Tru64 UNIX

Tuning Tru64 UNIX for Internet Servers

January 2003

Product Version: Tru64 UNIX Version 5.1B or earlier

This document contains information that systems engineers need when tuning the Tru64 UNIX operating system.

Hewlett-Packard Company
Palo Alto, CA

Contents

A Kernel Subsystem Attributes

B Revision History

Tables

About This Document

This document describes how to tune the Tru64 UNIX operating system to improve the performance of Internet servers, including Web servers, ftp servers, mail servers and relays, proxy servers, caching servers, gateway systems, and firewall systems.

Note

The recommendations result from testing Tru64 UNIX systems running Internet server software such as AltaVista (www.altavista.com). Not all recommendations are appropriate for all types of systems.

Audience

This document is intended for systems administrators who are responsible for managing a Tru64 UNIX operating system for Internet servers. Administrators should have an in-depth knowledge of their applications and users, in addition to operating system concepts, commands, and utilities. Such an understanding is crucial to successfully tuning a system for better performance.

New and Changed Features

There are no technical updates or changes for this version of the document.

Organization

This document is organized as follows:

Chapter 1	Provides information on how to improve Internet server performance.
Chapter 2	Provides tuning recommendations and examples on how to tune your system.
Appendix A	Lists each attribute and the version of Tru64 UNIX that supports them. Compares the attribute names for different versions of Tru64 UNIX and identifies which attributes can be tuned at run time.
Appendix B	Gives a revision history of the document.

Related Documentation

The following documentation supplements information in this document:

Best Practices

For information on improving large-memory system performance, refer to the Improving Large-Memory System Performance Best Practice at the following URL:

`http://www.tru64unix.compaq.com/docs/best_practices/VLM_BP/TITLE.HTM`

Manuals

The following Tru64 UNIX operating system manuals provide important information that supplements the information in this document:

- *System Configuration and Tuning* contains detailed information on configuring and tuning high-performance and high availability systems.
- *System Administration* describes how to configure, use, and maintain the Tru64 UNIX operating system.

You can access these manuals on line for you version of the operating system at the following URL:

`http://www.tru64unix.compaq.com/docs/pub_page/doc_list.html`

Reference Pages

The following Tru64 UNIX reference pages contain descriptions of the attributes:

- `sys_attrs_generic`(5)
- `sys_attrs_inet`(5)
- `sys_attrs_net`(5)
- `sys_attrs_proc`(5)
- `sys_attrs_socket`(5)
- `sys_attrs_vm`(5)

You can access the reference pages on line for your version of the operating system at the following URL:

`http://www.tru64unix.compaq.com/docs/pub_page/doc_list.html`

Reader's Comments

We welcome any comments and suggestions you have on this and other Tru64 UNIX manuals.

You can send your comments in the following ways:

- Fax: 603-884-0120 Attn: UBPG Publications, ZKO3-3/Y32
- Internet electronic mail: `readers_comment@zk3.dec.com`

  A Reader's Comment form is located on your system in the following location:

  `/usr/doc/readers_comment.txt`

Please include the following information along with your comments:

- The full title of the book and the order number. (The order number is printed on the title page of this book and on its back cover.)
- The section numbers and page numbers of the information on which you are commenting.
- The version of Tru64 UNIX that you are using.
- If known, the type of processor that is running the Tru64 UNIX software.

The Tru64 UNIX Publications group cannot respond to system problems or technical support inquiries. Please address technical questions to your local system vendor or to the appropriate Hewlett-Packard Company technical support office. Information provided with the software media explains how to send problem reports to Hewlett-Packard Company.

Conventions

This document uses the following conventions:

. . .	A vertical ellipsis indicates that a portion of an example that would normally be present is not shown.
. . .	In syntax definitions, a horizontal ellipsis indicates that the preceding item can be repeated one or more times.
file	Italic (slanted) type indicates variable values, placeholders, and function parameter names.
`buf`	In function definitions and syntax definitions used in driver configuration, this typeface indicates names that you must type exactly as shown.
[ ]	In formal parameter declarations in function definitions and in structure declarations, brackets

indicate arrays. Brackets also specify ranges for device minor numbers and device special files in file fragments. However, for the syntax definitions that are used in driver configuration, these brackets indicate items that are optional.

| Vertical bars separating items that appear in the syntax definitions used in driver configuration indicate that you choose one item from among those listed.

1

Improving Internet Server Performance

This chapter describes how to improve your Internet server performance. It offers various configuration guidelines and describes several monitoring tools, including the following topics:

- Configuring hardware (Section 1.1)
- Configuring memory and swap space (Section 1.2)
- Logging IP addresses (Section 1.3)
- Monitoring network statistics (Section 1.4)
- Monitoring socket statistics (Section 1.5)
- Monitoring virtual memory statistics (Section 1.6)
- Gathering configuration information (Section 1.7)

Note

Install the latest release of Tru64 UNIX and the latest patches that are recommended for your operating system version.

1.1 Configuring Hardware

The following hardware configuration guidelines can help to improve Internet server performance:

- Make sure you have the latest version of the firmware for your system, disks, adapters, and controllers.
- Ensure that you have sufficient memory and swap space to handle the workload. Refer to Section 1.2 for more information.
- Use high-performance storage hardware, including disks, adapters, and controllers in your Internet server configuration.
- Use Logical Storage Manager (LSM) or hardware RAID storage configurations for high performance and high availability.
- Use write-back caches in hardware RAID configurations to significantly improve Internet server performance.
- Place the `/tmp` and `/var/tmp` directories on different file systems and, if possible, different disks. For optimal performance, place the directories

on disks under control of a RAID controller with the write-back cache option enabled.

1.2 Configuring Memory and Swap Space

You must provide sufficient memory and swap space to handle the server workload. Insufficient memory resources and swap space will cause performance problems. To configure memory and swap space, follow these steps:

1. Determine how much physical memory your workload requires.
2. Choose a swap space allocation mode, either immediate or deferred.
3. Determine how much swap space you need.
4. Configure the swap space in order to efficiently distribute the disk I/O.

In addition to the memory needed for system and application operations, each connection to an Internet server requires memory resources for the following:

- Kernel socket structure
- Internet protocol control block (`inpcb`) structure
- TCP control block structure
- Any socket buffer space that is needed as packets arrive and are consumed

These memory resources total 1 KB for each connection endpoint (not including the socket buffer space), which means that you will need 10 MB of memory to accommodate 10,000 connections.

You must ensure that your server has enough memory to handle demanding peak loads. Configure ten times more memory than what the server requires on a busy day, so that you have sufficient memory to handle occasional spikes of activity.

There are no limitations on a server's ability to handle millions of TCP connections if memory resources are available to service the connections. However, if you do not have sufficient memory, the server will reject new connection requests until enough existing connections are freed. Use the `netstat -m` command to monitor the memory that is currently being used by the network subsystem. Refer to Section 1.4 for more information on the `netstat` command.

1.3 Logging IP Addresses

If your Internet server logs client host names, the application software may force the system to perform a reverse DNS lookup in order to obtain the client's host name. Reverse DNS lookups are time-intensive and may cause performance problems on busy servers with many clients.

You can modify the Internet software to log client Internet Protocol (IP) addresses, instead of client host names, without losing any significant information. Logging IP addresses may significantly improve the efficiency of the Internet server.

Consult the documentation provided by the Internet server software vendor to determine how to disable the logging of client host names. For example, you can obtain information about modifying Apache HTTP Server software from the Apache HTTP Server documentation site at this URL:

```
http://httpd.apache.org/docs/
```

1.4 Monitoring Network Statistics

The `netstat` command displays network statistics, including information about network routes and active sockets for each protocol. The command also displays cumulative statistics for network interfaces, including the number of incoming and outgoing packets and packet collisions, information about memory used for network operations, and statistics related to IP, ICMP, TCP, and UDP protocol layers.

You can use the `netstat` command to check the following:

- Input and output errors and collisions (Section 1.4.1)
- Network memory usage (Section 1.4.2)
- Socket connections (Section 1.4.3)
- Device driver errors (Section 1.4.4)
- Dropped or lost packets (Section 1.4.5)
- Retransmissions, out-of-order packets, and bad checksums (Section 1.4.6)
- Routing statistics (Section 1.4.7)
- Network protocol statistics (Section 1.4.8)

The following sections describe how to use `netstat` to check for these conditions. Refer to `netstat`(1) for your version of the operating system for more information:

```
http://www.tru64unix.compaq.com/docs/pub_page/doc_list.html
```

1.4.1 Input and Output Errors and Collisions

Network collisions are a normal part of network operations. A collision can occur when two or more Ethernet stations attempt to transmit simultaneously on the network. If a station is unable to access the network because another one is already using it, the station will stop trying to access the network for a short period of time, before attempting to access the network again. A collision occurs each time a station fails to access the network. Most network cards will attempt to transmit a maximum of 15 times, after which they will drop the output packet and issue an `excessive collisions` error.

