A Security Analysis of the Precise Time Protocol (Short Paper)

Jeanette Tsang and Konstantin Beznosov

Laboratory for Education and Research in Secure Systems Engineering
University of British Columbia
Vancouver, Canada
tsangj@interchange.ubc.ca, beznosov@ece.ubc.ca
https://lersse.ece.ubc.ca

Abstract. This paper reports on a security analysis of the IEEE 1588 standard, a.k.a. Precise Time Protocol (PTP). We show that attackers can use the protocol to (a) incorrectly resynchronize clocks, (b) rearrange or disrupt the hierarchy of PTP clocks, (c) bring the protocol participants into an inconsistent state, or (d) deprive victim slave clocks from synchronization in ways undetectable by generic network intrusion detection systems. We also propose countermeasures for the identified attacks.

Keywords: IEEE 1588, Precise Time Protocol, Network Time Protocol, security analysis, time synchronization.

1 Introduction

The ability to precisely synchronize clocks among distributed components is critical for electrical power systems, industrial automation, telecommunication systems, military applications, and other fields where timing is crucial to their correctness and performance. The components of these distributed systems often contain real-time clocks that control their performance and coordination.

The IEEE 1588 standard [1] specifies a precision clock synchronization protocol for networked measurement and control systems that may utilize non-IP networks. It is equivalent to the IEC 61588 standard [6]. Both standards are known as "Precise Time Protocol" (PTP). In this paper, we use the terms "PTP" and "IEEE 1588" interchangeably. Networked heterogeneous systems can employ this protocol to synchronize clocks with accuracy in the sub-microsecond range. Real-world examples of PTP applications can be found in [9,10].

Although existing protocols such as the Network Time Protocol (NTP) [7] and the Global Positioning System (GPS) are used to synchronize clocks within networks, PTP is the only one that offers accuracy at the sub-microsecond level for small self-administered networks [4,5]. NTP also requires the underlying network to be IP-based, whereas PTP does not have this restriction.

Even though PTP is being positioned by the industry to serve as a key time synchronization technology for automation and control [10], its resilience to security attacks has not yet been publicly studied. For PTP to serve its intended

P. Ning, S. Qing, and N. Li (Eds.): ICICS 2006, LNCS 4307, pp. 50–59, 2006.

role, the automation and control community needs to be aware of the protocol's security properties.

In this paper, we report on the results of our security analysis of the 2002 revision of the IEEE 1588 specification [1]. The analysis focused on PTP's message transmission period, i.e., when networked devices exchange synchronization messages. The results of this analysis can assist the developers and users of PTP-based technologies in identifying the security requirements and developing the necessary security mechanisms for the protocol.

We made several assumptions for the purpose of analysis. In order to focus on PTP-related attacks and to exclude attacks specific to other protocols from our analysis, we assumed that the network being analyzed is a "closed network," i.e., none of the network nodes is connected to other external networks, such as the Internet. For those environments where the above assumption does not hold, additional types of attacks (e.g., distributed denial of service), including those specific to general-purpose network protocols, such as IP, UDP, and TCP, have to be taken into account. As a consequence of the first assumption, our second assumption was that only adversaries who have direct access to the PTP network can initiate attacks. Our third assumption was that the attacker(s) can mount passive (message eavesdropping) as well as active (message modification, removal, and injection) attacks. On the other hand, we could not make the assumption that IPSec [8] and its supporting services (e.g., key management) are available in the automation and control system that uses PTP because the PTP specification does not mandate IPSec.

Due to the lack of built-in protection, PTP messages can be easily tampered with by anyone who has access to the network. More importantly, the results of our analysis also suggest that attackers can easily use this weakness to incorrectly resynchronize clocks or to illegally rearrange (or even disrupt) the hierarchy of PTP clocks. Additionally, the protocol lacks a mechanism for detecting and compensating "out of range" data that can result in an inconsistent state of PTP participants. Furthermore, we discovered several PTP-specific attacks that, we believe, are very hard to detect by a network intrusion detection system, unless it maintains the state of the victim PTP clock hierarchy, which could be expensive.

The rest of the paper is organized as follows. Background on the PTP is provided in Section 2. Section 3 describes the attacks that we identified PTP to be vulnerable to. We draw conclusions and outline future work in Section 4.

