INVESTIGATIONS ON SECURITY ASPECTS IN CLOCK SYNCHRONIZED INDUSTRIAL ETHERNET

A. Treytl, G. Gaderer, P. Loschmidt
Research Unit for Integrated Sensor Systems
Austrian Academy of Sciences
Viktor Kaplan Strasse 2, A-2700 Wiener Neustadt, Austria
E-mail: *{Georg.Gaderer, Patrick.Loschmidt, Albert.Treytl\}@OEAW.ac.at*

Nikolaus Kerö
Oregano Systems
Phorusgasse 8, 1040 Wien, Austria
E-mail: *Keroe@Oregano.at*

Abstract

The synchronization of clocks in distributed systems is an enabling technology for real-time industrial automation applications using Ethernet. Clock synchronization systems are enablers to bring real-time and automation constraints together with well-introduced technologies used in the office and management level. Nevertheless, the architectural advantages of Industrial Ethernet — seamless integration of management and maintenance — also open new vulnerabilities to the system, including the underlying service of clock synchronization. This paper takes a look at the vulnerabilities and attacks against the clock of synchronized nodes. Special focus is set on hardware-oriented and non-cryptographic measures.

INTRODUCTION

The synchronization of clocks in distributed systems is an enabling technology bringing real time into industrial automation applications using Ethernet. Industrial Ethernet is already said to become the new control bus. Clock synchronization is a vital aspect in using Industrial Ethernet: either it is used to guarantee proper arbitration for real-time services or to synchronize the process on the application level. In many cases, clock synchronization is done in a master-slave fashion.

Well known representatives are IEEE 1588 **[1]** and simplified forms of single-master NTP (Network Time Protocol) **[2]**. The common reason to use master/slave-based clock synchronization approaches is that state-of-the-art communication systems are usually structured in the same way and these protocols offer a simple structure. Famous examples of their application are ProfiNet **[3]**, time-triggered Ethernet **[4]** or Ethernet IP.

Nevertheless, the architectural advantages of Industrial Ethernet — seamless integration of management and maintenance — also open new vulnerabilities to the system, including the underlying service of clock synchronization. This article takes a look at the vulnerabilities and attacks against the clock of

synchronized nodes. Based on this analysis, countermeasures are derived and existing approaches discussed. Special focus is set on hardware-oriented and non-cryptographic measures.

Since synchronized clocks at the local nodes are considered as a middleware service for the applications, an attacker able to influence these local clocks can also compromise the functionality of applications or at least can degrade it. For example, industrial networks like ProfiNet substantially rely on the functionality of local clocks — disturbing their alignment disables the real-time capabilities of the network. Especially for master/slave-based systems like IEEE 1588, the potential to compromise all connected nodes is very high, since manipulating the master will cause the complete system to fail.

Consequently, the locally kept time has to be protected in such a manner, that: a) the clock rate is adjusted to the respective master, and b) during an observation period, the mean value of the clock state is kept within an application-defined error interval. While b) seems to be a clear requirement, as the goal of clock synchronization is to keep the local clocks as tightly together as possible, also a) is of crucial importance: In fact, the resynchronization period of the clock synchronization algorithm may be much longer than the accuracy of the derived time. For example, an application deriving a periodic interrupt from its local clock may generate interrupts more frequently than the synchronization interval. In this case, not only the correctness of the medium clock state is important, but also a correctly adjusted clock rate.

The main security goals for industrial systems, including their underlying infrastructure such as IEEE 1588, are integrity and authentication, as well as availability. Confidentiality and nonrepudiation are usually not considered to be important in today's system. For classification of attacks, on the one hand, the attack target (master, control loop, and slave) and the violated security goal will be used. Fig. 1 and Table 1 give an overview of possible points of attack. The attack numbers listed in brackets allow identification throughout the paper. Countermeasures will then be given in the next section.

