CONVEX CXbatch V2.0 Release Notice

Document No. 710-007830-004

November 1990

CONVEX Computer Corporation
Richardson, Texas USA

CONVEX
CXbatch Release Notice

Document No. 710-007830-004

Printed in the United States of America.

Table of Contents

CONVEX CXbatch Release Notice, V2.0

An Overview of the CXbatch V2.0 Release 1-1
- About This Package 1-1
  - Documentation 1-1
  - Prerequisites 1-1
  - Corequisites 1-1
  - Disk Space Requirements 1-2
  - Installing This Release 1-2
- New Features in CXbatch V2.0 1-2
  - Checkpoint/restart 1-2
  - Share Scheduler Integration 1-3
  - Batch Accounting 1-3
  - Batch Management 1-3
  - Batch Monitoring 1-3
  - Other Enhancements 1-4
- Impacts of the CXbatch V2.0 Release 1-4
  - Checkpoint/restart Impacts 1-4
  - Share Scheduler Impacts 1-4
  - Per-user Run-limit Impacts 1-4
  - Impacts of Obsolete Commands 1-4
- Notes and Cautions 1-5
  - Share Scheduler 1-5
  - Unsupported Limits 1-5
  - Background Processes 1-5
  - Login Shells 1-5
  - Load Balancing Pipe Client 1-6
  - COVUEbatch and COVUEshell 1-6
  - Unsupported NQS Features 1-6
  - Import_dir 1-6
  - Disk Space 1-6
  - System Management 1-7
- Software Problems and Fixes 1-7
  - Known Problems 1-7
  - Fixed Problems 1-9

CXbatch Initiator 2-1
- initiator.share 2-1

Overview

About This Package

CXbatch allows users to submit non-interactive jobs for batch execution on either the local machine or a remote machine in the CXbatch network. CXbatch provides a great deal of flexibility in queue configuration and management. Queues can be configured to enforce resource limits on requests, allow access only to selected groups and users, and route requests to lightly loaded machines.

CXbatch is based on NASA's Network Queueing System (NQS). Many CONVEX enhancements have been added, including checkpoint/restart, Share Scheduler integration, load balancing, batch accounting, automatic importing of files (via NFS), and direct submission to remote machines.

Documentation

The CXbatch Documentation Set included with this release contains the following documents:

- *Convex CXbatch Release Notice, Second Edition*
- *Convex CXbatch Installation Procedure, Second Edition*
- *Convex CXbatch Concepts Manual, Second Edition*
- *Convex CXbatch User's Guide, Second Edition*
- *Convex CXbatch System Manager's Guide, Second Edition*
- *Convex CXbatch System Manager's Utilities Reference, Second Edition*
- *Convex CXbatch Programmer's Reference, Second Edition*
- *Convex CXbatch Master Index, Second Edition*

Prerequisites

Before you can install this package, your system must already be running these software packages:

- ConvexOS V9.0 or a later release of the operating system.
- ConvexOS V9.0 Utilities or a later release of the system utilities.

If your system is not running these software releases, you must install them before continuing. If you need additional information on ConvexOS releases, contact the CONVEX Technical Assistance Center (TAC).

Corequisites

CXbatch V2.0 operates with a number of optional CONVEX software products. However, you must have the correct version of those packages in order to guarantee proper operation of all software.

CXbatch V2.0 operates with the following software packages:

- ConvexOS V9.0 Internet Services or a later release if you wish to use the load balancing option, the automatic NFS import option, or COVUE products.
- ConvexOS V9.0 NFS Utilities or a later release if you wish to use the automatic NFS import option or the COVUE products.
- ConvexOS V9.0 Share Scheduler or a later release if you wish to use Share Scheduler integration features.
- COVUEbatch V2.1 or a later release if you wish to use COVUEbatch functionality.
- COVUEshell V8.2 or a later release if you wish to use COVUEshell submission capability.

If you install CXbatch V2.0 and you are running older versions of any of the listed packages, please contact your CONVEX sales representative for information on upgrading your software.

Disk Space Requirements

CXbatch V2.0 requires 6.5 megabytes of disk space in the */usr* filesystem. If you are upgrading from a previous release of CXbatch, most of the disk space used by the previous installation is freed before the installation checks available disk space. In this case, you should have at least 2 megabytes of free space in */usr* before you attempt to install this release.

Installing This Release

Refer to *CONVEX CXbatch Installation Procedures* for special instructions on how to install the software. Refer to the *Impacts* section of this document for a list of changes that will impact users, operators, and managers of CXbatch V2.0.

