1.0 Introduction

1.1 What is fbatch - An Overview

fbatch is a package that provides the batch processing environment on FNALU/CLUBS. It is built as a layer of software on top of LSF (Load Sharing Facility) from Platform Computing. *fbatch* adds the following functionality to the LSF product:

- Interfaces to the Fermi products spacall (space allocator) and tcache/needfile.
- AFS token renewal at job execution time.
- Remote shell facility, allowing commands to be executed from any host.
- Provides case insensitive resource names.
- Complete implementation of LSF's b commands (bsub, bjobs, etc...).
- Man pages available for all fbatch commands.

fbatch does not reduce the functionality of LSF. All options supported by the LSF commands are supported by *fbatch.*

1.2 LSF and the batch environment.

LSF creates a single system image on a network of computers. The single image is referred to as a cluster, which has been named `fnalu_batch` on the FNALU system. `fnalu_batch` is composed of IRIX, AIX, SunOS, and DEC/ALPHA hosts.

A key concept within LSF is that of a resource. Resources are assigned to hosts. When a user submits a job to LSF, a set (possibly empty) of resource requirements are supplied. LSF will execute the job on the lightest loaded host that contains all the the resources requested. A common use of resources is to direct a job to a host of a particular operating system. All hosts within the `fnalu_batch` cluster have a resource defined that identifies the operating system flavor. For example, all of the IRIX machines have a resource named "irix". By supplying a resource requirement of "irix", the job will be executed on the lightest loaded IRIX machine in the cluster. Resource names can be combined in a logical expression giving you the ability to specify things like "irix && clubs", which will cause a job to execute on a host that has both of these resources defined.

LSF distinguishes between client machines and server machines. A job can be submitted from either a client or a server, however jobs can only run on server machines. In the

FNALU/CLUBS cluster, some nodes are batch only; i.e. users can not login to these machines. These nodes currently are prefixed "fncl". These are the nodes formerly designated as the CLUBS system.

Jobs are submitted to LSF queues. Queues are set up to accommodate jobs of various CPU run time lengths and scratch disk space requirements. You should submit your job to the queue that best matches these requirements. In general, the shorter queues have a higher LSF priority and a lower nice value (meaning a higher UNIX priority) than the longer queues. When a host becomes overloaded, LSF will start suspending jobs. LSF suspends jobs based on the LSF priority of the job. The LSF priority does not effect the UNIX scheduler, only the dipatching and suspending of jobs.

A more detailed description of LSF is given in section 7.

1.3 Features of fbatch

fbatch supplies additional features that make LSF easier to use in the Fermi environment. These additional features are briefly described.

1.3.1 Interface to the Space Allocator and Needfile/tcache.

fbatch provides an almost transparent interface to the space allocator and needfile/tcache products. The space allocator program (spacall) provides scratch disk for a users job. To use the space allocator in the `fnalu_batch` cluster, submit your job to one of the `*_disk` queues. The path of the scratch disk space will be stored in the `$CLUBS_WORK_DIR` environment variable. The scratch space will be reclaimed when the job completes. The queue definitions setup for access to the space allocator will be further discussed in section 4.

The needfile/tcache interface in *fbatch* allows data to be prestaged by embedding needfile/tcache statements in a job. *fbatch* will submit a job on hold and call the prestage manager server if it finds needfile statements in the job. When the data has been prestaged, your job will be released.

1.3.2 AFS token renewal

A job that writes data into AFS space must have an AFS token. fbatch gets a new token for the job <u>when the job begins execution on the execution host</u>. This prevents the potential problem of your token expiring while the job is pending. When a job is submitted through *fbatch* the user is prompted for their AFS password. This password is encrypted using PGP, and is passed to the execution host through the environment. When the job is released for execution,, a `klog` command is issued using the password supplied at submit time.

1.3.3 Remote Shell Facility.

fbatch provides the ability to run fbatch commands from any machine. When an *fbatch* command is executed on a host that is neither a client or a server, fbatch will issue the LSF command via the `rsh` facility. If this is used, the proper entries in the /etc/.rhosts or $HOME/.rhosts must be made.

1.3.4 Case Insensitive Resource Names

fbatch converts resource names to the proper case. Thus, the resource names "irix" and "IRIX" and "IrIx" are all identical.

1.3.5 Complete Implementation of LSF b- commands

All of the LSF "b" commands are implemented within *fbatch*. A standard naming convention is used so that LSF commands and be converted to their corresponding *fbatch* command easily. The LSF commands of the form `b<string>` are named `fbatch_<string>` in fbatch. For example, the LSF command `bsub` is implemented in the fbatch command `fbatch_sub`. All options supported by LSF are supported by *fbatch*.

1.3.6 Man pages

Man pages are available for all of the fbatch commands. The `fbatch(1)` man page lists all available *fbatch* commands, as well as a short description of each command.

2.0 Gaining Access

In order to use *fbatch*, all you need is an account on FNALU. This account is also sufficient for submitting jobs to the traditional CLUBS nodes (fnclxx).

