shyle-zacharias, sxc.hu

Cal, Date, Hwclock, and NTP

TIME WARP

I'm late, I'm late, for a very important date. For many applications, it is important that your PC has the correct time and time zone. We'll show you how to keep your PC clock ticking and how to use NTP to synchronize the time with a time server on the Web.

BY HEIKE JURZIK

An incorrectly set PC clock can bedisastrous – if your computer loses track of the time, you could end up juggling files from the future or emails from 30 years ago. The time warp could lead to misunderstandings, errors, or even crashes.

Almost all Linux distributions set the time and time zone during the installation phase, and desktop environments such as KDE and Gnome display a clock in the panel to give users quick access to tools for configuring the computer clock (Figure 1).

In the shell, *cal* displays a simple, but neatly formatted calendar. The *date* command gives you the date and time, although the output itself is fairly sparse. Additionally, this program can help the administrator set the date and time. *date* also demonstrates its potential in combination with other command-line tools and in scripts, for example, when programs generate file names that contain the current date.

The *hwclock* tool helps to synchronize the system time and the hardware clock. Of course, you will need to be root to run this program.

If the PC has a permanent Internet connection, you can automate the process of setting the clock by synchronizing your own timekeeper with a server on the Web via the Network Time Protocol (NTP).

Command-Line Calendar

If you call the calendar, *cal*, without any additional parameters, the program will display the current month of the current year. The *-3* option tells Cal to also show you last and next month; the *-y* flag gives you a calendar for the whole year (Figure 2).

To output a specific month, you need to pass the month in to Cal in the form of a two-digit number for the month and a four-digit figure for the year. By default, Cal will output the calendar in the language defined in the *LANG* environment variable [1]. If you prefer the time format for any other language, but would like to keep output from all other programs in the default language, you can set the *LC_TIME* variable to tell Cal to use the language of your choice.

The following example sets the date and time output to English:

```
LC_TIME=C cal -y
```

Of course, this command isn't necessary if your default language is already English.

In last month's column [1], I explained how to make variables permanent by adding the *export* command to the *~/.bashrc* file, as in *export LC_TIME=C*.

What's the Time?

If you type *date* at the command line, you will see the date, time, and also the time zone:

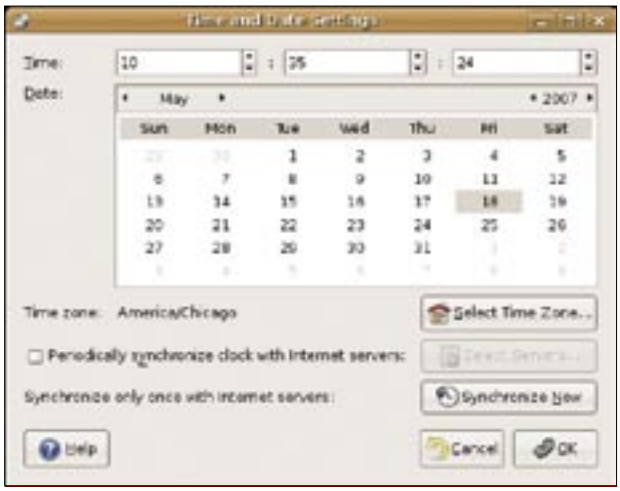

Figure 1: Both KDE and Gnome let users right-click on the clock in the panel to access the date and time settings.

```
$ <date>
Fri May 18 20:19:50 CEST 2007
```

Date also references the *LANG* variable to set the language, and it can also be influenced by setting *LC_TIME* just like Cal:

```
$ LC_TIME=C date
Tue Apr 24 19:42:30 CEST 2007
```

Date is even more flexible if you set the *TZ* (time zone) variable with the command.

Check your */usr/share/zoneinfo/* directory to find out which time-zone values your computer supports with *TZ*.

To find out the time in, for example, New York, you can simply run the following command:

```
$ TZ=America/New_York date
Tue Apr 24 13:43:02 EDT 2007
```

If you happen to live in Australia, for example, and need to phone friends in New York on a regular basis, you might like to set up an alias for the last command to avoid waking people up in the middle of the night.

