set large enough to meet system programming needs while protecting the system against excessive asynchronous I/O operations initiated by a malfunctioning process.

The value specified must not exceed the value of **`aio_max_ops`** (see *aio_max_ops*(5)).

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**`aio_listio_max`** was developed by HP.

## SEE ALSO

aio(5), aio_max_ops(5).

## NAME

aio_max_ops - the system-wide maximum number of POSIX asynchronous I/O operations that can be queued simultaneously at any given time

## VALUES

### Minimum

**1**

### Maximum

**0x100000**

### Default

**2048**

Specify an integer value.

## DESCRIPTION

This parameter places a limit on the system resources that can be consumed if a large number of POSIX asynchronous I/O operations are queued on the system at the same time. This parameter limits the ability of competing processes to overwhelm the system with large numbers of asynchronous I/O operations and the memory they require.

Each enqueued asynchronous operation requires allocation of system memory for its internal control structure, thus making this limit necessary. In addition to the system-wide limit, there is a per-process limit that is controlled using the argument **RLIMIT_AIO_OPS** to **getrlimit()** and **setrlimit()** calls.

**aio_listio_max** limits the number of operations that can be contained in a single **listio()** call from a given process, and the value of **aio_max_ops** must be sufficient to meet the reasonable needs of all processes that are making simultaneous or nearly simultaneous **listio()** calls without jeopardizing overall system balance.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**aio_max_ops** was developed by HP.

## SEE ALSO

aio(5), aio_listio_max(5), getrlimit(2).

## NAME

aio_physmem_pct - the maximum percentage of the total physical memory in the system that can be locked for use in POSIX asynchronous I/O operations

## VALUES

### Minimum

**5**

### Maximum

**50**

### Default

**0**

Specify an integer value.

## DESCRIPTION

This parameter places a limit on how much system memory can be locked by the combined total number of POSIX asynchronous I/O operations that are in progress at any given time. It is also important to be aware that an operation remains on the active queue and memory is not released, even if the operation is complete, until it is properly terminated by an **aio_return()** call for that operation.

Asynchronous I/O operations that use a request-and-callback mechanism for I/O must be able to lock the memory they are using. The request-and-callback mechanism is used only if the device drivers involved support it. Memory is locked only while the I/O transfer is in progress. On a large server it is better to increase **aio_physmem_pct** to higher values (up to 50).

**aio_physmem_pct** imposes a system-wide limit on lockable physical memory. A per-process lockable-memory limit can also be self-imposed by using the **setrlimit()** system call within the application program (see *setrlimit*(2)).

Remember too that the total amount of memory that can be locked at any given time for any reason, not just for asynchronous I/O, is controlled by the system-wide limit **lockable_mem**. Other system activity, including explicit memory locking with the **plock()** and/or **mlock()** interfaces can also affect the amount of lockable memory at any given time.

There is no kernel parameter named **lockable_mem**, but there is a parameter named **unlockable_mem** that affects it. The value of **lockable_mem** is determined by subtracting the value of **unlockable_mem** from the amount of system memory available after system startup. During startup, the system displays on the system console the amount of its lockable memory (along with available memory and physical memory). These values can be retrieved while the system is running with **/sbin/dmesg** command.

The amount of memory that can be locked under the limit imposed by **aio_physmem_pct** cannot exceed the total system-wide lockable-memory limit imposed by **unlockable_mem**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**aio_physmem_pct** was developed by HP.

## SEE ALSO

aio(5), aio_return(2), dmesg(1M), mlock(2), plock(2), getrlimit(2), unlockable_mem(5).

## NAME

aio_prio_delta_max - the maximum slow-down factor (priority offset) for POSIX asynchronous I/O operations

## VALUES

### Minimum

**0**

### Maximum

**20**

### Default

**20**

Specify an integer value.

## DESCRIPTION

This parameter places a limit on how much the priority of a POSIX asynchronous I/O operation can be reduced to slow it down. This is the maximum priority-offset value allowed for **aio_reqprio** in the asynchronous-I/O control block structure **aiocb**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**aio_prio_delta_max** was developed by HP.

## SEE ALSO

aio(5).

## NAME

allocate_fs_swapmap - determines when swapmap structures are allocated for filesystem swap

## VALUES

### Failsafe

**1** (on)

### Default

**1** (on)

### Allowed values

**1** (on) or **0** (off)

## DESCRIPTION

The **allocate_fs_swapmap** tunable is used to determine whether to allocate all needed filesystem swap structures during the initialization of the filesystem swap device or wait until they are needed. Pre-allocation of all filesystem swap structures upon initialization saves time during later usage of the filesystem swap, and prevents later allocation failure due to physical memory contention.

Dynamic allocation decreases the memory footprint of the filesystem swap system.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

Systems under heavy memory load may not be able to acquire enough memory to hold the swap mapping structures for an addition of file system swap. These calls to add swap would fail with the error [ENOMEM], though presumably they would be retried. Frequent failure on file system swap addition with [ENOMEM] would merit enabling this tunable.

### What Are the Side Effects of Raising the Value?

All the swap map structures for each file system swap device would be preallocated during swap initialization at kernel boot, increasing the memory footprint of the kernel. The amount of increase depends on the number and size of file system swap devices.

### When Should the Value of This Tunable Be Lowered?

Systems with limited memory available to the kernel that also have limited use of file system swap devices should disable this tunable to decrease kernel memory usage.

### What Are the Side Effects of Lowering the Value?

File system swap additions may fail with [ENOMEM] when memory contention is high.

### What Other Tunable Values Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**allocate_fs_swapmap** was developed by HP.

## NAME

alwaysdump - defines which classes of kernel memory pages are dumped when a kernel panic occurs

## VALUES

### Failsafe

**0**

### Default

**0** (Allow the kernel to choose which classes to dump.)

### Allowed values

**Integer values from 0 to 255.**

The integer value should be the sum of the integer values for the included classes as follows:

| | |
|---|---|
| **UNUSED** | 1 : Unused pages |
| **USERPG** | 2 : User pages |
| **BCACHE** | 4 : Buffer cache |
| **KCODE** | 8 : Kernel text pages |
| **USTACK** | 16 : Process stack |
| **FSDATA** | 32 : File-system Metadata |
| **KDDATA** | 64 : Kernel dynamic data |
| **KSDATA** | 128 : Kernel static data |

### Recommended values

**0** (Allow the kernel to choose which classes to dump.)

The value the kernel usually chooses is **240** = **KSDATA + KDDATA + FSDATA + USTACK**.

**KSDATA** and **KDDATA** are needed for debugging any kernel problem. **FSDATA** is needed for debugging file system problems. **USTACK** is needed for debugging problems involving user space applications. **UNUSED**, **USERPG**, **BCACHE**, and **KCODE** are not usually needed for debugging.

## DESCRIPTION

On large systems, the time required to dump system memory when a kernel panic occurs can be excessive or even prohibitive, depending on how much physical memory is installed in the system. Fast-dump capabilities controlled by the **dontdump** and **alwaysdump** parameters provide a means for restricting kernel dumps to specific types of information:

. Unused Physical Memory

. User Proces

. Buffer cache

. Kernel Code

. Process Stack

. File-System Metadata

. Kernel Dynamic Data

. Kernel Static Data

The **crashconf** command, and its associated configuration file **/etc/rc.config.d/crashconf**, control which of these memory classes are to be included in the memory dumps associated with a kernel panic. On rare occasions, the system may panic before *crashconf*(1M) is run during the boot process. On those occasions, the configuration can be set using the **alwaysdump** and **dontdump** tunables.

The bit-map value stored in **alwaysdump** specifies which of these memory classes are to be included in the memory dumps associated with a kernel panic.

The default value for this parameter is **0**. The system determines whether or not to dump classes of memory based on the type of crash that occurs, in this case.

Note that certain types of system crash, require a full crash dump. Also, the system operator may request a full crash dump at the time the dump is taken. In either of these cases, a full dump will be performed regardless of the classes selected using **alwaysdump**.

### Who Is Expected to Change This Tunable?

Only HP Field engineers should change the value of this tunable.

### Restrictions on Changing

Changes to this tunable take effect at next reboot. Use **crashconf** to change the selection of pages to take effect immediately.

### When Should the Tunable Be Turned On?

This tunable should be turned on to include certain classes of pages in the dump when there is a system crash.

### What Are the Side Effects of Turning the Tunable On?

If pages unnecessary for analyzing the dump are included, then the dump will take longer.

### When Should the Tunable Be Turned Off?

The tunable is turned off by default.

### What Are the Side Effects of Turning the Tunable Off?

The system decides the page classes that must be dumped based on the type of crash.

### What Other Tunables Should Be Changed at the Same Time?

The **dontdump** tunable should not contain the same page classes as **alwaysdump**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**alwaysdump** was developed by HP.

## SEE ALSO

crashconf(1M), dontdump(5).

## NAME

bufpages - the number of 4096-byte memory pages in the file-system buffer cache

## VALUES

### Minimum

**0** or **6** (**nbuf*2** or 64 pages)

### Maximum

Memory limited

### Default

**0**

Specify integer an value or use an integer formula expression. Use a nonzero value *only* if dynamic buffer cache is *not* being used.

## DESCRIPTION

**bufpages** specifies how many 4096-byte memory pages are allocated for the file system buffer cache. These buffers are used for all file system I/O operations, as well as all other block I/O operations in the system (**exec**, **mount**, inode reading, and some device drivers).

### Specifying a Value for bufpages

To enable dynamic buffer cache allocation, set **bufpages** to zero. Otherwise, set **bufpages** to the desired number of 4 KB pages to be allocated for buffer cache. If the value specified for **bufpages** is nonzero but less than 64, the number is increased at boot time and a message is printed, announcing the change. If **bufpages** is larger than the maximum supported by the system, the number is decreased at boot time and a message is printed.

### Related Parameters and System Values

**bufpages** interacts with **nbuf** (see *nbuf*(5)) as follows:

| bufpages | nbuf | Result |
|---|---|---|
| zero | zero | Enables dynamic buffer cache. |
| nonzero | zero | Creates **bufpages/2** buffer headers and allocates **bufpages*4KB** of buffer pool space at system boot time. |
| zero | nonzero | Allocates **nbuf*2** pages of buffer pool space and creates **nbuf** headers at boot time. |
| nonzero | nonzero | Allocates **bufpages** pages of buffer pool space and creates **nbuf** buffer headers at boot time. If the two values conflict, such that it is impossible to configure a system using both of them, **bufpages** takes precedence. |

**bufpages** controls how much actual memory is allocated to the buffer pool.

If **bufpages** is zero at system boot time, the system allocates two pages for every buffer header defined by **nbuf**. If **bufpages** and **nbuf** are both zero, the system enables dynamic buffer cache allocation and allocates a percentage of available memory not less than **dbc_min_pct** nor more than **dbc_max_pct** depending on system needs at any given time. (See *dbc_min_pct*(5) and *dbc_max_pct*(5).)

The maximum amount of memory that can be allocated to the buffer pool is also affected by the amount of memory allocated to the system for other purposes. Thus, modifying parameters that affect system memory may also affect the maximum amount of memory that can be made available to the buffer pool.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**bufpages** was developed by HP.

## SEE ALSO

dbc_max_pct(5), dbc_min_pct(5), nbuf(5).

## NAME

chanq_hash_locks - size of hashed pool of spinlocks protecting the channel queue hash tables

## VALUES

### Failsafe

**256**

### Default

**256**

### Allowed values

Any value between **16** and **16777216** that is a power of two.

### Recommended values

Default value only.

HP does NOT recommend changing this tunable from the default value. Consult your HP Field Service engineer before changing this tunable.

## DESCRIPTION

This tunable should NOT be changed. The default value is the best choice for optimal system performance.

This tunable controls the size of a pool of spinlocks (kernel data structures used for synchronization) that also protect channel queues. A very small value increases the probability of collisions in the channel queue hash tables and degrading performance. Very large values are unnecessary.

### Who Is Expected to Change This Tunable?

HP does NOT recommend changing this tunable from the default value. Consult your HP Field Service engineer before changing this tunable.

### Restrictions on Changing

This tunable should be a power of two. If another value is specified, the largest value (lower than the given value) that is a value of two is used.

### What Are the Side Effects of Raising or Lowering the Value?

The performance of the system could deteriorate.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**chanq_hash_locks** was developed by HP.

## NAME

clicreservedmem - amount of system memory (in bytes) reserved for Cluster Interconnect

## VALUES

### Failsafe

**0**

### Default

**0**

### Allowed values

**0**

### Recommended values

**0**

## DESCRIPTION

This tunable and the feature that it controlled are obsolete in the HP-UX 1li release. Please do not make any changes to this tunable as they will have no effect on the kernel. This tunable will be removed in the next release.

### Restriction on Changing

This tunable should not be modified.

## NAME

core_addshmem_read - determines the inclusion of readable shared memory in a process core dump

## VALUES

### Failsafe

**1** (on)

### Default

**1** (on)

### Allowed values

**1** (on) or **0** (off)

## DESCRIPTION

The **core_addshmem_read** tunable was added in response to a problem where a customer was limited in debugging a user-level process crash because shared memory segments were not written as part of the core file.

Upon core dump, the sections of user memory labeled as shared read-only are written (along with the normal data sections) if this tunable is set to **1** (on), and left out if it is set to **0** (off).

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect immediately.

### When Should the Value of This Tunable Be Raised?

When you wish to include read-only shared memory segments in user core files due to debugging needs by developers or maintainers on the system.

### What Are the Side Effects of Raising the Value?

User process core files would increase in most cases (most applications use at least some shared memory). This could be a problem on systems where disk space is at a premium.

### When Should the Value of This Tunable Be Lowered?

When there is no need for debugging core files where shared memory corruption or data values are in question.

### What Are the Side Effects of Lowering the Value?

Core files will tend to be smaller.

### What Other Tunable Values Should Be Changed at the Same Time?

Certainly think about **core_addshmem_write**, which does the same thing for read/write shared memory segments.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**core_addshmem_read** was developed by HP.

## NAME

core_addshmem_write - determines the inclusion of read/write shared memory in process core dump

## VALUES

### Failsafe

**1** (on)

### Default

**1** (on)

### Allowed values

**1** (on) or **0** (off)

## DESCRIPTION

The **core_addshmem_write** tunable was added in response to a problem where a customer was limited in debugging a user-level process crash because shared memory segments were not written as part of the core file.

Upon core dump, the sections of user memory labeled as shared read-write are written (along with the normal data sections) if this tunable is set to **1** (on), and left out if it is set to **0** (off).

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect immediately.

### When Should the Value of This Tunable Be Raised?

When you wish to include read-write shared memory segments in user core files due to debugging needs by developers or maintainers on the system.

### What Are the Side Effects of Raising the Value?

User process core files would increase in most cases (most applications use at least some shared memory). This could be a problem on systems where disk space is at a premium.

### When Should the Value of This Tunable Be Lowered?

When there is no need for debugging core files where shared memory corruption or data values are in question.

### What Are the Side Effects of Lowering the Value?

Core files will tend to be smaller.

### What Other Tunable Values Should Be Changed at the Same Time?

Certainly think about **core_addsmem-read**, which does the same thing for read-only shared memory segments.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**core_addshmem_write** was developed by HP.

## NAME

create_fastlinks - configure the system to use fast symbolic links

## VALUES

### Minimum

**0**

### Maximum

**1**

### Default

**0**

Specify integer value.

## DESCRIPTION

When **create_fastlinks** is nonzero, it causes the system to create HFS symbolic links in a manner that reduces the number of disk-block accesses by one for each symbolic link in a path name lookup. This involves a slight change in the HFS disk format, which makes any disk formatted for fast symbolic links unusable on Series 700 systems prior to HP-UX Release 9.0 and Series 800 systems prior to HP-UX Release 10.0 (this configurable parameter was present on Series 700 Release 9.0 systems, but not on Series 800 HP-UX 9.0 systems).

To provide backward compatibility, the default setting for **create_fastlinks** is zero, which does not create the newer, faster format. However, all HP-UX 10.0 kernels (and all Series 700 HP-UX 9.0 kernels) understand both disk formats, whether "create_fastlinks" is set to zero or non-zero.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**create_fastlinks** was developed by HP.

## NAME

dbc_max_pct - the maximum percentage of memory to be used by dynamic buffer cache

## VALUES

### Minimum

**2**

### Maximum

**90**

### Default

**50**

Specify an integer value.

## DESCRIPTION

When the parameters **bufpages** and **nbuf** are both set to their default value of 0, the size of the buffer cache grows or shrinks dynamically, depending on competing requests for system memory.

The value of **dbc_max_pct** sets the maximum percentage of physical memory that can be allocated to the dynamic buffer cache.

It is possible to set both **dbc_max_pct** and **dbc_min_pct** to the same value, 12 for example, and create a kernel that uses exactly that percentage of physical memory for the buffer cache, regardless of the size of physical memory.

### Related Parameters

**dbc_max_pct** must be greater than or equal to **dbc_min_pct**.

To use dynamic buffer caching, **bufpages** and **nbuf** must both be set to zero.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**dbc_max_pct** was developed by HP.

## SEE ALSO

bufpages(5), dbc_min_pct(5), nbuf(5).

## NAME

dbc_min_pct - the minimum percentage of memory to be used by the dynamic buffer cache

## VALUES

### Minimum

**`2`**

### Maximum

**`90`**

### Default

**`5`**

Specify an integer value.

## DESCRIPTION

During file-system I/O operations, data is stored in a buffer cache, the size of which can be fixed or dynamically allocated. When the parameters **bufpages** and **nbuf** are both set to their default values of 0, the size of the buffer cache grows or shrinks dynamically, depending on competing requests for system memory.

The value of **dbc_min_pct** specifies the minimum percentage of physical memory that is reserved for use by the dynamic buffer cache.

It is possible to set both **dbc_min_pct** and **dbc_max_pct** to the same value, 12 for example, and create a kernel that uses exactly that percentage of physical memory for the buffer cache, regardless of the size of physical memory.

### Selecting an Appropriate Value

If **dbc_min_pct** is set to too low a value, very high demand on the buffer cache can effectively hang the system. The is also true when using fixed buffer cache. To determine a reasonable (and conservative) value for the minimum cache size in megabytes, use the following formula:

**(**_number-of-system-processes_**)*(**_largest-file-system-block-size_**)/1024**

To determine the value for **dbc_min_pct**, divide the result by the number of megabytes of physical memory installed in the computer and multiply that value by 100 to obtain the correct value in percent.

Only those processes that actively use disk I/O should be included in the calculation. All others can be excluded. Here are some examples of what processes should be included in or excluded from the calculation:

**Include:**

NFS daemons, text formatters such as **nroff**, database management applications, text editors, compilers, and so on, that access or use source and/or output files stored in one or more file systems mounted on the system.

**Exclude:**

X-display applications, **hpterm**, **rlogin**, login shells, system daemons, **telnet** or **uucp** connections, and so on. These processes use very little, if any, disk I/O.

### Related Parameters

**dbc_min_pct** must be less than or equal to **dbc_max_pct**.

To use dynamic buffer caching, **bufpages** and **nbuf** must both be set to zero.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**dbc_min_pct** was developed by HP.

## SEE ALSO

hpterm(1), nroff(1), rlogin(1), telnet(1), uucp(1), bufpages(5), dbc_max_pct(5), nbuf(5).

## NAME

default_disk_ir - enable/disable the use of a device's write cache in the SCSI subsystem

## VALUES

### Failsafe

**0** (off)

### Default

**0** (off)

### Allowed values

**0** (off) or **non-zero** (on)

### Recommended values

**0** (off)

## DESCRIPTION

This tunable enables (1) or disables (0) the Immediate Reporting behavior of the SCSI subsystem, also known as Write Cache Enable (WCE). With Immediate Reporting enabled, disk drives that have data caches return from a **write()** system call, including raw writes, when the data is cached, rather than returning after the data is written to the media. This sometimes improves write performance especially for sequential transfers.

Cached data can be lost if a device power failure or reset occurs before the device writes the cached data to media. Because of these risks, the recommended value for this parameter on servers is Immediate Reporting disabled (0).

Although not an option to the mount command, this tunable has a profound effect upon filesystem and raw disk performance and, conversely, data integrity when system resets occur. This tunable also affects delayed-write versus write-through-filesystem behavior.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable will take effect when the device is first opened.

### When Should the Tunable Be Turned On?

When a third party application vendor recommends it. For normal use, HP strongly recommends you don't enable this tunable.

### What Are the Side Effects of Turning the Tunable On?

Since enabling this tunable circumvents the protections provided by LVM or RAID, there is a strong risk of filesystem corruption and data loss in case of a device power failure or reset.

### When Should the Tunable Be Turned Off?

HP recommends you always disable this tunable. This is especially true if you don't want to take the risk of filesystem corruption and data loss in case of a device power failure or reset.

### What Are the Side Effects of Turning the Tunable Off?

This might reduce the performance of disk write (not read) access while eliminating the risk of filesystem corruption and data loss in case of a device power failure or reset.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**default_disk_ir** was developed by HP.

**SEE ALSO**

write(2), scsi(7).

## NAME

dontdump - defines which classes of kernel memory pages are not dumped when a kernel panic occurs

## VALUES

### Failsafe

**0**

### Default

**0** (Allow the kernel to choose which classes to dump.)

### Allowed values

**Integer values from 0 to 255.**

The integer value should be the sum of the integer values for the excluded classes as follows:

| | |
|---|---|
| **UNUSED** | 1 : Unused pages |
| **USERPG** | 2 : User pages |
| **BCACHE** | 4 : Buffer cache |
| **KCODE** | 8 : Kernel text pages |
| **USTACK** | 16 : Process stack |
| **FSDATA** | 32 : File-system Metadata |
| **KDDATA** | 64 : Kernel dynamic data |
| **KSDATA** | 128 : Kernel static data |

### Recommended values

**0** (Allow the kernel to choose which classes to dump.)

The value the kernel usually chooses is **15** = **UNUSED + USERPG + KCODE + BCACHE**. The kernel chooses these classes because they are not usually useful in debugging kernel problems.

## DESCRIPTION

On large systems, the time required to dump system memory when a kernel panic occurs can be excessive or even prohibitive, depending on how much physical memory is installed in the system. Fast-dump capabilities controlled by the **dontdump** and **alwaysdump** parameters provide a means for restricting kernel dumps to specific types of information:

. Unused Physical Memory

. User Proces

. Buffer cache

. Kernel Code

. Process Stack

. File-System Metadata

. Kernel Dynamic Data

. Kernel Static Data

The **crashconf** command, and its associated configuration file **/etc/rc.config.d/crashconf**, control which of these memory classes are to be included in the memory dumps associated with a kernel panic. On rare occasions, the system may panic before *crashconf*(1M) is run during the boot process. On those occasions, the configuration can be set using the **alwaysdump** and **dontdump** tunables.

