Managing Accurate Date and Time

Avi Alkalay

avi at unix.sh
avi at br.ibm.com

Senior IT and Software Architect :: OpenSource/Linux Solutions
IBM Linux Impact Team :: ibm.com/linux

Kent Borg − Suggestion to use ntpq instead of ntpdc

Yura Moron − Good explanations on ntpq and ntpdc info

1.0 :: 2002/04/28

Revision History

Revision 1.0	28 Apr 2002	Revised by: avi
Finalized image.		
Revision 0.8	27 Apr 2002	Revised by: avi
Switched from ntpdc example to ntpq, based on contributions.		
Revision 0.8.1	20 Apr 2002	Revised by: avi
Improved graphic. Links to other doc locations.		
Revision 0.8	14 Apr 2002	Revised by: avi
Beter tunning of NTP graphic.		
Revision 0.76	13 Apr 2002	Revised by: avi
Inclusion of architecture graphic.		
Revision 0.75	10 Apr 2002	Revised by: avi
Spell checked. Using DocBook XSLT 1.50.		
Revision 0.65	31 Mar 2002	Revised by: avi
Upgraded to XML 4.1.2 DocBook format		
Revision 0.6	29 Mar 2002	Revised by: avi
Finished timezone mechanism on Linux. Created appendix. Now only drawings are needed.		
Revision 0.4	24 Mar 2002	Revised by: avi
All skeleton defined. All command examples written.		
Revision 0.2	19 Mar 2002	Revised by: avi
First DocBook version		

Table of Contents

1. Computer Global Date and Time Concept

To determine the current time for some planet region, a computer needs exactly this two informations:

1. Correct UTC (universal time as in Greenwich, but not GMT) time
2. Region's current Time Zone

For computers, there is also the hardware clock, which is used as a base by the OS to set his time.

OS date and time (we'll use only date *or* time from now on) is set on boot, by some script that reads the hardware clock, makes Time Zone calculations (there is no time zone data stored in BIOS) and sets the OS. After this synchronization, BIOS and OS time are completely independent. So after a while they may have some seconds of difference. Which one is correct? If you don't make special configurations, none.

We'll discuss here how to make them both *globally 100% accurate*.

2. What are Time Zones?

Time Zones are a geophysical world globe division of 15^o each, starting at Greenwich, in England, created to help people know what time is it now in another part of the world.

Nowadays it is much more a political division than geophysical, because sometimes people needs to have the same time as other people in not−so−far locations. And for energy savings reasons, we have today the Daylight Savings Time, that are also a Time Zone variation.

Time Zones are usually defined by your country government or some astronomical institute, and is represented by 3 or 4 letters. See Section 2.2 for examples.

If you want to know what time is it now in a different world region, you can use the timezoneconverter.com website.

2.1. Daylight Savings Time

For energy savings reasons, governments created the Daylight Savings Time. Our clocks are forwarded one hour, and this makes our days look longer. In fact, what really happens is only a Time Zone change. The primitive time (UTC) is still, and will allways be, the same.

Later we'll see how to enable and disable DST automatically in Linux.

2.2. Time Zones Examples

There is nothing better than examples:

Table 1. Brazilian Time Zones. Shifts relative to UTC

Name and Shift	DST Name and Shift	Locations
BREST −2:00	BREDT −1:00	Fernando de Noronha
BRST −3:00	BRDT −2:00	São Paulo, Rio, Brasilia, Minas Gerais, North East Region, South Region,etc
BRWST −4:00	BRWDT −3:00	West Region
BRAST −5:00	BRADT −4:00	Acre

Please send me contributions like this table for US Time Zone.

2.3. Time Zone Mechanism on Linux

Linux systems uses the GLIBC dynamic Time Zones, based on `/etc/localtime`. This file is a link to (or a copy of) a zone information file, usually located under `/usr/share/zoneinfo` directory.

From a geophysical perspective, there is only $360^o/15^o$=24 Time Zones in the world. But to make things easy to people, and to accommodate all the political variations (like Daylight Savings Time), you'll find hundreds of zoneinfo files in `/usr/share/zoneinfo`, each for every world city, country, etc, and its not complete (it can never be).