New Features in CXbatch V2.0

This release of CXbatch has a number of new features including support for checkpoint/restart, better integration with the Convex Share Scheduler, and improved batch accounting, management, and monitoring. Many bug fixes are also included in this release. Refer to the *Software Problems and Fixes* section of this document for more details. This section summarizes the new features of CXbatch V2.0.

Checkpoint/restart

This release of CXbatch adds support for the checkpoint/restart functionality included in ConvexOS V9.0. There are two new stand alone commands, `qchkpnt`(1) and `qrestart`(1), and two corresponding `qmgr`(8) commands, `chkpnt request` and `restart request`, that exercise this new functionality. In addition to these commands, the checkpoint/restart facility is used automatically during CXbatch shutdowns and start-ups.

Two `qmgr`(8) commands have been added which allow a checkpoint directory to be specified for CXbatch and a *checkpointable* attribute to be set on a queue by queue basis. The `qsub`(1) command has additional options which allow the user to set the checkpointable attribute on requests and to specify an automatic checkpoint interval.

Share Scheduler Integration

Additional support for using CXbatch in conjunction with the Convex Share Scheduler is included in this release. Previously, using CXbatch with Share required a pseudo-user to be added to */etc/passwd* and shares to be allocated to it for each batch queue created. There is now a share policy attribute for each CXbatch batch queue. This can be set at the time the queue is created, or changed afterwards using the `qmgr`(8) `set share_policy` command. The share policy can be set to charge share usage by the queue to a fixed UID's share group or tc the share group of the user who submits each request.

Batch Accounting

Improvements to batch accounting are included in this release of CXBatch. A new utility, `qsa`(8), has been added that interprets the data in the accounting log file and generates several types of reports based on the command line options. One new field, request start time, has been added to the accounting file.

There are two new accounting related options to the `qsub`(1) command. The `-b` option allows the user to indicate the desired billing activity which should be used by a request. This is validated in the same manner as the `bill`(1) command. The `-y` option causes some accounting information to be appended to a request's standard output file.

Batch Management

New batch management utilities are included in this release of CXbatch. `qrun`(8) allows a high priority request to be run immediately, temporarily bumping-up the run limit for the request's queue. There is a corresponding `qmgr`(8) command, `run request`, that serves the same purpose.

The new utility `qsnapshot` dumps the contents of the `qmgr`(8) or `qmapmgr`(8) database as a series of `qmgr`(8) or `qmapmgr`(8) commands. This allows a queue manager to duplicate a CXbatch configuration from one system to another system. Other applications, such as duplicating a complex queue, are also possible by editing the output of `qsnapshot`(8) before passing it back into `qmgr`(8).

Two new user limits have been added to the CXbatch system. The global per-user run-limit limits the total number of requests any one user may have running within the local CXbatch system at any one time. The per-queue per-user run-limit limits the number of requests any given user may have running simultaneously in a particular queue.

Batch Monitoring

Two new utilities aid in monitoring CXbatch activity. `qps`(1) allows processes associated with CXbatch daemons and requests to be displayed in a format similar to `ps`(1). It actually uses `ps`(1) internally and prepends queue and request information to the associated processes.

The second new monitoring utility, `qwatch`(8), is a `syspic`(8)-like utility that displays information similar to that from `qstat`(1) in a full screen format. The information is updated at intervals and can be 'zoomed' in on for more detail.

Other Enhancements

The `start CXbatch` command has been added to `qmgr`(8) to allow the daemons associated with CXbatch to be restarted from within `qmgr`(8) when they have been shutdown. There is a new `-h` option to `qsub`(1) which causes a request to be placed on hold at the time it is submitted. The request can be enabled to run subsequently using the `qmgr`(8) `release request` command.

Impacts of the CXbatch V2.0 Release

Many of the changes to CXbatch will be transparent to most CXbatch users. However, there are some changes of which the user community should be made aware. This section lists the major changes that may impact users when CXbatch V2.0 is installed.

Checkpoint/restart Impacts

The new checkpoint/restart facility in CXbatch requires some configuration on the part of the CXbatch administrator before it is available to the user. If checkpoint/restart is configured in CXbatch, users should be notified of the changes in behavior possible during CXbatch start-up after a shutdown or system crash. Requests may be restarted from the last successful checkpoint operation rather than starting over from the beginning. System managers should inform users who wish to take advantage of the checkpoint/restart facility of the system overhead associated with these operations.

