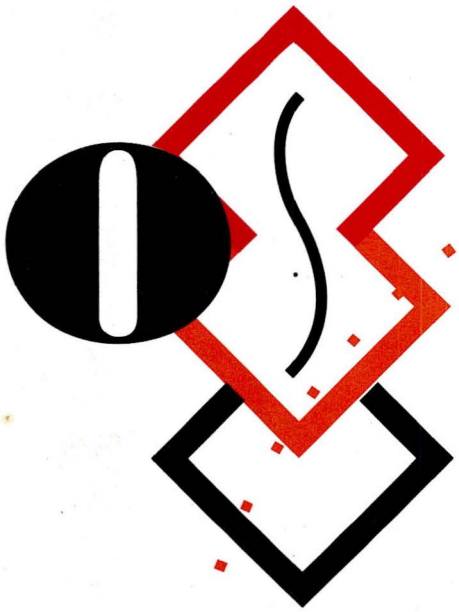


CONVEX

ConvexOS V11.5: Technical Overview and Managing ConvexOS Addendum

Second Edition

HEWLETT® PACKARD

Hewlett-Packard
CONVEX Technology Center
3000 Waterview Parkway
P.O. Box 833851
Richardson, TX 75083-3851
United States of America
(214)497-4000

ConvexOS V11.5: Technical Overview and Managing ConvexOS Addendum

710-001430-215

Second Edition
September 1996

CONVEX Press
Richardson, Texas
United States of America

HEWLETT PACKARD
Convex Technology Center

ConvexOS V11.5
Technical Overview and Managing ConvexOS Addendum

Document No. 710-001430-215

Notice

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

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


This entire book is recyclable.

Printed in the United States of America

Revision information for ConvexOS V11.5 Technical Overview and Managing ConvexOS Addendum

Edition	Document No.	Description
Second	710-001430-215	Released with ConvexOS V11.5.1, Sept. 1996.
First	710-001430-214	Released with ConvexOS V11.5, July 1996.

Contents

Using this guide

Managing ConvexOS: 11.5 Addendum describes changes to the C-Series operating system in the 11.5.1 release. The purpose of this document is to provide administrative details for operations that affect C-Series machine end-users.

Organization

This addendum works in conjunction with the existing ConvexOS documentation set:

- *Managing ConvexOS: Configuration Guide*
- *Managing ConvexOS: Operations Guide*
- *Managing Internet Services and NFS*

ConvexOS 11.5.1 differences are described in this addendum under corresponding sections from the appropriate ConvexOS guides.

Associated documents

Using the 11.5.1 ConvexOS software may require information fully described in one of the following documents:

- *ConvexOS dump and restore Quick Reference* (DSW-392), a quick reference for dumping and restoring file systems
- *Managing ConvexOS: Configuration Guide* (DSW-030), a guide for configuring ConvexOS
- *Managing ConvexOS: Operations Guide* (DSW-031), a guide for maintaining ConvexOS
- *Managing Internet Services and NFS* (DSW-108), a guide for configuration of network services
- *ConvexOS 11.5.1 Release Notice* (710-003430-066), a list of updates made in the 11.5.1 release of ConvexOS.

We recommend you be familiar with the contents of these documents to effectively use this addendum.

Technical assistance

Hardware, software, and documentation support can be obtained through the Hewlett-Packard Convex Technical Assistance Center (TAC):

- From locations in the continental United States:
  - Customers call (800)952-0379 or (214)497-4379
- From Canada, call 1(800)345-2384
- From all other locations, contact your local Hewlett-Packard Convex office.

Ordering documentation

To order the current edition of this or any other Hewlett-Packard Convex document, send requests to:

Hewlett-Packard Company
Convex Technology Center
Customer Service
P.O. Box 833851
Richardson TX 75083-3851 USA

Include the order number or the exact title, as listed on the front cover.

Technical Overview

1

Introduction

ConvexOS 11.5.1 is a new release of the C-Series architecture operating system in development at the Hewlett-Packard Convex Technology Center. This release is focused on eliminating existing OS inefficiencies and improving performance, in addition to incorporation of ConvexOS 11.0/11.1 TAC patches. ConvexOS 11.5.1 unifies the operating system as it works on all supported C-Series platforms[1].

The purpose of this section is to provide a technical overview of notable changes incorporated into the ConvexOS 11.5.1 release for end-users of C-Series supercomputers. This section does not constitute comprehensive coverage of all modifications. The following topics are discussed:

- Overview of benefits
- Performance improvements
- Enhancements
- Networking modifications
- Security
- Impact on C-Series end-users
- Patches

Benchmark data is not included as the intent is a high-level description of operating system and utility modifications and benefits for applicable situations. Performance statistics herein are based on limited lab testing. Performance may differ depending on specific system configurations.

[1]Does not include support for the C1 C-Series machines.

Overview of 11.5.1 Benefits

Notable changes in 11.5.1 are summarized below. Details are provided in sections to follow.

General

- Merge of ConvexOS 11.0, 11.1, 11.0 Realtime, and the Quad SCSI CCU (QSC).
- Integration of appropriate TAC patches.
- Increased range of ConvexOS user and group IDs from 32 K to 64 K.
- The Central Processing Complex (CPC) and Channel Control Unit (CCU) communication system (i.e., MBS) has been re-architected to improve system wide I/O performance. The new system eliminates processor contention, reduces latency to initiate I/O operations, and increases the overall pool of I/O subsystem packets.
- The amount of virtual memory set aside for kernel tables and buffers can now be up to 40 MB larger than in previous releases of ConvexOS.
- Improved raw disk I/O performance by implementing a lazy unwire algorithm for disk buffers.
- Various aggregate system-wide performance improvements for networking and other system calls. Networking protocol processing no longer requires exclusive kernel access.
- `libc` ZIC-based time library functions have been optimized for better performance when called repeatedly.
- Numerous additional statistics have been added to `pstat`.
- Newer version of `perl` (4.036 vs. 4.010).
- Newer version of `sendmail` (8.7.5 vs. 8.6).
- Newer version of GNU Emacs (19.28 vs 18.57).
- Standard support for several new tape devices.
- A kernel panic signature (characterization) has been added. This can eliminate the need for crash dump analysis in some cases.
- Miscellaneous CERT bug fixes and other utility bug fixes (`cron`, `syslog`, etc).
- New freeware utilities have been added, along with their source code.

File Systems

- Various file system performance improvements ranging from 3% to 15%.
- Increased virtual address space reserved for buffer cache from 512 MB to 1 GB. This can improve the buffer cache hit ratio.

Virtual Memory

- Initialization of processes with large virtual address spaces has been optimized, yielding a significant performance improvement at process creation time.
- Improved virtual memory paging performance via clustered paging for thread-private memory and certain types of shared memory.
- Frequently used kernel virtual memory functions have been optimized.
- The maximum addressability of a single process has been increased to allow access of up to 4 GB of physical memory via threads.

Networking

- Improved performance of the TCP checksum algorithm for small packets.
- The maximum NFS block size has been raised from 8 K to 60 K, which can produce a performance increase for high speed networks such as HiPPI.
- NFS client/server performance has been improved via a special parallel NFS protocol stack.
- Improved TCP performance over HiPPI and FDDI via enhanced TCP window scaling.
- Generic networking performance improvements; (reduced system time, improvements to protocol decode, and less system overhead via "smarter" packet processing algorithms).
- A newer version of `tcpdump` has been included which now supports HiPPI. In addition, security related to `tcpdump` has been improved.
- The resolver now supports search entries with multiple domains in `/etc/resolv.conf`.
- Raw socket support has been implemented.
- HiPPI errors, which formerly could hang a HiPPI interface, can now be recovered by resetting the interface with `ifconfig`.

C4600-Specific

- C4 kernel `vcopy` (vector copy) rate optimized to 500 MB/sec per processor. In addition, there are several non-aligned address copy cases which have about a 5X to 10X improvement in performance over ConvexOS 11.0/11.1, primarily observed as a reduction in system time to perform I/O.
- C4 kernel `bcopy` (scalar copy) rate optimized to 400 MB/sec per processor. This is about a 4X improvement over 11.0/11.1, primarily observed as a reduction in system time to perform I/O.
- The copy threshold where `vcopy` is used instead of `bcopy` has been moved from 8 K to 16 K due to improved `bcopy` performance.

Performance Improvements

A major objective of the ConvexOS 11.5 Development Team was to identify and implement performance improvements over ConvexOS 11.0/11.1. Performance benefits are described in this section.

General

This section describes overall system performance modifications.

I/O latency and throughput

The Message Based System has been improved for the following CCUs: HIPPI, VIOP, IDC, and QSC.

ConvexOS device drivers typically are constructed of two different components (a main ConvexOS kernel component and a CCU kernel component). These components use the Message Based System (MBS) to communicate I/O requests between the CPU and the CCU.

MBS is the transport mechanism for almost all communication between the CPU, SPU, and CCUs. MBS messages are 64 byte packets containing 16 bytes of header information and 48 bytes of data. In the original MBS design, a total of 703 MBS messages are available to support the entire system I/O requirements.

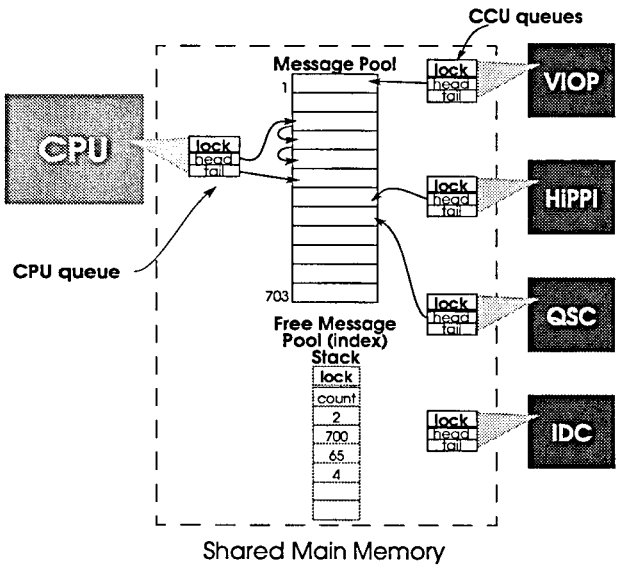
MBS messages are located in system Main Memory and never physically move. Instead, they are simply attached to one of many linked lists known as processor queues. Messages not in use are managed via their Message Pool Index on the free message stack (see Figure 1 on page 17). Each processor (CPU, CCU, SPU, etc.) has a dedicated MBS processor queue structure which contains a lock, plus head and tail pointers. This structure

is used to keep track of all messages currently enqueued for that processor.

The original MBS design has two fundamental limitations, capacity and latency. As for latency, when a processor receives an interrupt indicating one or more new messages on its queue, it must lock its queue to dequeue the new messages. This represents a possible lock contention since both the CPU and a CCU could be attempting to modify each other's processor queue at the same time. This problem is compounded by the fact that earlier CISC processor based CCUs, such as the VIOP, require much more time for queue manipulation than a Convex CPU. The effect is a very fast processor waiting for a slower one. Additionally, the rate at which messages can be placed onto the queue can be limited as system load increases.

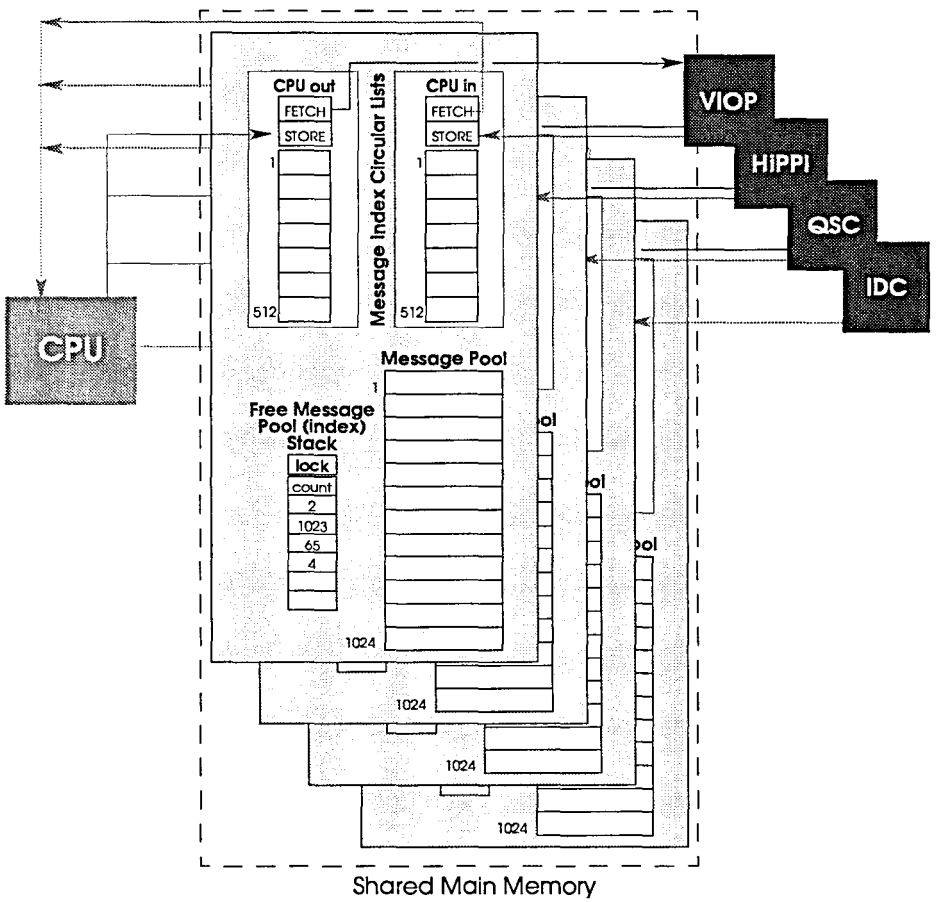
As for capacity, I/O to large disk stripes coupled with large amounts of network traffic can exhaust the pool of available messages. In most cases, this results in delayed (latent) I/O due to a wait for messages to become available. In extreme cases, a ConvexOS panic could occur.

Figure 1 MBS 11.0 System Design



The 11.5.1 MBS design addresses the above bottlenecks by creating a new MBS pool of messages for selected CPU/CCU interfaces. A pool of up to 1023 MBS messages are dedicated to each VIOP, IDC, HiPPI, or QSC CCU. Additionally, lock contention for processor queues has been eliminated due to the creation of a new type of processor queue. Each processor which can use the new method contains two distinct circular list queues, one for input and one for output. Each has a corresponding set of fetch and store queue indices. This design eliminates the requirement that queues be locked to either enqueue or dequeue messages. (See Figure 2 below).

Figure 2 MBS 11.5.1 System Design



The new MBS features are optionally activated at system startup. During a boot operation, the CPU receives I/O initialization commands from the SPU's `/ioconfig` file and sends them to the appropriate CCUs. When the CPU notices an I/O initialization command to a CCU of interest (VIOP, IDC, HiPPI, QSC), and that CCU's MBS tunable is set to a non-zero value, then the following special processing takes place:

- Memory is allocated on the CPU for an alternate MBS message pool. Alternate CPU and CCU input queue headers are created and initialized.
- A special command is sent to the CCU along with the necessary configuration data for allocating and mapping the CCU windows to the CPU memory.

Operating system tunables are available to selectively enable or disable the new MBS scheme on a per-CCU basis. These tunables disable the new MBS scheme when set to zero. This will cause the CPU to allocate MBS messages from the original MBS Pool for CCUs of that type. Alternatively, these tunables may specify the number of pages to allocate to an alternative MBS message pool (up to 16 pages or 1023 messages).

