NTP and GPS

Using GPS as a time reference for NTP

LinuxSA, 2004-10-19
Adelaide, Australia

Glen Turner

aarnet
Australia's Academic
and Research Network

Copyright © AARNet, 2004

→ Topics

- GPS
- NTP configuration for GPS reference clock
- NTP archiecture
- NTP and security
- An NTP appliance
- [Coding daemons]

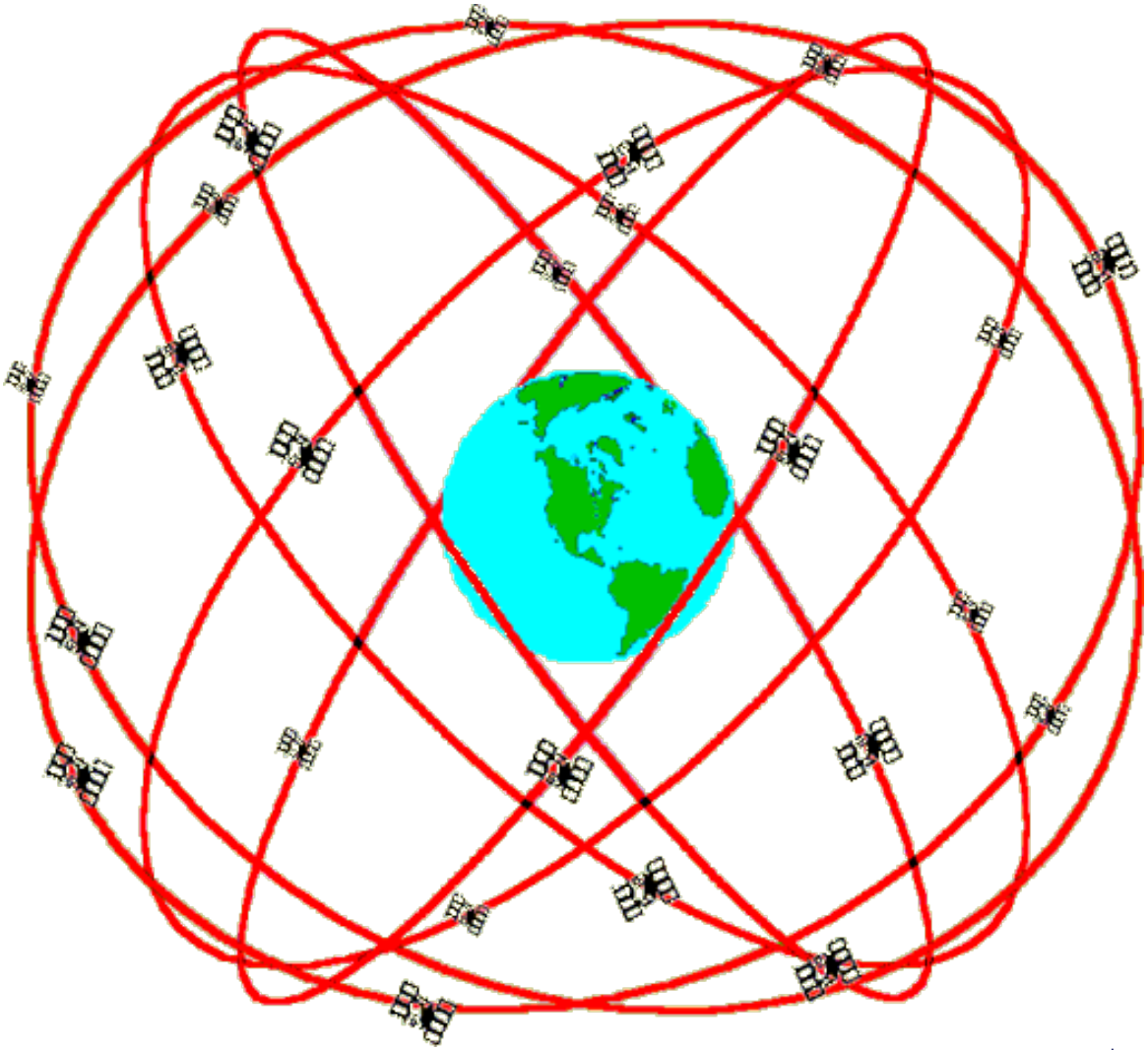
→Global Positioning System

aarnet

→GPS constellation

- GPS “space segment”
  - –24 to 30 low earth orbit satelites, currently 29
  - –6 orbital planes spaced 60° apart, 55° to equator, 20,200Km altitude, nominally with 4 satelites per plane
  - –So 5 to 8 satelites visible at any one time
    - Assuming visibility to both horizons
- Time signal on 1575.42MHz accurate to 200nS (or 340nS with selective availability)
  - –22m (or 100m) horizontal distance
  - –Another signal on 1227.60MHz for measuring propogation delay for military users