Use the output of the `netstat -i` command to check for input errors (`Ierrs`), output errors (`Oerrs`), and collisions (`Coll`). Compare the values in these fields with the total number of packets sent. High values may indicate a network problem. For example, cables may not be connected properly or the Ethernet may be saturated. A collision rate of up to 10 percent may not indicate a problem on a busy Ethernet. However, a collision rate of more than 20 percent could indicate a problem. For example:

```
# netstat -i
Name  Mtu  Network  Address             Ipkts Ierrs    Opkts  Oerrs  Coll
tu0   1500  Link    00:00:aa:11:0a:c1       0     0    43427  43427     0
tu0   1500  DLI     none                    0     0    43427  43427     0
tu1   1500  Link    bb:00:03:01:6c:4d  963447   138   902543  1118  80006
tu1   1500  DLI     none               963447   138   902543  1118  80006
tu1   1500  o-net   plume              963447   138   902543  1118  80006
.
.
.
```

1.4.2 Memory Usage

The `netstat -m` command shows statistics for network-related memory structures, including the memory that is being used for `mbuf` clusters. Use this command to determine if the network is using an excessive amount of memory in proportion to the total amount of memory installed in the system. If the `netstat -m` command shows several requests for memory (`mbuf`) clusters delayed or denied, this means that your system was temporarily short of physical memory. The following example is from a firewall server with 128 MB memory that does not have `mbuf` cluster compression enabled:

```
# netstat -m
 2521 Kbytes for small data mbufs (peak usage 9462 Kbytes)
78262 Kbytes for mbuf clusters (peak usage 97924 Kbytes)
 8730 Kbytes for sockets (peak usage 14120 Kbytes)
 9202 Kbytes for protocol control blocks (peak usage 14551
    2 Kbytes for routing table (peak usage 2 Kbytes)
    2 Kbytes for socket names (peak usage 4 Kbytes)
    4 Kbytes for packet headers (peak usage 32 Kbytes)
39773 requests for mbufs denied
    0 calls to protocol drain routines
98727 Kbytes allocated to network
```

The previous example shows that 39,773 requests for memory were denied. This indicates a problem because this value should be 0. The example also shows that 78 MB of memory has been assigned to `mbuf` clusters, and that 98 MB of memory is being consumed by the network subsystem.

If you increase the value of the `socket` subsystem attribute `sbcompress_threshold` to 600, the memory allocated to the network subsystem immediately decreases to 18 MB, because compression at the kernel socket buffer interface results in a more efficient use of memory. Refer to Section 2.2.3.3 for more information on the `sbcompress_threshold` attribute.

1.4.3 Socket Connections

Each socket results in a network connection. If the system allocates an excessive number of sockets, use the `netstat -an` command to determine the state of your existing network connections. The following example shows the contents of the protocol control block table and the number of TCP connections currently in each state:

```
# netstat -an | grep tcp | awk '{print $6}' | sort | uniq -c
    1 CLOSE_WAIT
   58 ESTABLISHED
   12 FIN_WAIT_1
    8 FIN_WAIT_2
   17 LISTEN
    1 SYN_RCVD
15749 TIME_WAIT
#
```

For Internet servers, the majority of connections usually are in a `TIME_WAIT` state. If the number of entries in the `FIN_WAIT_1` and `FIN_WAIT_2` fields represent a large percentage of the total connections (add together all of the fields), you may want to enable keepalive, as described in Section 2.3.2.5.

If the number of entries in the `SYN_RCVD` field represents a large percentage of the total connections, the server may be overloaded or may be experiencing TCP SYN attacks.

Note that in this example, there are almost 16,000 sockets being used, which requires 16 MB of memory. Refer to Section 1.2 for more information on configuring memory and swap space.

1.4.4 Device Driver Errors

Use the `netstat -is` command to check for network device driver errors. For example:

```
netstat -is
tu0 Ethernet counters at Tue Aug  3 13:57:35 1999
          191 seconds since last zeroed
     14624204 bytes received
      4749029 bytes sent
```

```
  34784 data blocks received
  11017 data blocks sent
2197154 multicast bytes received
  17086 multicast blocks received
   1894 multicast bytes sent
     17 multicast blocks sent
    932 blocks sent, initially deferred
    347 blocks sent, single collision
    666 blocks sent, multiple collisions
      0 send failures
      0 collision detect check failure
      1 receive failures, reasons include: Frame too long
      0 unrecognized frame destination
      0 data overruns
      0 system buffer unavailable
      0 user buffer unavailable
```

The previous example shows that the system sent 11,017 blocks. Of those blocks, 1,013 (347 + 666) blocks had collisions, which represents approximately 10 percent of the blocks sent. A collision rate of up to 10 percent may not indicate a problem on a busy Ethernet. However, a collision rate of more then 20 percent could indicate a problem.

In addition, the following fields should be 0 or a low single-digit number:

- send failures
- receive failures
- data overruns
- system buffer unavailable
- user buffer unavailable

1.4.5 Dropped or Lost Packets

Use the `netstat -p ip` command to check for bad checksums, length problems, excessive redirects, and packets lost because of resource problems for the IP protocol. Check the output for a nonzero number in the lost packets due to resource problems field. For example:

```
# netstat -p ip
ip:
        259201001 total packets received
        0 bad header checksums
        0 with size smaller than minimum
        0 with data size < data length
        0 with header length < data size
        0 with data length < header length
        25794050 fragments received
        0 fragments dropped (duplicate or out of space)
        802 fragments dropped after timeout
        0 packets forwarded
        67381376 packets not forwardable
                67381376 link-level broadcasts
        0 packets denied access
        0 redirects sent
        0 packets with unknown or unsupported protocol
        170988694 packets consumed here
```

```
160039654 total packets generated here
0 lost packets due to resource problems
4964271 total packets reassembled ok
2678389 output packets fragmented ok
14229303 output fragments created
0 packets with special flags set
```

Use the `netstat -id` command to monitor dropped output packets. Examine the output for a nonzero value in the `Drop` column. If a nonzero value appears in the `Drop` column for an interface, you may want to increase the value of the `ifqmaxlen` kernel variable to prevent dropped packets. Refer to Section 2.3.2.9 for more information on this attribute.

The following example shows 4,221 dropped output packets on the `tu1` network interface:

```
# netstat -id
Name Mtu  Network Address             Ipkts Ierrs  Opkts Oerrs  Coll  Drop
tu0 1500  Link    00:00:f8:06:0a:b1       0     0  98129 98129     0     0
tu0 1500  DLI     none                    0     0  98129 98129     0     0
tu1 1500  Link    aa:00:04:00:6a:4e  892390   785 814280 68031 93848  4221
tu1 1500  DLI     none               892390   785 814280 68031 93848  4221
tu1 1500  orange  flume              892390   785 814280 68031 93848  4221
  .
  .
  .
```

The output of the previous command shows that the `Opkts` and `Oerrs` fields have the same values for the `tu0` interface, which indicates that the Ethernet cable is not connected. In addition, the value of the `Oerrs` field for the `tu1` interface is 68,031, which is a high error rate. Use the `netstat -is` command to obtain detailed error information.

1.4.6 Retransmissions, Out-of-Order Packets, and Bad Checksums

Use the `netstat -p tcp` command to check for retransmissions, out-of-order packets, and bad checksums for the TCP protocol. Use the `netstat -p udp` command to look for bad checksums and full sockets for the UDP protocol. You can use the output of these commands to identify network performance problems by comparing the values in some fields to the total number of packets sent or received.

For example, an acceptable percentage of retransmitted packets or duplicate acknowledgements is 2 percent or less. An acceptable percentage of bad checksums is 1 percent or less.

In addition, a large number of entries in the `embryonic connections dropped` field may indicate that the listen queue is too small or server performance is slow and clients have canceled requests. Other important fields to examine include the `completely duplicate packets`, `out-of-order packets`, and `discarded` fields. For example:

```
# netstat -p tcp
tcp:
```

```
66776579 packets sent
        58018945 data packets (1773864027 bytes)
        54447 data packets (132256902 bytes) retransmitted
        5079202 ack-only packets (3354381 delayed)
        29 URG only packets
        7266 window probe packets
        2322828 window update packets
        1294022 control packets
40166265 packets received
        29455895 acks (for 1767211650 bytes)
        719524 duplicate acks
        0 acks for unsent data
        19788741 packets (2952573297 bytes) received in-sequence
        123726 completely duplicate packets (9224858 bytes)
        2181 packets with some dup. data (67344 bytes duped)
        472000 out-of-order packets (85613803 bytes)
        1478 packets (926739 bytes) of data after window
        43 window probes
        201331 window update packets
        1373 packets received after close
        118 discarded for bad checksums
        0 discarded for bad header offset fields
        0 discarded because packet too short
448388 connection requests
431873 connection accepts
765040 connections established (including accepts)
896693 connections closed (including 14570 drops)
86298 embryonic connections dropped
25467050 segments updated rtt (of 25608120 attempts)
106020 retransmit timeouts
        145 connections dropped by rexmit timeout
6329 persist timeouts
37653 keepalive timeouts
        15536 keepalive probes sent
        16874 connections dropped by keepalive
```

The output of the previous command shows that, out of the 58,018,945 data packets that were sent, 54,447 packets were retransmitted, which is a percentage that is within the acceptable limit of 2 percent.

