The School for Sysadmins Who Can't Timesync Good and Wanna Learn to Do Other Stuff Good Too

Paul Gear

Linux.conf.au 2017

Hobart

Thanks

GitHub

CANONICAL

The Plan

- caveats & justification for this talk
- overview of Linux timekeeping, NTP concepts & algorithms
- installation, configuration, & troubleshooting
- review monitoring tools
- introduce the NTP pool, comments on scale
- common myths & misconceptions, best practices for avoiding them
- ~~my experiences in building a sub-US$100 stratum 1 server~~

Caveats

- “I'm no expert, I just try my best not to be a total screw-up.”
  – Sarah White, NTP Pool mailing list
- I've only used the NTP reference implementation
  - … and only on Linux
  - … and only with one reference clock driver
  - … and only with a limited number of options
- Assumed environment: cloud, enterprise, SMB
- Basic-to-intermediate Linux knowledge assumed

Why care about time synchronisation?

- Running distributed systems
  - e.g. Ceph, Kerberos, Mongodb
- Log matching
- Learning & tinkering
- Nerd factor

What is NTP?

- Standardised protocol for time synchronisation, currently up to version 4, defined in RFC5905
- Arguably “the longest running, continuously operating, ubiquitously available protocol in the Internet”
  – David L. Mills, NTP.org
- Simplified version: SNTP
- Various implementations; reference implementation is from the Network Time Foundation

What's the issue with NTP?

- Not widely understood
- Behind-the scenes, unglamorous
- Old protocol, chequered security history
- Daunting documentation
- Misinformation, superstition, cargo-culting

Linux timekeeping

Linux timekeeping concepts

- Unix time
  - the number of seconds since the epoch: 1970-01-01 00:00 UTC
- UTC-only
  - time zones are a user space problem
- local clock
  - kernel maintains Unix time using regular timer interrupts
- real time clock (RTC/CMOS)
  - keeps time while system is off or suspended

Linux timekeeping concepts

- step – set the time
  - immediate change to the new time
  - local clock jumps
- slew – gradually adjust the time
  - time is sped up or slowed down, slightly changing the length of each second, to eventually reach the desired change in time
  - local clock remains relatively steady

NTP concepts

NTP concepts – assumptions

- One true time: UTC
- Nobody really has the one true time
- Bad time servers may be present due to inattention or malicious intent
- Network utilisation and topology change constantly

NTP concepts – time sources & strata

- Ultimate source: the oscillation rate of Caesium atoms
- Stratum 0: external sources, e.g. atomic, GPS, radio clocks
- Stratum 1: gets time from stratum 0 clocks
- Stratum 2+: gets time from stratum (n – 1) servers
- Strata are administrative boundaries analogous to IP subnets or Ethernet VLANs

Demo:
Installation & configuration

NTP concepts – terms

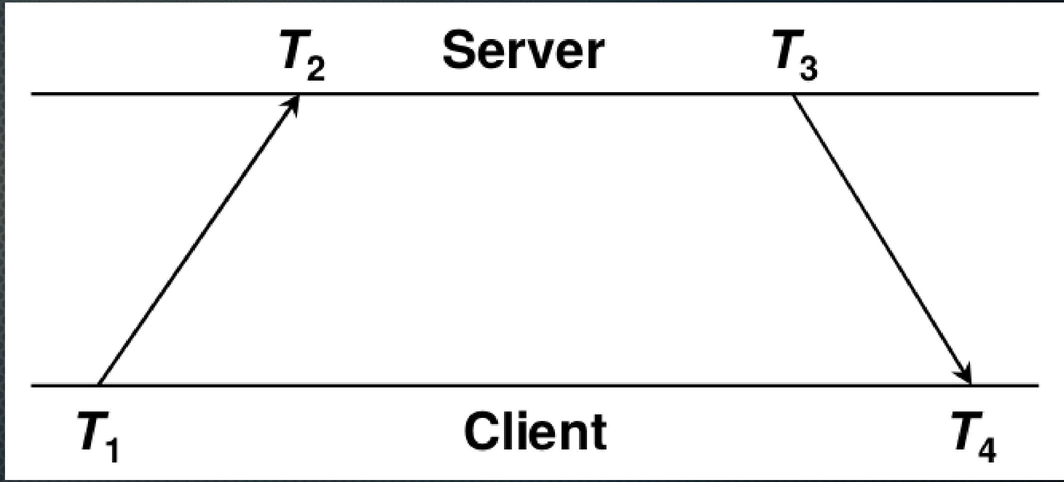
- offset – the difference between the local clock and a remote clock, after network delay is taken into account
- delay – round trip time on the network, not including the remote end's processing time
- frequency – error rate of the local clock; sometimes called drift
- poll – one round-trip check of a peer's clock

NTP concepts – polling

- Uses: UDP port 123
- Modes: broadcast, multicast, unicast
  - Unicast types: server, peer, pool
- Interval limits: 2^3 – 2^{17} seconds (in powers of 2)
  - Usual range: 2^6 – 2^{10} seconds (~1 – 17 minutes)

NTP concepts – polling

- 2 packets (client request, server reply) and 4 timers:
  - t1: origin time stamp
  - t2: receive time stamp
  - t3: reply time stamp
  - t4: destination time stamp
- Relative to client: t1 & t4
- Relative to server: t2 & t3
- This process is repeated for every poll of every time source



NTP concepts – algorithms

- filter – each source is polled independently and the samples from it are checked for correctness and filtered for anomalies
- selection – preferred sources are selected using the intersection algorithm
- clustering – the best of the surviving sources are determined via statistical analysis
- combining – the results from clustering are used to determine the correction to make to the local clock

