A Methodology for Clock Benchmarking

Julien Ridoux  j.ridoux@ee.unimelb.edu.au
Darryl Veitch  d.veitch@ee.unimelb.edu.au
Introduction

- **Higher demand on the network, better clocks**
  - Network applications are more and more distributed
  - Users/Providers need higher reactivity and more precision
  - Essential for testbeds and performance evaluations

- **Limitation: the quality of clock synchronisation**
  - Need for higher accuracy
  - Need for higher reliability

- **Difficult to benchmark timekeeping systems**
  - Against which reference?
  - How to access clock instantaneously?
Clock and Timestamping errors

Event occurring at true time

Clock Time

True Time

$t_k$

$t_k$

Timestamps

Perfect Clock

Drifting Clock
Clock and Timestamping errors

Event occurring at true time

Timestamps

Clock Time

$C(t'_k)$

$tk$

$tk'$

True Time

Perfect Clock

Drifting Clock

$t'_k$

$C(t'_k)$
Clock and Timestamping errors

- **Clock error or offset**
  \[ \theta(t_k) = C(t_k') - t_k \]

- **Timestamping error**
  \[ \xi(t_k) = t'_k - t_k \]

- **Total error:**
  \[ E(t'_k) = C(t'_k) - t_k = \theta(t'_k) + \xi(t_k) \]
In practice, no perfect clock for benchmarking

**Total relative error:**

\[ E_{C_1, C_2}(t_k) = C_1(t_k') - C_2(t_k'') \]

\[ = \theta C_1(t_k') - \theta C_2(t_k'') + \xi C_1(t_k) - \xi C_2(t_k) \]

- The clock and timestamping errors combine

**Without a perfect clock; benchmarking a challenge**

- Timestamping error: eliminate / estimate
- Clock error: relative / absolute

**Need a strong methodology**
3 Clocks under study (Linux & FreeBSD)

- **SW-GPS**: `ntpd + GPS sync.` Absolute Clock
- **SW-NTP**: `ntpd + Net. sync.` Absolute Clock
- **TSCclock**: Net. sync, Absolute & Difference Clock
CubinLab Testbed

- **Kernel timestamping of UDP packets**
  - Outgoing / Incoming directions
  - External: DAG Card
  - Internal: Multiple clocks simultaneously

![Diagram of experiment setup](image)
**Internal comparison**

- **Modified kernels: timestamps taken “back to back”**
  - Identical delay accessing the clocks $\rightarrow t'_k = t''_k$
  - Timestamping errors cancel $\rightarrow \xi C_1(t_k) = \xi C_2(t_k)$

- **Obtain a comparison of the two clocks offsets**
  \[
  E_{C_1,C_2}(t_k) \approx \theta C_1(t_k) - \theta C_2(t_k)
  \]
  - Free of timestamping error
  - No absolute performance with respect to true time
External comparison

- **Use of the DAG card**
  - Considered best absolute time reference available →
    \[
    E_{C,Dag}(t_k) = \theta_C(t_k) + \xi_C(t_k) - \xi_{Dag}(t_k)
    \]
  - Provides absolute reference
  - But suffers from timestamping error

- **Additional kernel modifications to reduce noise**
  - Standard location in kernel for all clocks
  - Improved locations for the TSCclock
    - As close as possible to the last bit transmitted/received
    - Interrupt bottom-half / driver implementation
Reducing kernel timestamping error

Unix PC

DAG

Host

\[ t_a \quad t_g \quad t_f \]

\[ d_{h\uparrow} \quad d_{h\downarrow} \]

\begin{align*}
\text{Outgoing} & : \text{med}= -268 \ iqr= 36.1 \ [\text{mus}] \\
\text{Incoming} & : \text{med}= 92.1 \ iqr= 8.01 \ [\text{mus}] \\
\text{Standard IN} & : \text{med}= 177 \ iqr= 10.2 \ [\text{mus}] \\
\text{Improved IN} & : \text{med}= -84.5 \ iqr= 14.9 \ [\text{mus}] \\
\text{Improved OUT} & : \text{med}= -84.5 \ iqr= 14.9 \ [\text{mus}] \\
\text{Standard OUT} & : \text{med}= 92.1 \ iqr= 8.01 \ [\text{mus}] \\
\end{align*}
Reducing kernel timestamping error

