Modeling and Security Analysis of IEEE 802.1AS Using Hierarchical Colored Petri Nets

Siyu Tang*, Xiaoya Hu*, Lian Zhao†

*School of Artificial Intelligence and Automation, Huazhong University of Science and Technology, Wuhan 430074, China
†Department of Electrical, Computer and Biomedical Engineering, Ryerson University, Toronto, ON M5B 2K3, Canada
Email: tangsiyu@hust.edu.cn, huxy@hust.edu.cn, l5zhao@ryerson.ca

GLOBECOM 2020 - 2020 IEEE Global Communications Conference | 978-1-7281-8298-8/20/$31.00 ©2020 IEEE | DOI: 10.1109/GLOBECOM42002.2020.9347988

***Abstract*—In recent decades, much attention has been paid to timely and guaranteed delivery in industrial automation networks. Toward this aim, the IEEE 802.1 Time-Sensitive Networking (TSN) task group has developed a series of standards. IEEE 802.1AS Timing and Synchronization protocol is the basis for TSN flow control mechanisms. As a rather new protocol, modeling and security analysis is a highly attractive candidate for developing IEEE 802.1AS. In this paper, we model the IEEE 802.1AS using Hierarchical Colored Petri Nets (HCPNs) and verify the proposed model by state space analysis and synchronization performance analysis. On the basis of our model, the security of the protocol is analyzed, including attack and defense against IEEE 802.1AS. Simulation results verify the validity and practicability of the model.**

***Index Terms*—Time-Sensitive Networking, IEEE 802.1AS, Hierarchical Colored Petri Nets, vulnerability analysis, timing and synchronization.**

I. Introduction

In industrial automation networks, time-critical messages are often used to control physical processes. Hence, it is significant to guarantee timely transmission of time-critical traffic flows [1]–[3]. To cope with this issue, there have been various network solutions, such as PROFINET, EthernetCAT, and T-TEthernet. However, these solutions are incompatible with each other, which results in complexity of network architectures. In this context, the IEEE 802.1 Time-Sensitive Networking (TSN) task group has developed a wide range of standards to enhance network standardization for offering deterministic services, particularly the Ethernet-based networks. According to the support that TSN standards provide, they can be divided into timing and synchronization, bounded low latency, reliability, and resource management [4], [5].

Timing and synchronization plays a core role for all TSN standards. The IEEE 802.1AS Timing and Synchronization protocol [6] which specifies a second-level profile of the IEEE 1588 Precision Time Protocol (PTP) [7] is developed to realize high synchronization accuracy over a network in the field of industrial automation control systems. It has been proved that IEEE 802.1AS can achieve a synchronization precision better than 1μs [8], [9]. However, IEEE 802.1AS is designed without an inherent security mechanism, which will be exposed to novel attacks [10], [11]. As a consequence, it is imperative to model IEEE 802.1AS to analyze its security performance characteristics. In this paper, we aim to study modeling and security analysis of IEEE 802.1AS by using Hierarchical Colored Petri Nets (HCPNs).

This work was supported by the National Key R&D Program of China under Grants 2020YFB1708601; Xiaoya Hu is the corresponding author.

Colored Petri Nets (CPNs) are graphical tools for constructing models of a wide range of applications and analyzing their properties, which combine Petri Nets (PNs) with a high-level program language [12]. The representative applications of CPNs involve communication protocols, embedded systems, distributed algorithms and data networks [13]. Whereas, setting up a CPN model for a complex object is impractical since it would be very large and inconvenient [12]. Hence, we use the HCPNs to model IEEE 802.1AS, by which a network can be organized as a set of modules. The modules in a HCPN can be treated as black boxes, which let us focus on only a few critical details at a time.