Demo: Troubleshooting

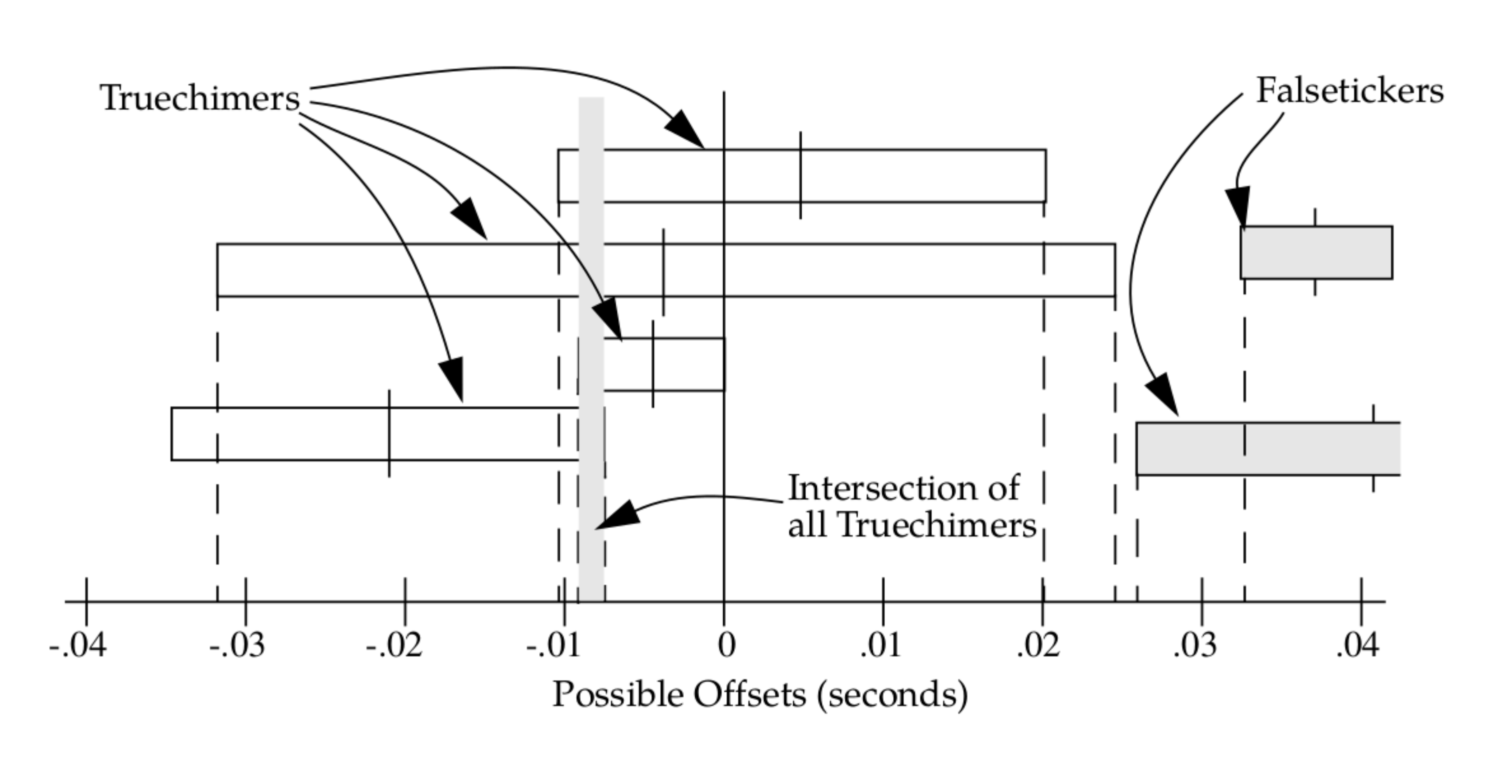
NTP algorithms: intersection

NTP algorithms – intersection

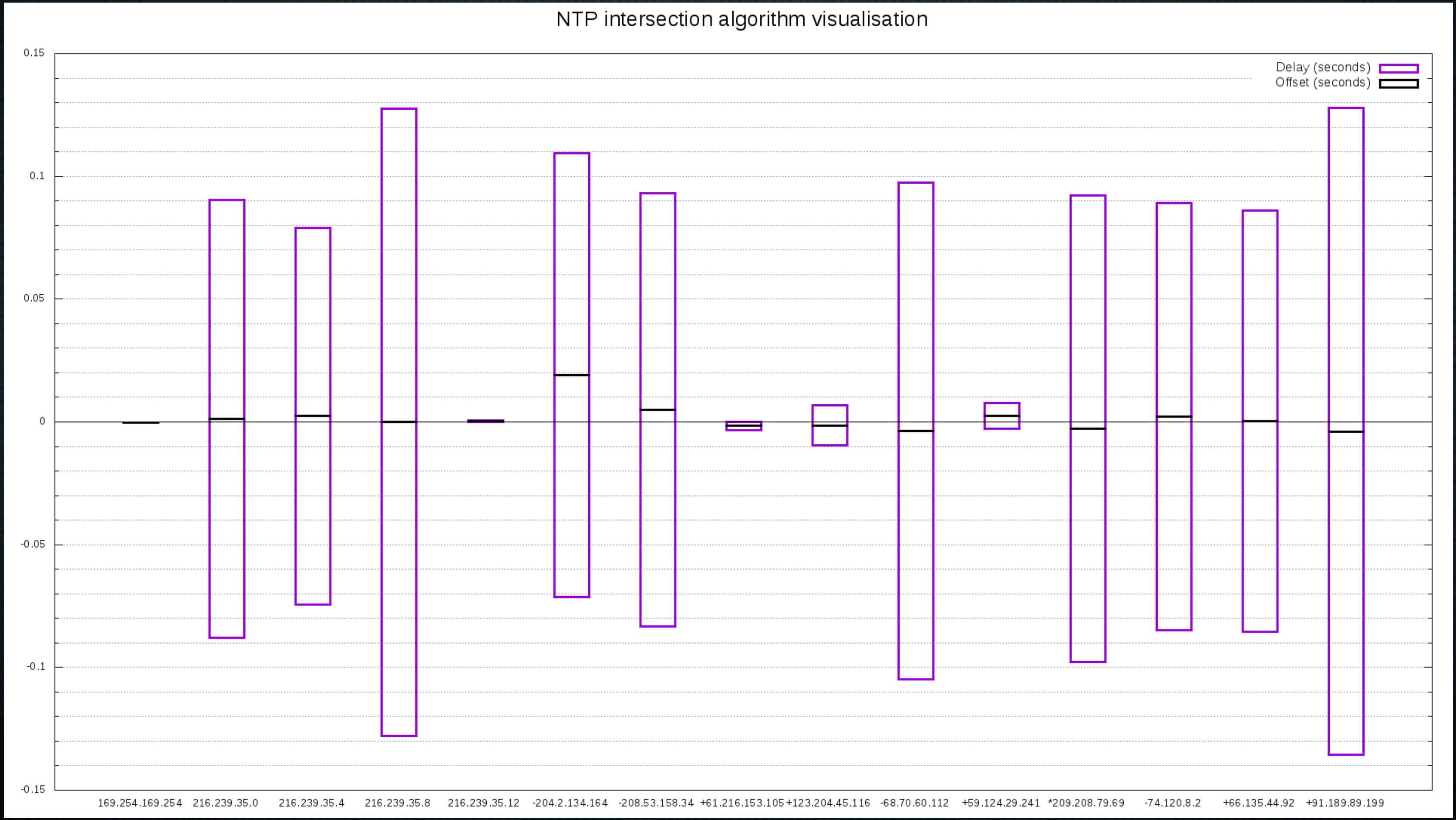
- Algorithm with the most significance for configuration and troubleshooting
- Goal: find the *largest possible agreement* about the true time
- How? Find the interval which includes the *highest* low point and the *lowest* high point of the *majority* of peers

NTP algorithms – intersection

highest low point and lowest high point of the majority of peers



NTP algorithms – intersection



Monitoring

Monitoring – general

- Set it up before you need it
  - enable statistics
- Make sure you're using **`pool`** rather than **`server`** to use NTP's self-healing process
- Decide in advance what to alert on
- Be careful what you wish for – NTP is a alerting canary for:
  - CPU/BIOS/firmware bugs
  - connection tracking limits
  - misconfigured DNS resolvers
  - bad hypervisor clock drivers
  - saturated uplinks

Monitoring – alerting

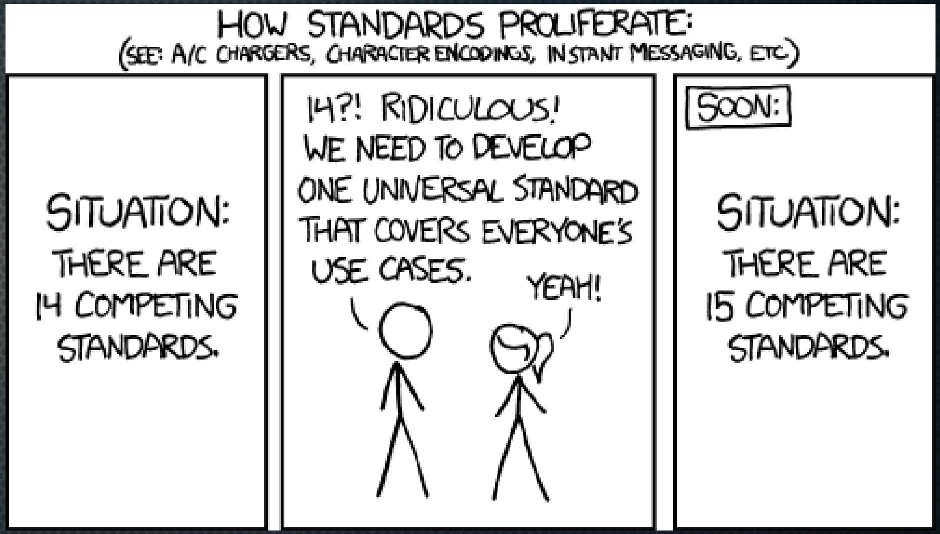
- Nagios – default plugins
  - check_ntp_time – checks remote host rather than local host – huh?
  - check_ntp_peer – not comprehensive, allows large offsets
- Nagios – 3rd-party plugins
  - check_ntpd – best of the bunch; use it if you like Perl

Monitoring – telemetry

- collectd – NTP plugin
  - some system & peer metrics
- prometheus node exporter – NTP collector
  - minimal statistics, well worth not graphing
- telegraf – ntpq input plugin
  - comprehensive peer metrics
  - immature code; show-stopper bugs

Monitoring

- alerting: check_ntpmon
- telemetry: ntpmon
- currently supports collectd & Nagios performance metrics; prometheus/telegraf soon
- actionable alerts & summary metrics about the local ntpd



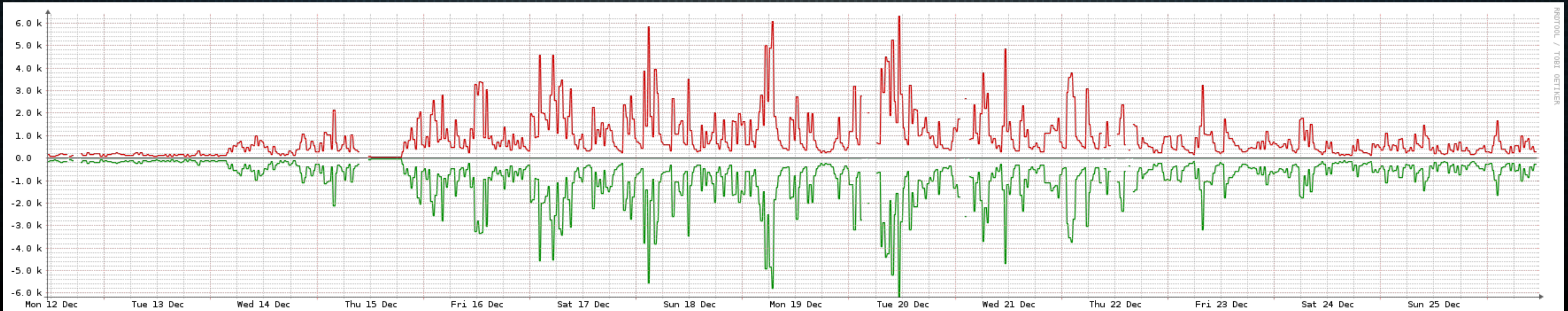
NTP pool

NTP pool

- Worldwide virtual cluster of NTP servers run by volunteers
- Approximately 2,600 IPv4 and 1,000 IPv6 servers in the pool as at 2017-01-01 – more needed!
- Default NTP service for many Linux distros & appliances
- Vendors using pool: please read guidelines

NTP pool – scale

- Ordinarily, low bandwidth, memory, CPU
  - watch conntrack tables
- On 2016-12-13, Snapchat released a new version of their iOS client. It included a timing library which queried between 35 and 60 NTP servers every time a user opened the app.
  - 40x unique IPs/hr, 2x/day; 7x peak packet count, 6x peak byte count



Myths, misconceptions, & best practices

Myths vs. Realities

- Local clock good enough
- Doesn't work in VMs
- Don't need NTP in VMs
- Don't need to be connected to the Internet
- You should have only one authoritative source
- Doesn't work behind ADSL

- Disable local clock
- Fine on modern kernels
- Separate kernel, separate clock
- Need connection to multiple stratum 1 servers
- NTP works best with multiple sources: 4-10 preferred
- Can achieve < 5 ms offset

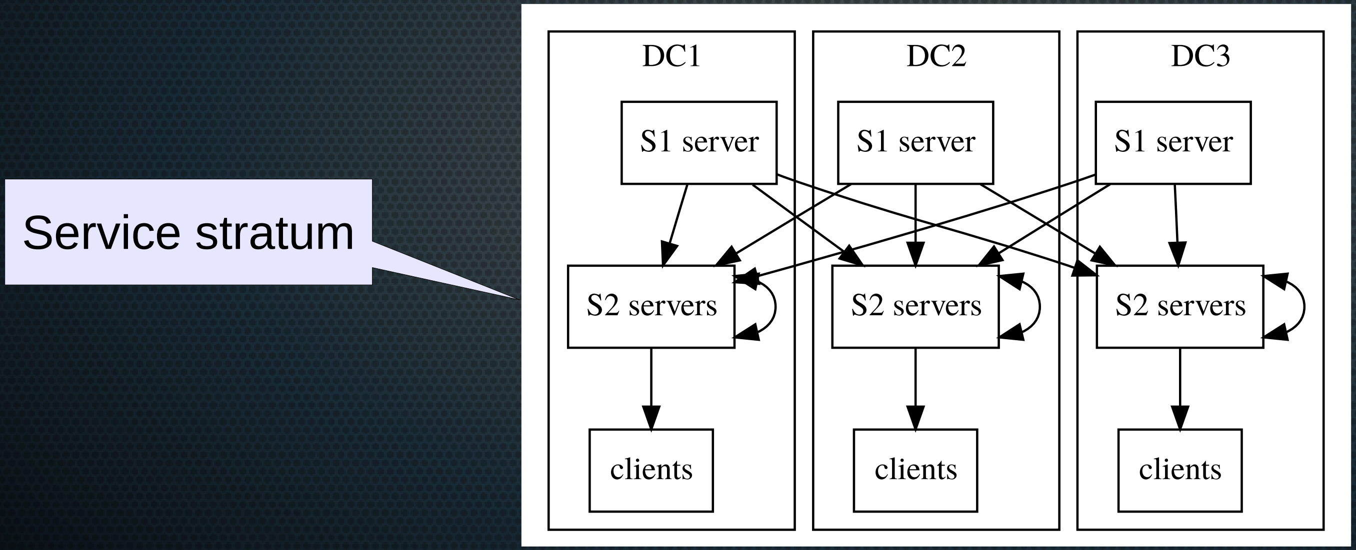
Preferred configurations

Preferred configuration – cloud

- standalone instance: default configuration
- environment with interrelated services: designated NTP servers
- don't run containers on a host you don't control

Preferred configuration – data centre/corporate

- Large data centres with thousands/millions of bare metal hosts and/or VMs should have a separate service stratum



- Large, distributed organisations: use distributed service stratum

Preferred configuration – small office

- Use pool, tinker with low cost GPS-based stratum 1 sources
- Dedicated servers or full-featured routers can be service stratum between clients and NTP pool

Takeaways

- Timesync is fun and not too hard
- Learn the basics of NTP & read the docs to avoid the myths
- Base your decisions on data
- Start with a good design and your solution should scale as large as you need without much effort

Thanks for listening!

