Challenges deploying PTPv2 in a Global Financial company

Pedro V. Estrela
IMC Financial Markets, Amsterdam, Netherlands
Email: pedro.estrela@imc.nl

Lodewijk Bonebakker
IMC Financial Markets, Amsterdam, Netherlands
Email: lodewijk.bonebakker@imc.nl

***Abstract*—This paper describes the challenges encountered when deploying PTPv2 on the worldwide network of a financial company, by upgrading nearly all servers in all data-centers over a period of two years, to achieve global microsecond level accuracy between any pair.**

Acknowledging that PTP was initially designed as a LAN protocol and that all current time-keeping industry efforts are focused on PTP, the issues can be broadly divided into a) issues on the PTPv2 standard itself, b) issues that have to be addressed when PTP is expanded to work over WANs, and c) issues that caused the biggest operational impact on the (tested) implementations.

In all, this paper contributes concrete examples where PTP's byzantine robustness, scalability and efficiency characteristics range between absent to poor – and attempts to raise awareness on the steps needed to build PTP solutions with the characteristics that global users want.

I. Introduction

This paper describes the challenges encountered when deploying PTPv2 on the worldwide network of a financial company, in order to achieve microsecond level accuracy between any two servers (globally). For this, we will describe the issues discovered over the last two years, while deploying PTP for nearly all servers in all our data-centers.

In this paper we specifically acknowledge two facts:

1) the on-going work performed by the industry on PTP has clearly provided customers with a substantial improvement in accuracy, compared to alternative time distribution mechanisms.
2) PTP, as it is currently defined [1], is not (yet) a viable solution beyond smallish LANs[1]

In our view, the future of network time synchronization will clearly be based on PTP, because this is the only network protocol that supports time synchronization on the actual network elements, which is essential for future nano-second accuracy across large networks. However, many large companies would like to achieve high accuracy in geographical scenarios greater than LANs, in order to improve upon the accuracy of their current NTP deployments. The challenge thus becomes how to improve time accuracy while maintaining the same levels of robustness, scalability and cost provided by NTP today.

[1]From the IEEE 1588 document, clause 1.1, "This standard defines a protocol (...) applicable to systems communicating by local area networks supporting multicast messaging"

Table I
A summary of the acronyms used in this paper

ACL	Access Control Lists
BC	Boundary Clock
BMC	Best Master Clock
DC	Data-Center
FINRA	Financial Industry Regulatory Authority
GM	GrandMaster
IGMP	Internet Group Management Protocol
LAN	Local Area Network
MAN	Metropolitan Area Network
NE	Network Equipment
NIC	Network interface controller
NTP	Network Time Protocol
PIM-SM	Protocol Independent Multicast - Sparse Mode
RP	Rendezvous Point
TTL	Time To Live
UTC	Universal Coordinated Time

Taking these considerations into account, this paper divides the encountered issues into a) those that affect PTPv2 as it is defined today (i.e., for LANs), b) the issues that have to be addressed when PTP is expanded to work over WANs and c) the issues that caused the biggest operational impact on the (tested) implementations. In all, this paper attempts to raise awareness on the steps needed to build PTP solutions with the nanosecond accuracy, robustness and scalability that global users want.

The rest of this paper is organized as follows. First, we present a high-level overview of our initial global timing facilities (section II); then, detailed sections describe the difficulties of extending PTP into a WAN protocol (section III), and how this exposed a scalability limitation that could be solved with a hybrid unicast/multicast mode (section IV). Then, the lack of Byzantine robustness is explained with a concrete failure example (section V), and how a future "Enterprise profile" could incorporate our findings (section VI). The paper finishes with a summary of the suggestions made to the standard itself. An overview of all the acronyms is found in Table I.

II. Initial IMC Timekeeping mechanisms

A. IMC and its global network

IMC financial markets is a leading proprietary trading firm and a key market maker [2] in various products listed on all the world's major exchanges, with offices in Amsterdam, Zug, Chicago, Hong Kong and Sydney. Our company provides liquidity to the users of the equities (e.g., stocks and indices)

and derivatives markets (e.g., futures and options). In this business, IMC vigorously competes with other similar companies to provide the best prices to the other market participants (e.g. pension funds, investment houses, banks and individual end-users). The competitive nature of our business drives a strong focus on technology, compounded by the regulatory and compliance requirements demanded of financial market participants.