Share Scheduler Impacts

If the new share policy attribute is used to charge share usage within a queue to the submitting users share, system managers should inform users of the possible decrease in their interactive response associated with submitting CPU-intensive requests to CXbatch.

Per-user Run-limit Impacts

If the global per-user or per-queue per-user run-limits are used on your system, system managers should inform users of the changes in job scheduling associated with these limits.

Impacts of Obsolete Commands

If users are still using the CONVEX Distributed Batch compatibility commands, system managers should inform them that these commands are no longer available and instruct them in the use of the equivalent CXbatch commands. The following obsolete commands are removed by this release: `list`(1), `submit`(1), `move`(1), `remove`(1), `hold`(1), `unhold`(1), `ssp`(1), and `sbatch`(1).

Notes and Cautions

This section contains useful information and words of caution about this release of CXbatch.

Share Scheduler

If the CONVEX Share Scheduler is installed on your system, an appropriate share policy should be set for each queue. The default share policy is `Fixed` = *<queuename>*. If the default queue configuration is to be installed from the release tape, the pseudo-users *short*, *long* and *verylong* must exist on your system and have been assigned shares before the installation is started. This is normally done using the `nu`(8) utility.

Unsupported Limits

The NQS implementation supports a number of resource limits. CXbatch can only enforce the resource limits supported by the underlying ConvexOS operating system. The actual limits supported by CXbatch, or any other NQS batch implementation, is reported by the `qlimit`(1) command or by the `qmgr`(8) `show limits_supported` command. All the standard NQS resource limits are accepted by the CXbatch `qsub`(1) and will be passed by CXbatch to the destination queue. If a requested resource limit is supported by the destination system, it will be enforced at run time. Otherwise, it will be silently ignored.

Some resource limits in CXbatch are supported only to the extent possible. For example, cpu time limits may be entered down to the milliseconds with `qsub`(1), but because ConvexOS only supports granularity in seconds, milliseconds are ignored.

Background Processes

When a request is selected to run, a shepherd process starts it by executing a top level shell which runs the submitted command script. When this top level shell process exits, the request is considered completed and the shepherd process cleans up by killing any remaining child processes started by the top level shell. This means that requests should not run any background commands without waiting for them to complete using the applicable shell commands.

Note

One side effect of this behavior has been found when using the `mail`(1) or `binmail`(1) commands from within a CXbatch request. If the addressee of a mail message is on the local (execution) system, a sub-process is forked by mail to deliver the mail message and the mail command itself exits. If the CXbatch request completes before the mail has been delivered, the delivery process may be killed by the shepherd process. This causes the mail to disappear without a trace. A workaround for this situation is the use of the -v option to `mail`(1) or sleep for a short period of time to allow the mail to be delivered.

Login Shells

By default, CXbatch runs a job using a non-login shell. This can be overridden with the `-l` option to `qsub`(1). If your job runs under a login shell, the */etc/login* (C shell) or */etc/profile* (Bourne shell and KornShell) and your *.login* (C shell) or *.profile* (Bourne shell and KornShell) will be read. A job running under the C shell will also read */etc/logout* and *.logout*.

The string **ENVIRONMENT=BATCH** is added to the environment of the batch request. With this variable, shell start-up scripts (such as *.profile*, *.login* and *.cshrc*) can test for batch request execution and not, for example, perform any setting of terminal characteristics. The C shell always reads your *.cshrc* file regardless of whether it is running as a login shell, so these precautions should be taken.

When running jobs under the C shell, if a login shell is requested using the `-l` option of `qsub`(1), the message "Warning: no access to tty..." will be included in the job's output file. There is no way CXbatch can suppress this message; it would require a change to the C shell.

Load Balancing Pipe Client

The CXbatch load balancing pipe client, `pipeldav`, works with CXbatch pipe queue destinations, but not with non-CONVEX NQS destinations.

COVUEbatch and COVUEshell

When COVUE is used with CXbatch, certain COVUE commands cannot obtain queue and job information from pipe queue destinations that are running non-CONVEX NQS. There should not be any problems obtaining information with CXbatch destination queues.

CXbatch V2.0 no longer contains `queued`(8), a daemon previously needed by COVUEbatch and COVUEshell to obtain queue information. If COVUEbatch or COVUEshell are to be used with this release of CXbatch, COVUEbatch V2.1 or later, and COVUEshell V8.2 or later must be used.