- The blog series on which this talk is based can be found at libertysys.com.au
- Any questions?

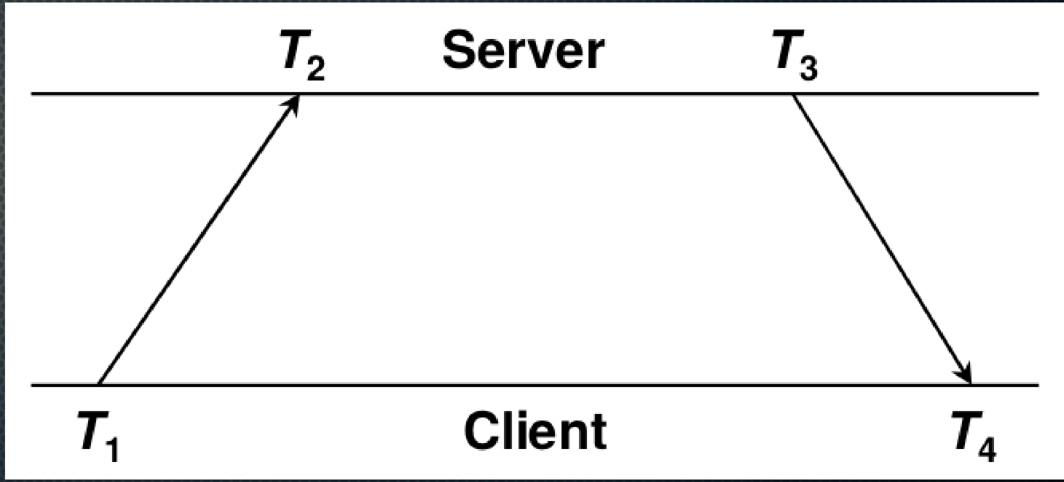
(Deleted scenes)

NTP concepts – polling

- The delay is the round trip time, minus the time taken to process the request on the server:

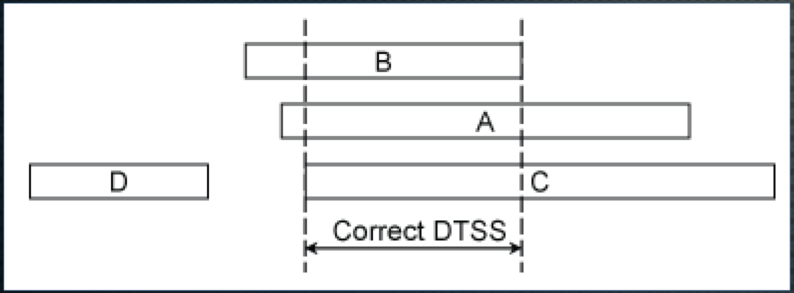
  (t4 – t1) – (t3 – t2)

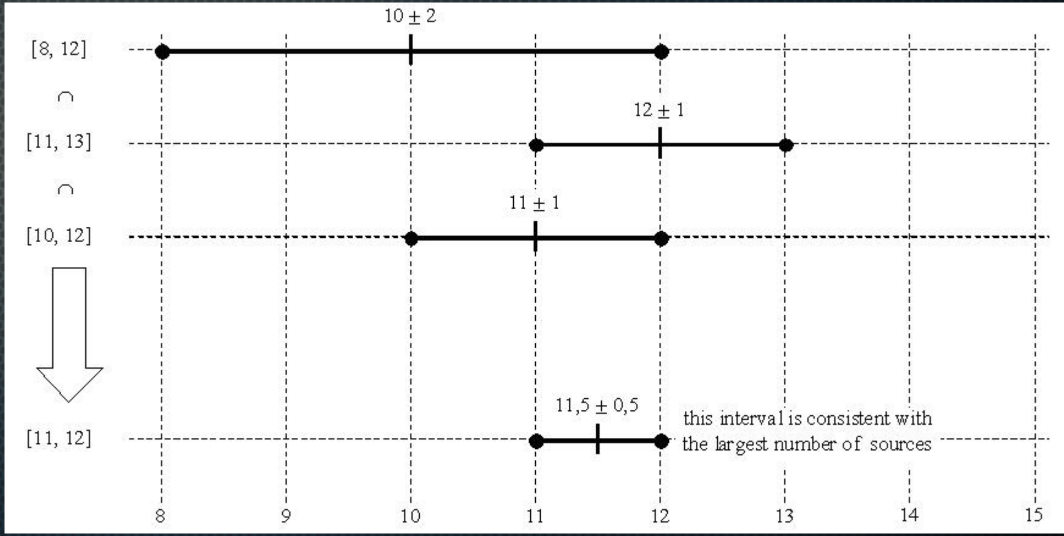
- The offset is the difference between the two clocks, with the travel time taken into account



NTP algorithms – intersection

highest low point and lowest high point of the majority of peers



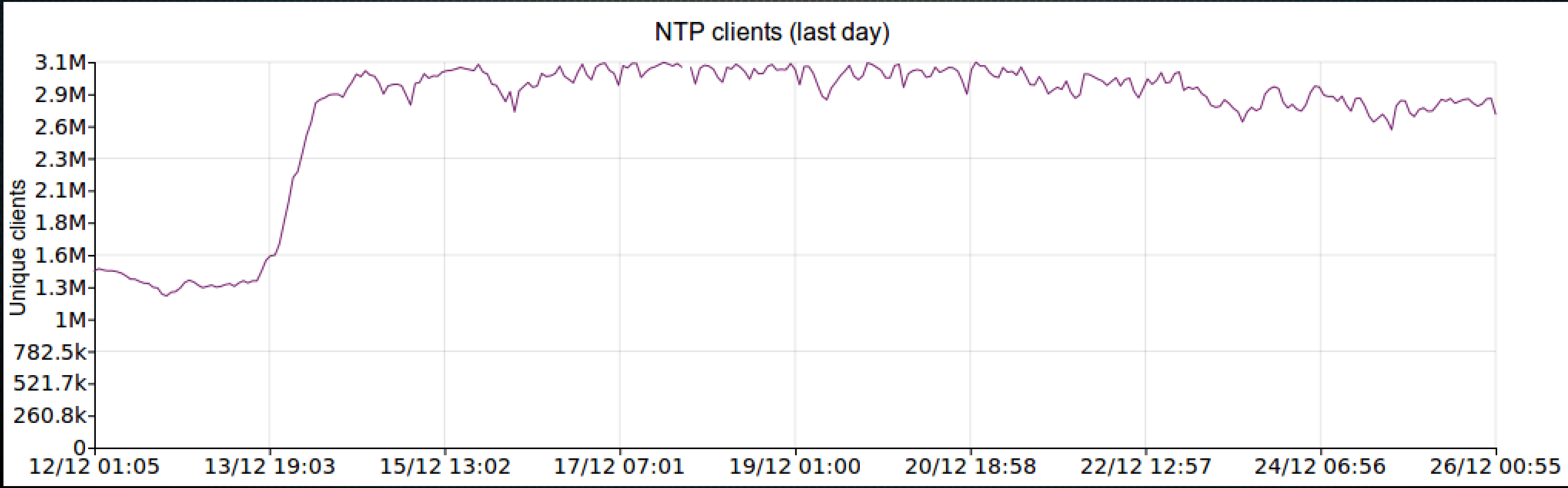


NTP pool

- Using the pool
  - you probably already are
  - use “pool” directive if available
- Participating in the pool
  - setup & monitoring
  - communication & longevity
  - manage load with bandwidth setting

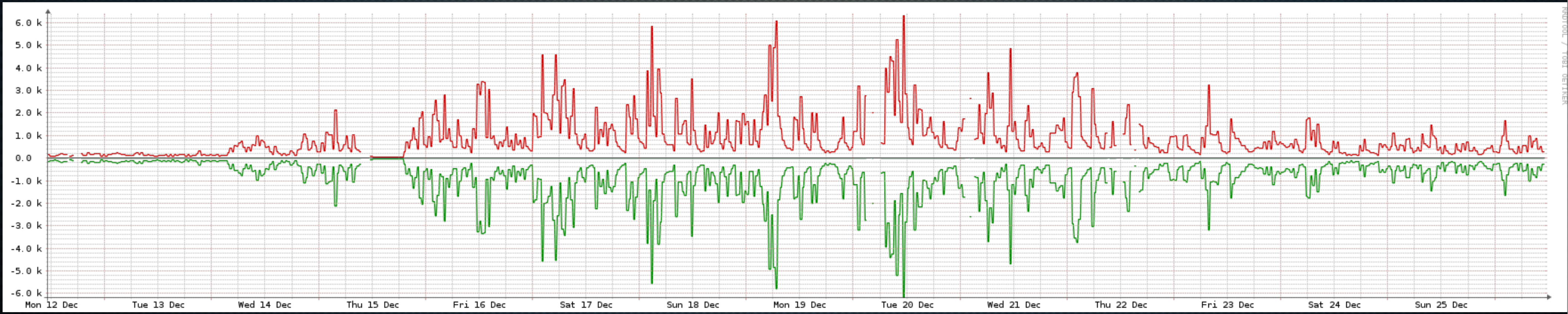
NTP pool – scale

- The Great Snapchat NTP Surge of 2016
  - unique clients per day doubled; 40x clients per hour
  - 7x peak packet count; 6x peak byte count