2 Background on the Precise Time Protocol

This brief overview of the standard is based on the 2002 revision of the IEEE 1588 specification [1]. A more detailed description can be found in [12]. The main two elements of any PTP network are the *clock* and *port*. The *clock* is a network node that can provide measurements of time. An example of a clock is a network switch connected to two or more subnets. Its connections with the subnets are referred to as *ports*. The switch is referred to as a *boundary clock*, which has more than one port. *Ordinary clocks* have only a single port.

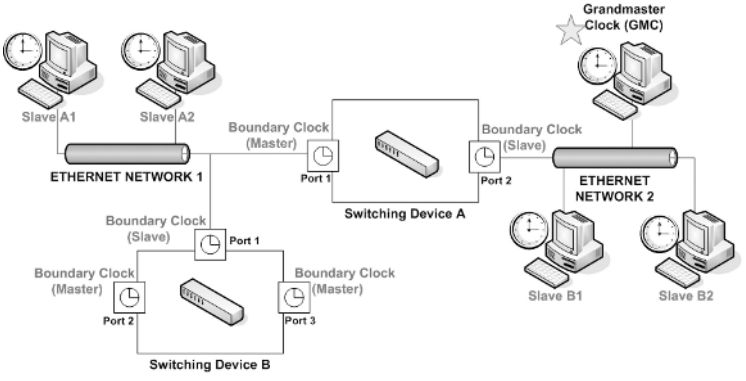


Fig. 1. A hypothetical PTP network (adapted from [3])

Each clock port serves as either a master or slave clock. The *master clock* is used as a reference for calibrating all of the *slave clocks*. Above all, the *grand master clock* (GMC) is the root clock that all the master clocks of the subnets are synchronized to. The master clock and the GMC are elected via the best master clock (BMC) algorithm, which is executed by every port individually and autonomously. The algorithm ensures that there is only one master clock active at any given moment on any subnet. The BMC algorithm is explained in detail in [12], and its vulnerabilities are analyzed in Section 3.1 of this paper.

Figure 1 shows a hypothetical PTP network with two subnets connected via one switch. Switching Device A is a boundary clock with two ports. Port 2 acts as a slave, and port 1 acts as the master clock. The standard allows messages to be transmitted either directly in Ethernet frames or as UDP payload. PTP defines both management and time synchronization messages. For the purposes of this paper, we are concerned only with the latter. There are four types of synchronization messages used in the PTP protocol: `Sync`, `Follow_Up`, `Delay_Req`, and `Delay_Response`. They are used for the regular synchronization procedure, and are propagated only within one PTP subnet. The contents of these messages are listed in Appendix B of [12]. Time synchronization is performed over three phases: master clock selection, time offset correction, and communication delay measurement. The first two phases are executed every synchronization interval, which by default is 2 seconds [5]. Delay measurement is initiated by each slave individually on irregular bases, between 4 and 60 seconds by default [5]. The message flow during time synchronization is illustrated by the example shown in Figure 2.

The `Sync` message is multicasted by the master clock to all of the slaves on the subnet. The main purpose of the message is to deliver the estimated time that it has left the master clock. Upon receipt of this message, the slave clocks record the reception time, which is used to calculate the offset between the slave and master clocks. The optional `Follow_Up` message is sent immediately following the `Sync` message by the master , and contains the precise sending time of the `Sync` message. After receiving the `Follow_Up` message, the slave clock calculates

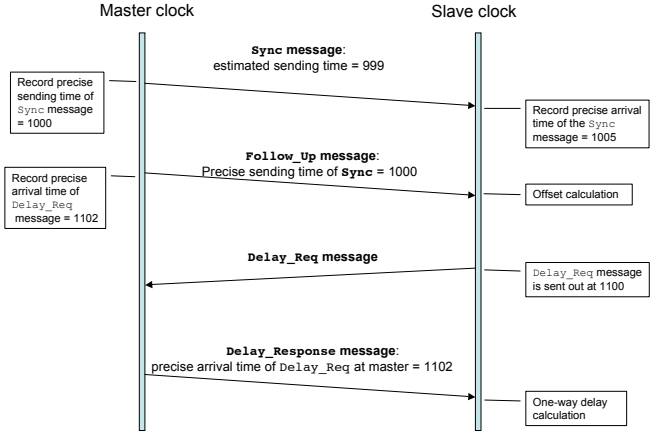


Fig. 2. An example of message flow during time synchronization

the offset. Communication delay between each slave and the master clock is computed with the aid of delay request and response messages.