3.0 Getting Started

This section describes the steps needed to use the *fbatch* product, as well as providing examples of commonly used commands.

3.1 Environment Setup

fbatch is installed as a UPS product on FNALU. To setup your environment, issue the following from the command line:

```
setup fbatch
```

This will setup all environment variables needed to use *fbatch*.

3.2 Prerequisite Information

This section describes important information needed before submitting a job to the batch system.

3.2.1 Environment issues

When a job is submitted to LSF, the shell environment is carried over to the execution host. All environment variables that have been exported on the submission host will be set on the execution host. Under the previous batch system based on LoadLeveler, your environment was not carried over. If you need certain environment variables cleared when the job runs, you must do so explicitly within your job.

3.2.2 Shells and UPS

Because your environment is carried over to the execution host, you should be careful about changing your UNIX shell. For example, if csh is your default login and your batch job shell is ksh, then the setup command will not be properly defined.

3.2.3 Determining Available Resources, Queues, and Hosts

Before a job is submitted, you need to determine the resources defined on a particular host. For example, if the job is running an executable that was built on an IRIX host, then the job must be submitted with a resource requirement of "irix". Using the *fbatch* command `fbatch_hostinfo`, you can find out the names of defined resources, and which hosts these resources are defined on. `fbatch_hostinfo` is one of the few *fbatch* commands that does not have a corresponding LSF command.

Issuing `fbatch_hostinfo` will result in output similar to the following:

```
fsgi02          SGI R4400Ch2 84.0    20   511M  2755M  Yes (irix any fsgi02)
fsui02       SUNSOL ULTRA167 154.0    4     -      -   Yes (sparc any fsui02 sun)
fibb01          AIX   RS6000 49.0     1   192M   400M  Yes (aix any fibb01)
fncl09          AIX   RS6000 49.0     1   128M  1136M  Yes (fncl09 aix any)
fncl10          AIX   RS6000 49.0     1   128M  1136M  Yes (aix any fncl10)
fncl06          SGI  R4400Cr 63.0     1   143M   738M  Yes (irix any fncl06)
```

```
fncl07          SGI  R4400Cr 63.0     1   143M   639M  Yes (irix any fncl07)
fncl11          AIX   RS6000 49.0     1   128M   368M  Yes (aix any fncl11)
fdei01       ALPHA 2100A500 136.0    4   637M  1499M  Yes (dec osf1 alpha fdei01)
fibi01          AIX   RS6000 49.0     -      -      -  No (aix any fibi01)
fsgi01          SGI    4D420 27.0     -      -      -  No (any fsgi01)
```

This table lists the name of the host(fdei01), the host type (ALPHA), the model (2100A500), the CPU factor (136.0), the number of processors on the host(4), the amount of memory on the machine (637M), the amount of swap space(1499M), whether this is a server machine (YES), and the resources defined for this host (dec,osf1,alpha,fdei01).

The set of resource requirements given when submitting a job can be empty, but it is advisable to list the resources that your job needs. If no resource requirements are given, LSF will execute the job on a host with the same model and type as the submission host. This may cause your job run on a heavily loaded host when a lightly loaded host is available. For example, assume that a job is submitted from fsgi02 with no resource requirements given, and that fsgi02 is heavily loaded. If your job is capable of running on any IRIX host, then it would be advisable to use "irix" as a resource. LSF would then pick the lightest loaded "irix" host available.

There may be times where you must have your job run on a particular host. If a resource requirement string is not supplied, there is no guarantee that LSF will run your job on the host from which the job was submitted. For example, fibi01 and fibb01 all have the same host type (AIX) and the same model (560). If a job is submitted from fibi01 without any resource requirements, it is eligible to run on either fibi01 or fibb01. By specifying a resource of "fibi01", you guarantee that your job will run on fncl04.

All hosts have the resource "`any`" defined. If you have job that can run on any one of the hosts in the cluster, then specify "`any`" as a resource requirement. Your job will then run on the lightest loaded host in the cluster.

A description of LSF's scheduling policy is given in section 7.

3.3 Job Submission.

Jobs are submitted using the `fbatch_sub` command. Control information needed by *fbatch* is supplied as command line arguments. No embedded control information is needed in your job. If you are converting a job from LoadLeveler to LSF, you can leave the LoadLeveler commands in your job. These are treated as comments, since the first column contains the # symbol.

The only required information to `fbatch_sub` is the name of an executable file, or a valid UNIX command. The following shows a job that is submitted with no arguments.:

```
fbatch_sub sleep 1
```

This submits a job to the default queue(`30min`). The job simply executes the UNIX sleep command.

To specify the name of a queue to submit a job to, use the `-q` option followed by the name of the queue. To submit a job called *myjob* to the `4hr`, the following command would be used:

```
fbatch_sub -q 4hr myjob
```

fbatch uses standard UNIX command line syntax. This means that all options must precede the command line arguments. For more information on command line syntax, see the `intro(1)` man page.