Some countries, like Brazil, don't have a fixed day to start Daylight Savings Time. It is defined every year, a couple of months before summer, and you may end up in a situation you'll have to change your zoneinfo file, which was compiled by **zic** from a text file like this.

Example 1. Brazilian Zone Info text file

```
# Brazil Time Zones
#
# Brazilian Time Zones are:
# BREST: East of Brasilia. Fernando de Noronha.
# BRST:  Brasilia, São Paulo, Rio, Northeast, South etc
# BRWST: West of Brasilia. Mato Grosso, Manaus
# BRAST: Acre.
#
# In daylight saving time, letter 'S' changes to 'D'.
#
# To install, make:
#
# # zic Brazil.txt
#
# Zone files will be installed in /usr/share/zoneinfo (depends on your
# distribution). Then, make a symbolic link from your zone to /etc/localtime:
#
# # ln −sf /usr/share/zoneinfo/Brazil/Brasilia /etc/localtime
#
#
# If you have updates and new standards to this file please send to
#
# Avi Alkalay <avi @ unix.sh>
#
# Last update: 18 Nov 2000
# This file is available at http://avi.alkalay.net/linux/zoneinfo/
#

# Rule  NAME    FROM  TO    TYPE  IN    ON   AT    SAVE   LETTER/S
Rule    Brazil  1931  1932  −     Oct   3    0:00  1:00   D
Rule    Brazil  1932  1933  −     Mar   31   0:00  0      S
Rule    Brazil  1949  only  −     Dec   1    0:00  1:00   D
Rule    Brazil  1950  only  −     Apr   30   0:00  0      S
Rule    Brazil  1950  1952  −     Dec   1    0:00  1:00   D
Rule    Brazil  1951  only  −     Apr   16   0:00  0      S
Rule    Brazil  1952  only  −     Mar   31   0:00  0      S
Rule    Brazil  1953  only  −     Feb   28   0:00  0      S
Rule    Brazil  1963  only  −     Oct   23   0:00  1:00   D
Rule    Brazil  1964  only  −     Mar   1    0:00  0      S
Rule    Brazil  1965  only  −     Jan   31   0:00  1:00   D
Rule    Brazil  1965  only  -     Mar   31   0:00  0      S
Rule    Brazil  1965  only  -     Dec   1    0:00  1:00   D
Rule    Brazil  1966  1968  -     Mar   1    0:00  0      S
Rule    Brazil  1966  1967  -     Nov   1    0:00  1:00   D
Rule    Brazil  1984  only  -     Nov   2    0:00  1:00   D
Rule    Brazil  1985  only  -     Mar   15   0:00  0      S
Rule    Brazil  1985  only  -     Nov   2    0:00  1:00   D
Rule    Brazil  1986  only  -     Mar   15   0:00  0      S
Rule    Brazil  1986  only  -     Oct   25   0:00  1:00   D
Rule    Brazil  1987  only  -     Feb   14   0:00  0      S
Rule    Brazil  1987  only  -     Oct   25   0:00  1:00   D
Rule    Brazil  1988  only  -     Feb   7    0:00  0      S
Rule    Brazil  1988  only  -     Oct   16   0:00  1:00   D
Rule    Brazil  1989  only  -     Jan   29   0:00  0      S
Rule    Brazil  1989  only  -     Oct   15   0:00  1:00   D
Rule    Brazil  1990  only  -     Feb   11   0:00  0      S
Rule    Brazil  1990  only  -     Oct   21   0:00  1:00   D
Rule    Brazil  1991  only  -     Feb   17   0:00  0      S
Rule    Brazil  1991  only  -     Oct   20   0:00  1:00   D
Rule    Brazil  1992  only  -     Feb   9    0:00  0      S
Rule    Brazil  1992  only  -     Oct   25   0:00  1:00   D
Rule    Brazil  1993  only  -     Jan   31   0:00  0      S
Rule    Brazil  1993  only  -     Oct   17   0:00  1:00   D
Rule    Brazil  1994  only  -     Feb   20   0:00  0      S
Rule    Brazil  1994  only  -     Oct   16   0:00  1:00   D
Rule    Brazil  1995  only  -     Feb   19   0:00  0      S
Rule    Brazil  1995  only  -     Oct   15   0:00  1:00   D
Rule    Brazil  1996  only  -     Feb   11   0:00  0      S
