How cron & at Commands Work

Special Notices

Information in this document is correct to the best of our knowledge at the time of this writing. Please send feedback by fax to "AIXServ Information" at (512) 823-4009.

Please use this information with care. IBM will not be responsible for damages of any kind resulting from its use. The use of this information is the sole responsibility of the customer and depends on the customer's ability to evaluate and integrate this information into the customer's operational environment.

About This Document

This document Describes how the cron and at commands work and applies to both AIX versions 3.2 and 4.1.

About the cron and at Commands 1
More About cron Commands 2
How to Setup a crontab Entry 2
0481-086 and 0481-079 Errors From the cron Daemon 2
Howto List crontab Entries 3
Howto Ensure That cron is Working 3
Changed the crontab Entry But cron Isn't Running 3
A Job Runs From the Command Line But Not From cron 4
Getting crout* Files in /tmp 4
About cron Mail: Sending and Routing 4
Stop cron from Sending Mail 4
Route cron Mail to Another User 5
Did not Receive cron Mail 5
More About the at Command 5
What is the at Command 5
Howto Submit an at Job 5
Reader's Comments 7

About the cron and at Commands

Cron is a daemon that is started by the init process (via /etc/inittab) during the startup of the system. The cron daemon is responsible for running jobs (processes) at scheduled times, days of the week or days of the month. Processes that are run on a regular basis are normally placed in crontab file in the /var/spool/cron/crontabs directory.

The at command is used for jobs that only need to be run once. They are run from either the command line or from inside of scripts. The "at" entries do not go in the crontabs files.

By default, any user can create jobs for the cron and at commands to run. In the /var/adm/cron directory are two files, cron.deny and at.deny. When first installed, both are empty. If a cron.allow file exists in this directory, the user id (including root) must be in the file before they can run cron jobs. For the "at" command, there is a at.allow file. If the .allow files do not exist, both cron and "at" look in the .deny files. If the user is not denied, they may submit cron jobs.

For additional information about the cron, at, and crontabs commands, see their respective entries in InfoExplorer.

More About cron Commands

How to Setup a crontab Entry

When logged in as the user doing the cron job, do:

```
crontab -e
```

This starts an editing session for the /var/spool/cron/crontabs/user_id file. By default the vi editor is used. If the EDITOR environment variable is set, that editor is used.

Any lines beginning with a pound sign (#) are comments and are ignored.

A crontab entry consists of six fields:

- The minute (0 through 59)
- The hour (0 through 23)
- The day of the month (1 through 31)
- The month of the year (1 through 12)
- The day of the week (0 through 6 for Sunday through Saturday)
- The command to run

The specification of days may be made by two fields (day of the month and day of the week). If you specify both as a list of elements, both are adhered to. For example, the following entry:

```
0 0 1,15 * 1 command
```

would run command on the first and fifteenth days of each month, as well as every Monday. To specify days by only one field, the other field should contain an "*".

Example: To run the date command at 2a.m. on Monday:

```
0 2 * * 1 /bin/date
```

The parameters can also be specified as a range with a dash (-) like 1-5 or a list separated by commas like 1,2,3,4,5. For more information see the crontabs entry in InfoExplorer.

0481-086 and 0481-079 Errors From the cron Daemon

If you get errors from the cron daemon check for the following:

- The crontab file CANNOT contain any blank lines, including at the end of the file.
- If you don't have ANY blank lines, then check each line and insure that the syntax is correct. Test by commenting out (using # at the beginning of the line) an entry until you find the invalid line.

Howto List crontab Entries

The crontab -l command will list the cron jobs for the current user.

Howto Ensure That cron is Working

You can do a simple test with:

```
* * * * * /bin/date
```

Then check your mail. Cron sends the output from standard out and standard error to the mailbox of the user submitting the job. You should receive the output of the date command once each minute. If you do not get mail back, check to see if the cron daemon is running, do:

```
ps -eaf |grep cron
```

look for a line with the last field being: /etc/cron If cron is not running, check the /etc/inittab file for a entry like:

```
cron:2:respawn:/etc/cron
```

If you have this entry and cron is still not running, try:

```
telinit q
```

Then recheck with ps to see if cron is now running. If nothing works, contact your support organization for assistance.