DIRECT ATTACKS ON MASTER AND SLAVES

The first issue is the protection of nodes themselves. In general, two kinds of influences can be identified: a) *direct manipulation* of data, such as malicious programs changing the local clock or manipulating data, and b) *indirect attacks* influencing the performance of the system. E.g., in systems without hardware timestamping, malicious programs affecting the performance of the network stack, i.e. influencing the time a packet spends between timestamping and actual sending by (over)loading other parts of the system, will influence the precision and accuracy of the time at the node side. Also, blocking of a single node due to a denial of service (DoS, (1) in Fig. 1) will be in this category and is a very important, yet easy to perform, attack.

Additionally, the severity of the attack will be defined by the targeted data. Compromising a common key such as used in IEEE 1588 will in general do more harm than slightly decreasing the accuracy of the clock. Protection, therefore, should follow a defense in depth principle.

BYZANTINE MASTERS

State of the art in IEEE 1588 is that initially every node announces its own accuracy. As a matter of fact, the announced accuracy is neither verified nor checked. Moreover, if two masters announce the same stratum, the decision as to who wins the master election is made by evaluation of the MAC address. Very famous attacks in the Ethernet world show how easily this address can be manipulated and an attacker can successfully set up a Byzantine or "babbling idiot" master, i.e., a master announcing a wrong time, and manipulate the system time ((2) in Fig. 1). In a similar attack, a Byzantine master gains control by

increasing the sequence number of packets. Only proper master authentication and authorization can help at this point.

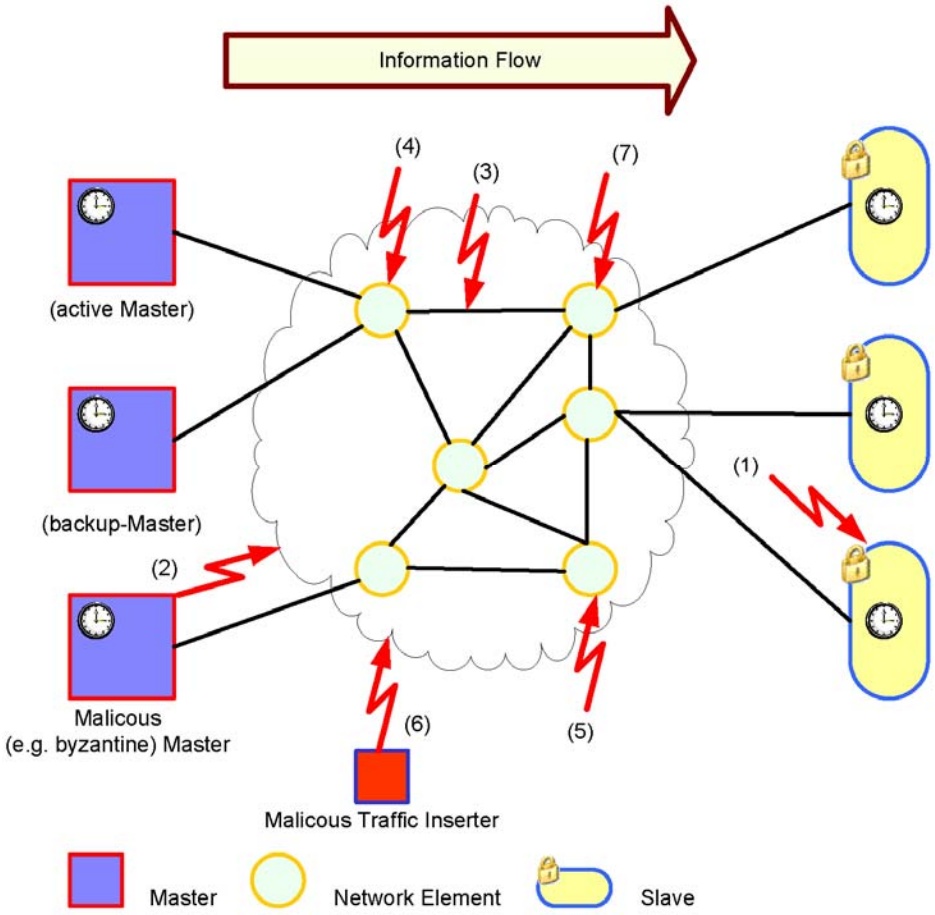

Figure 1. Principal system of master/slave-based clock synchronization and indication of attack points (numbers will allow lookup in text).