In addition, the command output shows that, out of the 29,455,895 acknowledgments, 719,524 were duplicates, which is a percentage that is slightly larger than the acceptable limit of 2 percent.

Important fields for the `netstat -p udp` command include the `incomplete headers`, `bad data length fields`, `bad checksums`, and `full sockets` fields, which should have low values. The `no port` field specifies the number of packets that arrived destined for a nonexistent port (for example, `rwhod` or `routed` broadcast packets) and were subsequently discarded. A large value for this field is normal and does not indicate a performance problem. For example:

```
# netstat -p udp
udp:
        144965408 packets sent
        217573986 packets received
        0 incomplete headers
        0 bad data length fields
        0 bad checksums
        5359 full sockets
```

```
        28001087 for no port (27996512 broadcasts, 0 multicasts)
        0 input packets missed pcb cache
```

The previous example shows a value of 5,359 in the `full sockets` field, which indicates that the UDP socket buffer may be too small.

1.4.7 Routing Statistics

Use the `netstat -rs` command to obtain routing statistics. The value of the `bad routing redirects` field should be small. A large value may indicate a serious network problem. For example:

```
# netstat -rs
routing:
        0 bad routing redirects
        0 dynamically created routes
        0 new gateways due to redirects
        1082 destinations found unreachable
        0 uses of a wildcard route
```

1.4.8 Protocol Statistics

Use the `netstat -s` command to simultaneously display statistics related to the IP, ICMP, IGMP, TCP, and UDP protocol layers. For example:

```
# netstat -s
ip:
        377583120 total packets received
        0 bad header checksums
        7 with size smaller than minimum
        0 with data size < data length
        0 with header length < data size
        0 with data length < header length
        12975385 fragments received
        0 fragments dropped (dup or out of space)
        3997 fragments dropped after timeout
        523667 packets forwarded
        108432573 packets not forwardable
        0 packets denied access
        0 redirects sent
        0 packets with unknown or unsupported protocol
        259208056 packets consumed here
        213176626 total packets generated here
        581 lost packets due to resource problems
        3556589 total packets reassembled ok
        4484231 output packets fragmented ok
        18923658 output fragments created
        0 packets with special flags set
icmp:
        4575 calls to icmp_error
        0 errors not generated because old ip message was too short
        0 errors not generated because old message was icmp
        Output histogram:
                echo reply: 586585
                destination unreachable: 4575
                time stamp reply: 1
        0 messages with bad code fields
        0 messages < minimum length
        0 bad checksums
        0 messages with bad length
```

```
        Input histogram:
                echo reply: 612979
                destination unreachable: 147286
                source quench: 10
                echo: 586585
                router advertisement: 91
                time exceeded: 231
                time stamp: 1
                time stamp reply: 1
                address mask request: 7
        586586 message responses generated
igmp:
        0 messages received
        0 messages received with too few bytes
        0 messages received with bad checksum
        0 membership queries received
        0 membership queries received with invalid field(s)
        0 membership reports received
        0 membership reports received with invalid field(s)
        0 membership reports received for groups to which we belong
        0 membership reports sent
tcp:
        66818923 packets sent
                58058082 data packets (1804507309 bytes)
                54448 data packets (132259102 bytes) retransmitted
                5081656 ack-only packets (3356297 delayed)
                29 URG only packets
                7271 window probe packets
                2323163 window update packets
                1294434 control packets
        40195436 packets received
                29477231 acks (for 1797854515 bytes)
                719829 duplicate acks
                0 acks for unsent data
                19803825 packets (2954660057 bytes) received in-sequence
                123763 completely duplicate packets (9225546 bytes)
                2181 packets with some dup. data (67344 bytes duped)
                472188 out-of-order packets (85660891 bytes)
                1479 packets (926739 bytes) of data after window
                43 window probes
                201512 window update packets
                1377 packets received after close
                118 discarded for bad checksums
                0 discarded for bad header offset fields
                0 discarded because packet too short
        448558 connection requests
        431981 connection accepts
        765275 connections established (including accepts)
        896982 connections closed (including 14571 drops)
        86330 embryonic connections dropped
        25482179 segments updated rtt (of 25623298 attempts)
        106040 retransmit timeouts
                145 connections dropped by rexmit timeout
        6329 persist timeouts
        37659 keepalive timeouts
                15537 keepalive probes sent
                16876 connections dropped by keepalive
udp:
        145045792 packets sent
        217665429 packets received
        0 incomplete headers
        0 bad data length fields
        0 bad checksums
        5359 full sockets
```

```
28004209 for no port (27999634 broadcasts, 0 multicasts)
0 input packets missed pcb cache
```

1.5 Monitoring Socket Statistics

Three `socket` subsystem attributes monitor socket listen queue events:

- The `sobacklog_hiwat` attribute counts the maximum number of pending requests to any server socket.
- The `sobacklog_drops` attribute counts the number of times the system dropped a received SYN packet, because the number of queued SYN_RCVD connections for a socket equaled the socket's backlog limit.
- The `somaxconn_drops` attribute counts the number of times the system dropped a received SYN packet, because the number of queued SYN_RCVD connections for the socket equaled the upper limit on the backlog length (`somaxconn` attribute).

The initial value of these attributes at boot time is 0. Use the `sysconfig -q` socket command to display the current attribute values. If the values show that the queues are overflowing, you may need to increase the socket listen queue limit. For example:

```
# sysconfig -q socket
socket:
pftimerbindcpu = 0
sbcompress_threshold = 0
sb_max = 1048576
sobacklog_drops = 0
sobacklog_hiwat = 21
somaxconn = 65535
somaxconn_drops = 0
sominconn = 65535
mbuf_ext_lock_count = 64
umc_min_len = 1024
umc = 0
```

It is recommended that the value of the `sominconn` attribute equal the value of the `somaxconn` attribute. If so, the value of `somaxconn_drops` will have the same value as `sobacklog_drops`.

However, if the value of the `sominconn` attribute is 0 (the default), and if one or more server applications uses an inadequate value for the backlog argument to its `listen` system call, the value of `sobacklog_drops` may increase at a rate that is faster than the rate at which the `somaxconn_drops` counter increases. If this occurs, you may want to increase the value of the `sominconn` attribute. Refer to Section 2.2.3.2 for more information on the `sominconn` attribute.

1.6 Monitoring Virtual Memory Statistics

The vmstat command provides data on virtual memory usage. This may help you determine if a system is paging excessively, which can degrade Internet server performance. For example:

```
# vmstat 1
Virtual Memory Statistics: (pagesize = 8192)
procs    memory         pages                            intr        cpu
r  w  u  act  free wire fault cow zero react pin pout  in  sy  cs  us  sy  id
7 526 59 80K  758  45K  402M  94M 132M   1M  74M 139K 757 42K  1K  38  14  48
7 526 59 81K  278  45K   939   15  896    0   11    0 824  2K  1K  85  11   4
6 528 59 81K  285  45K   595   67  411    0   10    0 983  5K  2K  81  17   2
7 526 59 81K  353  45K   560   31  446    0   17    0 781  2K  1K  87  10   3
7 526 59 81K  353  45K   406    0  406    0    0    0  1K  4K  2K  85  13   2
7 527 59 81K  288  45K   406    0  406    0    0    0  1K  7K  4K  81  18   1
9 524 59 81K  350  45K   640   72  420    0   13    0 999  3K  2K  85  13   2
 .
 .
 .
```

The values in the memory fields are specified in 8-KB pages. Check the size of the free page list (free). Compare the number of free pages to the values for the active pages (act) and the wired pages (wire). The sum of the free, active, and wired pages should be close to the amount of physical memory in your system. Although the value for free should be small, if the value is consistently small (less than 128 pages) and accompanied by excessive paging and swapping, you may have a physical memory shortage.

Also, examine the pageout (pout) field. If the number of pageouts is consistently high, you may have insufficient memory. You also may have insufficient swap space or your swap space may be configured inefficiently. Use the swapon -s command to display your swap device configuration, and use the iostat command to determine which swap disk is being used the most.

Refer to vmstat(1), swapon(8), and iostat(1) for your version of the operating system for more information:

http://www.tru64unix.compaq.com/docs/pub_page/doc_list.html

1.7 Gathering Configuration Information

The sys_check script is a ksh script that gathers configuration information and formats this information into an HTML file. It warns you if it detects configuration problems, checks your kernel subsystem attribute settings, and provides attribute tuning recommendations.

Be sure to use the latest version of sys_check. You can obtain this at:

http://www.tru64unix.compaq.com/sys_check/sys_check.html

2

Tuning Recommendations

This chapter offers tuning recommendations, including the following topics:

- Displaying and modifying kernel subsystem attributes (Section 2.1)
- Primary tuning recommendations (Section 2.2)
- Advanced tuning recommendations (Section 2.3)

Not all recommendations apply to all configurations, and some provide only marginal performance improvements. Therefore, you must fully understand your configuration and workload, and then carefully read the documentation before applying any recommendation.