Referring again to the example in Figure 2, the one-way delay was calculated from the `Delay_Req` and `Delay_Response` messages as $[(1005 - 1000) + (1102 - 1100)]/2 = (5 + 2)/2 = 3.5$ time units, whereas the offset was $1005 - 1000 - delay = 5 - 3.5 = 1.5$ time units. The slave's clock was then recalibrated using the offset value.

3 Threat Analysis

Like Bishop's security analysis [2] of the NTPv2, we studied the goals, attack methods, effects of attacks, and possible countermeasures for the following five types of threats: modification, masquerading, delay, replay, and denial of service. Specifically, we analyzed how these attacks could jeopardize synchronization objectives of PTP participants. The following subsections discuss results of our analysis for each of the five threat types. A more detailed analysis of the threats can be found in the long version of this paper [12].

3.1 Modification

The goal of a modification attack could be to: (a) cause denial of service, (b) cause slave clocks(s) to incorrectly resynchronize, or (c) alter the hierarchy of the master and slave clocks. The attack can be launched by manipulating the content of messages. Furthermore, the modification of the messages sent by a master clock would produce the greatest effect, since a master clock can send messages used for both time synchronization and management.

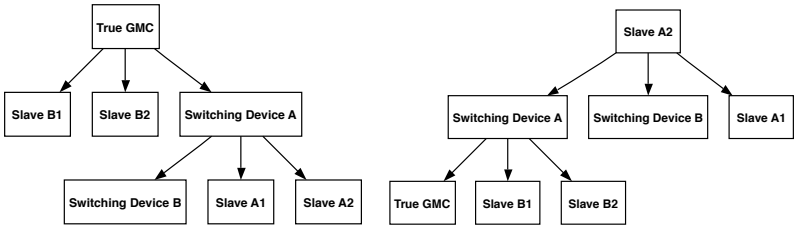
(a) Attacking to deny service: In the analyzed version of the PTP, there is no mechanism for checking the authenticity of a message other than by checking the source of the message against the node's data sets. The data sets contain information such as the local clock and parent clock attributes, and information about the "current master", i.e., the clock whose `Sync` messages are used for correcting time. Slave clocks verify that a message came from the correct master by comparing the `sourceCommunicationTechnology`, `sourceUuid` and `sourcePortId` of the message (see Appendix B of [12]) with the corresponding fields in the parent data set of the slave clock. If the comparison fails, the message is discarded. By modifying the above fields of the `Sync` messages, an attacker can make the matching of `Sync` messages fail; thus, slave clocks would refuse to synchronize with the true current master. This can cause a denial of service (DoS) attack without a generic network intrusion detection system (NIDS) detecting the attack, unless the NIDS "knows" the correct values of these fields. Furthermore, modifying the sequence ID of the message can also lead to a PTP-specific DoS attack variant, which we discuss in detail in Section 3.5.

(b) Attacking to cause incorrect resynchronization: Tampering with the timestamp clock and variance fields of `Sync` messages can cause an incorrect resynchronization of the slave clock(s) or a miscalculation of the network latency. The `originTimestamp` field serves as the record of time at which the `Sync` message leaves the master clock.

(c) Attacking to alter the hierarchy of the master and slave clocks: Wrong information about the grandmaster clock within `Sync` messages can lead to setting the port to a different mode, e.g., slave or passive. Each slave clock executes the BMC algorithm for electing the best master clock for the next round of synchronization. Since the BMC algorithm uses information about the grandmaster clock, by altering the grandmaster clock information in a `Sync` message, an attacker can easily make this message "better" than other `Sync` messages received by most clocks in the given PTP subnet. As a result, this crafted `Sync` message could become the best message for all local clocks from the attacker's subnet. Then, by winning all the comparisons used in the BMC algorithm (Figure 3 of [12]), the attacker can make the victim clock(s) switch into passive mode or slave mode. As a result, the attacker could disrupt or even destroy the synchronization hierarchy of clocks on the victim PTP network.