The bit-map value stored in **dontdump** specifies which of these memory classes are not to be included in the memory dumps associated with a kernel panic.

The default value for this parameter is **0**. The system determines whether or not to dump classes of memory based on the type of crash that occurs, in this case.

Note that certain types of system crash, require a full crash dump. Also, the system operator may request a full crash dump at the time the dump is taken. In either of these cases, a full dump will be performed regardless of the classes selected using dontdump.

### Who Is Expected to Change This Tunable?

Only HP Field engineers should change the value of this tunable.

### Restrictions on Changing

Changes to this tunable take effect at next reboot. Use **crashconf** to change the selection of pages to take effect immediately.

### When Should the Tunable Be Turned On?

This tunable should be turned on to exclude certain classes of pages from being dumped when there is a system crash. This can be done to speedup the dump.

### What Are the Side Effects of Turning the Tunable On?

If pages necessary for analyzing the dump are excluded, then the dump may not be useful for identifying the cause of the system crash.

### When Should the Tunable Be Turned Off?

The tunable is turned off by default.

### What Are the Side Effects of Turning the Tunable Off?

The system decides the page classes that must not be dumped based on the type of crash. The dump may take longer.

### What Other Tunables Should Be Changed at the Same Time?

The **alwaysdump** tunable should not contain the same page classes as **dontdump**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**dontdump** was developed by HP.

## SEE ALSO

crashconf(1M), alwaysdump(5).

## NAME

enable_idds - enable intrusion detection data source

## VALUES

### Failsafe

**0** (off)

### Default

**0** (off)

### Allowed values

**0** (off) or **1** (on)

### Recommended values

**1 (on)** if Praesidium IDS/9000 is installed,<br>
**0 (off)** otherwise.

## DESCRIPTION

If **enable_idds** is set to **1**, then Praesidium IDS/9000 can enable the collection of kernel data for intrusion detection. This also causes additional things to be tracked by the kernel, resulting in a small degradation in performance (and increase in kernel memory usage), even if Praesidium IDS/9000 is not in use.

### Who Is Expected to Change This Tunable?

Anyone using Praesidium IDS/9000.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Tunable Be Turned On?

This tunable should be turned *on* if Praesidium IDS/9000 is installed. The installation will automatically turn on **enable_idds**.

### What Are the Side Effects of Turning the Tunable On?

The name of the current working directory (and root directory) of every process is tracked, resulting in a change in memory usage and performance of the system.

### When Should the Tunable Be Turned Off?

If Praesidium IDS/9000 is not being used **enable_idds** should be turned *off*.

### What Are the Side Effects of Turning the Tunable Off?

When turned *off*, Praesidium IDS/9000 is unable to use any detection template that uses **kerndsp**. (See the documentation for Praesidium IDS/9000 for more information on **kerndsp**.)

### What Other Tunables Should Be Changed at the Same Time?

This tunable is independent of other tunables.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**enable_idds** was developed by HP.

## NAME

eqmemsize - determines the minimum size (in pages) of the equivalently mapped reserve pool

## VALUES

### Default

**15** pages

### Allowed values

**0** or any positive integer.

However, because this represents physical pages no longer available for general system use, caution is urged before using large values.

## DESCRIPTION

e

Equivalently mapped memory is a page which has the same physical and virtual address. This is useful for some applications, and for I/O. Since most memory is to be used in the traditional swapper/virtual address model, it is useful for the system to reserve some pages for this type of access at boot while the most physical memory is available.

**eqmemsize** provides the bottom value for the size of this reserved pool. The actual pool size is the sum of **eqmemsize** and a value determined dynamically at boot using the available size of physical memory, to account for large memory systems automatically. The scaled value increases the pool size by 1 page per 256 MB.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable should raised if the customer sees the error message:

```
Equivalently mapped reserve pool exhausted;
Overall application performance may be improved by
increasing the  eqmemsize" tunable parameter"
(currently set to  {X}")."
```

### What Are the Side Effects of Raising the Value?

Physical memory is reserved for this use and unavailable for the rest of the system. Increasing this tunable by a substantial amount would be roughly equivalent to removing the same amount of physical memory. It is not likely to be used for equivalent memory, and no one else can get to it either.

### When Should the Value of This Tunable Be Lowered?

In general, if the tunable is set to more than 10 pages over 15+(physical_memory/256 MB), it's probably too high. Specifically, if the system swaps heavily, and the above is true, the value should be lowered to free the physical memory back to the system.

### What Are the Side Effects of Lowering the Value?

Requests for equivalent memory may be denied. This is not a fatal error, but it should be avoided.

### What Other Tunable Values Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**eqmemsize** was developed by HP.

## NAME

executable_stack - controls whether program stacks are executable by default

## VALUES

### Failsafe

`1`

### Default

`1`

### Allowed values

`0-2`

### Recommended values

`0-2`

## DESCRIPTION

This tunable parameter controls whether program stacks are executable by default. It allows systems to be configured to have extra protection from stack buffer overflow attacks without sacrificing system performance. This class of attack very commonly attempts to trick privileged programs into performing unauthorized actions or giving unauthorized access. Background information on this type of attack is available on the web by searching for 'Smashing the Stack for Fun and Profit.'

The majority of programs that run on HP-UX do not need to execute code located on their stacks. A few programs, notably some simulators, interpreters and older versions of Java, may have a legitimate reason to execute code from their stacks. These programs typically have self-modifying code. Using a combination of this tunable and the **`+es`** option of the **`chatr`** command permits such executables to function without sacrificing protection for the rest of the system.

Refer to the 'Restricting Execute Permission on Stacks' section of the *chatr*(1) manpage for more information before changing this tunable.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect for new processes started after the change.

### When Should the Value of This Tunable Be Changed?

This tunable controls operational modes rather than data structure sizes and limits. The appropriate setting for a system depends on whether you consider security or compatibility to be most important.

A value of **`1`** is compatible with previous releases of HP-UX, but it is the least secure. This setting permits the execution of potentially malicious code located on a program's stack.

A value of **`2`** provides warnings about any program attempting to execute code on its stacks, but does not alter the program's behavior. Suspicious activity is logged in the kernel's message buffers. (See *dmesg*(1M).) This is a 'trial mode' setting intended to allow you to safely determine whether a tunable value of **`0`** would affect any legitimate application.

A tunable value of **`0`** is the recommended setting on systems where a higher level of security is important. This is essentially the same as a setting of **`2`**, but it will also terminate any process that attempts to execute code on its stacks. The process will be terminated before the potentially malicious code is executed.

### What Are the Side Effects of Changing the Value

This tunable has no effect on system behavior unless an application attempts to execute instructions located on its stacks. The majority of HP-UX applications are not programmed to do this.

### What Other Tunable Values Should Be Changed at the Same Time?

None.

## WARNINGS

None. All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**executable_stack** was developed by HP.

## NAME

fcp_large_config - define a Tachyon host bus adapter configuration as large or small

## VALUES

### Failsafe

**0** (small)

### Default

**0** (small)

### Allowed values

**0** (small) or **1** (large)

### Recommended values

**0** (small) or **1** (large)

A value of **0** defines a configuration in which an host can concurrently communicate with at most 64 ports behind any Tachyon host bus adapter. This configuration would be considered *small*.

A value of **1** defines a configuration in which a host can concurrently communicate with up to 125 ports behind any Tachyon host bus adapter. The maximum number of ports that may exist on a fibre channel arbitrated loop as defined in the fibre channel standard is 126. This configuration would be considered *large*. The fibre channel driver does not open a port to communicate with itself.

## DESCRIPTION

The **fcp_large_config** tunable is a hint to the system of how many fibre channel ports may be seen behind a Tachyon fibre channel adapter. In the *small* (default) configuration, during startup, HP-UX will allocate sufficient static resources to login concurrently to as many as 64 fibre channel ports. In a large configuration, during startup, HP-UX will allocate sufficient static resources to login concurrently to up to 125 fibre channel ports.

Only a single fibre channel login is performed from an initiator port (host) to a target port on an arbitrated loop. Multiple application *open* operations may be carried across that login session. A fibre channel logout is not performed until the last application with an outstanding *open* does a *close*.

Each initiator (host) behind the Tachyon host bus adapter attached to the fibre channel arbitrated loop through a port must be counted when determining how many ports are present.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Tunable Be Turned On?

The **fcp_large_config** tunable should be defined with a value of **1** (large) when the number of fibre channel ports behind any Tachyon fibre channel based adapter is greater than 64.

### What Are the Side Effects of Turning the Tunable On?

The side effects of turning this tunable *on* (setting the value to **1**) would be an increased consumption of memory during HP-UX operation. Static structures would be allocated during system startup, and the structures would remain allocated through the life of the system.

### When Should the Tunable Be Turned Off?

The **fcp_large_config** tunable should be defined with a value of **0** (off), when the number of fibre channel ports behind any Tachyon fibre channel based adapter is less than or equal to 64.

### What Are the Side Effects of Turning the Tunable Off?

The side effects of turning the tunable *off* (setting the value to **0**) would be the default memory utilization by the Tachyon driver.

If there are more than 64 fibre channel ports behind a host bus adapter, then the host could communicate concurrently with up to 64. If more than 64 ports exist on the fibre, then any login attempts by the host after 64 successful logins would wait until one of the other ports logged out. This could give the appearance of a hung port if applications run for long periods of time without doing *close* operations.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**fcp_large_config** was developed by HP.

## NAME

fs_async - synchronous or asynchronous writes of file-system data structures to disk

## VALUES

### Minimum

**0** (use synchronous disk writes only)

### Maximum

**1** (allow asynchronous disk writes)

### Default

**0**

Specify an integer value, **0** or **1**.

## DESCRIPTION

**fs_async** specifies whether or not asynchronous writing of file-system data structures to disk is allowed. If no value for **fs_async** is specified, synchronous writes are used.

Synchronous writes to disk make it easier to restore file system integrity if a system crash occurs while file system data structures are being updated on the file system.

If asynchronous writes are selected, HP-UX file system semantics for NFS cluster environments are preserved. In addition, files opened using **open()** with the **O_SYNC** flag (synchronous writing) will continue to be written synchronously when the asynchronous-writes feature has been configured into the kernel.

Asynchronous writes to disk can improve file system performance significantly. However, asynchronous writes can leave file system data structures in an inconsistent state in the event of a system crash. For more information about when to select synchronous or asynchronous writing, see the following tutorial.

### Tutorial: What are Synchronous and Asynchronous Writes?

If a file is open for writing and data is being written to a file, the data is accumulated in buffers and periodically written to disk. When an end-of-file condition occurs and the file is to be closed, any remaining buffer contents are written to the disk, the inode is updated with file size and block pointer information, and the file system's list of free disk blocks is updated. To ensure maximum protection of file system integrity, these operations are handled in a specific sequence that minimizes the risk of file system corruption on the disk if a system crash or power failure occurs while writing to the disk. This sequential update process is called *synchronous writing*.

HP-UX file systems store free space lists, blocks, inodes, and other file components in random and widely separate locations on disk devices. This means that writing file information blocks in a particular sequence requires additional time to move to the desired location on the disk before performing the write operation. If a power failure or system crash occurs during this sequence, one or more blocks may not be properly updated, leaving a potentially inconsistent file system. The **fsck** command is used to repair such inconsistencies.

Asynchronous writing as it relates to the **fs_async** kernel parameter allows the system to update file system information on the disk in a more convenient (hence faster) sequence rather than in a more secure (safer but slower) sequence, thus reducing search and move delays between writes. However, if a system crash occurs while these operations are being performed, the risk of an inconsistent file system that cannot be automatically repaired by **fsck** is significantly greater than with synchronous writes.

### Consequences of a Crash

If only synchronous writing is used, all updates to directories, file inodes, free space lists, and so on are handled in a sequence that is known to **fsck**. If a crash occurs while updating any disk block in the sequence, **fsck** can readily determine where the crash occurred and repair the missing update information, probably without assistance from the system administrator.

If **fs_async** is set to allow asynchronous writes and a crash occurs, **fsck** does not know what sequence was used, and thus will probably require interactive assistance from the administrator while fixing inconsistent file system information, repairing directory and inode entries, and so on.

### Why Allow Asynchronous Writes?

Waiting for synchronous writing and updating of disk blocks when closing files after writing to them degrades the performance of programs and applications that require frequent file and directory write and close operations. Allowing asynchronous writing significantly reduces those delays, producing a corresponding improvement in performance. However, when applications are CPU intensive with relatively little disk I/O, performance improvements are much lower.

### When Should I Use Asynchronous Writes?

Asynchronous writing is advisable for improving system performance if:

- Risk of power failure is low (very dependable power source and/or uninterruptible power sources).
- Precautions have been taken to enhance data security (sophisticated file system backup or redundancy strategies), or potential loss of data due to a system crash is less important than system performance.
- User applications require frequent opening, writing, and closing of disk files and directories.
- Elimination of synchronous writing would improve system performance sufficiently to offset any associated risks.

To enable asynchronous writing, set the **fs_async** kernel parameter to **1** instead of the default value of **0**.

### Related Parameters

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**fs_async** was developed by HP.

## SEE ALSO

fsck(1M), open(2).

## NAME

hdlpreg_hash_locks - determines the size of the pregion spinlock pool

## VALUES

### Default

**`128`**

### Allowed values

**`64`** to **`262144`**

## DESCRIPTION

Global kernel structures containing information on a running process or memory usage are frequently accessed or modified by several threads concurrently. To prevent race conditions, these structures are protected by spinlocks (kernel data structures used for synchronization) which allow only the spinlock 'holder' to proceed, while all others attempting to access the structure must wait.

Hashed spinlocks are used when each instance of such a data structure is to be protected, and there are several instances. Using a single spinlock for all instances would cause too much contention, but using one spinlock per structure wastes memory while the majority of the locks are unused at any given time.

By allocating a pool of hashed locks, a hash function picks one lock per group of structures, reducing contention while conserving memory. This tunable sets the size of such a pool for the **`pregion`** data structure spinlocks.

### Who Is Expected to Change This Tunable?

Only HP Field engineers should change the value of this tunable.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable should be raised on systems with both low memory contention and high lock contention due to a large number of **`pregion`** structures. Typically, such a system would be a large memory system with a large number of processes or threads.

### What Are the Side Effects of Raising the Value?

More memory is used by the kernel.

### When Should the Value of This Tunable Be Lowered?

This tunable should only be lowered on systems with high contention for physical memory and few processes and threads. The memory saved by making the table smaller will be freed for general use, and could cause contention if many **`pregions`** are present and requiring locks.

### What Are the Side Effects of Lowering the Value?

Hash contention for **`pregion`** locks is more likely, which increases the possibility of threads needing to wait on the lock shared between two or more **`pregions`**.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**`hdlpreg_hash_locks`** was developed by HP.

## NAME

hfs_max_ra_blocks - maximum number of read-ahead blocks that kernel may have outstanding for a single HFS file system

## VALUES

### Minimum

**0**

### Maximum

**128**

### Default

**8**

Specify an integer value or use an integer formula expression.

## DESCRIPTION

When data is read from a disk drive, the system may read additional data beyond that requested by the operation. This "read-ahead" speeds up sequential disk accesses by anticipating that additional data will be read and having it available in system buffers before it is requested. This parameter limits the number of read-ahead blocks that the kernel is allowed to have outstanding for any given HFS file system. The limit applies to each individual HFS file system, *not* to the system-wide total. **hfs_max_ra_blocks** and **hfs_ra_per_disk** should be adjusted according to the characteristics of the workload on the system.

h

To determine the block size of the file system containing the current directory, use the command:

**df**-g**.**

## EXAMPLES

A software development environment typically consists of small or medium sized I/Os with a fair number of disk seeks. Therefore, **hfs_max_ra_blocks** should be set to 8-to-16 blocks and **hfs_ra_per_disk** should be set to 32-to-64 kilobytes.

An out-of-core solver for an MCAE application has a significant sequential I/O component, so **hfs_max_ra_blocks** should be set to 64-to-128 blocks and **hfs_ra_per_disk** to 128-to-256 kilobytes.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**hfs_max_ra_blocks** was developed by HP.

## SEE ALSO

hfs_max_ra_blocks(5), hfs_max_revra_blocks(5), hfs_ra_per_disk(5), hfs_revra_per_disk(5), vxfs_max_ra_kbytes(5), vxfs_ra_per_disk(5)

## NAME

hfs_max_revra_blocks - maximum number of reverse read-ahead blocks that kernel may have outstanding for a single HFS file system

## VALUES

### Minimum

**0**

### Maximum

**128**

### Default

**8**

Specify an integer value.

## DESCRIPTION

When data is read from a disk drive, the system may read additional data beyond that requested by the operation. This "read-ahead" speeds up sequential disk accesses, by anticipating that additional data will be read, and having it available in system buffers before it is requested. This parameter limits the number of read-ahead blocks (when reading in the reverse direction) that the kernel is allowed to have outstanding for any given HFS file system. The limit applies to each individual HFS file system, *not* to the system-wide total.

Only HP Field Engineers should modify the **hfs_revra_per_disk** kernel parameter. Customers should not change this parameter from its default value.

To determine the block size of the file system containing the current directory, use the command:

```
df -g
```

This value should be raised when the workload is known to include frequent reverse-order sequential reading of files. This value should be lowered back to its default value if raising the value does not provide noteworthy performance improvement. Increasing this value has potential additional disk contention and performance penalty due to excess read-ahead.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**hfs_max_revra_blocks** was developed by HP.

## SEE ALSO

hfs_max_ra_blocks(5), hfs_max_revra_blocks(5), hfs_ra_per_disk(5), hfs_revra_per_disk(5), vxfs_max_ra_kbytes(5), vxfs_ra_per_disk

## NAME

hfs_ra_per_disk - amount of HFS file system read-ahead per disk drive, in KB

## VALUES

### Minimum

**0**

### Maximum

**8192**

### Default

**64**

Specify an integer value or use an integer formula expression.

## DESCRIPTION

When data is read from a disk drive, the system may read additional data beyond that requested by the operation. This "read-ahead" speeds up sequential disk accesses, by anticipating that additional data will be read, and having it available in system buffers before it is requested. This parameter specifies the amount of read-ahead permitted per disk drive.

The total amount of read-ahead is determined by multiplying **hfs_ra_per_disk** by the number of drives in the logical volume. If the file system does not reside in a logical volume, then the number of drives is effectively one.

The total number of read-ahead blocks that the kernel may have outstanding for a single HFS file system is constrained by **hfs_max_ra_blocks**.

**hfs_ra_per_disk**
and **hfs_max_ra_blocks** should be adjusted according to the characteristics of the workload on the system.

For more information, see the examples in the description of *hfs_max_ra_blocks*(5).

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**hfs_ra_per_disk** was developed by HP.

## SEE ALSO

hfs_max_ra_blocks(5), hfs_max_revra_blocks(5), hfs_ra_per_disk(5), hfs_revra_per_disk(5), vxfs_max_ra_kbytes(5), vxfs_ra_per_disk(5)

## NAME

hfs_revra_per_disk - maximum HFS file system blocks to be read in one read-ahead operation when sequentially reading backwards

## VALUES

### Failsafe

**64**

### Default

**64**

### Allowed values

**0** to **8192**

## DESCRIPTION

This tunable defines the maximum number of file system blocks to be read in one read-ahead operation when sequentially reading backwards.

### Who Is Expected to Change This Tunable?

HP Field engineers, but not customers directly.

### When Should the Value of This Tunable Be Raised?

The value of this tunable should be increased if there is a large number of reverse sequential file I/O on file systems with small file system block size.

### What Are the Side Effects of Raising the Value?

The system will consume more memory in the buffer cache.

### When Should the Value of This Tunable Be Lowered?

The value of this tunable should be lowered when there is a smaller number of reverse sequential file I/O on files systems with large file system block size.

### What Are the Side Effects of Lowering the Value?

The file throughput rate will be decreased.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**hfs_revra_per_disk** was developed by HP.

## NAME

ksi_alloc_max - system-wide limit of queued signals that can be allocated

## VALUES

### Default

**nproc * 8**

### Allowed Values

**32** through **maxint (0x7fffffff)**

## DESCRIPTION

**ksi_alloc_max** is the system-wide limit on the number of queued signals that can be allocated and in use. *ksi* stands for "kernel signal information" and identifies entries with information about queued signals. There is one per queued signal. Queued signals are used by the **sigqueue** system call, timer expiration, POSIX real time message queues, and asynchronous I/O. User generated signals (via **kill** and **raise**) are not queued.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

Frequent or heavy use of facilities which use queued signals, may require raising this value. Since the default value is based on the number of processes but the usage is thread based, a large number of threads per process which use queued signals would require a raising the tunable value. When [EAGAIN] is returned by applications using queued signals, this tunable should be raised.

### What Are the Side Effects of Raising the Value?

Increased memory usage, but only if used. Every allocated entry is 96 bytes.

### When Should the Value of This Tunable Be Lowered?

Only to control the applications' usage of queued signals.

### What Are the Side Effects of Lowering the Value?

Excessive memory consumption due to queued signals usage, and if too low, failure of applications using queued signals.

### What Other Tunable Values Should Be Changed at the Same Time?

**ksi_send_max** to limit the number of queued signals per process. Since **ksi_alloc_max** is per system based, and **ksi_send_max** is per process based, the **ksi_alloc_max** tunable should always be greater. Note that the default value for **ksi_send_max** is **32**, and **ksi_alloc_max** is **nproc * 8**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**ksi_alloc_max** was developed by HP.

## SEE ALSO

kill(2), sigqueue(2), ksi_send_max(5).

## NAME

ksi_send_max - limit on number of queued signals per process

## VALUES

### Default

**32**

### Allowed Values

**32** through **maxint (0x7fffffff)**

## DESCRIPTION

**ksi_send_max** is the per process limit on the number of queued signals that can be posted by a sender and currently pending at receivers. The enforced limit is per process sender based. *ksi* stands for "kernel signal information" and identifies entries with information about queued signals. There is one per queued signal. Queued signals are used by the **sigqueue** system call, timer expiration, POSIX real time message queues, and asynchronous I/O. User generated signals (via **kill** and **raise**) are not queued.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

Frequent or heavy use of facilities which use queued signals, may require raising this value. When [EAGAIN] is returned by the **sigqueue** system call.

### What Are the Side Effects of Raising the Value?

Increased memory usage, but only if used. Every allocated entry is 96 bytes.

### When Should the Value of This Tunable Be Lowered?

Only to control the applications' usage of queued signals.