Note

Some attribute names have changed for Tru64 UNIX Version 5.0 and later. Refer to Table A–2 for more information on attribute name equivalences.

2.1 Displaying and Modifying Kernel Subsystem Attributes

The operating system includes various subsystems that define or extend the kernel. Kernel subsystem attributes are used to set kernel variables, which control subsystem behavior or track subsystem statistics. Attributes are assigned default values at boot time. For Internet servers, the default values of some attributes may not be appropriate, so you must modify these values to provide optimal performance.

To display and modify kernel subsystem attributes, follow these steps:

1. Determine the operating system support for an attribute (Section 2.1.1)
2. Display the attribute values (Section 2.1.2)
3. Modify the attribute values (Section 2.1.3)

2.1.1 Operating System Support for Attributes

Some older versions of Tru64 UNIX do not support the attributes described in this document. Others restrict the methods that you can use to modify

and display attributes, or they require operating system patches to use the attributes.

To determine if your version of the operating system supports a particular kernel subsystem attribute, use one of the following methods:

- Refer to Table A–1 for support information.
- Use the `sysconfig -q` *subsystem* [*attribute*] command.

  If you do not specify an attribute, the system displays all the subsystem attributes that can be modified with the `sysconfig` or `sysconfigdb` command. If the subsystem is not configured, the operating system displays a message similar to the following:

  ```
  framework error: subsystem 'inet' not found
  ```

  If you specify an attribute, only the information specific to that attribute is displayed. For example:

  ```
  # sysconfig -q inet tcbhashsize
  inet:
  tcbhashsize = 32
  ```

  If the attribute is not supported or if it cannot be accessed by using `sysconfig`, the operating system displays a message similar to the following:

  ```
  inet:
  tcbhashsize = unknown attribute
  ```

- Use the `dbx p` (print) command. If you cannot access the attribute, the operating system displays a message stating that the attribute is not defined or active. For example:

  ```
  # dbx -k /vmunix
  dbx version 3.11.10
  Type 'help' for help.

  (dbx) p tcp_keepalive_default
  "tcp_keepalive_default" is not defined or not active
  (dbx)
  ```

Refer to `sysconfig`(8) and `dbx`(1) for your version of the operating system for more information:

```
http://www.tru64unix.compaq.com/docs/pub_page/doc_list.html
```

2.1.2 Displaying Attribute Values

There are various methods that you can use to display the current value of a kernel subsystem attribute and other descriptive information. Use the following methods to display attribute values:

- The Kernel Tuner (`dxkerneltuner`) graphical user interface (GUI) to display permanent, current (run-time), minimum, and maximum values of attributes. Access the GUI through the Common Desktop Environment (CDE) Application Manager window; select the `System_Admin` icon,

and then select the MonitoringTuning icon. You can then choose the subsystem whose attributes you want to display.

- The `sysconfig -q` *subsystem* [*attribute*] command to display the current (run-time) value of the specified attribute or, if an attribute is not specified, all the attributes for the specified subsystem:

  ```
  sysconfig -q subsystem [attribute]
  ```

  For example:

  ```
  # sysconfig -q vm ubc_maxpercent
  vm:
  ubc_maxpercent = 100
  ```

- The `sysconfig -Q` *subsystem* [*attribute*] command to display the maximum and minimum values of the attributes for the specified subsystem. If you specify a particular attribute, the system displays information only for that attribute.

  The command output also includes information about the operations that you can perform on the attribute. The following lists various operations:

  – `C` - The attribute can be modified when the subsystem is initially loaded; that is, the attribute supports boot time, permanent modifications.

  – `R` - The attribute can be tuned at run time; that is, you can modify the value that the system is currently using.

  – `Q` - The attribute's current value can be displayed (queried).

  For example:

  ```
  # sysconfig -Q vfs bufcache
  vfs:
  bufcache -     type=INT op=CQ min_val=0 max_val=50
  ```

  The output of the previous command shows that the minimum value of the `bufcache` attribute is 0 and the maximum value is 50. The output also shows that you cannot modify the current (run-time) value.

For some older versions of the operating system, you must use the `dbx p` (print) command to display the current value of a kernel variable, instead of an attribute. For example:

```
# dbx -k /vmunix
dbx version 3.11.10
Type 'help' for help.

(dbx) p ipport_userreserved
5000
(dbx)
```

Refer to `dxkerneltuner`(8X), `sysconfig`(8), and `dbx`(1) for your version of the operating system for more information:

```
http://www.tru64unix.compaq.com/docs/pub_page/doc_list.html
```

2.1.3 Modifying Attribute Values

There are various methods you can use to modify attribute values. The method you use depends on the version of the operating system you are running, as specified in Table A–1, and whether you want to modify the current (run-time) value of an attribute or modify an attribute's permanent value. You must be root to modify attribute values.

The following sections describe how to modify the current and permanent values.

2.1.3.1 Current Value

In some cases, you can modify the current (run-time) value of an attribute. This allows you to determine if modifying an attribute will improve your system performance without rebooting the system. Not all attributes can be tuned at run time, and the temporary modifications are lost when you reboot the system. Refer to Table A–2 or use the `sysconfig -Q` command to determine whether an attribute can be tuned at run time.

To modify an attribute's current (run-time) value, use one of the following methods:

- The Kernel Tuner (`dxkerneltuner`) GUI, if the attribute supports this operation. Access the GUI through the Common Desktop Environment (CDE) Application Manager window; select the `System_Admin` icon, and then select the `MonitoringTuning` icon. Choose the subsystem whose attribute you want to modify, and enter the new value in the `Current Value` field.
- The `sysconfig -r` command, if the attribute supports this operation. Use the following command syntax:

  ```
  sysconfig -r subsystem attribute=value
  ```

  For example:

  ```
  # sysconfig -r inet tcp_keepinit=30
  tcp_keepinit: reconfigured
  ```

For some older versions of the operating system, you must use the `dbx assign` command to modify the current value of a kernel variable, instead of an attribute. However, modifications made with the `dbx assign` command are lost when you reboot the system. Use the following command syntax:

```
dbx assign attribute=value
```

For example:

```
# dbx -k /vmunix
dbx version 3.11.10
Type 'help' for help.

(dbx) assign ipport_userreserved=60000
60000
```

```
(dbx)
```

Refer to dxkerneltuner(8X), sysconfig(8), and dbx(1) for your version of the operating system for more information:

```
http://www.tru64unix.compaq.com/docs/pub_page/doc_list.html
```

2.1.3.2 Permanent Value

To modify an attribute's permanent (boot-time) value, the sysconfigtab file must contain the subsystem name, the attribute name, and the value of the attribute. Do not manually modify the sysconfigtab file. To make these modifications, use one of the following methods:

- The Kernel Tuner (dxkerneltuner) GUI. Access the GUI through the Common Desktop Environment (CDE) Application Manager window, select the System_Admin icon, and then select the MonitoringTuning icon. Choose the subsystem whose attribute you want to modify, and enter the new value in the Boot Time Value field.
- The sysconfigdb command. Use the following command syntax:

  ```
  sysconfigdb -a -f stanza_file subsystem
  ```

  The *stanza_file* is a specially formatted file that contains the name of the subsystem and a list of attributes and their values. This file is merged into the sysconfigtab file. Refer to stanza(4) for your version of the operating system for more information:

  ```
  http://www.tru64unix.compaq.com/docs/pub_page/doc_list.html
  ```

  To use the new attribute value, you must invoke the sysconfig -r command if the attribute can be tuned at run time, or reboot the system.

In addition, you can use the dbx patch command to modify the value of a variable, as well as the on-disk /vmunix image value.

Refer to dxkerneltuner(8X), sysconfig(8), and sysconfigdb(8) for more information. Refer to the *System Administration* manual for your version of the operating system for information about modifying the system configuration file:

```
http://www.tru64unix.compaq.com/docs/pub_page/doc_list.html
```

2.2 Primary Tuning Recommendations

There are many kernel subsystem attributes that affect Internet server performance. Internet servers include Web servers, ftp servers, mail servers and relays, proxy servers, caching servers, gateway systems, and firewall systems. This section offers primary tuning recommendations for some of the attributes for the following subsystems:

- Internet (Section 2.2.1)

- Process (Section 2.2.2)
- Socket (Section 2.2.3)
- Virtual Memory (Section 2.2.4)

Note

Some kernel subsystem attributes enable you to modify their value and apply the value to a running system. Other attributes require you to reboot the system to use a new value. Refer to Section 2.1.3.1 to determine if an attribute can be tuned at run time.

The primary tuning recommendations provide the best performance improvement for most Internet server configurations. If performance is still deficient after applying these recommendations, you may be able to improve performance by modifying additional kernel subsystem attributes, as discussed in Section 2.3.