Rule    Brazil  1996  only  -     Oct   06   0:00  1:00   D
Rule    Brazil  1997  only  -     Feb   16   0:00  0      S
Rule    Brazil  1997  only  -     Oct   06   0:00  1:00   D
Rule    Brazil  1998  only  -     Mar   01   0:00  0      S
Rule    Brazil  1998  only  -     Oct   11   0:00  1:00   D
Rule    Brazil  1999  only  -     Feb   21   0:00  0      S
Rule    Brazil  1999  only  -     Oct   03   0:00  1:00   D
Rule    Brazil  2000  only  -     Feb   27   0:00  0      S
Rule    Brazil  2000  only  -     Oct   8    0:00  1:00   D
Rule    Brazil  2001  only  -     Feb   18   0:00  0      S


# Zone  NAME                    GMTOFF  RULES/SAVE      FORMAT  [UNTIL]
Zone    Brazil/DeNoronha        -2:00   Brazil          BRE%sT
Zone    posix/Brazil/DeNoronha  -2:00   Brazil          BRE%sT
Zone    right/Brazil/DeNoronha  -2:00   Brazil          BRE%sT
Zone    Brazil/East             -2:00   Brazil          BRE%sT
Zone    posix/Brazil/East       -2:00   Brazil          BRE%sT
Zone    right/Brazil/East       -2:00   Brazil          BRE%sT

Zone    America/Sao_Paulo       -3:00   Brazil          BR%sT
Zone    America/Rio_de_Janeiro  -3:00   Brazil          BR%sT
Zone    America/Brasilia        -3:00   Brazil          BR%sT
Zone    posix/America/Sao_Paulo -3:00   Brazil          BR%sT
Zone    posix/America/Rio_de_Janeiro -3:00   Brazil     BR%sT
Zone    posix/America/Salvador  -3:00   Brazil          BR%sT
Zone    posix/America/Brasilia  -3:00   Brazil          BR%sT
Zone    posix/Brazil/Central    -3:00   Brazil          BR%sT
Zone    posix/Brazil/Brasilia   -3:00   Brazil          BR%sT
Zone    posix/Brazil/Sao_Paulo  -3:00   Brazil          BR%sT
Zone    posix/Brazil/Salvador   -3:00   Brazil          BR%sT
Zone    posix/Brazil/Rio_de_Janeiro  -3:00   Brazil     BR%sT
Zone    right/America/Sao_Paulo -3:00   Brazil          BR%sT
Zone    right/America/Rio_de_Janeiro -3:00   Brazil     BR%sT
Zone    right/America/Salvador  -3:00   Brazil          BR%sT
Zone    right/America/Brasilia  -3:00   Brazil          BR%sT
Zone    right/Brazil/Central    -3:00   Brazil          BR%sT
Zone    right/Brazil/Brasilia   -3:00   Brazil          BR%sT
Zone    right/Brazil/Sao_Paulo  -3:00   Brazil          BR%sT
Zone    right/Brazil/Salvador   -3:00   Brazil          BR%sT
Zone    right/Brazil/Rio_de_Janeiro   -3:00   Brazil    BR%sT
Zone    Brazil/Central          -3:00   Brazil          BR%sT
Zone    Brazil/Brasilia         -3:00   Brazil          BR%sT
Zone    Brazil/Sao_Paulo        -3:00   Brazil          BR%sT
Zone    Brazil/Rio_de_Janeiro   -3:00   Brazil          BR%sT
Zone    Brazil/Salvador         -3:00   Brazil          BR%sT

Zone    Brazil/West             -4:00   Brazil          BRW%sT
Zone    Brazil/Manaus           -4:00   Brazil          BRW%sT
Zone    Brazil/Rondonia         -4:00   Brazil          BRW%sT
Zone    Brazil/Roraima          -4:00   Brazil          BRW%sT
Zone    Brazil/Mato_Grosso      -4:00   Brazil          BRW%sT
Zone    posix/Brazil/Manaus     -4:00   Brazil          BRW%sT
Zone    posix/Brazil/Mato_Grosso -4:00  Brazil          BRW%sT
Zone    right/Brazil/Manaus     -4:00   Brazil          BRW%sT
Zone    right/Brazil/Mato_Grosso -4:00  Brazil          BRW%sT
Zone    posix/America/Manaus    -4:00   Brazil          BRW%sT
Zone    right/America/Manaus    -4:00   Brazil          BRW%sT

Zone    Brazil/Acre             -5:00   Brazil          BRA%sT
```