DISTURBANCE OF THE CONTROL LOOP

Interrupting the synchronization by preventing transportation of packets for clock synchronization over the network is a low-cost attack. Points of attack are physical interruption of the network ((3) in Fig. 1); deletion of packets by removing the packet within malicious switches, routers, or gateways if the attacker has access to these devices; and blocking of packets by overloading the transmission capacity via a DoS attack ((4) in Fig. 1). A complete service interruption can easily be detected, but is often only to be solved by organizational measures outside the clock synchronization protocol. More complicated is the detection if the deletion only concerns individual and selected clock synchronization packets, since these messages are not acknowledged. The loss of single packets can hardly be detected, but will heavily affect the control loop. Experience gained in the REMPLI project shows that local substitution (artificial *sync*

packets) of lost or deleted packets by delay values calculated statistically from previous delays can minimize these negative influences.

Manipulation of control loop packets will allow complete control of clocks at the slave. All types of packets *sync*, *delay*, and *delay-request/-response* packets) are affected, and the only prerequisite is that an attacker is not violating timeouts giving an additional delay due to manipulation ((5) in Fig. 1).

Insertion of packets ((6) in Fig. 1) at this point is different from packets inserted by a Byzantine master. While a Byzantine master is creating a phony but correct sequence of packets, this kind of insertion deliberately adds packets to a regular sequence. Especially, replay attacks reusing old intercepted packets have to be warded off, but also packets causing state changes, e.g. to a new delay measurement, can cause problems due to interruption of normal operation.

The last category is the malicious delay of packets within the boundaries of the protocol ((7) in Fig. 1). This kind of attack emulates a changing delay of the network and will introduce a time offset up to the cycle of *sync* messages. Hence, mismatch of clocks in the magnitude of seconds can be achieved.

Table 1. Security threats for clock synchronization.

	Attack	Result of Attack
1	denial of service	no service available
2	Byzantine master	complete control
3	interruption of control loop	deviation determined by precision of local clock
4	removal of packets from control loop	deviation determined by precision of local clock
5	packet manipulation	complete control
6	packet insertion	offset up to sync cycle depending on implementation
7	selective packet delay	offset up to sync cycle

COUNTERMEASURES

Countermeasures have to ensure the required security goals. For protection of messages, organizational as well as cryptographic measures will be used: a) O*rganizational measures* handle attacks manipulating the ability to transmit messages. First approaches investigated and developed by the authors within the REMPLI project **[5]** are QoS (Quality of Service) monitoring or the insertion of virtual interpolation values. b) *Cryptographic measures* protect against manipulation of messages. Symmetric as well as asymmetric cryptographic algorithms can be used. The main difference between the two kinds of algorithms is the overhead for message protection. The following subsection will give an overview what can be achieved including security measures planned for IEEE 1588 version 2.

CRYPTOGRAPHIC COUNTERMEASURES

IEEE 1588 version 2 introduces a security system based on secure hash functions. For message and node authentication, the cryptographic hash function HMAC-SHA-1 **[6,7]** will be used. Security data are appended to the IEEE 1588 message in a special security TLV. This TLV includes all measures to protect message integrity and prevent replay attacks. This is achieved by building a cryptographic checksum (ICV) from a key and the complete message data using the HMAC-SHA-1 function. The resulting ICV can be either 12 or 16 bytes long. Replay protection is realized by introducing a counter that is increased for every message sent. Receivers on the one hand recalculate the ICV and detect manipulations by comparing the ICVs. If they are different, the message has been manipulated. In a similar way, the replay counter of an incoming message is compared to the replay counter of the last message received. If the new replay counter is less than or equal to the old replay counter, the message is replayed and must be discarded.

Crucial to this system is that all nodes in a domain possess the same key and have established a relationship of trust. This is done by a three-step mutual challenge-response algorithm that establishes a security association. Only if master and slave successfully authenticated themselves will they accept messages to be further processed. In this way, the impact of DoS attacks can be mitigated, since illegal packets can be silently discarded at a very early stage in the protocol stack and do not consume resources of further processing.