Unsupported NQS Features

Although CXbatch is based on NQS, some NQS features not supported include:

- device queues
- queue complexes
- `qmgr`(8) `set global run_limit` command

Import_dir

When the `import_dir` attribute is set for a job request that is to run on a remote machine, the current working directory of the request is made available to the remote system through the use of an NFS mount. These mounts are made onto temporary directories in the */tmp* filesystem. System administrators should pay particular attention to any automatic clean-up procedures for the */tmp* filesystem and assure that such procedures do not traverse NFS mount points.

The current implementation of CXbatch cannot import a current working directory that is already within an NFS filesystem. Requests which attempt to do this will fail.

Disk Space

System administrators should also take measures to insure that the filesystem containing */usr/spool/nqs* has a reasonable amount of free disk space during regular queue processing. Running out of disk space can cause job status and output to be lost if output files cannot be copied back to a request's current working directory due to disk space problems. Mail notification can also be lost due to the inability to write into */usr/spool/mail* due to disk space problems.

System Management

While every effort is made to ensure that batch jobs are controlled and cannot disrupt the operation of the system, just as in any user environment, a persistent user may be able to circumvent some of the controlling mechanisms. System administrators have the responsibility to monitor and prevent system abuse through the use of online protection mechanisms (i.e. disk quotas, resources limits, etc.) and offline human management techniques.

Software Problems and Fixes

This section contains a list of software bug reports. The list is divided into known problems and problems that have been fixed since the last release.

Known Problems

This section lists the known problems with CXbatch V2.0 as of October 28, 1990. Problems reported after this date are not reflected in this document. Please refer to this list before reporting a problem.

qmgr (PR-09971)

There should be `qmgr` commands that reset the accounting file, and maybe a command that resets the current sequence number.

qdel (PR-09988)

`qdel` needs an option to remove "all my jobs" like "`remove -`".

qmgr (PR-09995)

It would be nice to have an auto command completion feature in `qmgr`.

netdaemon (PR-10095)

The netdaemon will report that a request's output file was left in the user's home directory on the execution machine, if it was unable to return the output file to the primary destination. The mail notifying the user of this fact should include the name of the execution machine.

logfile (PR-11078)

It is desirable to be able to turn on a switch with CXbatch that will cause the commands issued to be printed in the logfile prior to the output from that command. This is very useful when debugging batch command scripts.

qmapmgr (PR-11228)

A name was changed in the `qmapmgr`(8) to correspond to our new name and added the old name as an alias. `show` displayed the results and the new alias was not shown. `qmapmgr`(8) on another CONVEX system was updated with the same changes and the `show` command displays the alias, OK.

doc (PR-11422)

A summary for the syntax of the various `qmgr` commands (the top part of each man page) would be helpful as a quick reference guide instead of flipping pages. Also a cross reference in each man page to its opposite command would be helpful (if it has an opposite).

start/stop batch (PR-11810)

It would be nice to say "start queue" all or some such thing and have it start them all. The same should work for stop.

qmgr (PR-11954, PR-12109)

Need a `qmgr` (8) command showing the status of running batch requests such as starting time, CPU-time consumed and name of the actual running program. There's no way to find out the starting time of a batch request and with `ps -x` one can only see the CPU time used by the now running process in the batch request. This feature would also be important for batch operators when they have to decide to shut down the batch system they should have the possibility see how long a job has been running.

CXbatch load average (PR-12009)

In the old batch system the system administrator could operate the queues depending on the load average (the `-flag` in */etc/batchcap*). This is no longer possible with CX-batch. They would have to use `pipeldav`(8) (with `rwhod`(8C)).

import (PR-12156)

If the working directory of a request is in an NFS filesystem, and the request is submitted to a remote queue with the `import_dir` attribute set to *yes*, the request will fail because nqsdaemon is unable to import the current working directory.

qsub (PR-12487)

It would be nice if it was possible to specify a dependency between jobs that are submitted to the same queue. This most useful if the queue has a width that is greater than 1.

qmgr (PR-12984)

It would be helpful to be able to do the following two functions. 1) A submitted job should be able to get the name of the queue it is running from. A suitable mechanism may be the setting of an environment variable QSUB_QUEUE. 2) A utility like `qstat`(1) should be able to get information on `request-id`. It should at least return the queue in which the job is running and the name of the request, but other attributes may be useful as well.

qsub (PR-13249)

It would be useful to be able to pass parameters to submitted batch jobs. For example: `qsub -av arg1 arg2 arg3.`

qmgr (PR-13306)

The CXbatch `qmgr`(8) command displays per-process limit values of unlimited as "UNLIMITED" (all caps). When setting a value, `qmgr`(8) will not accept input in caps; only in lower case.