As a rather new protocol, very limited research has been done on the modeling and security analysis of IEEE 802.1AS. Several research works focused on PTP simulation and synchronization precision analysis [9], [14]–[16]. These studies built simulation frameworks on OMNeT++ or OPNET simulator to test protocol performances. However, these frameworks were not suitable to analyze the security of PTP. Many studies used CPNs for security analysis, which proved that CPNs are effective [13], [17]. Some related works were made on PTP [18], [19]. In [18], the author used Stochastic Petri Net (SPN) to model PTP, and changed model parameters to analyze the vulnerability of PTP. Although SPN has been widely used in the communication field, a very large model size of a SPN may cause state space to increase exponentially with the increase of system scale. The work [19] used timed CPNs to model PTP and performed the synchronization ability analysis of PTP. In view of these, we propose a novel modeling method for IEEE 802.1AS and analyze its security in this paper.

The remainder of this paper is organized as follows: In Section II, we propose HCPNs model of IEEE 802.1AS. Section III verifies the validity of the protocol model and analyzes the protocol security based on the model. Finally, we conclude our work in Section IV.

978-1-7281-8298-8/20/$31.00 ©2020 IEEE

II. HCPNs Model of IEEE 802.1AS

In this section, the overview of IEEE 802.1AS mechanisms and HCPNs is given at first, after that we describe the modeling process and each module in detail to show our HCPNs model of IEEE 802.1AS.

A. Overview of IEEE 802.1AS Mechanisms and HCPNs

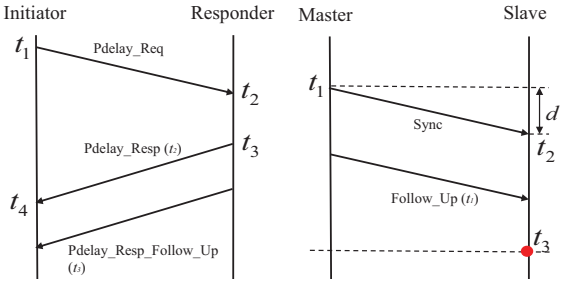
There are two stages in IEEE 802.1AS to realize time synchronization, one is propagation delay measurement and the other is transport of time-synchronization information. The process of propagation delay measurement is illustrated in Fig. 1(a). As shown in Fig. 1(a), the measurement contains four steps:

1) At $t = t_1$, the initiator starts with issuing a Pdelay_Req message.

2) At $t = t_2$, the responder receives the Pdelay_Req message.

3) At $t = t_3$, the responder returns a Pdelay_Resp message, which includes the time t_2.

4) At $t = t_4$, the initiator receives the Pdelay_Resp message.



(a) Delay measurement. (b) Time synchronization.

Fig. 1. IEEE 802.1AS mechanism.

At last, the responder returns a Pdelay_Resp_Follow_Up message with the time t_3 to the initiator. Then the initiator uses these four timestamps to compute the mean propagation delay d as:

$$d = \frac{(t_4 - t_1) - (t_3 - t_2)}{2} \tag{1}$$

The transport of time-synchronization information using Sync and Follow_Up message is illustrated in Fig. 1(b), it contains two steps:

1) At $t = t_1$, the grandmaster transports a Sync message.

2) At $t = t_2$, the slave receives the Sync message.

And then the grandmaster sends a Follow_Up message to inform the slave the time t_1. Upon knowing the two timestamps, the slave synchronizes its time as follows:

$$T_3 = t_3 - t_2 + d + t_1 \tag{2}$$

where T_3 is the synchronized time of the previous time t_3.

A HCPN model is often created as a graphical drawing containing *places*, *transitions* and directed *arcs* connecting places and transitions. Furthermore, there are *substitution transitions* drawn as rectangular boxes with double-line borders in HCPNs, which representing the submodule of the hierarchical protocol model. Each place can be marked with one or more *tokens*, and each of them has a data value attached to it, which is called the *token color*. The set of possible token colors is specified by means of a type, as known from programming languages, and it is called the *color set* of the place.

B. Top-level Module of Protocol Model

Referring to the mechanisms of IEEE 802.1AS, the main color sets and variable declarations defined in our HCPNs model can be seen in Table I. After defining the color sets and variables, we create the HCPNs model in Fig. 2, which is the top-level module of protocol model.