Another tunable is available that will globally disable the new MBS scheme for all CCUs. The system will then allocate all MBS messages from a single MBS pool of 703 messages, as before.

The original MBS and improved MBS schemes coexist, so every CCU has access to two MBS message pools. The CCU will receive a unique interrupt depending on which message pool has the message. All communication to devices attached to the CCU using the new system will be via the improved MBS scheme while SPU communication (e.g. `adbccu`) to the CCU itself will use the original MBS scheme.

Raw disk I/O

The operations for performing I/O on raw disk devices have been optimized to eliminate overhead on successive reads or writes. Such operations require that the memory they use be resident and "wired" (i.e. locked in core and non-swappable). Previous versions of ConvexOS unwired and re-wired the memory between successive reads and writes. Through the use of a lazy (i.e. delayed) unwiring algorithm, pages can now remain resident and wired. This eliminates the time required to re-wire them. The pages remain available for other purposes if they are needed, but are kept wired as long as possible. Note that this only affects raw devices, and therefore has no effect on native ConvexOS file systems.

Non-network system calls

System call performance has been improved through changes to the network protocol stack (i.e. STREAMS) which reduce the requirement for exclusive kernel access to the processor complex. In versions 11.0 and 11.1, exclusive access caused certain situations involving high network loads to degrade overall system call performance. By eliminating some of the sources of contention, the kernel's ability to handle non-networking system calls concurrently with network operations has been improved.

ZIC time libraries

Some of the library functions used by ConvexOS utilities have been optimized to yield better performance by reducing the number of system calls they make. These libraries impact a variety of utilities. The improvements are particularly noticeable as a reduction in the amount of CPU time used. The `cron` and `syslogd` daemons are examples of affected processes.

Kernel buffers and tables

Kernel virtual address space has been increased to approximately 125 MB up from 85 MB, an increase of 40 MB or approximately 50%. For large physical memory systems, this allows more physical memory to be set aside for tables and buffers, thereby improving performance. One of the main advantages of this modification is the potential increase of STREAMS data blocks (DBLKs) created for sites with one or more HiPPI interfaces. Only an additional 45 K of physical memory is reserved for the required memory map entries.

File Systems

Performance has been enhanced for file system writes and buffer cache virtual memory usage.

Full block writes

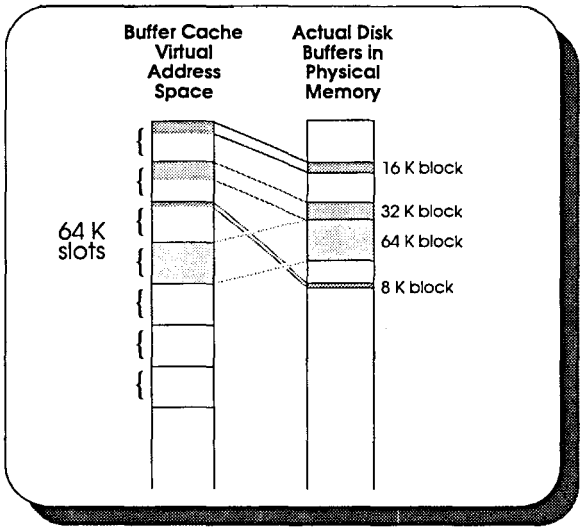
With ConvexOS 11.5.1, when writing a new disk block, if the entire block is written, overhead is reduced. The result is an increase in speed and a reduction in system time. Preliminary lab testing has shown a performance improvement ranging from 3% to 10% depending on configuration and architecture. The `libc` standard I/O library (`stdio`) does full block writes where possible; hence, many applications will benefit from this change.

Buffer cache

The maximum virtual address space available for the buffer cache was previously fixed at 512 MB; this limit has been raised to 1 GB by default. On systems with small physical memory configurations, buffer cache is scaled down appropriately. A tunable has also been added to allow the size to be limited if necessary. Note that, in general, the total amount of cached data will not attain the limit because of how the buffer cache address mapping is implemented. Each buffer uniquely maps to a 64 K virtual address slot regardless of its actual disk block size (from 512 bytes to 64 K). The actual amount of physical memory associated with the buffer is based on the block size of the underlying file system; hence the address space for the buffer cache is likely to be sparsely filled since block sizes are often smaller than 64 K, as shown in Figure 3.

This change can be seen as a 2X read performance increase due to more frequent buffer cache hits, but only for systems which tend to not require a lot of page outs; i.e., they usually have available free memory).

Figure 3 Memory Mapping



Virtual Memory

Frequently-used virtual memory routines have been re-engineered to reduce system time (i.e., system overhead). In addition, various other changes have been made to improve paging performance.

Process initialization

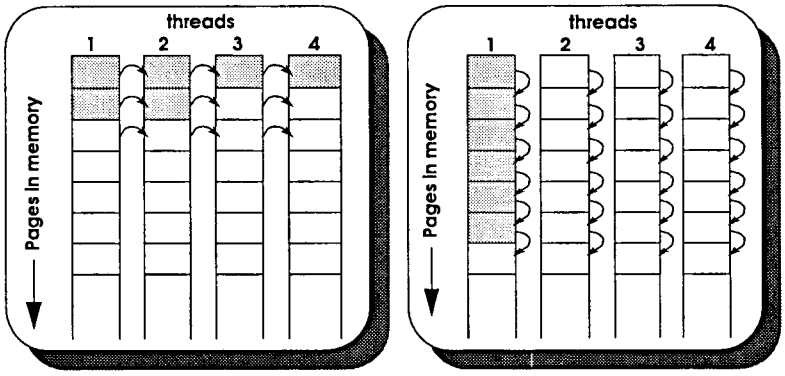
Virtual memory initialization has been sped up by using the system's ability to invalidate the entire Address Translation Unit (ATU) cache. The performance improvement is especially noticeable when initializing a process with a large address space. A 2 GB process, for example, has 500,000 pages, whose ATU cache entries were previously updated on a page-by-page basis. This process was fairly time consuming. Applications that allocate many large arrays (totaling approximately 512 MB or more) will benefit most from this enhancement.

Thread address space paging

The I/O subsystem can read and write in 64 K increments per operation. Therefore, paging speed can be significantly improved by reading or writing multiple pages at once. Page-ins and page-outs of thread-private memory are now clustered to avoid the need to perform repeated operations on the same disk block. This change can speed up paging of thread memory by as much as a factor of sixteen. The procedure for paging multiple threads

has also been fundamentally redesigned, so that it is done on a per-thread basis, rather than across all threads simultaneously. The difference between these two approaches is illustrated in Figure 4.

Figure 4 Revised Thread Paging Sequence



Old method of thread paging **Improved thread paging**

Thread virtual space allocation

Existing versions of ConvexOS do not accurately track usage of thread virtual memory, making it possible for the kernel to grant virtual memory to processes for which it was unable to actually provide swap space. This situation could lead to serious problems, including the inability to create new processes or a complete hang of the operating system. Due to more accurate tracking in ConvexOS 11.5.1, some applications could require more swap space than when running on an 11.0/11.1 system.

Shared memory

In earlier versions of ConvexOS, shared memory that was mapped with the mapping attribute MAP_ANON was found to be inefficient when paged in or out. The problem was due to a limitation of the virtual memory subsystem which prevented clustering. In version 11.5.1, this limitation has been eliminated. The performance improvement this provides is similar to that for thread memory; i.e. a speed-up of as much as a factor of sixteen. Note that for both these types of clustering, performance will already be non-optimal since swapping is taking place; however, these changes reduce the severity of the performance loss when swapping.

Networking

Considerable effort has been put into improving networking performance in the ConvexOS kernel. This has included numerous optimizations and performance-oriented enhancements, as well as several important improvements to the parallelization of network routines. See the "Networking" section on page 34 for details.

C4600-Specific

The kernel functions (`vcopy` and `bcopy`) used to move data between buffers (user process and-or kernel) have been modified to take advantage of architectural improvements unique to the C4600. The function `vcopy` uses the vector registers to copy data while the function `bcopy` only uses scalar registers.

The `bcopy` function has been modified to use all 28 scalar registers. Preliminary lab testing has shown performance improvements ranging from 3.5X to 4X depending on address alignment. For "like" address aligned copies, preliminary lab testing has revealed a maximum copy performance of approximately 400 MB/sec.

The `vcopy` function has been modified to take advantage of a C4-unique vector instruction. This improves copy performance when copying data between memory addresses which don't share "like" alignments. This results in performance improvements ranging from 2X to 4X depending on alignment differences. For "like" address aligned copies, preliminary lab testing has revealed a maximum copy performance of approximately 500 MB/sec.

Note that all of the above rates are raw memory-to-memory copy rates and do not include any additional overhead in the kernel. In general, these improvements reduce process system time required for I/O, resulting in improved system-wide aggregate performance. For file system reads producing a buffer cache hit, a significant performance increase is achieved.

Similar improvements have accelerated the computation of TCP network data checksums, especially when the quantity of data is small. This results in faster computation of TCP checksums for small packets (less than about 1024 bytes).

Enhancements

In addition to increased performance, new features have been added and existing features have been improved upon in 11.5.1.

Networking

Due to the number of network-specific enhancements, a separate section dedicated to detailed descriptions has been included beginning on page 34.

New Devices

Ethernet and QSC devices have been added to ConvexOS in 11.5.1.

Ethernet

Support for the Interphase VME 4207 Eagle Ethernet LAN-205 controller has been added to provide an alternative for the VME Excelan Model 302 Ethernet controller on C-Series architectures (other than the C4). A sysgen is required for the Interphase controller for C-Series C2 and C3 machines.

QSC

To allow customers to take advantage of cost-effective SCSI peripherals, support for several new tape and disk devices has been included in ConvexOS 11.5.1.

The Quad SCSI Channel (QSC) is a PBUS CCU providing four fast and wide differential SCSI busses. The QSC and related hardware are used with many types of SCSI peripherals such as the Elite 9 disk drive and the various high performance tape drives.

The QSC relies on the CPU for virtual-to-physical address translations. By eliminating the concept of virtual memory address space on the CCU and using the CPU's ability to do virtual-to-physical address translation, the QSC realizes a significant improvement over previous channel controllers in PBUS efficiency.

Other basic features of the QSC include:

- Full support of PBUS transfer protocol
- Four fast & wide differential SCSI-3 16-bit differential channels
- A maximum cable length of 30 feet
- 80 Mbyte/second peak PBUS transfer rate
- 20 Mbyte/sec burst rate of fast & wide SCSI-2 or SCSI-3
- 16 kbytes of SCSI channel to PBUS buffer space per channel
- On-board 88100 RISC microprocessor
- Window-mapped main memory access
- Independent DMA (Direct Memory Access) channel for MPU to main memory block transfers
- Local channel control
- An NCR 53C720 SCSI Controller Chip with on-board SCRIPTS processor for each of the four channels

Disk

Seagate Elite 9 QSC disk device support has been added to ConvexOS in 11.5.1.

Tape

QSC tape devices added to ConvexOS in 11.5.1:

- HP/Convex C590 IBM Magstar (NTP)
- STK SD-3 (Redwood)
- STK 9490 (Timberline)

VME SCSI tape devices added to ConvexOS in 11.5.1:

- Quantum DLT-2000
- Quantum DLT-4000
- Metrum RSP 2150
- Exabyte 8200, 8500, and 8505
- Ciprico VME SCSI RF3576 (F&N)

Kernel

Many improvements have been made to the ConvexOS kernel in the 11.5.1 release, including the merge of ConvexOS 11.0, ConvexOS 11.1, 11.0 Realtime and the Quad SCSI CCU (QSC).

TAC patch integration

Refer to page 48 for a list and description of the ConvexOS 11.0/11.1 kernel TAC patches incorporated into ConvexOS 11.5.1.

UIDs and GIDs

The number of possible UIDs and GIDs has been effectively doubled, now allowing 65,000 of each. Both ConvexOS and its utilities have been modified to support this change.

Per process addressability

The maximum size of thread virtual memory has been expanded from 64 MB per thread to 384 MB per thread. This raises the maximum possible thread memory for a process from 512 MB to 3 GB, allowing the creation of a 4.6 GB process.

System statistics

The `pstat` utility has been enhanced to include additional system information:

- TCP/IP connection information
- Buffer information
- Stack backtrace
- Statistics on MBLKs, DBLKs, and queues
- All network STREAMS information
- All STREAMS queues
- USER area information
- MBS information

ConvexOS Utilities

New releases of `perl`, `sendmail`, and GNU Emacs are included in 11.5.1. In addition, `cron` daemon reliability has been improved.

Perl

`/usr/bin/perl` has been upgraded from version 4.010 to version 4.036, which we feel is the most stable version of Perl that has been released. (Perl 5 has not been included due to compatibility issues.) Perl 4.036 offers numerous fixes and compiler enhancements. For example, on an 11.0 system, the command

```
/usr/bin/perl -e 'eval "1 #comment" && print "foo\n"'
```

produces no output. It actually should print "foo", which is indeed what it does on an 11.5.1 system. The compiler in Perl 4.036 is stricter in some ways than the one in 4.010. Certain instances of poor syntax slipped past the older version but are not accepted by the newer one. To insure compatibility with existing scripts, the installation procedure will save a copy of the existing `perl` binary before installing the new one.

Previously, when specifying a field width for a number in a call to `printf`, the number would be padded on the left with zeros. Spaces are now used instead. For example,

```
printf("|%05.10s|\n","1");
```

previously produced the following as output:

```
|00001|
```

Now it produces the output:

```
|     1|
```

Some scripts that relied on the old behavior (to generate filenames, for example) may need to be modified in order to work correctly with the new version of Perl.

Sendmail

The utilities release will include an upgraded version of `sendmail`, based on Sendmail 8.7.5. This provides numerous improvements over the currently available version, including, but not limited to the following:

- Basic reliability improvements
- Much improved MIME support
- Accepts SIGHUP to restart the daemon
- Enhanced `sendmail.cf` readability with descriptive option names, instead of single letters
- Automatic lookup of the addresses of all the network interfaces to discover "alternate" hostnames for the machine

Some security fixes are also included with Sendmail 8.7.5:

- Checks for buffer overflows for DNS/NIS lookups
- Protects against overflowing `syslog` buffer; (`syslog` has also been fixed in 11.5.1. See CERT Advisories and Other Security Issues on page 38 for more information about `syslog` modifications.)
- Supports a maximum message size to prevent denial of service attacks
- Tries to validate outgoing messages in the queue to prevent forgery attacks on queued messages

For a complete list of Sendmail 8.7.5 enhancements, see the file `/usr/lib/conf/sendmail.11.5/RELEASE_NOTES`.

GNU Emacs

Emacs is now a part of the 'contrib' product, and is based on the freely copyable GNU Emacs 19.28, a version which has been proven to be quite stable. It offers considerable new functionality over the version shipped with ConvexOS 11.0, which was based on GNU Emacs 18.57.

The X-windows support was rather minimal in Emacs 18.57. Windowing capabilities have been greatly expanded in 19.28.

Several packages have been added in Emacs 19.28, including the following:

- GNUS — a UseNet news reader
- Hexl mode — for editing binary files in hex dump format
- Emerge — allows interactively merging two sets of modifications to a single file
- New editing modes — including modes for C++, `awk`, makefiles, `perl`, and SGML

Additionally, there is now a history for the mini-buffer, allowing previous commands to be recalled. Numerous other key bindings and editing capabilities have been added to improve the overall environment. One especially nice feature is the addition of the variable 'next-line-add-new-lines', which can be used to disable the annoying addition of blank lines when attempting to move beyond the end of the buffer.