The **Rule** block defines the date and time we change the Time Zone, while in the **Zone** block we reference the **Rule** will manage it. Note that the **Zone** name is actually the file name under `/usr/share/zoneinfo` directory, and here we defined several different names for the same Time Zone, just to be easyer for people to find their zone.

This file's comments explains how to install these time zones, using the **zic** zoneinfo compiler (which already installs them also). To make it effective, you only have to link (or copy) the zoneinfo file to `/etc/localtime`. In some distributions, there is a higher level (and preferred) way to set the Time Zone, described in Section 3.1.

After making `/etc/localtime` pointing to the correct zoneinfo file, you are already under that zone rules and DST changes are automatic −− you don't have to change time manually.

The following commands sequence shows Linux Time Zone mechanics dynamism. Note they were all issued in less than one minute:

```
bash$ ls -al /etc/localtime
lrwxrwxrwx  1 root root 35 May 22  2001 /etc/localtime -> /usr/share/zoneinfo/Brazil/Brasilia
bash$ date
Fri Mar 29 20:13:38 BRST 2002
bash# ln -sf /usr/share/zoneinfo/GMT /etc/localtime
bash$ date
Fri Mar 29 23:13:47 GMT 2002
bash# ln -sf /usr/share/zoneinfo/Brazil/Brasilia /etc/localtime
bash$ date
Fri Mar 29 20:14:03 BRST 2002
```

At 20:13, I was in my default brazilian Time Zone (BRST), then I switched to GMT and my system time changed to 23:13! When your Time Zone enters DST, you'll see a similar effect, but the rules are all inside your Time Zone (`/etc/localtime` link doesn't change like this example).

An application running in this machine (eg. web−server generating access logs) will feel this change, so it is very important for developers to remember that the full Time Concept is the current *time* plus current *Time Zone*, as described in Section 1.

In the end, I switched back to my correct Time Zone.

---

3. The Correct Settings for Your Linux Box

For any OS installation, you must know your Time Zone. This is expressed in terms of a city, a state or a country. You must also decide how to set BIOS time, and we may follow two strategies here:

Linux Only Machine

> In this case you should set BIOS time to UTC time. DST changes will be dynamically managed by Time Zone configurations.

Dual Boot Linux and MS Windows Machine

> Windows handles time in a more primitive way than Linux. For Windows, BIOS time is allways your local time, so DST changes are more aggressive because they directly change hardware clock. And since both Linux and Windows initially get and set time from the hardware, when they are together, Linux must handle it in the same way. So set BIOS time to your localtime.

3.1. Setting Time Zone

On Red Hat Linux and derived systems, you can set the hardware clock strategy and Time Zone using the **timeconfig** command, that shows a user−friendly dialog. You can also use it non−interactively:

Example 2. Time Configuration Tool

```
bash# timeconfig "Brasil/East"   # set HC to localtime, and TZ to "Brazil/East"
bash# timeconfig --utc "Brasil/East"   # set HC to UTC, and TZ to "Brazil/East"
```

Anyway, it changes `/etc/sysconfig/clock` file that is read at boot time. You can edit it by hand, and that is how it looks:

Example 3. `/etc/sysconfig/clock` file

```
ZONE="Brazil/East"
UTC=true
ARC=false
```

3.2. Setting the Hardware Clock

I encourage you to only set your HC after understanding how to get accurate time, described on Section 4.