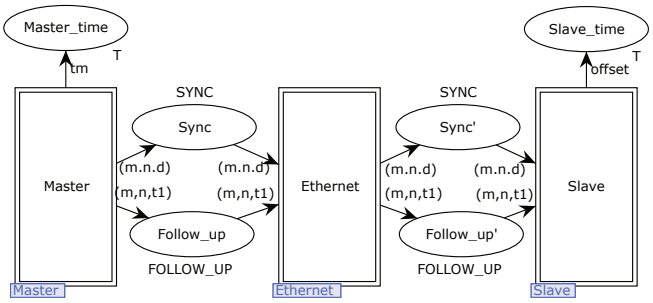


Fig. 2. Top-level module of protocol model.

As is depicted in Fig. 2, there are three substitution transitions, which present the submodule of master, Ethernet and slave respectively. The master clock sends a Sync message to the Ethernet and then it is received by the slave, with that a Follow_Up message is also sent to slave to synchronize time. As we can see in Fig. 2, the top-level module shows a more abstract view of the time synchronization process between master and slave clock without more detailed information of the behavior represented by the substitution transitions, and it is clear and straightforward to concentrate on the model architecture.

C. Master Module of Protocol Model

As shown in Fig. 3, the master module contains a number of places and transitions to describe the sending of Sync message and Follow_Up message. In Fig. 3, the output port *Master_time* presents the local time of the master clock which is the reference time for the slave clock, while *Master_Init* is the internal place including the color token of initial time value. Moreover, the module formed by the place *Master_time* and the transition $T1$ is used to model the variation of the time, referred to as a global clock of the model. As for the transitions, the *Frame Generation* indicates the occurrence of messages. When the token in each input place satisfies the arc expressions on the input arcs, it is possible for a transition to be enabled, which stands for the Sync and Follow_Up message sent by the master clock.

D. Ethernet Module of Protocol Model

The Ethernet module used for modeling the communication channel between the master and slave clocks is shown in Fig. 4. The Ethernet module has two input ports, *Sync* and *Follow_up*,

TABLE I
THE MAIN COLOR SETS DEFINED IN THE PROTOCOL MODEL

Data Type	Color Set	Variable
Source and destination address	colset MAC=int timed	var *m*,*n*: MAC
Message data	colset DATA=string timed	var *d*:DATA
Timestamps	colset T=int timed	var $t1$,$t2$,$t3$,$t4$,tm,ts,ti,tr: T
Sync message	colset SYNC=product MAC*MAC*DATA timed	var *sync*: SYNC
Follow_Up message	colset FOLLOW=product MAC*MAC*T timed	var *follow*: FOLLOW
Pdelay_Req message	colset REQ=product MAC*MAC*DATA timed	var *req*: REQ
Pdelay_Resp message	colset RESP=product MAC*MAC*T timed	var *resp*: RESP
Propagation delay	colset DELAY=int timed	var *delay*: DELAY

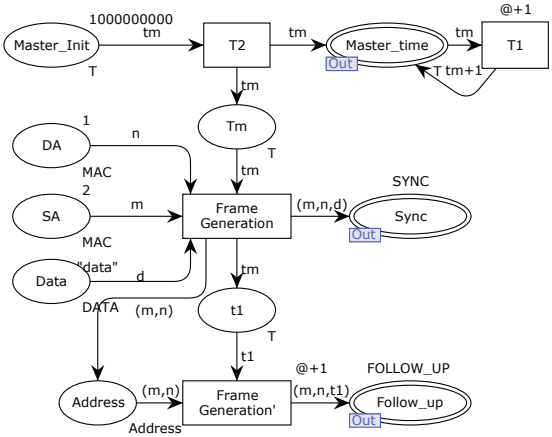


Fig. 3. Master module of protocol model.

together with two output ports, *Sync*′ and *Follow_up*′. The transition *Channel*_1 and *Channel*_2 signify the transport of messages from master to slave. Furthermore, we have associated a time delay expression @+Pdelay() with the transitions, denoting that there is a delay in the propagation process. The function Pdelay is defined as follow:

$$\text{fun Pdelay()=discrete}(\alpha,\beta);$$

where the value of α and β present the thresholds of delay, and the predefined function discrete provides discrete uniform distribution over the closed interval, which means that Pdelay() returns an integer from the interval $[\alpha,\beta]$ and that all numbers in the interval have the same probability of being chosen.

