Vulnerability Analysis of Time Synchronization in Automotive Ethernet

Rishikesh Kakade∗1, Joey Chou∗2, Shannon Torcato∗3
∗Department of Electrical and Computer Engineering, University of Waterloo
{∗rkakade, ∗t6chou, ∗storcato}@uwaterloo.ca

Abstract—The operation of many network communication protocols require accurate time synchronization between nodes. In the automotive space, IEEE 802.3bw (commonly referred to as automotive ethernet) is quickly becoming the most popular in-vehicle communication protocol between electronic control units (ECUs). The rapid advance of autonomous vehicles is predicated on a high throughput of multiple HD video streams, LiDAR readings, ML inferences, and vehicular control signals propagating on the same bus. To ensure reliability, security, and safety, for millions of passengers and bystanders, it is essential that the stack is robust at even the lowest level. The time synchronization protocol in this standard is IEEE 802.1 AS which was initially developed for virtual and virtual bridge local area networks (LANs). This paper presents an analysis of failure modes affecting the ability of in-vehicle automotive ethernet networks to synchronize clocks between ECUs. To demonstrate this, we use the OMNet++ discrete event simulator to showcase normal operation, failover to the hot-standby clock, and a black hole attack on one network switch.

I. INTRODUCTION

In the early 1980s, automotive manufacturers began including electronic control units inside their cars. These tiny devices were the first step away from manual, mechanical control of cars by users, toward increased automation. In 1983, Bosch introduced the Controller Area Network protocol that arbitrated communication between ECUs. The protocol synchronizes between nodes using a synchronization byte that is transmitted at the start of each data packet transmitted. In the present day, this progression has led to an average car containing over 100 discrete ECUs, each responsible for controlling the millions of tiny electromechanical adjustments that make driving as effortless as it is today.

The next evolution of this rapid progress is the autonomous vehicle space. The amount of data a vehicle generates and processes has increased drastically due to an increase in the number of sensors present in the vehicle, some of which are LiDAR, cameras, and radar. These sensors are used to scan the surroundings of the vehicle to detect other vehicles and road conditions in real time. Due to this increase in data throughput required inside a vehicle, car manufacturers have reached the limits of the measly 2-wire, 500kbps, CAN protocol.

The leading candidate to succeed the CAN protocol is ethernet. Ethernet has data throughput rates of 100 Mbps — representing a 200x improvement in transmission rate. It is a natural choice because the concept has been well tested and proven as the interface that connects the the internet. The large body of knowledge already available enables the industry to quickly adopt it and integrate into existing systems.

A foundational, yet often overlooked component of the ethernet standard in this application is its method of synchronizing time across ECUs. It leverages Generic Precision Time Protocol (gPTP). The protocol synchronizes by comparing the difference in time between 2 master clocks. All other slave nodes synchronize to the master nodes by listening periodically for synchronization pulses transmitted from the master clock to many slaves clocks. As a redundancy measure, the protocol allows slaves to synchronize to more than one master clock. This acts as a fail safe in case one master clock loses its link to the rest of the network.

This method of synchronization is robust enough for most internet applications. However in the automotive space, every component in the car is subject to an incredible amount of scrutiny because of the enormous potential for loss of life. The time synchronization layer provides room for a number of failure modes that could cause critical loss of function and possibly, of life. Additional risk is introduced with the fact that modern vehicles are exposed to the outside world over the internet. From any layer of the stack, external adversaries could carry out an attack that exploits a vulnerability.

TSN which stands for Time-Sensitive Networking is a protocol defined by IEEE 802 standards that by default makes Ethernet deterministic. In particular, TSN addresses the transmission which are bounded by low latency and which has high availability. Some of the real-world applications are automotive vehicles or industrial control facilities that use real-time control streams.

Attacks could include jamming, routing infrastructure attacks, and byzantine attacks. With this large attack surface, it is especially important that designers keep robustness in mind. By simulating this time synchronization protocol we are able to perform experimentation at a lower cost. When it comes to a global time clock, there is no one standard time that is followed. Each node has its own internal clock that it uses to keep a track of events. Over time the clocks in the network diverge due to finite clock accuracy.

This paper will investigate a number of failure modes Key metrics of interest are global convergence time, clock offset, clock drift, and the maximum number of adversarial faults that can be tolerated.
In [9], the authors propose a new gateway framework based on the controller area network (CAN) for in-vehicle networks (IVNs). The proposed system is easy to reuse and verify this helps reduce the development costs and time. This system introduces new functionalities such as parallel programming, diagnostic routing, network management (NM), dynamic routing update, multiple routing configuration, and security.