You can also use the Compaq Continuous Profiling Infrastructure (CPI, formerly known as DCPI) tool to obtain detailed information about system components that heavily utilize CPU cycles. CPI is offered as an Advanced Developement Kit. Refer to the following location for more information:

`http://www.tru64unix.compaq.com/dcpi`

2.2.1 Modifying Internet Attributes

You may be able to improve Internet server performance by tuning the following Internet (`inet`) subsystem attributes:

- `tcbhashsize` (Section 2.2.1.1)
- `pmtu_enabled` (Section 2.2.1.2)
- `ipport_userreserved` (Section 2.2.1.3)

2.2.1.1 Increasing the Size of the TCP Hash Table

The **`tcbhashsize`** attribute specifies the number of buckets in the Transmission Control Protocol (TCP) `inpcb` hash table. The kernel must look up the connection block for every TCP packet it receives; therefore, increasing the size of the table can speed up the search and improve performance.

However, increasing the size of the hash table will cause a slight increase in wired memory. It can also cause a bottleneck at the TCP hash table in SMP systems.

The default value is 512 buckets (32 on systems running a version of Tru64 UNIX prior to Version 4.0E). The *recommended value* is 16384.

2.2.1.2 Disabling PMTU Discovery

Packets transmitted between servers are divided into equal-sized units to facilitate the transmission of the data over routers and small-packet networks, such as Ethernet networks.

When the **pmtu_enabled** attribute is enabled, the operating system determines the largest common path maximum transmission unit (PMTU) value between servers and uses it as the unit size. A routing table entry is also created for each client network that attempts to connect to the server.

If you have a poorly performing Internet server that handles mainly remote traffic and the routing table increases to more than 1000 entries, disabling the PMTU discovery can decrease the size of the routing table, which may improve server efficiency. However, if a server handles mainly local traffic and only some remote traffic, disabling PMTU discovery can degrade bandwidth. Use the `netstat -r` command to display the contents of the routing table.

The default value is 1 (PMTU enabled). The *recommended value* is 0 (PMTU disabled).

2.2.1.3 Increasing the Number of Outgoing Connection Ports

When a TCP or UDP application creates an outgoing connection, the kernel dynamically allocates a nonreserved port number for each connection.

The kernel selects the port number from a range of values between `ipport_userreserved_min` (if you are running Tru64 UNIX Version 4.0E or higher) or 1024 (if you are running Version 4.0D or earlier), and `ipport_userreserved`.

Using the default attribute values, the range of outgoing ports begins at port 1024 and ends at port 5000, and the number of simultaneous outgoing connections is limited to 3976 (5000 minus 1024).

If you have a proxy server, caching server, gateway system, or firewall system, with a load of more than 4000 simultaneous connections, you can modify the value of the **ipport_userreserved** attribute. The default value is 5000, which is the minimum value. The *recommended value* is 65535, which is the maximum value. Do not specify a value that is greater than 65535 or lower than 5000.

2.2.2 Modifying Process Attributes

You may be able to improve Internet server performance by tuning the following process (`proc`) subsystem attributes:

- `maxusers` (Section 2.2.2.1)
- `max_proc_per_user` (Section 2.2.2.2)
- `max_threads_per_user` (Section 2.2.2.3)
- `max_per_proc_data_size` (Section 2.2.2.4)
- `max_per_proc_address_space` (Section 2.2.2.5)

These attributes set limits on system resources. If your Internet server appears to be reaching the resource limits, you may want to increase the value of one or more of these attributes. However, increasing the value of these attributes will allow the system to consume more memory.

2.2.2.1 Increasing the Size of System Tables and Data Structures

System algorithms use the **`maxusers`** attribute to size various system data structures and system tables. Increasing the value of `maxusers` provides more system resources to processes. However, this will increase the amount of wired memory.

If your system experiences a lack of resources (for example, `Out of processes`, `No more processes`, or `pid table is full` messages) and you have sufficient memory, increase the value of the `maxusers` attribute.

To determine an appropriate value for the `maxusers` attribute, you can double the default value until you improve performance. For example, if you have up to 1 GB of memory, increase the value of the `maxusers` attribute to 512. If you have up to 2 GB, increase the value to 1024. If you have an Internet, Web, proxy, caching, firewall, or gateway server, increase the value of the `maxusers` attribute to 2048.

The default value varies from 16 to 2048, depending on the amount of physical memory in the system. It is not recommended that you increase the value to more than 2048.

System administrators can change the `maxusers` attribute with the following command:

```
# sysconfig -r proc maxusers=N
```

The value *N* is the desired new value. For Tru64 UNIX version 5.0A or later, this command triggers the automatic expansion of the `pid table`. The resizing of other system tables is not performed until you specify a new value for the `maxusers` attribute in the `/etc/sysconfigtab` file and reboot the system.

2.2.2.2 Increasing the Number of Processes per User

The **max_proc_per_user** attribute specifies the maximum number of processes that can be allocated at any one time to each user, except superuser.

If your system experiences a lack of processes, increase the value of this attribute. If you have a multiprocess Internet server (for example, a server running IPlanet, Apache, CERN, or Zeus), you also may want to increase the value of this attribute.

The default value is 64. The *recommended value* is 2000. The value you choose must not be more than the maximum number of processes that can be started by your system. For Internet servers, these processes include CGI processes. If you specify a value of 0 (zero) for this attribute, there is no limit on the number of processes per user.

2.2.2.3 Increasing the Number of Threads per User

The **max_threads_per_user** attribute specifies the maximum number of threads that can be allocated at any one time to each user, except superuser.

If your system experiences a lack of threads, increase the value of this attribute. If you have a multithreaded Internet server (for example, a server running Netscape FastTrack or Netscape Enterprise), you also may want to increase the value this attribute.

The default value is 256. The *recommended value* is 4096. The value must not be more than the maximum number of threads that can be started by your system.

2.2.2.4 Increasing the User Process Data Segment Size Limits

The **max_per_proc_data_size** attribute specify the maximum limit of data segment sizes. Some large programs and large-memory processes may not run unless you increase the values of this attribute. Increase the limits if you receive an `Out of process memory` message.

The default value is 1073741824 (1 GB). The *recommended value* is 10737418240 (10 GB). If your system has more than 10 GB of memory, you can further increase this value.

2.2.2.5 Increasing the User Process Address Space Limits

The **max_per_proc_address_space** attribute specifies the maximum limit of user process address space (number of bytes of virtual memory). Some large programs and large-memory processes may not run unless you increase the value of this attribute. However, increasing the address space limits will cause a small increase in memory consumption.

The default value is 1073741824 (1 GB) for systems running Tru64 UNIX Version 4.0G and earlier, and 4294967296 (4 GB) for systems running Tru64 UNIX Version 5.0 or higher.

The *recommended value* is 10737418240 (10 GB). If your system has more than 10 GB of memory, you can further increase this value.

2.2.3 Modifying Socket Attributes

You may be able to improve Internet server performance by tuning the following socket attributes:

- `somaxconn` (Section 2.2.3.1)
- `sominconn` (Section 2.2.3.2)
- `sbcompress_threshold` (Section 2.2.3.3)

2.2.3.1 Increasing the Maximum Number of Pending TCP Connections

The **`somaxconn`** attribute specifies the maximum number of pending TCP connections (the socket listen queue limit) for each server socket (for example, for the HTTP server socket). Pending TCP connections can be caused by lost packets in the Internet or denial of service attacks. Busy Internet servers often experience large numbers of pending connections. If the listen queue connection limit is too small, incoming connect requests may be dropped.

The default value is 1024. The *recommended value* is 65535, which is the maximum value. Do not specify a value that is higher than the maximum value because this can cause unpredictable behavior.

2.2.3.2 Increasing the Minimum Number of Pending TCP Connections

The **`sominconn`** attribute specifies the minimum number of pending TCP connections (`backlog`) for each server socket. The attribute controls the maximum number of SYN packets that the system can handle simultaneously before additional requests are discarded. Network performance can degrade if a client saturates a socket listen queue with erroneous TCP SYN packets, which blocks other users from the queue.

The value of the `sominconn` attribute overrides the application-specific `backlog` value, which may be set too low for some server software. If you do not have your application source code, use the `sominconn` attribute to set a pending-connection limit that is appropriate for your application.

The default value is 0. The *recommended value* is 65535, which is the maximum value. It is recommended that the value of the `sominconn`

attribute be the same as the value of the somaxconn attribute. See Section 2.2.3.1 for more information in the somaxconn attribute.

2.2.3.3 Enabling the mbuf Cluster Compression

The **sbcompress_threshold** attribute controls whether mbuf clusters are compressed at the socket layer. By default, mbuf clusters are not compressed, which can cause proxy servers and caching servers to consume all the available mbuf clusters. This problem is more likely to occur if you are using FDDI, instead of Ethernet. Refer to Section 1.4.2 for information about monitoring mbuf clustering.

To enable mbuf cluster compression, modify the sbcompress_threshold attribute and specify a value. Packets will be copied into the existing mbuf clusters if the packet size is less than this value.