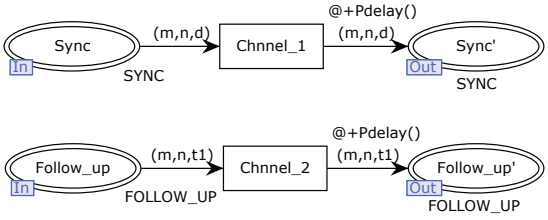


Fig. 4. Ethernet module of protocol model.

E. Slave Module of Protocol Model

The slave module of our model is illustrated in Fig. 5. Corresponding to the master module, the slave module describes the receiving of Sync message and Follow_Up message. As shown in Fig. 5, the slave module contains two input ports, *Sync*′ and *Follow_up*′, and the next transitions denote that the slave is receiving the message from the Ethernet. Moreover, it is worth mentioning that the substitution transition, defined as *Propagation Delay Measurement*, presents the measurement process of propagation delay, which consists of initiator and responder as shown in Fig. 6 and Fig. 7.

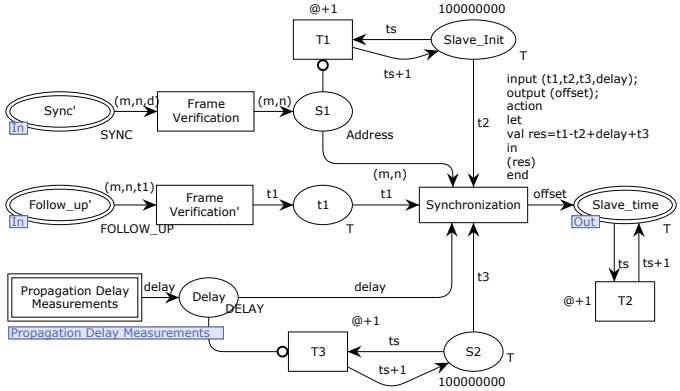


Fig. 5. Slave module of protocol model.

In Fig. 6, the initiator module has one output port, called *Pdelay_Req*, and two input ports, *Pdelay_Resp* and *Pdelay_Resp_Follow_Up*, representing the message sending and receiving respectively. At the same time, the responder module shown in Fig. 7 has one input ports and two output ports corresponding to the initiator module.

Once the substitution transition *Propagation Delay Measurement* occurs, there will be a color token containing the time value of delay taken into the place *Delay*. Upon the slave obtains the delay time value and the two timestamps, t_1 and t_2, the transition, *Synchronization*, will be enabled to occur, indicating that the slave adjusts its local time according to the expression in (2).

III. MODEL VERIFICATION AND SECURITY ANALYSIS

In this section, we firstly verify our proposed model by state space analysis and synchronization performance analysis. Then, we investigate the attack and defense against IEEE 802.1AS based on the model.

In order to analyze protocol performance, a time reference should be applied to the HCPNs model. Some studies [9] show

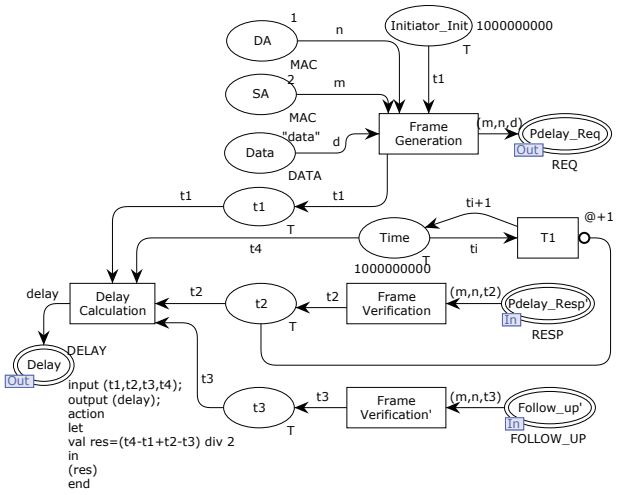


Fig. 6. Initiator module of *Propagation Delay Measurement*.