IMC has built and continuously maintains a state-of-the-art technological infrastructure that keeps us competitive. In particular, IMC features a global network that directly connects to over 40 exchanges worldwide, in all major financial locations and spanning all time zones. This network has dozens of data-centers (DCs), either co-located or in proximity to the financial exchanges, all with state-of-the-art switching backbones and inter-connected with a variety of both leased and partially shared high-speed interconnection lines. Together, all these DCs host thousands of servers, most of them performing critical real-time activities and all of them requiring strict traceability of their clocks to UTC for both current (e.g., [3]) and anticipated regulatory/compliance reasons, for risk mitigation and for internal performance testing. The trend for more accurate timekeeping is reflected in the latest upgrades to the financial exchanges themselves, e.g. [4], which now provide customers with micro-second accurate time services.

B. Initial time distribution facilities

Our initial time distribution facilities were similar to other large distributed companies. Most of the offices had a GPS antenna on their roof, each connected to an enterprise-class NTP server. In turn, all clients used standard NTP software packages, while querying multiple servers for robustness reasons. Even with the sometimes very large distances to the GPS, and the well-known NTP protocol limitations [7], this setup provided an acceptable (at that time) worst-case performance of several milliseconds error against UTC.

To improve accuracy when comparing co-located hosts, we deployed an initial software-only trial of PTPv1 on every IP subnet, using the default multicast profile. Although this enabled microsecond comparisons between these hosts, this improvement could not be scaled beyond the single LAN subnet. This happens because the local PTP GrandMasters (GMs) would still be synced to UTC through NTP to our offices' GPS antennas, resulting in the very same [ms] errors between LANs. On the other hand, the clusters are limited in nature, because standard PTP messages do not cross over IP routers, due to their TTL=1 nature[2].

III. Challenges supporting WANs with PTPv2

In this section we discuss our experience of pushing the envelope by deploying PTPv2 on a WAN topology, even though we fully realize this was never the design intention. Yet, the NTP limitations and the desire for better accuracy, combined with the availability of the open source PTPv2 client [8], provided an excellent opportunity for experimentation. During this project, we also collaborated with our partners and vendors on major operational improvements, which now appear in updates to their products and in the current stable version of the open source PTPd implementation [13].

A. Adding Equipment

First, to reduce the average distance to the PTP clients, multiple new GPS-based timeservers were added to selected data-centers in the global network. However, it is completely infeasible to equip every datacenter with a GPS synchronized PTP server; apart from the costs, there are multiple cases where our data-centers are located in the basement of a high-rise building, where roof-access is impractical.

Similarly, it is also infeasible to exchange all IP routers with PTP-enabled versions in a "big bang operation". While our company upgrades its network equipment (NE) quite regularly, mostly following vendor cycles, at time of writing we are not aware of fast, enterprise class, cut-through switches that support PTP in a reliable way; additionally there will always be lower priority network zones that are not upgraded immediately.

B. Extending PTP packet reach

For the above reasons, our first efforts focused on how to extend PTP into a MAN/WAN protocol. The first step was to allow multicast packets to cross IP subnet boundaries, by raising the TTL of the multicast packets to more than "1" in all PTP masters and slaves[3].

However, this in turn raised the complementary problem: different clusters from different countries now see each other, and the Best Master Clock (BMC) algorithm will elect a single GM to be the single active master. While such a design would provide robustness to any of these GMs failing, it could also significantly limit the accuracy for slaves far away from the winning GM (which defeats the whole purpose of deploying multiple GPSs for accuracy improvements).

To avoid this problem, the PTP clusters needed to be separated using one of multiple options:

- The first option is to use PTPv2 "domains" field to group the clocks together. Here, each cluster would be assigned a different domain, and packets belonging to other clusters would be ignored. Unfortunately, this option is not scalable, because all clients from all clusters will be continuously receiving foreign packets at userspace, only to be silently discarded. Additionally, this setup incurs a per-client maintenance overhead each time the topology changes.