Another important option is the `-R` option, which allows you to supply a set of resource requirements to the job. To submit *myjob* to an IRIX host using the *30min* queue, the following command would be used:

```
fbatch_sub -q 30min -R irix myjob
```

The resource string may be quoted, and is case insensitive. The following command is identical the the previous command:

```
fbatch_sub -q 30min -R "IRIX" myjob
```

Logical expressions can be formed with resource strings. To run a job on a CLUBS IRIX node, combine the resources with a logical AND as follows:

```
fbatch_sub -q 30min -R "irix && clubs" myjob.
```

Another useful option is the `-b` option, which will cause the job to be submitted after a specified time. The following job submits a job to an aix host using the 4hr queue. The job will not be scheduled to run until after 8pm:

```
fbatch_sub -q 4hr -R "aix" -b 20:00 myjob
```

There is also a `-t` option, which causes the job to be terminated at a specified time.

By default, stdout and stderr will be emailed to the user when the job completes. This can be changed by using the `-o` and `-e` options. The following job submits *myjob,* causing stdout to be written to *myjob.out,* and stderr to *myjob.err:*

```
fbatch_sub -q 30min -R "alpha" -o myjob.out -e myjob.err -N myjob
```

Using the `-o` and `-e` option will supress the mailing of these files. To have mail sent when the job is complete, you should use the `-N`(notify) option in conjunction with `-e` and `-o.` The `-N` flag will remove the job summary fom the `-o` file.

3.3.1 Tracking Batch Jobs

There are several command available that report and change the status of your batch job. Topicscovered in this section are:

- Displaying the current status of a job
- Finding out why jobs are pending or suspended
- Displaying the execution history of jobs
- Checking the output from jobs that have not completed yet
- Killing, suspending, and resuming jobs
- Changing the order of your pending jobs within a queue
- Moving jobs to other queues
- Parameter modification.

3.3.2 Displaying the current status of a job

The `fbatch_jobs` command reports the status of batch jobs. Without supplying any options, `fbatch_jobs` will display all of your running jobs.

```
<fsgi02> fbatch_jobs
JOBID USER     STAT  QUEUE      FROM_HOST   EXEC_HOST   JOB_NAME   SUBMIT_TIME
19106 fromm    RUN   test_queue fsgi02      fsui02      myjob1     Feb 12 14:22
19108 fromm    RUN   test_queue fsgi02      fsgi02      myjob2     Feb 12 14:22
```

`fbatch_jobs` has several flags that it can be supplied with. To see jobs run by a particular user, use the `-u <userid>` option. The keyword `all` can be used to see all running jobs in the system:

```
<fsgi02> fbatch_jobs -u all
JOBID USER     STAT  QUEUE      FROM_HOST   EXEC_HOST   JOB_NAME   SUBMIT_TIME
19078 e831tod  RUN   e831_z       fdei01      fdei01      zrecon    Feb 12 07:51
19098 clubstst RUN   30min_disk   fsgi02      fncl01      bb.cmd AIX Feb 12 13:36
19099 ledovsko RUN   30min_disk   fsgi02      fncl07      *ndeeg.cmd Feb 12 13:39
```

```
19083 e815     RUN   12hr_disk    fsgi02      fncl07      *302_12_22 Feb 12 09:50
19086 e815     RUN   12hr_disk    fsgi02      fncl07      *302_23_34 Feb 12 09:52
19101 clubstst RUN   30min_disk   fsgi02      fncl09      bb.cmd AIX Feb 12 14:01
19102 ledovsko RUN   30min_disk   fsgi02      fncl04      *_8364.cmd Feb 12 14:04
19041 e815     RUN   12hr_disk    fibi01.fnal fncl04      *170_11_22 Feb 11 16:46
19085 e815     RUN   12hr_disk    fsgi02      fncl04      *179_13_24 Feb 12 09:51
```

You can also find jobs on specific queues or hosts, and check the status of specific jobs using their job IDs or names. See the `fbatch_jobs` man page for more information.

3.3.3 Finding out why jobs are pending or suspended

When you submit a job to *fbatch*, it may be held in the queue before it starts running, and it may be suspended while it is running. The `fbatch_jobs -p` option displays the reasons why a job is pending.

```
<fsgi02> fbatch_jobs -u all -p
JOBID USER     STAT  QUEUE      FROM_HOST     JOB_NAME             SUBMIT_TIME
18191 soft-781 PSUSP 12hr       fsgi02        6_21                Feb  6 17:56
The job was suspended by user or system admin while pending;
19093 ledovsko PEND  12hr_disk  fsgi02        find3pi0d_8282.cmd  Feb 12 12:06
Not enough eligible hosts satisfying job's resource requirements;
Some hosts already have enough jobs from this queue;
The queue is configured to not start jobs on some hosts;
```

The status of the two jobs listed in this example are the two most commonly seen cases on the `fnalu_batch` system. Job 18191 is in a `PSUSP` state, which means that the job was suspended while pending. This is the mechanism that *fbatch* used to put a users job on "hold" if needfile/tcache statements are found in the users job. When the data has been prestaged, the prestager will issue a command to release the job.