In [12], the authors describe why Ethernet is more efficient than other legacy technologies such as CAN and MOST. Ethernet is preferred to its increased bandwidth. The sensors all around the vehicle help build new and exciting features for safety such as collision avoidance, lane departure detection, traffic sign classification, blind spot detection, driver intent detection, pedestrian detection, automatic cruise control, and many others as described in [7].

The authors of [14] compare different clock synchronization algorithms against traffic types with individual timing requirements. The authors were able to focus on strategies for scheduling flows and factors leading to delay. They were able to find that the bandwidth of the flow is reserved based demand of the actual bandwidth, this helps introduce a more stable transmission. When adopted alone SPQ ruler can lead to high-priority data with large flow bursts which would lead to low priority flow to have high delays.

In the paper [10], the authors present a novel approach in PTP systems to reduce the clock time drifting significantly called timing fault recovery (TFR). In this approach, the fault occurrence is detected by TFR and within a short duration, it is recovered using a handshake mechanism. With increased reliability and availability of PTP systems, the TFR approach also provides constancy and clock stability.

The authors of [13] proposed and designed an IEEE 1588 protocol for a clock synchronization system using FPGA. This system is very accurate and can detect time precision at the microsecond level in the synchronization of the master clock and slave clock. This design is an improvement to the current time synchronization precision, it also helped reduce the complex hardware circuit design, and helped reduce the development cost and difficulty, this provides a benchmark value to research done in the future.

In [8], the authors discuss the performance of an autonomous vehicle. The data measured from various sources are integrated and processed, depending on whether time information is available from the sensor measurement the data will vary. Temporal distortion of the controller in which missing patches of information are patched up at different instants so the situation can be perceived. On the other hand, time information-based sensor measurements are aware of the time differences among the values this allows the autonomous driving controller to have a better understanding of the situation by compensating for the time differences. The authors conclude that to increase autonomous vehicle performance, time synchronization-based research should be looked into thoroughly sensors and controllers connected via communication networks.

In [11], the authors created a simulation model of the IEEE 802.11AS generalized Precision Time Protocol (gPTP) standard on OMNET++ for TSN specific synchronization. The authors utilize measurements of propagation delay and time synchronization between systems that are time-aware. The model described in this papers utilizes clocks with constant drift and end stations. The results observed that every one second propagation delay is measured whereas it was every 125 ms or 62.5 ms for synchronization message sent from GM. As a result of this there is a difference in clock drifts. The time difference of all the nodes after synchronization to GM is zero as expected from the gPTP simulation model.

Black Hole Attack In this type of attack, the adversary stops propagating forward the information or data but is still an active node in the routing protocol. The adversarial node when part of a routing protocol path prevents further communication to take place on that part. This type of attack affects secure and insecure routing protocols due to their nature of rendering the normal methods of route maintenance useless.

Flood Rushing Attack In this attack, the flood duplicate suppression technique of the node is exploited. The adversarial flood tries to be the first to affect the nodes over the actual flood during propagation. When a node receives a flood packet from an adversary then the nodes will ignore the legitimate version and will forward propagate the adversarial version. Due to this, it is almost impossible for the node network to be free from adversarial propagation.

Byzantine Wormhole Attack In this kind of attack, the compromised nodes communicate with each other in order to gain an advantage. By communicating with each other they are able to create a shortcut in the network. By sending out a route request an adversary can discover a route in the network, to execute the attack, the tunnel packets through the non-adversarial nodes. In order to increase the probability of being selected as the route path, the adversaries use the low cost appearance of the wormhole links. They then drop all of the data packets to disrupt the network. By having as minimum as two compromised nodes this particular attack can be performed.

Byzantine Overlay Network Wormhole Attack This kind of attack is similar to the wormhole attack but in this case,
several nodes are compromised and they form a network on top of the existing network. The adversaries tunnel packets through the overlay network and make it seem like they are all neighbours to the routing protocol. This helps increase the chances of being selected on routes [1].

III. METHODOLOGY

To build our model we use an OMNet++ simulation with the INET 4.4 library [3].

We can combine the gPTP model with the INET library along with the other standard network protocols that are present in the library. For the simulation model, we do not consider the best master clock algorithm of the gPTP protocol. Therefore, we predefine the state of the node as master or slave. We are more interested in clock synchronization and measurement of propagation delay. We introduce a clock drift so that we can implement a time synchronization protocol.