[2]PTPv1 specifically forces the TTL to be zero for all PTP messages (p140, "the time to live (...) value for all PTP messages shall be 0. That is, these messages shall not be forwarded by network routers"); PTPv2 actually leaves this undefined for all messages, except Path-delay (p220, "For messages sent to the PTP-pdelay address, the Time to Live (TTL) field shall be set to 1"), being however implicit that the default is still 1

[3]Reference [9] presents a similar method for unicast packets. While initially considered, this option was not pursued because it would introduce an extra layer on the time distribution, and because it suffers from both scalability limitations (see section IV-C) and limited implementation support.

- A second option is to keep all hosts in the same domain, but to add access control lists (ACLs) in the routers to limit the packets to the vicinity of the GMs. Here, the major downside is the substantial work needed to carefully setup the ACLs for the first time, and maintain them when any part of the topology changes, which is an error-prone operation.
- A third option would be to revert to the PTPv1 style of supporting multiple domains, by separating clocks using different multicast groups[4]. In this case, by using separate multicast trees, the underlying multicast routing protocol would only connect together the actual members of the cluster. The downside would be again the per-client maintenance overhead each time the topology changes. In practice, the best case would be to specifically allow for a configurable group, because IMC has more than 4 time domains (a limit on the possible multicast groups defined by the standard itself).
- A fourth option is to create several multicast distribution trees for the same (multicast) group, by tuning the underlying multicast protocol. Taking PIM-SM as an example, multiple Rendezvous Points (RP's) can be setup in multiple locations, each close to the active GMs. Each RP will be configured as an anycast address, i.e. a duplicated IP address. When hosts join the group, multicast routers will translate the IGMP joins into PIM-SM joins, and they simply build a multicast distribution tree back towards the nearest RP, as per the unicast routing table. When packets are sent to the group, these are first routed towards the nearest RP, and then propagated inside the multicast tree as normal.

This last scenario has the least amount of manual work of all four options, because all clients are equal, and it is fully robust to topology changes (like link failures). Manual work is limited to RP manipulation to redefine the cluster's locations, if and when new GMs are added/removed.

C. Dealing with WAN network jitter

A second challenge that becomes more prominent in large networks appears when the number of hops between the GM and slave increases. In this case the probability of packets being delayed increases substantially, mostly due to the queuing effects during bursts in the links connecting DCs (which, due to cost and/or availability constraints, have less available bandwidth than the DC backbones).

The standard solves this issue by adding time support to the network itself, which, as previously mentioned, is infeasible for a company like ours. Instead, the most practical solution at this stage would be to improve the end-points' clock servos with better jitter filters that are able to detect the delayed packets, and lower their weight in the synchronization activities. We observed promising improvements in the presence of network jitter while experimenting with the servo parameters of the open source client, but more work is certainly required.

[4]This feature is already present in the latest open source client.

While we understand that PTP purposefully leaves the actual timing recovery algorithm to the implementation, we believe that the standard should provide some guidance on how to test network jitter. This in turn could be used to perform a fair review and classification of the quality of vendor implemented algorithms[5]. A fundamental problem is that such a test cannot use the same path to more accurately measure the quality of the original algorithm. A possible solution would be to use alternative jitter-free network paths to selected test clients; reference [15] explores this notion for the LAN case, by connecting the client directly to the GM.

D. Monitoring stability and accuracy

As mentioned before, IMC has thousands of hosts located around the globe. Maintaining such a large number of hosts requires continuous monitoring to find all kinds of configuration, functional and performance problems. The most complex of those - performance - is already supported by the PTP protocol through its management messages[6] which return the client's own estimated OffsetFromMaster. The original open source client was less informative by reporting only absolute values.

In addition, we consider that some provisions could be introduced to better estimate the error associated with this value. This could be done by adding a new field to this structure, with the estimated confidence interval of this offset value. With this addition, the performance monitoring systems could better inform the network administrators of the expected validity of the reported offsets, no matter how small or large.