qsub (PR-13531)

It would be helpful to have an option to batch queues so that jobs are run in order of shortest to longest. If this capability existed, 2 or 3 queues could be replaced with one. The only purpose is to insure that the shorter jobs get run before the long ones.

qdel (PR-13814)

A batch job can be deleted during execution by the `qdel -k` command, or the COVUE DELETE/ENTRY command on the VAX. If this is done, the batch account record for the job does not include the resources consumed by the process that was active when the delete signal was received. For a site with very long jobs (which is typical for CONVEX customers), that charges its users based on the information found in the batch accounting file (*/usr/adm/batch-acct*), this can mean losing a lot of money.

qsub (PR-13893)

If a user with */bin/false* as the login shell attempts to run a job in the CXbatch system, it will stop the queue. This problem occurs when using COVUEnet 2.0 and COVUEbatch 2.0. COVUEnet requires the default access account to be created with */bin/false* as the login shell. This account may be used for COVUEbatch requests connecting by the default account. If the shell strategy is not fixed, CXbatch has trouble.

qsub (PR-14022)

CXbatch V1.1 no longer runs jobs under a login shell by default. To control this behavior, the '-l' switch was added. However, this '-l' switch is not recognized by non-CXbatch NQS clients. On the command line, this option is refused, as is the whole command. As an embedded option, '-l' causes an error message, but at least it is passed on to the CONVEX CXbatch NQS server. At that point, however, nqsdaemon/netserver do not evaluate options, and the '-l' is lost.

CXbatch (PR-14105)

A CXbatch request in a pipe queue to a remote machine that is unreachable will not ever run if the system holding the request is down at the scheduled retry time. It will never retry.

CXbatch (PR-14453)

Request that CXbatch include an command to enable/disable and start/stop all queues listed with one command, instead of having to issue a command for each queue to be enabled/started.

suspend (PR-15384)

A request for users to have the ability to suspend and restart their own jobs.

qsub (PR-15417)

It would be nice if CXbatch would have a option to enable a bell or notice signal to the user at the end of a batch job, like it was in the old version of the batch system.

qstat (PR-15418)

It would be nice to have a option with `qstat` to get information about the job limits. This should show job limits regardless of whether a job is currently running. Output should be similar to the `qmgr show long queue` command.

CXbatch (PR-15523)

CXbatch is a lot less flexible than the former distributed batch system. For instance, though a pipe queue may have its entries forwarded to batch queues on different systems, there seems to be no way to move an entry from one batch queue to a batch queue on a different system. Using qmgr's move request command to attempt to do this provokes either a "no such request at local host" or "no such queue."

qsub (PR-15653)

A request that qsub accept the "new" form of NQS commands (lines that contain 'QSUB' rather than '@$') for embedded commands.

Fixed Problems

This section lists the fixes that have been made to CXbatch since the V1.1 release.

submit (PR-03789)

`submit`(1) needs an option to force jobs to run on the host they were submitted from. When queues are not set up to serve particular jobs, you have no control where the job goes to. If a queue serves only cc jobs, then submit it to that queue. But if a generic X job may go to any system, then I want a way to force it to stay here.

Resolution: This is a restriction in the old batch system and is incorporated in the new batch system.

batchstat (PR-09675)

The CXbatch system seems to lack the functionality of the batchstat command that was in the previous Distributed Batch System. There does not appear to be any simple way of listing CXbatch system information such as number of jobs submitted, the time the last job was submitted, the average time between job submissions, the average and maximum amount of time a job sits in the queue before the job begins executing, the average and maximum amount of cpu time used by the execution of jobs from the queue, etc.

Resolution: The `qsa`(8) command displays individual statistics, averages, or totals of information about CXbatch requests. This data may be accessed by queue and/or user.

qmgr (PR-09964)

It would be nice if we could start-up CXbatch with `qmgr`(8). Then a batch operator could bring it up easily.

Resolution: Operators or managers may now start CXbatch from the `qmgr`(8) using the `start cxbatch` command.

CXbatch (PR-09970)

We need to use the $PAGER more consistently in `qmgr`(8) and other CXbatch utilities.

Resolution: `qmgr`(8) now uses $PAGER for `show all`, `show long queue`, and `show queue`.

CXbatch (PR-09972)

CXbatch should support checkpoint/restart like Cray NQS.

Resolution: CXbatch now supports Checkpoint/Restart.

CXbatch (PR-09974)

We need an interactive display utility for batch like syspic.

