Content block expected for the “container” directive; none found.

```
.. container:: navbar
```

Time

Time waits for no man. Or it flies like an arrow. Something like that. But still keeping accurate time is as important now as it was when Harrison built his first clock. In a networked world, having one machine with accurate time is not the worst problem - all the machines we use need to be synchronised to the same time, else trying to compare logs on different machines is quite simply a nightmare.

The solution to both these problems is called Network Time Protocol or NTP. NTP is a way for any server to get the current time from an atomic clock, even if it is thousands of miles away. Atomic clocks keep, what is simply the most accurate time humankind can keep. These clocks then are used to keep *stratum 1* servers up to date. These stratum 1 servers are the starting point for NTP, and are primarily run by academic and governmental institutions. They are the source point for, rather boringly, *stratum 2* servers. These stratum 2 servers are run, often by volunteers to provide you and I with a machine that knows precisely what the time is, and more importantly, are willing to tell us.

Whilst the servers themselves are run in wildly different locations, and by different organisations, a sensible DNS round-robin is in place.

```
Name:   pool.ntp.org
Address: 64.202.112.75 (ntp.your.org.)

Name:   pool.ntp.org
Address: 38.99.80.156 (clock.fihn.net.)
```

Here two servers run by totally different organisations are pulled up when I ask for pool.ntp.org. So all I need to remember is one dns name and my ntp requests will alwasy find a suitable match. This is good news for me, but also very good news for Poul Henning Kemp[1]

Anyway, NTP works in a clever and comlicated way, but the Simple IT manager can get by with a good enough explanation, and a reference for more info. The good enough explanation is that the NTP client asks for a time check and waits for the answer to come back several times. With enough times and a clever algorithm, the latency in the network can be estimated and that latency can be added to the time in the received message to give an accurate time *right now*.

```
 NTPv4 can usually maintain time to within 10 millisec-
onds (1/100 s) over the public Internet, and can achieve accura-
cies of 200 microseconds (1/5000 s) or better in local area networks un-
der ideal conditions.

http://en.wikipedia.org/wiki/Network_Time_Protocol
```

How to get the time right on your server

1. Set your current timezone
2. set the time correctly from an atomic clock
3. keep on adjusting it

1. Set your current timezone In /usr/share/zoneinfo are a series of binary files that represent to the server how to calculate the timezone - so /usr/share/zoneinfo/Europe/London tells us how to adjust for the GMT zone. We want to copy the one that is right for our location in to /etc/localtime

diff /etc/localtime /usr/share/zoneinfo/Europe/London

(there may not be a /etc/localtime file. In fact if this is the first install, there wont be)

So let us have some fun:

```
manhattan# cp /usr/share/zoneinfo/Asia/Tokyo  /etc/localtime
manhattan# adjkerntz -a
manhattan# date
Sat Aug 16 20:16:02 JST 2008
manhattan#
manhattan#
manhattan# cp /usr/share/zoneinfo/Europe/London /etc/localtime
manhattan# adjkerntz -a
manhattan# date
Sat Aug 16 12:16:23 BST 2008
```

2. clock synchronisation

Ok, we can control what timezone the machine thinks it is in, but how do we get the time correct? The configuration file for ntp daemon is /etc/ntpd.conf

put this into /etc/ntpd.conf:

```
driftfile /var/lib/ntp/ntp.drift

server 0.pool.ntp.org
server 1.pool.ntp.org
server 2.pool.ntp.org
server pool.ntp.org
```

(NB there is a diff between the location in freebsd handook and the ntp ssite for conf file ntpd.conf vs ntp.conf. ntp.conf is right - follow the handbook)

This tells the server to use a drift file, which is how ntpd calculates the average latency for its requests, and which servers it should contact - if you wanted a really minimalist conf file, the last line will suffice.

Enable the server at boot time:

```
ntpd_enable="YES" to /etc/rc.conf
```

TO force the server to change to a new time, especially if the time is a long way out, then we can use

ntpdate pool.ntp.org

ntpdate forces a time update no matter how far out the server is - this is useful, but you should rely on ntpd to correct the time - if it finds itself correcting large leaps it will scream to the logs. This is good - there are only two reasons for regular large corrections, either your motherboards clock is dying or you have invented time travel.

Sample rc.conf file with a full time set on boot:

```
ntpdate_enable="YES"
ntpdate_flags="pool.ntp.org"
#enable ntpd after ntpdate - ntpdate does nothing if  ntpd is running.
ntpd_enable="YES"
```

Notes

Explicit markup ends without a blank line; unexpected unindent.

Fuller story is at http://www.lightbluetouchpaper.org/2006/04/07/when-firmware-attacks-ddos-by-d-link/

Bibliography

http://www.freebsd.org/doc/en/books/handbook/network-ntp.html http://support.ntp.org/bin/view/Servers/NTPPool

[1] Poul-Henning Kemp is a respected FreeBSD developer who ran a Stratum 1 Server. His server was referenced by a commercial domestic router sold in its millions, and by IP address. He was getting 3 million packets a day extra.