The version of Emacs included in the 'contrib' product will not replace the 11.0 version of Emacs; instead it is installed in `/usr/contrib`. To use the new version of Emacs, place `/usr/contrib/bin` ahead of `/usr/convex` in your PATH environment variable. Users who wish to continue to use the old version can simply place `/usr/convex` ahead of `/usr/contrib/bin` in their PATH variable. In either case, adding `/usr/contrib/bin` will make available several new copylefted utilities; namely Gzip 1.2.4, Lynx 2.5FM, tcsh 6.04, and GNU xargs 4.1.

Cron

The reliability of the `cron` daemon has been improved by eliminating certain circumstances that could cause it to skip jobs for arbitrary periods of time, or simply terminate. The scheduling algorithm has been fundamentally redesigned.

Public Domain Utilities

ConvexOS 11.5.1 `/usr/contrib/bin` contains a number of helpful utilities. Users may wish to add `/usr/contrib/bin` to their PATH environment and `/usr/contrib/man` to their MANPATH to easily take advantage of these utilities. Public domain utilities shipped with ConvexOS 11.5.1 include the following:

- `aliastree` prints recipients of an alias in `/usr/lib/aliases`
- `backtrace` print a backtrace of a sleeping process
- `chroot` simple wrapper around the `chroot()` system call. This will execute the specified command in an environment whose root directory is the specified path
- `csh` C-shell version of `tcsh` 6.04
- `dblk` displays which connections are currently holding MBLKs and DBLKs
- Executables for the GNU Emacs 19.28 package:
  - `emacs`
  - `emacs-19.28.0`
  - `emacsclient`
  - `ctags`
  - `etags`
  - `b2m`
  - `rcs-checkin`
- Executables for the Gzip 1.2.4 package:
  - `gzip`
  - `gunzip`
  - `gzexe`
  - `zcat`
  - `zcmp`
  - `zdiff`
  - `zforce`
  - `zgrep`
  - `zmore`
  - `znew`

- `lynx` — Lynx 2.5-FM World Wide Web browser
- `noenv` — Simple program to execute a command with an empty environment. Usage: `noenv` *command*
- `setreuid` — simple wrapper around the `setreuid()` system call. Usage: `setreuid` *UID eff_UID command* Both UID's should be specified as the numeric UID desired. Only root can successfully execute this utility
- `tcsh` — `tcsh` 6.04
- `udate` — Prints out the number of seconds since 12:00 GMT Jan 1, 1970
- `xargs` — GNU `xargs` 4.1

TAC Tools

A number of utilities developed by the Hewlett-Packard Convex TAC have been included in the `/tac/tools` directory. These tools are unsupported scripts without man pages. For more information about any of these tools, contact the TAC. The `/tac/tools` directory may contain the following utilities:

- `autonice` — perl auto-nice script. Looks for interactive jobs and renices them in favor of batch jobs. Sends encouraging email to remind users to use the batch queues
- `chkpntall` — perl script that checkpoints all running batch jobs; allows batch system to recover in case of a system crash and restore jobs to their checkpointed state
- `clearsema` — C program used to clear tape drive problems
- `diskinfo` — perl script that parses the `/etc/disktab` file to display the specific information for a given device, such as a DKD-505 disk
- `filewatch` — perl script that monitors a list of files and sends email if they change
- `getrfc` — `sh` script to obtain a copy of a desired RFC from the InterNIC, keeping a local cache for subsequent requests
- `gipcp` — perl script that copies a GIP `installsw` product image between disks and/or tapes. Can be used to copy the files from a tape to a directory so the directory can be shipped via FTP to another site, where

	`gipcp` can be used to copy the files back onto a tape
• `isbackground`	C program that checks to see if the process is running in the background. Can be used in a script to test whether to produce output or prompt for input
• `mkbc`	perl script that make barcodes for Metrum tapes via PostScript
• `netcheck`	C program that will go through various portions of the local network and determine if they are in use and working
• `nicerun`	runs a command as a specified user at a specified priority
• `pt`	prints a process tree, showing parent/child relationships
• `revcheck`	`csh` script that prints the version of various parts of a CONVEX system, such as software revision table and `/etc/fstab` contents
• `stage`	sh script that stages batch jobs to be run serially. This can be used when a series of jobs needs to be run in which one job requires the output from the previous job to be delivered before starting
• `tapescan`	C program that analyzes a tape, reporting tape device flags and statistics

Networking

A number of the improvements made in 11.5.1 are in the area of networking. This section is dedicated to a more thorough description of these modifications.

NFS

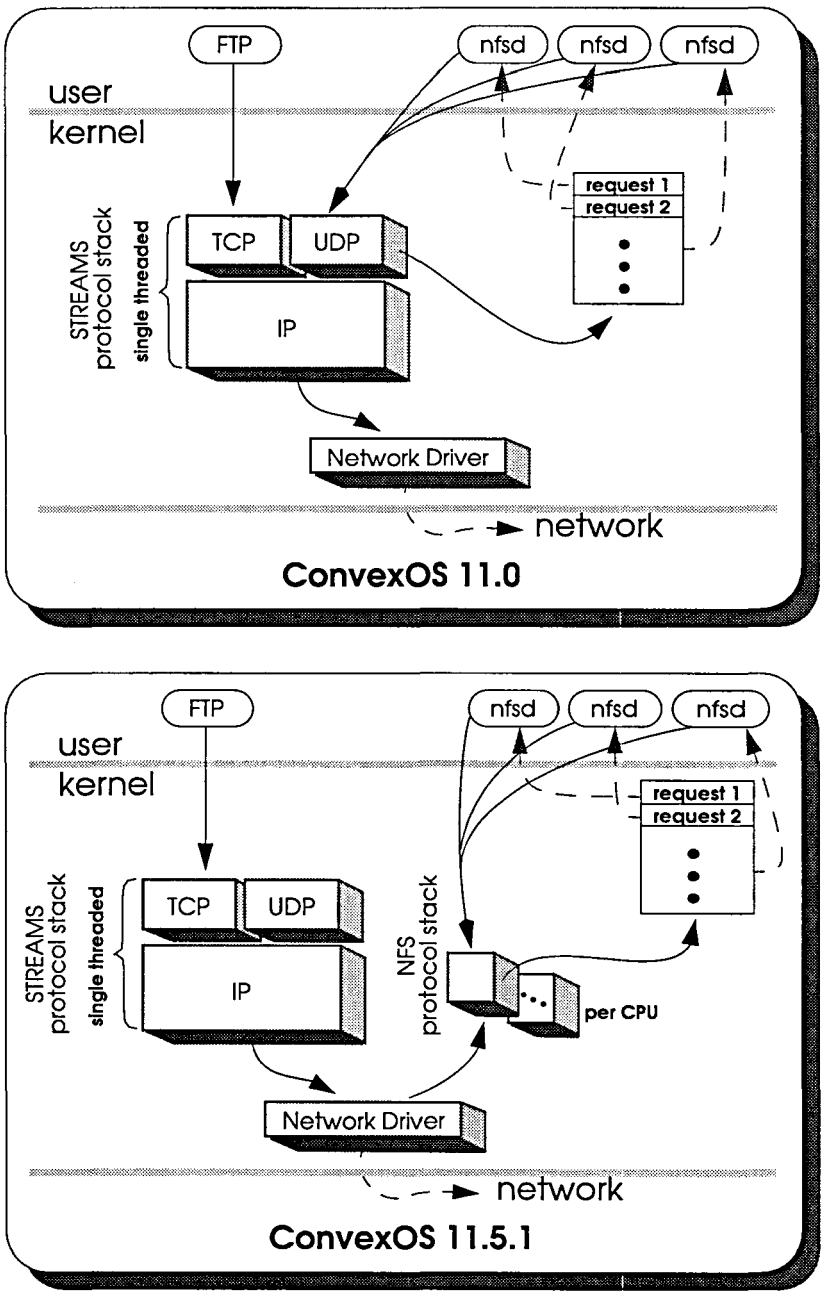
A separate path through the kernel has been implemented for handling NFS traffic. This allows NFS to operate independently of all other networking, further reducing resource contention, as shown in Figure 5 on page 35. The new NFS code represents an improvement for both the ConvexOS NFS client and the server.

Large NFS block sizes are now supported. The NFS client in ConvexOS 11.5.1 can handle power-of-two sized blocks up to 32 K; the server can handle arbitrary-sized blocks up to 60 K. This should make transfers more efficient over networks such as HiPPI that support large packets.

Client performance of the `stat` system call over NFS file systems was inefficient in ConvexOS 11.0/11.1. This showed up as a delay in commands such as '`ls -F`'. In 11.5.1, the performance of flushing NFS client buffers back to the server was improved, resulting in much more reasonable performance.

UDP checksums are now turned on by default. The code for computing NFS UDP checksums has been moved to the process level, further reducing contention for kernel resources.

Figure 5 Comparison of NFS Operations



General

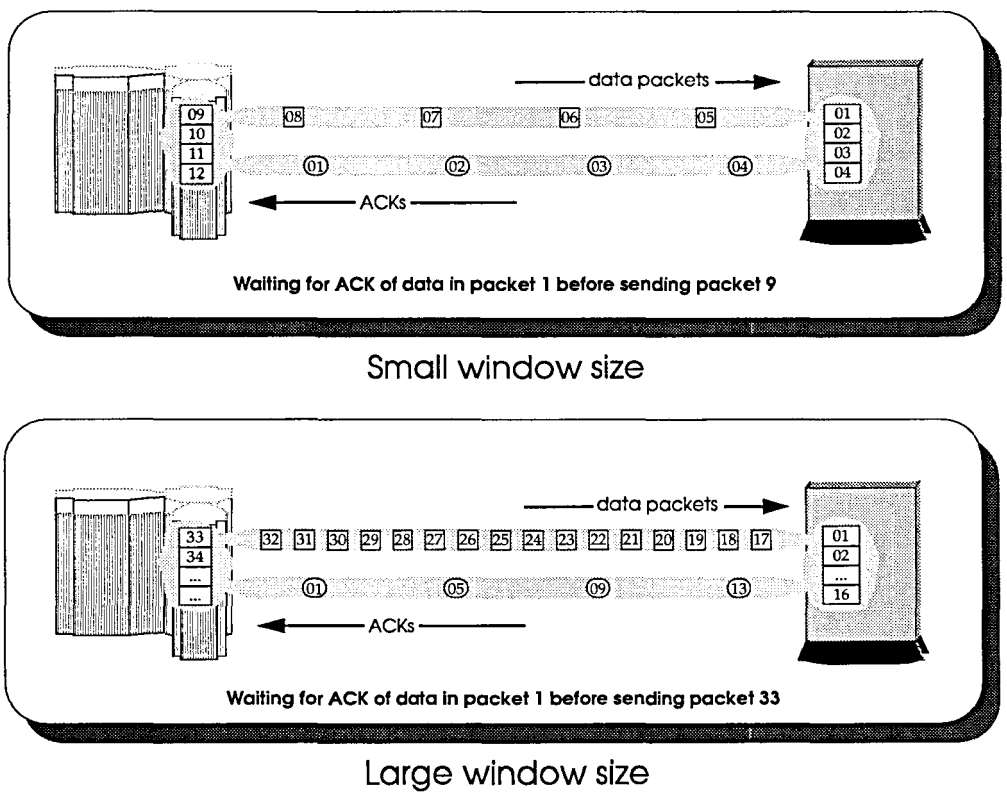
In addition to NFS changes, several improvements have been made that affect network operations in general.

Performance

TCP window scaling is now performed by default over all network interfaces. This improves communication speeds over high-bandwidth networks such as HiPPI, since it reduces the time spent waiting for packet acknowledgment (ACKs) from the remote host, as shown in Figure 6. Tunables exist for changing the default window size for individual interfaces.

Kernel overhead has been reduced on a per-packet basis for some aspects of networking by reducing unnecessary scheduling activity in the kernel. Frequently-used networking functions have been optimized and inlined; this improves performance for all aspects of networking.

Figure 6 Window Scaling



Monitoring

The `strstat` utility has been enhanced to give more detailed information about networking buffers. It now displays statistics regarding the number of DBLK allocation requests promoted to the next larger size because they could not be satisfied using the size requested. This provides administrators with the information needed to tune DBLK distribution for maximum efficiency.

Additional network statistics are available using the ConvexOS 11.5.1 `pstat` utility. For a list of enhancements to `pstat`, see System statistics on page 27.

Updates have been made to the `tcpdump` utility, which provides the ability to monitor all ethernet network traffic. HiPPI can now be monitored with `tcpdump`, and the traffic filtering capabilities have been significantly improved. The kernel's support for the ethernet driver in promiscuous mode has been made more robust and secure. Promiscuous mode is automatically turned off whenever a process using this mode is terminated. Log entries are also made whenever promiscuous mode is turned on or off. These changes significantly improve the usability of `tcpdump` over ethernet.

Miscellaneous

The `ifconfig` utility can now be used to restart the HiPPI interface (by taking it down and back up again). Before this change, there was no way to recover from a HiPPI hang other than rebooting the system.

The BIND resolver library's hostname lookup functions now support searching of multiple domains via a new "search" configuration directive.

CERT Advisories and Other Security Issues

CERT Advisory CA-94:15 documents a vulnerability in portmappers that allow proxy access, which makes it possible for unauthorized systems to obtain an NFS file handle if the server exports to itself. The 11.5.1 version of `portmap` now rejects certain proxy requests from remote hosts to prevent this sort of attack.

Various other NFS bugs have been found recently that affect many operating systems. No CERT advisories have been issued for these yet, but the fixes have been incorporated into ConvexOS 11.5.1. One of these changes has imposed a restriction on the `bind` and `connect` system calls. `connect` must either be called by root or by the same UID that called `bind`.

CERT Advisory CA-95:13 documents a vulnerability in the `syslog` routine which potentially makes it possible for local and remote users to execute arbitrary commands. In particular, a remote user could fool `sendmail` into calling this routine with arbitrary data that would overflow the internal buffer. Though exploitation would require in-depth knowledge of the C-Series architecture, `syslog` has now been modified to insure that such an overflow cannot happen.

All occurrences of the `system` function have been removed from `/bin/login` to prevent environment-based attacks similar to those described in CERT Advisory CA-95:14.

In ConvexOS 11.0/11.1, non-root users could access raw network devices, which made it possible to compromise security by, for example, turning on ethernet promiscuous mode. Direct access to these devices has been disabled, but various utilities have been modified to allow users to continue to access status information for these interfaces. (Commands such as '`ifconfig eth0`' still work, for example.)