- **Standard timestamping location with the TSCclock**
  - Outgoing direction **noisier**: IQR 100 µs larger
  - **Asymmetry** of 100 µs between Outgoing / Incoming
  - Improved location much better

![Graphs showing data distribution for Standard IN (1), Improved IN (2), Improved OUT (3), and Standard OUT (4)]
Reducing kernel timestamping error

- **Standard timestamping location with the TSCclock**
  - Outgoing direction **noisier**: IQR 100 µs larger
  - **Asymmetry** of 100 µs between Outgoing / Incoming
  - Improved location much better

- **The same clock in both directions !!!**
  - Which direction to trust?

![Graph showing the comparison between standard and improved timestamps for incoming and outgoing directions.](image)
### Host RTT measurement

- **If we use both directions**
  - Minimum Host RTT: \( r^h = d^{h\uparrow} + d^{h\downarrow} \)
  - Available timestamps: \( R^h = r^h + \xi(t_f) - \xi(t_a) \)
  - Host RTT available since measured with the same clock
  - Minimum can be filtered; noise is the width of histogram of \( R^h \)
Recovering one-way measurements

- Ambiguity due to the asymmetry that we can’t evaluate

\[ \text{asym} = d_{h\downarrow} - d_{h\uparrow} \quad \text{asym} \in [-r^h, r^h] \]

- One way delays can’t be recovered individually
- Host RTT \( \rightarrow \) impact of noise on one-way measurement

  - Median is ambiguous but bounded by \( 2 \times r^h \)
  - Histograms are broadened because of \( \text{IQR}(R_h^h) \)
Beware of problematic drivers/NIC

- **Two hosts with same OS / hardware**
  - FreeBSD 6.1
  - Pentium-D architecture
  - But different NIC / Driver
    - Maxwell : Broadcom 5157 Gig-E (Brown)
    - Tastiger : 3Com 10/100 Mbps (Black)

- **Choose carefully!**
  - The quality of $R^h$ measurement drives the accuracy of the methodology!
Let’s get started

- **Now that we have**
  - an accurate testbed
  - internal / external timestamping and validation
  - improved kernel timestamping
  - removed problematic hardware

- **... we can start the detective work**
**SW-NTP vs. TSCclock**

- **Internal comparison** → large oscillations ± 1ms

- **External comparison** → SW-NTP responsible
  - Noise: IQR($R^h$) = 37µs, $r^h$ = 10µs (ambiguity = 20µs)
  - Accurate view for SW-NTP, difficult diagnosis for TSCclock
SW-GPS vs. TSCClock

- **Internal comparison → similar behavior (IQR = 14µs)**

- **External Comparison**
  - Noise: IQR($R^h$) = 23µs  
    - $r^h = 10µs$ (ambiguity = 20µs)
  - One clock may have (IQR = 0 + noise). But can’t be verified!
  - TSCclock slightly ahead but which clock is worse?
SW-GPS vs. TSCclock (Zoom)

- Observe SW-GPS and TSCclock more closely
  - Observe oscillations with a 20mn period

- Both clocks show oscillations
- Temperature effect (air-conditioning variations)
SW-GPS and (TSCclock) Difference Clock

- **Measure UDP packets inter-arrivals**
  - Compare SW-GPS IAT and TSCclock Difference Clock IAT

- **Final error in a ±1µs band**
  - Resolution of SW-GPS is 1µs (struct timeval)
  - Can’t interpret errors within this band

- **Spikes of up to 10µs magnitude**
  - By construction, can’t be due to the difference clock
  - Small time scale stability of the oscillator
  - Due to the SW-GPS clock!!
Conclusion

- **Clock benchmarking**
  - is a challenge
  - requires good quality hardware
  - a strong methodology
  - requires rigour and attention to details

- **Our methodology and testbed**
  - highlights the need for kernel modifications
  - presents Internal / External complementary comparisons
  - provides comparison down to the system clock resolution
  - allows to track causes of observed strange behaviors

j.ridoux@ee.unimelb.edu.au
http://www.cubinlab.ee.unimelb.edu.au