The default value is 0 (mbuf compression is disabled). If you have a proxy server, caching server, gateway system, or firewall system, the *recommended value* is 600 bytes.

2.2.4 Modifying Virtual Memory Attributes

For systems running Tru64 UNIX Version 4.0G and earlier, you may be able to improve Internet server performance by tuning the following Virtual Memory (vm) subsystem attributes:

- vm-mapentries (Section 2.2.4.1)
- vm-maxvas (Section 2.2.4.2)

2.2.4.1 Increasing the Maximum Number of Memory-Mapped Files

The **vm-mapentries** attribute specifies the maximum number of memory-mapped files for a user process for Tru64 UNIX Version 4.0G and earlier.

The operating system limits the number of memory-mapped files that are available to each process. Each map entry describes one unique disjoint portion of a virtual address space.

The two primary types of Internet servers are multiprocess and multithreaded Internet servers. Because multithreaded Internet servers are more likely to use memory-mapped files, you may want to increase the maximum number of memory-mapped files if you have this type of system.

In addition, after several hours of use, Netscape Enterprise Server users may receive forbidden messages in response to Web page requests. The errors file may also contain a *URL* could not load message, where URL

specifies the location of the requested page. If this occurs, the server may have the value of the `vm-mapentries` attribute set too low.

The default value is 200 map entries. If you have a busy very-large memory (VLM) system running multithreaded Internet server software, the *recommended value* is 20000. The `vm-mapentries` attribute affects all processes, so increasing its value will allow the system to consume more memory.

2.2.4.2 Increasing the Maximum Amount of Valid Virtual Address Space

The **`vm-maxvas`** attribute specifies the maximum amount of valid virtual address space for a process (that is, the sum of all the valid pages) for Tru64 UNIX Version 4.0G and earlier.

The default value is 1073741824 bytes (1GB). If you have an Internet, Web, proxy, caching, firewall, or gateway server, the *recommended value* is 10737418240 (10 GB).

2.3 Advanced Tuning Recommendations

This section offers advanced tuning recommendations for some of the attributes for the following subsystems:

- Generic (Section 2.3.1)
- Internet (Section 2.3.2)
- Network (Section 2.3.3)
- Socket (Section 2.3.4)
- Virtual memory (Section 2.3.5)

These recommendations are appropriate only for systems that are primarily used as Internet servers and are configured with sufficient physical memory. Using a recommended attribute value in a non-Internet server may cause a degradation in system performance.

Because Internet server configurations differ and a recommended value may not provide optimal performance for all configurations, be careful when modifying attributes. Read the attribute descriptions and determine which values are appropriate for your configuration. If modifying an attribute does not improve performance, you may want to return to the default value.

2.3.1 Modifying Generic Attributes

You may be able to improve Internet server performance by tuning the **`kmemreserve_percent`** generic (`generic`) subsystem attribute. This attribute increases the percentage of physical memory reserved for

kernel memory allocations that are less than or equal to the page size (8 KB). Increasing the value of `kmemreserve_percent` improves network throughput by reducing the number of packets that are dropped while the system is under a heavy network load. However, increasing this value consumes memory.

You may want to increase the value of the `kmemreserve_percent` attribute if the output of the `netstat` command shows dropped packets, or if the output of the `vmstat -M` command shows dropped packets under the `fail_nowait` heading. This may occur under a heavy network load.

The default value is 0 (the percentage of reserved physical memory will be the smallest of 0.4 percent of available memory and 256 KB). Increase the value (up to a maximum of 75) by small increments until the output of the `vmstat -M` command shows no entries under the `fail_nowait` heading.

2.3.2 Modifying Internet Attributes

You may be able to improve Internet server performance by tuning the following internet (`inet`) subsystem attributes:

- `tcbhashnum` (Section 2.3.2.1)
- `inifaddr_hsize` (Section 2.3.2.2)
- `tcp_keepinit` (Section 2.3.2.3)
- `tcp_rexmit_interval_min` (Section 2.3.2.4)
- `tcp_keepalive_default` (Section 2.3.2.5)
- `tcp_msl` (Section 2.3.2.6)
- `ipport_userreserved_min` (Section 2.3.2.7)
- `ipqs` (Section 2.3.2.8)
- `ipqmaxlen` (Section 2.3.2.9)

2.3.2.1 Increasing the Number of TCP Hash Table

The **`tcbhashnum`** attribute specifies the number of TCP hash tables. Increasing the number of hash tables distributes the load and may improve performance. However, this will slightly increase the amount of wired memory in the system.

The default value is 1 hash table, which is the minimum value. For busy Internet server SMP systems, the *recommended value* is 16. The maximum value is 64.

If you increase the number of hash tables, decrease the size of the hash table. Refer to Section 2.2.1.1 for more information. In addition, it is recommended that you make the value of this attribute the same as the

value of the ipqs attribute. Refer to Section 2.3.2.8 for more information on the ipqs attribute.

2.3.2.2 Increasing the Number of Hash Buckets

The **inifaddr_hsize** attribute specifies the number of hash buckets in the kernel interface alias table (in_ifaddr).

If a system is used to serve many different server domain names, each of which are bound to a unique IP address, the code that matches arriving packets to the right server address uses the hash table to speed lookup operations for the IP addresses. These addresses are usually set using the ifconfig alias or ifconfig aliaslist command. Increasing the number of hash buckets in the table can improve performance on systems that use large numbers of IP alias addresses.

The default value is 32 hash buckets. For most Internet servers that do not use interface IP aliases or if you are using less than 250 aliases, the *recommended value* is 32. If you are using more than 500 interface IP aliases, the *recommended value* is 512, which is the maximum value.

For the best performance, the value of this attribute must be rounded down to the nearest power of 2.

2.3.2.3 Modifying the TCP Partial Connection Timeout Limit

The **tcp_keepinit** attribute specifies the amount of time that a partially established TCP connection remains on the socket listen queue before it times out. The value of the attribute is in units of 0.5 seconds. Partial connections consume socket listen queue slots and fill the queue with connections in the SYN_RCVD state.

The default value is 150 units (75 seconds). You do not need to modify the TCP partial-connection timeout limit unless the value of the somaxconn_drops attribute often increases. Refer to Section 1.5 for more information on the event counter.

If your socket queue limit is set to the maximum value, the default value of this attribute is usually adequate. If the somaxconn_drops attribute often increases, and increasing the socket queue limit does not prevent the listen queue from filling up, you can decrease the value of this attribute to make partial connections to time out sooner.

In addition, network performance can degrade if a client overfills a socket listen queue with TCP SYN packets, which blocks other users from the queue. To eliminate this problem, increase the socket listen queue limit to its maximum value. If the system continues to drop SYN packets, decrease the value of this attribute to 30 (15 seconds). Monitor the values of the

`sobacklog_drops` and `somaxconn_drops` event counters to determine if the system is dropping packets.

Do not set the value of this attribute too low, because you may prematurely break connections with clients on slow network paths or network paths that lose many packets. Do not set the value to less than 20 units (10 seconds).

2.3.2.4 Decreasing the Rate of TCP Retransmissions

The **`tcp_rexmit_interval_min`** attribute specifies the minimum amount of time between the first TCP retransmission. For some wide area networks (WANs), the default value may be too small and premature retransmission timeouts may occur, which cause duplicate transmission of packets and the erroneous invocation of the TCP congestion-avoidance algorithms.

You can increase the value of this attribute to slow the rate of TCP retransmissions, which decreases congestion and improves performance.

The default value is 2 units (1 second). Not every connection needs a long retransmission time. Usually, the default value of this attribute is adequate. However, for some WANs, the default retransmission interval may be too small.

To check for retransmissions, use the `netstat -p tcp` command and examine the output for data packets retransmitted.

You can increase the value of this attribute to slow the rate of TCP retransmissions. The attribute is specified in units of 0.5 seconds.

Do not change the default value of this attribute unless you fully understand TCP algorithms. Do not specify a value that is less than 1 unit.

2.3.2.5 Enabling TCP Keepalive Functionality

Keepalive functionality enables the periodic transmission of messages on a connected socket to keep connections active and to time out inactive connections. Sockets that do not exit cleanly are cleaned up when the keepalive interval expires. If keepalive is not enabled, those sockets continue to exist until you reboot the system.

Applications enable keepalive for sockets by setting the `setsockopt` function's `SO_KEEPALIVE` option. The default value is 0 (keepalive is disabled). To enable keepalive for programs that do not set keepalive on their own, or if you do not have access to the application source code, set this attribute to 1. After you set the attribute, all new connections will have keepalive enabled; existing connections will continue to use the previous keepalive setting.

If you modify this attribute without rebooting the system, sockets that already exist will continue to use the old behavior until the applications are restarted.