### What Are the Side Effects of Lowering the Value?

Excessive memory consumption due to queued signals usage, and if too low, failure of applications using queued signals.

### What Other Tunable Values Should Be Changed at the Same Time?

**ksi_send_max** to limit the number of queued signals per process. Since **ksi_alloc_max** is per system based, and **ksi_send_max** is per process based, the **ksi_alloc_max** tunable should always be greater. Note that the default value for **ksi_send_max** is **32**, and **ksi_alloc_max** is **nproc * 8**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**ksi_send_max** was developed by HP.

## SEE ALSO

kill(2), sigqueue(2), ksi_alloc_max(5).

## NAME

maxdsiz, maxdsiz_64bit - maximum size (in bytes) of the data segment for any user process

## VALUES

### Default

32bit: **256 MB**

64bit: **1 GB**

### Allowed values

32 bit minimum: **0x40000**
32 bit maximum: **0xfffff000**

64 bit minimum: **0x40000**
64 bit maximum: **0x3ffbffffff**

## DESCRIPTION

User programs on HP-UX systems are composed of five discrete segments of virtual memory: text (or code), data, stack, shared, and I/O. Each segment occupies an architecturally defined range of the virtual address space that sets the upper limit to their size, but text, data and stack segments may have smaller maxima enforced by the **maxtsiz**, **maxdsiz**, and **maxssiz** tunables.

This tunable defines the maximum size of the static data storage segment for 32-bit and 64-bit processes. The data storage segment contains fixed data storage such as globals, arrays, static variables, local variables in main(), strings, and space allocated using **sbrk**() and **malloc**(). In addition, any files memory mapped as private and shared library per-invocation data also resides in the data segment.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable should be raised if user processes are receiving the [ENOMEM] error message: **exec(2): data exceeds maxdsiz** or **exec(2): data exceeds maxdsiz_64bit**. This may or may not cause a process failure depending on the program code.

### What Are the Side Effects of Raising the Value?

Raising this tunable by definition allows larger data segments for every process. The **maxdsiz** and **maxssiz** tunables limit the amount of swap space that can be reserved or used by each process, but using more virtual address space does not translate directly into using more physical address space because virtual pages can be swapped out.

Note that if swap space on the machine is near capacity, raising this tunable increases the amount of reservable swap per process. This could exhaust the swap space on the system by allowing a process with a memory leak or a malicious program that uses huge amounts of memory to reserve too much swap space.

### When Should the Value of This Tunable Be Lowered?

This tunable should be lowered if swap space is at a premium on the machine and programs that are using too much swap space are affecting the execution of other critical user processes.

### What Are the Side Effects of Lowering the Value?

If swap space on the machine is near capacity, lowering this tunable will limit the amount of swap reserved for each process and will cause the processes that consume large amounts of swap space to receive the [ENOMEM] error.

### What Other Tunable Values Should Be Changed at the Same Time?

The **maxssiz** tunable should be considered because it too limits swap usage by process stack segment.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**maxdsiz** was developed by HP.

## NAME

maxfiles - sets the soft limit for the number of files a process is allowed to have open simultaneously

## VALUES

### Minimum

**30**

### Maximum

**60000**

### Default

**60**

Specify an integer value.

## DESCRIPTION

**maxfiles** number of files a process is allowed to have open at any given time. It is possible for a process to increase its soft limit and therefore open more than **maxfiles** files.

Nonsuperuser processes can increase their soft limit until they reach the hard limit, **maxfiles_lim**.

### Related Parameters

To be useful, the value assigned to **maxfiles** must be less than the value of **maxfiles_lim**. **maxfiles_lim** is useful only if it does not exceed the limits imposed by **nfile** and **ninode**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**maxfile** was developed by HP.

## SEE ALSO

maxfiles_lim(5), nfile(5), ninode(5).

## NAME

maxfiles_lim - hard limit for the number of files that a process is allowed to have open simultaneously

## VALUES

### Minimum

**30**

### Maximum

**nfile**

### Default

**1024** Specify an integer value.

## DESCRIPTION

**maxfiles_lim** specifies the system default hard limit for the number of open files a process may have. It is possible for a nonsuperuser process to increase its soft limit, **maxfiles**, up to this hard limit.

**maxfiles_lim** can be set by rebuilding the kernel, or can be set in the running kernel with **settune()**. The System Administration Manager (**sam**) and **kmtune** use **settune()**. Dynamic changes affect all existing processes in the system except:

- Processes that are already over the new limit;
- Processes that have specifically set their limits through a call to **setrlimit()** or **ulimit()**.

### Related Parameters

To be useful, the value assigned to **maxfiles** must be less than the value of **maxfiles_lim**. **maxfiles_lim** is useful only if it does not exceed the limits imposed by **nfile** and **ninode**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**maxfile_lim** was developed by HP.

## SEE ALSO

kmtune(1M), sam(1M), setrlimit(2), settune(2), ulimit(2), maxfiles(5), maxfiles_lim(5), nfile(5), ninode(5).

## NAME

maxssiz, maxssiz_64bit - maximum size (in bytes) of the stack for any user process

## VALUES

### Default

**0x800000** (**8MB**)

### Allowed values

32 bit minimum: **0x40000**
32 bit maximum: **0x17F00000**

64 bit minimum: **0x40000**
64 bit maximum: **0x40000000**

## DESCRIPTION

User programs on HP-UX systems are composed of five discrete segments of virtual memory: text (or code), data, stack, shared, and I/O. Each segment occupies an architecturally defined range of the virtual address space which sets the upper limit to their size. However, text, data and stack segments may have a smaller maximum enforced via the **maxtsiz**, **maxdsiz** and **maxssiz** tunables.

**maxssiz** and **maxssiz_64bit** define the maximum size of the stack segment for 32-bit and 64-bit processes. The stack segment contains the actual program stack and the storage space for registers on a process or thread context switch.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

**maxssiz** should be raised if user processes are generating the console error message:

```
Warning: maxssiz value too small.
```

Processes generating this error message will likely terminate with the segmentation violation error [SIGSEGV] and dump core.

### What Are the Side Effects of Raising the Value?

Raising this tunable by definition allows larger stack segments for every process. This means that **maxdsiz** and **maxssiz** function as limitations on the amount of swap space which can be reserved or used by each process. Therefore, using more virtual address space does not translate directly to using more physical address space because virtual pages can be swapped out.

If swap space on the machine is near capacity, raising this tunable will increase the amount of swap that is reservable per process. This would allow a process with a memory leak, or written to use a great deal of memory, to reserve more swap, and possibly exhaust the swap space on the system.

### When Should the Value of This Tunable Be Lowered?

This tunable should be lowered if swap space is at a premium on the system, and several poorly written programs, or malicious programs, are using swap space to the detriment of more critical user processes. For example, several students at a university might execute less than production level code and leak memory all over the place.

### What Are the Side Effects of Lowering the Value?

If swap space on the machine is near capacity, lowering this tunable will limit the amount of swap that is reservable, per process. This could cause processes with large stack requirements to terminate with a [SIGSEGV] error.

### What Other Tunable Values Should Be Changed at the Same Time?

**maxdsiz** should be considered because it performs a similar function of limiting swap usage by process data segment.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**maxssiz** was developed by HP.

## SEE ALSO

maxdsiz(5), maxtsiz(5).

## NAME

maxswapchunks - maximum number of swap chunks which can be managed by swap system

## VALUES

### Default

**256** chunks

### Allowed values

Minimum: **1** chunk

Maximum: **16384** chunks

A swap chunk is a collection of 1 KB blocks (the exact size is determined by the tunable **swchunk**).

## DESCRIPTION

Swap space in the kernel is managed using 'chunks' of either physical device space or file system space. These chunks are used as the indices on the first layer of the two-layer swap table. Address offsets are then used to find the position within the current chunk, providing the second layer of indexing. This two-layer schema allows the swap table to be kept relatively small, as opposed to a large table indexed directly by swap page.

**maxswapchunks** controls the size of the global swap table, setting the upper limit on the number of swap chunks accessible by the system.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable should be raised if more swap is present than the present table can accommodate. The kernel checks for this condition when adding swap (either on boot or during runtime) and prints a console message:

```
Unable to add all swap from <device or file system>.
Increase the tunable parameter maxswapchunks by < X >
and re-configure your system.
```

All the swap possible (however many swap chunks are left in the table) will be added to the system in the above scenario.

If no more swap will be added after the device triggering this message, adding X as stated in the message (presuming this is still below the maximum of 16384) will suffice. If more swap will be added, the customer should add sufficiently to accommodate all the needed chunks.

To calculate the number of needed chunks, divide the device size by the value of tunable **swchunkS** * 1024 bytes.

If the maximum of **maxswapchunks** is reached, but **swchunkS** is below the maximum, **swchunkS** should be raised and the required **maxswapchunks** should be recalculated.

### What Are the Side Effects of Raising the Value?

The kernel will use more memory for the larger swap table.

### When Should the Value of This Tunable Be Lowered?

The tunable should only be lowered if it is significantly higher than the amount of chunks which will be used on the system, and physical memory is at a premium.

### What Are the Side Effects of Lowering the Value?

Future swap additions may partially fail due to insufficient chunks as previously described.

### What Other Tunable Values Should Be Changed at the Same Time?

**swchunk** should be considered when modifying **maxswapchunks**. **swchunk** sets the size of a swap chunk in 1024 byte pages, so increasing the chunk size yields less chunks needed in the table (and a smaller needed maxswapchunks) for a given swap space. Similarly, lowering swchunk means that more chunk indices in the swap table will be needed.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**maxswapchunks** was developed by HP.

## SEE ALSO

swchunk(5).

## NAME

maxtsiz, maxtsiz_64bit - maximum size (in bytes) of the text segment for any user process

## VALUES

### Default

32 bit: **256MB**
64 bit: **1GB**

### Allowed values

32 bit minimum: **0x40000**
32 bit maximum: **0x40000000**

64 bit minimum: **0x40000**
64 bit maximum: **0x400000000000**

## DESCRIPTION

User programs on HP-UX systems are composed of five discrete segments of virtual memory: text (or code), data, stack, shared, and I/O. Each segment occupies an architecturally defined range of the virtual address space which sets the upper limit to their size, but text, data and stack segments may have a smaller maximum enforced via the **maxtsiz**, **maxdsiz** and **maxssiz** tunables.

**maxtsiz** controls the size of the text segment, which is the read-only executable object code for the process that can be shared by multiple processes executing the same program. For example, all copies of vi on the system use the same text segment.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect immediately.

### When Should the Value of This Tunable Be Raised?

**maxtsiz** should be raised if user processes are receiving the [ENOMEM] error with the following message:

```
exec(2): text exceeds maxtsiz
```

or

```
exec(2): text exceeds maxtsiz_64bit
```

### What Are the Side Effects of Raising the Value?

Raising this tunable allows larger text segments for every process. This generally does not have side effects unless the program is executing from a remote nfs drive and the **page_text_to_local** tunable is set to TRUE because the program instructions are not swapped to a device, but simply cleared from physical memory and re-read from the source file on the disk under memory pressure.

### When Should the Value of This Tunable Be Lowered?

This tunable should be lowered if you want to limit the text size of running processes, but there is not a system performance reason to do so.

### What Are the Side Effects of Lowering the Value?

None.

### What Other Tunable Values Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**maxtsiz** was developed by HP.

## SEE ALSO

maxdsiz(5), maxssiz(5).

## NAME

maxuprc - limits the maximum number of user processes per user

## VALUES

### Failsafe

**75**

### Default

**75**

### Allowed values

Between **3** and (**nproc -5**).

### Recommended value

**75**

## DESCRIPTION

**maxuprc** is a dynamic tunable that limits the maximun number of users processes per user. Only root can have more than the number of user processes limited by **maxuprc**.

### Who is Expected to Change This Tunable?

System administrators can change the value of **maxuprc** depending on the usage of the system.

### Restrictions on Changing

None.

### When Should the Value of This Tunable Be Raised?

The value of **maxuprc** should be changed if users require more processes than what they are currently allowed by **maxuprc**. If **fork()** fails with an error value of [EAGAIN], it could be an indication that **maxuprc** was reached by that particular user. However, that is not the only reason that could cause **fork()** to fail with an errno value of [EAGAIN].

### What Are the Side Effects of Raising the Value?

Raising the value of **maxuprc** allows a single user to consume a greater percentage of the system.

### When Should the Value of This Tunable Be Lowered?

The value of **maxuprc** should be lowered when you feel that limiting the maximum number of user processes is necessary.

### What Are the Side Effects of Lowering the Value?

Applications that depend on a large number of processes may behave differently or fail.

### What Other Tunable Values Should Be Changed at the Same Time?

When tuning **maxuprc** you should keep in mind the values of **nproc** and **maxusers**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**maxuprc** was developed by HP.

## SEE ALSO

maxusers(5), nproc(5).

## NAME

maxusers - expected number of simultaneous users on the system

## VALUES

### Failsafe

**32**

### Default

**32**

### Allowed values

**0**

Limited by physical memory. Maximum physical memory is the main limiting factor because kernel data structures that use up memory are controlled by parameters whose values are derived from **maxusers**. The other limits to the range of acceptable values for **maxusers** is the range allowed values of the other tunables that it is related to by expressions.

### Recommended values

**32**

## DESCRIPTION

The **maxusers** tunable is used to calculate the default values of **nclist**, **nproc**, **nfile** and **ninode**. All of which control kernel data structures that determine the system resource allocation. **maxusers** does not affect the actual number of users who can log on. **maxusers** provides a convenient, single point mechanism to control many tunables and optimize memory usage. It provides a form of coarse tuning for the system. The individual tunables should be changed for finer tuning. **maxusers** is not used within the kernel. It is only used when calculating the values of all the other tunables during the kernel build process.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot. It should be noted that changes to **maxusers** affects the values of all the tunables to which it is related by expressions. The value of **maxusers** may not be set such that it makes the value of any of the related tunables invalid.

### When Should the Value of This Tunable Be Raised?

The value of **maxusers** should be raised to accommodate an increase in the number of concurrent users to reflect the corresponding change in resource allocation.

### What are the Side Effects of Raising the Value?

The side effect is that all related kernel structures grow and hence consume more memory.

### When Should the Value of This Tunable Be Lowered?

The value of **maxusers** may be lowered to reduce the size of the kernel. It may also be done to provide resource allocation for a fewer number of users.

### What are the Side Effects of Lowering the Value?

The kernel becomes smaller in size, saving memory. If reduced, the kernel structures become smaller and this increases the danger of application failures caused by a lack of resources.

### What Other Tunable Values Should Be Changed at the Same Time?

None. It should be noted that the tunables whose value is based on expressions containing **maxusers** are all being changed with it. The default values of the tunables **nclist**, **nfile**, **ninode** and **nproc** are related directly to **maxusers**. The defaults of some other tunables are indirectly related to **maxusers**. The default values of **ksi_alloc_max**, **ncallout**, **nkthread** and **nsysmap[64]** are also affected by changes to **maxusers** as they are dependent on tunables whose values are derived from **maxusers**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**maxusers** was developed by HP.

## SEE ALSO

ksi_alloc_max(5), ncallout(5), nclist(5), nfile(5), ninode(5), nkthread(5), nproc(5).

## NAME

maxvgs - maximum number of LVM Volume Groups that can be created/activated on the system

## VALUES

### Failsafe

**10**

### Default

**10**

### Allowed values

**1-256**

### Recommended values

**10-256**

The value for **maxvgs** should be no larger than the anticipated needs of the system. Resources are reserved for the number of volume groups indicated, so setting an unnecessarily large **maxvgs** value on systems with limited resources is not advisable. Conversely, it does not make sense to set **maxvgs** to a low value since it will likely have to be raised later when a greater number of volume groups is utilized.

## DESCRIPTION

Maximum number of LVM Volume Groups which may be created or activated on the system. This value also limits the range of Volume Group numbers allowed. A **maxvgs** value of 10, for example, allows 10 Volume Groups to be created on the system, and the Volume Group number of each Volume Group is a unique value in the range of 0 to 9.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

When **maxvgs** is modified the change takes effect at the next system reboot.

### When Should the Value of This Tunable Be Raised?

A Volume Group cannot be created or activated if its volume group number exceeds **maxvgs**. If this occurs, the customer should choose a volume group number that does not exceed **maxvgs**, or raise the **maxvgs** value to accommodate the volume group number chosen.

### What Are the Side Effects of Raising the Value of This Tunable?

Additional kernel resources are consumed. Approximately 4-8KB of lockable physical memory is consumed by advancing the **maxvgs** value by 1.

### When Should the Value of This Tunable Be Lowered?

On systems with very limited kernel memory, lowering **maxvgs** to 10 or less may free memory for other uses. Lower the **maxvgs** value only when it is certain that the number of Volume Groups to be created or activated on the system will not exceed the new value.

### What Are the Side Effects of Lowering the Value of This Tunable?

If **maxvgs** is set to too small a value, it is possible that some volume groups cannot be created or activated.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**maxvgs** was developed by HP.

## SEE ALSO

lvm(7).

## NAME

max_async_ports - maximum number of asynchronous disk ports that can be open at any time

## VALUES

### Failsafe

**1**

### Default

**50**

### Allowed values

**1** - **4294967295** (2ˆ64 - 1) (limited by available physical memory)

### Recommended values

There is no recommended value for this tunable. The number of processes that will be accessing the driver simultaneously can be used as a guideline to set a value.

## DESCRIPTION

The asynchronous disk driver provides an interface to execute high-performance I/O to the raw disk. Instead of reading or writing directly to the raw disk, a process writes requests to the asynchronous driver which then makes appropriate calls to the disk driver.

Every process that opens the asynchronous disk driver is assigned a port. The driver uses the port to keep track of I/O and other internal resources for that process. The number of opens to the asynchronous disk driver is limited by the number of available ports.

The value of this tunable should not be set greater than the value of **nproc** since such a value cannot not be used.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

When the number of processes opening the driver needs to be more than the current value of the tunable.

### What Are the Side Effects of Raising the Value of This Tunable?

The resources for the port are allocated from the kernel memory. Setting a higher value would result in the driver using more kernel memory. Other kernel components which may need kernel memory could suffer.

### When Should the Value of This Tunable Be Lowered?

When the system is low on physical memory and there are not as many opens necessary to access the driver.

### What Are the Side Effects of Lowering the Value of This Tunable?

The number of simultaneous opens are limited by this tunable. This may impact the performance of applications that need to have a higher number of processes accessing the driver.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**max_async_ports** was developed by HP.

## SEE ALSO

nproc(5).

## NAME

max_fcp_reqs - maximum number of concurrent fibre channel requests outstanding on a Tachyon host bus adapter

## VALUES

### Default

**512**

### Allowed values

**0-1024**

### Recommended values

**0-1024**

While any value in the specified range is acceptable, the value specified will be rounded up to the next higher power of 2 to determine the final value.

## DESCRIPTION

The tunable **max_fcp_reqs** is used to define the maximum number of fibre channel requests that may be outstanding on any Tachyon-based host bus adapter at any one time. Fibre channel requests are the equivalent to I/O operations.

During startup of HP-UX, internal tables are constructed based on the value provided. The larger the value, the more memory that will be consumed for the tables. In all but very extreme situations, the default value indicated is sufficient for construction of the internal tables.

If at any time more fibre channel requests are available than can be put on the fibre, then the remaining requests must wait until a table entry frees up. Table entries are freed as outstanding requests are completed.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If the internal tables should ever fill, this may indicate that the tunable value should be raised. The message **SEST depleted - performance may be degraded.** will be output to the system diagnostic log if a situation ever arises where any fibre channel request must wait for entries to become available in the internal tables.

A check is made every hour to determine if the internal tables have filled within the previous hour. If so, the message is printed. A single occurrence of the message, or a few occurrences over weeks should not be interpreted as an explicit request to increase the value. The occurrence of the message 2 or more times a day would be an indication to increase the value.

### What Are the Side Effects of Raising the Value of This Tunable?

Raising the value of this tunable will increase the amount of memory used by the HP-UX kernel to construct internal tables. Additionally, raising the value will mean that more fibre channel requests can be outstanding on any given host bus adapter at any time. The implication here is that the number of completions may also increase. This could bog down the system SPU processing the completion interrupts, or if internal host bus adapter queues were to fill, completions could be lost and the original requests would need to be retried. Data will NOT be lost, the operation transferring the data will be retried.

### When Should the Value of This Tunable Be Lowered?

The tunable should be lowered only on systems where memory resources are tight (memory allocations are not always successful).

### What Are the Side Effects of Lowering the Value of This Tunable?

The side effects of lowering the value of this tunable would be that some fibre channel requests would wait until internal table entries became available before being able to proceed. For these cases, performance would be adversely affected.

If lowering the value of the tunable results in fibre channel requests waiting for internal resources to become available, the following message would appear in the diagnostic log:

```
SEST depleted - performance may be degraded.
```

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**max_fcp_reqs** was developed by HP.

## NAME

max_mem_window - maximum number of group-private 32-bit shared memory configurable by users

## VALUES

### Default

**0**

### Allowed values

Minimum: **0** shared memory windows

Maximum: **8192** shared memory windows

## DESCRIPTION

Processes in a PA-RISC 32-bit architecture usually share the global quadrants 3 and 4 for such things as I/O mappings, shared libraries, shared mapped files, etc. However, you may wish to use quadrant 3 in a more limited sharing fashion with only a select group of processes. Memory windows allow this functionality.

If this tunable is set to **0**, 32-bit programs always use the global Q3 and Q4 for shared memory. If this tunable is greater than **0** processes in defined groups use a Q3 private to that group, and then the globally shared Q4 if Q3 is full.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable should only be raised if more group-private Q3 areas are needed by the system users. If more areas are not needed, there is literally no point to raising this value.

### What Are the Side Effects of Raising the Value?

A little more memory is used in the kernel.

### When Should the Value of This Tunable Be Lowered?

If process groups are not being used, or not being used in the quantity set by the tunable, there is really no point to having this value higher than what is actually needed.

### What Are the Side Effects of Lowering the Value?

The kernel uses a little less memory. If the value is set to **0**, no group-private Q3's are allowed.

### What Other Tunable Values Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**max_mem_window** was developed by HP.

## NAME

max_thread_proc - defines the maximum number of threads allowed per process

## VALUES

### Failsafe

**64**

### Default

**64**

### Allowed values

Between **64** and **nkthread**

### Recommended values

**64**

## DESCRIPTION

**max_thread_proc** is the maximum number of threads allowed per process on the system. When tuning **max_thread_proc**, the maximum number of threads allowed per process will be the new value of **max_thread_proc**. No process will be able to create new threads so the total count of its threads exceeds **max_thread_proc**.