The security gained is paid for by the overhead of the security TLV appended to every message. The size of the security authentication TLV is 26 bytes for 96 bit HMAC-SHA-1. This will result in a typical overhead between 10 and 50 percent for typical payloads of IEEE 1588 messages.

Confidentiality protection might only be important for commercial use in public networks where the service should be restricted to a certain user group. In this case, the additional overhead due to padding can be estimated to be a maximum 8 to 32bytes, depending on the block size of the used cipher. Other applications show that the overhead is significantly lower than for integrity protection. In IEEE 1588, services for confidentiality are not offered.

AUTHENTICATION AND KEY SYSTEM

Version 2 of IEEE 1588 defines a system that uses symmetric cryptography with a common key shared within a domain (a master and the slaves connected to it). The actual process of key distribution is left open, yet there exists the possibility to change between different keys. A first approach will be to "manually" distribute these keys at setup time.

Although this approach satisfies the requirements of an IEEE 1588 system, including transparent switches from the security point of view, a real-source identification would be favorable. Authentication of a legitimate master is a very important issue in this context to protect against Byzantine and malicious masters, as well as packet insertion. Due to the usage of multicast messages, only asymmetric cryptography can be used to achieve secure and distinct, even irrefutable, authentication of the sender of a message. The main difference between symmetric algorithms is the overhead introduced.

Overhead for symmetric algorithms can be typically assumed to have a size of 16 bytes, while the RSA asymmetric algorithm requires 128 bytes, and asymmetric elliptic curve cryptography (ECC) around 20 bytes **[8]**. This results in an overhead of 250 percent for RSA due to its large block size of minimum 128 bytes, which does not efficiently fit IEEE 1588 messages, and 30 to 40 percent for ECC. Since most implementations of IEEE 1588 are based on UDP/IP over Ethernet, which requires a minimum frame size of 64 bytes or 512 bytes for Gigabit networks, the overhead, although high, usually can be neglected.

Only for dedicated implementations on bandwidth-limited communication media such as power lines **[5]**, the overhead must be accounted for and might object asymmetric algorithms. Nevertheless, it must be always considered that asymmetric algorithms require higher resources and are hard to implement on small-scale embedded systems. Beside required resources, the time to calculate the ICV is critical due to jitter introduction during the sending process. E.g., ECC takes 11.3 ms for signature and 60 ms for verification, whereas RSA requires 43.5 ms for signature and 0.65 ms for verification (see also the section about hardware timestamping).

Key management is the second important issue, since in systems that require high frequency sync messages to keep the demanded precision, keys also need to be exchanged frequently. Key management is, therefore, an important performance issue for such systems and will also heavily influence the scalability of the system. It must also be considered that using different keys for clock synchronization, entity authentication and key management increases security and demands for advanced key management.

For symmetric algorithms, incremental key distribution and hierarchical key-sub-groups might be used to reduce the amount of messages needed to distribute keys. The problem with symmetric algorithms is that the key has to be distributed confidentially, since both sides can use it for encryption. A master has to hold keys for broadcast in the group, keys for exchanging keys, and one key for the unicast delay for each slave.

Asymmetric algorithms have advantages in this respect, since the public keys necessary for verification can be freely distributed and, therefore, broadcasted without protection. For asymmetric algorithms, a slave node only has to hold the key of the master and its private key for sending messages to the master. The master still has to keep public keys for each slave and one private key for sending broadcasts.

In systems including transparent clocks, topics like recording and securing of the itinerary are important. Although IEEE 1588 also covers secure transparent clocks, they must possess the symmetric secret keys to manipulate messages. If multiple synchronization domains are routed via a transparent clock, this clock also has to identify packets depending on source and destination in order to select the right key. This causes serious overhead and might even become infeasible for multicast/broadcast messages. Special new TLVs that contain different resident times or partial ICV calculations are possible solutions to this problem.

Although several approaches exist in the IT sector, e.g., IPsec or NTP authentication, to address the above issues, these approaches have to be carefully adapted. The three major problem fields identified so far are scalability, delay attacks, and the compatibility with hardware timestamping needed to reach high accuracy and precision.