To set up an alias, just add the following line to your Bash configuration file, *~/.bashrc*:

```
alias NY='TZ=America/ ➥
New_York date'
```

Table 1: Date: Command-Line Parameters

Parameter	Meaning
%M	Minutes (00 to 59)
%H	Hours, 24-hour clock
%I	Hours, 12-hour clock
%a	Weekday, short form
%A	Weekday, long form
%d	Day as two-digit number
%b	Name of month, short form
%B	Name of month, long form
%m	Name as two-digit number
%y	Year as two-digit number
%Y	Year as four-digit number
%D	Four-digit date (mm/dd/yy)
%T	Time in 24-hour clock format (hh:mm:ss)
%r	Time in 12-hour clock format (hh:mm:ss)
%t	Tabulator
%n	Line break
%%	% sign

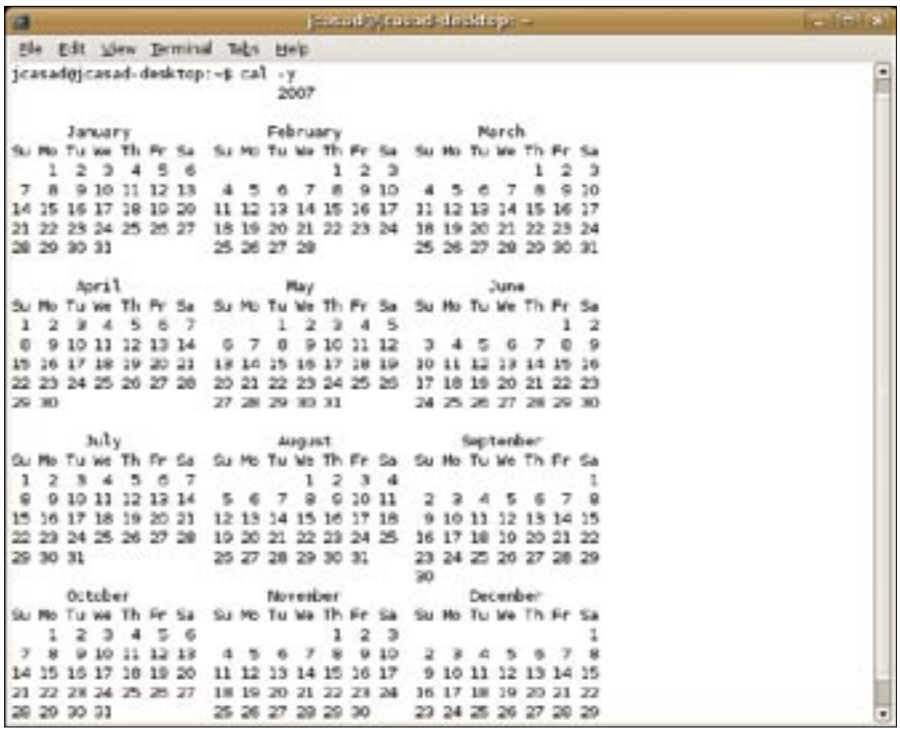

Figure 2: Thanks to Cal, you can display a calendar for the whole year at the console.

and re-parse the settings after saving them by giving the *source ~/.bashrc* command. You can then simply type *NY* to output New York time at the command line.

Formatted Output

The Date program has a large number of parameters that influence the output format. You can format the *date* output with a plus sign, followed by a percent sign, and a letter.

For example:

```
$ date +%Y_%m
2007_04
```

Table 1 lists some of the more common options; you can refer to the man page (*man date*) for a complete list of the options.

These formatting options are particularly practical if you use Date to automatically generate file names made up of date and/or time values.

The command:

```
tar -cvjf backup_$➥
(date +%d_%m_%Y).tar.bz2 *
```

creates a Bzip2 compressed tarball with a name comprising the text string *backup_*; the date – that is the day, month, and year separated by underlines; and the file extension *.tar.bz2* (for example, *backup_25_04_2007.tar.bz2*).

Setting the System Time

The root user can use Date to set the time and date for a machine.

To do so, you just enter an absolute time string by appending the data as a 12-digit number to the Date command line – two digits for the month, two for the day, two for the hours, two for the minutes.