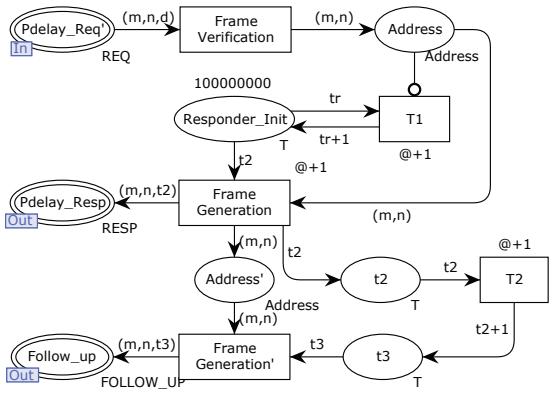


Fig. 7. Responder module of *Propagation Delay Measurement*.

that the time measurement of IEEE 802.1AS is in nanoseconds. Therefore, we define the time unit of our HCPNs model to be a nanosecond and set the initial value of time offset to be 900 milliseconds between the master and slave clock.

A. Model Verification

For verifying the validity of our model, the first step is to ask for a state space report to check the consistency between behavioural properties of model and protocol. The state space report contains information about a number of key properties of the model, which can be produced completely automatically by state space tool. Table II presents the state space analysis results.

In Table II, the nodes and arcs in the Strongly Connected Component (SCC) graph are the same as that in the state space, indicating that all state nodes in the HCPNs model are reachable and there are no cycles in the state space of the model. This means that the occurrence sequences must be finite and the protocol will necessarily terminate. Meanwhile, the home markings, dead transitions and live transitions are all 0, stating that there are no marking which can be reached from any state and no transition which can never or always

TABLE II
STATE SPACE REPORT RESULTS

Property	Value
State space nodes	30567
State space arcs	97403
SCC graph nodes	30567
SCC graph arcs	97403
Home markings	0
Dead markings	1
Dead transitions	0
Live transitions	0

be activated. There is one dead marking in our model, and it indicates that synchronization process has been successfully executed by slave. Through state space analysis, it can be seen that all the behavioural properties of our model match expected behavior of the protocol.

To further verify the validity of our proposed model, we analyze synchronization performance by model simulation and compare it with theoretical results. In our study, we use the parameter "precision" to evaluate synchronization performance, which is defined as the maximum difference between the local clocks in the network.

Here, we analyze the synchronization precision in the contect of both symmetric and asymmetric path to verify our HCPNs model [20]. The reference [9] indicates that the value of propagation delay ranges from 52 ns to 72 ns and the average is 62 ns in a typical scenario, thereby we set the propagation delay to the constant value of 62 ns in symmetry, which means that the value of α and β are both 62, and the propagation delay ranges from 52 ns to 72 ns in asymmetry, which means that the value of α is 52 and the value of β is 72.

For symmetric path, the simulation results are shown in Table III. From it, we can see that the slave clock synchronizes to the master clock completely without any error, which is consistent with the theoretical analysis.

TABLE III
SIMULATION RESULTS WITH PATH SYMMETRIC

	Initial time value	Final time value
Master clock	1000000000	1000000364
Slave clock	100000000	1000000364
Offset between master and slave	900000000	0

For asymmetric path, there will be an error according to the protocol mechanism. In Fig. 1(a), the path delay from initiator to responder is denoted as d_1, d_2 denotes the opposite path delay, and d_3 is the real delay in the transport of time-synchronization information. The theoretical synchronization precision P is calculated as follow:

$$P = T_3' - T_3 = d_3 - \frac{1}{2}(d_1 + d_2) \tag{3}$$

where T_3 is the measurement value while T_3' is the real and correct value. According to Equation (3), the theoretical synchronization precision ranges from 0 ns to 20 ns and the average is 10 ns in our model. Then we run the HCPNs model for 200 times, the simulation results are shown in Fig. 8, where

the black line is the evolution of the measured synchronization precision, and the red line is the cumulative average of the synchronization precision. From Fig. 8 we can see that the simulation results match the theoretical outcome, which verifies the validity of the proposed model.