To generate this confidence value, new algorithms could be introduced that take advantage of the known minimum short-term (<20 minutes) frequency stability of the common oscillators, present on all modern motherboards [5].

IV. Return path scalability and operational issues

A. Problem description

The PTPv2 protocol states that all messages are transported using multicast and to a single and common well-known address. We suspect that the reason for this design is its simple configuration and robustness (i.e., the Best Master clock algorithm is highly simplified by assuming that all clocks can hear themselves), and to provide better efficiency on the downstream direction (master to slave), which enables a higher message rate improving the clock accuracy.

However, this design is not scalable for large LANs and WANs, as the reverse private communication from each individual slave to the master (delayReq and delayResp) induces a major performance impact - each private message is received by all members of the multicast group, processed and dropped.

[5]This is the approach introduced by ITU in their telecommunications profile G.8265

[6]See clause 15.5.3.4.1 of the PTPv2 standard

B. Operational challenges

Requiring a single multicast address and group caused severe operational problems for IMC, as it requires all hosts to be both senders and receivers of a single shared WAN multicast group (with the clock separation methods described in section III.B). This requires an "all-to-all" semantic that is, in practice, far more complex to build and maintain than the regular "one-to-many" multicast semantics, primarily because of asymmetric routing issues.

This design specification was the root cause for a multitude of operational problems that plagued our PTP deployments, with a particular incidence at the edges between PTP clusters. Here, the affected slaves would receive Sync packets correctly, but their delayReqs would be multicast-routed to other GMs, which would silently drop those packets.

C. Standard solutions

The standards bodies (IEEE and ITU) offer two possible solutions to this problem. However, the following will clarify why neither of these solutions are suitable to IMC:

- PTP reduces the reverse multicast overhead by adding hardware components inside the network equipment to serve as boundary clocks (BC), and using peer-to-peer delay measurements instead of end-to-end delay measurement. In this case, the multicast usage is severely minimized, as the clients only communicate directly with their BC (i.e., their switches). However, this solution is unsuitable to IMC as we desire a transparent solution that uses the existing network "as-is", to leverage our existing substantial investment.
- an alternative is proposed by the telecommunications profile - ITU G.8265 [6]. This is an unicast-only approach, where all packets are sent in unicast to pass over legacy networks without multicast capabilities. This is supported by adapting other mechanisms (e.g., BMC algorithm) to the unicast versions without relying on the existence of a shared communication media. Again, this solution is unsuitable for IMC, as it is not supported by the open source client, and it would just move the performance hit to the downstream direction instead: because of the unicast nature of the downstream Sync packets, the rate of these packets arriving at each client is dramatically reduced, leading to much longer drift periods.

D. Proposed solution: Hybrid mode

In the end, a new mechanism was developed and contributed to the open source client, that combines the best possible efficiency with scalability[7]. In this scheme, all public messages continue to be transmitted in efficient multicast mode, but the private end-to-end delayReq/Resp exchange messages are sent in unicast directly to the IP address of the GM/BC or to the Slave, respectively:

1) Upstream, the delayReq is unicast to the IP address of the current Grandmaster (GM) or local Boundary Clock, with the L5 Unicast bit set. When the BMC algorithm chooses a new master, e.g. after a fail-over event, the slave will immediately send the subsequent requests to the IP address of the new master. It should be noted that the internal L2 addresses present inside the L5 PTP fields are still used, in order to maintain transparency to the rest of the protocol.
2) Downstream, the delayResp is unicast to the IP address of the requesting slave, by reversing the source and destination IP addresses, and keeping the L5 Unicast bit on the delayResp message.

As with any upgrades dependent on distributed features, they have to work with the variety in the hardware and software of masters already deployed on a network. While companies like IMC can perform orchestrated upgrades in their key servers fairly quickly, if this variety becomes a significant problem (because of backwards compatibility), then an unused flag in the announce message could be used to identify the hybrid-mode capable masters. Alternatively, a simple solution is to let the slaves cancel this mode if no unicast delayResp is received after a pre-configured number of tries.