Impact on User Interface

This section summarizes all changes incorporated into 11.5.1 categorized into sections applicable to different types of C-Series users. Although these topics are discussed in detail in other sections of this document, the focus may not have been on the impact to the end user. The information is categorized as follows:

- Administration utilities
- User utilities
- Library functions
- System calls
- Performance
- Kernel tunables
- New devices

Administration utilities

- `gated(8)` updated to version 3.0
- `gut(8)` larger resulting file (output now includes symbol hash table)
- `spu/jpstat(8)` new option prints swap map
- `/etc/resolv.conf` new `search` option for multiple domain searches
- `ifconfig` up and down operations on HiPPI are restricted to the superuser; for HiPPI, "down" resets CCU driver
- `inetd` new -s option for denial of service attacks
- `snmpd` new configuration file allows specification of MIB variables
- `timed` raw socket emulation added, which is required by the `timed` utility
- `mount` new NFS feature to use large packets, between 8k and 60k
- `strstat` new -p option aids in tuning DBLK allocations
- `MAKEDEV` new options provide ability to take advantage of SCSI `mt reset` (through creation of new device entries)

• `rdump`	new options **S** and **U** for unattended remote dump
• `spu/crashdump(8)`	verifies `osclean` has been executed
• `spu/prtlog(8)`	improved to provide more information and to simplify `/mnt/errlog` output
• `sa(8)`	multiple simultaneous invocations are prevented by locking the file `/usr/adm/lastacct`
• `/etc/sbtab`	new file that automatically records superblock locations when file systems are created
• `portmap(8)`	rejects proxy requests for security reasons
• `dump(8)/restore(8)`	options added to correctly process migrated files
• `backtrace(1)`	new utility to display a process backtrace
• `lpd(8)`	`/etc/printcap` new "ac=" value provided for access control
• `tcpdump`	now supports HiPPI/FDDI; promiscuous mode (-p) is more secure
• `sysgen`	Hyperchannel, FDDI, Eagle Ethernet now require a sysgen
• `syslog`	improved security as per CERT advisories
• Network devices	for security purposes, the /dev entries are no longer world-writable, and the utilities that had relied on the open permissions have been fixed to work with the new restrictions
• `quota(8)`	output format changes
• `/usr/lib/contactcap`	hostname changed from convex.com to convex.hp.com
• `zic`	DST rules for Europe have been changed
• kernel VM space	maximum possible kernel VM space increased to allow more buffers for improved performance (e.g., DBLK allocations)

• `sendmail`	version 8.7.5 is now incorporated; contains multiple improvements and additional internet functionality
• troubleshooting	in event of a ConvexOS system panic, backtrace/signature is recorded in `/mnt/errlog` for more expeditious debug analysis. This can eliminate the need for a crashdump

User utilities

• `pstat`	new functionality/options: **-b -c -d -m -n -q -u -o -L -T**
• `df(1)`	new **-I** option displays current in-use NFS rsize and wsize values
• `ping(1)`	can now ping a broadcast address; new **-f** option allows the superuser to flood a host with packets in order to test reliability under high load conditions
• `getsysinfo(8)`	**num_vregs** option added
• `find(1)`	new **-fsonly** option
• `sysstat(8)`	new utility for monitoring system activities
• `mt(1)`	new options **reset** and **eod**
• `make(1)`	now imports the environment and presets internal variables
• `ls(1)`	new **-e** option enables expanded time format
• `sed(1)`	restrictions on arbitrary regular expression delimiter removed
• `sh(1)`	**-h** option is now an alias for **-l;** built-in `pwd` command removed
• `perl(1)`	newer 4.036-based perl; was 4.010 in 11.0
• `cron(8)/tellcron(1)`	many improvements,including how DST is handled and security changes
• `less(1)`	newer version - options have changed
• `ftp(1)`	reliability improvements and output format changes
• `syspic(8)`	now includes striped file system information and multiple network interfaces

• `tpdaemon(8)`	was not shipped with ConvexOS 11.0; is now bundled with the 11.5.1 release. Reliability has been improved via incorporation of STK ACSLS 4.X interface libraries

Library functions

• `gethostbyname(3)`	searchable domains increased from 3 to 6. Also affects the following functions: – `res_mkquery(3)` – `res_send(3)` – `res_init(3)` – `dn_expand(3)`
• `seekdir(3)/telldir(3)`	token value returned is encoded differently to better handle large directories more efficiently
• UID typecast	changed from short to unsigned short
• GID typecast	changed from short to unsigned short
• ANSI C	prototype "setgroups" now defined in `unistd.h`

System calls

• `bind(2)/connect(2)`	UID security restrictions added
• `socket(2)`	types of "SOCK_RAW" now supported
• `mount(2)`	now returns NFS **actual** mounted rsize/wsize values in the `nfs_args` structure. This value is clipped to the tunable value for the specified interface.
• `listen(2)`	backlog value can now be up to 100 by the tunable `somaxconn`. This was limited to 5 in 11.0

Performance

1. Parallel NFS protocol stack bypass implemented (STREAMS protocol stack bypassed)
2. Large block reads/writes for NFS now supported; up to 60k from a maximum of 8k
3. Exclusive kernel access no longer required by protocol stack
4. Buffer cache virtual space increased from 512MB to 1024MB
5. CCU transport mechanism (MBS) has improved scalability and reduced latency
6. Automatic TCP window scaling for network interfaces with a large MTU
7. TCP checksum algorithm improved for all architectures
8. Initialization of buffers has been eliminated for file system full-block writes
9. `bcopy` for buffer cache improved for the C4 architecture
10. Initialization of huge VM spaces at process creation time is now much faster
11. Clustered page-in of thread and shared memory implemented
12. Optimization of frequently-used kernel VM functions
13. `localtime(3)` used to build utilities has been dramatically improved. Improvement most noticeable in utilities that call `localtime` potentially hundreds or thousands of times per invocation; e.g. `perl`. TAC patch all/3.0.131 contains the fixes for the library routine, so that user-compiled programs can also benefit from this change
14. Dramatic speed increase to find and display man pages

Kernel tunables

A summary of changes to kernel tunables is provided in this section.

New tunables

File migration

- `nfs_ret_mig_stat` Default: 0 (=disabled)

NFS per interface tunables for large packets

- `nfstsize_loopback` Default: 61440
- `nfstsize_hippi` Default: 61440
- `nfstsize_fddi` Default: 8192
- `nfstsize_eth` Default: 8192
- `nfstsize_default` Default: 8192

NFS protocol stack bypass

- `nfsfastsend` Default: 1 (= enabled)
- `nfsfastrecv` Default: 1 (= enabled)

Explicit table sizing

- `explicit_table_size` Default: 0 (=disabled)

  If `explicit_table_size` is 1 (enabled), the following values are used, else table sizes are automatically calculated based on physical memory:

  - `inode_table_size` Default: 1024
  - `file_table_size` Default: 1024
  - `ncache_table_size` Default: 1024
  - `dquot_table_size` Default: 1024

Delayed Unwire Cache enable for raw disk

- `duc_enable` Default: 1 (=enabled)

Automatic TCP window scaling

- `ethernet_default_window` Default: 31744
- `fddi_default_window` Default: 64512
- `tcphippi_default_window` Default: 2097152

Realtime extensions

- `rt_enabled` Default: 0 (=disabled)
- `realtime_grpid` Default: 0
- `roundrobin_freq` Default: 10
- `fast_clock` Default: 0 (100 Hz)

MBS tunables

- si_mbs_limit Default: 1023
- mbs_enabled Default: 1 (=enabled)

  If mbs_enabled is 0 (disabled), the xxx_mbs_limit and xxx_mbs_drain tunables are ignored
  - hippi_mbs_drain Default: 5
  - hippi_mbs_limit Default: 16
  - org_mbs_drain Default: 2
  - qsc_mbs_drain Default: 5
  - qsc_mbs_limit Default: 16
  - viop_mbs_drain Default: 2
  - viop_mbs_limit Default: 16
  - idc_mbs_drain Default: 5
  - idc_mbs_limit Default: 4

VIOP tunables

- st_3480_synch Default: 1
- st_3490_read_retry Default: 64
- st_3490_synch Default: 1
- st_3490_write_retry Default: 64
- st_dlt_synch Default: 1
- st_exb8500_synch Default: 1
- st_exb8505_112m_bit Default: 1
- st_exb8505_density Default: 0
- st_ignore_attach_err Default: 1
- st_max_busy_retry Default: 30
- st_rsp2150_synch Default: 1
- st_show_recovered_err Default: 0
- st_spt_synch Default: 1
- st_trace_pg_cnt Default: 2
- ta_viop_prt_mask Default: 0xffbf

Other new parameters

- tcp_loopback_mtu Default: 65536
- n_async_proc Default: 4
- bufcache_size Default: 1024 (limited by the kernel to 4 times physical memory)
- elite9_max_out Default: 8
- halt_on_panic Default: 0 (=disabled)

- qsc_max_io_reqs — Default: 400
- sig_cannot_mask — Default: 0x600
- somaxconn — Default: 5 (limited to 100 by kernel)

Obsoleted/removed tunables

- nfs_disable_wc
- trace_off_on_ifree_rebuild
- ifree_panic
- ifree_rebuild

Default value changes

Buffer cache headers freed

- getnewbuf_goal — From 32 to 512

Streams dblk tunables

- str_dblk_2048 — From 1024 to 1280
- str_dblk_1024 — From 1024 to 512
- str_dblk_512 — From 1024 to 1536
- str_dblk_128 — From 1024 to 256
- str_dblk_64 — From 1024 to 1536
- str_dblk_16 — From 1024 to 128
- str_dblk_4 — From 1024 to 64
- str_dblk_0 — From 1024 to 100

MBS tunables

- du_mbs_limit — From 500 to 1023

Streams threshold tunables

- str_lo_pct — From 60 to 80
- str_met_pct — From 80 to 90
- str_ctl_sz — From 16384 to 65536

UDP protocol checksums

- udpcksum — From 0 to 1

FDDI (changed for new MBS design)

- fd_max_recv — From 28 to 128
- fd_max_xmit — From 28 to 64

Limits changed

- nstbuf — Maximum increased from 8192 to 16384

New devices

New devices supported in ConvexOS 11.5.1:

- Standard QSC
- Interphase VME 4207 LAN-205 Eagle Ethernet
- Ciprico VME SCSI RF3576 (fast and narrow); the original Ciprico SCSI controller is slow and narrow
- Tape drives on VME SCSI
  - DLT-2000
  - DLT-4000
  - DAT (Archive Python)
  - DAT (Diamond Back)
  - Metrum RSP-2150
  - Fujitsu 3480 w/stacker
  - Fujitsu 3490E w/stacker
  - Exabyte 8200
  - Exabyte 8500
  - Exabyte 8505

New SCSI peripherals supported on QSC

- Tape
  - IBM Magstar (NTP)
  - STK-SD3 (Redwood)
  - STK-9490 (Timberline)
- Disk
  - Seagate Elite 9

TAC Patches

The following ConvexOS TAC patches have been integrated into the 11.5.1 release.

Kernel

The list below summarizes the kernel patches made since the release of ConvexOS 11.0/11.1 which have been checked in for ConvexOS 11.5.1. Patches that are not explicitly listed may be included in another patch.

- 11.0.129 — Networking Abort: Heavy TCP/IP activity can result in aborted connection (Connection Reset).
- 11.0.130 — Kernel Panic: kernel aborts immediately when running allocating a RAM disk via `rdcontrol`.
- 11.0.131.2 — Networking Hang: Networking subsystem hangs due to DBLK leak in NFS.
- 11.0.132.1 — Kernel Panic: System crashes with a (vm,6637) cmwire: write count < 0.
- 11.0.144 — Performance: Poor system performance may occur during some allocations of large amounts of memory (typically arrays totaling approximately 512 MB or more).
- 11.0.152.1 — Process Hang, Networking corruption, or Kernel Panic: Packet duplication seen with UniTree+ when DBLK allocation errors occur, process hangs writing to a pipe, or system crashes with the following cause:"`osi_Alloc: Unable to allocate 16 bytes - out of physical memory!`"
- 11.0.155 — Kernel Error: Disk quota system fails due to insufficient table space. The following message is seen: "`dquot table is full`". This patch allows sizing several kernel tables via tunables.
- 11.0.158.1 — File System Error: Advisory locks are left in place for a file after the locking process has exited. In addition, multiple processes can obtain an exclusive lock on the same file.
- 11.0.159 — Kernel Panic: The system crashes inside the IDC disk driver with message: "`I/O System: FATAL ERROR: (disk,3252) dustrategy:`" `pq_cmi_fm_alloc`

- 11.0.167.2 Networking Error: NFS uses incorrect file attributes under certain circumstances.
- 11.0.168.2 Process Hang: Processes on multi-CPU systems can hang due to a corrupt inode free list in the kernel. Also fixes file system hangs due to UniTree+ local loopback mounts.
- 11.0.169 Kernel Hang: Obscure swap code hang.
- 11.0.174 Kernel Error: Added `ifconfig` entry support for TCP/IP HiPPI.
- 11.0.178 Process Hang: Process hangs reading from a tty when using timed reads (commonly used by applications like `zmodem`).
- 11.0.179 Kernel Panic: Setting an DBLK allocation size group to count zero would crash the system if that size increment was referenced.
- 11.0.180 Process Hang, Networking Hang: An NFS client can hang during an attribute cache invalidation inside the kernel.
- 11.0.182 Performance: Corrects poor performance when overwriting a migrated file. Applies to FileServ and CVDM.
- 11.0.184 Kernel Panic: System crashes due to a heavy IP packet load.
- 11.0.185 Process Hang: Corrects a CVDM daemon hang.
- 11.0.186 Kernel Panic: System crashes with the message "`cmgetblk assumption`"
- 11.0.190.8 This patch is a combination of numerous networking patches:
  - Kernel Panic: System panics with the message "`tcp,7712 - tcp dropped data.`"
  - Kernel Panic: System panics with a PTE violation when a certain type of IP source routing packet is received.
  - Kernel Error: MBLK leaks occur in TCP processing code.

- Kernel Error: A UniTree+ disksrvr process spins in an infinite loop, writing many error messages of the form: "`Jul 20 16:54:04 recv(46, 0x809702b8, 8, 0x0) returned -1, error 60 (Connection timed out)`" to the UniTree+ log file. Other network applications may also experience this problem.
- Kernel Error: Corrected TCP/HiPPI routing when out of sync TCP sequence numbers occur.
- Kernel Error: Restricted which applications may open network drivers.
- Kernel Error: Applications experience "connection reset" on receive or send. Typical error messages are: "`connection reset by peer`" and "`read/write pipe: broken pipe`"
- Kernel Error: Applications using TCP connections and the `select` system call can loop.
- Kernel Error: Unable to add a route even though '`netstat -r`' does not show the route in the route table. This causes interface routes to be removed by utilities such as `gated`. In addition, cross-device link errors can occur when pinging a host.
- Process Abort: A process may abort with a bus error or segmentation violation. This is due to corruption of an internal network data structure. FileServ processes such as fs_alloc_s and fs_alloc_t experience this problem.
- Process Hang: HiPPI/TCP connections can hang if the TCP window shift value is greater than two.
- Kernel Error: Multiple redirected route entries (through the same gateway) are left in the route table.
- Kernel Error: If UDP checksums are enabled, some or all RPC services including user applications which use UDP may not work.

- 11.0.193.1 Numerous VME SCSI tape driver fixes (hangs, timeouts, etc.).
- 11.0.196 Kernel Hang: File systems hang when disk quotas are turned on due to corruption of the internal quota table. Often, the message "`dquot table full`" will be seen in the system errlog.
- 11.0.198.1 Kernel Panic: System can crash when closing a socket with the SO_LINGER option.
- 11.1.144 Kernel Error: Redundant stripe driver parity check problem corrected.
- 11.1.150 Kernel Panic, Kernel Hang: System hangs or crashes with the message "`push_ctxblk: thread overflow`" when running an application with profiling on a C4600.

Accounting

- 11.0.128 `/usr/etc/sa -mP` produces incorrect output for the cpu time field.
- 11.0.129.1 `/usr/etc/repquota` and `/usr/ucb/quota` produce incorrect output for large quotas. Block values are incorrect.
- 11.0.132 `/usr/etc/ac` aborts with a bus error when using a `wtmp` greater than 2.2 MB in size.
- 11.0.133 `/usr/etc/diskuse` no longer terminates when a `statfs` fails due to an inaccessible file. This could occur with symbolic links to remote file systems.
- 11.0.134 Multiple copies of the accounting system could erroneously execute at the same time. The accounting system now uses an exclusive file lock to prevent this.