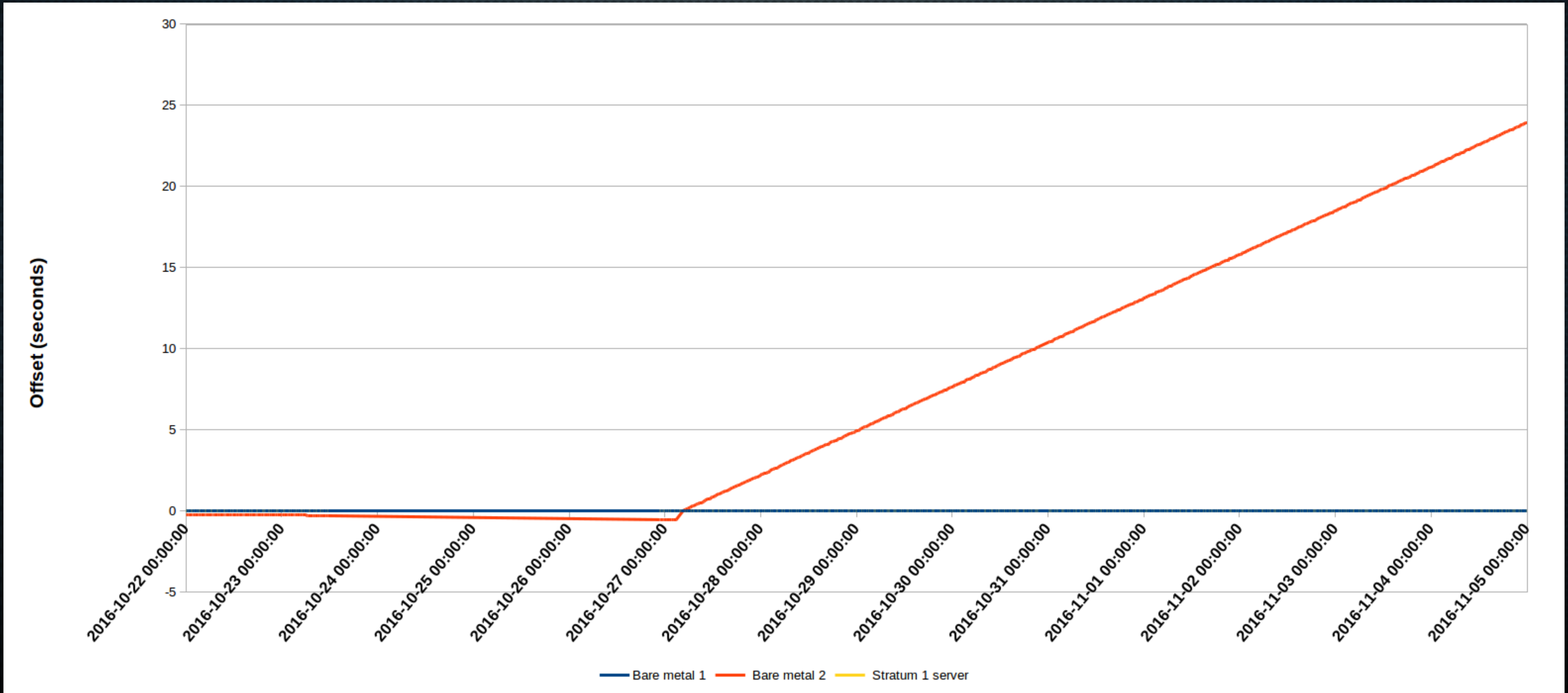
NTP pool – scale

- The Great Snapchat NTP Surge of 2016
  - Huge surge in NTP requests, but server not over-taxed in terms of bandwidth, CPU, memory



Myth: the local clock is good enough

Reality: it's only good enough if you don't care about time sync



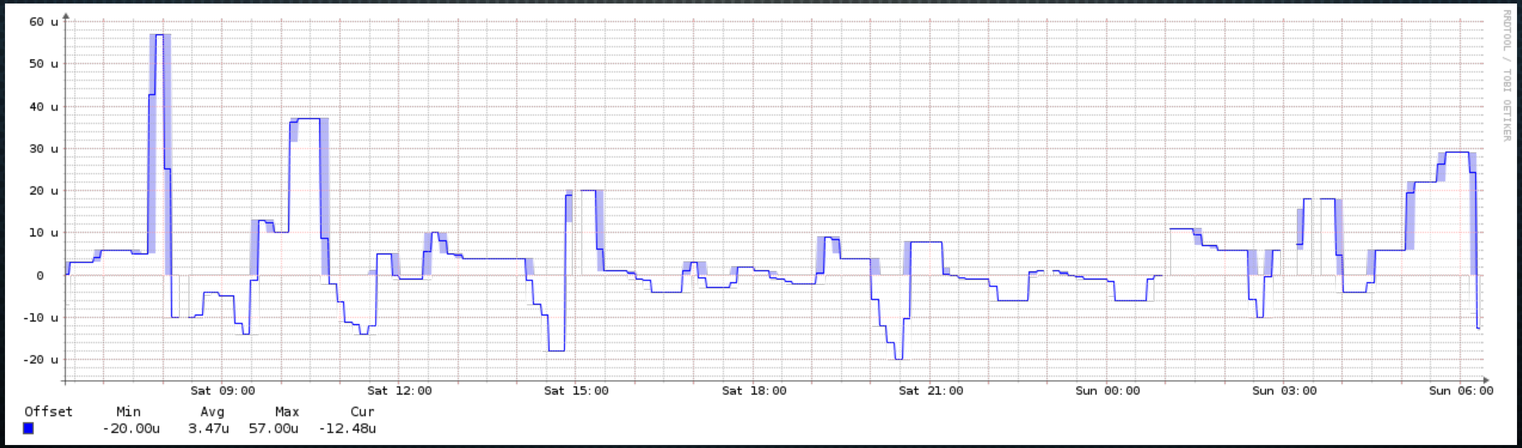
Myth: the local clock is good enough

- Reality: can vary by seconds every day
- Best practice: disable the local clock
  - this is the default on modern distributions
  - use orphan mode to handle temporary disconnections

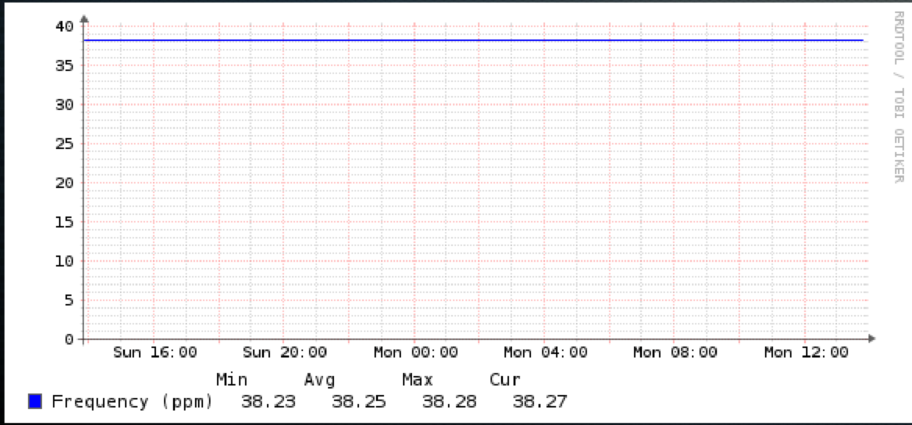
```
#server 127.127.1.0
#fudge 127.127.1.0 stratum 10

tos orphan 5
```

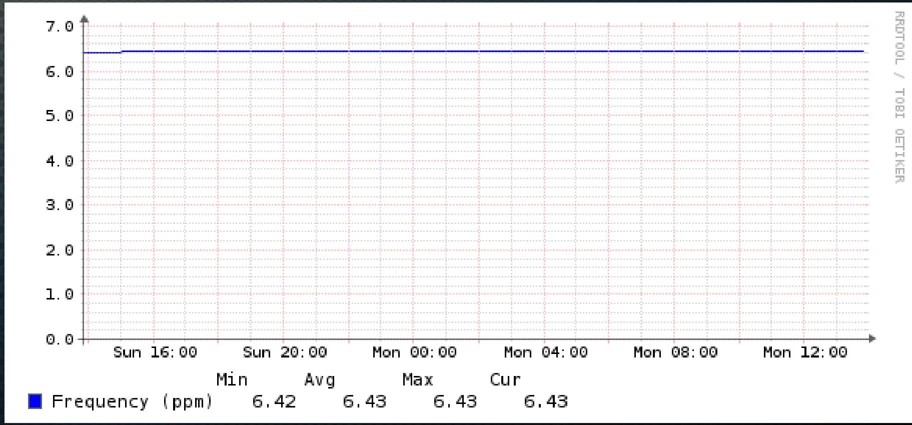
Myth: time sync in VMs doesn't work

- Reality: any recent Linux kernel should be able to maintain reasonable time sync in a VM; many pool servers are VMs
- Best practice: Host service stratum on bare metal if you have spare hardware & good deployment tools, but don't hesitate to use VMs where this makes sense