E. PTP GrandMaster overload

The work performed on the hybrid mode also exposed another delayReq issue that could disrupt time in real-world deployments. The PTP standard defines that the maximum allowed rate of delayReqs a particular GM can receive is contained on the delayResp message itself. If a very large number of clients try to connect to a GM for the first time, they will send delayReq packets at the default rate of 1 per second[8]. If the GM becomes overloaded (because of the large number of requests), then the clients are implicitly allowed to continue to send delayReqs packets at this rate without actually receiving the correct number (perpetuating the lock-up situation).

While there are multiple ways to avoid this situation, the only way to address all corner cases is to make sure that the slaves always know the right rate for this particular deployment before they send any delay packets. A trivial way to correct this implicit circular dependency is to include the maximum delayReq rate in the Announce message instead, since it is processed earlier in the state machine.

V. Robustness to Byzantine faults and Security

A. Problem description

The PTP protocol is a fairly recent standard, and was clearly optimized for small and highly controlled environments only. It achieves its superior accuracy by a straightforward approach, where a single time source sends its own time to multiple slaves at high rates, enabling them to continuously adjust their

[7]This proposal is similar to the idea introduced in Appendix I of the telecommunications profile [6]

[8]See clause J.3.2 of the PTPv2 standard

own clock offset. This asymmetric design (called "master-slave" in [11]) is one of the reasons why the protocol is more accurate: by not having to wait for the response of multiple time sources, there is no need to wait for consensus, so the PTP slaves are free to adjust the clock immediately every time they receive a Sync packet. On the other hand, this also explains why the protocol is very fragile in the presence of failures or other unwanted behaviour: by depending on a single external time source, the slaves cannot compare the just-received offset to anything else, so they implicitly need to trust it blindly.

In contrast, the NTP protocol has a much longer history, spanning 30 years of electronic time distribution, and was specifically built for robustness in uncontrolled environments - without special high accuracy considerations. The normal operation mode of NTP is highly symmetric (called "democratic" in [11]) where every client periodically questions its whole list of servers sequentially, and then performs a consensus check in order to find which ones to trust. In this process, the "true-chimers" are separated from the "false tickers", to find the average time as given by the group as a whole. More details of the NTP operations can be found in [14], which provides a informative overview on the complexities of building a time protocol.

B. Initial mitigation of production issues

The implications of this aspect of the PTP protocol only became clear after PTP was deployed worldwide, using multiple GMs to service thousands of clients. On several occasions, a GM bug caused the (single) time source to send time without leap seconds information, for two (!) hours. As the active GM continued to send "Announce" packets as normal, with the same BMC values (priority, clock class, etc.), the inactive GMs had no reason to take over. 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. Both cases are unacceptable in our regulatory environment (namely FINRA rule 7430 [3]).

In other cases, the active GM stopped being the best source for some reason (no GPS connectivity, stopped sending packets, GM bugs, etc.), but the PTP robustness mechanism failed as well (incorrect BMC configuration, multicast problems, etc.), allowing all clients to drift their clock for hours.

These multiple issues were first controlled by deprecating clock stepping, and later by deprecating long (i.e., more than 2 seconds) slews. At the same time, the specific issues were addressed in cooperation with the vendor[9]. In addition, the identified problems resulted in multiple improvements to the configuration of the production time infrastructure and the network itself, especially on the multicast-forwarding configuration.

While all these measures have objectively improved the situation, they are still insufficient to cover all known and unknown corner cases, as the slave clocks may still drift for long periods of time (at best), or be slowly slewed to any desired offset by a faulty, hacked or GPS-spoofed server (at worst). In all cases, this susceptibility introduces a significant operational overhead to keep the time synchronization system operating and monitored 24 hours/7 days a week.

C. Possible Byzantine robustness

The issue of how to make PTP and other master-slave protocols more robust has been discussed in the past. The work presented in [10] and [11] proposes a two-level approach, where multiple GMs are tied together in a robust "master-group", and present themselves as a single virtual master to the slaves via a so-called "master group speaker".