Core Utilities

- 10.1.233 — Corrects a core dump problem with `/bin/od` when dumping floating point format.
- 11.0.128.1 — C4600 systems only. The `/etc/cpu_monitor` daemon and supporting files are missing from some C4600 systems. This means that a head cannot be automatically returned on-line in the event of a C4600 CPU failure.
- 11.0.129.1 — `/usr/ucb/rcp` incorrectly truncates the destination file before the source has successfully been read.
- 11.0.132 — `/bin/sh` dies if one of its children is killed by a signal > 19 (such as XCPU). This affects CXbatch by making jobs silently disappear.
- 11.0.133 — `/bin/df` hangs referencing an NFS partition mounted on an alternate port (such as a UniTree+ mount with port=2094 in the `/etc/fstab` file).
- 11.0.134 — `/usr/etc/portmap` forwards proxy mount/nfsd rpc requests from remote hosts.
- 11.0.136 — Correct ZIC rules for European DST beyond 1995. Affect UK, Eire, and MET timezones.
- 11.0.138 — `/usr/etc/zdump` (ZIC dump) has been fixed to display all entries when the "`-v`" flag is specified.
- 11.0.140 — Increases directory size handled by `/bin/pax`, `/bin/tar`, and `/bin/cpio`. The utilities now allow approximately 50,000 files in a single directory.

Extended Utilities

- 11.0.128 — `/etc/dump` - when running unattended dumps, if a tape switch fails, dump does not abort.
- 11.0.130 — If a password has expired and domainname is set, `/bin/login` incorrectly executes `yppasswd` instead of `passwd` even if the user is not in the NIS passwd map.

- 11.0.132.1 — If the root partition (/) is full `/usr/etc/vipw` or `/bin/passwd` will "remove" `/etc/passwd`. Execution is now aborted if this is the case.
- 11.0.134 — When `/etc/cron` performs a re-scan for changed password entries and crontabs, it stats home directories. This can cause unnecessary automounter activity on systems which use it.
- 11.0.136.2 — `/usr/bin/touch` would silently exit when file names which looked like a date-time argument were specified.
- 11.0.137 — `/bin/du` can now handle more than 1,000 different files with multiple hard links.
- 11.0.139.1 — `/usr/convex/op` will now zero out every other entry in group vector.
- 11.0.141 — The spacing between output fields in `/usr/ucb/vmstat` has been modified to guarantee a space between all fields regardless of size.
- 11.0.142 — Network status is not supported in `/etc/stat`.
- 11.0.143 — `/bin/du` goes into an infinite loop traversing very large directories.
- 11.0.144 — `csh` now sets permissions 0600 on `/etc/shadow.{`*dir,pag*`}` instead of 0644, eliminating a security problem.
- 11.0.145 — Aborting an invocation of `/usr/convex/tellcron` can result in `cron` aborting shortly afterwards.
- 11.0.146.1 — `/etc/rdump` and `/etc/rrestore` now support unattended tape switching by using the 'S' or 'U' options.
- 11.0.147.1 — `/etc/xdump` improvements: `/etc/dumpdates` now works with ZIC. `xdump` will now dump a 2 GB file system that's built using 8 K blocks. The software timeout has been increased for slower tape devices.
- 11.0.148 — The `/usr/bin/find` option **-fsonly** was added by this patch to prevent `find` from descending into file systems other than the specified types. It is otherwise identical to **-fstype**.

Internet Services

- 11.0.134 — `/usr/ucb/ftp` incorrectly checked the verbose flag, printing a verbose message in non-verbose mode, and vice-versa. The `/usr/etc/in.ftpd` code has been modified to recognize when the PASS command is being logged, and now suppresses the password text with "`<not shown>`".
- 11.0.136 — `/usr/etc/named` can experience integer arithmetic overflow which results in incorrect behavior.
- 11.0.137 — `/usr/ucb/netstat -s` reports negative byte totals in TCP section of output on heavily-used networks.

NFS Utilities

- 11.0.129 — `/usr/etc/rpc.yppasswdd` improperly parses a pwrestrict entry which has an empty date field (i.e. :0,0:).
- 11.0.130 — `/usr/bin/rpcgen` reports an incorrect type definition.

Programming Tools

- 11.0.128.1 — `/bin/make` will now import environment variables as initial values for variable names. Environment variables will override make's internal variables.
- 11.0.129 — `/usr/lib/libcurses.a` - Format strings passed to the functions `mvprintw` and `mvwprintw` were corrupted.

Managing ConvexOS: Configuration Guide

2

Changes and enhancements available in ConvexOS 11.5.1 that fall under sections of the *Managing ConvexOS: Configuration Guide* are described in detail in this chapter.

Affected sections from *Managing ConvexOS: Configuration Guide*:

- Chapter 2: Adding devices
- Chapter 4: Scheduling file system backups
- Chapter 5: Setting up the line printer system
- Chapter 7: Setting up user accounts
- Chapter 15: Customizing kernel boot-time parameters
- Chapter 16: Generating system images
- Appendix C: Controller, device, and driver `/ioconfig` designations

Adding devices

This section lists supported devices and describes the procedure for creating new device files not found in *Managing ConvexOS: Configuration Guide* , Chapter 2.

Supported devices

Several SCSI devices have been incorporated into ConvexOS 11.5.1. Corresponding `/ioconfig` information is shown in Table 4 "New ConvexOS device /ioconfig designations" on page 68.

Disk

Seagate Elite 9 QSC disk device support has been added.

Tape

QSC tape devices added to ConvexOS in 11.5.1:

- HP/Convex C590 IBM Magstar (NTP)
- STK SD-3 (Redwood)
- STK 9490 (Timberline)

VME SCSI tape devices added to ConvexOS in 11.5.1:

- Quantum DLT-2000
- Quantum DLT-4000
- Metrum RSP-2150
- Exabyte 8200, 8500, and 8505

Device files

If you have VME SCSI tape devices, to take advantage of the new features of the `mt` utility, a new device file with the "spt" suffix must be created. See "The mt utility" on page 77 for more information about specific enhancements to `mt`.

For example, a DLT drive may have the existing nodes:

- `/dev/rdlt0`
- `/dev/rdlt0n`
- `/dev/rdlt0i`
- `/dev/rdltni`

Upon execution of "`MAKEDEV dlt0`", the node `/dev/rdlt0spt` would be created. However, a "`file exists`" error message would be displayed for each of the four existing devices.

Scheduling file system backups

There are a number of utilities that you can use to dump file systems to tape: `dump`, `xdump`, and `rdump`. The `dump` and `xdump` utilities back up files from a local machine; `rdump` dumps files over Ethernet. This section describes updates not found in *Managing ConvexOS: Configuration Guide* , Chapter 4.

Unattended dumps

The `rdump` utility is part of the ConvexOS Internet Services, which is an optional product that will not exist on your machine unless you have installed it. The **-S** and **-U** options, which provide the ability to perform unattended dumps and restores, have not been available for the `rdump`/`rrestore` utilities until ConvexOS 11.5.1.

Migrated files

Normally, it is considered an error to dump a file system which is either under event-daemon control or contains migrated data blocks. Two new options are provided with `dump` and `restore` for use on daemon-controlled file systems; (e.g. a file system under FileServ Software control).

m The IMIGRATED bit (0x80) of the inode dmonflags field, indicates that some of a file's data blocks are not disk resident. The **-m** option enables `dump` to handle dmon conditions as specified by the arguments.

The **-m** option requires two hexadecimal arguments, a mask and a list of match patterns. The values are applied to the `dmonflag` field of each inode in the following manner:

```
(dmonflag & Mask) == MatchPattern
```

If the above condition evaluates to true for any **MatchPattern** specified, the file will be dumped as a zero-length file. If the the above evaluates to false, disk resident files will be dumped while migrated files will be dumped as zero-length. This option is mutually exclusive with the **-M** option.

Example syntax (the "0x"prefix is optional):

```
dump 0mE 0x80 0x80 /home
```

M This option is intended for file systems under event-daemon control. It is identical to the **-m** option, except that files which match the test condition are NOT dumped.

Setting up the line printer system

Access control has been added to the configuration of line printers in ConvexOS 11.5.1. This section contains corresponding information not found in *Managing ConvexOS: Configuration Guide* , Chapter 5.

User access

A new field, "ac", has been added to the `printcap` file. The "ac" field contains a comma-separated list of valid userIDs; i.e. "ac=userID,[userID...]". The `lpr` utility will verify that a user has access to the specified or default printer by checking for an entry in the "ac" field in the `/etc/printcap` file. If the "ac" field is left blank, or the user has an entry in the "ac" list, their job is spooled. Else, the job is rejected and an error message is displayed.

Example `/etc/printcap` entry:

```
hp|hpps:\
    :rm=clapple.convex.com:rp=hp:lp=:sd=/usr/spool/lpd/hp:\
    :lf=/usr/adm/lpd-errs:mx#0:ac=smith,jones,paul,tom:
```

Setting up user accounts

Each person wishing to log into the Convex computer must have a user account and belong to a group.

Each user account is identified by its userID (UID). Groups allow two or more accounts to share access to files without granting access to the entire user community. Each user can belong to up to 16 groups. Each group is assigned a group identified called a GID.

This section contains corresponding information not found in *Managing ConvexOS: Configuration Guide* , Chapter 7.

Increased UID and GID range

The range of numbers available for assignment as user and group IDs has increased from 32,767 to 65,000. It is recommended to recompile all utilities that touch UID/GID if using IDs in the range from 32,768 to 65,000.

Customizing kernel boot-time parameters

When you boot your system, the `boot` command reads a file containing parameters that control the way ConvexOS handles CPUs and CCUs at your site. You can change these parameters to optimize performance and the behavior of your system without recompiling the system image.

To set boot-time parameters specific to your site, modify the SPU disk file `/mnt/os/bootcmd.local`.

The SPU `bootcmd.local` file

If spaces are not surrounding the "="sign in each entry of the `bootcmd.local` file, when a new kernel is installed, the tunable values may not be properly interpreted. A lack of spaces has no impact on other operations (such as a reboot).

Example `bootcmd.local` entry:

```
tune cpu max_user_processes = 64
```

A new man page, `tunables(5)`, is provided with ConvexOS 11.5.1 to provide an on-line description of each of the kernel tunables.

New boot-time parameters

The tables below lists parameters new to ConvexOS in 11.5.1.

Table 1 CPU boot-time parameters

Parameter	Definition
`bufcache_size`	Size of the buffer cache in MB, limited by the kernel to no more than four times the size of available physical memory. Default = 1024, min = 64, max = 1024
`duc_enable`	This tunable enables/disables delayed unwire caching for raw disk I/O. Set to 1 to enable delayed unwire caching; set to 0 to disable. Default = 1, min = 0, max = 1.

`explicit_table_size`
`inode_table_size`
`file_table_size`
`ncache_table_size`
`dquot_table_size`
	Allows explicit tuning of table sizes. If `explicit_table_size` is zero (default), the table sizes are automatically based on the value of the `maxusers` parameter. The formulas used to compute the values are:
	`file_table_size` = (9 * `maxusers`) + 356
	`inode_table_size` = ((16 * ((9 * `maxusers`) + 56)) / 10) + 336
	`ncache_table_size` = (`inode_table_size` * 5) / 8
	`dquote_table_size` = ((`maxusers` * 256) / 4) + ((8 * `maxusers`) + 40)
	If `explicit_table_size` is nonzero, the tuned values override the computed values.
	Default = 0, min = 0, max = 1.
	Default = 1024, min = 512, max = 4096.
	Default = 1024, min = 512, max = 4096.
	Default = 1024, min = 512, max = 4096.
	Default = 1024, min = 512, max = 4096.

Table 1 CPU boot-time parameters (Continued)

Parameter	Definition
n_async_proc	Number of asynchronous I/O processes; increase if using more than two applications that do asynchronous I/O. Default = 4, min = 0, max = 128
nfsfastrecv nfsfastsend	Enable in order to bypass the STREAMS stack for NFS. Bypassing the STREAMS stack improves NFS performance. Set to 1 to enable STREAMS bypass; set to 0 to disable. Default = 1, min = 0, max = 1
nfs_ret_mig_stat	Set sticky bit to tell NFS client that a file is migrated. Default = 0, min = 0, max = 1.
nfstsize_loopback nfstsize_hippi nfstsize_fddi nfstsize_eth nfstsize_default	NFS transfer size for reads/writes is now tunable on an interface basis. In 11.0, the transfer size was 8K for all interfaces. nfstsize_default is used when none of the other nfstsize tunables apply. Default = 61440, min = 8192, max = 61440 Default = 61440, min = 8192, max = 61440 Default = 8192, min = 8192, max = 61440 Default = 8192, min = 8192, max = 61440 Default = 8192, min = 8192, max = 61440
tcphippi_default_window fddi_default_window ethernet_default_window	TCP window size is now tunable on an interface basis. In 11.0, the default TCP window size was 61K for all interfaces. Automatic window sizing, controlled by the use of these tunables, eliminates the need for applications to explicitly increase the window size. Default = 2097152, min = 64512, max = 2097152 Default = 64512, min = 31744, max = 262144 Default = 31744, min = 31744, max = 64512
somaxconn	Backlog of listen(). Default = 5, min = 5, max = 20.

Table 1 CPU boot-time parameters (Continued)

Parameter	Definition
`rt_enabled`	For sites with Realtime systems. Contact your Hewlett-Packard, Convex support representative before changing the `rt_enabled` tunable. Default = 0, min = 0, max = 2
`roundrobin_freq`	Sets the number of hardclock ticks between each roundrobin reschedule. Default = 10, min = 1, max = 1000.
`realtime_grpid`	Specifies a special realtime group. Members of this group will be permitted to execute selected realtime system calls that are otherwise restricted to root access. Default = 0, min = 0, max = 32767.
`fast_clock`	Specify hardclock frequency. Increasing this can result in fewer 'connection refused' conditions under a high TCP connection load. Impacts applications such as http servers. Default = 0 (100 Hz), min = 0 (100 Hz), max = 1 (1000 Hz).
`halt_on_panic`	Halt instead of panic. This can be useful in debugging, since more of the machine state is saved. Default = 0, min = 0, max = 1.
`sig_cannot_mask`	Signal handler bit mask. Certain signals (like SIGBUS,SIGSEGV) shouldn't be ignored/masked since they are usually generated by hardware conditions and would immediately be re-asserted on return to user code as this generates a ping-pong effect with the process continuously bouncing between exception and trap. Default = 0x600, min = 0, max = 0x7fffffff.

Table 1 CPU boot-time parameters (Continued)

Parameter	Definition
	`mbs_enabled` enables Enhanced MBS when set to 1 and disables it when set to 0. When disabled, the `xxx_mbs_limit` and `xxx_mbs_drain` tunables are ignored. (Also, when disabled, set `fd_max_recv` and `fd_max_xmit` to their original value of 28.)
`mbs_enabled`	Default = 1, min = 0, max = 1.
	The `xxx_mbs_limit` tunables specify the number of pages to reserve for an alternate MBS memory pool. Each page represents 64 messages with a maximum of 16 pages (or 1023 messages, where 1 is unused). The original MBS pool, which is still used, contains 11 pages or 703 messages. Individual CCUs are disabled and use the original MBS pool when their value is set to 0.
`viop_mbs_limit`	Default =16, min = 0, max = 16.
`idc_mbs_limit`	Default = 4, min = 0, max = 16.
`qsc_mbs_limit`	Default = 16, min = 0, max = 16.
`hippi_mbs_limit`	Default = 16, min = 0, max = 16.
	The `xxx_mbs_drain` tunables specify the number of messages to read from each MBS pool before sequencing to the next pool. This has the effect of increasing a CCU's priority when more messages are read on each pass. It is not recommended to set to zero. The `org_mbs_drain` tunable applies to the original MBS message pool rather than the alternate MBS message pools.
`org_mbs_drain`	Default = 2, min = 0, max = 10.
`viop_mbs_drain`	Default = 2, min = 0, max = 10.
`idc_mbs_drain`	Default = 5, min = 0, max = 10. (See also `du_mbs_limit`)
`qsc_mbs_drain`	Default = 5, min = 0, max = 10. (See also `si_mbs_limit`)
`hippi_mbs_drain`	Default = 5, min = 0, max = 10.
`du_mbs_limit`	Represents the total number of MBS messages available to all IDC-based devices on a system-wide basis. The `idc_mbs_limit` tunable is different in that it represents the total number of MBS messages available for a single IDC CCU. Default = 1023, min = 10, max = 4092.