Resolution: `qwatch`(8) is an interactive CXbatch `qstat` display.

qstat (PR-09975)

There should be an option that displays all processes associated with each batch job.

Resolution: `qps`(1) displays process status of CXbatch related processes by queue-name, request-id, or process-id.

CXbatch (PR-09985)

It would be nice if `queued` was automatically started and stopped (like the other CXbatch daemons) when the batch system is started or shutdown.

Resolution: `queued` is no longer included in the CXbatch V2.0 release COVUEbatch V2.1 and COVUEshell V8.2 do not require queued to interoperate with CXbatch V2.0.

qmgr (PR-09994)

There is not a command to remove a queue description. The best you can do is overwrite it with spaces.

Resolution: `SEt DESCription = (null) <queue-name>` will now remove a queue description.

qmgr (PR-09996)

Why can't I `set mail CXbatch-daemon` (unknown user) when CXbatch does it itself? Also, the default address should be something you can reply to.

Resolution: This problem is not reproducible. CXbatch has always had root as the default mail_uid.

qmgr (PR-10218)

It would be useful if `qmgr`(8) could dump the current parameters to a file in such a way that they could be recreated from that output. This command file could be used to build up similar queues on another system, or recreate them on the current one. An awk (or perl) script that massaged the output of `qmgr show long q` would probably suffice.

Resolution: The `qsnapshot`(8) command dumps the current CXbatch queue or networking configuration as a series of `qmgr`(8) or `qmapmgr`(8) commands.

qmgr (PR-10219, PR-11952)

It would be VERY useful if a command `freeze q` analogous to `stop q` existed that would send a (non-trappable) SIGSTOP to the currently executing jobs. The shepherd would have to wait until he was told to continue, ensuring that no one tried to SIGCONT their job to circumvent system administrative fiats.

Resolution: CXbatch queues may be "frozen" by first `suspending` running jobs and then `stopping` the queue. The `suspend request` command checkpoints the jobs. It was determined that this would be more useful than simply stopping jobs, because their resources would not be freed.

qmgr (PR-10370)

If a job fills its output file due to an endless loop, and hence the entire file system, CXbatch does not recover gracefully. CXbatch appears to have no controls about using up all file system space in */usr/spool/nqs* and does not clean up after the user's job abnormally terminated.

Resolution: Unable to reproduce this behavior.

per-user limit (PR-11235)

Customer requests that CXbatch allow a limit of the number of jobs that can be run on a per user basis. Cray's NQS supports limits on a per user basis.

Resolution: CXbatch V2.0 now supports per-user run limits on per-queue and global basis.

CXbatch (PR-11358)

In an attempt to prevent users from popping X windows or using rsh to run commands while in stand-alone batch mode, `rshd` was disabled in */etc/inetd.conf*. However this action caused all the batch commands die with everything from "no access to terminal" to not being able to find files that are on LOCAL disks. This may cause real problems for site that only have COVUEnet for their networking software.

Resolution: This problem cannot be reproduced.

CXbatch (PR-11487)

Batch jobs which use a system(3F) call from FORTRAN to deliver mail do not perform correctly. The mail is never delivered.

Resolution: This situation occurs due to a design decision in NQS. The mail delivery is being done by a process forked by the mail command. If the delivery isn't completed when the request exits, the shepherd process kills the delivery process while cleaning up stray process created by the request. Refer to the CXbatch V2.0 release notice for more details. This report is being closed as a restriction.

accounting (PR-11568)

Some facility should be provided to dump accounting information about a batch job at the end of the log file, as happens in VMS. The UNIX `time` command does provide a partial workaround though it is not very usable in COVUE Batch jobs.

Resolution: CXbatch V2.0 has added an option to `qsub`(1), the `-y` option, that causes accounting information to be appended the standard output file of jobs running in queues with accounting enabled.

CXbatch (PR-11630)

The times the batch system measures via `getrusage`(2) differ greatly from the time `cputime`(3B) measures (not because of the fact that `cputime`(3B) only accounts for user time but for the fact that `getrusage`(2) is not as exact (only tenths of seconds?) as `cputime`(3B)).

Resolution: CXbatch accounting determines the CPU usage of a job as accurately as possible.

CXbatch (PR-11950, PR-12366, PR-12547)

It should be possible to specify a run limit per user for each batch queue and for the whole batch system so that no user can submit such many requests that the batch system is blocked "forever" for other users.

