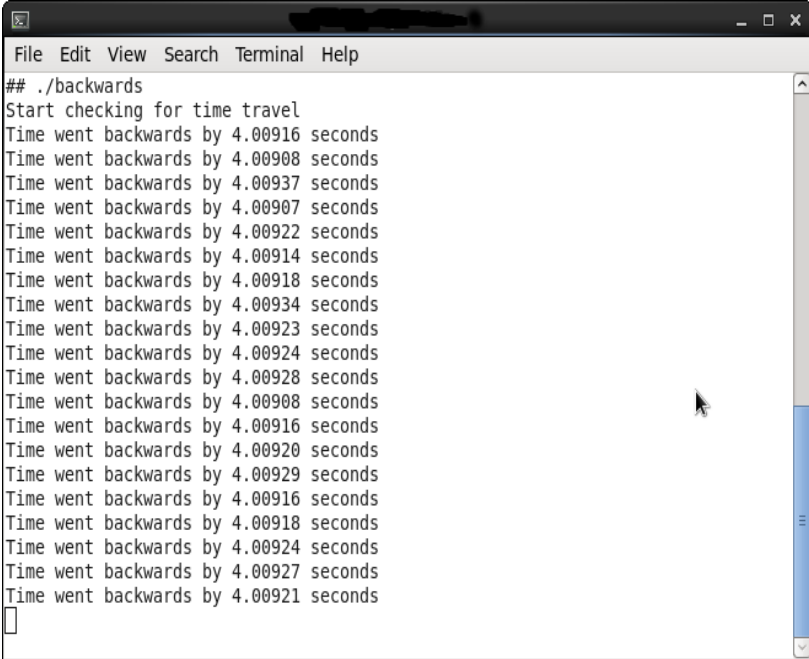


Time should not go backwards.

All data integrity is lost if the clock ever runs backwards. A bid sent by a trading program and confirmation received milliseconds later might be recorded as completing in reverse order. Analytical systems looking for trends would see time series that were wrong. The trading record would be useless in data analysis and dispute resolution. We tested PTPd2¹ software using a source that periodically jumps time backwards by four seconds and then corrects the time on the next update. This is a failure similar to ones seen in production systems, a transient error – fixed one second later. But the error is enough to make PTPd2 jump the clock backward so that applications see time go backward. No alarms or alerts are produced, and the “offset” log has no indication of error. A trading application experiencing this condition would quietly violate FINRA 7430 by multiple seconds. PTPd2 synchronization software is in use in many financial trading firms and organizations.

Worse errors are known to occur in production systems. For example, IMC engineers reported that their GPS clocks experienced multi-second “jumps” that caused non-TimeKeeper client software to jump the time backwards.

*“On several occasions, a [hardware] bug caused the (single) time source to send time without leap seconds information, for two (!) hours. [...] All clients, however, saw a **34 second offset** without any indication that this time might be invalid or suspicious in any way. Consequently, they either ‘corrected’ this situation by stepping (jumping) the clock backwards, or by slewing (slowing down) the clock at maximum speed to the ‘new’ value.*



```

File Edit View Search Terminal Help
## ./backwards
Start checking for time travel
Time went backwards by 4.00916 seconds
Time went backwards by 4.00908 seconds
Time went backwards by 4.00937 seconds
Time went backwards by 4.00907 seconds
Time went backwards by 4.00922 seconds
Time went backwards by 4.00914 seconds
Time went backwards by 4.00918 seconds
Time went backwards by 4.00934 seconds
Time went backwards by 4.00923 seconds
Time went backwards by 4.00924 seconds
Time went backwards by 4.00928 seconds
Time went backwards by 4.00908 seconds
Time went backwards by 4.00916 seconds
Time went backwards by 4.00920 seconds
Time went backwards by 4.00929 seconds
Time went backwards by 4.00916 seconds
Time went backwards by 4.00918 seconds
Time went backwards by 4.00924 seconds
Time went backwards by 4.00927 seconds
Time went backwards by 4.00921 seconds

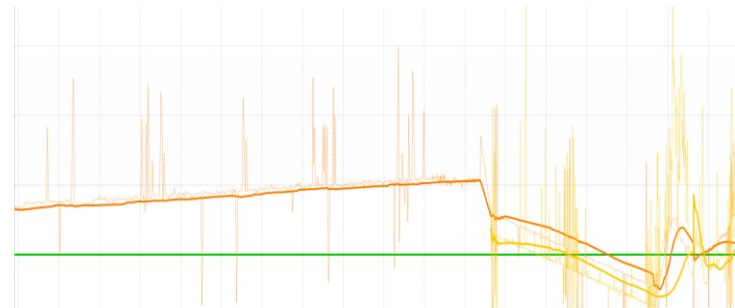
```

¹ Version 2.3.1

The second type of “correction” mentioned by IMC silently slows down the clock on the application server until time is adjusted to the source. Experiments show this behavior on some variants of the open-source clients. The effects of incorrect slewing are harder to detect but also destroy data integrity: computations and transactions will appear to complete far more rapidly than they actually do, timestamps on servers that receive time from alternate sources or slew at a different rate will be out of synch. When the error at the source is corrected, time will speed up on the client while the clock is slewed back. If the error repeats, which is not unusual, the clock will oscillate between too slow and too fast.

Engineering and equipment constraints make it difficult for consumers of time synchronization technology to carry out these kinds of tests – even for the simplest of conditions. Yet there are many more subtle failures that can produce synchronization errors. TimeKeeper® Client software correctly handled the repeated 4 second jumps of our experiment because it has smart filtering that detects the single bad update as spurious. That filtering is designed for the specific characteristics of enterprise packet networks – which are subtle and complex.

With a single time source, TimeKeeper® will reject transient errors and alarm on long duration errors, relying on holdover to stay close to actual time. Given multiple time sources, TimeKeeper can fail-over and alert on microsecond or even nanosecond level errors in the primary time source. In the graph to the left, TimeKeeper is checking a usually reliable PTP source (green) against three unreliable NTP sources pulled in over the Internet. Even in this case, where the secondary sources are low quality, errors can be detected from the contrasting long term behaviors of the sources. In all cases, the TimeKeeper produced traceable log shows how all accessible time sources tracked each other.



TimeKeeper is designed for fault-detection and fault-tolerance and subjected to rigorous on-going test to meet needs of organizations that have due diligence requirements.

For more information, email sales@fsmllabs.com. TimeKeeper is a registered trademark of FSMLabs.