Table 2 QSC boot-time parameters

Parameter	Definition
qsc_max_io_reqs	Maximum number of outstanding I/O requests per QSC CCU. Default = 400, min = 160, max = 400.
elite9_max_out	Maximum number of outstanding requests per Elite9 disk. Default = 8, min = 1, max = 20.
si_mbs_limit	Represents the total number of MBS messages available to all QSC devices on a system-wide basis. The qsc_mbs_limit tunable is different in that it represents the total number of MBS messages available for a single SCSI CCU. Default = 1023, min = 10, max = 4092.

Table 3: VIOP boot-time parameters

Parameter	Definition
st_3480_synch	Use synchronous data transfers for VME SCSI 3480 drives. Default = 1, min = 0, max = 1.
st_3490_read_retry	Number of retries on read errors for VME SCSI 3490E drives. Default = 64, min = 0, max = 127.
st_3490_synch	Use synchronous data transfers for VME SCSI 3490E drives. Default = 1, min = 0, max = 1.
st_3490_write_retry	Number of retries on write errors for VME SCSI 3490E drives. Default = 64, min = 0, max = 127.
st_dlt_synch	Use synchronous data transfers for VME SCSI DLT drives. Default = 1, min = 0, max = 1.

Table 3: VIOP boot-time parameters

Parameter	Definition
st_exb8500_synch	Use synchronous data transfers for VME SCSI Exabyte 85xx drives. Default = 1, min = 0, max = 1.
st_exb8505_112m_bit	Selects auto-sizing of 112m tape, when tape is not detected as <=54m. This is for VME SCSI Exabyte 85xx drives only. Default = 1, min = 0, max = 1.
st_exb8505_density	Determines density selected for VME SCSI Exabyte 8500/8505 tape drives. The default (st_exb8505_density = 0) Drive Type — rexb0/rexb0n — rexb0i/rexb0ni EXB-8200 (MTD-20A) — 2 GByte — N/A EXB-8500 (MTD-20B) — 2 GByte — 5 GByte EXB-8505 (MTD-20B) — 2 GByte — 5 GByte New Mode (st_exb8505_density = 1) Drive Type — rexb0/rexb0n — rexb0i/rexb0ni EXB-8200 (MTD-20A) — 2 GByte — N/A EXB-8500 (MTD-20B) — 5 GByte — N/A EXB-8505 (MTD-20B) — 5 GByte — 10 GByte Default = 0, min = 0, max = 1.
st_ignore_attach_err	Ignore attach errors for VME SCSI drives, and attach drive anyway. Default = 1, min = 0, max = 1.
st_max_busy_retry	Maximum device busy retries on a VME SCSI device. Default = 30, min = 0, max = 6500.
st_rsp2150_synch	Use synchronous data transfers for VME SCSI Metrum drives. Default = 1, min = 0, max = 1.
st_show_recovered_err	Show recovered errors to the errlog for VME SCSI 3480/3490E drives. Default = 0, min = 0, max = 1.

Table 3: VIOP boot-time parameters

Parameter	Definition
`st_spt_synch`	Use synchronous data transfers for VME SCSI SPT devices. Default = 1, min = 0, max = 1.
`st_trace_pg_cnt`	VME SCSI trace page count in viop 4k pages. Default = 2, min = 0, max = 4.
`ta_viop_prt_mask`	Controls printing of error messages for VME nine-track tape drives. The following bit flags apply to `ta_viop_prt_mask`: 0x0001 /* prt cmd/esb/fsb on err */ 0x0002 /* prt cmd/csb/esb/fsb/mux1-decode*/ 0x0004 /* prt mux[0-3]-hex */ 0x0008 /* prt mux1 velocity errors */ 0x0010 /* prt chain overruns */ 0x0020 /* prt offline/notready messages */ Default = 0xffbf, min = 0x0, max = 0xffff

Generating system images

Two changes in ConvexOS 11.5.1 affect generation of system images. This section contains system image information not found in *Managing ConvexOS: Configuration Guide* , Chapter 16.

System generation configuration file

The system configuration file contains two sections that specify:

- System parameters
- Hardware device types

Several sample configuration files are shipped with ConvexOS 11.5.1:

- REL_C2
- REL_C3
- REL_C4

Default drivers

To reduce the size of the default viop core image, some drivers are not included in the generic sysgen templates. A sysgen is required for support of the following devices:

- Hyperchannel driver
- Eagle ethernet driver (except for the C4600-specific file REL_C4)
- User-written driver test module (UDD)
- FDDI driver

System image size

The `gut` utility is used to reduce the size of the system image, `/vmunix`, by removing text/data/bss sections. To use the `gut` command, instead of copying the system image from the SPU with

```
spu -r /mnt/os/vmunix > /vmunix
```

it would be copied with

```
spu -r /mnt/os/vmunix | gut > /vmunix.
```

In 11.5.1, a symbol hash table is now included to support enhancements to the `pstat` command. Hence, the 11.5.1 `gut` operation results in a "gutted" `/vmunix` that is slightly larger than the one produced using ConvexOS 11.0.

/ioconfig designations

The *Managing ConvexOS: Configuration Guide, Appendix C* contains a table that maps controller, device, and driver names used in the `/ioconfig` file on the SPU to their textual descriptions. This section contains additional information not found in *Managing ConvexOS: Configuration Guide*, Appendix C.

New devices

Support for several new tape devices have been incorporated into ConvexOS 11.5.1, as listed in the following table.

Table 4 New ConvexOS device `/ioconfig` designations

/ioconfig designation	Description
DKD-402	Disk Drive, Seagate Elite 9 Quad SCSI
LAN-205	Interphase VME Eagle Ethernet Ethernet Driver
MTC-202	VME SCSI Ciprico 3576 Differential Fast & Narrow Adapter
MTD-20A	Exabyte 8200, VME SCSI
MTD-20B	Exabyte 8500 and 8505, VME SCSI
MTD-20C	Tape Drive, Quantum DLT-2000, VME SCSI Digital Linear Tape Tape Drive, Quantum DLT-4000, VME SCSI Digital Linear Tape[a]
MTD-22C	Tape Drive, Quantum DLT-2000, VME SCSI Digital Linear Tape, w/compression
MTD-20D	Tape Drive, Fujitsu model MC2488 3490E, VME SCSI
MTD-21D	Tape Drive, Fujitsu model MC2488 3490E, VME SCSI, w/stacker
MTD-22D	Tape Drive, Fujitsu model MC2488 3490E, VME SCSI, w/compression
MTD-23D	Tape Drive, Fujitsu model MC2488 3490E, VME SCSI, w/stacker and compression
MTD-207	Tape Drive, Fujitsu 3480, VME SCSI
MTD-217	Tape Drive, Fujitsu 3480, VME SCSI, w/stacker
MTD-227	Tape Drive, Fujitsu 3480, VME SCSI, w/compression
MTD-237	Tape Drive, Fujitsu 3480, VME SCSI, w/stacker and compression
MTD-208	Tape Drive, AT (Python), VME SCSI
MTD-209	Tape Drive, Metrum RSP-2150, VME SCSI
MTD-228	Tape Drive, Diamond Back DAT, VME SCSI Digital Linear Tape

a. Also requires bootcmd.local entry "`tune viop st_dlt_synch = 0`".

Managing ConvexOS: Operations Guide

3

Changes and enhancements available in ConvexOS 11.5.1 that fall under sections of the *Managing ConvexOS: Operations Guide* are described in detail in this chapter.

Affected chapters from *Managing ConvexOS: Operations Guide*:

- Chapter 1: Monitoring resources
- Chapter 8: Checking the file system
- Chapter 9: Performing crash dumps

In addition, changes that do not fall into the above categories are described in the following sections:

- Utilities
- Console information

Monitoring resources

Many utilities exist to help the administrator monitor the resources configured in the Convex machine. This section describes the monitoring utilities affected by the ConvexOS 11.5.1 release not found in *Managing ConvexOS: Operations Guide*, Chapter 1.

Monitoring current use and limits

A new option, **-w**, has been added to the `quota` and `repquota` utilities that provides wider screen output. This ensures that fields whose values can be large will be separated with white space. This enhancement is useful for shell-script parsing of the command output.

System statistics

The `pstat` utility interprets the contents of certain system tables. If *`corefile`* is given, the tables are sought there, otherwise in `/dev/mem`. The required *`name`* list is taken from `/vmunix` unless *`system`* is specified.

If the *`system`* or *`corefile`* arguments are specified, `pstat` gives up its membership in group kmem, so the user must have read access to the files specified.

Usage:

```
pstat -abcmnquilovpfstILT -d[class] [-Ppid] [-Cname]
      [-S [UID]] [system] [corefile]
```

Several of the `pstat` options are 11.5.1 enhancements. The new options, and the information they provide, are listed below:

-b	Dump all system buffer headers.
-c	Print kernel stack backtrace information for all processes.
-d[*class*]	Display each MBLK which has a DBLK class associated with it. Valid classes are: DB0, DB4, DB16, DB64, DB128, DB256, DB512, DB1024, DB2048, DB4096 DB8k DB64k.
-m	Display DBLK usage.
-n	Prints all STREAM heads information.
-q	Display all STREAM heads and associated STREAM queues.
-u	Display detailed information about all processes and user areas.
-o	Display information and statistics on the original and alternate MBS message pools.
-l	Display information regarding messages in the message log, i.e., the owner of the message, source and destination processors, etc.
-L	Prints the number of messages successfully sent and received via tm_send(). Also, prints the number of messages received having an invalid reference to the Tm_pool message log (Tm_bad_indx), or that refer to an invalid entry in the Tm_pool message (Tm_bad_proc).

-T Prints the number of used and free slots in several system tables. Useful for checking to see how full system tables have become if the system is under a heavy load.

Example `pstat -b` output

```
% pstat -b
buf 1a479b4 nbuf 14630
0x1a479b4: 8192/8192 {0x12932204,0x0,48} ->0x40000000   READ|DONE|HEAD|RECLAIM
0x1a47a58: 8192/8192 {0x12507e04,0x420,10752} ->0x40010000      READ|DONE|HEAD|RECLAIM
0x1a47afc: 8192/8192 {0x129b6004,0x0,0} ->0x40020000    READ|DONE|HEAD|RECLAIM
0x1a47ba0: 8192/8192 {0x1286a004,0x0,160} ->0x40030000  READ|DONE|HEAD|RECLAIM
0x1a47c44: 8192/8192 {0x128f6604,0x0,144} ->0x40040000  WRITE|DONE|RECLAIM
0x1a47ce8: 8192/8192 {0x1244da04,0x700005,2716768} ->0x40050000 WRITE|DONE|RECLAIM
0x1a47d8c: 8192/8192 {0x12923c04,0x0,0} ->0x40060000    READ|DONE|HEAD|RECLAIM
0x1a47e30: 8192/8192 {0x1281f604,0x0,16} ->0x40070000   WRITE|DONE|RECLAIM
0x1a47ed4: 8192/8192 {0x12507e04,0x420,11568} ->0x40080000      READ|DONE|HEAD|RECLAIM
...
```

Example `pstat -c` output

```
% pstat -c
182/296 processes
 1707118   4 0000000b        0 00000000      0 65535 65535     0 -64       0
0       0   c94d000     0     0      0   1c278c   swapout
    <swapout-0> frame pointer 0e19cd78 ctxpgzstk 00000000 propgzstk 00000000
        swtch() from 16edc2 [ap = e19cda8]
        _v_ctxswtch+308() from 98846 [ap = e19ce08]
        _pr_wait+58a() from 9936a [ap = e19ce48]
        _td_wait+7a() from 96e86 [ap = e19ce90]
        _sleep+15a() from 14b5c2 [ap = e19cef0]
        _swapout+164() from 16440 [ap = e19cf48]
        _main+aec() from 1113a [ap = e19cf8c]
        start_jpunix+106() from 80001000
 1707714   4 00000009 1.3e+02 00000000      0   113 65535     0   0       1
1       0   c94f000    18     0     15  1707714   init
    <init-1> frame pointer 0e1abc9c ctxpgzstk 00000000 propgzstk 00000000
        swtch() from 16edc2 [ap = e1abccc]
        _v_ctxswtch+308() from 986b2 [ap = e1abd2c]
...
```

Example `pstat -d[class]` output

```
% pstat -dDB1024
=== mblk : 0x15aee40
Pointers: next 0x15aa1a0 prev 0x0 cont 0x0
Data: read 0x1402000 write 0x140224e
=== dblk : 0x1551570
Pointers: base 0x1402000 limit 0x1402400
Counts: Refs 0x0 type 0x0
%
```

Example `pstat -m` output

```
% pstat -m
     RESOURCE          CONFIG    IN USE     FREE      TOTAL      HIGH      FAILED
                                                      USED      WATER
     dblks(0)             100         0      100    3301616          4           0
     dblks(4)              64         0       64     167209         15           0
     dblks(16)            128         0      128     316379         73           0
     dblks(64)           1536       180     1356   39189276        396           0
     dblks(128)           256        46      210    8112669         54           0
     dblks(256)          1024        20     1004   14996401        321           0
     dblks(512)          1536       103     1433    3219378        385           0
     dblks(1024)          512         0      512     407710        180           0
     dblks(2048)         1280        92     1188   10398489        312           0
     dblks(4096)         1024         0     1024       3162          4           0
     dblks(8192)          512         0      512       1889          3           0
     dblks(65536)         100         7       93     875250         13           0
     mblks               8072       448     7624   88116932        920           0
     queues              4096       596     3500    8832600        792           0
     streams             1024       149      875    2208150        198           0
     event cells          256         0      256          0          0           0
     timeout cells         16         5       11   22004603          6           0
%
```

Example `pstat -n` output

```
% pstat -n | head
    Stream     Major  Minor      Pgrp    Flags   Linked
  160D6C8        102      1         0      101
  160D780        106     52         0      100
  160D838        106     58         0      100
  160BDA0        106     33         0      100
  160C360        106    117         0      101
  160DBD0        106    111         0      100
  160BBD4        106    108         0      100
  160C868        106     59         0      101
  160CCB8        106     48         0      100
%
```

Example `pstat -q` output

```
% pstat -q | head
    Stream     Major  Minor      Pgrp    Flags   Linked
Stream head:  160D6C8    102:1
Down Stream:
    Queue          1st       Last     q_ptr      Count       Flags
  15F1630            0          0   160D6C8          0        1020
  15F15A0            0          0    2478A4          0        1040
Up Stream:
    Queue          1st       Last     q_ptr      Count       Flags
%
```

Example `pstat -u` output

```
% pstat -u
176/296 processes
 1707118  4 0000000b        0 00000000      0 65535 65535     0 -64       0
0       0  c94d000     0     0      0   1c278c    swapout
comm      swapout
rtn pc    _pr_wait+58e
reschfp   0e19cd78
ctxpgzstk   00000000
propgzstk   00000000
sysc_ctx 0e19cf90
arg       0 0 0 0 0
t_qsav     9ef64,   86000080,    e19ce48,    e19ce38,    e19cdf8
scjsav          0,          0,          0,          0,          0
syncflag          0
error 74
uids     0,0
procp    1707118
ap       0
r_val?   0.1x 0.1x
cdir rdir 16a6aa0 0
file [index       ofile      flags]
 sigs    0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
code     0
uctxjmp 0
...
```