### Who is Expected to Change This Tunable?

System administrators that run applications on their systems requiring a high number of threads per process.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

The value of **max_thread_proc** should be raised if there is a need to run an application that requires a higher number of threads per process than what **max_thread_proc** is currently set to accommodate. Also, if thread creation fails with the **errno** value of [EAGAIN], then it could indicate that the **max_thread_proc** has been reached. However, reaching **max_thread_proc** is not the only reason why a thread creation could fail with [EAGAIN]. [EAGAIN] may be returned if the **nkthread** system tunable has been reached or memory on the system is exhausted.

### What Are the Side Effects of Raising the Value?

A group of processes may be able to exhaust the system wide limit of threads on the system.

### When Should the Value of This Tunable Be Lowered?

If you feel that it is necessary to limit the number of threads allowed per process.

### What Are the Side Effects of Lowering the Value?

Some applications which need a high number of threads may behave differently or fail to operate.

### What Other Tunable Values Should Be Changed at the Same Time?

You may want to look at the value of **nkthread**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**max_thread_proc** was developed by HP.

## SEE ALSO

nkthread(5).

## NAME

mesg - enable or disable System V IPC messages at boot time

## VALUES

### Failsafe

**1** (on)

### Default

**1** (on)

### Allowed values

**1** (on) or **0** (off)

## DESCRIPTION

The **mesg** tunable is a "switch" designed to enable the entire System V IPC message subsystem. It not only allows user access, but when disabled, memory space will *not* be reserved on system boot, and the System V messaging system will be totally removed. However, the default state is enabled so the system exists after the first cold install. Considering the minimal impact for the default tunable values, a disabled message system will be the exception.

### Overview

A System V message is just a sequence of bytes that can be passed between cooperating processes via a message queue. Messages can be of any length (up to a kernel-tunable maximum) and can be "typed" with a process-specified number. Each message queue has a unique ID but can contain any mixture of message types. The process receiving a message can get the "first" message (FIFO), the first of a specified type, the first of a group of types or even wait for a specific type to appear. Message passing between processes is supported by shared memory and globally defined data structures which each process must use.

Message queues are implemented as linked lists in shared memory. Each message in a queue consists of an ID, a header (links, type, size and address) and one or more contiguous "segments" of reserved memory space. As messages are added and removed, the message resource map tracks available free space. The size of the resource map will help determine the potential fragmentation of the reserved shared memory.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### What Are the Side Effects of Turning the Tunable Off?

Memory space will *not* be reserved on system boot, and the System V messaging system will be totally removed.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V IPC Message subsystem tunables are interrelated and should *not* be treated as independent variables. The set of eight tunables must be evaluated as a system to ensure they reflect the application requirements. The **mesg** tunables include **msgmap**, **msgmax**, **msgmnb**, **msgmni**, **msgseg**, **msgssz**, and **msgtql.**

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**mesg** was developed by AT&T.

## SEE ALSO

msgmap(5), msgmax(5), msgmnb(5), msgmni(5), msgseg(5), msgssz(5), msgtql(5).

## NAME

msgmap - number of entries in the System V IPC message space resource map

## VALUES

### Failsafe

**`(2+msgtql)`**

### Default

**`(2+msgtql)`**

### Allowed values

Minimum: **`3`**

Maximum: **`msgtql+2`** or **`msgtql+2`**, whichever is lower.

## DESCRIPTION

The **`msgmap`** tunable specifies the size of the message space resource map that tracks the free space in shared IPC message space. Each resource map entry is an offset-space pair which points to the offset and size (bytes) of each contiguous series of unused message space "segments".

Message queues are implemented as linked lists in shared memory. Each message consists of one or more contiguous segments of reserved memory space. As messages are added and removed, the reserved shared memory area can become fragmented. Free-space fragmentation generally increases as message size variation increases. Since the resource map requires an entry pair for each fragment of free space, excess fragmentation can cause the free-space map to completely fill. If the map is full when message space is acquired or released, the system issues the **`DANGER: mfree map overflow.`** warning message.

For more information about System V message queues, refer to the "Overview" section of the *mesg*(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

When the free-space map is full and the system issues the warning message **`DANGER: mfree map overflow.`** A larger value for **`msgmap`** can correct this issue.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V message queue tunables are interrelated and should *not* be treated as independent variables. The set must be evaluated as a system to ensure the tunables reflect the application requirements. The **`mesg`** tunables include **`msgmap`**, **`msgmax`**, **`msgmnb`**, **`msgmni`**, **`msgseg`**, **`msgssz`**, and **`msgtql`**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**`msgmap`** was developed by AT&T.

## SEE ALSO

mesg(5), msgmax(5), msgmnb(5), msgmni(5), msgseg(5), msgssz(5), msgtql(5).

## NAME

msgmax - maximum System V IPC message size in bytes

## VALUES

### Default

**8192**

### Failsafe

**8192**

### Allowed values

Minimum: **0** or **msgmnb**

Maximum: **64*1024*1024** or (**msgssz*msgseg**)

## DESCRIPTION

The **msgmax** tunable specifies the maximum allowable size, in bytes, of any single message in a System V message queue. **msgmax** limits the required number of contiguous bytes of storage for any given message. Obviously, **msgmax** must be no larger than **msgmnb** (the size of a queue) nor can it be larger than the preallocated system-wide message storage space. Possible strategies for deciding this tunable value include setting it equal to **msgmnb** (one message per queue) or some value slightly larger than the biggest expected message which is less than **msgmnb**.

The purpose behind the **msgmax** limit is protection from malicious or poorly written applications taking excessive message buffer space. Any **msgsnd()** system call that attempts to send a message larger than **msgmax** bytes will return [EINVAL].

For more information about System V message queues, refer to the "Overview" section of the *mesg*(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

**msgmnb** is dependent on the need to use single large messages which may fill the queue. Increases are limited to **magmnb** (the maximum bytes on a queue) if requiring larger single message sizes. It has no direct impact on memory space unless actually used.

### When Should the Value of This Tunable Be Lowered?

If applications are only using smaller message sizes and greater protection from errant applications is needed.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V message queue tunables are interrelated and should *not* be treated as independent variables. The set must be evaluated as a system to ensure the tunables reflect the application requirements. The **mesg** tunables include **msgmap**, **msgmax**, **msgmnb**, **msgmni**, **msgseg**, **msgssz**, and **msgtql**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**msgmax** was developed by AT&T.

## SEE ALSO

mesg(5), msgmap(5), msgmnb(5), msgmni(5), msgseg(5), msgssz(5), msgtql(5).

## NAME

msgmnb - maximum number of bytes on a single System V IPC message queue

## VALUES

### Default

**16384**

### Failsafe

**16384**

### Allowed Values

Minimum: **0** or **msgmnb**

Maximum: **64*1024*1024** or (**msgssz*msgseg**).

## DESCRIPTION

The **msgmnb** tunable specifies the maximum allowable total combined size, in bytes, of all messages queued in a single given System V IPC message queue at any one time. Each of the **msgmni** number of message queues in the system has the same limitations and **msgmnb** sets the maximum number of message bytes that queue can store. However, other tunables, like **msgmax**, **msgtql**, **msgssz**, and **msgseg**, influence how that space is utilized and even if all of that space can be utilized. **msgmni** is only an upper bound for any given queue.

If the granularity of message storage, as set by **msgssz** is large compared with the message size, a larger fraction of queue space cannot be used. The same effect exists for messages whose sizes are a large fraction of the queue space. Finally, **msgmnb** may have no effect at all with few messages of smaller sizes spread over a large number of queues.

Note that if a **msgsnd()** system call attempts to exceed the limit imposed by **msgmnb**, it will return [EAGAIN] if **IPC_NOWAIT** is set. If **IPC_NOWAIT** is not set, it will return [EINTR].

For more information about System V message queues, refer to the "Overview" section of the *mesg*(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

**msgmnb** must be grater than or equal to the largest queue required. If single large messages are expected, then it must also be grater than or equal to the largest single message expected.

Raise the value if requiring more total messages on any queue or if requiring very large single message that would fill a queue. It has no direct impact in memory space unless actually used.

### When Should the Value of This Tunable Be Lowered?

If applications are only using smaller message sizes and greater protection from errant applications is needed.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V message queue tunables are interrelated and should *not* be treated as independent variables. The set must be evaluated as a system to ensure the tunables reflect the application requirements. The **mesg** tunables include **msgmap**, **msgmax**, **msgmnb**, **msgmni**, **msgseg**, **msgssz**, and **msgtql**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**msgmnb** was developed by AT&T.

## SEE ALSO

mesg(5), msgmap(5), msgmax(5), msgmni(5), msgseg(5), msgssz(5), msgtql(5).

## NAME

msgmni - maximum number of system-wide System V IPC message queues (IDs) allowed

## VALUES

### Default

**50**

### Failsafe

**50**

### Allowed Values

Grater than or equal to **1**.

The upper limit is not "range" specified.

## DESCRIPTION

The **msgmni** tunable specifies the maximum number of system-wide System V IPC message queues identifiers (one per queue). Each message queue created has an identifier (ID) and there is an upper limit of **msgmni** identifiers. Applications use the **msgget()** system call to create new queues and use the queue's identifier returned by the **msgget()** in the **msgctl(IPC_RMID)** call or **ipcrm** command line to destroy the queue. **msgmni** should be limited to a value which does not greatly exceed application requirements. While the upper limit is not "range" specified in the **./pa/master/sysv-msg** file, the value is internally saved in an "int" field of a structure.

If all IDs have been used, the **msgget()** system call will return [ENOSPC]. Processes using **msgget()** to allocate messages should use the **IPC_RMID** command with **msgctl()** to remove that space. If a process acquires message space but the process terminates without releasing that space, the messages (and the space being used) remains. Abandoned message space can be removed with the **ipcrm** command. Message queue status can be monitored using the **ipcs** command.

For more information about System V message queues, refer to the "Overview" section of the *mesg*(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If requiring additional queues.

### When Should the Value of This Tunable Be Lowered?

If the required number of queues has diminished, or if greater protection from errant applications is needed. It has minimal impact on memory space.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V message queue tunables are interrelated and should *not* be treated as independent variables. The set must be evaluated as a system to ensure the tunables reflect the application requirements. The **mesg** tunables include **msgmap**, **msgmax**, **msgmnb**, **msgmni**, **msgseg**, **msgssz**, and **msgtql**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**msgmni** was developed by AT&T.

## SEE ALSO

mesg(5), msgmap(5), msgmax(5), msgmnb(5), msgseg(5), msgssz(5), msgtql(5).

## NAME

msgseg - number of System V IPC message segments in the system

## VALUES

### Default

**2048**

### Failsafe

**2048**

### Allowed Values

Minimum: **1**

Maximum: **32767**

Must be less than 32768 so that if 32M were allocated, the segment size **msgssz** would need to be set to 1K bytes.

## DESCRIPTION

The **msgseg** specifies the total number of "segments" of system-wide shared memory message storage space which is shared among all IPC message queues. The potential for fragmentation should be considered. The total available space for messages in the system is defined by the product of **msgseg*msgssz** or, the number of segments multiplied by the segment size. Changing the ratio between these two tunables changes the granularity for message storage and therefore the fragmentation to be expected for any given message usage pattern. Note that the size of a segment is defined by **msgssz**.

**msgssz*msgseg** is the total message storage space. Set **msgssz*msgseg** equal to or grater than the maximum a single large message queue can hold. If many large queues are expected, that will need to be factored in.

For more information about System V message queues, refer to the "Overview" section of the *mesg*(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If increasing **msgssz** (the size of segments) will cause the total message space (**msgssz*msgseg**) to exceed an unsigned 32-bit value.

### When Should the Value of This Tunable Be Lowered?

**msgssz*msgseg** must remain greater than **msgmnb** and **msgmax**. It has direct effect on reserved message space.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V message queue tunables are interrelated and should *not* be treated as independent variables. The set must be evaluated as a system to ensure the tunables reflect the application requirements. The **mesg** tunables include **msgmap**, **msgmax**, **msgmnb**, **msgmni**, **msgseg**, **msgssz**, and **msgtql**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**msgmni** was developed by AT&T.

## SEE ALSO

mesg(5), msgmap(5), msgmax(5), msgmnb(5), msgmni(5), msgssz(5), msgtql(5).

## NAME

msgssz - number of bytes in a System V IPC message segment

## VALUES

### Default

**8**

### Failsafe

**8**

### Allowed Values

Greater than or equal to **1**.

## DESCRIPTION

The **msgssz** tunable specifies the size, in bytes, of a "segment" of memmory space reserved for storing IPC messages. Space for messages is acquired in segment-sized increments as required to contain the message. Some space may be unused and inaccessible in the last segment acquired while the message remains stored. The potential for fragmentation should be considered.

The total available space for messages in the system is defined by the product of **msgseg*msgssz** or, the number of segments multiplied by the segment size. Changing the ratio between these two tunables changes the granularity for message storage and therefore the fragmentation to be expected for any given message usage pattern. Note that the total number of segments in the system is defined by **msgseg**, and **msgssz*msgseg** is the total message storage space.

Set **msgssz*msgseg** equal to or grater than the maximum a single large message queue can hold. **msgssz** is the "granularity" for message storage. For example, if **msgssz** is equal to 256, then a 257-byte message will consume two segments. If **msgssz** is equal to **8** (the default), **msgseg** is limited to 2ˆ29 (536,870,912) because **msgssz*msgseg** is limited to an unsigned long.

For more information about System V message queues, refer to the "Overview" section of the *mesg*(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If increasing **msgseg** (the number of segments) will cause the total message space (**msgssz*msgseg**) to exceed an unsigned 32-bit value.

It has direct effect on reserved message space.

### When Should the Value of This Tunable Be Lowered?

**msgssz*msgseg** must remain greater than **msgmnb** and **msgmax**.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V message queue tunables are interrelated and should *not* be treated as independent variables. The set must be evaluated as a system to ensure the tunables reflect the application requirements. The **mesg** tunables include **msgmap**, **msgmax**, **msgmnb**, **msgmni**, **msgseg**, **msgssz**, and **msgtql**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**msgssz** was developed by AT&T.

## SEE ALSO

mesg(5), msgmap(5), msgmax(5), msgmnb(5), msgmni(5), msgseg(5), msgtql(5).

## NAME

msgtql - maximum number of System V IPC messages in the system at any time

## VALUES

### Default

**40**

### Failsafe

**40**

### Allowed values

Greater than or equal to **1**

### Recommended values

The largest number of messages expected.

## DESCRIPTION

The **msgtql** tunable specifies the maximum total system-wide individual messages across all message queues. Every message has a header to specify message type and location and the total number of headers is limited by **msgtql**. Headers are stored in shared swappable memory. Therefore there is some impact on memory usage, though minimal, resulting from the value of this tunable.

If **msgtql** is set high as a result of expecting large numbers of small messages, the impact of all other System V message tunables should be considered.

Note that if a **msgsnd()** system call attempts to exceed the limit imposed by **msgtql**, it will return [EAGAIN] if **IPC_NOWAIT** is set. If **IPC_NOWAIT** is not set, it will block waiting for a free header.

For more information about System V message queues, refer to the "Overview" section of the *mesg*(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If requiring more total message rather than bigger messages. It has no direct impact on memory space unless actually used.

### When Should the Value of This Tunable Be Lowered?

If applications no longer require as many total messages, and if greater protection from errant applications is needed.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V message queue tunables are interrelated and should *not* be treated as independent variables. The set must be evaluated as a system to ensure the tunables reflect the application requirements. The **mesg** tunables include **msgmap**, **msgmax**, **msgmnb**, **msgmni**, **msgseg**, **msgssz**, and **msgtql**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**msgtql** was developed by AT&T.

## SEE ALSO

mesg(5), msgmap(5), msgmax(5), msgmnb(5), msgmni(5), msgseg(5), msgssz(5).

## NAME

nbuf - sets the system-wide number of file-system buffer and cache buffer headers

## VALUES

### Minimum

**0** or **16**

### Maximum

Memory limited

### Default

**0**

Specify an integer value of zero (see below).

## DESCRIPTION

**nbuf** sets the system-wide number of file-system buffer and cache buffer headers (determines maximum total number of buffers on the system).

### NOTE

This parameter is for backwards compatibility and should be set to zero because dynamic buffer cache is preferred.

If set to a nonzero value, **nbuf** specifies the number of buffer headers to be allocated for the file system buffer-cache. Each buffer is allocated 4096 bytes of memory unless overridden by a conflicting value for **bufpages**.

If **nbuf** is set to a nonzero value that is less than 16 or greater than the maximum supported by the system, or to a value that is inconsistent with the value of **bufpages**, the number will be increased or decreased as appropriate, and a message printed at boot time.

## Related Parameters

**nbuf** interacts with **bufpages**. See *bufpages*(5) for details.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**nbuf** was developed by HP.

## SEE ALSO

bufpages(5).

## NAME

ncallout - size of the kernel callout table

## VALUES

### Failsafe

**`1024`**

The worst case requires one callout entry for every active thread, and some number of callout entries for use by different kernel subsystems and device drivers. A value of 1024 should be failsafe when booting the system to make system repairs.

### Default

**`nkthread+64`** (in HP-UX 11i)

HP-UX 11.0 systems have their default value derived from the **`nproc`** tunable. These systems may panic with the **`Callout Table Overflow`** panic message if **`nproc`** is smaller than **`nkthread`** and the system callout usage exceeds the available callout entries. The default value should be changed as above on these systems.

### Allowed values

Any positive value is *valid* for this tunable, although in practice a very small value will likely cause a panic during boot (or soon after), and a very large one will consume unnecessary kernel memory resources. Changing the tunable from the default value should not be required.

### Recommended values

Use the default value as derived based on **`nkthread`** tunable.

## DESCRIPTION

The **`ncallout`** tunable defines the size of the callout table allocated during system initialization. When a timeout service is requested by kernel subsystems either for their internal usage or due to a user initiated timer request, an entry is used from the callout table. When the callout table runs out of available free entries, the system will panic with the **`Callout Table Overflow`** message.

HP-UX 11.0 systems have no mechanism for dynamic expansion of the callout table, so the system will panic when it runs out of free entries in the callout table.

HP-UX 11.11 systems have a partial dynamic expansion mechanism in place. If there is no available callout entry to service the request, the kernel will try to dynamically allocate new callout entries and service the request. If the kernel cannot allocate memory (in tight memory situations), the system will eventually panic.

### Who Is Expected to Change This Tunable?

Only the 11.0 systems whose default is not set correctly.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If the system panics with the **`Callout Table Overflow`** panic message, the tunable value needs to be raised.

### What Are the Side Effects of Raising the Value of This Tunable?

The tunable value defines how much memory the kernel allocates, so a large value will result in additional physical memory consumption thus reducing the memory available for applications and other uses.

### When Should the Value of This Tunable Be Lowered?

There is no need for it. The **`nkthread`** tunable is expected to define the system workload, the **`ncallout`** tunable is derived from there.

### What Are the Side Effects of Lowering the Value of This Tunable?

The lowering of the **`ncallout`** tunable will increase the chances of hitting the callout table overflow panic in high workload conditions.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**`ncallout`** was developed by HP.

## SEE ALSO

nkthread(5).

## NAME

ncdnode - maximum number of open CD-ROM file system nodes that can be in memory

## VALUES

### Minimum

**14**

### Maximum

Memory limited

### Default

**150**

Specify an integer value or use an integer formula expression.

## DESCRIPTION

**ncdnode** specifies the maximum number of CD-ROM file system nodes that can be in memory (in the vnode table) at any given time. It is functionally similar to **ninode** but applies only to CD-ROM file systems. Behavior is identical on Series 700 and Series 800 systems.

Each node consumes 288 bytes which means, for example, that if **ncdnodes** is set to 10000, nearly 3 MB of memory is reserved exclusively for CD-ROM file system node tables.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**ncdnode** was developed by HP.

## SEE ALSO

ninode(5)

## NAME

nclist - number of cblocks for pty and tty data transfers

## VALUES

### Default

**(100+16*MAXUSERS)**

### Allowed Values

**Minimum 132 and maximum limited by available memory.**

## DESCRIPTION

**nclist** specifies how many **cblocks** are allocated in the system. Data traffic is stored in **cblocks** as it passes through tty and pty devices.

The default value for **nclist**, **(100 + 16 * MAXUSERS)**, is based on a formula of 100 **cblocks** for system use in handling traffic to the console, etc., plus an average of 16 **cblocks** per user session. **cblocks** are also used for serial connections other than login sessions, such as as SLIP connections, UUCP transfers, terminal emulators, and such. If your system is using these other kinds of connections, **nclist** should be increased accordingly.

If the **cblock** pool is exhausted, data being passed through a tty or pty device might be lost because a **cblock** was not available when it was needed. If this occurs, the **WARNING: cblock exhaustion has occurred n times (see termio(7)).** message is placed in the system message buffer.

### Who is expected to change this tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

When the kernel sends the error message **WARNING: cblock exhaustion has occurred n times (see termio(7)).**, the system is running out of **cblocks**. This indicates that **nclist** needs to be increased.

The minimum value for **nclist** is **132**. There is no maximum, but each **cblock** consumes 32 bytes of resident (non-swappable) machine memory, so the value should be selected with this in mind.

### What Are the Side Effects of Raising the Value of This Tunable?

More resident (non-swappable) machine memory is used.

### When Should the Value of This Tunable Be Lowered?

This value should be lowered when a minimal system is being created.

### What Are the Side Effects of Lowering the Value of This Tunable?

The system may run out of **cblocks**.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**nclist** was developed by HP.

## SEE ALSO

termio(7).

## NAME

ndilbuffers - number of Device I/O Library (DIL) buffers

## VALUES

### Failsafe

**0**

### Default

**0**

### Allowed values

**0**

### Recommended values

**0**

## DESCRIPTION

This tunable and the feature that it controlled are obsolete in the HP-UX 1li release. Please do not make any changes to this tunable as they will have no effect on the kernel. This tunable will be removed in the next release.

### Restriction on Changing

This tunable should not be modified.

## NAME

nfile - set maximum number of files that can be open simultaneously on the system at any given time

## VALUES

### Minimum

**14**

### Maximum

Memory limited

### Default

**((16*(nproc+16+maxusers)/10)+32+2*(npty+nstrpty+nstrtel)**

Specify an integer value or use an integer formula expression.

## DESCRIPTION

**nfile** defines the maximum number of files that can be open at any one time, system-wide. It is the number of slots in the file descriptor table.

Be generous with this number because the required memory is minimal, and not having enough slots restricts system processing capacity.

### Related Parameters and System Factors

The value used for **nfile** must be sufficient to serve the number of users and processes allowed by the combination of **nproc**, **maxusers**, **npty**, and **nstrpty**.