The **hwclock** command reads and sets the HC, based on several options you give him, documented in its man page. But you don't have to use it if you have a modern Linux distribution. After defining your HC strategy and Time Zone, you can use the high level **setclock** command to correctly set your HC. You don't need to pass any parameters because **setclock** intelligently calls **hwclock** to set the BIOS based on your OS current date and time. *So you should allways use the **setclock** command.*

But if you are a minimalist and prefer hard work, here are some **hwclock** examples:

Example 4. setclock and hwclock usage

```
bash# setclock                                # The easy way to set HC
bash# hwclock                                 # reads HC
bash# hwclock --systohc --utc                 # set HC with UTC time based on OS current time
bash# hwclock --systohc                       # set HC with local time based on OS current tim
bash# hwclock --set --date "22 Mar 2002 13:17"  # set HC with time specified on string
```

Since the OS time is independent from the hardware clock, any BIOS change we make will take place in the next boot.

Another option to change HC is rebooting and accessing your computer BIOS screens. On IBM e−server zSeries platforms you'll have to do it on z/VM level, because Linux here runs on virtual machines created by z/VM.

4. Accurate Global Time Synchronization

To have accurate time in all your systems is as important as having a solid network security strategy (achieved by much more than simple firewall boxes). It is one of the primary components of a system administration based on good practices, which leads to organization and security. Specially when administering distributed applications, web−services, or even a distributed security monitoring tool, accurate time is a must.

4.1. NTP: The Network Time Protocol

We'll not discuss here the protocol, but how this wonderfull invention, added to the pervasiveness of the Internet, can be usefull for us. You can find more about it at www.ntp.org.

Once your system is setup, NTP will manage to keep its time accurate, making very small adjustments to not impact the running applications.

People can get exact time using hardware based on atom's electrons frequency or cyclotrons. There is also a method based on GPS (Global Positioning System). The first is more accurate, but the second is pretty good also. Both require very special and expensive equipment, but their owners (usually universities and research labs) connects them to computers, that run an NTP daemon, that are connected to the Internet, that finally lets us access it for free. And this is how we'll synchronize our systems.