To illustrate the above attack, consider the PTP network depicted in Figure 1. If, say, the attacker controls Slave A_2, it can start sending `Sync` messages that are "better" than those sent by the true GMC. As a result, Slave A_1 as well as switching devices A and B will elect Slave A_2 as their new master clock. Switching device A will also change its port 1 into slave mode and port 2 into master mode. As a result, the true GMC will switch into slave mode, and the original hierarchy of PTP clocks shown in Figure 3(a), will transform into the hierarchy shown in Figure 3(b), with Slave A_2 being a rogue GMC.

Suggested countermeasures: We recommend employing cryptographic integrity protection on all PTP messages as a basic countermeasure. However, it is not obvious how the issue of key management can be addressed in PTP



(a) Before the attack (b) After the attack

Fig. 3. PTP clock hierarchy before and after a hypothetical attack on the PTP network shown in Figure 1

networks, as the protocol does not have any provisions for registering slave or master clocks, or binding their identities to keys. Efficiency is another issue to be considered when cryptographic integrity protection is employed. Further investigation is needed to decide which cryptographic integrity protection scheme is most suitable for PTP. An alternative countermeasure is to employ port-level security [11] to enforce such simple rules as "only the network interfaces of the true GMC, port 1 of the switching device A, as well as ports 2 and 3 of the switching device B can send `Sync` messages." These access controls based on port-level security might be more effective performance-wise than those based on cryptography. In addition, key management can be avoided. However, the use of port-level security might fall short of enforcing all rules necessary to counter attacks on PTP clock hierarchies. For instance, if switching device A is compromised, then its port 1 can send bogus `Sync` messages even with the above rule enforced. Port-level security also increases the overhead of configuring network switches, and, as a result, increases the risk of "friendly denial of service" due to configuration errors.

3.2 Masquerading

The goal of a masquerading attack in a PTP network is to masquerade as the master clock and use the false identity to launch other attacks. To launch the attack, an adversary can first obtain information about the "true" master clock, and then eavesdrop on `Sync`, `Delay_Req`, and `Delay_Resp` messages that are sent to the slave clocks from the master. Once necessary data are obtained, the attacker can spoof `Sync`, `Delay_Req`, and `Delay_Resp` messages to masquerade as a master clock. Masquerading as a master clock could permit the attacker to send out incorrect timing and management messages to other slave clocks, causing different kinds of damage to the system. For example, the attacker could send out incorrect timing information to slave clocks, leading to errors in the synchronization process.

Suggested countermeasures: We recommend using a centralized or chained authentication process. For centralized authentication, the grandmaster clock

can act as the authentication server. For the chained authentication process, the authentication information of the new clock is passed on to the existing PTP network for verification via a network component that is already connected to the PTP network. This network component has previously authenticated the new clock by its own means.

As with the modification attack described in Section 3.1, port-level security can be used to control which network device can send `Sync`, `Delay_Req`, and `Delay_Resp`. However, such controls reduce the robustness of the PTP in the presence of network failures. Since PTP clocks locally elect best master clocks for time synchronization, the role of master clock needs to be passed from clock to clock as the network topology changes due to device additions and failures.

3.3 Delay

The goal of a delay attack in a PTP network is to delay the arrival of messages at the recipient nodes, thus causing an increase in the values used in the offset and one-way delay calculations. The attack can be carried out through the use of hardware or software to interrupt the transmission of a message between nodes and later re-inject it into the communication channel. By intentionally delaying the reception time of the `Sync` message by a certain slave clock, the attacker may dramatically increase the offset of the slave clock with respect to the master clock, setting the slave clock off synchronization with the rest of the system.

Delaying the reception of the `Follow_Up` message at the slave clock can cause a timeout of the synchronization event. If this condition continues, it may lead to the slave clock being denied synchronization with the master. The slave will either pick the wrong clock on the subnet to synchronize with, or operate based on its local clock, eventually drifting from the true master clock.