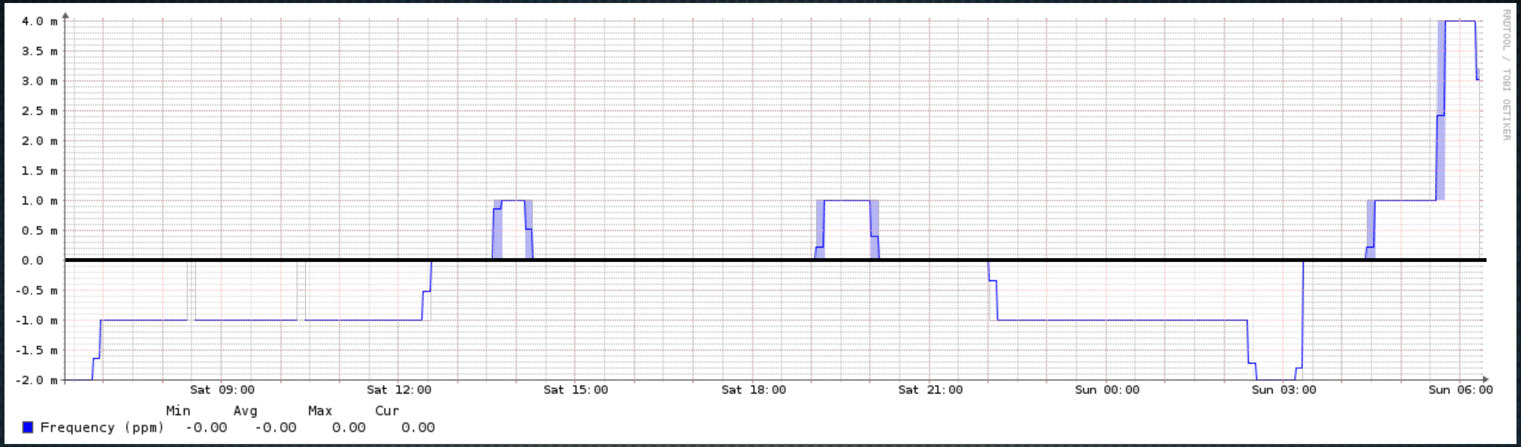


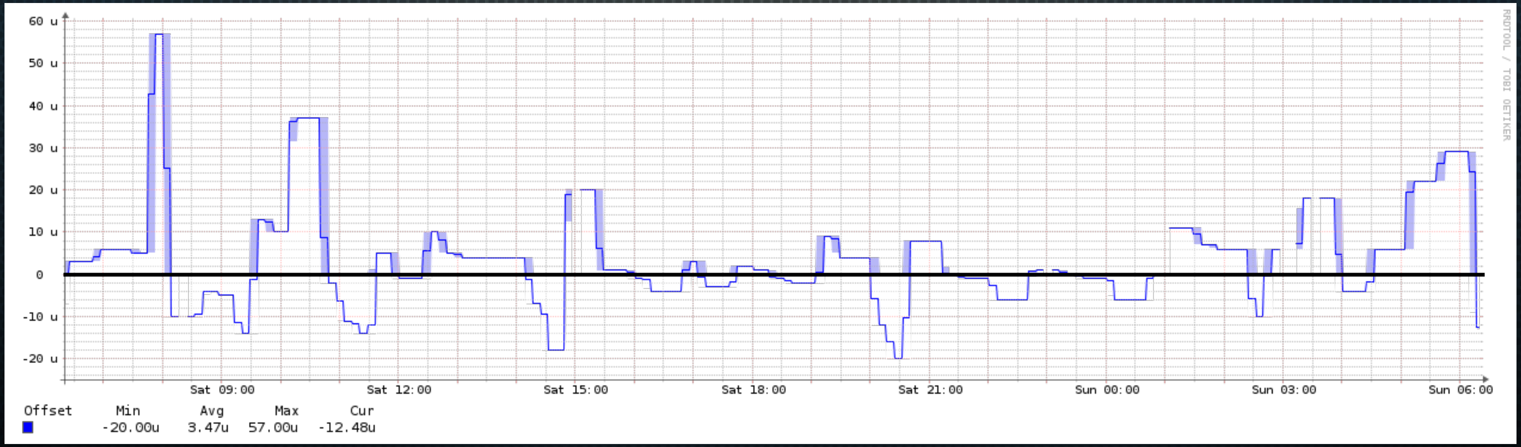
Myth: time sync in VMs doesn't work





Myth: time sync in VMs doesn’t work

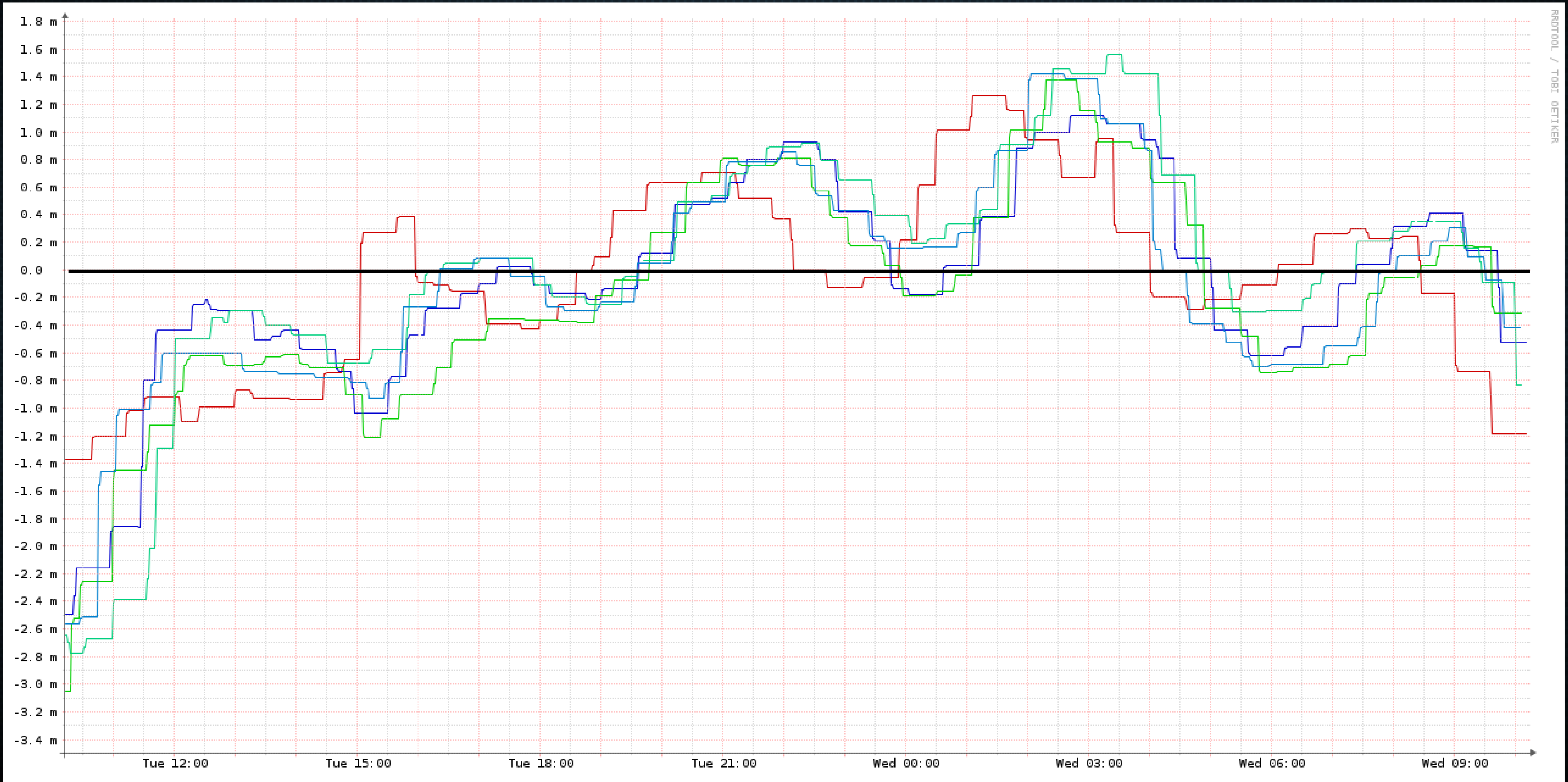




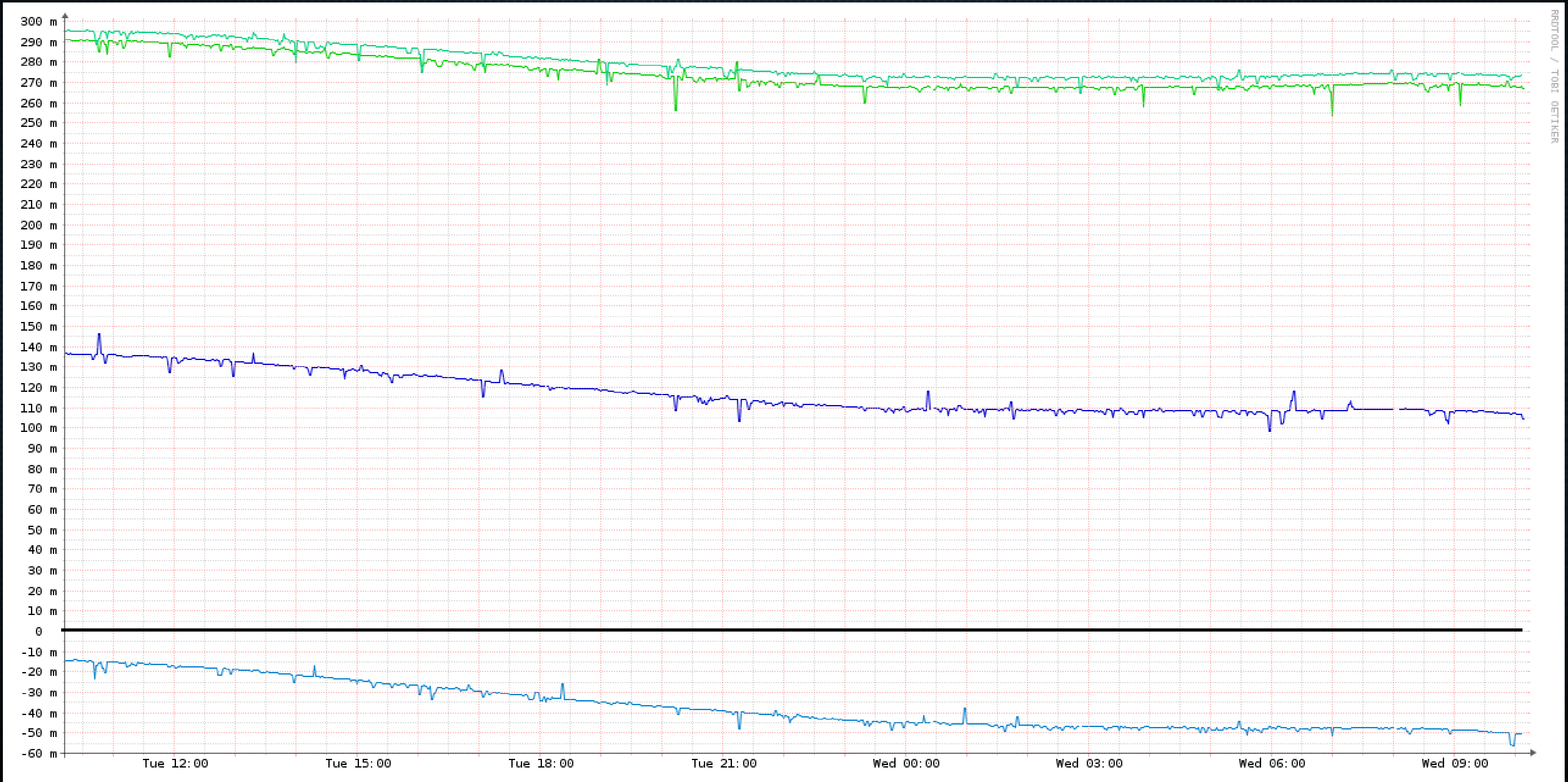
Myth: You don’t need NTP in VMs

- Reality: If you need time sync on bare metal, you need it in VMs
- Best practice: Deploy NTP configurations for VMs as you would for bare metal