→GPS constellation



Peter Dana
www.colorado.edu/geography/gcraft/notes/gps

aarnet

→GPS control system

- Control segment
  - –Master control: 50th Space Wing, Schriever Air Force Base, CO
    - All USAF (and US Army) satelites controlled from here, including the main battle communications
    - 24×7 monitoring by 5 staff
    - Sync to USNO UTC Master Clock every 15 minutes, backup clock maintained is case link to USNO lost
  - –Monitor stations: Kwajalein, Hawaii, Ascension, Diego Garcia, Colorado Springs, Cape Canaveral

→National Marine Electronics Association 182

- NMEA-182 is a 4800bps asynchronous serial protocol, compatible with RS-232.
  - –Somewhat multidrop: one talker, multiple receivers
- Protocol defines “sentences” for various maritime measurements
  - –Basic format is a sentence type, followed by comma-seperated fields, followed by a * and checksum
  - –All ASCII with bit 7 set

```
$GPGGA,043539,3455.3002,S,13836.2963,E,1,04,14.0,62.1,M,-2.2,M,,*49
```

- Usually repeated once per n seconds

→$GPGGA: GPS fix data

```
$GPGGA,043539,3455.3002,S,13836.2963,E,1,04,14.0,62.1,M,-2.2,M,,*49
```

043539,	04:35:39 UTC
3455.3002,S,	34°55.3002'S
13836.2963,E,	138°36.2963'E
1,	Good GPS fix (0 = No fix, 2 = Differential)
04,	4 satelites being tracked
14.0,	14m horizontal dilution of position
62.1,M,	Altitude above sea level
-2.2,M,	Height of mean sea level above WGS84
,	Time since last Differential GPS input
,	Differential GPS station ID
*49	Checksum

→Pulse per second

- PPS output is raised once per second, rising upon the “tick” of the second
- An analogue signal
  - Control load so that rise time is not delayed
  - Load includes capacitance of cable, so cable needs to be kept within bounds
- Connect to an input that will cause an interrupt
  - Data Carrier Detect pin of RS-232 serial port
  - Acknowledge pin of Centronics parallel port

→Programming the GPS

- Turn off all output

  `$PGRM0,,2`

- Turn on GPS position output

  `$PGRM0,GPGGA,1`

- Enable pulse per second output

  `$PGRMC,,,,,,,,,,,,2`

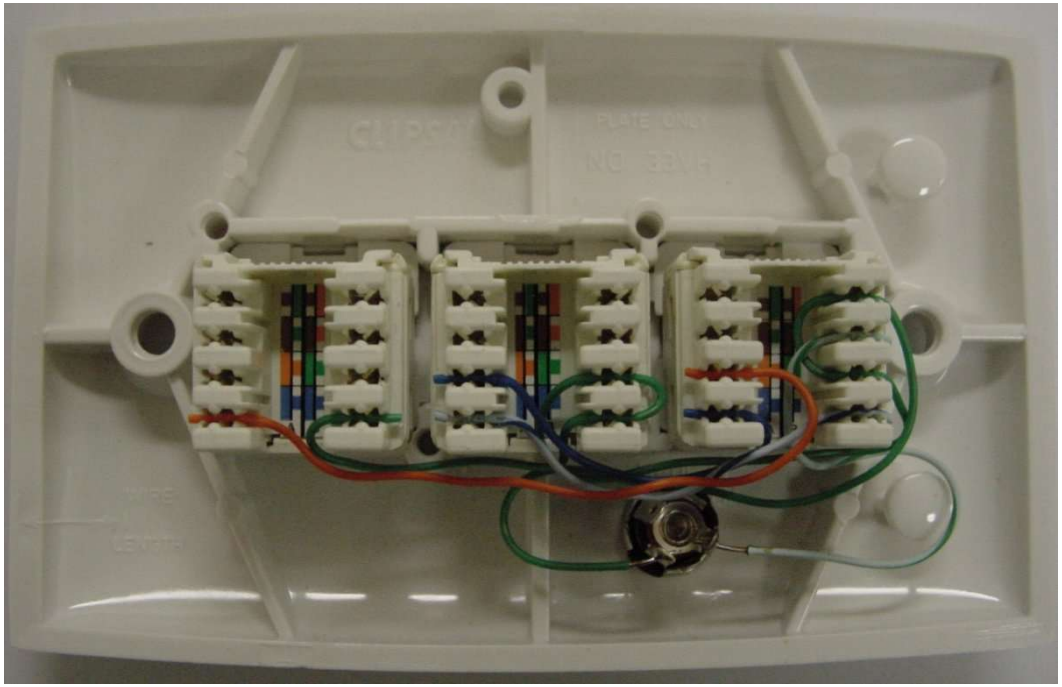
- Alter datum from WGS84 to to Geocentric Datum of Australia