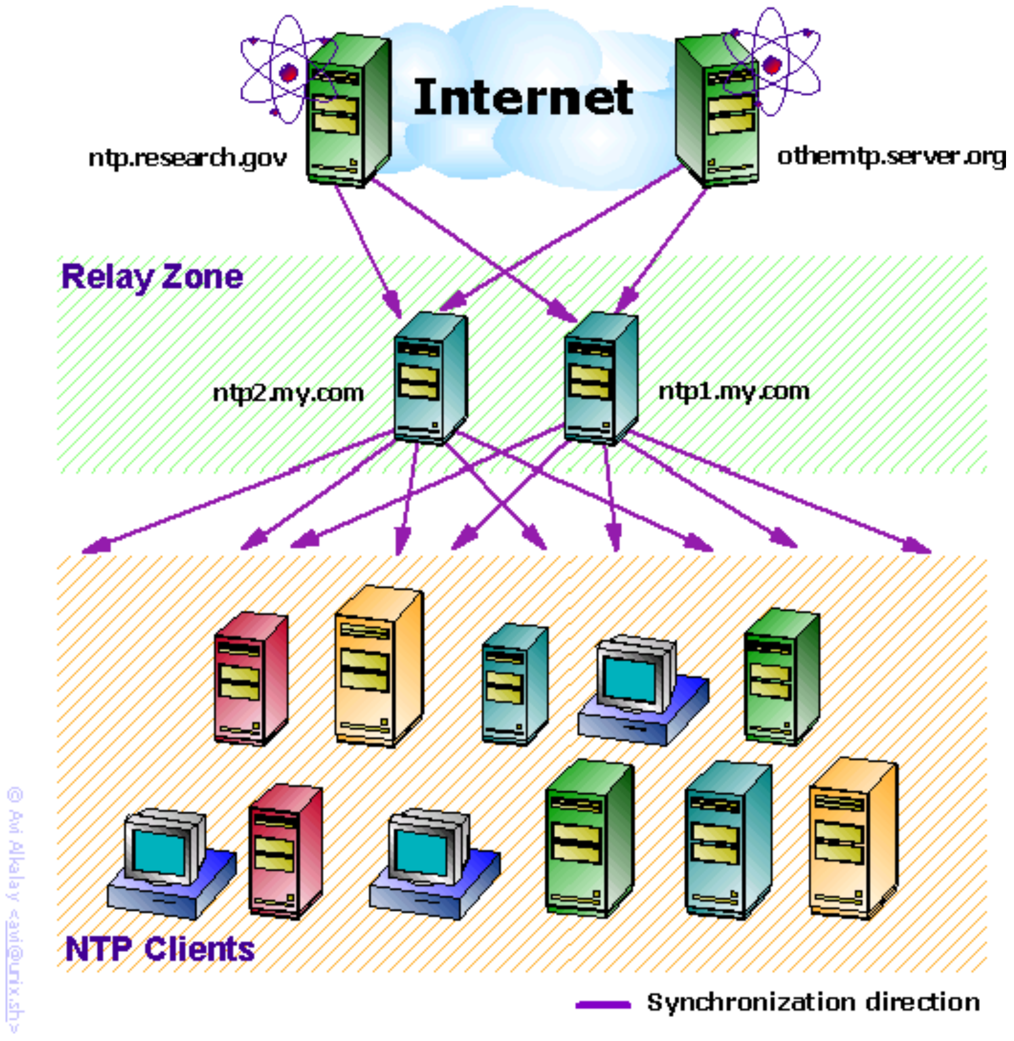
4.2. Building a Simple Time Synchronization Architecture

You will need:

1. A direct or indirect (through a firewall) connection to the Internet.
2. Choose some NTP servers. There is a list of public time servers on NTP website. If you don't have an Internet access, your WAN administrator (must be a cleaver guy) can provide you some internal addresses.
3. Have the NTP package installed in all systems you want to synchronize. You can find RPMs in your favorite Linux distribution CD, or make a search on rpmfind.net.

Here is an example of good architecture:

Figure 1. Local Relay Servers for NTP



If you have several machines to synchronize, *do not* make them all access the remote NTP servers you choosed. Only 2 of your server farm's machines must access remote NTP servers, and the other machines will sync with these 2. We will call them the *Relay Servers*.

Your Relay Servers can be any machine already available in your network. NTP consumes low memory and CPU. You don't need to have a dedicated machine for it.

It is a good idea to create hostname aliases for your local Relay Servers like ntp1.mydomain.com and ntp2.maydomain.com, and use only these names when configuring the client machines. This way you can move the NTP functionality to a new Relay Server (with a different IP and hostname), without having to reconfigure the clients. Ask your DNS administrator to create this aliases.

4.3. NTP Configurations

For Your Relay Servers

Edit `/etc/ntp.conf` and add the remote servers you choosed:

Example 5. Relay machines `/etc/ntp.conf`

```
.
.
server  otherntp.server.org     # A stratum 1 server at server.org
server  ntp.research.gov        # A stratum 2 server at research.gov
.
.
```

For Your Clients

Edit `/etc/ntp.conf` and add your Relay Servers with a standard name:

Example 6. Client machines `/etc/ntp.conf`

```
.
.
server  ntp1.my.com             # My first local relay
server  ntp2.my.com             # My second local relay
.
.
```

If your machine has a UTC time difference bigger than some minutes comparing to the NTP servers, NTP will not work. So you must do a first full sync, and I recommend you to do it in a non−production hour. You need to do it only when you are making the initial NTP setup. Never more:

Example 7. First sync

```
bash# ntpdate otherntp.research.gov❶
24 Mar 18:16:36 ntpdate[10254]: step time server 200.100.20.10 offset −15.266188 sec
bash# ntpdate otherntp.research.gov❷
24 Mar 18:16:43 ntpdate[10255]: adjust time server 200.100.20.10 offset −0.000267 sec
```

❶
First full sync. We were 15 seconds late.