This has the advantage of supporting GM failures in a transparent way, because it requires no changes to the regular IEEE 1588 slaves. However, as it was illustrated in the "leap seconds bug" example, this actually moves the single point of failure to the "group speaker" instead, with the same implications.

To address these problems, we identified that the root cause lies in the PTPv2 standard itself; the problem being common to any other implementation or any other master-slave protocol where clients trust a single time source.

To address Byzantine failures, where "traitors" actively send the worst possible misleading time - for whatever reason - it is imperative that the clients themselves are able to listen and reply to multiple sources simultaneously, using a minimum of $2T + 1$ reference time servers to successfully absorb T "traitors", according to the proof of Fetzer and Cristian ([12])[10]. One of the possibilities to achieve this would be to leverage the alternateMasterFlag, already present in the standard.

However, the difficulty then becomes how to maintain the high accuracy of the current PTP GM, with a guaranteed worst case accuracy on the backups. Therefore, the PTP client needs a heuristic to judge the 'trustworthiness' of its current GM, as compared to the group, and to be able to switch when needed.

D. Security considerations

A related issue to Byzantine robustness is the topic of security. This feature is important to prevent both targeted attacks from malicious third-parties, but also to serve as a simple safety-net on basic configuration mistakes[11]. PTP already has basic features (Annex K) for endpoints' authentication and message integrity, based on symmetric pre-shared keys. While this setup may be acceptable for controlled environments like IMC, adoption at larger scale may still be delayed because of its experimental nature, and lack of (commercial) implementation support.

[9]We intentionally choose to not disclose the specific vendor name

[10]From [12], "we prove that at least $2F + 1$ reference time servers are necessary for achieving external clock synchronization when up to F reference time servers can suffer arbitrary failures "

[11]For example, simply launching a software client with the wrong flags (master mode, low Priority1 value) is enough to redirect all clients away from the correct GM.

VI. Proposal: Enterprise PTP profile

Reviewing the work done and looking towards the future, it is clear that IMC would like a standards compliant and supported PTP implementation. This PTP implementation would follow a standard adapted to our reality, by covering our efficiency, transparency and scalability concerns:

- Limited number of very high quality time sources;
- A large, but highly controlled network, which initially lacks any form of PTP support;
- Thousands of remote software clients;
- Dozens of remote clients that will be expanded with hardware support for PTP.

We believe that this reality fits other industries outside financial services as well - global companies may be delaying their PTP deployment because of these same concerns.

While most companies would definitely appreciate a NTP replacement with higher accuracy, they most likely will require a *smooth* migration path which allows them to replace their NTP Clients and TimeServers without (necessarily) having to immediately upgrade their legacy network switches/routers as well. Taking advantage of the fact that most private networks are typically highly controlled, we think that there is a window of opportunity to create a new *Enterprise PTP profile* that provides both immediate gains and is future-proof for later network upgrades.

This enterprise profile would be similar to other standardized profiles already available for other industries (namely telecommunications and the power industry), and would cover the following aspects:

- UTC distribution to user-space applications, with sub microsecond best-case LAN accuracy, and single digit microsecond WAN accuracy;
- Smooth Migration, by not requiring - but, if present, taking advantage of - network time-aware components;
- Scalability, via a hybrid Multicast+Unicast distribution model, and using L3/End2End, and either 1- or 2-step clocks;
- Efficiency, by using improved jitter filters and local clocks on the network edges, and NIC timestamps if present;
- Robustness to Byzantine errors, and more deployment control (e.g., customizable multicast groups);

VII. Conclusions

This paper described the challenges encountered in deploying PTP in the WAN of a global company, by exposing the actual issues that only appear at such a large scale. We identified the following issues and limitations of the PTPv2 standard itself, which combined, may be delaying adoption of PTP at large. These issues are:

- The inability to separate clocks in a efficient way, using custom multicast groups;
- The absence of a management field to store the confidence interval of the OffsetFromMaster slave value;
- The standardization of a highly efficient Hybrid multicast-unicast mode;
- The relocation of the GM maximum allowed delayReq period to the Announce message;
- The standardization of a multi-source mechanism that is robust to Byzantine failures;
- The absence of standardized network jitter models, to enable testing of competing stability algorithms;
- The standardization of a security mechanism based on pre-shared keys;

Acknowledgments

The authors would like to thank the vendors for their collaboration and their colleagues for discussions and review.