HARDWARE TIMESTAMPING

Finally, additional network delay caused by overload of a switch or node can be tackled by the introduction of layer 2 on-the-fly timestamping, thus cancelling the residence time and load dependency **[9]**. Such methods use hardware support for replacing the timestamp field of any event message with the actual sending time of the currently transmitted message. Consequently, it is not necessary to send a *follow-up* message which is considered as an one-step-clock in the upcoming PTP standard v 2. One of the main problems implied by this method is that the modifications to the message require recalculation of not only checksums but also ICVs in security TLVs.

For these modifications, two issues have to be considered: a) data to be replaced and b) location of data. Whereas calculation of checksums (e.g. CRC, parity) and replacement of data can be done on the fly at the MII Interface, security functions require considerably longer times. E.g., HMAC-SHA-1

implementations can cope with the MII rate of 12.5 Mbyte/s, whereas RSA will not be able to keep pace. The second issue is the location of the data in the transmitted packet. While the timestamp is normally at the end of the packet, there are some checksums, e.g. UDP, which are located in the header of the packet and others at the end, e.g. Ethernet CRC. Other data are not even at a fixed position in the packet, such as the security TLVs.

For security, the data to be protected must be considered. E.g., the actual UDP header is not of primary interest; it is rather the source address that matters. Hence, inclusion of the UDP header can be avoided if the source address is already *a priori* included in the ICV calculation. Since the new hardware-drawn timestamp is known from the first sent bit on, at least 344 bit-times due to the PTP header size of 34 bytes plus 9 bytes for the TLV specification are saved for security calculation if the checksum for this part is already pre-calculated. This will give a reasonable time window for calculating the ICV for the timestamp and other changed information on slow systems or systems with slow cryptographic functions respectively. This gain can additionally be increased if the order of data for the ICV calculation — not for transmission — is modified. Currently, this measure requires dedicated security policies to be indicated, since a mixed order will result in failure of the ICV comparison and, therefore, in dropped packets.

A special focus of our ongoing research is the development of schemes that allow for on-the-fly protection in combination with hardware timestamping, which requires that security measures will only introduce very small constant delay and no additional jitter.

MEASURES BEYOND CRYPTOGRAPHY

Another severe issue is hostile introduction of varying delays. This method allows the attacker to prevent slaves from reaching a certain accuracy boundary. The only protection against this kind of attack is QoS monitoring, possibly combined with building a trusted chain of PTP ports. While the first can be done using client-based statistics like calculating an accuracy confidential interval or the well-known Allan deviation **[10]**, the second requires additional protocol support.

The principle involved in detecting malicious introduction of delays is to measure the QoS parameters over a sufficiently long period of time. Consequently, if the node knows the allowed parameter limits, any interference can be detected if it exceeds the expected statistical range. This method can be further supported by a trusted chain of PTP ports which requires authentication between each node on a per-port basis. Thus, the only remaining way to manipulate the delay between ports would be the introduction of varying delays on the physical layer. Here, the knowledge of the typical statistical parameters allows the identification of hostile delays. Due to the fact that the variability of the line parameters normally is below the desired accuracy, the method allows reliable attack detection. Additionally, all methods which support the detection of physical link manipulation, e.g. heartbeat and line power, can be used to further increase the security. QoS monitoring can also be used to detect malicious master overtake due to changes in the quality of the received synchronization information.

Finally, additional network delay caused by overload of the switch can be tackled by the introduction of layer 2 on-the-fly timestamping, thus cancelling the residence time on the switch and load dependency **[9]**. In combination with higher level QoS, which even does packet prioritization in favor of time synchronization, the synchronization interval cannot be influenced.

OUTLOOK

The security measures considered so far offer a good basic protection of the transmitted messages. Yet considering attacks in a more complicated network, including switches and nodes under the control of different entities, will require further work. Also, measures beyond cryptography have to be introduced to protect against maliciously introduced delays or packet deletion.