Every process uses at least three file descriptors per process (standard input, standard output, and standard error).

Every process has two pipes per process (one per side), each of which requires a pty. Stream pipes also use streams ptys which are limited by **nstrpty**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**nfile** was developed by HP.

## SEE ALSO

maxusers(5), nproc(5), npty(5), nstrpty(5).

## NAME

nflocks - the maximum combined total number of file locks that are available system-wide to all processes at any given time

## VALUES

### Minimum

**2**

### Maximum

Memory limited

### Default

**200**

Specify an integer value or use an integer formula expression.

## DESCRIPTION

**nflocks** gives the maximum number of file/record locks that are available system-wide. When choosing this number, note that one file may have several locks, and databases that use **lockf()** may need an exceptionally large number of locks.

Open and locked files consume memory and other system resources. These resources must be balanced against other system needs to maintain optimum overall system performance. Achieving an optimum balance can be quite complex, especially on large systems, because of wide variation in the kinds of applications being used on each system and the number and types of applications that might be running simultaneously, the number of local and/or remote users on the system, and many other factors.

## RELATED PARAMETERS

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**nflocks** was developed by HP.

## SEE ALSO

lockf(2).

## NAME

ninode - the maximum number of open inodes that can be in memory

## VALUES

### Minimum

**14**

### Maximum

Memory limited

### Default

**nproc+48+maxusers+(2*npty)**

Specify an integer value or use an integer formula expression.

## DESCRIPTION

**ninode** defines the number of slots in the inode table, and thus the maximum number of open inodes that can be in memory at any given time. The inode table is used as a cache memory. For efficiency reasons, the most recent **ninode** (number of) open inodes is kept in main memory. The table is hashed.

Each unique open file has an open inode associated with it. Therefore, the larger the number of unique open files, the larger **ninode** should be.

## RELATED PARAMETERS

The default value of **ninode** is defined in terms of **nproc**, **maxusers**, and **npty**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**ninode** was developed by HP.

## SEE ALSO

maxusers(5), nproc(5), npty(5).

## NAME

nkthread - limits the number of threads allowed to run simultaneously

## VALUES

### Failsafe

**499**

### Default

**(((nproc*7)/4)+16)** or **499**

### Allowed values

**50 - 250,000**

This may be set higher, but more will not be used.

### Recommended values

Unless there are heavily threaded applications on the system, the default formula is adequate if **nproc** is tuned correctly.

## DESCRIPTION

The **nkthread** tunable controls the absolute number of threads allowed on a system at any given time. Increasing it will allow more threads to run simultaneously, and lowering it will restrict the number of threads that can run simultaneously.

It can be determined that **nkthread** is too low when the **kthread: table is full** message is seen in the message buffer The message can be read via **dmesg** or **syslog**. This message indicates that an application was unable to create a thread. Setting **nkthread** too low can cause application failures due to an inability to create new threads or fork new processes.

It can be determined how many threads have been used simultaneously prior to the above message by calling **pstat_dynamic** and examining the **psd_numkthreadsallocd**. This field indicates the "high water" mark of the number of threads that have been simultaneously used.

### Who Is Expected to Change This Tunable?

Anyone expecting to run large numbers of threads. The default value of this is extremely low, and so you may want it to be larger than the default.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This value should be increased, either directly or as a result of tuning **nproc**, on most systems that are running many threads or processes.

### What are the Side Effects of Raising the Value?

There is a slight increase in memory usage (about 100 bytes per kthreads structure) whether or not the structure is used.

### When Should the Value of This Tunable Be Lowered?

The value should be lowered only to limit the number of threads on the system, or when there is memory pressure and the value of **nkthread** is far above the expected usage.

### What are the Side Effects of Lowering the Value?

Increased risk of application failure do to the inability to create new threads or fork new processes.

### What Other Tunable Values Should Be Changed at the Same Time?

**ncallout** should be strictly greater than **nkthread**. The default formula of **ncallout** will ensure this.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**nkthread** was developed by HP.

## SEE ALSO

ncallout(5).

## NAME

nproc - limits the number of processes allowed to run simultaneously

## VALUES

### Failsafe

**256**

### Default

**(20+8*maxusers)** or **256**

### Allowed values

**10 - 30000**

This may be set higher, but more will not be used. Setting **nproc** below **110** will interfere with the systems ability to execute in multi-user mode.

### Recommended values

**1000** processes per processor.

## DESCRIPTION

The **nproc** tunable controls the absolute number of processes allowed on a system at any given time. Increasing it will allow more processes to run simultaneously, and lowering it will restrict the number of processes that can run simultaneously.

It can be determined that **nproc** is too low when the **proc: table is full** message is seen in the message buffer. The message buffer can be read via **dmesg** or **syslog**. This message indicates that an application was unable to fork a new process. Setting **nproc** too low can cause application failures due to an inability to fork new processes.

It can be determined how many processes have been used simultaneously prior to the above message by calling **pstat_dynamic** and examining the **psd_numprocsallocd**. This field indicates the "high water" mark of the number of processes that have been simultaneously used.

### Who Is Expected to Change This Tunable?

Anyone expecting to run large numbers of processes. The default value of this is extremely low, and so you may want it to be larger than the default.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

The tunable should be increased when running large numbers of processes simultaneously.

### What are the Side Effects of Raising the Value?

Memory usage increases largely due to second order tunables such as **nfile**, **ninode**, **nkthread**, and **ncallout**.

### When Should the Value of This Tunable Be Lowered?

The value should be lowered only to limit the number of processes on the system, or when there is memory pressure and the value of **nproc** is far above the expected usage. What are the side effects of lowering the value?

### What are the Side Effects of Lowering the Value?

Increased risk of application failure do to the inability to fork new processes. What other tunables should be changed at the same time as this one?

### What Other Tunable Values Should Be Changed at the Same Time?

**nkthread** must be strictly greater than **nproc**. Default equations ensure this, as well as runtime kernel checks. **nfile** and **ninode** should be increased (and are by default) to allow applications to manipulate files.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**nproc** was developed by HP.

## SEE ALSO

ncallout(5), nfile(5), ninode(5), nkthread(5).

## NAME

npty, nstrpty - maximum number of pseudo-teletypes and streams-based pseudo-teletypes (ptys)

## VALUES

### Default

npty = **60**

nstrpty = **0**

### Allowed Values

Minimum for both: **0**

Maximum for both: **System memory size**

## DESCRIPTION

**npty** is the number of pseudo-teletype (pty) drivers that a system can support. The pty driver provides support for a device-pair called a pseudo terminal. A pseudo terminal is a pair of character devices, a master device, and a slave device. These allow communication between an application process and a server process.

**nstrpty** limits the number of sreams-based ptys that are allowed system-wide. When sending data to pty devices (such as windows), a pty device must exist for every window that is open at any given time.

**nstrpty** should be set to a value that is equal to, or greater than, the number of pty devices on the system that will be using streams-based I/O pipes. Using a parameter value significantly larger than the number of ptys is not recommended. **nstrpty** is used when creating data structures in the kernel to support those streams-based ptys, and an excessively large value wastes kernel memory space.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to these tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable value needs to be raised when the sytem runs out of ptys.

### What Are the Side Effects of Raising the Value?

More system memory is used.

### When Should the Value of This Tunable Be Lowered?

This value should be lowered when a minimal system is being created.

### What Are the Side Effects of Lowering the Value?

The system may run out of pttys.

### What Other Tunable Values Should Be Changed at the Same Time?

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**npty** and **nstrpty** were developed by HP.

## SEE ALSO

pty(7).

## NAME

NSTREVENT - maximum number of outstanding STREAMS bufcalls

## VALUES

### Failsafe

**50**

### Default

**50**

### Allowed values

**0 - 2147483647**

### Recommended values

**50**

## DESCRIPTION

This tunable limits the maximum number of outstanding bufcalls that are allowed to exist in the system at any given time.

This tunable is intended to protect the system against resource overload caused by the combination of modules running in all streams issuing an excessive number of bufcalls. The value selected should be equal to or greater than the combined maximum number of bufcalls that can be reasonably expected during normal operation from all streams on the system. Bufcalls are used by STREAMS modules in low memory situations.

### Who Is Expected to Change This Tunable?

Any customer.

### Restrictions on Changing

Changes to this tunable take effect at next reboot.

n

### When Should the Value of This Tunable Be Raised?

When the system has a lot of low memory situations.

### What Are the Side Effects of Raising the Value of This Tunable?

If too big a number is chosen, the STREAMS subsystem preallocates more memory for internal data structures than may be needed. This reduces the amount of memory available to applications and the system.

### When Should the Value of This Tunable Be Lowered?

If the tunable is increased for a particular STREAMS module/driver, this tunable can be lowered when that STREAMS module/driver is removed. It should be returned to its previous value. However, HP does not recommend a value lower than the default value.

### What Are the Side Effects of Lowering the Value of This Tunable?

During low memory situations, it will bring down system performance.

### What Other Tunable Should Be Changed at the Same Time as This One?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**NSTREVENT** was developed by HP.

## NAME

NSTRPUSH - maximum number of STREAMS modules in a single stream

## VALUES

### Failsafe

**16**

### Default

**16**

### Allowed values

**0 - 2147483647**

### Recommended values

**16**

## DESCRIPTION

This tunable defines the maximum number of STREAMS modules that can be pushed onto a stream. This provides some protection against run-away processes that might automatically select modules to push onto a stream. It is not intended as a defense against malicious use of STREAMS modules by system users.

Most systems do not require more than about three or four modules in a stream. However, there may be some unusual cases where more modules are needed. The default value for this tunable allows as many as 16 modules in a stream, which should be sufficient for even the most demanding installations and applications.

### Who Is Expected to Change This Tunable?

Any customer.

### Restrictions on Changing

Changes to this tunable take effect at next reboot.

### When Should the Value of This Tunable Be Raised?

When the customer needs to push more STREAMS modules in a single stream.

### What Are the Side Effects of Raising the Value of This Tunable?

Might allow run-away applications to unduly consume system resources.

### When Should the Value of This Tunable Be Lowered?

There is no need to lower the tunable value from the default value.

### What Are the Side Effects of Lowering the Value of This Tunable?

Too low of a value might cause network commands to fail.

### What Other Tunable Should Be Changed at the Same Time as This One?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**NSTRPUSH** was developed by HP.

## NAME

NSTRSCHED - number of STREAMS scheduler daemons to run

## VALUES

### Failsafe

`0`

### Default

`0`

### Allowed values

`0 - 2147483647`

### Recommended values

`0`

## DESCRIPTION

This tunable defines the number of multiprocessor (MP) STREAMS scheduler daemons to run on systems containing more than one processor. Note that uniprocessor (UP) systems do not use an MP scheduler daemon, but both MP and UP systems always have one UP STREAMS scheduler (supsched).

If the tunable value is set to zero, the system determines how many daemons to run, based on the number of processors in the system. If the tunable value is set to a positive, non-zero value, that is the number of smpsched daemons that will be created on an MP system.

Note: This tunable is for use by specific HP products only. It may be removed in future HP-UX releases.

### Who Is Expected to Change This Tunable?

Any customer.

### Restrictions on Changing

Changes to this tunable take effect at next reboot.

n

### When Should the Value of This Tunable Be Raised?

This tunable is for use by specific HP products only. It may be removed in future HP-UX releases.

### What Are the Side Effects of Raising the Value of This Tunable?

It could change the system performance unpredictably.

### When Should the Value of This Tunable Be Lowered?

This tunable is for use by specific HP products only. It may be removed in future HP-UX releases.

### What Are the Side Effects of Lowering the Value of This Tunable?

It could change the system performance unpredictably.

### What Other Tunable Should Be Changed at the Same Time as This One?

None.

## WARNINGS

This tunable is for use by specific HP products only.

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**NSTRSCHED** was developed by HP.

## NAME

nstrtel - specifies the number of telnet device files the kernel can support for incoming telnet sessions

## VALUES

### Failsafe

**`60`**

### Default

**`60`**

### Allowed values

**`Any positive integer.`** (Subject to available physical memory.)

It is best to use the default value, and there should not be any need to lower it. However, if the simultaneous telnet connection load is very high, then the value of **`nstrtel`** could be increased.

### Recommended values

**`60`** (Default value.)

## DESCRIPTION

The telnet daemon uses two STREAMS-based pseudo-terminal drivers (**`telm`** and **`tels`**). The kernel parameter, **`nstrtel`**, can be used to tune the number of pseudo-terminals. **`nstrtel`** specifies the number of kernel data structures that are created at system boot time that are required to support the device files used by incoming telnet sessions on a server. If the **`insf`** command or SAM is used to create more telnet device files, the value of **`nstrtel`** must be increased accordingly or the device files cannot be used because there are no kernel data structures available for communicating with the system.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

When there is an increase in the load of telnet connections and there are no device files available to open telnet connections, then this kernel parameter should be increased.

When the **`telnetd: Telnet device drivers missing: No such device`**, error is encountered, then the device files have been exhausted and **`nstrtel`** should be raised. Once the value of **`nstrtel`** is raised, **`insf`** should be run to create new device files. (If SAM is used to raise the value of **`nstrtel`**, **`insf`** is run automatically.)

### What Are the Side Effects of Raising the Value of This Tunable?

More resources would be consumed. Extra kernel data structures, and extra device files may clog the system.

### When Should the Value of This Tunable Be Lowered?

It is not advisable to lower the tunable from the default value, in all probability unnecessary. Consult HP Support before lowering this tunable from the default value.

### What Are the Side Effects of Lowering the Value of This Tunable?

Though not recommended, there would not be any side effects.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**`nstrtel`** was developed by HP.

## SEE ALSO

insf(1M), telnetd(1M), telm(7), tels(7).

## NAME

nswapdev - maximum number of devices that can be enabled for swap

## VALUES

### Default

**10** devices

### Allowed values

Minimum: **1** device
Maximum: **25** devices

## DESCRIPTION

Swap devices are managed in a table for easier indexing in the kernel. **nswapdev** sets the kernel variable responsible for the upper limit on this table, and thus the upper limit to devices which can be used for swap.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If another swap device is added to the system which would increase the number of devices above **nswapdev**, and **swapon()** returns [ENOENT] to the caller (see the *swapon*(2) manpage).

### What Are the Side Effects of Raising the Value?

More devices can be added to the system, and the kernel will need a little more memory for the table. A small performance side effect of the kernel having to scan more devices to check for a duplicate device during **swapon** is also true, but realistically negligible.

### When Should the Value of This Tunable Be Lowered?

Only if you are sure the system will never go over a certain number of swap devices, and you wish to lower this tunable to save a small amount of kernel memory and kernel performance during **swapon** operations.

### What Are the Side Effects of Lowering the Value?

No side effects other than the primary and presumably desired new limitation on the number of swap devices.

### What Other Tunable Values Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**nswapdev** was developed by HP.

## NAME

nswapfs - maximum number of file systems that can be enabled for swap

## VALUES

### Default

**10** file systems

### Allowed values

Minimum: **0** file systems
Maximum: **25** files systems

## DESCRIPTION

File system swap devices are managed in a table for easier indexing in the kernel. **nswapfs** sets the kernel variable responsible for the upper limit on this table, and thus the upper limit to file systems which can be used for swap.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If another file system swap is added to the system which would increase the number above **nswapfs**, then **swapon()** returns [ENOENT] to the caller (see the *swapon*(2) manpage).

### What Are the Side Effects of Raising the Value?

More file systems for swap can be added to the system, and the kernel will need a little more memory for the table. A small performance side effect of the kernel having to scan more file systems to check for a duplicate during swapon is also true, but realistically negligible.

### When Should the Value of This Tunable Be Lowered?

Only if you are sure the system will never go over a certain number of swap file systems, and you wish to lower this tunable to save a small amount of kernel memory and kernel performance during swapon operations.

### What Are the Side Effects of Lowering the Value?

No side effects other than the primary and presumably desired new limitation on the number of swap file systems.

### What Other Tunable Values Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**nswapfs** was developed by HP.

## NAME

nsysmap, nsysmap64 - number of entries in a kernel dynamic memory allocation map

## VALUES

### Default

**2 * nproc**, if **nproc** is > 800;
**800**, otherwise.

### Allowed Values

**Any positive value.**

In practice, very small values will likely cause a panic during boot (or soon after) and very large values are probably unnecessary. Changing the tunable from the default is only necessary under certain conditions.

## DESCRIPTION

This value sizes the kernel dynamic memory resource map, an array of address/length pairs that describe the free virtual space in the kernel's dynamic address space.

This array has historically been statically-sized. That means certain pathological workloads that fragment the kernel address space will result in too many entries for the array. Rather than panicking when this condition occurs, the system throws away the last entry, which results in "leaked" kernel virtual address space. If this overflow happens often enough, the system eventually runs out of virtual space and will panic with the **kalloc: out of virtual space** message.

By making the map size tunable, the system can automatically scale the map size according to the system workload size and avoid this problem. If the automatic scaling does not work, you can hand-tune to fit a particular workload. When the default value is overridden, the kernel may increase the value beyond the your specification depending on the system size.

There are different tunables for 32- and 64-bit kernel because the 64-bit kernel has more virtual address space. The **nsysmap** tunable controls 32-bit kernels and the **nsysmap64** tunable controls 64-bit kernels.

### Who Is Expected to Change This Tunable?

Only those experiencing the resource map overflow will need to modify this tunable.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

The following message will appear on the console when the resource map overflow occurs: **sysmap{32|64}: rmap ovflo, lost [X,Y)**.

If this happens only occasionally, no action is necessary. If this happens with any sort of frequency, the tunable should be increased, or the **kalloc: out of kernel virtual space** panic message may occur.

When increasing **nsysmap{32|64}**, doubling the tunable value is a reasonable rule of thumb. If the problem persists after doubling the tunable several times from the default, there is likely to be a more serious kernel problem, and you should contact HP support to investigate.

### What Are the Side Effects of Raising the Value of This Tunable?

Kernel memory use increases very slightly. Depending on the workload, if the tunable is quite large, the performance of kernel memory allocation may be affected.

### When Should the Value of This Tunable Be Lowered?

Lowering this tunable from the default is risky, in all probability unnecessary, and should only be done after first consulting HP support.

### What Are the Side Effects of Lowering the Value of This Tunable?

A much higher probability of resource map overflows eventually leading to a kernel panic.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**nsysmap** and **nsysmap64** were developed by HP.

## NAME

pfdat_hash_locks - determines the size of the pfdat spinlock pool

## VALUES

### Default

`128`

### Allowed values

`64` to `262144`

## DESCRIPTION

Global kernel structures containing information on a running process or memory usage are frequently accessed or modified by several threads concurrently. To prevent race conditions, these structures are protected by spinlocks (kernel data used for synchronization) which allow only the spinlock structure must wait.

Hashed spinlocks are used when each instance of such a data structure is to be protected, and there are several instances. Using a single spinlock for all instances would cause too much contention, but using one spinlock per structure wastes memory while the majority of the locks are unused at any given time.

By allocating a pool of hashed locks, a hash function picks one lock per group of structures, reducing contention while conserving memory. This tunable sets the size of such a pool for the pfdat data structure spinlocks.

### Who Is Expected to Change This Tunable?

Only HP Field engineers should change the value of this tunable.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable should be raised on systems with both low memory contention and high lock contention due to a large number of pfdat structures. Typically, such a system would be a large memory system with a large amount of processes or threads.

### What Are the Side Effects of Raising the Value?

More memory is used by the kernel.

p

### When Should the Value of This Tunable Be Lowered?

This tunable should only be lowered on systems with high contention for physical memory and few processes and threads. The memory saved by making the table smaller will be freed for general use, and could cause contention if many pfdat structures are present and requiring locks.

### What Are the Side Effects of Lowering the Value?

Hash contention for pfdat locks is more likely, which increases the possibility of threads needing to wait on the lock shared between two or more pfdats.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**`pfdat_hash_locks`** was developed by HP.

## NAME

region_hash_locks - determines the size of the region spinlock pool

## VALUES

### Default

**128**

### Allowed values

**64** to **262144**

## DESCRIPTION

Global kernel structures containing information on a running process or memory usage are frequently accessed or modified by several threads concurrently. To prevent race conditions, these structures are protected by spinlocks (kernel data structures used for synchronization) which allow only the spinlock 'holder' to proceed, while all others attempting to access the structure must wait.

Hashed spinlocks are used when each instance of such a data structure is to be protected, and there are several instances. Using a single spinlock for all instances would cause too much contention, but using one spinlock per structure wastes memory while the majority of the locks are unused at any given time.

By allocating a pool of hashed locks, a hash function picks one lock per group of structures, reducing contention while conserving memory. This tunable sets the size of such a pool for the region data structure spinlocks.

### Who Is Expected to Change This Tunable?

Only HP Field engineers should change the value of this tunable.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable should be raised on systems with both low memory contention and high lock contention due to a large number of region structures. Typically, such a system would be a large memory system with a large number of processes or threads.

### What Are the Side Effects of Raising the Value?

More memory is used by the kernel.

### When Should the Value of This Tunable Be Lowered?

This tunable should only be lowered on systems with high contention for physical memory and few processes and threads. The memory saved by making the table smaller will be freed for general use, and could cause contention if many regions are present and requiring locks.

### What Are the Side Effects of Lowering the Value?

Hash contention for region locks is more likely, which increases the possibility of threads needing to wait on the lock shared between two or more regions.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**region_hash_locks** was developed by HP.

## NAME

remote_nfs_swap - enable swapping across NFS

## VALUES

### Default

**0** (off)

### Allowed values

**0** (off) or **1** (on)

## DESCRIPTION

This tunable controls adding a NFS filesystem for use as swap. If **remote_nfs_swap** is set to **0** (off), only local filesystems and devices can be used for swap. If it is set to **1** (on), both local and networked file systems can be used for swap.

Historically, this tunable was used in NFS clusters that are no longer supported, but this capability has not yet been removed.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect on the next boot.

### When Should the Tunable Be Turned On?

Almost never. As previously mentioned, this tunable was designed for systems which are not in use today. Only systems with extremely robust NFS capabilities should even consider using NFS for swap.

### What Are the Side Effects of Turning the Tunable On?

If the value is set to **1** (on), and a NFS partition is added as swap, then some kernel memory will be set aside for NFS transactions to the swap file system. The kernel will then use the NFS swap partition just the same as a local file system. If the NFS capabilities of the system are not robust, this could lead to extremely long swap times (both swap in and swap out), and possibly a loss of memory because all the reserved memory and more could be used by NFS (just when memory pressure is high) causing the need for swap.

### When Should the Tunable Be Turned Off?

Unless you are positive that the NFS system can handle the load of being used as swap, and does not have a local file system or disk drive to spare, this tunable should **always** be set to **0** (off).