References

[1] "IEEE standard for a precision clock synchronization protocol for networked measurement and control systems," *IEEE Std 1588-2008 (Revision of IEEE Std 1588-2002)*, pp. c1 –269, 24 2008.

[2] Authority for the Financial Markets, "High frequency trading: The application of advanced trading technology in the european marketplace," November 2010. [Online]. Available: http://www.afm.nl/en/professionals/afm-actueel/rapporten/2010/hft-rapport.aspx

[3] FINRA, *Order Audit Trail System (OATS) Reporting Technical Specifications*, Financial Industry Regulatory Authority Std., April 2012. [Online]. Available: http://www.finra.org/Industry/Compliance/MarketTransparency/OATS/TechnicalSpecifications/

[4] Deutsche Börse, "Exserves time service," "Deutsche Börse", Tech. Rep., April 2012. [Online]. Available: http://deutsche-boerse.com/it/dispatch/en/kir/gdb_navigation/technology/30_Access_Products/20_ExServes/80_ExServes_Time_Service

[5] J. Ridoux, D. Veitch, and T. Broomhead, "The case for feed-forward clock synchronization," *IEEE/ACM Trans. Netw.*, vol. 20, no. 1, pp. 231–242, Feb. 2012. [Online]. Available: http://dx.doi.org/10.1109/TNET.2011.2158443

[6] J.-L. Ferrant, M. Gilson, S. Jobert, M. Mayer, L. Montini, M. Ouellette, S. Rodrigues, and S. Ruffini, "Development of the first ieee 1588 telecom profile to address mobile backhaul needs," *Communications Magazine, IEEE*, vol. 48, no. 10, pp. 118 –126, october 2010.

[7] J. Ridoux and D. Veitch, "Ten microseconds over lan, for free (extended)," *Instrumentation and Measurement, IEEE Transactions on*, vol. 58, no. 6, pp. 1841 –1848, june 2009.

[8] K. Correll and N. Barendt, "Design considerations for software only implementations of the ieee 1588 precision time protocol," in *In Conference on IEEE 1588 Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, 2006.

[9] A. Novick, M. Weiss, K. Lee, and D. Sutton, "Examination of time and frequency control across wide area networks using ieee-1588v2 unicast transmissions," in *Frequency Control and the European Frequency and Time Forum (FCS), 2011 Joint Conference of the IEEE International*, may 2011, pp. 1 –6.

[10] G. Gaderer, P. Loschmidt, T. Sauter, and G. Bumiller, "Investigations on fault tolerant clock synchronization within a powerline communication structure," in *Power Line Communications and Its Applications, 2006 IEEE International Symposium on*, 0-0 2006, pp. 178 –183.

[11] G. Gaderer, S. Rinaldi, and N. Kero, "Master failures in the precision time protocol," in *Precision Clock Synchronization for Measurement, Control and Communication, 2008. ISPCS 2008. IEEE International Symposium on*, sept. 2008, pp. 59 –64.

[12] C. Fetzer and F. Cristian, "Integrating external and internal clock synchronization," *Real-Time Syst.*, vol. 12, no. 2, pp. 123–171, Feb. 1997. [Online]. Available: http://dx.doi.org/10.1023/A:1007905917490

[13] "Ptpd svn changelog," 2012. [Online]. Available: http://ptpd.svn.sourceforge.net/viewvc/ptpd/trunk/ChangeLog?revision=179

[14] D. L. Mills, "A brief history of ntp time: memoirs of an internet timekeeper," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 2, pp. 9–21, Apr. 2003. [Online]. Available: http://doi.acm.org/10.1145/956981.956983

[15] Symmetricon, "Measuring software based ieee 1588/ptp slave accuracy," Symmetricon, Tech. Rep., 2010.