Example `pstat -o` output:

```
% pstat -o
MBS      CCU    CCUID    PAGES/MSGS   DRAIN   IN_USE   FREE      TOTAL   HIGH
POOL                                                              MSGS   WATER
  0     orig        -       11/703       1       57    646     512791       4
  1     VIOP        1      16/1023       1       65    958    9612581       6
  2      QSC        0      16/1023       4        1   1022    1294998       4
  3      IDC        2       4/255        4        1    254    4031200       6
```

Example `pstat -l` output

```
% pstat -l
 POOL:INDX      FUNC    TIME     SLPC FLGS OWN SRC DST     SRCID    DSTID      CMD
    0: 666     12ad4    2145        0   20   1 cpu   1       396   13000e     2005
    0: 660     12ad4    1064        0   20   1 cpu   1       398   13000e     2005
    0: 669     12ad4    2325        0   20   1 cpu   1       39b   13000e     2005
    1:1010     44a1e       0        0   00   1 cpu   1       39c    20005   48335e
    1:1013     44a1e       0        0   00   1 cpu   1       3a1    20005   48335f
    0: 649     12ad4      97        0   20   1 cpu   1       3a3   13000e     2005
    1:1009     44a1e       0        0   00   1 cpu   1       3ab    20005   48335b
    0: 672     12ad4    3046        0   20   1 cpu   1       3b1   13000e     2005
    0: 650     12ad4      97        0   20   1 cpu   1       3b4   13000e     2005
    0: 656     12ad4     163        0   20   1 cpu   1       3b7   13000e     2005
 POOL:INDX      FUNC    TIME     SLPC FLGS OWN SRC DST     SRCID    DSTID      CMD
    0: 673     12ad4    3226        0   20   1 cpu   1       3b8   13000e     2005
    0: 654     12ad4      97        0   20   1 cpu   1       3bc   13000e     2005
    0: 646     12ad4      97        0   22   1 cpu   1       3c2   13000e     2005
    1: 956    17c1b4       0        0   01 cpu   1 cpu    10001      3c3        5
    0: 661     12ad4     884        0   20   1 cpu   1       3c4   13000e     2005
    0: 663     12ad4    1424        0   20   1 cpu   1       3c7   13000e     2005
    0: 671     12ad4    2686        0   20   1 cpu   1       3c9   13000e     2005
    1:1011     44a1e       0        0   00   1 cpu   1       3ab    20005   483362
    0: 652     12ad4      97        0   20   1 cpu   1       3d0   13000e     2005
    0: 659     12ad4     703        0   20   1 cpu   1       3d1   13000e     2005
 POOL:INDX      FUNC    TIME     SLPC FLGS OWN SRC DST     SRCID    DSTID      CMD
    0: 647     12ad4      97        0   20   1 cpu   1       3d4   13000e     2005
    0: 667     12ad4    1965        0   20   1 cpu   1       3d5   13000e     2005
    0: 662     12ad4    1244        0   20   1 cpu   1       3d9   13000e     2005
    0: 657     12ad4     343        0   20   1 cpu   1       3de   13000e     2005
    0: 670     12ad4    2866        0   20   1 cpu   1       3e0   13000e     2005
    1:1014     44a1e       0        0   00   1 cpu   1       3f5    20005   483360
    0: 665     12ad4    1785        0   20   1 cpu   1       3e9   13000e     2005
    0: 651     12ad4      97        0   20   1 cpu   1       3ea   13000e     2005
    1:1012     44a1e       0        0   00   1 cpu   1       3cb    20005   483363
    0: 668     12ad4    2505        0   20   1 cpu   1       3f3   13000e     2005
 POOL:INDX      FUNC    TIME     SLPC FLGS OWN SRC DST     SRCID    DSTID      CMD
    0: 653     12ad4      97        0   20   1 cpu   1       3f4   13000e     2005
    0: 648     12ad4      97        0   20   1 cpu   1       3f8   13000e     2005
    0: 655     12ad4      97        0   20   1 cpu   1       3fa   13000e     2005
    0: 664     12ad4    1604        0   20   1 cpu   1       3fe   13000e     2005
    0: 658     12ad4     523        0   20   1 cpu   1       400   13000e     2005
...
```

Example `pstat -L` output

```
% pstat -L
Tm_bad_indx: 0
Tm_bad_proc: 0
Tm_send_cnt: 15457727
Tm_recv_cnt: 15457804
%
```

Example `pstat -T` output

```
% pstat -T
 400/ 886 files          ( 84k/186k)
 615/ 644 inodes         (189k/198k)
 175/ 296 processes      (261k/442k)
   0/2344 quota          (  0k/146k)
  31/1652 M swap
%
```

The `sysstat` utility

`sysstat` is a new utility in ConvexOS 11.5.1 that reports disk, tape and CCU performance to `stdout`. `sysstat` produces information similar to that of the `syspic` utility to `stdout` on an interval basis, much like the `vmstat` utility.

Usage:

```
sysstat [-w windowname] [-l ccu#s] [-i interval] [-c count] [-u unixbin]
```

Example `sysstat`:

```
% sysstat -w "CCU Busy" -l 20 21 22 23 36 37 38 39 -i 10
     CCU Busy (%)
     ccu20 ccu21 ccu22 ccu23 ccu36 ccu37 ccu38 ccu39
         3     0    12     1     1     3     0     1
         2     0    10     2     0     2     0     0
         3     0    20     0     0     0     0     0
         5     0    13     6     7    11     4     4
```

File system checking

Chapter 2 of the *Managing ConvexOS: Operations Guide* presents an overview of the ConvexOS file system and use of the file system check (`fsck`) utility to resolve problems. This section describes additions to the operating system not found in *Managing ConvexOS: Operations Guide,* Chapter 2.

Internal view of the file system

The `fsck` utility is used to repair ConvexOS file systems. Knowing the alternate superblock location information can be critical for recovery purposes.

The superblock

A file system is described by its superblock. The superblock is built when the file system is created and never changes. Alternate copies of the superblock are written to the device and are listed upon execution of `newfs` or `newst`. This information allows you to run `fsck` with an alternate super-block.

In 11.0, superblock information had to be manually captured and saved by the administrator.

In 11.5.1, each time the `mkfs` utility is executed (via `newfs` or `newst`), the super-block location information is automatically appended to the file `/etc/sbtab`. If the `/etc/sbtab` file is removed, the output will not be captured.

Figure 7 Example `/etc/sbtab` file:

```
# Super-block log
# The mkfs utility will append its output to this file; useful
# when running fsck with an alternate super-block.
#
/dev/rst0: super-block backups (for fsck -b#) at:
     32, 40544, 81056,

/dev/rdu2e: super-block backups (for fsck -b#) at:
     32, 40544, 81056, 121568, 162080, 202592, 243104, 283616,
     322592, 363104, 403616, 444128, 484640, 525152,

/dev/rdu2d: super-block backups (for fsck -b#) at:
     64, 40576, 81088,
```

Performing crash dumps

The *Managing ConvexOS: Operations Guide* Chapter 9 lists the specific steps to take a crash dump. This section describes additions to the crash dump procedure not found in *Managing ConvexOS: Operations Guide*, Chapter 9.

Verification of `osclean` execution

The `crashdump` utility now attempts to verify that the administrator has run `osclean`. If `osclean` has not been executed, the `crashdump` utility terminates.

Utilities

A summary of the utilities improved in ConvexOS 11.5.1, but not documented elsewhere in this addendum, are described in this section.

Enhanced utilities

Enhanced utilities in ConvexOS 11.5.1 include:

- `mt`
- `ls`
- `sed`
- `find`
- `make`
- `df`

The `mt` utility

The `mt` utility now supports the SCSI bus reset command for adapter recovery. New commands include:

- `eod:` skip to SCSI end of data
- `reset:` reset SCSI bus

The `mt reset` command is provided to recover tape devices that would otherwise require a reboot. The `reset` command is only available when `mt` is run as the superuser, and is only supported by VME SCSI devices. It must be issued to a device on the bus that is NOT currently opened, otherwise, the command will fail with EBUSY.

The `reset` command causes a SCSI bus reset for all devices that share a bus with the specified device. A reset aborts any current and future commands to these devices until the device files have been closed and reopened.

Caution

Extreme caution must be used when using this command, as any data in write buffers for devices on that SCSI bus will be lost, even though a previous write command may have returned successfully. Also, end-of-file tape marks may have not been written to signify logical end of data.

In the following example, executing the command "`mt -f /dev/tc0spt reset`" would cause dat0, tc0, and tc1 to be reset. Any pending commands would be lost and the drives reset. The devices tc2 and tc3 are unaffected, as they are on a separate adapter (SCSI Bus) even though on the same VME Bus.

```
viop 0
  vme 0
    ctlr MTC-202 csr 0xee00 int 2
        unit 0 subunit 0 type MTD-208  # dat0
        unit 5 subunit 0 type MTD-207  # tc0
        unit 5 subunit 1 type MTD-207  # tc1
    ctlr MTC-202 csr 0xec00 int 3
        unit 5 subunit 0 type MTD-207  # tc2
        unit 5 subunit 1 type MTD-207  # tc3
```

The `ls` utility

New `ls -e` option prints file times in expanded format. This option is particularly useful when viewing files over 6 months old, as it does not truncate the time in favor of displaying the year.

Example 11.0 long listing:

```
% ls -l /etc/ac /etc/fstab
-rw-r--r--  1 root        10363 Jul 13 00:02 /etc/ac
-rw-r--r--  1 root        11167 Sep 14  1993 /etc/fstab
```

Example 11.5.1 long listing, with the **-e** option:

```
% ls -el /etc/ac /etc/fstab
-rw-r--r--  1 root        10363 Sat Jul 13 00:02:02 1996 /etc/ac
-rw-r--r--  1 root        11167 Tue Sep 14 16:52:03 1993 /etc/fstab
```

The `sed` utility

In previous releases, some `sed` commands only accepted "/" as the delimiter for a regular expression. This made it difficult to create a regular expression that contained a "/".

In 11.5.1, arbitrary separation characters are allowed to delimit regular expressions for all `sed` operations.

Format of a regular expression in 11.0:

/regular expression/

Format of a regular expression in 11.5.1 with a delimiter of "?":

\?regular expression?

Instead of always having the next character be the delimiter, any character following the "\" can be designated.

The `find` utility

A new option, `fsonly`, to the `find` command allows you to limit the search to the type of file system specified, either `nfs` or `4.2`. This functionality was possible in ConvexOS 11.0, using the following, more complicated syntax:

```
\( \! -fstype 4.2 -prune \) -o \
```

Example ConvexOS 11.5.1 `find` command:

```
% find /home -fsonly 4.2 -name foobar -print
```

The `make` utility

An enhancement has been made to the `make` utility to allow `make` to import environment variables as if they were `Makefile` variables.

For instance, the following command line operation:

```
setenv SRCDIR /usr/contrib/src
make
```

has the same effect as the following entry in the Makefile:

```
SRCDIR=/usr/contrib/src
```

Support for the `timed` utility

ConvexOS 11.5.1 provides raw socket emulation, which is required by the `timed` utility.

The `timed` utility is the time server daemon and is normally invoked at boot time from the `rc` file. It synchronizes the host's time with the time of other machines in a local area network running `timed`. These time servers will slow down the clocks of some machines and speed up the clocks of others to bring them to the average network time. The average network time is computed from measurements of clock differences using the ICMP timestamp request message.

The `df` utility

The `df` utility reports the amount of free disk available per file system. A new option to the `df` utility, **-I**, limits the report output to NFS mounted file systems and provides the actual read and write transfer size for each.

Example: `df -I` command:

```
% df -I
Filesystem              rsize/wsize   Mounted on
  systhost:/tst/syssw   8192/8192     /rmt/systhost/tstrel/syssw
  csthost:/tst/cseries  8192/8192     /rmt/crypto/tstrel/cseries
  localhost:/work       32768/32768   /tmp_mnt
%
```

Convex tape system

The Convex Tape System was not shipped with ConvexOS 11.0, but is now bundled with the 11.5.1 release. Also, improvements to the reliability of the tape system have been made. ConvexOS 11.5.1 `tpdaemon` uses ACS 4.x libraries to communicate with the STK ACSLS software. In addition to improved reliability, these improvements makes it easier to provide compatibility with STK software upgrades.

New utility

The new `backtrace` utility displays a symbolic backtrace (kernel and user) for a sleeping process, plus the following information:

- System memory statistics
- System tables (addresses and sizes)
- Target process' memory
- Miscellaneous process information, such as amount of time asleep
- Comprehensive signal handling information
- Process sleep address and associated information
- Kernel stack signature and user backtrace

The default is to display all information possible about a process. Several options are provided to limit the display.

Usage:

```
backtrace [-kpsu] [-a ARCH] [-D FLAGS] PID [-A ADB] [-P PSTAT] [object] [core]
 Where:
    -a ARCH   : force OS/Architecture to ARCH (example: 11.5:C38)
    -k        : display the kernel stack backtrace
    -p        : display process information
    -s        : display system information
    -u        : display the user stack backtrace
    -A ADB    : specify alternate ADB (default: /bin/adb) environ: $ADB
    -D FLAGS  : display debug info specified by bitflags (bit-ORed)
    -P PSTAT  : specify alternate PSTAT (default: /etc/pstat) environ: $PSTAT
    object    : namelist source (default: /vmunix)
    core      : corefile (default: /dev/mem)
```

Example full `backtrace` output:

```
% backtrace 246
System S/N: 8737, Architecture Class: 2 (C2), OS: 11.5
Memory:
            Kernel    Kernel     Kernal VM
  Physical    Base    w/tables   Available    Freemem  CPU(s)
    128M      2.8M     35.4M      100.5M       14.3M     2
System Tables:
  002dc000 - 0152dc5f | cnt: 1      | strbase (streams tables)
  015c3198 - 0164da87 | cnt: 1796   | inode
  0164da88 - 016dd91f | cnt: 2729   | file
  016dd920 - 018cb47f | cnt: 1320   | proc
  01ed6280 - 01ee7e7f | cnt: 128    | vspace
  01f355e0 - 01f49ddf | cnt: 512    | stbuf
  0226def4 - 023612bf | cnt: 6075   | buf
  1615a000 - 163d9fdf | cnt: 81919  | core_map
Process Memory:
  80001000 - 80044fff | size: 0.26M   | text
  80055000 - 800f5fff | size: 0.62M   | data/bss
  ffff9000 - ffffafff | size: 0.00M   | stack
Miscellaneous:
  Pid..  slp-time  loadtime  procp.....  upage.....  name
  246    00:00:32  06:56:59  0x016eda74  0x0c9b7000  cron
Signal disposition:
  Pending: None
   Masked: None
  Ignored: <PIPE-13,TTIN-21,TTOU-22>
   Caught: <HUP-1,INT-2,QUIT-3,ALRM-14,TERM-15,XCPU-24,XFSZ-25,VTALRM-26,PROF-27>
   Active: None
Signal vectors:
  HUP-1     - 8000923c (death)
  INT-2     - 8000923c (death)
  QUIT-3    - 8000923c (death)
  ALRM-14   - 8000923c (death)
  TERM-15   - 8000923c (death)
  XCPU-24   - 8000923c (death)
  XFSZ-25   - 8000923c (death)
  VTALRM-26 - 8000923c (death)
  PROF-27   - 8000923c (death)
Sleep Info:
  Addr: selwait, Pri: 26 (PPIPE) [interruptable by signals]
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=- Kernel Stack -=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
  swtch() from v_ctxswtch+0x30c [ap = e29bcc4]
  v_ctxswtch() from pr_wait+0x3fa [ap = e29bd24]
  pr_wait() from td_wait+0x7e [ap = e29bd64]
  td_wait(1a) from pend+0x184 [ap = e29bdac]
  pend(25ce40,25ce20,1a,a1a62) from select+0x684 [ap = e29bedc]
  select+0x684(192be88) from syscall+0x570 [ap = e29bf54]
  syscall(5d,80043114) from C2_user_syscall+0x160 [ap = e29bf88]
** Stack Signature: 1177b570 [select]
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-  User Stack  -=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
  cerror(5,ffffce30,0,0,ffffce28) from __ap$select+0xe [ap = ffffce04]
  select(5,ffffce30,0,0,ffffce28) from uc_sleep+0x8c [ap = ffffce04]
  uc_sleep(3c) from 0x800060dc [ap = ffffce74]
  main(1,ffffcf90,ffffcf98) from aligned+0xa [ap = ffffcf84]
```