### What Are the Side Effects of Turning the Tunable Off?

Any previously defined NFS swap filesystems will no longer be allowed. Kernel memory will not be reserved for NFS transactions due to swap.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**remote_nfs_swap** was developed by HP.

## NAME

rtsched_numpri - number of priority values to support for POSIX.1b realtime applications

## VALUES

### Failsafe

**32**

### Default

**32**

### Allowed Values

Any value in the range of **32** to **512** is allowed. A value lower than **32** is reset to **32**, whereas a value higher than **512** is reset to **512**.

### Recommended Value

**32**

## DESCRIPTION

The **rtsched_numpri** tunable defines how many priority values to support for POSIX 1.b realtime applications. A larger value provides additional flexibility to applications in managing relative priorities of all threads with respect to each other and with respect to other applications running in the system. However, a larger value adds processing overhead in the operating system in terms of managing larger run queues with the possibility of increased search times.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at next reboot.

### When Should the Value of This Tunable Be Raised?

When the system is primarily running POSIX realtime applications, and requires additional flexibility in managing relative priorities of applications.

### What Are the Side Effects of Raising the Value?

Raising the value of the **rtsched_numpri** tunable may cause some operating system overhead due to larger and possibly more sparse run queues.

### When Should the Value of This Tunable Be Lowered?

The default value of the **rtsched_numpri** tunable is already set to the minimum possible value.

### What Other Tunable Values Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**rtsched_numpri** was developed by HP.

## NAME

scroll_lines - number of scrollable lines used by the Internal Terminal Emulator

## VALUES

### Failsafe

`100`

### Default

`100`

### Allowed values

`60` to `999`

### Recommended values

No larger than is sufficient for a user's purposes.

## DESCRIPTION

This tunable specifies the total number of scroll buffer lines used by the HP-UX graphics console Internal Terminal Emulator (ITE). This value is the sum of both on-screen and off-screen lines. For instance, if the ITE has 68 on-screen lines, and the value of scroll_lines is 128, then there will be 60 off-screen lines that can be scrolled back into view.

During boot, the system may adjust the value of `scroll_lines` upward, depending upon the installed graphics hardware. On graphics cards capable of running at more than one resolution, the ITE will first determine the maximum number of on-screen lines across all possible resolutions. The ITE will then ensure that `scroll_lines` is at least equal to this number of on-screen lines. For example, assume the graphics card supports two possible resolutions which result in one ITE resolution of 160 characters by 62 lines and another ITE resolution of 120 characters by 78 lines. If a value of 62 had been specified for `scroll_lines` in the system file, the ITE would then adjust this value upwards to 78.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

The value of `scroll_line` can be increased if the user would like to retain additional lines in the scroll buffer.

### What Are the Side Effects of Raising the Value of This Tunable?

Increasing the size of the ITE scroll buffer consumes kernel memory which will be dedicated to the ITE and cannot be used for other purposes by the rest of the system. Memory is used at the rate of 2 bytes per off-screen character.

### When Should the Value of This Tunable Be Lowered?

If the additional off-screen lines in the scroll buffer are not required, `scroll_lines` can be decreased to free up a small amount of memory. The savings will be two bytes per off-screen character. For instance, if the display is 160 characters wide, decreasing `scroll_lines` by 10 will free up 3200 bytes.

### What Are the Side Effects of Lowering the Value of This Tunable?

If `scroll_lines` is set below or equal to the number of lines displayed on the ITE screen, no off-screen text will be accessible.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

`scroll_lines` was developed by HP.

## NAME

scsi_maxphys - maximum allowed length of an I/O on all SCSI devices

## VALUES

### Failsafe

**1048576**

### Default

**1024*1024**

### Allowed values

**1048576** (**16777215** and **33554432** for V-Class systems)

### Recommended values

**1048576** (**16777215** for V-Class systems)

## DESCRIPTION

This tunable sets the maximum data size the SCSI subsystem will accept for an I/O. Depending on the device's characteristics and the device driver configuration, the maximum size allowed by the SCSI subsystem for a particular SCSI device might be lower than or equal to this tunable's value. It will never be greater.

## USAGE NOTES

### Who Is Expected to Change This Tunable?

Any customer.

### Restrictions on Changing

Changes to this tunable will take effect when the device is first opened.

### When Should the Value of This Tunable Be Raised?

The value should never be raised except on V-Class systems where it can be raised to 32 MB.

### What Are the Side Effects of Raising the Value of This Tunable?

### When Should the Value of This Tunable Be Lowered?

The value should never be lowered.

### What Are the Side Effects of Lowering the Value of This Tunable?

### What Other Tunables Should Be Changed at the Same Time as This One?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**scsi_maxphys** was developed by HP.

## NAME

scsi_max_qdepth - maximum number of I/Os that target will queue up for execution

## VALUES

### Failsafe

**1**

### Default

**8**

### Allowed values

**1 - 255**

### Recommended values

**1 - 255**

Most SCSI-2 and above devices accept multiple commands and have enough internal memory to support the default queue depth set by HP. You may change the default value to tune devices for higher throughput or load balancing.

## DESCRIPTION

Some SCSI devices support tagged queuing, which means that they can have more than one SCSI command outstanding at any point in time. The number of commands that can be outstanding varies by device, and is not known to HP-UX. To avoid overflowing this queue, HP-UX will not send more than a certain number of outstanding commands to any SCSI device. This tunable sets the default value for that limit. The default value can be overridden for specific devices using **ioctl**.

Queue depth is synonymous to the tagged queuing. When supported by a target, it allows the target to accept multiple SCSI commands for execution. Some targets can allow up to 256 commands to be stored from different initiators. This mechanism can help optimization for better performance. Once the target command queue is full, the target terminates any additional I/O and returns a **QUEUE FULL** status to the initiator. Targets may support less than 256 commands to be queued, hence the factory defaults to **8**.

If the system has a combination of devices that support small and larger queue depths, then a queue depth can be set to a value which would work for most devices. For specific devices, the system administrator can change the queue depth on a per device basis using **SIO_SET_LUN_LIMIT ioctl()**. See *scsictl*(1M) for more on how to use **ioctl()**.

The values for both 32-bit and 64-bit kernel are the same.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect immediately.

### When Should the Value of This Tunable Be Raised?

SCSI devices that have enough memory to support higher queue depth than the default set by HP. Such devices may offer better performance if the queue depth is set to a higher value.

### What Are the Side Effects of Raising the Value of This Tunable?

The queue depth applies to all the SCSI devices that support tag queuing. Setting the queue depth to a value larger than the disk can handle will result in I/Os being held off once a **QUEUE FULL** condition exists on the disk. A mechanism exists that will lower the queue depth of the device in case of **QUEUE FULL** condition avoiding infinite **QUEUE FULL** conditions on that device. Nevertheless, this mechanism will periodically try higher queue depths and **QUEUE FULL** conditions will arise.

### When Should the Value of This Tunable Be Lowered?

When the connected SCSI devices support smaller queue depth or for load balancing.

### What Are the Side Effects of Lovering the Value of This Tunable?

Devices that support higher queue depth may not deliver optimal performance when a lower queue depth value is set.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**scsi_max_qdepth** was developed by HP.

## SEE ALSO

scsictl(1M), ioctl(2), scsi(7).

## NAME

sema - enable or disable System V IPC semaphores at boot time

## VALUES

### Failsafe

**1** (on)

### Default

**1** (on)

### Allowed values

**1** (on) or **0** (off)

## DESCRIPTION

The **sema** tunable is a "switch" designed to enable the entire System V (InterProcess Communications package (IPC) semaphore subsystem. It not only allows user access, but when disabled, no memory space will even be reserved on system boot, and the System V IPC semaphore subsystem will be totally removed. However, the default state is enabled so the system exists after the first cold install. Considering the minimal impact for the default tunable values, a disabled semaphore system will be the exception.

### Overview

System V IPC is comprised of mechanisms for arbitrary processes to send and receive data messages, share virtual address space and use semaphores to synchronize execution.

A System V semaphore is a synchronization method for user processes to obtain atomic access to common data and resources. Each semaphore "set" has an ID, but each set can contain one or more independent **sema()** system calls. All operations are done atomically. Either all semaphores are set or none will be set. You determine whether the **sema()** system call will sleep or return with an error should the requested operation fail.

Every semaphore set has an ID which is derived from a user-specified "key." The two "operations" on a semaphore you can request are increment(>0) and decrement(<0). An increment adds **1** to a **sema**, and the kernel issues a wakeup on any processes sleeping on that **sema** (waiting for just such an increment). A decrement subtracts **1** (if possible), but may sleep waiting for another process to increment the **sema**. A "binary" semaphore is simply the use of only **0** and **1** states for a semaphore. A "counting" semaphore provides many processes to increment a single semaphore, and represents a single resource with many waiting processes. The choice is up to you. The kernel does not decide this aspect.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### What Are the Side Effects of Turning the Tunable Off?

Memory space will *not* be reserved on system boot, and the System V IPC semaphore subsystem will be totally removed.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V semaphore tunables are interrelated and should *not* be treated as independent variables. The tunables must be evaluated as a system to ensure they reflect the application requirements. The **sema** tunables include **semaem**, **semmap**, **semmni**, **semmns**, **semmnu**, **semmsl**, **semume**, **semvmx**, and **sysv_hash_locks**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**sema** was developed by AT&T.

## SEE ALSO

semaem(5), semmap(5), semmni(5), semmns(5), semmnu(5), semmsl(5), semume(5), semvmx(5), sysv_hash_locks(5).

## NAME

semaem - maximum cumulative value changes per System V IPC semop() call

## VALUES

### Failsafe

**1024**

### Default

**16384**

### Allowed values

Minimum: **0**

Maximum: **semvmx** or less than or equal to **32767**.

## DESCRIPTION

The **semaem** tunable specifies the maximum cumulative value change for all System V IPC semaphores changed by each undo within any one single **semop()** operation. This refers to the cumulative sum of the undo value changes for each individual semaphore. Any **semop()** call which attempts to exceed this limit will return [ERANGE].

For more information about System V semaphores, refer to the "Overview" section of the *sema*(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If an application is expected to hold and change large numbers of semaphores and large semaphore value changes as compared with the default value.

### When Should the Value of This Tunable Be Lowered?

If requirements for the numbers of semaphores and large semaphore value changes has greatly diminished, or there is a need to prevent large numbers of semaphores or large value changes of a few semaphores.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V semaphore tunables are interrelated and should *not* be treated as independent variables. The tunables must be evaluated as a system to ensure they reflect the application requirements. The **sema** tunables include **semaem**, **semmap**, **semmni**, **semmns**, **semmnu**, **semmsl**, **semume**, **semvmx**, and **sysv_hash_locks**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**semaem** was developed by AT&T.

## SEE ALSO

sema(5), semmap(5), semmni(5), semmns(5), semmnu(5), semmsl(5), semume(5), semvmx(5), sysv_hash_locks(5).

## NAME

semmap - number of entries in a System V IPC semaphore map

## VALUES

### Failsafe

**(semmni+2)**

### Default

**(semmni+2)**

### Allowed values

Minimum: **4**

Maximum: **semmni+2**, or less than or equal to **32767**.

## DESCRIPTION

The **semmap** tunable specifies the size of a System V IPC message space resource map which tracks the free space in shared semaphore memory. Each resource map entry is an offset-space pair which points to the offset and size (bytes) of each contiguous series of unused semaphore space segments.

Free-space fragmentation generally increases as semaphore arrays are added. Since the resource map requires an entry pair for each fragment of free space, excess fragmentation can cause the free-space map to completely fill. If the map is full when message space is acquired or released, the system issues the **DANGER: mfree map overflow** warning message.

For more information about System V semaphores, refer to the "Overview" section of the *mesg*(5) manpage.

### Who Is Expected to Change This Tunable?

This is a kernel-calculated value.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

When the free-space map is full and the system issues the warning message **DANGER: mfree map overflow.** A larger value for **semmap** can correct this issue.

### When Should the Value of This Tunable Be Lowered?

If requirements for semaphores has significantly decreased.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V semaphore tunables are interrelated and should *not* be treated as independent variables. The tunables must be evaluated as a system to ensure they reflect the application requirements. The **sema** tunables include **semaem**, **semmap**, **semmni**, **semmns**, **semmnu**, **semmsl**, **semume**, **semvmx**, and **sysv_hash_locks**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**semmap** was developed by AT&T.

## SEE ALSO

sema(5), semaem(5), semmni(5), semmns(5), semmnu(5), semmsl(5), semume(5), semvmx(5), sysv_hash_locks(5).

## NAME

semmni - number of System V IPC system-wide semaphore identifiers

## VALUES

### Failsafe

**64**

### Default

**64**

### Allowed values

Minimum: **2**

Maximum: **semvmx**.

## DESCRIPTION

The **semmni** tunable specifies the maximum number of System V IPC system-wide semaphore sets (and identifiers) which can exist at any given time. The **semmni** value is used at boot time to size the **sema[]** array of **semid_ds{}** structures. A single identifier (ID) is returned for each **semget()** system call to create a new set of one or more (up to **semmsl**) semaphores. Each semaphore is a single integer value limited by **semvmx**.

If **semget()** is called when there are no free entries in the **sema[]** array, [ENOSPC] is returned.

For more information about System V semaphores, refer to the "Overview" section of the *sema*(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If there is a requirement of more total unique semaphore IDs in the system, or one or more applications require more semaphore IDs. There is no reason to specify more system-wide identifiers than the total of the **semmns** semaphores.

### When Should the Value of This Tunable Be Lowered?

If requirements for semaphores has significantly decreased, or there is a need to hard-limit applications from acquiring larger numbers of semaphore IDs.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V semaphore tunables are interrelated and should *not* be treated as independent variables. The tunables must be evaluated as a system to ensure they reflect the application requirements. The **sema** tunables include **semaem**, **semmap**, **semmni**, **semmns**, **semmnu**, **semmsl**, **semume**, **semvmx**, and **sysv_hash_locks**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**semmni** was developed by AT&T.

## SEE ALSO

sema(5), semaem(5), semmap(5), semmns(5), semmnu(5), semmsl(5), semume(5), semvmx(5), sysv_hash_locks(5).

## NAME

semmns - number of System V IPC system-wide semaphores

## VALUES

### Failsafe

**128**

### Default

**128**

### Allowed values

Minimum: **2**

Maximum: **32767**

## DESCRIPTION

The **semmns** tunable specifies the maximum total individual System V IPC system-wide semaphores which can be assigned by applications. Semaphores are assigned in "sets" (with **semmni** maximum sets) with one ID per set and up to **semmsl** semaphores per set. Thus semaphores can be distributed in any manner accross the range of IDs with one or more per ID.

There is no reason to specify **semmns** less than **semmni** (the maximum number of identifiers) as some IDs would never be used.

For more information about System V semaphores, refer to the "Overview" section of the **sema**(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If requiring more total semaphores in the system, or one or more applications require more semaphores.

### When Should the Value of This Tunable Be Lowered?

If requirements for semaphores has significantly decreased, or there is a need to hard-limit applications from acquiring larger numbers of semaphores.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V semaphore tunables are interrelated and should *not* be treated as independent variables. The tunables must be evaluated as a system to ensure they reflect the application requirements. The **sema** tunables include **semaem**, **semmap**, **semmni**, **semmns**, **semmnu**, **semmsl**, **semume**, **semvmx**, and **sysv_hash_locks**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**semmns** was developed by AT&T.

## SEE ALSO

sema(5), semaem(5), semmap(5), semmni(5), semmnu(5), semmsl(5), semume(5), semvmx(5), sysv_hash_locks(5).

## NAME

semmnu - number of System V IPC system-wide semaphore undo structures

## VALUES

### Failsafe

**30**

### Default

**30**

### Allowed values

Minimum: **1**

Maximum: **(nproc-4)**

## DESCRIPTION

The **semmnu** tunable specifies the maximum number of System V IPC system-wide processes that can have "undo" operations pending at any given time. If a process which has invoked a semaphore operation with the **SEM_UNDO** flag.....

For more information about System V semaphores, refer to the "Overview" section of the **sema**(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If the semaphore undo activity is causing contention between processes with increased numbers of system-wide semaphores.

### When Should the Value of This Tunable Be Lowered?

If requirements for semaphores has significantly decreased, or for limiting protection on run-away applications which change semaphores.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V semaphore tunables are interrelated and should *not* be treated as independent variables. The tunables must be evaluated as a system to ensure they reflect the application requirements. The **sema** tunables include **semaem**, **semmap**, **semmni**, **semmns**, **semmnu**, **semmsl**, **semume**, **semvmx**, and **sysv_hash_locks**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**semmnu** was developed by AT&T.

## SEE ALSO

sema(5), semaem(5), semmap(5), semmns(5), semmni(5), semmsl(5), semume(5), semvmx(5), sysv_hash_locks(5).

## NAME

semmsl - maximum number of System V IPC semaphores per identifier

## VALUES

### Failsafe

**2048**

### Default

**2048**

### Allowed values

Minimum: **2048**

Maximum: **10240**

## DESCRIPTION

The **semmsl** tunable specifies the maximum number of individual System V IPC semaphores per semaphore identifier (ID). This is without regard to the value each semaphore may have.

For more information about System V semaphores, refer to the "Overview" section of the **sema**(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

None. This tunable is dynamic.

### When Should the Value of This Tunable Be Raised?

If applications require a larger number of semaphores per ID, an application no longer requires as many semaphores per ID, or there is a need to protect against ill-behaved applications.

### When Should the Value of This Tunable Be Lowered?

If requirements for semaphores has significantly decreased.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V semaphore tunables are interrelated and should *not* be treated as independent variables. The tunables must be evaluated as a system to ensure they reflect the application requirements. The **sema** tunables include **semaem**, **semmap**, **semmni**, **semmns**, **semmnu**, **semmsl**, **semume**, **semvmx**, and **sysv_hash_locks**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**semmsl** was developed by AT&T.

## SEE ALSO

sema(5), semaem(5), semmap(5), semmns(5), semmni(5), semmnu(5), semume(5), semvmx(5), sysv_hash_locks(5).

## NAME

semume - maximum number of System V IPC undo entries per process

## VALUES

### Failsafe

**10**

### Default

**10**

### Allowed values

Minimum: **1**

Maximum: **semnms**

## DESCRIPTION

The **semume** tunable specifies the maximum number of System V IPC semaphore undo operations which can be specified by a single process call to **semop()** to get or release semaphores. [ENOSPC] is set if the number of individual processes that requested a **SEM_UNDO** will be exceeded. [EINVAL] is set if the the process request for **SEM_UNDO** exceeds the limit.

For more information about System V semaphores, refer to the "Overview" section of the **sema**(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If applications require more semaphore undo operations per process.

### When Should the Value of This Tunable Be Lowered?

If per-process requirements for semaphore operation has significantly decreased, or if there for limit protection on run-away applications which change semaphores.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V semaphore tunables are interrelated and should *not* be treated as independent variables. The tunables must be evaluated as a system to ensure they reflect the application requirements. The **sema** tunables include **semaem**, **semmap**, **semmni**, **semmns**, **semmnu**, **semmsl**, **semume**, **semvmx**, and **sysv_hash_locks**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**semume** was developed by AT&T.

## SEE ALSO

sema(5), semaem(5), semmap(5), semmni(5), semmnu(5), semmsl(5), semmns(5), semvmx(5), sysv_hash_locks(5).

## NAME

semvmx - maximum value of any single System V IPC semaphore

## VALUES

### Failsafe

**1024**

### Default

**32767**

### Allowed values

Minimum: **1**

Maximum: **65535**

## DESCRIPTION

The **semvmx** tunable specifies the maximum value any given System V IPC semaphore can have. Any **semop()** call which attempts to set a semaphore value beyond the **semvmx** limit will return [ERANGE].

For more information about System V semaphores, refer to the "Overview" section of the **sema**(5) manpage.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If an application requires larger semaphore values, for example, when many child processes use one counting semaphore.

### When Should the Value of This Tunable Be Lowered?

If requirements for the single semaphore value has significantly decreased, or if there for limit protection on run-away applications which change semaphores.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V semaphore tunables are interrelated and should *not* be treated as independent variables. The tunables must be evaluated as a system to ensure they reflect the application requirements. The **sema** tunables include **semaem**, **semmap**, **semmni**, **semmns**, **semmnu**, **semmsl**, **semume**, **semvmx**, and **sysv_hash_locks**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**semvmx** was developed by AT&T.

## SEE ALSO

sema(5), semaem(5), semmap(5), semmni(5), semmns(5), semmnu(5), semmsl(5), semume(5), sysv_hash_locks(5).

## NAME

sendfile_max - maximum number of Buffer Cache Pages used by sendfile

## VALUES

### Failsafe

**`0`** unlimited

### Default

**`0`** unlimited

### Allowed values

**`0`** unlimited

Or **`1`** to **`262144`**

### Recommended values

**`0`** unlimited

## DESCRIPTION

This variable is used to limit how many Buffer Cache Pages the **`sendfile()`** system call can use. This might be useful in a machine that has a large **`sendfile()`** load but is limited in memory.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable might be useful in a machine that has a large **`sendfile()`** load but is limited in memory.

### What Are the Side Effects of Raising the Value?

Possible side effects would be slower **`sendfile()`** performance.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**`sendfile_max`** was developed by HP.

## NAME

shmem - enable or disable System V shared memory

## VALUES

### Failsafe

**1** (on)

### Default

**1** (on)

### Allowed values

**0** (off) or **1** (on)

## DESCRIPTION

Shared memory is an efficient InterProcess Communications (IPC) mechanism. One process creates a shared memory segment and attaches it to its address space. Any processes looking to communicate with this process through the shared memory segment, then attach the shared memory segment to their corresponding address spaces as well. Once attached, a process can read from or write to the segment depending on the permissions specified while attaching it.

This tunable controls the availability of System V shared memory in the HP-UX kernel. Setting it to *off* changes the tunables **shmmax**, **shmmni**, and **shmseg** to **0** (off), effectively allowing no shared memory segments to be created.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect on the next reboot.

### When Should the Tunable Be Turned On?

If System V shared memory is disabled and a system user wishes to access this functionality, setting the value to **1** (on) will enable it.

### What Are the Side Effects of Turning the Tunable On?

The tunables **shmmax**, **shmmni**, and **shmseg** use the values set in the tunable configuration files instead of the value of **0** (off).

### When Should the Tunable Be Turned Off?

If System V shared memory segments are not needed, this functionality can be disabled. This leaves the option of shared memory via the **mmap()** system call instead.