Changed the crontab Entry But cron Isn't Running

When you directly edit a file in the /var/spool/cron/crontabs directory, cron will never look at the new file unless you tell it to. The correct way to add or change a cron entry is to login or su - to the userid then issue the crontab -e command. If you have manually edited one of the files, you must kill the cron daemon and let it respawn (restart). To find what to kill do:

```
ps -eaf |grep cron
```

The output will look something like:

```
root  20294      1   0   Sep 26      -  4:18 /etc/cron
        -
```

The marked number is the process id (PID), use kill -9 PID to stop cron and cause it to start and reread the crontabs files.

A Job Runs From the Command Line But Not From cron

The cron command starts a subshell from your home directory. If you want your .profile to be executed, you must do so in the program or script. The default shell for cron commands is /usr/bin/sh and the default search path is /usr/bin. To establish the same environment you normally expect to have, you can add something like this to the script:

```
#!/bin/ksh
cd /
. .profile
```

This will only pick up the environment variables that are set in the .profile file. To fully duplicate your environment, from the command line, run the env command and explicitly set all the variables in the cron script with:

```
export VARIABLE_NAME=value
```

Next:

```
#establish the proper environment
run commands
```

Note: Other environment variables are set in: /etc/environment and /etc/profile depending upon the setting of the ENV environment variable, the file pointed by the variable is also run. You may have to explicitly set all the variables you need in the cron script.

Getting crout* Files in /tmp

The /tmp directory has allot of files that look like:

```
croutAxwAXAxBC8
```

These files are created as temporary files in /tmp when cron runs a job. When the job finishes the crout file is usually removed. If the files are still there, the cron daemon is dying part way through the job and is getting respawned by init. Therefore the job in question never completes.

To locate the bad job, check the timestamp with:

```
ls -l crout*
```

and match it up to the entry in the crontabs file that it belongs to.

About cron Mail: Sending and Routing

Stop cron from Sending Mail

By default, cron will send any output going to standard out or standard error to the user that invoked the cron job. If you don't want mail sent to you, you can redirect standard out and standard error like:

```
cron_job_file >/dev/null 2>&1
```

If you are running a script, you may have to individually redirect any commands inside the file that write to either standard error or standard out.

Route cron Mail to Another User

To mail to another user, you can use:

```
cron_job_file | mail user_name
```

Again you may have to also do this inside the script file to catch all occurrences of output.

Did not Receive cron Mail

If you did not receive mail, first test if cron is working. See “Howto Ensure That cron is Working” on page 3.

If cron is running and you can get the output from the date command in the test above, then the program or script is probably not running. See “A Job Runs From the Command Line But Not From cron” on page 4.

If you can see that the job is running but you get no mail, test to see if you can send mail to the user submitting the cron job.

If you can send mail, then the command or script probably is not outputting anything to standard out or standard error or it is redirecting it somewhere else.

More About the at Command

What is the at Command

The "at" command reads from standard input the names of commands to be run at a later time and allows you to specify when the commands should be run. The at command follows the same rules with the .allow and .deny files as cron does.

Commands submitted by the at command are actually run by the cron command.

- at is just used to submit jobs.
- at will mail the output from standard out and standard error back to the user submitting the job.

- Unlike cron, the at command preserves the current environment including the userid and current directory.

Howto Submit an at Job

To schedule a job to run at a later time, you must specify a time to start the job. You may specify the time by using either the -t Date flag or the Time, Day, and Increment parameters.

The Date variable to the -t flag is specified using the following format:

```
[[CC]YY]MMDDhhmm[.SS]
```

The digits in the Date variable are defined as follows:

- CC Specifies the first two digits of the year (the century).
- YY Specifies the second two digits of the year.
- MM Specifies the month of the year (01 through 12).
- DD Specifies the day of the month (01 through 31).
- hh Specifies the hour of the day (00 through 23).
- mm Specifies the minute of the hour (00 through 59).
- SS Specifies the second of the minute (00 through 59).
- Both the CC and YY digits are optional. If neither is given, the current year is assumed

More information on submitting using the -t flag is available in InfoExplorer.

The most common way to submit a job is:

```
echo "command to execute" |at -t 950603220000 or
echo "command to execute" |at now + 200 minutes
```

See InfoExplorer for a complete list of time, day and increment parameters.