The status of 19093 is `PEND`. A PEND state indicates that the job has not yet started. To see the reasons why a job has not started, use the -p flag on the `fbatch_jobs` command. This will list every reason why a job has not started. Job 19093 is not running for the following three reasons:

- Not enough eligible hosts satisfying job's resource requirements.
- Some hosts already have enough jobs from this queue. This means that there as at least one host that has the resources to run this job, but can not because it is already running the maximum number of jobs from this queue.

- The queue is configured to not start jobs on some hosts. This means that there is at least one host with the resources required by the job that is not defined to run jobs from the submitting queue.

If your job starts running, but is suspended by LSF it will enter a `SSUSP` state. `fbatch_jobs -p` will show all suspended jobs and list a reason for their suspension.

The `-l` option gives a long listing. To get a long listing of user smith's pending jobs, you can enter the following:

```
fbatch_jobs -u smith -p -l
```

3.3.4 Displaying the execution history of jobs

To find out what has happened to your job since it was submitted, you can use the `fbatch_hist` command. `fbatch_hist -l` gives a long listing.

```
<fsgi02> fbatch_hist -l 19118

Job Id <19118>, User <fromm>, Project <default>, Command <sleep 5>
Wed Feb 12 14:55:30: Submitted from host <fsgi02> to Queue <test_queue>, CWD </
                     afs/fnal.gov/products/UNIX/fbatch/v3_0a/bin>, Requested Re
                     sources <irix>;
Wed Feb 12 14:55:31: Started on <fsgi02>, Pid <29410>.

Summary of time in seconds spent in various states by Wed Feb 12 14:55:41 1997
  PEND      PSUSP     RUN       USUSP     SSUSP     UNKWN     TOTAL
  1         0         10        0         0         0         11
```

LSF keeps information about job history in log files. By default, the fbatch_hist command looks in the current log file. You can look at jobs that ran a long time ago by specifying by using the -n option. fbatch_hist -n 3 123 will search the last 3 log files for job 123.

3.3.5 Checking Partial Job Output

You can view the output of an executing job by using the `fbatch_peek` command.

```
<fsui02> fbatch_jobs
```

```
JOBID USER     STAT  QUEUE      FROM_HOST   EXEC_HOST   JOB_NAME   SUBMIT_TIME
19258 fromm    RUN   test_queue fsui02.fnal fsui02      mytest     Feb 13 09:19
```

```
<fsui02> fbatch_peek 19258
Processing run 1....
Processing run 2....
Processing run 3....
Processing run 4....
```

3.3.6 Displaying Host and Queue Status

The `fbatch_hosts` and `fbatch_queues` display information about LSF hosts and queues. Without any arguments, `fbatch_hosts` lists all hosts defined in the cluster:

```
<fsui02> fbatch_hosts
HOST_NAME           STATUS    JL/U  MAX  NJOBS  RUN  SSUSP USUSP
fdei01              ok          -     -     1     1     0     0
fibb01              ok          -     -     0     0     0     0
fncl00              ok          -     -     0     0     0     0
fncl01              ok          -     -     3     3     0     0
fncl02              unavail     -     -     1     1     0     0
fncl04              closed      -     -     4     4     0     0
fncl05              ok          -     -     0     0     0     0
fncl06              ok          -     -     2     2     0     0
fncl07              ok          -     -     1     1     0     0
fncl09              ok          -     -     0     0     0     0
fncl10              ok          -     -     0     0     0     0
fncl11              ok          -     -     0     0     0     0
fsgi02              ok          -     -     1     1     0     0
fsui02              ok          -     -     1     1     0     0
```

The status field reflects the state of the host. The most common states and their meanings are:

ok

The host is available to accept batch jobs.

unavail

The host is down, or the Load Information Manager (LIM) and the slave batch daemon (sbatchd) on the host are unreachable.

unreach

The LIM on the host is running but the slave batch daemon (sbatchd) is unreachable.

closed

The host is not allowed to accept any remote batch job. There are several reasons causing the host to be closed. The long format shown by the -l option gives the possible reasons :

closed_Adm

The host is closed by the LSF administrator or root.

closed_Lock

The host is locked by the LSF administrator.

closed_Wind

The host is closed by its dispatch windows, which are defined in the configuration file lsb.hosts (see lsbatch(5)). All batch jobs on the host are suspended by the lsbatch system.

closed_Full

The configured maximum number of batch jobs on the host has been reached (see MAX field below).

closed_Busy

The host is overloaded because some load indices go beyond the configured thresholds. The displayed thresholds that cause the host to be busy are preceded by a '*'.

closed_LIM

The LIM on the host is unreachable, but the sbatchd is ok.

You can get a long listing by using the `-l` option. You can also limit the report to a specific host by supplying the host name to the command. See the `fbatch_hosts` and `fbatch_queues` man pages for more details.

3.3.7 Killing, suspending, and resuming jobs

The following commands are used to kill, suspend, and resume jobs. They all require a job number as an argument:

```
fbatch_kill
fbatch_stop
fbatch_resume
```

`fbatch_kill` can be used to send an arbitrary signal to a job.

```
fbatch_kill -s 1 123
```

sends signal 1 to the job 123.