Open issues affecting the overall system performance and security heavily are:

• Key management — It will include the selection of key hierarchies and key distribution schemes to efficiently distribute keys. These procedures are most likely variants of already established protocols adapted to the needs of clock synchronization. Yet the development of new protocols for extreme application requirements cannot be excluded, especially if overheads should be reduced. Integration in special security signalling TLVs might be necessary.

• Source authentication and in-transit modification — IEEE 1588 version 2 will offer a group authentication only allowing one to see that a message is coming from a member of a group. In the future, a direct identification of the source and all entities amending or manipulating a packet during transition would increase the security level.

• Security measures for hardware timestamping — High precision synchronization requires hardware units to avoid jitter within the protocol stack. Security measures introduce additional jitter. Concepts for hardware timestamping have to be accounted for, and measures to calculate ICVs on the fly, similar to the way this is done now for CRCs, have to be introduced.

• Maliciously caused delays — These delays are a major topic for measurement applications. The impact of malicious loop parameter manipulation on the control algorithms has not yet been studied. Organizational measures have to be taken to protect the system.

• Security policies — One very controversial issue is the combination of secure and insecure nodes in a system. It is clear that a system only consisting of highly protected entities is most secure. Yet such systems are only working in very strict boundary conditions to maintain the security. For practical use, such as for migration from secure to insecure networks or for economical reasons, mixed networks will be necessary. Security policies handling security-unaware nodes or transparent clocks or allowing for limited reduction of services need to be developed and are investigated by the authors.

Research going beyond IEEE 1588 will require mapping the result gained from Master-Slave to peer-to-peer networking and democratic clock synchronization.

REFERENCES

[1] IEEE TC 9 Test and Measurement Society 2002, *"1588 IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems,"* IEEE Standard (IEEE, New York).

[2] D. L. Mills, 1991, *"Internet time synchronization: the network time protocol,"* **IEEE Transactions on Communications, 39,** 1482–1493.

[3] M. Popp, J. Feld, and R. Büsgen, 2005, *"Principles and Features of PROFInet,"* **The Industrial Communication Technology Handbook** (CRC Press/Taylor & Francis), chapter 11, p.11.1.

[4] H. Kopertz, G. Bauer, and W. 2005, *"Dependable Time-Triggered Communication,"* **The Industrial Communication Technology Handbook** (CRC Press, Taylor & Francis), chapter 12, pp. 12.1.

[5] G. Gaderer, T. Sauter, and G. Bumiller, 2005, *"Clock Synchronization in Powerline Networks,"* in Proceedings of the 2005 IEEE International Symposium on Power Line Communications and its Applications, 6-8 April 2005, Vancouver, Canada (IEEE), pp. 71-75

[6] National Institute of Standards and Technology, *"Secure Hash Signature Standard (SHS) (FIPS PUB 180-2),"* available at *http://csrc.nist.gov/publications/fips/\\fips180-2/fips180-2.pdf* or *http://csrc.nist.gov/publications/fips/fips180-2\\/fips180-2withchangenotice.pdf.*

[7] National Institute of Standards and Technology, *"The Keyed-Hash Message Authentication Code (HMAC) (FIPS PUB 180-2),"* available at *http://csrc.nist.gov/\\ publications/fips/fips198/fips-198a.pdf.*

[8] A. Treytl and T. Sauter 2005, *"Security Concept for a Wide-Area Low-Bandwidth Power-Line Communication System,"* in Proceedings of the 2005 International Symposium on Power Line Communications and Its Applications, 6-8 April 2005, Vancouver, Canada (IEEE), pp. 66-70

[9] R. Höller, T. Sauter, and N. Kerö, 2003, *"Embedded SynUTC and IEEE 1588 clock synchronization for industrial Ethernet,"* in Proceedings of the 9th IEEE Conference on Emerging Technologies and Factory Automation (ETFA '03), 16-19 September 2003, Lisbon, Portugal (IEEE), pp. 422-426.

[10] D. B. Sullivan, D. W. Allan, D. A. Howe, and F. L. Walls, 1990, **Characterization of clocks and Oscillators,** NIST Technical Note 1337 (National Institute of Standards and Technology, Boulder, Colorado).