If you enable keepalive, you can also configure the following TCP options for each socket:

- The `tcp_keepidle` attribute specifies the amount of idle time, in 0.5-second units, before sending a keepalive probe. The default value for this attribute is 2 hours.
- The `tcp_keepintvl` attribute specifies the amount of time, in 0.5-second units, between retransmission of keepalive probes. The default value for this attribute is 75 seconds.
- The `tcp_keepcnt` attribute specifies the maximum number of keepalive probes that are sent before the connection is dropped. The default value for this attribute is 8 probes.
- The `tcp_keepinit` attribute specifies the maximum amount of time, in 0.5-second units, before an initial connection attempt times out. The default value for this attribute is 75 seconds.

2.3.2.6 Increasing the TCP Connection Context Timeout Rate

The **`tcp_msl`** attribute determines the maximum lifetime of a TCP segment and the timeout value for the `TIME_WAIT` state. The TCP protocol includes a concept known as the Maximum Segment Lifetime (MSL). When a TCP connection enters the `TIME_WAIT` state, it must remain in this state for twice the value of the MSL, or else undetected data errors on future connections can occur.

You can decrease the value of this attribute to make the TCP connection context time out more quickly at the end of a connection. However, this will increase the chance of data corruption.

The default value is 60 units (30 seconds, which means that the TCP connection remains in `TIME_WAIT` state for 60 seconds or twice the value of the MSL). The value of this attribute is set in units of 0.5 seconds. The *recommended value* is the default value; if you use a different value, there is the potential for data corruption.

Although the TCP specifications specify an MSL of 120 seconds, most TCP implementations use a value that is less than 120. The *Internet FAQ Consortium* Web site offers more information. For RFC793, refer to the following URL:

`http://www.faqs.org/rfcs/rfc793.html`

For RFC1122, refer to the following URL:

`http://www.faqs.org/rfcs/rfc1172.html`

In some situations, the default timeout value for the TIME_WAIT state is too large, so reducing the value of this attribute frees connection resources sooner than the default behavior.

Do not reduce the value of this attribute unless you fully understand the design and behavior of your network and the TCP protocol.

2.3.2.7 Modifying the Range for Outgoing Connection Ports

When a TCP or UDP application creates an outgoing connection, the kernel dynamically allocates a nonreserved port number for each connection.

The kernel selects the port number from a range of values between **ipport_userreserved_min** (if you are running Tru64 UNIX Version 4.0E or higher) or 1024 (if you are running a prior version), and ipport_userreserved.

If you are running Tru64 UNIX Version 4.0E or a higher version of the operating system, and your system requires a particular range of ports, you can modify the value of this attribute.

The default value is 1024. The maximum value is 65535. Do not specify a value for this attribute that is greater than 65535 or less than 1024.

2.3.2.8 Increasing the Number of IP Input Queues

For SMP systems, increasing the number of IP input queues can reduce lock contention at the input queue and distribute the load. The **ipqs** attribute specifies the number of IP input queues.

The default value is 1 queue, which is the minimum value. For busy Internet server SMP systems, the *recommended value* is 16. The maximum value is 64.

It is recommended that you make the value of this attribute the same as the value of the tcbhashnum attribute. Refer to Section 2.2.1.1 for more information on the tcbhashnum attribute.

2.3.2.9 Increasing the Maximum Lenght of the IP Input Queue

If the network load is heavy, input packets may be dropped if the IP input queue becomes filled. The **ipqmaxlen** attribute specifies the maximum length (in bytes) of the IP input queue (ipintrq) before input packets are dropped.

If your system drops input packets, you may want to increase the value of the ipqmaxlen attribute. Check for dropped input packets by using dbx to examine the ipintrq kernel structure. For example:

```
# dbx -k /vmunix
(dbx) print ipintrq
struct {
     ifq_head = (nil)
     ifq_tail = (nil)
     ifq_len = 0
     ifq_maxlen = 512
     ifq_drops = 128
     ifq_slock = struct {
         sl_data = 0
         sl_info = 0
         sl_cpuid = 0
         sl_lifms = 0
      }
}
```

If the `ifq_drops` field is not zero, the system is dropping IP input packets.

The default value is 1024 (512 on systems running a version of Tru64 UNIX prior to Version 5.0). The minimum value is the default value; the maximum value is 65535. If your system is dropping input packets, the *recommended value* is 2048. You may also want to increase the value of the `ifqmaxlen` attribute, which controls the output queue. Refer to Section 2.3.3.1 for more information on the `ifqmaxlen` attribute.

2.3.3 Modifying Network Attributes

You may be able to improve Internet server performance by tuning the following Network (`net`) subsystem attributes:

- `ifqmaxlen` (Section 2.3.3.1)
- `screen_cachedepth` (Section 2.3.3.2)
- `screen_cachewidth` (Section 2.3.3.2)
- `screen_maxpend` (Section 2.3.3.3)

2.3.3.1 Increasing the Number of Output Packets Before Packets are Dropped

If the network load is heavy, output packets may be dropped if the interface's output queue becomes filled. The **`ifqmaxlen`** attribute specifies the number of output packets that can be queued to a network adapter before packets are dropped.

You can use the `netstat -id` command to check for dropped output packets. If the command output shows a nonzero value in the `Drop` column for an interface, the system is dropping output packets and you may want to increase the value of this attribute.

The default value is 1024 (512 on systems running a version of Tru64 UNIX prior to Version 5.0). The minimum value is the default value; the maximum value is 65535. If your system is dropping input packets, the *recommended value* is 2048.

2.3.3.2 Reducing Screening Cache Misses

If your machine is acting as a screening router, or a screening firewall running the `screend` facility, and has a high number of concurrent pass-through connections, you could be experiencing screening cache misses.

A screening cache miss can occur when the kernel screening table is trying to screen a packet that does not have an entry, based on address/port pairs and protocol. In that case, the table must queue the packet and the `screend` daemon must examine it. This can normally occur for the first packet of a connection, and can also occur if the cache is too small to hold many entries.

Check for screening cache misses by using `dbx` to examine the number of screening cache hits and misses. For example:

```
(dbx) p screen_cachemiss
616738
(dbx) p screen_cachehits
11080198
```

If the ratio of misses to hits is high, you may want to increase the values of the `screen_cachedepth` and `screen_cachewidth` attributes.

The default value for the **`screen_cachedepth`** attribute is 8, which is the minimum value. If you have high screening cache miss rates, the *recommended value* is 16, which is the maximum value.

The default value for the **`screen_cachewidth`** attribute is 8, which is the minimum value. If you have high screening cache miss rates, the *recommended value* is 2048, which is the maximum value.

It is recommended that you first increase `screen_cachewidth` before increasing `screen_cachedepth`. Also note that tuning these attributes will not necessarily reduce screening cache misses to zero. A reboot is required for the changes to take effect.

Increasing these values will cause a small increase in memory consumption.

2.3.3.3 Reducing the Screening Buffer Drops

If your machine is acting as a screening router, or a screening firewall running the `screend` facility, and is under heavy network load, you may be experiencing screening buffer drops.

You can use the `screenstat` command to view the current status:

```
# /usr/sbin/screenstat
total packets screened: 11696910
total accepted: 11470734
total rejected: 225453
packets dropped:
        because buffer was full:        34723
        because user was out of sync:   0
        because too old:        0
```

```
total dropped:  34723
```

If the number of packets dropped `because buffer was full` is high, you may want to increase the value of the **`screen_maxpend`** attribute. The default value is 32, which is the minimum value. If you have a high screening buffer full value, the *recommended value* is 8192. The maximum value is 16384.

Increasing this value will cause a small increase in memory consumption. You must reboot the system to modify this attribute.

2.3.4 Modifying Socket Attributes

You may be able to improve Internet server performance by tuning the **`sb_max`** socket (`socket`) subsystem attribute. In addition, the `socket` subsystem attributes `sobacklog_hiwat`, `sobacklog_drops`, and `somaxconn_drops` track events related to socket listen queues. By monitoring these attributes, you can determine if the queues are overflowing. Section 1.5 discusses these attributes.

The **`sb_max`** attribute specifies the maximum size of a socket buffer. Increasing the maximum size of a socket buffer may improve performance if your applications can benefit from a large buffer size.

The default value is 1048576 bytes (131072 bytes on a system running a version of Tru64 UNIX prior to Version 4.0E). If your applications require a socket buffer that is larger than the default value, increase the value of this attribute.

2.3.5 Modifying Virtual Memory Attributes

You may be able to improve Internet server performance by tuning the following virtual memory (`vm`) subsystem attributes:

- For Tru64 UNIX Version 5.1A and earlier
  - `ubc_maxpercent`, `ubc_minpercent`, and `ubc_borrowpercent` (Section 2.3.5.1)
- For Tru64 UNIX Version 4.0G and earlier
  - `vm-vpagemax` (Section 2.3.5.2)

2.3.5.1 Increasing the Memory Available to Processes

Busy Internet servers usually consume a moderate amount of virtual memory and use a large set of files. Both processes and the Unified Buffer Cache (UBC), which caches file system data, share the physical memory that is not wired by the kernel.

Too much memory allocated to the UBC can cause excessive paging and swapping, which may degrade overall system performance. However, an insufficient amount of memory allocated to the UBC can degrade file system performance.