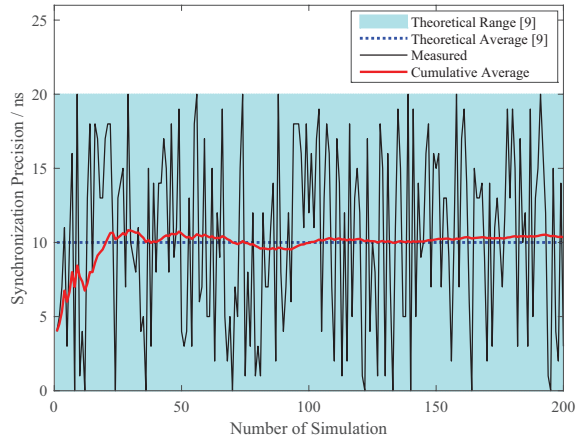


Fig. 8. Simulation results with path asymmetric.

B. Security Analysis of IEEE 802.1AS

To make security analysis of IEEE 802.1AS in our study, we firstly consider the threats against the protocol, and then analyze the protection methods against it. We investigate the spoofing threat on IEEE 802.1AS and perform the attack simulation on our model. We add an attacker node to our model, as shown in Fig. 9. In the spoofing threat, on the one hand, the attacker spoofs the master, on the other hand, it spoofs the slave, like a Man-in-the-Middle (MITM) attacker. The simulation results are shown in Table IV, showing that the time value of the slave is synchronized to that of the attacker, which means the spoofing threat accomplished.

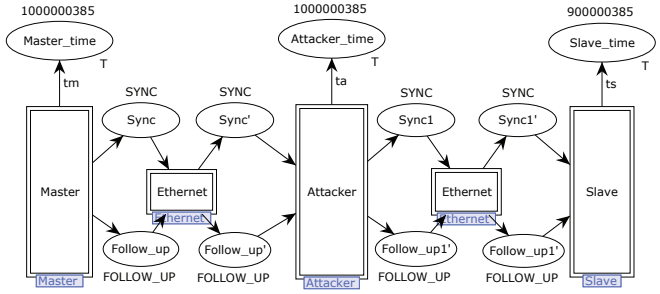


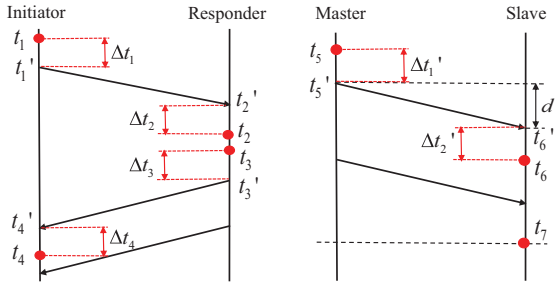
Fig. 9. HCPNs model with an attacker node.

As for how to secure the time synchronization protocol, some research has indicated that MACsec is an efficient way in native L2 Ethernet network [21]. Although the MACsec mechanism can be implemented in hardware, the encryption and decryption hop-by-hop will degrade the performance of the protocol, since it will cause additional processing delay and jitter. Thereby, it is crucial to analyze the jitter tolerance caused

TABLE IV
SIMULATION RESULTS WITH AN ATTACKER

	Initial time value	Final time value
Master clock	1000000000	1000000385
Attacker clock	900000000	1000000385
Slave clock	100000000	900000385
Offset between master and slave	900000000	100000000

by the MACsec mechanism. In our study, the jitter tolerance is denoted as J. In order to investigate the J, we consider the processing time of the MACsec mechanism and add this value to our model. Before the simulation, we analyze the theoretical J, as shown in Fig. 10. All the explanations of symbols in Fig. 10 can be found in Table V.



(a) Delay measurement. (b) Time synchronization.

Fig. 10. IEEE 802.1AS with MACsec mechanism.

Then the measurement value of propagation delay, d, is calculated as follows:

$$d = d' + \frac{\Delta t_1 + \Delta t_2 + \Delta t_3 + \Delta t_4}{2} \tag{4}$$

while d' is the actual value of propagation delay.