In order to meet gPTP requirements, we implement time synchronization and propagation delay measurements using a class from the OMNeT++ library [4]. We use the in-built EthernetIIFrame available in the INET library which has encapsulated gPTP messages.

In INET, network nodes and interfaces do not have local time. Instead, the global time is defined as the simulation time. Network clock submodules are required by network nodes to simulate local time-based network nodes and to demonstrate effects such as clock drift. The oscillator submodules are used to keep a track of time. The basic working is that the oscillator periodically produces ticks which are counted by the clock modules. Inaccuracies in oscillators can lead to clock drift.

There can be drift in clocks of the time kept between different network devices. When two clocks are synchronized, the time difference between them has an upper bound. We can achieve this between clocks by periodically synchronizing them. In gPTP, there is a time synchronization of the slave clocks to the time of the master clocks within a time domain of the gPTP. Multiple time domains of gPTP can be obtained by a network. Multiple clocks can be synchronized to multiple master clocks in case there is a failure in the master clock or a link break causes the system to go offline. The general structure is that there exists a single master clock followed by a number of slave clocks. Synchronization by the protocol is achieved by sending sync messages to the slave clocks from the master clock. By having multiple gPTP submodules, a node can have multiple gPTP time domains. Depending on their location, a gPTP can follow one out of three possible roles: master, bridge, or slave. By having multiple gPTP submodules, a node can have multiple gPTP time domains. Depending on their location, a gPTP can follow one out of three possible roles: master, bridge, or slave. Master gPTP submodules of nodes contain the time domain master clock.

The fundamental device underlying the simulation the clock seen in Figure 2. It is modelled using basic clock classes from the INET library [3]. The basic models include the ability to represent clock drift. Clock drift arises due to inaccuracies in the low-cost crystal oscillators often used in automotive ECUs.

In the simulation, we have modelled the drift as a random walk, with steps varying by $\pm 100\mu s$ every second. In the real world, this variance is generally caused by process variations, thermal expansion, and device aging. The change in clock drift over time is ignored because it changes very slowly relative to the time frames on which automotive ethernet networks operate. For this short simulation, clock frequency is assumed to be constant.

We emulate 4 ECUs controlling the 4 DC motors of an electric car. The ECUs are synchronized to a central body controller and the main computer as a redundancy measure. In the simulation the network is set up with two master clocks; each disseminating two time domains. Each node in the network keeps track of 4 total times. The switching network is shaped as a ring to further emphasize the benefit of having multiple time domains propagating in different directions as a redundancy measure.

To summarize the way the network in Figure 4 works [3];

1) Primary clock has one clock, two master time domains.

The domains are propagated in both clockwise and
counterclockwise directions along the ring
2) Secondary clock has 4 time domains. Two are slaves to the primary clock’s domains while the others are redundant domains.
3) all other nodes in the network are slaves to the four total time domains propagating in the network

This paper presents three scenarios. The first is operation of the network under normal conditions. Under normal circumstances, the only disturbance that needs to be corrected for is clock drift as described in Figure 1. It shows that between successive synchronization pulses from the master nodes to each slave, the relative clock drift causes the delta between each local time to be non-constant. In the second scenario, we simulate a failure of the central body controller. This could be caused by a number of failure modes, such as power loss, bus message overloads, and firmware failures. We expect that the slaves in the network should be able to switch over to the redundant secondary master in the car’s central computer. In the third, we simulate a black hole attack on one of the network switches. Here, we make the switch unable to forward synchronization messages to the slave ECU it is connected to. It is expected that the device should be unable to synchronize with the rest of the network.

The relevant simulation files are made open source on Github [5].

IV. Results

A. Normal operation

Figure 5 shows the difference between the clocks from ‘true’ simulation time of the network under normal conditions. The clock in the front right wheel has the most dramatic clock drift which explains why peak to peak difference is higher than in others. Despite this, the network is still able to synchronize all components to within about ±100ns of one another. We include simulation of the other, redundant time domain disseminated by the main computer at this time. We find that all nodes in the network are able to synchronize to this as well, even though it has a different clock drift. When the end-to-end delay is constant there is no clock drift as the generated packets are in sync with the source.

B. Failure of primary clock

Figure 7 shows the difference between the clocks from ‘true’ simulation time of the network when the primary clock in the body controller fails. Unlike in the normal mode simulation, we find that as soon as the primary clock fails, all time domains disseminated by it cease to exist. We find that the simulator does not simulate past this event where a domain stops existing. However, the overall system is able to switch over to time domain 2 which is disseminated by the secondary clock in the main computer. The primary master clock in this configuration has a link failure as a result there is a stoppage...
in the two time domains’ time synchronization of the primary master clock. After our master clock fails, the active clocks differ in their time stamp.