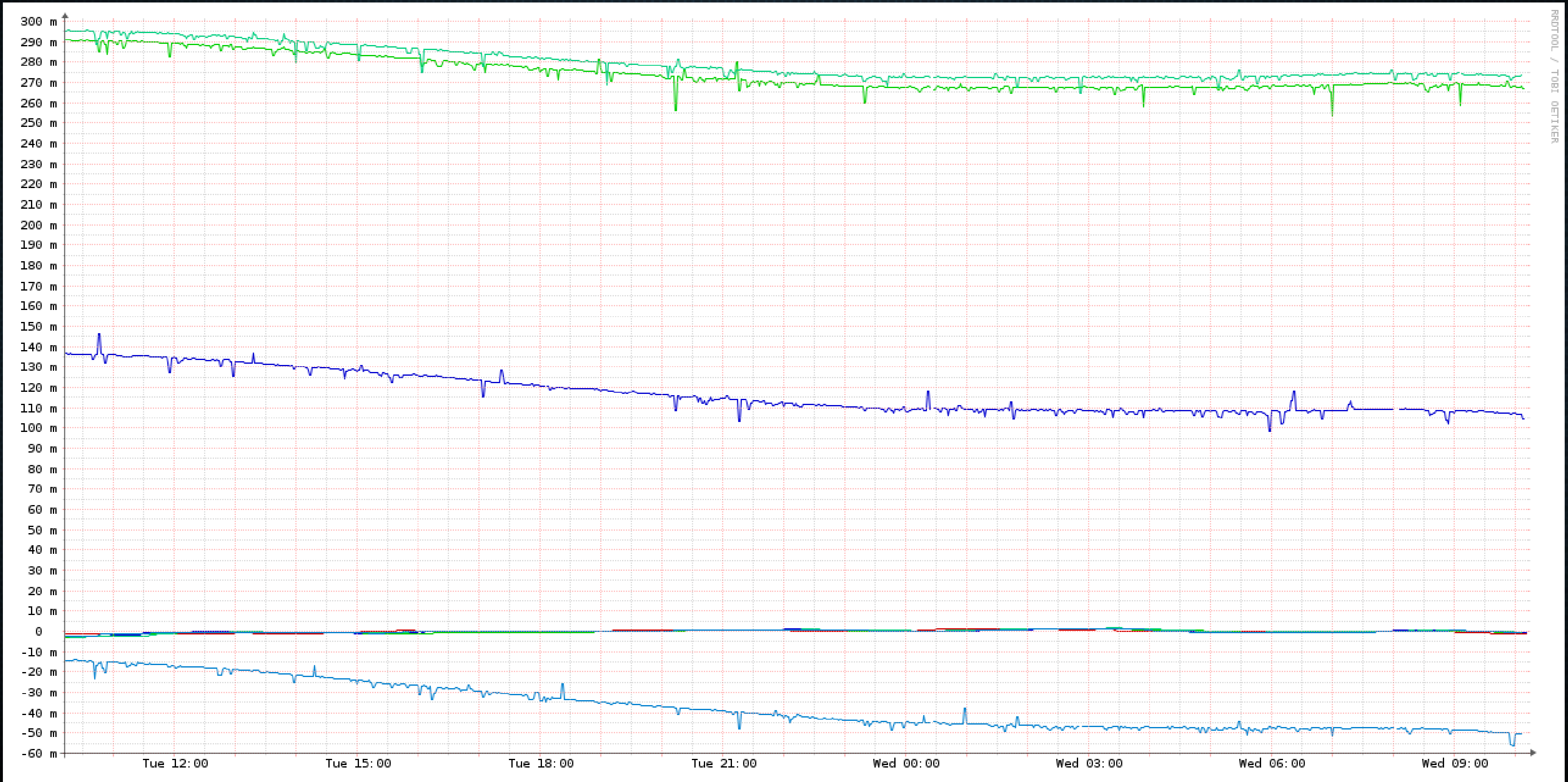
Myth: You don't need NTP in VMs



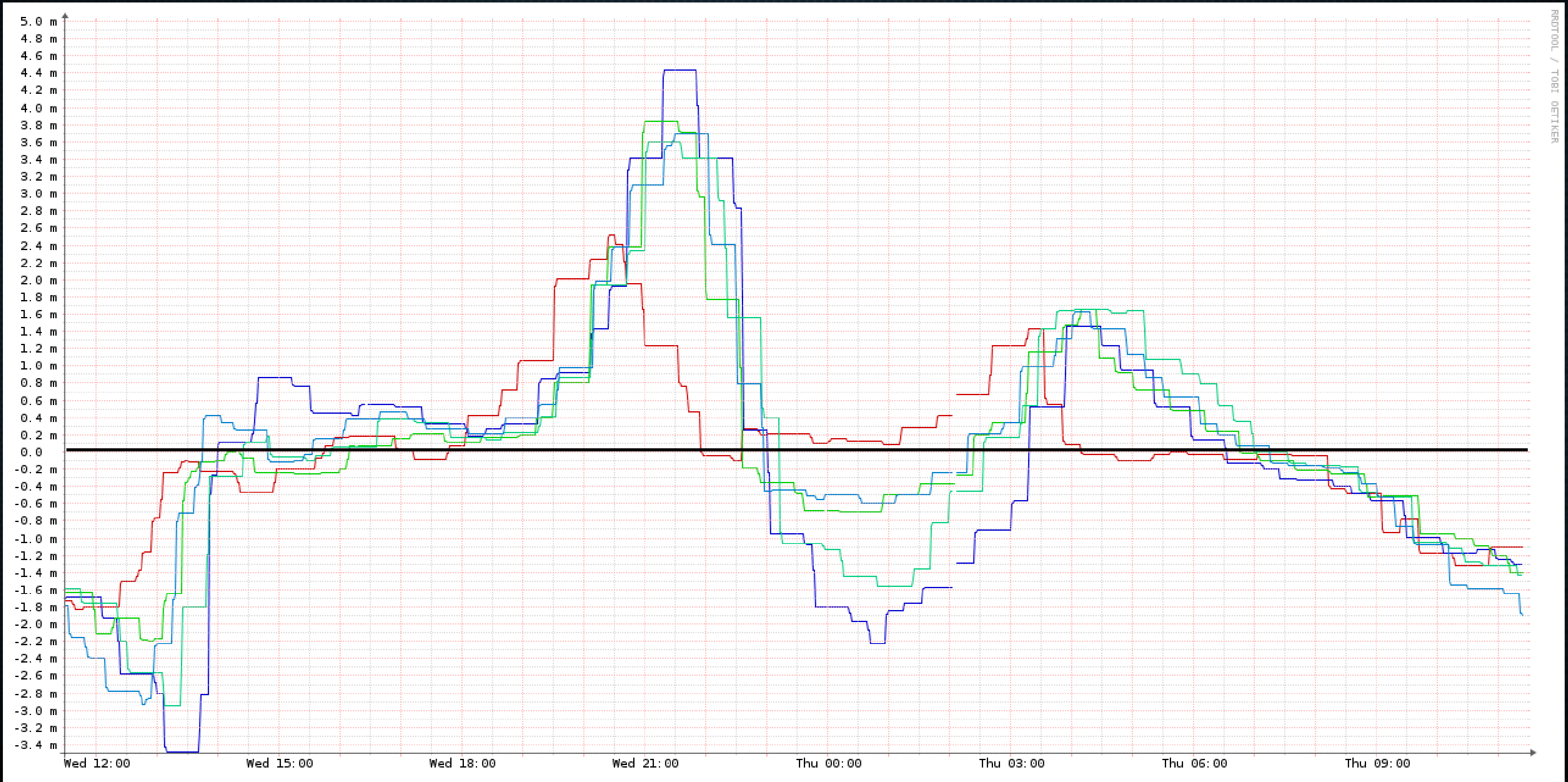
Myth: You don’t need NTP in VMs



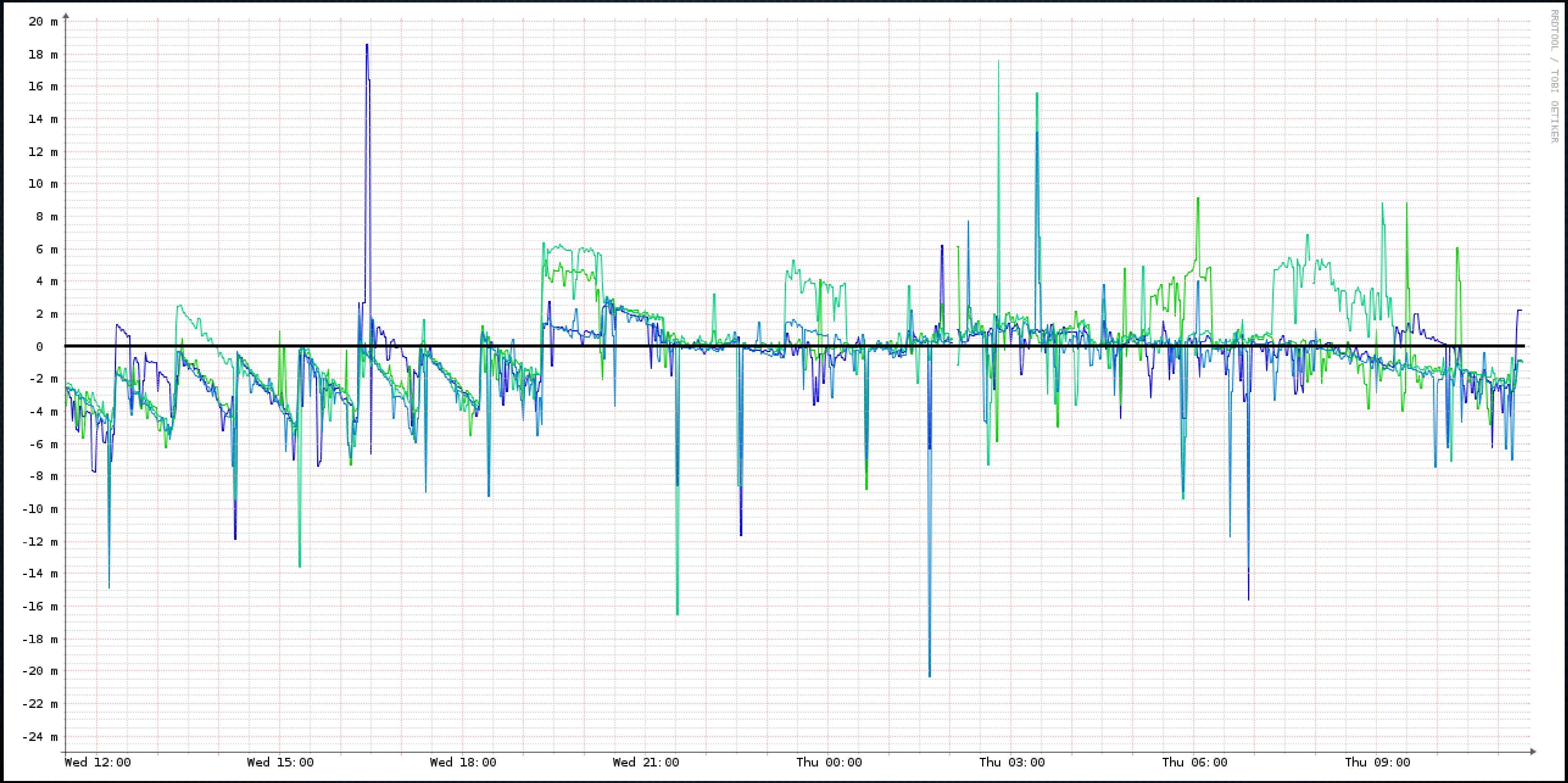
Myth: You don’t need NTP in VMs



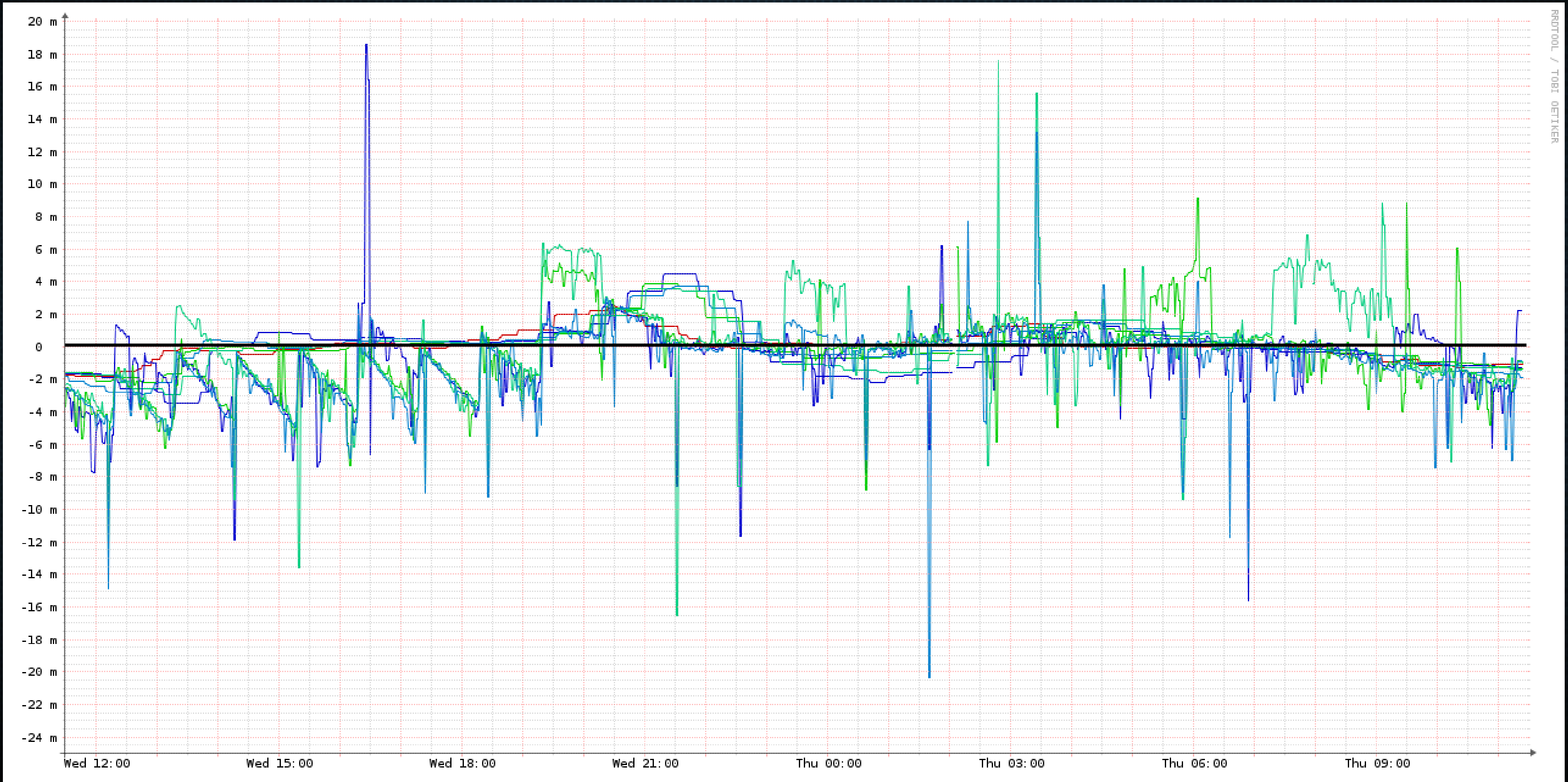
Myth: You don’t need NTP in VMs



Myth: You don't need NTP in VMs



Myth: You don't need NTP in VMs



Myth: You don’t need to be connected to the Internet to get accurate time

- Are you getting time from a local stratum 1 source?

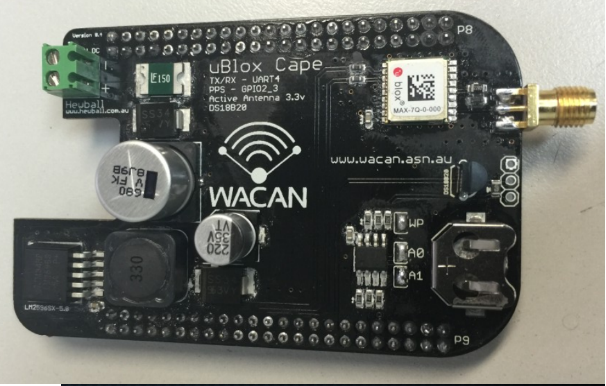