  `$PGRMC,,,108`

→Interfacing the GPS

- Garmin GPS-16HVS has a 8 core cable carrying
  - DC input
  - NMEA input/output
  - PPS output
- Unshiedled twisted pair structured cabling gives a nice result
  - Build a “break out box” from Krone face plate
  - Use RJ-45/DB9 adapters for NMEA and PPS interfacing to PC
  - But don't use structured cabling building plant
    - Remember that we need to avoid long cable runs

→Break out box detail



aarnet

→NTP configuration for GPS reference clock

→Configuration: Device files

- Each NTP driver has a IP address, used in ntp.conf, and a device file, which points to the reference clock
- NMEA
  - 127.127.20.*x*
  - /dev/gps*x*
- In our case:

```
cd /dev
ln -s ./usb/ttyUSB0 gps0
```

→Configuration: /etc/ntp.conf for NMEA input

```
server 127.127.20.0 prefer mode 2
fudge 127.127.20.0 time1 0.0 stratum 2
```

`.20.0`	Unit 0 of clock type 20 (NMEA)
`mode 2`	Use $GPGGA sentences
`time1 0.0`	Calibration factor
`stratum 2`	Stratum less than PPS clock

→What about the PPS input?

- Use a PPS-enabled kernel
  - –Requires PPSkit patch from kernel.org
  - –For 2.4, and usually some minor versions behind
    - A pain when kernel updates fix security issues
- Interface PPS via the shared-memory interface
  - –Purely user-space approach
  - –Requires shm daemon from www.wraith.sf.ca.us
  - –This software needs work, which I've done

Configuration: /etc/ntp.conf for shared memory PPS input

```
server 127.127.28.0 prefer
fudge 127.127.28.0 time1 0.0 stratum 1
```

`.20.0`	Unit 0 of clock type 28 (shared memory)
`time1 0.0`	Calibration factor
`stratum 1`	The ultimate time source

→Configuration: Starting shm daemon

- DCD input on COM1:

```
shm -s /dev/ttyS0
```

- Ack input on Centronics

```
modprobe parport_pc
modprobe ppdev
shm -p /dev/parport0
```

→Debugging

- ntpq
  - peers
  - readlist
  - associations
- ntpdc
  - peers
  - kerninfo

→NTP architectures

→NTP principles

- Clock *filters* select best 8 recent samples from a peer
- *Selection and clustering* algorithms select *truechimers* and discard *falsetickers*
  - There can be no truechimers, if so then no stratum claims are made for the clock
  - Three truechimers can vote down one falseticker
- *Combining* creates a weighted average of surviving samples
- A phase- and frequency-locked loop forms a filter to *discipline* the computer's clock oscillator to the NTP time

→NTP design

- We want four sources of time
  - So a falseticker can be voted down
- Not all of these need to be precise
  - One stratum 1 and three stratum 2 is fine
- Peer each client to all four clocks
  - Scaling problem
    - Of network connections
    - Of authentication

→Distributing time: multicast or manycast

- Each clock talks into a multicast group
  - –224.1.1.1, prevent ingress and egress at the network edge
  - –It makes sense to use 244.1.1.1 for unauthenticated time and an administratively-scoped group for autokey authenticated time
- May be authenticated using pubilc key
  - –NTP has an autokey algorithm wich uses public keys for authentication, but not in the time-critical path
- Problem: most enterprise networks don't run multicast

→Distributing time: routers

- Many routers can act as NTP clients and servers
- These are attractive to use as the time traffic follows the most optimal path through the network
- Really only useful for distributing unauthenticated time
  - That is, for the average client

→Distributing time: unicast

- Distributing time by unicast is not scalable
- But it can do symmetric encryption
- And it does have wide platform support
- So it is useful for providing time to machines that must not be subverted: particularly authentication platforms such as Kerberos KDCs, LDAP servers and Windows Domain servers