3.3.8 Changing the order of your pending jobs within a queue

You can move pending jobs within a queue using the `fbatch_top` and `fbatch_bot` commands. `fbatch_bot` moves jobs toward the bottom of the queue so they are dispatched later. The default behavior is to move the job as close to the top or bottom of the queue as possible. By specifying a position on the command line, you can move a job to an arbitrary position relative to the top or bottom of a queue.

These commands do not allow users to move their own jobs ahead of those submitted by other users; only the dispatch order of the user's own jobs is changed. Only an LSF administrator can move one user's job ahead of another user's.

You can switch queues of a pending job by using the `fbatch_switch` command.

```
fbatch_switch 4hr 123
```

moves job `123` to the `4hr` queue. If the job that you are switching has already begun execution, then the CPU time allotment will not be increased. Thus, if the above job was submitted to the `30min` queue and it had started running before the `fbatch_switch` command was entered, the job would still only get 30 minutes of CPU time. If the job had not been dispatched, then the CPU limit would be set according to the queue definition.

3.3.9 Parameter modification

The `fbatch_modify` command changes the parameters of a job. This is useful if you want to change resources. For example, assume that you submit a job with a resource of "`aix`", but meant to submit the job with a resource of "`irix`". If the job has not been released, you can change the parameters using the `fbatch_modify` command:

```
fbatch_modify -R "irix" 123
```

This changes the resource of job 123 to "`irix`".

4.0 FBATCH Disk Queues

The *fbatch* product provides an easy interface to the *spacall* package. If your job is submitted to any of the *_disk queues, a temporary file system will be provided to your job. This file system can be accessed through the `$CLUBS_WORK_DIR` variable. All queues with fbatch 3.0 are setup to provide your job with 2.0 GB of temporary file space. When your job completes, the space will be reclaimed by the space allocator.

5.0 AFS Issues

fbatch provides an AFS token renewal service. When you submit your job to the batch system, you are prompted to enter your AFS password. *fbatch* encrypts your password using PGP (Pretty Good Privacy) and stores it in your environment. LSF copies over your environment to the host that the job will execute on. When your job is released, your encrypted password is decrytped and supplied to the `klog` command to get a new token on the execution host. Tokens on FNAL have a lifespan of 6 days. You do not have to have a valid token on the execution host when you submit the job. *fbatch* will automatically get a new token for you when your job starts executing.

Your AFS password in encrypted using the PGP software package. PGP uses RSA public key encryption along with the IDEA encryption algorithm. fbatch uses a 1024 bit RSA key, the longest and most secure key provided by PGP.

fbatch will check to see if the environment variabels `AFS_PASSWORD` or `AFS_ENCRYPTED_PW` are set. If either is set, then prompting for an AFS password is disabled. If AFS_ENCRYPTED_PW is set, then fbatch assumes that AFS_ENCRYPTED_PW contains the users encrypte password. This value will be input to the PGP decrypt function, and the decrypted password will be used as the password for the klog command.

If AFS_PASSWORD is set, then *fbatch* assumes this value is the users plaintext password. This value will be used as the password for the klog command. It is NOT recommended that you store your plaintext password in AFS_PASSWORD. This was the method used by *fbatch* in version 3.0. It was left in version 3.1 to insure backward compatability.

6.0 Advanced Topics.

This section covers some advanced topics on the use of *fbatch* and LSF. In particular, the following topics are discussed:

- Converting your LoadLeveler job to LSF.
- Sending email when job completes
- Submitting multiple jobs through 1 script.
- Running jobs through cron
- Submitting jobs to run at a specific time.
- Running jobs through cron
- Job Dependencies

6.1 Converting a LoadLeveler job to LSF

LSF replaced the LoadLeveler product, so many people will have LoadLeveler scripts that they wish to convert to LSF. Many of the arguments that are supplied to LSF via command line arguments were embedded in the LoadLeveler job as a form of job control. Conveniently, these job control statements began with a #. You can submit your LoadLeveler job to LSF without changing the job script. The LoadLeveler control statements will be interpreted as comments.,

There may need to be some changed related to your environment that are necessary.

6.2 Sending Email

Normally, *fbatch* will send mail to <user>@fnal.gov. This can be overridden with the `-u` option:

```
fbatch_sub -q 30min -u jdoe@fsgi02 myjob
```

This will mail `stderr` and `stdout` to `jdoe@fsgi02`. The `-u` and `-o/-e` options are mutually exclusive. If you specify the `-o` and `-e` options, then no mail will be sent when your job completes. If you want `stdout` and `stderr` redirected to a file and still receive notification when you job completes, use the -`N` option to `fbatch_sub`. There is no way to have `stdout/stderr` redirected to a file and mailed.