  “Time synchronisation is a critical service, and we can’t trust random servers on the Internet to provide it.”

- Or is someone just repeating the local clock myth?

  “It’s not important that clocks are correct, as long as they are consistent.”

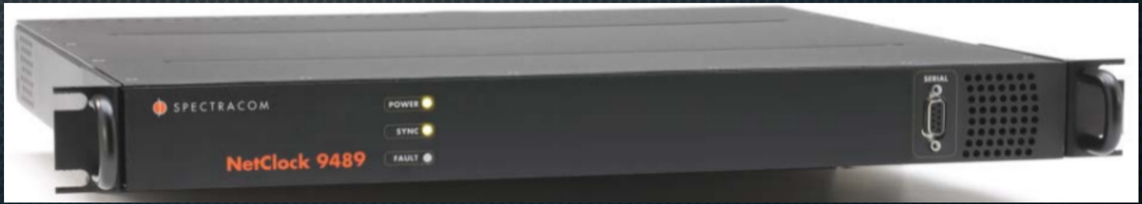
Myth: You don’t need to be connected to the Internet to get accurate time

- Reality: You need connections to multiple reliable stratum 1s
- Best practice: Run your own stratum 1 servers
  - There are suitable options for basically every budget
  - Alternatively, use local stratum 2 and public stratum 1