### What Are the Side Effects of Turning the Tunable Off?

Code that is trying to use System V shared memory functionality will receive the [EINVAL] error message because any request for a segment will be greater than the system-imposed maximum set to zero.

### What Other Tunables Should Be Changed at the Same Time?

If the System V shared memory is being enabled, a check of the values for **shmmax**, **shmmni**, and **shmseg** for reasonable bounds should be performed. These values represent the maximum size of a segment, number of identifiers, and maximum number of shared segments per process respectively.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**shmem** was developed by HP.

## SEE ALSO

shmmax(5), shmmni(5), shmseg(5).

## NAME

shmmax - maximum size (in bytes) for a System V shared memory segment

## VALUES

### Default

**0X4000000**

### Allowed values

Minimum: **2048**

Maximum: **0X4000000** for a 32 bit kernel,
and **0x40000000000** for a 64 bit kernel

## DESCRIPTION

Shared memory is an efficient InterProcess Communications (IPC) mechanism. One process creates a shared memory segment and attaches it to its address space. Any processes looking to communicate with this process through the shared memory segment, then attach the shared memory segment to their corresponding address spaces as well. Once attached, a process can read from or write to the segment depending on the permissions specified while attaching it.

This tunable sets the maximum size for such a segment within the system, and is dynamic as of HP-UX 11i.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect immediately.

### When Should the Value of This Tunable Be Raised?

**shmmax** should be raised if it is below the maximum and user programs are attempting to **shmget** segments larger than the current value, and receiving an [EINVAL] error message.

### What Are the Side Effects of Raising the Value?

The only effect is that user programs can use **shmget** to get larger segments.

### When Should the Value of This Tunable Be Lowered?

If you wish to enforce behavior on the user's code, limiting the maximum size of System V segments.

### What Are the Side Effects of Lowering the Value?

None.

### What Other Tunable Values Should Be Changed at the Same Time?

**shmseg** and **shmmni** should be considered because enforcing smaller segments may cause the user's code to try to create more segments to accomplish the task.

If this tunable is being set to the minimum to try to discourage the use of System V shared memory, **shmem** should be used instead.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**shmmax** was developed by HP.

## SEE ALSO

shmem(5), shmmni(5), shmseg(5).

## NAME

shmmni - number of System V shared memory segment identifiers in the system

## VALUES

### Default

**200** identifiers

### Allowed values

Minimum: **3**

Maximum: **1024** for a 32 bit kernel,
and **8192** for a 64 bit kernel

## DESCRIPTION

Shared memory is an efficient InterProcess Communications (IPC) mechanism. One process creates a shared memory segment and attaches it to its address space. Any processes looking to communicate with this process through the shared memory segment, then attach the shared memory segment to their corresponding address spaces as well. Once attached, a process can read from or write to the segment depending on the permissions specified while attaching it.

This tunable effectively sets the number of unique segments creatable system wide, since each segment is assigned an identifier by the kernel. The identifier is simply a reference generated by the kernel such that any user process can request a particular segment for sharing with a simple integer, and let the kernel determine which segment this corresponds to.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

**shmmni** should be raised if users of System V shared memory are receiving the [ENOSPC] error message on **shmget()** calls.

### What Are the Side Effects of Raising the Value?

Kernel memory usage will be slightly increased, as the data structures used to track the segments are preallocated based on this tunable.

### When Should the Value of This Tunable Be Lowered?

If kernel memory is at a premium, or it is known that few segments will be needed, a slight savings can be gained from decreasing this tunable, and thus decreasing the data structure memory usage associated with it.

If System V shared memory is not to be used at all, simply set the **shmem** tunable to *off* instead.

### What Are the Side Effects of Lowering the Value?

Kernel memory usage will be slightly reduced.

### What Other Tunable Values Should Be Changed at the Same Time?

**shmmax** and **shmseg** should be considered. **shmseg** should be changed in the same manner as **shmmni**, since lowering the total number of segments but raising the number available per process only makes sense if you want a few processes taking all the segments.

**shmmax** is more complex and any changes to it really depend on the effect desired. Refer to the *shmmax*(5) manpage for more information before changing this tunable.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**shmmni** was developed by HP.

## SEE ALSO

shmem(5), shmmax(5), shmseg(5).

## NAME

shmseg - maximum number of System V shared memory segments per process

## VALUES

### Default

**120** segments

### Allowed values

Minimum: **1**

Maximum: Any value less than or equal to **shmmni**.

## DESCRIPTION

Shared memory is an efficient InterProcess Communications (IPC) mechanism. One process creates a shared memory segment and attaches it to its address space. Any processes looking to communicate with this process through the shared memory segment, then attach the shared memory segment to their corresponding address spaces. Once attached, a process can read from or write to the segment depending on the permissions specified while attaching it.

This tunable sets an upper limit to the number of segments which can be attached per process.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If user processes reach their limit, more segments are desired, and the current value is less than the total number of segments in the system (**shmmni**).

### What Are the Side Effects of Raising the Value?

A single process will be able to acquire more segments, possibly starving another process which previously was able to acquire all the segments it needed. In this case, **shmmni** should be raised if it is below it's maximum.

### When Should the Value of This Tunable Be Lowered?

This tunable should only be lowered to enforce segment policy on user processes, or if one runaway process is hogging the global-segment pool. Otherwise, keeping the maximum higher than the common usage is harmless.

### What Are the Side Effects of Lowering the Value?

None.

### What Other Tunable Values Should Be Changed at the Same Time?

**shmmni** should be considered, as previously described.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**shmseg** was developed by HP.

## SEE ALSO

shmmni(5).

## NAME

STRCTLSZ - maximum size of streams message control (bytes)

## VALUES

### Failsafe

**1024**

### Default

**1024**

### Allowed values

**0 - 2147483647**

### Recommended values

**1024**

## DESCRIPTION

STRCTLSZ limits the maximum number of bytes of control data that can be inserted by **putmsg()** in the control portion of any streams message on the system. If the tunable is set to zero, there is no limit on how many bytes can be placed in the control segment of the message.

**putmsg()** returns ERANGE if the buffer being sent is larger than the current value of STRCTLSZ.

### Who Is Expected to Change This Tunable?

Any customer.

### Restrictions on Changing

Changes to this tunable take effect at next reboot.

### When Should the Value of This Tunable Be Raised?

The tunable should be increased by any customer if the customer's STREAMS module/driver(s) require more bytes in the control portion of any streams message than the current value.

### What Are the Side Effects of Raising the Value of This Tunable?

The kernel will use more memory. During low memory situations, it may bring down system performance due to frequent swapping.

### When Should the Value of This Tunable Be Lowered?

The tunable could be lowered by any customer if the customer's STREAMS module/driver(s) do not require a longer message size in the control portion than the current value.

### What Are the Side Effects of Lowering the Value of This Tunable?

Possible improper functioning in any of the STREAMS module/driver(s). Possible performance degradation, particularly in networking.

### What Other Tunable Should Be Changed at the Same Time as This One?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**STRCTLSZ** was developed by HP.

## NAME

streampipes - force all pipes to be STREAMS-based

## VALUES

### Failsafe

**0**

### Default

**0**

### Allowed values

**0 - 2147483647**

### Recommended values

**0**

## DESCRIPTION

This tunable determines the type of pipe that is created by the **pipe()** system call. If set to the default value of zero, all pipes created by **pipe()** are normal HP-UX file-system pipes. If the value is non-zero, **pipe()** creates STREAMS-based pipes, and STREAMS modules can be pushed onto the resulting stream.

If this tunable is set to a non-zero value, the **pipemod** and **pipedev** module and driver must be configured in file **/stand/system**.

### Who Is Expected to Change This Tunable?

Any customer.

### Restrictions on Changing

Changes to this tunable take effect at next reboot.

### When Should the Tunable Be Turned On?

If the customer uses applications that require STREAMS-based pipes, this tunable should be turned on.

### What Are the Side Effects of Turning the Tunable On?

STREAMS-based pipes performance may differ from normal file system pipes.

### When Should the Tunable Be Turned Off?

If the customer does not need the STREAMS-based pipes, this tunable should be turned off.

### What Are the Side Effects of Turning the Tunable Off?

Applications that try to push STREAMS modules onto the pipe will fail.

### What Other Tunable Should Be Changed at the Same Time as This One?

If this tunable is set to a non-zero value, the **pipemod** and **pipedev** module and driver must be configured in the file **/stand/system**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**streampipes** was developed by HP.

## NAME

STRMSGSZ - maximum size of streams message data (bytes)

## VALUES

### Failsafe

`0`

### Default

`0`

### Allowed values

`0 - 2147483647`

### Recommended values

`0`

## DESCRIPTION

This tunable limits the number of bytes of message data that can be inserted by **putmsg()** or **write()** in the data portion of any streams message on the system. If the tunable is set to zero, there is no limit on how many bytes can be placed in the data segment of the message.

**putmsg()** returns ERANGE if the buffer being sent is larger than the current value of STRMSGSZ; **write()** segments the data into multiple messages.

### Who Is Expected to Change This Tunable?

Any customer.

### Restrictions on Changing

Changes to this tunable take effect at next reboot.

### When Should the Value of This Tunable Be Raised?

The tunable should be increased by any customer if the customer's STREAMS module/driver(s) require a longer message size in the data portion than the current value.

### What Are the Side Effects of Raising the Value of This Tunable?

The kernel will use more memory. During low memory situations, it may bring down system performance due to frequent swapping.

### When Should the Value of This Tunable Be Lowered?

The tunable could be lowered by any customer if the customer's STREAMS module/driver(s) do not require a longer message size in the data portion than the current value.

### What Are the Side Effects of Lowering the Value of This Tunable?

Possible improper functioning in any of the STREAMS module/driver(s). Possible performance degradation, particularly in networking.

### What Other Tunable Should Be Changed at the Same Time as This One?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**STRMSGSZ** was developed by HP.

## NAME

st_ats_enabled - determines whether to reserve a tape device on open

## VALUES

### Failsafe

**0** (off)

### Default

**1** (on)

### Allowed values

**0** (off) or **any positive integer** (on)

### Recommended values

**0** (off) or **1** (on)

## DESCRIPTION

This tunable notifies the stape driver whether it needs to reserve a tape device on open and subsequently release it on close. With this tunable *on*, the stape driver won't necessarily attempt to reserve any tape device on open. The driver has a few tape devices flagged as suitable devices for this functionality. This list includes the DLT 8000, DLT 7000, STK 9840, HP Ultrium, and STK SD-3 drives. *DDS drives are not flagged for this functionality.*

The **st_ats_enabled** tunable allows the safe sharing of tape devices in multi-node configurations like MC/ServiceGuard's Advanced Tape Sharing. The automatic reserve and release protects the tape device from being accessed from multiple nodes which would corrupt a backup. If the reserve portion of the stape driver's open routine fails, a status of [EBUSY] is returned.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect on the next reboot.

### When Should the Tunable Be Turned On?

This tunable should be turned *on* if the system is going to be used in an MC/SG Advanced Tape Sharing configuration or the user wants the stape driver to use automatic reserve/release on open and close for multimode manageability.

### What Are the Side Effects of Turning the Tunable On?

A reserve is sent to the tape device by the stape driver on open and a release is sent on close. No other initiator/HBA will be able to access the tape device while it is opened on another initiator/HBA.

### When Should the Tunable Be Turned Off?

It is recommended for any SAN tape sharing solution configuration other than ATS that this tunable should be turned *off*. Most multi-node backup applications like Omniback manage device reservation themselves and any interference by the stape driver may produce problems. This tunable should also be turned *off* for any multi-platform SAN configuration to keep tape access across platforms more uniform.

### What Are the Side Effects of Turning the Tunable Off?

Unauthorized access from another initiator might interfere with any current tape operation.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**st_ats_enabled** was developed by HP.

## SEE ALSO

scsi_tape(7).

**(Tunable Kernel Parameters)**

## NAME

st_fail_overruns - determines whether variable block mode read requests smaller than the physical record size will fail

## VALUES

### Failsafe

**0** (off)

### Default

**0** (off)

### Allowed values

**0** (off) or **non-zero** (on)

### Recommended values

**0** (off) or **non-zero** (on)

## DESCRIPTION

This tunable determines whether variable block mode read requests through the stape driver, that are smaller than the physical record size, will fail with the error [EFBIG].

For example, if a variable block mode read request of 32K was sent to a tape device with media of record size 64K, the entire record's data would not be returned to the host. If the tunable was set to **non-zero** (on), the read would be returned as failed with error [EFBIG]. If the tunable were set to **0** (off), the read request would be returned as successful with a return value equal to the number of bytes read.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect on the next boot.

### When Should the Tunable Be Turned On?

If you want the stape driver to fail overrun variable block mode reads to tape as some applications require.

### What Are the Side Effects of Turning the Tunable On?

Variable block mode read requests through the stape driver that are smaller than the record fail with [EFBIG].

### When Should the Tunable Be Turned Off?

If you want the stape driver to handle an overrun read as a successful read, forcing you to retry the read with the proper request size. This is the behavior most backup software expects.

### What Are the Side Effects of Turning the Tunable Off?

The return value of the read must be checked to verify the number of bytes read equals the amount of data requested on all variable block mode reads.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**st_fail_overruns** was developed by HP.

## SEE ALSO

scsi_tape(7).

## NAME

st_large_recs - determines maximum logical record size allowed through the stape driver

## VALUES

### Failsafe

**0** (off)

### Default

**0** (off)

### Allowed values

**0** (off) or **non-zero** (on)

### Recommended values

**0** (off) or **non-zero** (on)

## DESCRIPTION

This tunable governs the maximum size of records allowed for tape device I/O through the stape driver. Larger records can be sent, but the I/O subsystem will split the request into multiple requests with a maximum record size determined by this tunable.

For instance, if a read of 512K were sent to a tape device on a system with **MAXPHYS** at 256K, two scenarios would exist. With **st_large_recs** set to any positive integer, the read would be sent requesting a record size of 512K, and two read requests of record size 256K would be sent with **st_large_recs** set to **0** (off).

Although the stape driver allows a maximum record size of 16MB-1 with **st_large_recs** set to a positive integer, the actual maximum record size supported on a particular system may be restricted by the maximum I/O size supported by the tape device, the SCSI subsystem, or interface, for example FC, SCSI, etc.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect on the next boot.

### When Should the Tunable Be Turned On?

When on a system with SCSI subsystem and tape device support for record sizes greater than **MAXPHYS** (512K), this tunable should be set to **non-zero** (on).

S

### What Are the Side Effects of Turning the Tunable On?

If the tape device, SCSI subsystem, or interface do not support larger I/O record sizes, requests may be split into records that are too large causing a returned error of [EINVAL].

### When Should the Tunable Be Turned Off?

When on a system with SCSI subsystem, tape device, or interface support for maximum record sizes of **MAXPHYS** (512K), this tunable should be set to **0** (off).

### What Are the Side Effects of Turning the Tunable Off?

If the tape device, SCSI subsystem, and interface allow larger records than **MAXPHYS**, you may see slower I/O throughput because the full potential I/O throughput of the configuration is not being used.

### What Other Tunables Should Be Changed at the Same Time?

If on a system that supports large I/O requests like a V-class, you would also want to be sure the tunable **scsi_maxphys** is larger than 16MB-1.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**st_large_recs** was developed by HP.

## SEE ALSO

scsi_maxphys(5), scsi_tape(7).

## NAME

swapmem_on - allow physical memory size to exceed the available swap space

## VALUES

### Failsafe

**1** (on)

### Default

### Failsafe

**1** (on)

### Allowed values

**0** (off) or **1** (on)

## DESCRIPTION

In previous versions of HP-UX, system configuration required sufficient physical swap space for the maximum possible number of processes on the system. This is because HP-UX reserves swap space for a process when it is created, to ensure that a running process never needs to be killed due to insufficient swap.

This was difficult, however, for systems needing gigabytes of swap space with gigabytes of physical memory, and those with workloads where the entire load would always be in core. This tunable was created to allow system swap space to be less than core memory. To accomplish this, a portion of physical memory is set aside as 'pseudo-swap' space. While actual swap space is still available, processes still reserve all the swap they will need at fork or execute time from the physical device or file system swap. Once this swap is completely used, new processes do not reserve swap, and each page which would have been swapped to the physical device or file system is instead locked in memory and counted as part of the pseudo-swap space.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect on the next reboot.

### When Should the Tunable Be Turned On?

If this tunable is set to **0** (off), and a **fork()** or **exec()** process is failing with an [ENOMEM] error message and has sufficient system memory but insufficient unreserved swap space, then setting this tunable to **1** (on) will allow these processes to use pseudo-swap space and execute.

### What Are the Side Effects of Turning the Tunable On?

Some physical memory is set aside for pseudo-swap, but since the kernel can steal pages from this allocation if needed (for locked memory or kernel memory), and the rest of the pages will only be used when physical swap is completely filled anyway, this is quite harmless.

### When Should the Tunable Be Turned Off?

This tunable can be turned *off* if there are sufficient physical swap devices or file systems such that the system workload never fails to reserve swap. However, turning this feature *off* really does not gain the system anything.

### What Are the Side Effects of Turning the Tunable Off?

Processes will be limited to physical swap devices or file systems for reserving their swap on a **fork()** or an **exec()**.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**swapmem_on** was developed by HP.

## NAME

swchunk - swap chunk size in 1 KB blocks

## VALUES

### Default

**2048** blocks

### Allowed values

Minimum: **1** blocks

Maximum: **65536** blocks

## DESCRIPTION

Swap space in the kernel is managed using 'chunks' of physical device space. These chunks contain one or more (usually more) pages of memory, but provide another layer of indexing (similar to *inodes* in file systems) to keep the global swap table relatively small, as opposed to a large table indexed by swap page.

**swchunk** controls the size in physical disk blocks (which are defined as 1 KB) for each chunk. Combining this information with the value of **maxswapchunks** (the maximum number of swap chunks in the swap table) provides the total bytes of swap space manageable by the system.

The way to think of **swchunk** is not as the size of the I/O transactions in the swap system (in disk blocks), but as the number of blocks that will be placed on one swap device (or file system) before moving to the next device (assuming all priorities are equal). This spreads the swap space over any devices and is called *swap* interleaving. Swap interleaving spreads out the swap over many devices and reduces the possibility of one single device becoming a bottleneck for the entire system when swap usage is heavy.

### Who is Expected to Change This Tunable?

This tunable should only be modified by those with a complete knowledge of both kernel behavior and underlying device hardware.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If **maxswapchunks** has already been raised to its limit and the system owner wishes to add more swap to the system, but the additional swap chunks needed are unavailable, raising this tunable will work around the problem. By increasing the size of each chunk, fewer total chunks are needed.

### What Are the Side Effects of Raising the Value?

The second level of the swap table (used to track pages within a chunk) will increase, resulting in more memory used by the kernel. If **maxswapchunks** is reduced to have fewer larger chunks to represent the same swap space, this will be offset by the smaller first level of the swap table. If **swchunk** is being increased to allow for mapping of a larger swap space, increased memory usage by the kernel to track the swap space is unavoidable.

This means that more swap is allocated to each device (or file system) in the round-robin interleaving scheme and all priorities are equal. If **maxswapchunks** is not at its maximum, increasing **swchunck** instead of **maxswapchunks** could hinder system performance by creating unneeded I/O bottlenecks; where, two pages that were in different chunks using the smaller value and were previously on different swap devices and thus accessible independently of one another (with no read head or controller issues) are now on the same device and slowing down access time for the second page.

### When Should the Value of This Tunable Be Lowered?

If the amount of swap space mappable by the system is much larger than the total amount of swap space which is attached (or going to be attached) to the system, which is calculable by multiplying **maxswapchunks*** **swchunk*** 1KB, then kernel memory usage can be reduced by lowering either **swchunk** or **maxswapchunks** to fit the actual swap space. Note that modification of **maxswapchunks** is preferable to **swchunk** unless **swchunk** I/O bottlenecks are being created due to the loss of round-robin interleaving.

### What Are the Side Effects of Lowering the Value?

It may have to be raised back if more swap is added to the system and there is not enough room in the swap table to allow for the increased space. If this is not the case, then there is a finer grain of interleaving on the system (assuming there is more than one swap device) that can provide a performance gain under heavy swap usage.

### What Other Tunable Values Should Be Changed at the Same Time?

**maxswapchunks** should be considered when changing this tunable.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**swchunk** was developed by HP.

## SEE ALSO

maxswapchunks(5).

## NAME

sysv_hash_locks - System V IPC hashed spinlock pool size

## VALUES

### Failsafe

**128**

### Default

**128**

Required to be a power of two.

## DESCRIPTION

The **sysv_hash_locks** tunable specifies the size of the hashed spinlock (kernel data structures used for synchronization) pool (number of available spinlocks). The System V IPC semaphore functions acquire hashed spinlocks based on the semaphore ID. The spinlocks are hashed to avoid the waste of **semmni** spinlocks or the contention of just one spinlock.

For more information about System V semaphores, refer to the "Overview" section of the **sema**(5) manpage.

### Who Is Expected to Change This Tunable?

This is a kernel-calculated value.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

If the increase to **semmni** is causing spinlock contention.

### When Should the Value of This Tunable Be Lowered?

If **sysv_hash-locks** is no longer as large, and spinlock contention is no longer an issue.

### What Other Tunable Values Should Be Changed at the Same Time?

All the System V semaphore tunables are interrelated and should *not* be treated as independent variables. The tunables must be evaluated as a system to ensure they reflect the application requirements. The **sema** tunables include **semaem**, **semmap**, **semmni**, **semmns**, **semmnu**, **semmsl**, **semume**, **semvmx**, and **sysv_hash_locks**.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**sysv_hash_locks** was developed by AT&T.

## SEE ALSO

sema(5), semaem(5), semmap(5), semmni(5), semmns(5), semmnu(5), semmsl(5), semume(5), semvmx(5).

## NAME

tcphashsz - determines the size of the networking hash tables

## VALUES

### Failsafe

**0**

### Default

**2048**

### Allowed values

**256** to **65535**

### Recommended values

**2048**

Specify as a power of two or it will be rounded down to the nearest power of two.

## DESCRIPTION

This variable is used to set the size of the networking hash tables. A system that is going to have a large number of connections on it all of the time may see some benefit of increasing this value.

This tunable needs to be a power of two. If it is not specified as a power of two, then it is rounded *down* to the nearest power of two.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This value may be raised in the case of a machine with a large number of connections that last for an extended period of time.

### What Are the Side Effects of Raising the Value?

More memory is used when this value is raised.

### When Should the Value of This Tunable Be Lowered?

If a system is tight on memory and it has just a few connections on it, then lowering the value may reclaim some memory.

### What Are the Side Effects of Lowering the Value?

If set too low, the system will have long hash chains which will slow it down.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**tcphashsz** was developed by HP.