6.3 Submitting multiple jobs

It may be necessary or convenient to submit multiple jobs through a script. To do this, you will need to turn off the AFS password prompt. Password prompting can be disabled by setting the environment variable `AFS_ENCRYPTED_PW` or the environment variable `AFS_PASSWORD`. If `AFS_ENCRYPTED_PW` is set, then fbatch will decrypt this value using PGP, and use the decrypted value as your password. If `AFS_PASSWORD` is set, then fbatch will use the value of `AFS_PASSWORD` as your plaintext password. If both variables are set, then the value in `AFS_PASSWORD` is used.

The preferred method is to store the encrypted password in `AFS_ENCRYPTED_PW` using PGP. *fbatch* provides scripts to do this for you. If you are a sh, ksh, or bash user the variable is set by executing the following:

```
> . $FBATCH_DIR/bin/fbatch_setpgp.sh
```

The . before the command is essential. If this is omitted, the AFS_ENCRYPTED_PW variable will not be exported properly.

If you are using a csh or tcsh shell, the command is:

```
> source $FBATCH_DIR/bin/fbatch_setpgp.csh
```

These commands will prompt you for your AFS password, encrypt it, and store it in your environment.

After executing the command, you can echo $AFS_ENCRYPTED_PW to see that it is properly set:

```
>  echo "$AFS_ENCRYPTED_PW"
```

----BEGIN PGP MESSAGE-----
Version 2.6.2
hIwDJ+TsBb2ZLpUBA/0ey6epAbJmL99mHB0WjQeTZV6IIfNNEbSBuDCgvxizV4rz
KOfEHPKnuV8LX3X+CcaH1u59w4Nl4IjYowwZmFAD/6UIVGjvY7b+KAdGvBBaedjr
KfsQ5V3PVDsd+HU8J3nQ6XJnsLjOWjAB9dsiomd/+e8VRFbnlf+LrW8Vow7Uk6YA
AAios1oMEVgPjgreStmPexpHNJ09eZ+93XbZMD50PlZ4ey9/A==

=iRu1
-----END PGP MESSAGE-----

6.4 Submitting Jobs Through Cron

The biggest concern when submitting a job through cron is getting a valid AFS token. In order to turn off prompting of an AFS password, the AFS_ENCRYPTED_PW or `AFS_PASSWORD` variable must be set as mentioned in secion 6.3. The more secure method is to store the encrypted password in $AFS_ENCRYPTED_PW. One way to get your cron job to work is to store your encrypted password in a file, and modify your cron job to echo the contents of this file into the $AFS_ENCRYPTED_PW variable. For example, you can store the encrypted password in a file called encrypt.pw by issuing the following (assuming a csh or tcsh script):

```
setup fbatch

source $FBATCH_DIR/bin/fbatch_setpgp.csh

echo $AFS_ENCRYPTED_PW > encrypt.pw.
```

The file encrypt.pw can then be used in the cron job by adding the following line:

```
unsetenv AFS_PASSWORD
setenv AFS_ENCRYPTED_PW "`cat encrypt.pw`"
fbatch_sub ......
```

It is important to unset AFS_PASSWORD. If this is set, then *fbatch* will use this as the plaintext password.

The preferred method is to store the PGP encrypted password in the `AFS_ENCRYPTED_PW` variable. This can be done by sourcing (or using . for sh family) the fbatch_setpgp.csh (or fbatch_setpgp.sh for the family) script. These scripts will prompt you for your AFS password, encrypt it using PGP, and store it in the `AFS_ENCRYPTED_PW` variable. Although not recommended, you can store your plaintext password in `AFS_PASSWORD`. This was the original method used to disable prompting in fbatch 3.0.

6.5 Running Jobs at a Specific Time

You can submit a job to the batch system, and have execution deferred until a specific time window. This is done using `fbatch_sub -b -t` options. To submit a job that will begin at 22:00 on April 10 and will terminate at 06:00 on April 11, the following command could be used:

```
fbatch_sub -q 30min -b 04:10:22:00 -t 04:11:06:00 myjob
```

6.6 Job Dependencies

The `fbatch_sub -w` *depend_cond* option specifies a dependency condition, which is a logical expression based on the execution states of preceding batch jobs. When *depend_cond* evaluates to `TRUE`, the batch job can be started. Complex conditions can be written using the logical operators `"&&" (AND), "||" (OR), "!" (NOT)` and parentheses `"()"`.

Batch jobs are identified by job ID number or job name. The job ID number is displayed by the `fbatch_sub` command. The job name is a string specified by the `fbatch_sub -J` *jobname* option.

The conditions that can be tested are:

`started {jobID | jobName}`

TRUE if `{jobID|jobName}` has started running or run to completion.

`done {jobID | jobName}`

TRUE if `{jobID|jobName}` has completed successfully

`exit {jobdID | jobName}`

TRUE if `{jobID|jobName}`has completed abnormally

`ended {jobdID | jobName}`

TRUE is `{jobID|jobName}`has completed.

The following is an example of the use of job dependencies:

```
fbatch_sub -q 30min -w done(312) && (started(Job2)||exit(Job3))
```

7.0 LSF Details

This section provides users with some background information on the LSF product. Particular attention will be given to the LSF job scheduler.

LSF is the layer of software directly under the *fbatch* product. In most cases, the *fbatch* command is directly translated to an LSF command.