The **ubc_minpercent** attribute specifies the minimum percentage of memory that only the UBC can utilize. The remaining memory is shared with processes. The **ubc_maxpercent** attribute specifies the maximum percentage of memory that the UBC can utilize. The **ubc_borrowpercent** attribute specifies the UBC borrowing threshold.

Between the value of the ubc_borrowpercent attribute and the value of the ubc_maxpercent attribute, the memory that is allocated to the UBC is considered borrowed from processes. When paging begins, these borrowed pages are reclaimed first, until the amount of memory allocated to the UBC decreases to the value of the ubc_borrowpercent attribute.

The default value for ubc_minpercent is 10 percent. The default value for ubc_maxpercent is 100 percent. The default value for ubc_borrowpercent is 20 percent. On a typical Internet server, the default value for each attribute is usually adequate. Also, if your disks are busy with file system I/O and the system has sufficient free pages, use the default values.

Use the vmstat command to display information about virtual memory, including the free page count.

If you have a low free page count, you may want to increase the memory available to processes by reducing the memory available to the UBC. You should attempt to keep in memory the working set of your processes, even if it increases the number of UBC misses.

You can reduce the default value of the ubc_maxpercent attribute in decrements of 10 percent.

Reducing the borrowed memory threshold by decreasing the value of the ubc_borrowpercent attribute may improve the system response time when memory is low. However, this may also reduce UBC performance.

2.3.5.2 Increasing the Maximum Number of Protected Virtual Pages

The **vm-vpagemax** attribute specifies the maximum number of virtual pages within a process' address space that can be given individual protection attributes for Tru64 UNIX Version 4.0G and earlier. These protection attributes differ from the protection attributes associated with the other pages in the address space.

Changing the protection attributes of a single page within a virtual memory region causes all pages within that region to be treated as though

they had individual protection attributes. For example, each thread of a multithreaded task has a user stack in the stack region for the process in which it runs. Because multithreaded tasks have guard pages (that is, pages that do not have read/write access) inserted between the user stacks for the threads, all pages in the stack region for the process are treated as though they have individual protection attributes.

The two primary types of Internet servers are multiprocess and multithreaded Internet servers. Because multithreaded Internet servers are more likely to use memory-mapped files, you may want to modify the value of this attribute if you have this type of system. However, this attribute affects all processes, so increasing its value will allow the system to consume more memory.

The default value is the value of the `vm-maxvas` attribute (the size of valid virtual address space in bytes) divided by 8192. If a stack region for a multithreaded task exceeds 16 KB pages, you may want to increase the value of `vm-vpagemax`.

For example, if the value of the `vm-maxvas` attribute is 1 GB (the default), set the value of this attribute to 131072 pages (1073741824 / 8192 = 131072). Using this value can improve the efficiency of Internet servers that maintain large tables or resident images.

A

Kernel Subsystem Attributes

This appendix lists the Tru64 UNIX support for the attributes discussed in this document. It provides attribute name equivalence between Tru64 UNIX Version 5.0 or higher, and earlier versions. You can also view which attributes are tuneable at run time and which ones require you to reboot your system.

A.1 Operating System Support for Attributes

Table A–1 provides the following information about which versions of the operating system support the attributes described in this document:

- An * (asterisk) specifies that the version supports the attribute and does not require any patch.
- A - (dash) specifies that the version does not support the attribute.
- If you must display or modify an attribute by using a method other than `sysconfig`, `sysconfigdb`, or `dxkerneltuner`, this information is specified in the table entry.

Table A–1: Operating System Support for Attributes

Attribute	Version 4.0D, 4.0E, 4.0F, or 4.0G	Version 5.0 or higher
ifqmaxlen	Must use dbx	*
inifaddr_hsize	*	*
ipport_userreserved	*	*
ipport_userreserved_min	Not supported in Version 4.0D	*
ipqmaxlen	Must use dbx for Version 4.0D	*
ipqs	Not supported in Version 4.0D	*
kmemreserved_percent	*	*
max_per_proc_address_space	*	*
max_per_proc_data_size	*	*
max_proc_per_user	*	*
max_threads_per_user	*	*
maxusers	*	*
pmtu_enabled	*	*

Table A–1: Operating System Support for Attributes (cont.)

Attribute	Version 4.0D, 4.0E, 4.0F, or 4.0G	Version 5.0 or higher
sbcompress_threshold	Must use dbx for Version 4.0D	*
sb_max	*	*
screen_cachedepth	Must use dbx for Version 4.0D	*
screen_cachewidth	Must use dbx for Version 4.0D	*
screen_maxpend	Must use dbx for Version 4.0D	*
sobacklog_drops	*	*
sobacklog_hiwat	*	*
somaxconn	*	*
somaxconn_drops	*	*
sominconn	*	*
tcbhashnum	Not supported in Version 4.0D	*
tcbhashsize	*	*
tcp_keepalive_default	*	*
tcp_keepinit	*	*
tcp_msl	*	*
tcp_rexmit_interval_min	*	*
ubc_maxpercent	*	*
ubc_minpercent	*	*
vm-mapentries	*	–
vm-vpagemax	*	–
vm-maxvas	*	–

A.2 Attribute Name and Tuning Comparisions

Table A–2 lists the attribute name equivalence for Tru64 UNIX Version 5.0 and higher, and Tru64 UNIX Version 4.0G and earlier. Although Tru64 UNIX Version 5.0 and higher offer backward compatibility, we recommend that you use the new names for that version of the operating system. It also lists whether the attributes are tuneable at run time. If Yes is specified in the table, you can change the attribute value at run time; if No is specified, you need to reboot your system to change the attribute value.

Table A–2: Attribute Name and Tuning Comparisons

Tru64 UNIX Version 4.0G and earlier	Tru64 UNIX Version 5.0 and later	Tune at Run Time?
ifqmaxlen	ifqmaxlen	Yes for Tru64 UNIX Version 5.0 and later. No for Tru64 UNIX Version 4.0G and earlier.
inifaddr_hsize	inifaddr_hsize	Yes for Tru64 UNIX Version 5.0 and later. No for Tru64 UNIX Version 4.0G and earlier.
ipport_userreserved	ipport_userreserved	Yes
ipport_userreserved_min	ipport_userreserved_min	Yes
ipqmaxlen	ipqmaxlen	Yes for Tru64 UNIX Version 5.0 and later. No for Tru64 UNIX Version 4.0G and earlier.
ipqs	ipqs	No
kmemreserved-percent	kmemreserved_percent	Yes
max-per-proc-address-space	max_per_proc_address_space	No
max-per-proc-data-size	max_per_proc_data_size	No
max-proc-per-user	max_proc_per_user	No
max-threads-per-user	max_threads_per_user	No
maxusers	maxusers	Yes for Tru64 UNIX Version 5.0 and later. No for Tru64 UNIX Version 4.0G and earlier.
pmtu_enabled	pmtu_enabled	Yes
sbcompress_threshold	sbcompress_threshold	Yes
sb_max	sb_max	Yes
screen_cachedepth	screen_cachedepth	No
screen_cachewidth	screen_cachewidth	No
screen_maxpend	screen_maxpend	No
sobacklog_drops	sobacklog_drops	Yes
sobacklog_hiwat	sobacklog_hiwat	Yes
somaxconn	somaxconn	Yes
somaxconn_drops	somaxconn_drops	Yes
sominconn	sominconn	Yes

Table A–2: Attribute Name and Tuning Comparisons (cont.)

Tru64 UNIX Version 4.0G and earlier	Tru64 UNIX Version 5.0 and later	Tune at Run Time?
tcbhashnum	tcbhashnum	No
tcbhashsize	tcbhashsize	Yes
tcp_keepalive_default	tcp_keepalive_default	Yes
tcp_keepinit	tcp_keepinit	Yes
tcp_msl	tcp_msl	Yes
tcp_rexmit_interval_min	tcp_rexmit_interval_min	Yes
ubc-borrowpercent	ubc_borrowpercent	Yes for Tru64 UNIX Version 5.0 and later. No for Tru64 UNIX Version 4.0G and earlier.
ubc-maxpercent	ubc_maxpercent	Yes for Tru64 UNIX Version 5.0 and later. No for Tru64 UNIX Version 4.0G and earlier.
ubc-minpercent	ubc_minpercent	Yes for Tru64 UNIX Version 5.0 and later. No for Tru64 UNIX Version 4.0G and earlier.

B

Revision History

Periodically, this document is updated as new information becomes available.

This document's revision history is as follows:

Version 3.3 - January, 2003
Version 3.2 - January, 2002
Version 3.1 - September, 2000
Version 3.0 - September, 1999
Version 2.3 - December, 1998
Version 2.2 - July, 1998
Version 2.1 - May, 1998
Version 2.0 - February, 1998
Version 1.2.2 - October, 1997
Version 1.2.1 - July, 1997
Version 1.2 - April, 1997
Version 1.1.1 - February, 1997
Version 1.1 - November, 1996
Version 1.0 - October, 1996