Considering the deviation between the delay value in the propagation delay measurement and that in the transport of time-synchronization information, we denote the deviation as d_{dev}, then the slave synchronizes its time as shown in Equation (5):

$$T_7 = t_7 - t_6' + d + t_5' - \left(\Delta t_1' + \Delta t_2'\right) + d_{dev} \tag{5}$$

where T_7 is the synchronized time with an error. Considering the Equation (4), we can obtain:

$$T_7 = T_7' + \frac{\Delta t_1 + \Delta t_2 + \Delta t_3 + \Delta t_4}{2} - \left(\Delta t_1' + \Delta t_2'\right) + d_{dev} \tag{6}$$

whereas the correct value T_7' is:

$$T_7' = t_7 - t_6' + t_5' + d' \tag{7}$$

Comparing the Equations (6) and (7), we can calculate the synchronization precision P as follows:

$$P = \frac{\Delta t_1 + \Delta t_2 + \Delta t_3 + \Delta t_4}{2} - \left(\Delta t_1' + \Delta t_2'\right) + d_{dev} \tag{8}$$

Considering the threshold of P as μ, we define the theoretical J as follows:

$$J = \frac{1}{2}\left(\mu - d_{dev}\right) \tag{9}$$

TABLE V
NOTATIONS

Symbol	Definition
t_1	Measured sending timestamps of Pdelay_Req message
t_1'	Actual sending timestamps of Pdelay_Req message
t_2	Measured receiving timestamps of Pdelay_Req message
t_2'	Actual receiving timestamps of Pdelay_Req message
t_3	Measured sending timestamps of Pdelay_Resp message
t_3'	Actual sending timestamps of Pdelay_Resp message
t_4	Measured receiving timestamps of PdelayvResp message
t_4'	Actual receiving timestamps of Pdelay_Resp message
Δt_1	Encryption time of Pdelay_Req message
Δt_2	Decryption time of Pdelay_Req message
Δt_3	Encryption time of Pdelay_Resp message
Δt_4	Decryption time of Pdelay_Resp message
t_5	Measured sending timestamps of Sync message
t_5'	Actual sending timestamps of Sync message
t_6	Measured receiving timestamps of Sync message
t_6'	Actual receiving timestamps of Sync message
$\Delta t_1'$	Encryption time of Sync message
$\Delta t_2'$	Decryption time of Sync message
t_7	Time of slave for correction

After the theoretical analysis, we carry out the simulation based on the protocol model with MACsec. We consider the evolution of the synchronization precision with J ranging from 0 ns to 500 ns. To compare the simulation results with the theoretical outcome, we use the least square method to fit the data. The simulation results are shown in Table VI. It is not hard to see that the simulation results match the theoretical analysis.

TABLE VI
SIMULATION RESULTS WITH MACSEC

Best fitting line	Slope	Intercept
Theoretical value [9]	2.0000	10.0000
Measured value	1.9997	9.0690

IV. CONCLUSION AND FUTURE WORKS

In this paper, we propose a Hierarchical Colored Petri Nets model for IEEE 802.1AS and a spoofing threat on IEEE 802.1AS has been performed based on it. Meanwhile, the MACsec is also investigated based on the model. Results indicate that the HCPN is an efficient way for modeling of IEEE 802.1AS, and it is really suitable to analyze the security of the protocol. The IEEE 802.1AS may present some security vulnerabilities and threats since it is designed without an inherent security mechanism. Therefore, the MACsec mechanism will be an alternative way to secure IEEE 802.1AS traffic flow from our analysis results. Future works aim at improving the HCPNs model and analyzing more threats on the IEEE 802.1AS based on the model.

REFERENCES

[1] L. Lo Bello and W. Steiner, "A Perspective on IEEE Time-Sensitive Networking for Industrial Communication and Automation Systems," *Proceedings of the IEEE*, vol. 107, no. 6, pp. 1094–1120, June 2019.

[2] A. Nasrallah, A. S. Thyagaturu, Z. Alharbi, C. Wang, X. Shao, M. Reisslein, and H. ElBakoury, "Ultra-Low Latency (ULL) Networks: The IEEE TSN and IETF DetNet Standards and Related 5G ULL Research," *IEEE Communications Surveys Tutorials*, vol. 21, no. 1, pp. 88–145, Firstquarter 2019.