## NAME

timeslice - scheduling interval in clock ticks per second

## VALUES

### Failsafe

(**HZ/10**)

Where **HZ** defines the number of clock ticks per second for which the system is configured.

### Default

(**HZ/10**)

Where **HZ** is equal to **100**.

### Allowed values

Any value in the range of **-1** to **2147483647** is allowed.

A value of **-1** indicates no timeslice based scheduling preemption, and threads will continue to run until they voluntarily switch out or higher priority threads preempt them.

### Recommended values

Use the default value in normal cases. In special cases where quicker round robin scheduling is required, a value of **1** may be used. However, a change in value may have a direct impact on system performance. Customers must evaluate performance impact in their workload environment before changing the value on production systems.

## DESCRIPTION

The **timeslice** tunable defines the scheduling time interval that a thread may execute on a processor before the kernel scheduler will context switch out the thread for other same priority threads to run. When a thread starts executing on a processor, the thread is set up to run for the number of ticks in the **timeslice** tunable. On every clock interrupt that a thread is found executing, the time quantum balance for the thread is decremented, and when the balance reaches zero, the thread is context switched out.

The **timeslice** value controls one method of user preemption that the operating system implements. A larger value will reduce preemption of running threads; however, there are other reasons for user preemption of threads, and the **timeslice** tunable has no control there.

A change in the **timeslice** value may have direct impact on system throughput and response times. A very small value may result in too many context switches, and a very large value may result in the starvation of runnable threads.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

Since the **timeslice** tunable is globally applicable to all threads (except **sched_fifo**) in the system, irrespective of their scheduling policies and priorities. Any increase in value of this tunable will give equal time quantum boost to all threads.

If the system has too many context switches due to preemptions, caused by higher priority threads, you can raise the value to provide more time for lower priority threads to execute when they get scheduled, because higher priority threads will preempt the lower priority threads when they become runnable.

### What Are the Side Effects of Raising the Value?

Raising the value of the **timeslice** tunable may cause starvation of some threads, as they have to wait longer for their turn to execute. This may cause performance throughput issues.

### When Should the Value of This Tunable Be Lowered?

The **timeslice** tunable value should be lowered if better turnaround in response time is required at the cost of additional context switches. When the system does not have too many compute intensive applications, threads will block and preempt much more frequently without utilizing their complete time quantum.

### What Are the Side Effects of Lowering the Value?

The lowering of the `timeslice` tunable will result in many more context switches which will increase the time spent in SYSTEM space and less time spent in USER space. Also, the applications that are compute intensive will suffer performance degradation.

### What Other Tunable Values Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

`timeslice` was developed by HP.

## NAME

timezone, dst - difference between Universal (Greenwich mean) time and Local time

## VALUES

### Default

timezone = **420**

dst = **1**

A default value of **1**, indicates that the default value for the timezone tunable (420 minutes west of Greenwich, England ) is to be interpreted as a daylight savings time value.

### Allowed values

**-720** to **720**.

The value for the **timezone** tunable should correspond to a defined time zone, and thus should be at least a multiple of 15 minutes. More commonly the value will be a multiple of 60 or 30 minutes, corresponding to an hour or half-hour time zone.

The value of the **dst** tunable specifies if the time is with daylight savings or not. When the value is set to **1**, then the system is using daylight savings time, and if it is set to **0**, the system is not.

### Recommended values

Any allowed value is equally recommended. However, the chosen value should correspond to the time zone and daylight saving scheme of the system site or to a time zone and daylight saving scheme which has meaning for applications or users.

## DESCRIPTION

The **timezone** tunable is the difference between Greenwich mean time (Universal Time) and local time, expressed as minutes west of Greenwich, England. The **dst** tunable indicates whether the **timezone** tunable is to be interpreted as a standard or daylight savings time value.

These tunables provide a way to convert between Grenwich Mean Time (or Universal Time) and local time. Although **timezone** is a tunable, it is not used to affect the behavior of the system. Rather, it is used to remember time zone information when the system is rebooted. This information is returned by **gettimeofday** under HP-UX extensions. The **timezone** tunable is independent of other methods of indicating time zones. For example, **date** gets or sets the time in either Universal or local time by using the **TZ** environment variable. In this case the **timezone** tunable has no effect. In general, it is best to refer to the manpages for specific system calls such as **localtime**() and **tzset**() to check if there is a dependency on the **timezone** tunable.

### Who Is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to these tunables take effect at the next reboot.

### When Should the Value of This Tunable Be Changed?

These tunables may be changed to match the system's geographic time zone or the time zone of users.

### What Other Tunable Values Should Be Changed at the Same Time?

The **timezone** and **dst** tunables should always be set and interpreted as a pair of values.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**timezone** and **dst** were developed by HP.

## NAME

unlockable_mem - amount of physical memory that may not be locked by user processes

## VALUES

### Default

**0** pages

### Allowed values

Minimum: **0** pages

Maximum: Any value less than the available physical memory.

If **unlockable_mem** is set less than or equal to **0** (such as the default value), the kernel uses the value of:

**(200 + UPAGES / 2) * (***number-of-enabled-processors***)**

where **UPAGES** is the minimum amount of pages for thread user areas, and is set to **4** on 32-bit PA-RISC architectures, and **8** on 64-bit PA-RISC. On the IA64 architecture, **UPAGES** is the sum of the pages needed for the RSE stack, the traditional stack, the stack red zone (between the traditional and RSE stacks) and the size of the usual user area.

## DESCRIPTION

Memory locking allows the privileged user to specify which pages need to remain in memory, and unaffected by the swap process. This feature allows you to ensure that memory access times are unaffected by delays introduced by memory paging and swapping. For example, locking is a tool provided to privileged users on a system that is short on physical memory. Instead of having these privileged processes swap like the rest of the processes, they can lock portions of their address space. Once the pages are locked in for the privileged processes, they will no longer have to worry about memory contention. The unprivileged processes however, will have to compete for memory.

**unlockable_mem** provides you with a limiting factor on this privileged behavior, by setting the amount of memory which cannot be locked by user processes.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable should be raised on systems operating with low available physical memory where privileged processes are using memory which would be better used by processes which are considered more important, but which do not lock memory. Raising the tunable would force less pages locked and therefore usable by all processes.

Notice that none of this makes any difference to the kernel itself as it can steal pages from the locked pool when it needs to.

### What Are the Side Effects of Raising the Value?

Processes which seek to lock their memory have a smaller pool to work in, and if large amounts of locked memory are requested and the pages are simply in use, lock operations will fail. How the process handles the failure will vary by implementation.

### When Should the Value of This Tunable Be Lowered?

If the lockable memory pool is smaller than the demand, processes requesting locked memory will fail. Lowering **unlockable_mem** will provide more lockable memory.

### What Are the Side Effects of Lowering the Value?

Processes which do not use locked memory will be more likely to swap since less memory is available to the system for general usage, if privileged processes have locked the increased space in their pool.

### What Other Tunable Values Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**unlockable_mem** was developed by HP.

## NAME

vas_hash_locks - determines the size of the vas spinlock pool

## VALUES

### Default

**128**

### Allowed values

**64** to **262144**

## DESCRIPTION

Global kernel structures containing information on a running process or memory usage are frequently accessed or modified by several threads concurrently. To prevent race conditions, these structures are protected by spinlocks (kernel data structures used for synchronization) which allow only the spinlock 'holder' to proceed, while all others attempting to access the structure must wait.

Hashed spinlocks are used when each instance of such a data structure is to be protected, and there are several instances. Using a single spinlock for all instances would cause too much contention, but using one spinlock per structure wastes memory while the majority of the locks are unused at any given time.

By allocating a pool of hashed locks, a hash function picks one lock per group of structures, reducing contention while conserving memory. This tunable sets the size of such a pool for the vas data structure spinlocks.

### Who Is Expected to Change This Tunable?

Only HP Field engineers should change the value of this tunable.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable should be raised on systems with both low memory contention and high lock contention due to a large number of vas structures. Typically, such a system would be a large memory system with a large number of processes or threads.

### What Are the Side Effects of Raising the Value?

More memory is used by the kernel.

### When Should the Value of This Tunable Be Lowered?

This tunable should only be lowered on systems with high contention for physical memory and few processes and threads. The memory saved by making the table smaller will be freed for general use, and could cause contention if many vas structures are present and requiring locks.

What are the side-effects of lowering the value? Hash contention for vas locks is more likely, which increases the possibility of threads needing to wait on the lock shared between two or more vas structures.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**vas_hash_locks** was developed by HP.

## NAME

vps_ceiling - maximum (in kilobytes) of system-selectable page size

## VALUES

### Default

**16** KB

### Allowed values

Minimum: **4** KB
Maximum: **65536** KB

## DESCRIPTION

The Translation Look-aside Buffer (TLB) is a microprocessor feature for virtual memory, where the most recent physical to virtual address translations are cached, in the expectation that these translations are likely to be needed again soon. This is based on the principles of spatial and temporal locality of address references in programs. Historically, the TLB was entirely managed within hardware to achieve speed optimizations while sacrificing the flexibility of software implementations. For example, easily changed algorithms or table implementations.

In recent years, the flexibility of a software implementation of the TLB has regained importance over pure hardware speed. Specifically, the idea of logical grouping of physical frames (whose size is fixed in hardware) into 'superpages' or 'large pages', that can be represented in software TLB algorithms using a single base address translation for many physical frames, significantly reduces the lost cycles due to page faults (assuming reasonable spatial and temporal locality). For example, consider a scientific application working on an array where each element requires 1 KB of memory. Using the usual 4 KB physical frame size and referencing the array sequentially causes a page fault that requires the page be read into memory from disk or swap, and loads the TLB with the frame base address translation at every fifth element.

If a user application does not use the **chatr** command to specify a page size for the program text and data segments, the kernel automatically selects a page size based on system configuration and object size. This selected size is then compared to the maximum page size defined by the **vps_ceiling** tunable, and if the selected size is larger, the value of **vps_ceiling** is used instead. Then, the value is compared against the minimum page size as set by **vps_pagesize**, and the larger of the two values is used.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable can be raised when processes on the system access their text and data in a regular fashion, and over a range of data larger than the current value. For example, if this tunable is set to 16 KBs, but almost every process on the system repeatedly works with a four or five distinct 256 KB data sets, then raising the tunable to 256 would reduce the page faulting for these processes because 16 of the previously 16 KB pages are now addressed by a single 256 KB translation.

Average system behaviour is not likely to display uniformity of memory access and the optimal value is not easy to determine, so this tunable only represents the upper value for the kernel heuristic and may not change the actual system behaviour.

### What Are the Side Effects of Raising the Value?

Memory allocations will require larger groups of contiguous pages, if either **vps_pagesize** is also raised or the kernel heuristic chooses a larger value. This can lead to undesired behaviour. For example, when a program is reading in the last 4 KBs of code from disk with the default value, this means 16 KBs of contiguous physical memory must be found and set up with the appropriate virtual translation, even though, only 4 KBs of data will actually be on it. Consider the maximum, where 64 megabytes of contiguous physical memory is allocated for every new virtual page the program uses, even if, only 4 KBs of that is actually used. Besides the wasted physical memory here, there is also an issue of delays due to fragmentation that many contiguous frames of physical memory may not be available and a process may be stalled waiting on the allocation when the amount of memory it actually needs is available.

v

Therefore, it is best to only raise this tunable if you know precisely the memory usage of the system. In general, increasing the variable page size on a per application basis for known applications, such as, databases which scan large amounts of data with only one page fault, is a much better practice.

### When Should the Value of This Tunable Be Lowered?

The tunable should be lowered if physical memory fragmentation is preventing small memory processes from running due to waiting on contiguous chunks of memory, or if the overall system usage of memory displays poor spatial locality (virtual accesses are not close to each other) producing wasted physical frames.

### What Are the Side Effects of Lowering the Value?

Applications such as databases will suffer more page faults to get their working set into memory, but this can be handled by using **chatr** with the appropriate application.

### What Other Tunables Should Be Changed at the Same Time?

**vps_pagesize** should be considered, being the minimum bound on the kernel heuristic range.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**vps_ceiling** was developed by HP.

## SEE ALSO

vps_pagesize(5).

## NAME

vps_chatr_ceiling - maximum (in kilobytes) of user selectable page size

## VALUES

### Default

**1048576** (KB)

### Allowed values

Minimum: **4** (KB)

Maximum: **1048576** (KB)

## DESCRIPTION

The Translation Look-aside Buffer (TLB) is a microprocessor feature for virtual memory, where the most recent physical to virtual address translations are cached, in the expectation that these translations are likely to be needed again soon. This is based on the principles of spatial and temporal locality of address references in programs. Historically, the TLB was entirely managed within hardware to achieve speed optimizations while sacrificing the flexibility of software implementations. For example, easily changed algorithms or table implementations.

In recent years, the flexibility of a software implementation of the TLB has regained importance over pure hardware speed. Specifically, the idea of logical grouping of physical frames (whose size is fixed in hardware) into 'superpages' or algorithms using a single base address translation for many physical frames, significantly reduces the lost cycles due to page faults assuming reasonable spatial and temporal locality. For example, consider a scientific application working on an array where each element requires 1K of memory. Using the usual 4K physical frame size and referencing the array sequentially causes a page fault that requires the page be read into memory from disk or swap, and loads the TLB with the frame base address translation every fifth element.

This tunable sets the upper bound for virtual page size requested by a user application, as set with the **chatr** command on the binary.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable should be raised when a user application with known large memory set usage (such as a database) is expected to need larger pages than the current value allows.

### What Are the Side Effects of Raising the Value?

The side effects depend on the actual memory usage of the chatr'ed application, and if many users on the system **chatr** their applications for no good reason. In the first case, mistakenly chatr'ing an application to use a large page size (512 MB or more) when the application uses memory in a sparse pattern, or has a much smaller working set in general. For example, an application uses a shell script which only needs 64KB of memory total, or a scientific spare array analysis program that works on large data sets, but only on very small portions of the data where the rest can be swapped out or not even allocated. Setting this value for the application will result in several frames of physical memory being wasted if any of a virtual large page is in core memory then all of it must be there. This is true because a virtual large page must be constructed of contiguous physical frames, which may not always be available in the quantity desired, leading to unneeded allocation delays.

In the second case, where several users **chatr** their application for no good reason, keeping this tunable low minimizes the performance hit on the rest of the system.

### When Should the Value of This Tunable Be Lowered?

This tunable should be lowered if no user application actually needs large pages sized at the current tunable value to minimize the chance of a mistaken or malicious user causing wasted physical frames when using **chatr** with their applications.

### What Are the Side Effects of Lowering the Value?

The only side effect is that applications must run with smaller page sizes.

### What Other Tunables Should Be Changed at the Same Time?

None.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**vps_chatr_ceiling** was developed by HP.

## SEE ALSO

chatr(1).

## NAME

vps_pagesize - minimum (in kilobytes) of system-selected page size

## VALUES

### Default

**4** (KB)

### Allowed values

Minimum: **4** (KB)

Maximum: **65536** (KB)

## DESCRIPTION

The Translation Look-aside Buffer (TLB) is a microprocessor feature for virtual memory, where the most recent physical to virtual address translations are cached, in the expectation that these translations are likely to be needed again soon. This is based on the principles of spatial and temporal locality of address references in programs. Historically, the TLB were entirely managed within hardware to achieve speed optimizations while sacrificing the flexibility of software implementations. For example, easily changed algorithms or table implementations.

In recent years, the flexibility of a software implementation of the TLB has regained importance over pure hardware speed. Specifically, the idea of logical grouping of physical frames (whose size if fixed in hardware) into 'superpages' or 'large pages', that can be represented in software TLB algorithms using a single base address translation for many physical frames, significantly reduces the lost cycles due to page faults (assuming reasonable spatial and temporal locality). For example, consider a scientific application working on an array where each element requires 1 KB of memory. Using the usual 4 KB physical frame size and referencing the array sequentially causes a page fault that requires the page be read into memory from disk or swap, and loads the TLB with the frame base address translation at every fifth element.

If a user application does not use the **chatr** command to specify a page size for the program text and data segments, the kernel automatically selects a page size based on system configuration and object size. This selected size is then compared to the maximum page size defined by the **vps_ceiling** tunable, and if the selected size is larger, the value of **vps_ceiling** is used instead. Then, the value is compared against the minimum page size as set by **vsp_pagesize**, and the larger of the two values is used.

### Who is Expected to Change This Tunable?

Anyone.

### Restrictions on Changing

Changes to this tunable take effect at the next reboot.

### When Should the Value of This Tunable Be Raised?

This tunable can be raised when processes on the system access their text and data in a regular fashion, and over a range of data larger than the current value. For example, if this tunable is set to 16 KB, but almost every process on the system repeatedly works with a four or five distinct 256 KB data sets, then raising the tunable to 256 would reduce the page faulting for these processes because 16 of the previously 16 kilobyte pages are now addressed by a single 256 kilobyte translation.

|v|

Average system behavior is not likely to display uniformity of memory access and the optimal value is not easy to determine, so this tunable only represents the lower value for the kernel heuristic and may not change the actual system behavior.

### What Are the Side Effects of Raising the Value?

Memory allocations will require larger groups of contiguous pages because the kernel heuristic was not already choosing the larger value.

Requiring larger virtual pages may lead to undesirable system behaviour. This is especially true when many processes with small or fragmented data/code sets are active. Every virtual page referenced by the application, regardless of actual usage within that page, requires that the entire page work of contiguous physical frames of memory be present. For example, you cannot swap out half of a large virtual page. Many contiguous frames may not always be possible and may cause memory stalls on allocation that are not strictly needed. In addition, the waste of physical frames in this case would probably lead to increase swap usage, further degrading system performance.

### When Should the Value of This Tunable Be Lowered?

The tunable should be lowered if physical memory fragmentation is preventing small memory processes from running due to waiting on contiguous chunks of memory, or if the overall system usage of memory displays poor spatial locality (virtual accesses are not close to each other) producing wasted physical frames.

### What Are the Side Effects of Lowering the Value?

If **vps_ceiling** is lowered as well, applications with large data sets (such as databases) may suffer a performance degradation due to increased page faults. This can be corrected with a **chatr** of the appropriate application. If **vps_ceiling** is not modified, the side effects should be minimal as the kernel will now have a larger range to choose an appropriate page size for each non-chatr'ed application.

### What Other Tunables Should Be Changed at the Same Time?

**vsp_ceiling** should be considered, being the minimum bound on the kernel heuristic range.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**vps_pagesize** was developed by HP.

## SEE ALSO

vps_ceiling(5).

**(Tunable Kernel Parameters)**

## NAME

vxfs_max_ra_kbytes - maximum amount of read-ahead data, in KB, that kernel may have outstanding for a single VxFS file system

## VALUES

### Minimum

**0**

### Maximum

**65536**

### Default

**1024**

Specify an integer value or use an integer formula expression.

## DESCRIPTION

When data is read from a disk drive, the system may read additional data beyond that requested by the operation. This "read-ahead" speeds up sequential disk accesses, by anticipating that additional data will be read, and having it available in system buffers before it is requested. This parameter limits the number of read-ahead blocks that the kernel is allowed to have outstanding for any given VxFS filesystem. The limit applies to each individual VxFS filesystem, *not* to the system-wide total.

This parameter and **vxfs_ra_per_disk** work together to control the amount of read-ahead for VxFS filesystems. The maximum read-ahead for VxFS filesystems is the lesser of:

**vxfs_ra_per_disk *** number_of_disks_in_logical_volume

and

**vxfs_max_ra_kbytes**

In general, larger values of **vxfs_max_ra_kbytes** improve sequential I/O performance.

## EXAMPLE

With a 4-way parallel stripe of Seagate Barracuda 9GB drives, good performance can be achieved with **vxfs_max_ra_kbytes** set to 1024 and **vxfs_ra_per_disk** set to 256.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**vxfs_max_ra_kbytes** was developed by HP.

## SEE ALSO

hfs_max_ra_blocks(5), hfs_max_revra_blocks(5), hfs_ra_per_disk(5), hfs_revra_per_disk(5), vxfs_max_ra_kbytes(5), vxfs_ra_per_disk(5)

## NAME

vxfs_ra_per_disk - amount of VxFS file system read-ahead per disk, in KB

## Acceptable Values:

### Minimum

**0**

### Maximum

**8192**

### Default

**1024**

Specify an integer value or use an integer formula expression.

## DESCRIPTION

When data is read from a disk drive, the system may read additional data beyond that requested by the operation. This "read-ahead" speeds up sequential disk accesses, by anticipating that additional data will be read, and having it available in system buffers before it is requested. This parameter specifies the amount of read-ahead permitted per disk drive.

The total amount of read-ahead is determined by multiplying **vxfs_ra_per_disk** by the number of drives in the logical volume. If the filesystem does not reside in a logical volume, then the number of drives is effectively one.

The total amount of read-ahead that the kernel may have outstanding for a single VxFS filesystem is constrained by **vxfs_max_ra_kbytes**.

For more information, see the example in the description of *vxfs_max_ra_kbytes*(5).

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**vxfs_ra_per_disk** was developed by HP.

## SEE ALSO

hfs_max_ra_blocks(5), hfs_max_revra_blocks(5), hfs_ra_per_disk(5), hfs_revra_per_disk(5), vxfs_max_ra_kbytes(5), vxfs_ra_per_disk(5)

## NAME

vx_ncsize - the number of bytes to be reserved for the directory path-name cache used by the VxFS file system

## VALUES

Specify an integer value.

### Minimum

**`0`**

### Maximum

None

### Default

**`1024`**

## DESCRIPTION

The VxFS file system uses a name cache to store directory path name information related to recently accessed directories in the file system. Retrieving this information from a name cache allows the system to access directories and their contents without having to use direct disk accesses to find its way down a directory tree every time it needs to find a directory that is used frequently. Using a name cache in this way can save considerable overhead, especially in large applications such as databases where the system is repetitively accessing a particular directory or directory path.

**`vx_ncsize`** specifies how much space, in bytes, is set aside for the VxFS file system manager to use for this purpose. The default value is sufficient for most typical HP-UX systems, but for larger systems or systems with applications that use VxFS disk I/O intensively, some performance enhancement may result from expanding the cache size. The efficiency gained, however, depends greatly on the variety of directory paths used by the application or applications, and what percentage of total process time is expended while interacting with the VxFS file system.

## WARNINGS

All HP-UX kernel tunable parameters are release specific. This parameter may be removed or have its meaning changed in future releases of HP-UX.

## AUTHOR

**`vx_ncsize`** was developed by HP.

# Index

**Description** **Entry Name(Section)**

**Description** **Entry Name(Section)**

# Notes