❷
Second full sync, just to be sure. Now we are virtually 0 seconds late, which is good.

The last step is to start or restart the NTP daemons in each machine:

```
bash# service ntpd restart
```

4.4. Watching Your Box Synchronizing

Now you have everything setup. NTP will softly keep your machine time synchronized. You can watch this process using the NTP Query (**ntpq** command:

Example 8. A time synchronization status

```
bash# ntpq −p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
−jj.cs.umb.edu   gandalf.sigmaso  3 u   95 1024  377   31.681  −18.549   1.572
 milo.mcs.anl.go ntp0.mcs.anl.go  2 u  818 1024  125   41.993  −15.264   1.392
−mailer1.psc.edu ntp1.usno.navy.  2 u  972 1024  377   38.206   19.589  28.028
−dr−zaius.cs.wis ben.cs.wisc.edu  2 u  502 1024  357   55.098    3.979   0.333
+taylor.cs.wisc. ben.cs.wisc.edu  2 u  454 1024  347   54.127    3.379   0.047
−ntp0.cis.strath harris.cc.strat  3 u  507 1024  377  115.274   −5.025   1.642
*clock.via.net   .GPS.            1 u  426 1024  377  107.424   −3.018   2.534
 ntp1.conectiv.c 0.0.0.0         16 u    − 1024    0    0.000    0.000 4000.00
+bonehed.lcs.mit .GPS.            1 u  984 1024  377   25.126    0.131  30.939
−world.std.com   204.34.198.40    2 u  119 1024  377   24.229   −6.884   0.421
```

The meaning of each column

remote

Is the name of the remote NTP server. If you use the `−n` switch, you will see the IP addresses of these servers instead of their hostnames.

refid

Indicates where each server is getting its time right now. It can be a server hostname or something like .GPS., indicating a Global Positioning System source.

st

Stratum is a number from 1 to 16, to indicate the remote server precision. 1 is the most accurate, 16 means server unreachable. Your Stratum will be equal to the less accurate remote server plus 1.

poll

The polling interval (in seconds) between time requests. The value will range between the minimum and maximum allowed polling values. Initially the value will be smaller to allow synchronization to occur quickly. After the clocks are 'in sync' the polling value will increase to reduce network traffic and load on popular time servers.

reach

This is an octal representation of an array of 8 bits, representing the last 8 times the local machine tried to reach the server. The bit is set if the remote server was reached.

delay

The amount of time (seconds) needed to receive a response for a "what time is it" request.

offset

The most important value. The difference of time between the local and remote server. In the course of synchronization, the offset time lowers down, indicating that the local machine time is getting more accurate.

jitter

Dispersion, also called Jitter, is a measure of the statistical variance of the offset across several successive request/response pairs. Lower dispersion values are preferred over higher dispersion values. Lower dispersions allow more accurate time synchronization.

The meaning of the signs before server hostname

–

Means the local NTP service doesn't like this server very much

+

Means the local NTP service likes this server

x

Marks a bad host

*

Indicates the current favorite

4.5. Configure to Automatically Run NTP at Boot

You may want to have NTP running all the time even if you reboot your machine. On each machine, do the following:

```
bash# chkconfig --level 2345 ntpd on
```

This will ensure autostart.

If your machine is up and running for a long time (months, years) without rebooting, you'll find a big discrepancy between the inaccurate hardware clock and the (now very accurate) system time. Modern Linux distributions copy OS time to the HC everytime the system is shutdown, using a mechanism similar to the **setclock** command. This way, in the next OS boot, you'll get date and time almost as accurate as it was when you shutdown.

A. About this Document

This document is published in the following locations:

- Main distribution [XML Source]
- LinuxDoc, as a HOWTO [single page] [PDF]