The four-digit number for the year comes at the end:

```
# date 060916101973
Sat Jun 9 16:10:00 CET 1973
```

The first three parts of this are mandatory; if you leave out the year, *date* will just default to the current year.

If you would like to set the date with seconds precision, you can do so by entering the seconds as a two-digit number after the year.

Everything Is Relative

As an alternative to the absolute date and time, Date also understands relative values and even has a couple of predefined strings to help you:

- *yesterday*
- *tomorrow*
- *today*

Figure 3: Convenient documentation at the command line.

- *now*
- *sec(s)/second(s)*
- *min(s)/minute(s)*
- *hour(s)*
- *day(s)*
- *week(s)*
- *fortnight*
- *month(s)*
- *year(s)*

Additionally, *date* understands concepts such as *ago*, so you can say *day ago* instead of *yesterday*.

If you use one of these strings to set the time, you must specify the *-s* parameter like so:

```
# date -s '+3 mins' ➥
Wed Apr 25 18:03:44 CEST 2007
```

To display a relative time, you need the *-d* parameter instead:

```
# date -d '+5 days -2 hours'
Wed Apr 30 16:03:44 CEST 2007
```

The *date* info page tells you more about strings and how to use them.

Documentation

To read the documentation at the command line, you use *info coreutils date* (see Figure 3) or, in the KDE Konqueror file manager, you enter the URI *info:/coreutils/Date input formats*.

Setting the Hardware Clock

Besides the software clock, your computer also has another timekeeper, and one that will even keep on counting down the days when your computer is switched off and not even connected to the mains.

To ensure uninterrupted timekeeping, computer mainboards have a battery-buffered clock, referred to as the CMOS clock, RTC (Real Time Clock), BIOS clock, or even hardware clock.

hwclock program

The *hwclock* program lets you read and set the hardware clock; the commands all require root privileges.

Used in combination with the *-r* option, you can display the local hardware time like so:

```
# hwclock -r
Wed 25 Apr 2007 ➥
19:24:26 CEST ➥
-0.435565 seconds
```

Additionally, Hwclock has options for setting the system time to reflect the hardware clock time (*hwclock -s*) or vice versa (*hwclock -w*).

A combination of *--set* and *--date* sets a specific time. You need to enter a string to describe the new date and time after the *--date* parameter.

The format is exactly the same as the Date program's *-s* option.

The commands:

```
# hwclock --set ➥
--date="+2 hours"
```

set the hardware clock to a time two hours in the future.

Automated!

Network Time Protocol (NTP) is a standard for automating the synchronization of clocks in computer systems. The time signal propagates over the network from an NTP server to a client. You can configure the point in time when your Linux machine's NTP client contacts a server on the network.

This could be at boot time, when you get onto the Internet, or you could use a manual command in the shell.

Distributions

Various distributions have different approaches to installing and setting up NTP. OpenSUSE stores the software in a package called *xntp*.

For example, for Debian, you need to install the *ntp* package. Users with openSUSE can configure NTP using YaST in the *Network Services* tab.

For Debian, you need to edit the */etc/ntp.conf* configuration file, preferably with a text editor.

As an alternative, you can become root and give the *ntpdate* command.

When you run the tool, you need to pass in the name of the server to contact, as in:

```
# ntpdate de.pool.ntp.org
25 Apr 23:58:21 ➥
ntpdate[24405]: ➥
adjust time server ➥
85.25.141.60 offset ➥
0.037843 sec
```

Ntpdate might conflict with an NTP daemon running on your machine; in this case, you need to disable the daemon first. On Debian GNU/Linux, you can become root and type */etc/init.d/ntp stop* to do this. ■

INFO

[1] "Environment Variables" by Heike Jurzik, *Linux Magazine*, July 2007, pg. 89, *http://www.linuxpromagazine.com/issue/80*

THE AUTHOR

Heike Jurzik studied German, Computer Science and English at the University of Cologne, Germany. She discovered Linux in 1996 and has been fascinated with the scope of the Linux command line ever since. In her leisure time you might find Heike hanging out at Irish folk sessions or visiting Ireland.