LSF dispatches jobs based on a batch priority of the job. The LSF priority of your job depends on the queue that you submitted your job to. Generally, queues with shorter CPU limits have higher priorities than queues with longer CPU limits. This priority is not to be confused with the UNIX priority of a running process. The batch priority is simply the priority used by LSF to dispatch and suspend jobs.

Queues also have a nice value associated with them. Jobs submitted to a queue are niced down to that value. In general, queues with shorter CPU limits have lower nice values (hence a higher UNIX priority) then queues with longer CPU limits. This value does not affect the batch priority.

LSF attempts to balance the load within the cluster. This means that if you submit a job from host A, LSF will determine which host at that time is the best candidate to run your job. If no resources are supplied to your job, LSF will only run your job on a host that is of the same type (irix, aix, etc..) and model (rs6000). This means that a job submitted from fsgi02 with no resource will always run on fsgi02, regardless of the load. This is because fsgi02 is the only machine in the cluster with a model of R4400Ch200. It is probably better to supply a string of resource requirements when submitting your job. This will allow LSF to balance the load across the processors. For example, if your job can run an any IRIX host in the cluster, then specify "`irix`" as a resource. LSF will then pick the best IRIX host to run your job.

LSF uses several factors to determine what the best host is to run your job. These include the host load, the number of processors on a host, and the speed of those processors. Therefore, if you submit a job from fsgi02 that can run on any IRIX host, it is possible that your job will run on fsig02 even if there are other IRIX hosts that are better suited to run your job.

LSF uses a variety of parameters to determine which host to dispatch a job to, as well as when jobs need to be suspended. If a host becomes overloaded, LSF will begin suspending jobs until the load falls below a preconfigured level. Jobs are suspended based on priority. The lowest priority job will be suspended first. If this does not cause the load to decrease below the preconfigured level, then the next lowest priority job will be suspended, and so on. If there is more than one job with the same priority running on a host, then the job that was submitted last will be suspended.

8.0 Getting Additional Information

8.1 Man Pages

The best source of help is the *fbatch* man pages. When you setup *fbatch*, your `$MANPATH` environment is setup to pick up the man pages for *fbatch*. The `fbatch(1)` man page gives a quick overview of *fbatch*, as well as a listing of all supported commands and a brief description of these commands. Each command has a man page where detailed information can be found.

8.2 World Wide Web Pages

Documentation and announcements can be found on the web. The following is a list of information found on the web. These sites can be found by going to the HPPC main web page (http://fnhppc.fnal.gov) and following the appropriate links.

- FBATCH main page (http://fnhppc.fnal.gov/fbatch/fbatch.html)
- LSF web page (http://www.platform.com/)
- FBATCH overview page (http://fnhppc.fnal.gov/fbatch/overview.html)
- FBATCH announcements (http://fnhppc.fnal.gov/fbatch/announcements.html)
- Documentation page (http://fnhppc.fnal.gov/fbatch/docs.html)

The documentation page has the following documents:

- FBATCH overview document
- FBATCH deployment plan/Installation Guide
- LSF User's Quick Reference
- LSF User's Guide
- LSF Administrator's Quick Reference

- LSF Administrator's Guide
- LSF Programmer's Guide

8.3 EMAIL Addresses

To contact support for the *fbatch* product, send mail to the helpdesk or fbatch-support@fnal.

9.0 FAQ

- **In fbatch3.0, I could type in an invalid AFS password and I would still get an AFS token, but in 3.1 this doesn't happen. What is going on?**

  With fbatch3.0, the LSF supplied method of passing AFS tokens between submission and execution hosts was used. Unfortunately, this method was not 100% reliable. Sometimes it worked, sometimes it didn't. With fbatch 3.1, a more secure and reliable method of getting an AFS token was implemented. Since the LSF supplied token passing mechanism was unreliable, we decided to scrap it in favor of our own method. With fbatch 3.1, if you do not supply the correct password your job will NOT get an AFS token. There is now a confirmation prompt when submitting jobs to reduce the possibility of mistyping your password.

- **How long is my AFS token good for with fbatch 3.1?**

  The default token lifetime has been extended to 6 days on FNALU. With fbatch 3.1, your job will receive a new token with a 6 day lifetime when your job starts executing

- **How to I pass parameters to a batch job?**

  You pass parameters to a job just as you would if you submitted the job interactively. For example, `fbatch_sub -q 30min myjob 10 100` would pass 10 and 100 to myjob.

- **I submitted my job to a *_disk queue, and it is staying in a `PEND` state. Why?**

  This could indicate a problem with the space allocation mechanism. Issue `fbatch_jobs -p` to find out why the job is pending. If you see something like "Pre execution command ended with a non-0 return code", then you should report the problem immediately.

- **I submitted my job with needfile statements, and the job is in a `PSUSP` state. What does this mean?**

  When you submit a job with needfile statements, your ob is intentionally put into a PSUSP state until your job is ready to run. This avoids LSF from dispatching your job to a host prematurely. Once a job is sent to a host to execute, it will stay there until the job completes. It does not make sense to dispatch your job to a processor until it is ready to run. By delaying the dispatching of your job to a host until the job is ready to run, LSF will make the best determination as to which host is best suited to run your job.