Resolution: CXbatchV2.0 now supports per user run limits on per-queue and global bases.

output (PR-11953)

Output file should contain: the executed commands and some job information--start/end time, cpu time, and resource usage of the job. Currently, one can only have this information in the OUTPUT file adding date and time commands to your batch job file. On batch systems on other machines this is done automatically.

Resolution: Accounting information may be appended to the stdout file of a request by using the `qsub -y` option. This data includes: queue, host, sequence number, remote host, submission time, start time, completion time, time spent executing in user mode, and time spent executing in the system.

Starting jobs (PR-11955)

Operator notices that the interactive load of the system is lower than usual and wants to start another batch job. There is no command to do this except to increase the run limit by one. Then the job starts, but the operator has to remember to set the run limit back. There should also be way to start a specific job in a queue. You can only start the first queued job in the queue.

Resolution: CXbatch V2.0 provides the `qrun`(8) command, which will force a specific request to begin execution, irrespective of the queue's run limit.

`qmgr` description (PR-12211)

After using the `set description` command in `qmgr`(8) to set the description for a queue, it is impossible to remove that description. It is possible to overwrite the description, replacing it with a new one, but there is no way to get back to the initial configuration (i.e., no description).

Resolution: `SEt DESCription = (null) <queue-name>` will now remove a queue description.

`qsub` (PR-12349)

The `qsub`(8) command should allow the user to change billing and user groups. Currently there is no way to use the `bill`(1) command in a `qsub`(1) script. If options were available on the `qsub` command line, it would help greatly!

Resolution: The new `-b` option to `qsub`(1) offers this functionality.

shares (PR-12374)

For processes run via CXbatch, Share uses the lnode of the batch queue in which the process is running. A user's past usage has no effect on the resources his processes receive in batch. Thus, a user who has been greedy and shouldn't receive any CPU time (or other resource) can circumvent Share by submitting a batch request. It should be possible for the system manager to configure CXbatch and Share so that batch requests use the lnode of the submitter.

Resolution: It is now possible to configure CXbatch to either charge CPU usage to the lnode of the submitter or to charge CPU usages to a specific lnode. It is accomplished using the `qmgr SEt SHAre_policy Fixed = <account> <queue>`, the `SEt SHAre_policy User <queue>`, or through the [`SHAre_policy`] option to the `CREate Batch` command.

qstat (PR-12382)

It would be nice if there was a simple, supported way to determine (from within a program) whether or not a specific process is running from CXbatch, given the process id. If the process is running from CXbatch, it would be nice to be able to easily determine the queue name and request id. Currently the only way (I know of) to tell if a process was started by CXbatch is to trace that process's parentage to determine if it is a descendant of a "CXbatch shepherd" process. A simpler solution would be nice.

Resolution: `qps`(1) displays process status of CXbatch related processes by *queuename*, *request-id*, or *process-id*.

accounting (PR-12413)

When accounting data is logged in the file specified with the `qmgr`(8) command `set acc_logfile`, the queue name written to that file is the name specified by the user when the request was submitted. It would be nice if there was an additional field which specified the "official" name of the batch queue. (Alternatively, names which are aliases should not be written to the log file.)

Resolution: The *queuename* field of the *batch_acct* structure now contains the actual name of the queue the request executed in rather than an alias.

mail--execution star (PR-12458)

Currently, batch accounting includes the time a job was submitted to CXbatch and the time the job completed. It would be helpful if the time the job actually started executing was also included. That information could then be used to determine the effectiveness of the queue configuration.

Resolution: The batch accounting record now contains the time at which a request began execution.

qsub (PR-12786)

This is an enhancement request for an option to `qsub`(1) to allow a user to place their job on hold when it is submitted to the queue.

Resolution: The `qsub -h` option specifies that a request be held at submission.

qmapmgr (PR-12926)

While changing the `mid` for a system, the error was reported when trying to add the new mid mapping: NMAP_ECONFLICT: Already exists. Trying to delete the same mid generated the error: NMAP_ENOMAP: No such mapping. `qmapmgr`(8) eventually coredumped.

Resolution: The */etc/nmap* database was corrupted by non-CXbatch utilities.

qsub man page (PR-13070)

The man pages for `qsub(1)` indicate that CXbatch does not support per-request limits. All of the per-request options (F, M, T, & V) are listed and fully explained. The man pages even contain examples of using the `-lT` flag. The majority of users will use the man pages vs. the *CXbatch User's Guide*, distributed with the product, which lists a caveat in section 2.5.2 that the per-process limits are ignored by Convex.