Console information

Numerous corrections and enhancements to `prtlog` have been made to improve reliability and provide additional information in the `/mnt/errlog` file.

prtlog enhancements

Some of the behavioral changes in the `prtlog` logging for ConvexOS 11.5.1 are listed below:

- A message is logged as to the known cause of `prtlog` termination.
- A large buffer is used to gather write data to minimize time spent doing disk access.
- Race conditions have been eliminated.
- Every 10th occurrence of a 'repeat' message causes the original error message to be reprinted. This is to prevent having to search far back through `/mnt/errlog`. Example:

```
[CPU01@11:41:52] /11.0usr: file system full
[CPU00@11:43:52] {11:41:52} repeated 3 times (0:02:00 elapsed)
[CPU00@11:45:52] {11:41:52} repeated 5 times (0:04:00 elapsed)
[CPU01@11:47:52] {11:41:52} repeated 7 times (0:06:00 elapsed)
[CPU00@11:49:52] {11:41:52} repeated 9 times (0:08:00 elapsed)
[CPU01@11:51:52] {11:41:52} repeated 10 times (0:10:00 elapsed)
[CPU01@11:53:52] {11:41:52} repeated 12 times (0:12:00 elapsed)
[CPU00@11:55:52] {11:41:52} repeated 13 times (0:14:00 elapsed)
[SPU  @11:57:36] <Thu Sep  5 1996>
[CPU01@11:57:52] {11:41:52} repeated 14 times (0:16:00 elapsed)
[CPU00@11:59:52] {11:41:52} repeated 15 times (0:18:00 elapsed)
[CPU00@12:01:52] {11:41:52} /11.0usr: file system full
[CPU00@12:01:52] {11:41:52} repeated 17 times (0:20:00 elapsed)
[CPU01@12:03:52] {11:41:52} repeated 18 times (0:22:00 elapsed)
```

- If a repeated message doesn't occur after 6 minutes, the repeat mode is cleared.
- A 'repeat' message is logged after 2 minutes of active repeats. This was added to make it obvious there is a continuous flow of the same error message.
- When `prtlog` exits, all partial messages and repeat counts are flushed to the log file. This will help in situations where the system crashes before the entire message is sent.
- If an incoming message isn't completed within 30 seconds, it will be flushed with an 'incomplete' message attached.
- Separate console logger (file sniffer) is used to help reduce memory requirements.

- When SIGHUP is received, the log file is closed and reopened. If the size of the file is in excess of 250,000 bytes and a back up log file does not exist, the log file will be renamed and a new one started. The `printf` window is scanned with each SIGHUP. SIGHUP is also sent to the `sniff` utility so `sniff` will close and open the file too.

  Example log file entry when a SIGHUP is sent to the `prtlog` program:

```
[SPU  @11:57:36] Errlog responding to SIGHUP.  Closing errlog.
[SPU  @11:57:36] Errlog responding to SIGHUP.  Opening errlog.
[SPU  @11:57:36] <Thu Sep  5 1996>
```

- No more spurious INT10 error messages.

System panic

New information is now printed to the system console and written to the SPU file `/mnt/errlog` if a system panic occurs. The output includes a backtrace with arguments and offsets in the head of the panic. Also printed is a signature value, which can be used to quickly identify panics that have an identical subroutine execution path. This information can be very useful in diagnosing problems.

Sample output:

```
[CPU01@15:51:13] Panic backtrace:
[CPU01@15:51:13] _Panic(78cc2,117114,26cc,1199f3)
[CPU01@15:51:13] _blkfree+0x5f0(1845920,1a83,2000)
[CPU01@15:51:13] _realloccg+0x514(1845920,cd093,cd020,2000,4000)
[CPU01@15:51:13] _bmap+0x844(1845920,3,0,4000,0,bea9b28)
[CPU01@15:51:13] _rwip+0x416(1845920,bea9d3c,1,0)
[CPU01@15:51:13] _ufs_rdwr+0x12c(1845928,bea9d3c,1,0,1ef6274)
[CPU01@15:51:13] _rfs_write+0x4a4(fe3be80,fd53200,fdc6280,bea9e60)
[CPU01@15:51:13] _rfs_dispatch+0x476(bea9e60,fe21100)
[CPU01@15:51:13] _svc_getreq+0x30c(fe21100)
[CPU01@15:51:13] _svc_run+0x78(fe21100)
[CPU01@15:51:13] _nfs_svc+0x342(1ef61dc)
[CPU01@15:51:13] _syscall+0x512(9b,800236fc)
[CPU01@15:51:13] _C2_user_syscall+0x160(...)
[CPU01@15:51:13]         ***  PANIC SIGNATURE: d7fd64ab  ***
[CPU01@15:51:13]       ConvexOS: FATAL ERROR: (ufs,9932) free: freeing free frag
[CPU01@15:51:13]           sp:     0bea9988        a1:     0bea995c
[CPU01@15:51:14]           a2:     00000000        a3:     7f3600dd
[CPU01@15:51:14]           a4:     fffffff7        a5:     001199da
[CPU01@15:51:14]           ap:     0bea999c        fp:     0bea9988
[CPU01@15:51:14]           s0: 0000000000000000    s1: 0000000000000021
[CPU01@15:51:14]           s2: 000000000000df9d    s3: 00000000000000ff
[CPU01@15:51:14]           s4: 00000000000000ff    s5: 000000000000000a
[CPU01@15:51:14]           s6: 0000000000000001    s7: 0000000000000000
[CPU01@15:51:14]           int. mask: 000000ff
[CPU01@15:51:15] syncing disks...
```

Managing Internet Services and NFS

4

Changes and enhancements available in ConvexOS 11.5.1 that fall under sections of the *Managing Internet Services and NFS* guide are described in this chapter.

Affected chapters from *Managing Internet Services and NFS*:

- Chapter 10: Setting up the host name database
- Chapter 13: Configuring inetd
- Chapter 15: Configuring the name server
- Part 5: Testing and troubleshooting the network
- Chapter 28: Using ping
- Chapter 32: Using SNMP
- Chapter 33: Using strstat

In addition, changes that do not fall into the above categories are described in the following section:

- FTP

Setting up the host name database

This section describes host name database setup information not found in *Managing Internet Services and NFS* Chapter 10.

The mountd utility

The `/etc/hosts` file contains the database of host names and addresses for all hosts on your local network, as well as hosts on networks interconnected with yours. When using `/etc/hosts`, it is common to specify short names in the `/etc/exports` file. However, using DNS requires the fully qualified host names.

Using the **-e** option with the `mountd` utility causes `mountd` to expand all host names specified as mount options to fully qualified domain names.

Configuring inetd

This section describes `inetd` information not found in *Managing Internet Services and NFS* Chapter 13.

Denial-of-service protection

A new option, **-s**, has been added to the `inetd` service. The **-s** option specifies a semi-secure mode that prevents the built-in services (`echo`, `time`, `daytime`, `discard` and `chargen`) from accepting connections from ports less than 1024.

This option attempts to protect against a common denial-of-service attack that spoofs a connection from one machine's chargen port to another machine's echo port, or similar. This solution does not guarantee protection against the attack, but does decrease the likelihood of it happening.

Configuring the name server

This section describes name server configuration formation not found in *Managing Internet Services and NFS* Chapter 15.

A name server is a network service that enables clients to name resources or hosts and share this information with other hosts in the network. This is a distributed database system for hosts in a computer network.

Control file descriptions

This section of the *Managing Internet Services and NFS* guide contains detailed descriptions and formats for files used to control the execution of the name server.

The existence of the `use_nameserver` file on a system activates the name server resolver for that system. Create `/etc/use_nameserver` by using the `touch` command.

To set up a host that will use a remote server instead of a local server to answer queries, create the file, `/etc/resolv.conf`. This file specifies the name servers on the network that should be sent queries.

New 'search' directive

A new directive, "search", has been added to the `/etc/resolv.conf` file. The `search` directive allows you to specify multiple domains to search when attempting to resolve a short (non-fully-qualified) hostname. Do not specify domains

for which your site has slow DNS resolution. A slow-to-resolve domain in the search list can reduce resolution time for all hosts (local and remote).

The maximum number of searchable domains is six. At the present time, only ConvexOS 11.5.1 utilities can take advantage of the "search" directive.

In the following example `/etc/resolv.conf` file, if a user attempts to `telnet` to "baz", the resolver will perform a lookup in the following order, until a matching address is found:

- baz.foo.com
- baz.bar.com
- baz.ms.com

Example `/etc/resolv.conf` file:

```
domain foo.com
search bar.com
search ms.com
nameserver 111.134.70.1
nameserver 111.134.64.1
nameserver 111.134.65.72
```

Testing and troubleshooting the network

Some basic procedures for identifying network problems are described in Part 5 of the *Managing Internet Services and NFS* guide. This section describes related changes in ConvexOS 11.5.1.

The `tcpdump` utility

The `tcpdump` utility allows the super-user to listen in on the local network to view all network traffic on the wire. For ethernet, all traffic is visible. For FDDI and HiPPI, only traffic to or from the host from which the command is executed.

The `tcpdump` utility is useful in debugging network problems because it lets you see the exact packets as they appear on the wire. You can see if there are problems like duplicates, bad addresses, malformed packets, or even bad data from the application level.

Example output:

```
% tcpdump
tcpdump: listening on eth0
235144 foo.ms.com.telnet > bar.ms.com.43601: P 1:4(3) ack 0 win 31744 [tos 0x10]
082101 foo.ms.com.4711 > acc.ms.com.domain: 1+ PTR? 5.74.168.130.in-addr.arpa. (43)
008893 acc.ms.com.domain > foo.ms.com.4711: 1* 1/0/0 PTR foo.ms.com. (75)
008292 acc.ms.com.domain > foo.ms.com.4712: 2* 1/0/0 PTR bar.ms.com. (74)
046028 foo.ms.com.4713 > acc.ms.com.domain: 3+ PTR? 1.70.168.130.in-addr.arpa. (43)
008339 acc.ms.com.domain > foo.ms.com.4713: 3* 1/0/0 PTR acc.ms.com. (75)
0 packets received by filter
0 packets dropped by kernel
```

Using ping

The simplest way to test the network is to use `ping`. `ping` tests connections with other hosts by sending them a particular type of datagram.

This section describes information about the `ping` utility not found in *Managing Internet Services and NFS* Chapter 28.

Broadcast addresses

Enhancements to the `ping` utility in ConvexOS 11.5.1 provide the ability to ping a broadcast address. This allows you to get responses from all hosts on the subnet that honor broadcast packets. One use of this feature is caching the hosts' physical address information in the system's ARP table.

Flood option

The -f option has been added to allow the superuser to flood a host with packets. This can be helpful in testing heavy network loads or troubleshooting network problems.

Example default `ping -f` of 100 packets:

```
# ping -f convex.convex.com
PING convex [130.168.1.1]: 56 data bytes
Flooding: 100 64-byte packets.
----convex PING Statistics----
100 packets transmitted, 100 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 7/92/189
```

Example `ping -f` with 400 packets specified:

```
# ping -f localhost 64 400
PING localhost [127.0.0.1]: 56 data bytes
Flooding: 400 64-byte packets.
----localhost PING Statistics----
400 packets transmitted, 400 packets received, 0% packet loss
round-trip (ms)  min/avg/max = 1/1/4
```

Using SNMP

SNMP is the "Simple Network Management Protocol". It is a low-level protocol upon which most network management software is built. The `snmpd` daemon is responsible for maintaining a standardized set of variables, the "Management Information Base", or MIB, that reflect the current system state.

This section describes `SNMP` information not found in *Managing Internet Services and NFS* Chapter 32.

MIB configuration

In ConvexOS 11.5.1, a new file, `/usr/etc/snmpd.conf`, is provided to allow the administrator to define certain MIB variables. The following values can be defined in `/usr/etc/snmp.conf`:

- system.sysContact
- system.sysLocation
- system.sysDescr

Example `/usr/etc/snmpd.conf` file:

```
contact:      John Doe
location:     Tanelorn
description:  ConvexOS C-Series
```

Using strstat

The `strstat` utility displays STREAMS statistics. The amount of system resources used by STREAMS are monitored with `strstat`. STREAMS resources are allocated at boot-time and can be controlled by modifying the boot-time parameters associated with that resource.

This section describes `strstat` utility information not found in *Managing Internet Services and NFS* Chapter 33.

Troubleshooting resource problems

Failures illuminated by `strstat` may indicate a need to change boot-time parameters. Case scenario:

```
% strstat
RESOURCE          CONFIG    IN USE      FREE       TOTAL      HIGH      FAILED
                                                    USED     WATER
dblks(0)             100         0       100     1017239         4           0
dblks(4)              64         0        64      116583        15           0
dblks(16)            128         0       128      212669        73           0
dblks(64)           1536      1530         6    16477237      1533         100
dblks(128)           256       256         0     2012715       256       20000
dblks(256)          1024      1014        10    10081475      1014          10
dblks(512)          1536       124      1412     2094123       385           0
dblks(1024)          512       500        12      241534       500          20
```

A good (but incorrect) guess would be that all failed STREAMS classes need to be tuned up. For the above example, that would include the STREAMS classes dblks(64) , dblks(128), dblks(256), and dblks(1024).

However, DBLK allocation uses a roll over method, i.e., if a smaller sized dblk is exhausted, it will roll over to the next available larger dblks. Therefore, dblk(128) could be a victim of dblk(64) running out.

To provide more definitive information for tuning dblks, `strstat` has a new flag, **-p**, that does not count rollovers in the requested statistics. If MAX is over the configured value , it's value should be increased by modifying the appropriate boot-time parameter. In the example below, only the parameters `str_dblk_64` and `str_dblk_256` need to be increased.

```
% strstat -p
  DBLKSIZE          REQUESTED          CONFIG          TUNE
                    NOW       MAX
dblks(0)              0         4         100          NoChange
dblks(4)              0        15          64          NoChange
dblks(16)             0        73         128          NoChange
dblks(64)          1530      1700        1536          UP
dblks(128)           15        40         256          NoChange
dblks(256)          168      1054        1024          UP
dblks(512)          124       480        1536          NoChange
dblks(1024)           8       352         512          NoChange
```

FTP

Statistical information about each FTP transfer is printed upon completion of the transfer.

File transfer information

In ConvexOS 11.5.1, the transfer rates are easier to read. Results are automatically scaled to kilobytes or megabytes, as appropriate.

Example for ConvexOS 11.0, showing the scientific notation for the transfer rate, in kilobytes. This format is still available using the -s option to FTP:

```
1752282 bytes sent in 1.1 seconds (1.5e+03 Kbytes/s)
```

Example for ConvexOS 11.5.1, illustrating that the units have been appropriately scaled to megabytes:

```
1752282 bytes sent in 1.137 seconds (1.541 Mbytes/s)
```

Index

Symbols

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

U

V

W

X

Z

ORDER NUMBER
DXX-999

DOCUMENT NUMBER
710-001430-215