A delay caused in the transmission of the `Delay_Req` message has a similar effect as a delay in the `Sync` message. Yet, a delay attack in the `Delay_Req` message can cause a more significant disruption of the synchronization process, because the calculation of the one-way delay is not done as frequently as the offset correction synchronization process. An incorrect value of one-way delay can cause errors in all upcoming offset calculations.

Finally, if the `Delay_Response` message is not received back at the slave clock after a fixed delay-request interval, the whole calibration process of the one-way delay would be voided. An adversary can even launch this attack, and then add in more delay fluctuations, such as new network components, to the system. Since the one-way delay is not being recalculated due to the timeout of the delay request interval, the additional delay from the new components is not being accounted for in the calculation of the offset.

Suggested countermeasures: PTP can be modified to have a backup plan to compensate for the missing or delayed messages. For example, by taking the averages of the `Delay_Req` and `Delay_Response` messages in combination with previous values, the effects of timed out or postponed messages can be reduced.

Also, there should be an algorithm in PTP to determine any abnormal values of the timestamps in the messages. If any abnormality is found for an extended period of time (e.g., the last four one-way delays are significantly higher than the first four), validation of the data against other neighboring nodes might be helpful.

3.4 Replay

The goal of a replay attack is to either create congestion in the network stacks of the clocks, or to desynchronize clocks. The attack can be executed by recording legitimate message(s) being transmitted on the communication path and, at a later time, slightly modifying and then re-injecting the recorded message(s) into the network. The replayed messages would be interpreted as genuine messages. Any occurrence of events caused by the messages would be processed in a First-In-First-Out order. For example, when the `Sync` message is being replayed to the slave clock, the slave clock would record the precise reception time of it. However, we were uncertain how the precise reception time is stored. If there is one storage location for it, then the later replayed message would overwrite the precise reception time of the first `Sync` message. If there are multiple storage places, then the precise recorded times can be queued up and would not lead to a problem. Furthermore, replaying messages can saturate the processing queue at the clocks and congest their network stacks, which may result in dropping authentic synchronization messages from the true master clocks.

Suggested countermeasures: We suggest using an integrity protected network path, e.g., a VPN connection, to prevent messages from being spoofed or injected into the network. An alternative would be to use a capable message authentication mechanism to ensure the authenticity of the PTP messages.

3.5 Denial of Service (DoS)

In addition to generic DoS attacks through flooding communication channels and overflowing communication stacks, an adversary can also deny PTP clock(s) time synchronization service in a protocol-specific way. The significance of this attack is that generic NIDS's are unlikely to detect it.

An adversary could trick the attacked slave clock into rejecting `Sync` and `Follow_Up` messages from the true master clock. Since the attack is the same for both types of messages, we explain it using the `Sync` message case.

For performing DoS using `Sync` messages, the adversary first spoofs the victim slave clock with a `Sync` message using the true master clock address, albeit with a `sequenceID` value greater than the one in the previous `Sync` message. Upon processing this message, the victim clock updates its parent data set accordingly. Denial of service occurs when the true master sends its next `Sync` message to the victim. Since the `parent_last_sync_sequence` in the slave clock has already been incremented upon receiving the spoofed `Sync` message, the `Sync` message from the true master clock has `sequenceID` value less than or at most equal to

the `parent_last_sync_sequence`. Therefore, this `Sync` message is rejected by the victim slave without the master being notified. No synchronization can take place until the true master's `Sync` message has a `sequenceID` greater than the `parent_last_sync_sequence` in the slave clock. The greater the `sequenceID` in the adversary's `Sync` messages, the longer the synchronization service is denied. If the adversary keeps on sending `Sync` messages to increase the victim's value of `parent_last_sync_sequence` before the true master can catch up with it, the victim can be permanently deprived of synchronization with the true master. The adversary starts controlling the time of the slave clock without the victim or its master clock realizing this.

A small-scale DoS attack may cause time synchronization to be less accurate across the system. If only slave clocks are affected, they may then be set to run on a local clock, drifting away from each other due to the differences in each clock's skew. The bigger the scale of the attack, the greater the number of PTP clocks that would have to run on local clocks. Furthermore, the tree-like hierarchical organization of PTP networks allows attacker(s) to increase the scale of the attack dramatically by denying synchronization to just a few boundary clocks that are close to the GMC.