→Other ways of distributing time: Samba

- Samba

```
[global]
  time server = yes
  # UTC + 9:30
  time offset = 570
```

- NetWare has a similar “set time on login” feature

→Other ways of distributing time: Daytime

- Daytime, provided by inetd, such as */etc/xinetd.d/daytime*

```
service daytime
 {
   type = INTERNAL
   id = daytime-stream
   socket_type = stream
   user = root
   wait = no
   disable = no
 }
```

- And similarly for UDP and the file */etc/xinetd.d/daytime-udp*

→Other ways of distributing time: TIMEP

- Mainly used by simple devices such as hubs
- See *intimed* v2 at http://goliat.eik.bme.hu/~balaton/inet/
- Newer network devices use Simple NTP, which is a small-footprint NTP client which operates against a standard NTP server

→What appliance manufacturers should do

- Run Simple NTP in unauthenticated broadcast client mode
- ISPs run NTP on routers in broadcast mode for customer subnets
- What some ADSL routers do
  - Hardcode NTP server addresses
  - This lead to a denial of service on the CSIRO NTP server, so it has been removed from service
  - That IP address is now unusable
  - When people complain, chose another NTP server!
- Irresponsible manufacturers
  - Netgear
  - SMC

→Time on Linux

- We're now running a lot of clocks
  - –The system time, maintained by regular CPU interrupts
    - Disciplined by NTP
    - Set from time of day clock on kernel boot
    - Maybe set by NTP step ticker on init boot, see */etc/ntp/step-ticker*
  - –The hardware time of day clock
    - Updated from the system time every 11m
    - The time of day clock may be in UTC or in system local time (for Windows & BIOS compatibility)

→Time zones

- The system keeps time in UTC
- A system's time zone is set from */etc/localtime*
- A terminal's time zone set from */etc/ttydefs*
- A shell's time zone is set using *TZ*
  - For the current time anywhere in the world
    TZ=`tzselect` date
- The current timezone file source code is found at *ftp://elsie.nci.nih.gov/pub/tzcode2004e.tar.gz* and is compiled using *zic* into */usr/share/zoneinfo*
- Time zones in Australia are under the power of the Commonwealth, but since no power has been exercised state legislation sets time zones
- SA has CST/CST, S=standard, S=summer, duh!

→NTP security

→Access control

- The `restrict` keywork is misleading, as it can unrestrict access too
- `restrict default kod`
  - Send “kiss of death” response to unknown servers
  - Alternative is `restrict default ignore` but misconfigured clients will accelerate requests assuming packet loss
- `restrict 127.0.0.1 nopeer`
  `restrict ::1 nopeer`
  - Allow everything from localhost

→Access control

- `restrict 1.2.3.4 nomodify notrap noquery server 1.2.3.4`
  - Typical client setup, don't let the NTP server access my client NTP process
- `restrict 10.0.0.0 mask 255.0.0.0`
  - Typical server setup, let clients request time but no more
- `restrict 5.6.7.8 nomodify notrap noquery peer 5.6.7.8`
  - Typical peer setup, such as when peering the four master clocks in the network

→Kerberos replay attack

- If we can alter time on the KDC we can replay a Kerberos ticket grant
  - Since this is a single sign on technology we then have access to everything
- Kerberos technology is everywhere
  - Kerberos :-), Windows Domain, LDAP
- The lesson
  - use only authenticated time to KDC-like servers
  - need a trusted time source
    - a feed from some public NTP stratum 1 server is horrible: untrusted source, untrusted path
    - A GPS-derived time is attractive

→Kerberos replay attack

- BTW, some Kerberos implementations don't need an alter KDC time to allow a replay attack, as they don't list IP addresses in tickets
  - –Windows 2000, you can blow away the entire domain!

→NTP across firewall

- NTP is UDP port 123
- Block it on the firewall, both ways
- Provide a trusted internal NTP source
- List it in DHCP responses for casual users
  - */etc/dhcpd.conf* says

```
# Router is NTP server for subnets
options ntp-servers 200.201.202.254;
# Adelaide is UTC+9:30
options time-offset 34200;
```

- Give DNS alias for formal users
  - *ntp.greenhill.office.example.com.au*
  - Aliases are a good idea for all services

→Symmetric keys

- `controlkey`
  - –*ntpq* access
- `requestkey`
  - –*ntpdc* access
- `trustedkey`
  - –Client and peer access
  - –Multiple keys are fine
- Key identifier and key must match
- Generate keys with `ntp-keygen -M`
- Ensure *ntpd* not started with `-A` option

aarnet

Australia's Academic
and Research Network

www.aarnet.edu.au

Glen Turner
glen.turner@aarnet.edu.au
http://www.aarnet.edu.au/~gdt/presentations/2000-10-19-linuxsa-ntp/