[3] X. Jiang, H. Shokri-Ghadikolaei, G. Fodor, E. Modiano, Z. Pang, M. Zorzi, and C. Fischione, "Low-Latency Networking: Where Latency Lurks and How to Tame It," *Proceedings of the IEEE*, vol. 107, no. 2, pp. 280–306, Feb 2019.

[4] N. Finn, "Introduction to Time-Sensitive Networking," *IEEE Communications Standards Magazine*, vol. 2, no. 2, pp. 22–28, June 2018.

[5] J. L. Messenger, "Time-Sensitive Networking: An Introduction," *IEEE Communications Standards Magazine*, vol. 2, no. 2, pp. 29–33, June 2018.

[6] "IEEE Standard for Local and Metropolitan Area Networks - Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks," *IEEE Std 802.1AS-2011*, pp. 1–292, March 2011.

[7] "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. 1–300, July 2008.

[8] K. B. Stanton, "Distributing deterministic, accurate time for tightly coordinated network and software applications: IEEE 802.1 AS, the TSN profile of PTP," *IEEE Communications Standards Magazine*, vol. 2, no. 2, pp. 34–40, 2018.

[9] M. Gutirrez, W. Steiner, R. Dobrin, and S. Punnekkat, "Synchronization Quality of IEEE 802.1AS in Large-Scale Industrial Automation Networks," in *2017 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, April 2017, pp. 273–282.

[10] N. Moreira, J. Lzaro, J. Jimenez, M. Idirin, and A. Astarloa, "Security mechanisms to protect IEEE 1588 synchronization: State of the art and trends," in *2015 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*, Oct 2015, pp. 115–120.

[11] T. Mizrahi, "Time synchronization security using IPsec and MACsec," in *2011 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control and Communication*, Sep 2011, pp. 38–43.

[12] K. Jensen and L. M. Kristensen, *Coloured Petri nets: modelling and validation of concurrent systems*. Springer Science & Business Media, 2009.

[13] K. Jensen and G. Rozenberg, *High-level Petri nets: theory and application*. Springer Science & Business Media, 2012.

[14] H.-T. Lim, D. Herrscher, L. Völker, and M. J. Waltl, "IEEE 802.1 AS time synchronization in a switched Ethernet based in-car network," in *2011 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2011, pp. 147–154.

[15] M. Pahlevan, B. Balakrishna, and R. Obermaisser, "Simulation Framework for Clock Synchronization in Time Sensitive Networking," in *2019 IEEE 22nd International Symposium on Real-Time Distributed Computing (ISORC)*, May 2019, pp. 213–220.

[16] W. Wallner, A. Wasicek, and R. Grosu, "A simulation framework for IEEE 1588," in *2016 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*, Sep 2016, pp. 1–6.

[17] Y. Xu and X. Xie, "Modeling and Analysis of Security Protocols Using Colored Petri Nets," *Journal of Computers*, vol. 6, no. 1, pp. 19–27, 2011.

[18] S. Jiajun and F. Dongqin, "Vulnerability Analysis of Clock Synchronization Protocol Using Stochastic Petri Net," in *2014 IEEE Intl Conf on High Performance Computing and Communications, 2014 IEEE 6th Intl Symp on Cyberspace Safety and Security, 2014 IEEE 11th Intl Conf on Embedded Software and Syst (HPCC,CSS,ICESS)*, Aug 2014, pp. 615–620.

[19] R. R. Igorevich and P. Park, "A Timed Colored Petri-Net modeling for precision time protocol," in *2016 International Conference on Information Science and Communications Technologies (ICISCT)*, Nov 2016, pp. 1–5.

[20] R. Exel, T. Bigler, and T. Sauter, "Asymmetry Mitigation in IEEE 802.3 Ethernet for High-Accuracy Clock Synchronization," *IEEE Transactions on Instrumentation and Measurement*, vol. 63, no. 3, pp. 729–736, March 2014.

[21] E. Itkin and A. Wool, "A security analysis and revised security extension for the precision time protocol," in *2016 IEEE International Symposium on Precision Clock Synchronization for Measurement, Control, and Communication (ISPCS)*, Sep 2016, pp. 1–6.