Suggested countermeasures: Employ message authenticity and integrity protection or use port-level security to limit the set of the devices authorized to send synchronization messages. The limitations and drawbacks of both types of countermeasures were discussed in the previous sections.

4 Conclusions and Future Work

Since it is able to provide synchronization accuracy in the sub-microsecond range, the PTP has been deployed in a wide range of application domains, from factory automation to avionics, to power grids, and to military systems. However, the protocol lacks security mechanisms necessary to ensure the integrity of transmitted messages and to validate the authenticity of the sender. We analyzed the effects of five different types of attacks on a PTP network in this paper: modification, masquerading, delay, replay, and denial of service. It can be seen that PTP alone is weak against these attacks, and that additional security mechanisms are necessary to protect a PTP network. Damage caused by failure in time synchronization can be dramatic.

We did not implement the discovered attacks or countermeasures due to the lack of publicly available PTP implementations. When the situation changes, developing a proof of concept for our attacks and countermeasures could be very helpful for evaluating their feasibility and for testing corresponding countermeasures. Because of the critical nature of the PTP application domains [9,10], we prefer to make the community of security professionals aware of our results now.

We limited our analysis to synchronization messages. The damage caused by corrupting management messages or using them to launch attacks (e.g., to change information on data sets of PTP clocks) can be much greater than that resulting from corrupted time synchronization messages, because they can control

all of the data sets as well as the status of a PTP clock. Analysis of management messages is a promising avenue for future research. Another interesting direction is to validate the feasibility of cryptographic countermeasures applied to PTP.

Acknowledgments. The authors would like to thank the following people who provided comments on earlier versions of the paper: Galina Antonova, Matt Bishop, and John Eidson. Craig Wilson helped us to improve the readability of the paper.

References

1. A Precision Clock Synchronization Protocol for Networked Measurement and Control Systems, IEEE Standard 1588-2002, November 2002.
2. M. Bishop, "A Security Analysis of the NTP Protocol Version 2," Proceedings of the Sixth Annual Computer Security Applications Conference, 3-7 December 1990, pp. 20-29, Tucson, AZ, USA.
3. P. Doyle, "Introduction to Real-Time Ethernet II," The Extension—A Technical Supplement to Control Network, vol.5, issue. 4, July-August 2004. Available: `www.ccontrols.com/pdf/Extv5n4.pdf`
4. J. Eidson, M.C. Fischer, and J. White, "IEEE-1588 Standard for a precision clock synchronization protocol for networked measurement and control systems," Proceedings of the 34th Annual Precise Time and Time Interval Systems and Applications Meeting, December 3-5, 2002, Reston, Virginia.
5. D. Mohl, "IEEE 1588-Precise Time Synchronization as the Basis for Real Time Application in Automation," Available: `www.industrialnetworking.com/support/general_faqs_info/Precise_Time_Sync.pdf`
6. IEC, "Precision clock synchronization protocol for networked measurement and control systems" IEC 61588, First Edition, 2004, pp. 158.
7. "NTP: Network Time Protocol," Available: `www.ntp.org`
8. N. Dunbar, "IPsec Networking Standards—An Overview, " Information Security Technical Report, Volume 5, Issue 1, 1 March 2001, pp.35-48.
9. K. R. Harris, S. Balasubramanian, and A. Moldovansky, "The Application of IEEE 1588 to a Distributed Motion Control System," presented at the ODVA CIP Networks Conference, Nov. 16-18, 2004.
10. G. Gaderere, T. Sauter, G. Bumiller, "Clock Synchronization in Powerline Networks," Proceedings of International Symposium on Power Line Communications and Its Application, 2005, pp. 71-75.
11. J. B. Hansen and S. Young, The Hacker's Handbook, CRC Press, 2004, pp.512.
12. J. Tsang and K. Beznosov, "A Security Analysis of the Precise Time Protocol," LERSSE technical report, Electrical and Computer Engineering, University of British Columbia, Vancouver, Canada, LERSSE-TR-2006-02, October., 2005, pp.20, `http://lersse-dl.ece.ubc.ca`.