- **Is there a way that my application can tell if it is being run interactively or through the batch system?**

  Yes. LSF sets many environment variables when your job runs. Since your environment is carried over to the execution host, you can check to see if these variables exist. A good variable to check is `LSB_JOBID`.

- **I am not worried about getting a new AFS token, but fbatch keeps asking me for my password. Can I turn this feature off?**

  Yes. With fbatch 3.1, you can store your encrypted password in an environment variable. To do this, you must source $FBATCH_DIR/bin/fbatch_setpgp.csh (or . $FBATCH_DIR/bin/fbatch_setpgp.sh for sh users). You can also store your plaintext password in AFS_PASSWORD, but this is not recommended for security reasons.

- **Can I direct my job to go to a particular host?**

  Yes. There is the `-m` option on `fbatch_sub`, which directs your job to a particular host. In addition, each host in the `fnalu_batch` cluster has a resource defined for it that is the name of the host. Therefore, you can use the `-R` option as well.

- **Can I direct my job to go to any host on the cluster?**

  Yes. All hosts in the cluster have a resource named `any`. By specifying this resource, all hosts in the cluster are eligible to run your job..

- **My job is ending but I am not getting email notification. Why?**

  If you are using the `-e` and `-o` options, then your stdout and stderr are being redirected to files instead of email. If you want notification, you must use the `-N` option. Otherwise, your mail should be going to <user>@fnal.gov. If this is not correct, then you should explicitly specify where the mail should be sent with the `-u` option.

- **How can I tell which resources are defined in the batch system?**

  `fbatch_hostinfo` will give you this information.

- **I noticed job ABC has been running for a long time in the 30min queue. Shouldn't LSF be cancelling this job?**

  LSF normalizes runtimes of jobs based on the CPU speed. The queues are setup in the batch cluster to give the job at least that much time on the fastest machine in the cluster. Jobs submitted to the `30min` queue will get 30 minutes of CPU time on the fastest machine. The hosts in the fnalu_batch cluster vary greatly in speed. Therefore, it is possible that a job submitted to a slower host will get much more than 30 minutes of CPU time.

- **I have a problem with the batch system. What type of information should I provide when I report the problem?**

  The job number and a description of the problem are the most important pieces of information to provide. If you do not know the job number, give the account that submitted the job and the approximate time that the job was submitted.

- **On IRIX 5.3, my batch job seems to run very slowly, but if I submit it interactively it runs fine. What's going on?**

  There seems to be a bug in the IRIX 5.3 scheduler if certain resource limits are not set. When running through *fbatch*, the resource limit for the resident set size is set to unlimited. This apparently causes IRIX to give your process a very small amount of memory, causing the process to be swapped out often. To work around this, you can submit your job using the -M option as follows:

  ```
  fbatch_sub -q test_queue -M 505784 myjob.
  ```

- **I want to use the needfile/tcache product. How do I setup nt in my job?**

  The handling of the nt product is different with the new fbatch. You should not explicitly setup the nt product before you run your job. If needfile statements are found in your job, fbatch will automatically call the correct version of the nt product. This is dependent on where the job is dispatched to. If you do have nt setup, your script should unset `FERMINT_DPOOL_DIR`. You should not setup nt in your job.

- **Do the queue names reflect CPU time or wall clock time?**

  The queue names reflect CPU time. The amount of CPU time your job will get is a normalized value based on the speed of the host. The 4hr queue is setup to give a job 4 hours of CPU time on the fastest host in the cluster. It may get more time than that on a slower machine.

10.0 Troubleshooting

This section describes common problems that can occur using the *fbatch* product.

LIM unavailable message.

If you submit an *fbatch* command and receive a message stating that the LIM is unavailable, then there is a problem with the LSF software. Report the problem to fbatch-support.

Job stays in `PEND` state when submitted to the *_disk queues

When a job is submitted to the *_disk queues, a pre-execution command is run to obtain scratch disk space. If this command fails for any reason, your job will be put in a PEND state. This problem should be reported to fbatch-support.

Command not found message when using fbatch commands

You must setup the *fbatch* product to set the environment variables appropriately. If you get command not found, make sure you have done this.

No notification when job completes

By default, mail will be sent to user@fnal containing the stdout and stderr of your job. If you redirect this using the -o and -e options, then you will not receive mail when your job completes. To receive notification, use the -N option.

Resources not available

If your job is in a `PEND` state and the reason is "Not enough eligible hosts satisfying job's resource requirements;", you need to verify that the hosts containing the specified resource are available. Use the `fbatch_hostinfo` command to find out which hosts contain the specified resource. You can then use the `fbatch_hosts` command to check the status of these hosts. If the hosts are closed due to load thresholds exceeded, then all the hosts with the specified resource are simply to overloaded to accept jobs. If any of the hosts report an unavail status, report the problem to fbatch-support@fnal.