Resolution: This information has been documented as clearly as possible. The examples have been updated.

qsub (PR-13136)

Request that a `-h` flag be added to the `qsub`(1) command. The idea is that a job could be submitted to a queue and be put in a hold state until an operator released that job to run at some time determined by the operator.

Resolution: The `qsub -h` option specifies that a request be held at submission.

qmgr (PR-13155, PR-13507, PR-13508)

Since the change to daylight savings time, jobs submitted to run at a specific time have been 1 hour off. In the US, you can add an option for daylight savings time (`qsub -a "2am CDT"`) and the job will run at the appointed time. However, sites outside the US can't do this, as the abbreviations for timezones are not recognized for batch.

Resolution: The Daylight Savings Time code in `qsub`(1) has been improved and this problem should be corrected. All the european and australian timezones which are supported by our current libc are now recognized by `qsub`(1) (wet, met, eet, wetdst, metdst, eetdst, aest, acst, awst, aedt, acdt and awdt).

qsub (PR-13377)

In Europe since the change of legal hour, when using the `-a` option of `qsub`(1), all jobs begin one hour late. Investigating this problem found that if one specifies a non-American time zone in this option, CXbatch gives an error message.

Resolution: The Daylight Savings Time code in `qsub`(1) has been improved and this problem should be corrected. All the European and Australian timezones that are supported by our current *libc* are now recognized by `qsub`(1) (wet, met, eet, wetdst, metdst, eetdst, aest, acst, awst, aedt, acdt and awdt).

nqsdaemon (PR-13792)

When using CXbatch to pipequeues with COVUEbatch as the front end, CXbatch causes two kill signals to return to COVUEbatch. The effect of this is that COVUEbatch processes on the VAX do not die and retain logical links. After enough jobs run in this configuration COVUEnet no longer has logical links available and all DECnet traffic to the Convex is halted.

Resolution: CXbatch now signals COVUEbatch only once through the use of */bin/kill* rather than the shell built-in kill.

CXbatch (PR-14150)

A university with an IBM as its primary name server had a problem installing CXbatch V1.1. The name server had to be turned off to complete the installation. When the name server was turn back on, CXbatch complained: netsever client unknown to local host netserver aborted, when a job was submitted. To resolve problem, the hostname had to be added to the mid in all upper case. This was not required in CXbatch 1.0. Please document change or return to previous behavior.

Resolution: No change has been made to CXbatch in this area. If your name server is returning hostnames in uppercase letters, then these names should also appear in the `qmapmgr`(8) database. This report is closed as a restriction.

CXbatch Initiator

2

CONVEX has received several requests to implement alternative scheduling algorithms for CXbatch requests. Each request is usually accompanied by a description of the scheduling practices desired by a particular site. It became evident that some improvement needed to be done in this area, but no single scheduler would meet the needs of all customers.

To address this problem, CONVEX is investigating a concept we refer to as the `initiator` daemon. This daemon runs in conjunction with the `nqsdaemon` and is notified whenever the `nqsdaemon` is about to enter its job scheduling loop. The `initiator` can request information about queues and requests and can issue commands to modify the priority of queued requests based on this information and information obtained through other ConvexOS facilities.

The goal of this project is to provide a set of library routines and documentation which would allow customer sites to develop an initiator daemon which implements their own request scheduling algorithms. The interface would be designed so that an experienced C programmer could prototype and implement various scheduling algorithms.

For this release of CXbatch, CONVEX developed a sample initiator daemon for customer evaluation and feedback on the concept. We wish to determine if the initiator concept is the proper solution to the scheduler problem and if the interface would be used sufficiently to warrant its development.

initiator.share

The sample CXbatch initiator daemon is installed in */usr/lib/nqs/initiator.share* by the CXbatch V2.0 installation. The daemon is activated by renaming it */usr/lib/nqs/initiator* and restarting CXbatch. Each time a CXbatch request exits, this daemon is notified and given the opportunity to reorder CXbatch local batch queues based on a scheduling algorithm.

The algorithm employed by the sample CXbatch initiator daemon takes into account both the relative share assigned to the owner of a request and the time a request has been waiting in the queue. A ranking is generated for requests in each queue, and intra-queue request priorities are set accordingly.

Relative priorities are generated by the following formula:

$$rank = share + \frac{ln(age)}{2}$$

where *share* is assigned percentage of machine and *age* is request age in seconds.

Customers are encouraged to activate and observe the operation of the sample CXbatch Initiator daemon and provide suggestions to CONVEX about its usefulness or about alternative solutions.