C. Black hole network switch

Figure 9 shows the difference between the clocks from ‘true’ simulation time of the network when a black hole attack is performed on the front-left network switch which forces the front-left motor ECU to be left nonoperational. We see that this is the case by the thin blue line in Figure 9. The clock drift of this ECU was preset to be the same as the master clock to create a simpler graph.

V. Conclusion

In this paper, we have proposed a gPTP simulation model containing the main operations: time synchronization and propagation delay measurements. We demonstrated the failure of different modes affecting the ability of synchronization of ethernet networks present in automotive vehicles between ECUs. We used OMNet++ as the simulation tool to showcase different scenarios such as normal operation, fail-over to the hot-standby clock, and a black hole attack on one network switch. We also showed how a black hole attack can effect the time synchronization between master clock and slave clocks. Through our experimentation, we demonstrated the vulnerabilities that exist in the present ethernet networks present in automotive vehicles. Further research can be done to overcome this vulnerability and create a more secure ethernet network system by implementing additional redundancy checks that each ECU node is synchronized with the rest of the network.

ACKNOWLEDGMENT

The authors would like to thank Prof. Otman Basir for giving the time and resources to choose and develop our ideas into this report.

REFERENCES

[1] [Online]. Available: https://standards.ieee.org/wp-content/uploads/import/documents/other/d1-10_jesse_making_gptp_capable_for_secure_time_synchronization.pdf
[2] [Online]. Available: https://www.ieee802.org/1/pages/802.1as.html
[3] “Effects of time synchronization on time-aware shaping.” [Online]. Available: https://inet.omnetpp.org/docs/showcases/tnscombiningfeatures/gptppandtas/doc/index.html
[4] “Using gptp.” [Online]. Available: https://inet.omnetpp.org/docs/showcases/tns/simtimesynchronization/gptp/doc/index.html
[5] “Vulnerability-analysis-of-time-synchronization-in-automotive-ethernet.” [Online]. Available: https://github.com/RishiKakade/Vulnerability-Analysis-of-Time-Synchronization-in-Automotive-Ethernet
[6] B. Bob and L. Lo, “The case for ethernet in automotive communications,” SIGBED Rev., vol. 8, no. 4, p. 7–15, dec 2011. [Online]. Available: https://doi.org/10.1145/2095256.2095257
[7] A. Doshi, B. T. Morris, and M. M. Trivedi, “On-road prediction of driver’s intent with multimodal sensory cues,” IEEE Pervasive Computer, vol. 10, p. 22–34, 2011.
[8] H. J. Kim, U. Lee, M. Kim, and S. Lee, “Time-synchronization method for can–ethernet networks with gateways,” *Applied Sciences*, vol. 10, no. 24, p. 8873, Dec 2020. [Online]. Available: [http://dx.doi.org/10.3390/app10248873](http://dx.doi.org/10.3390/app10248873)

[9] J. Kim, S.-H. Seo, N. T. Hai, B. M. Cheon, Y. S. Lee, and J. W. Jeon, “Gateway framework for in-vehicle networks based on can, flexray, and ethernet,” *IEEE Transactions on Vehicular Technology*, vol. 64, pp. 4472–4486, 2015.

[10] A. A. Omar and J. M. Rhee, “Tfr: A novel approach for clock synchronization fault recovery in precision time protocol (ptp) networks,” *Applied Sciences*, vol. 8, no. 1, 2018. [Online]. Available: [https://www.mdpi.com/2076-3417/8/1/21](https://www.mdpi.com/2076-3417/8/1/21)

[11] H. Puttnies, P. Danielis, E. Janchivnyambuu, and D. Timmermann, “A simulation model of ieee 802.1as gtp for clock synchronization in omnet++,” 2018.

[12] S. Tuohy, M. Glavin, C. Hughes, E. Jones, M. Trivedi, and L. Kilmartin, “Intra-vehicle networks: A review,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 2, pp. 534–554, 2015.

[13] B. Zhao and N. Wang, “The implementation of ieee 1588 clock synchronization system based on fpga,” pp. 216–220, 2014.

[14] L. Zhao, F. He, E. Li, and J. Lu, “Comparison of time sensitive networking (tsn) and ttethernet,” *2018 IEEE/APAA 37th Digital Avionics Systems Conference (DASC)*, pp. 1–7, 2018.