

NAME

crontab – maintains crontab files for individual users

SYNOPSIS

```
crontab [-u user] <file | ->
crontab [-T] <file | ->
crontab [-u user] <-l | -r | -e> [-i] [-s]
crontab -n [ hostname ]
crontab -c
crontab -V
```

DESCRIPTION

Crontab is the program used to install a crontab table *file*, remove or list the existing tables used to serve the **crond**(8) daemon. Each user can have their own crontab, and though these are files in */var/spool/*, they are not intended to be edited directly. For SELinux in MLS mode, you can define more crontabs for each range. For more information, see **selinux**(8).

In this version of *Cr on* it is possible to use a network-mounted shared */var/spool/cron* across a cluster of hosts and specify that only one of the hosts should run the crontab jobs in the particular directory at any one time. You may also use **crontab** from any of these hosts to edit the same shared set of crontab files, and to set and query which host should run the crontab jobs.

Scheduling cron jobs with **crontab** can be allowed or disallowed for different users. For this purpose, use the *cron.allow* and *cron.deny* files. If the *cron.allow* file exists, a user must be listed in it to be allowed to use **crontab**. If the *cron.allow* file does not exist but the *cron.deny* file does exist, then a user must *not* be listed in the *cron.deny* file in order to use **crontab**. If neither of these files exist, then only the super user is allowed to use **crontab**.

Another way to restrict the scheduling of cron jobs beyond **crontab** is to use PAM authentication in */etc/security/access.conf* to set up users, which are allowed or disallowed to use **crontab** or modify system cron jobs in the */etc/cron.d/* directory.

The temporary directory can be set in an environment variable. If it is not set by the user, the */tmp* directory is used.

When listing a crontab on a terminal the output will be colorized unless an environment variable *NO_COLOR* is set.

On edition or deletion of the crontab, a backup of the last crontab will be saved to *\$XDG_CACHE_HOME/crontab/crontab.bak* or *\$XDG_CACHE_HOME/crontab/crontab.<user>.bak* if **-u** is used. If the *XDG_CACHE_HOME* environment variable is not set, *\$HOME/.cache* will be used instead.

OPTIONS

- u** Specifies the name of the user whose crontab is to be modified. If this option is not used, **crontab** examines "your" crontab, i.e., the crontab of the person executing the command. If no crontab exists for a particular user, it is created for them the first time the **crontab -u** command is used under their username.
- T** Test the crontab file syntax without installing it. Once an issue is found, the validation is interrupted, so this will not return all the existing issues at the same execution.
- l** Displays the current crontab on standard output.
- r** Removes the current crontab.
- e** Edits the current crontab using the editor specified by the *VISUAL* or *EDITOR* environment variables. After you exit from the editor, the modified crontab will be installed automatically.
- i** This option modifies the **-r** option to prompt the user for a 'y/Y' response before actually removing the crontab.

- s** Appends the current SELinux security context string as an `MLS_LEVEL` setting to the crontab file before editing / replacement occurs - see the documentation of `MLS_LEVEL` in **crontab(5)**.
- n** This option is relevant only if **cron(8)** was started with the **-c** option, to enable clustering support. It is used to set the host in the cluster which should run the jobs specified in the crontab files in the `/var/spool/cron` directory. If a hostname is supplied, the host whose hostname returned by **gethostname(2)** matches the supplied hostname, will be selected to run the selected cron jobs subsequently. If there is no host in the cluster matching the supplied hostname, or you explicitly specify an empty hostname, then the selected jobs will not be run at all. If the hostname is omitted, the name of the local host returned by **gethostname(2)** is used. Using this option has no effect on the `/etc/crontab` file and the files in the `/etc/cron.d` directory, which are always run, and considered host-specific. For more information on clustering support, see **cron(8)**.
- c** This option is only relevant if **cron(8)** was started with the **-c** option, to enable clustering support. It is used to query which host in the cluster is currently set to run the jobs specified in the crontab files in the directory `/var/spool/cron`, as set using the **-n** option.
- V** Print version and exit.

CAVEATS

The files *cron.allow* and *cron.deny* cannot be used to restrict the execution of cron jobs; they only restrict the use of **crontab**. In particular, restricting access to **crontab** has no effect on an existing *crontab* of a user. Its jobs will continue to be executed until the crontab is removed.

The files *cron.allow* and *cron.deny* must be readable by the user invoking **crontab**. If this is not the case, then they are treated as non-existent.

SEE ALSO

crontab(5), **cron(8)**

FILES

`/etc/cron.allow`

`/etc/cron.deny`

STANDARDS

The *crontab* command conforms to IEEE Std1003.2-1992 (“POSIX”) with one exception: For replacing the current crontab with data from standard input the **-** has to be specified on the command line if the standard input is a TTY. This new command syntax differs from previous versions of Vixie Cron, as well as from the classic SVR3 syntax.

DIAGNOSTICS

An informative usage message appears if you run a crontab with a faulty command defined in it.

AUTHOR

Paul Vixie <vixie@isc.org>

Colin Dean <colin@colin-dean.org>