Myth: You don’t need to be connected to the Internet to get accurate time

Reality: You need connections to multiple reliable stratum 1s

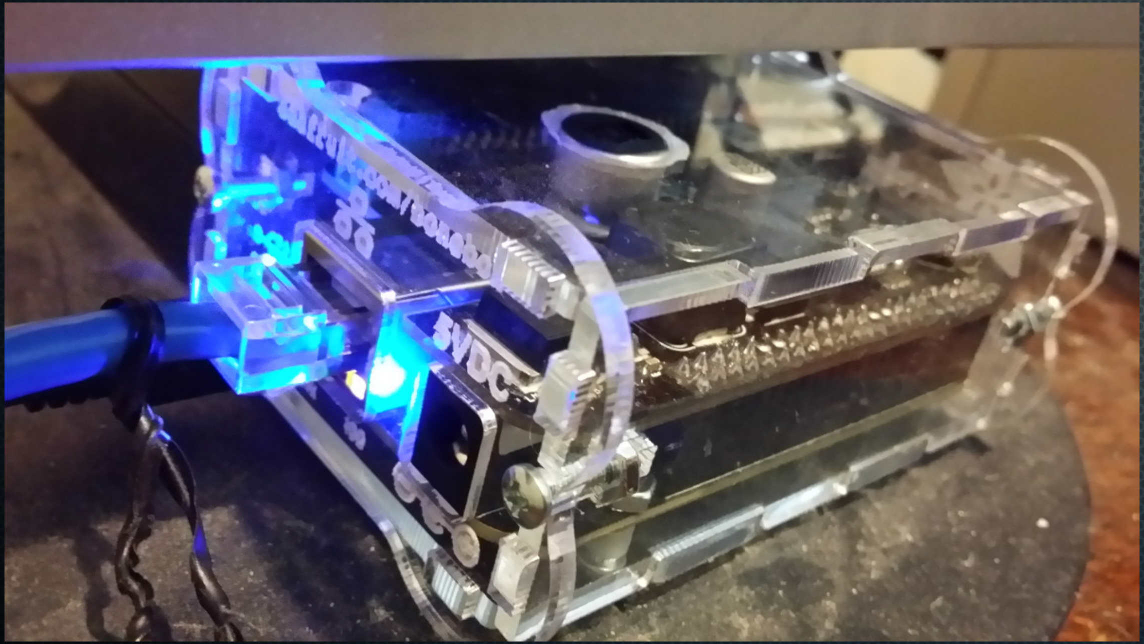


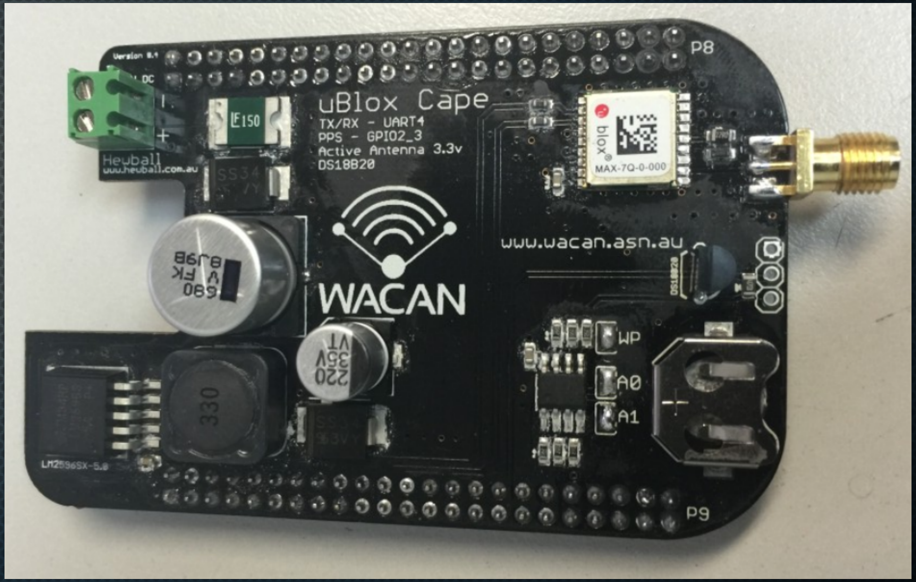
Myth: You don't need to be connected to the Internet to get accurate time

Reality: You need connections to multiple reliable stratum 1s

Myth: You don’t need to be connected to the Internet to get accurate time

Reality: You need connections to multiple reliable stratum 1s





Myth: You should only have one authoritative source of time

A person with a watch knows what time it is.
A person with two watches is never sure.
 – Segal's Law

Myth: You should only have one authoritative source of time

- Doesn't reflect how time measurement really works
- Doesn't fit how NTP's algorithms work
- Doesn't fit experimental data

Myth: You should only have one authoritative source of time

- Experiment:
  - 2 hosts, 8 VMs each; 4 VMs use their own host; 4 use local pool
- Results:
  - negligible difference in frequency; average offsets mixed
  - single-source hosts: 50-70% lower system offset
  - multi-source hosts: 77-79% lower root dispersion
  - With remote sources, multi-source hosts had minimum 9% lower frequency, 40% lower average offset, 60% lower root dispersion, 30% lower system offset

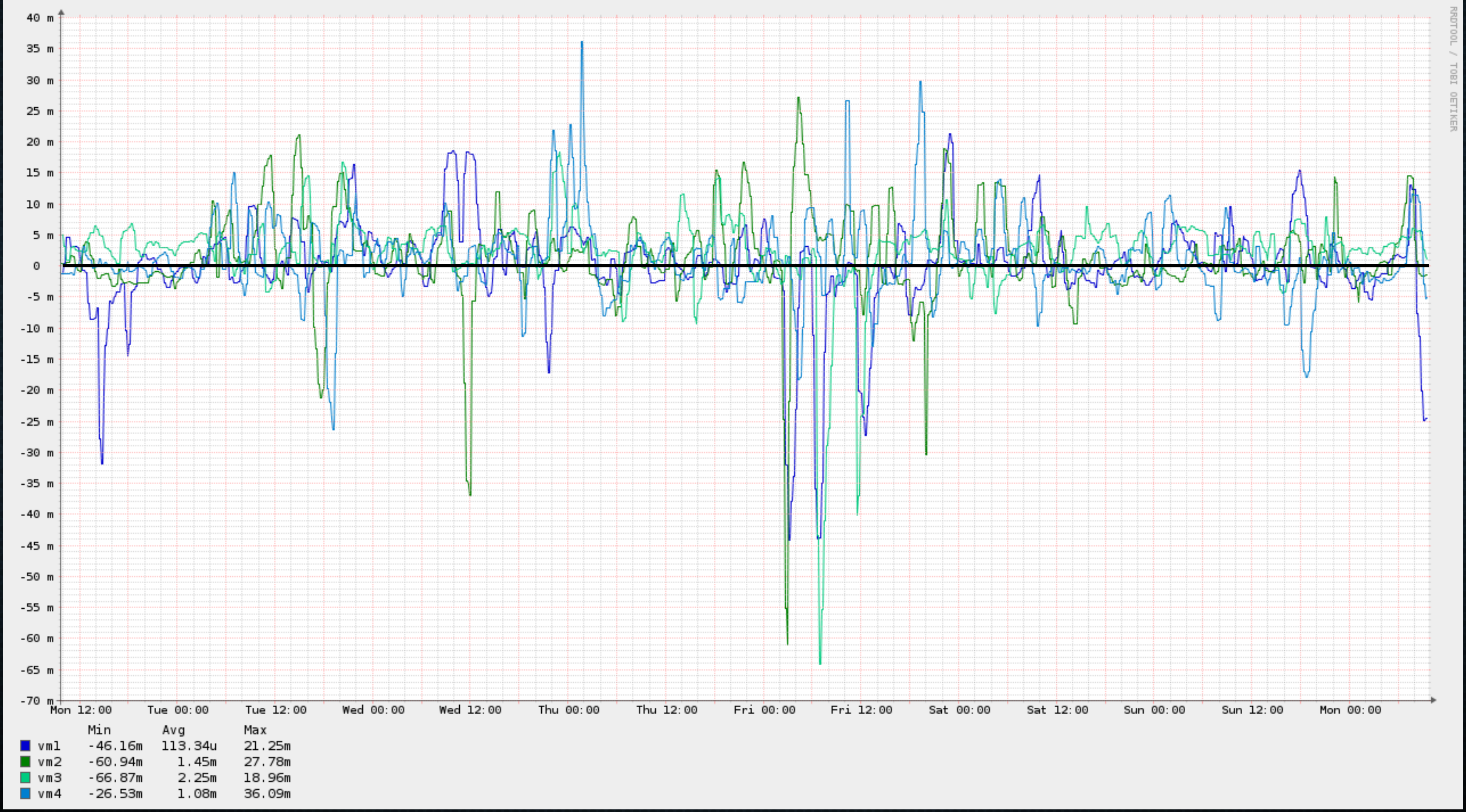
Myth: You should only have one authoritative source of time

- Reality: NTP is more accurate when it has multiple sources
- Best practice: 4-10 sources representing diverse stratum 1s
  - The default NTP configuration tries to do this for you

Myth: You can’t get accurate time behind asymmetric links like ADSL

- Reality: average < 5 ms offset
- Best practice:
  - minimising latency is preferred, but not essential
  - try NTP's huff-n-puff filter if it's a problem for you

Myth: You can't get accurate time behind asymmetric links like ADSL

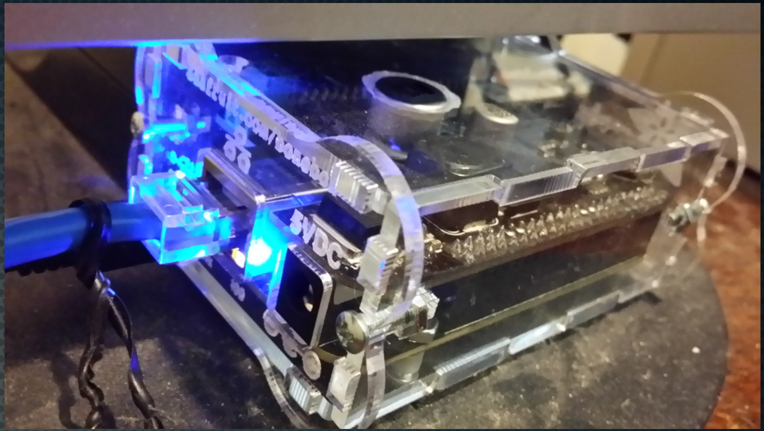


More myths

- Run ntpdate before ntpd – no need; ntpd will step on startup
- Run NTP in containers – no need; shared kernel = shared time

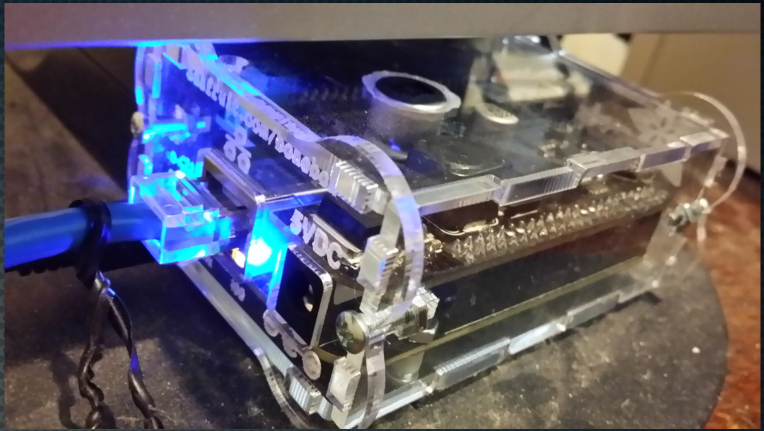
Building a stratum 1 server

- Budget: ~~AU$100~~ US$100
- Hardware:
  - BeagleBone Black (AU$70)
  - Snazzy case (AU$35)
  - WACAN GPS receiver (mates rates)
  - GPS antenna from eBay (AU$8)
  - USB & Ethernet cables (spares crate in back of cupboard)



Building a stratum 1 server

- Software:
  - Ubuntu 16.04 armhf (32-bit) from ports.ubuntu.com
  - Kernel with specific support for BeagleBone expansion cards
- Patience to learn about ARM, BeagleBone, GPS



Building a stratum 1 server

- Changes from standard config:

```
restrict 127.127.20.0
server  127.127.20.0 mode 65552 \
        minpoll 3 iburst prefer
  # mode 0x00010: 9600 baud
  # mode 0x10000: extended clockstats
fudge   127.127.20.0 flag1 1 \
        refid PPS
  # flag 1: enable PPS
  # refid: indicate that we're using \
  #         PPS signal from GPS
```