Proceedings of the 2019 Winter Simulation Conference
N. Mustafee, K.-H.G. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y.-J. Son, eds.

A PARALLEL SIMULATION PLATFORM FOR TRAIN COMMUNICATION NETWORKS

Xin Liu and Dong Jin

Department of Computer Science
Illinois Institute of Technology
10 West 31st Street
Chicago, IL, USA

Tairan Zhang

CRRC Zhuzhou Institute Co., Ltd.
No. 169 Shidai Road
Zhuzhou, Hunan, CHINA

ABSTRACT

The next-generation of railway systems are increasingly adopting new and advanced communication technology to boost control efficiency. The challenges of this transition arising from the mission-critical and time-critical nature of railway control systems are to meet specific requirements, such as continuous availability, real-time operations, function correctness despite operational errors and growing cyber-attacks. In this work, we develop S3FTCN, a simulation platform to evaluate the on-board train communication network (TCN). S3FTCN is composed of a parallel discrete event simulation kernel and a detailed packet-level TCN simulator. S3FTCN demonstrates good scalability to support large-scale network experiments using parallel simulation. We also conduct a case study of a double-tagging VLAN attack to illustrate the usability of S3FTCN.

1 INTRODUCTION

Railway systems are national critical infrastructures. The next-generation high-speed railway system aims to integrate advanced computer and communication technologies to boost control efficiency. As the brain of the railway system, the communication and control system is responsible for monitoring, control and management with demanding requirements on automaticity, security, robustness, resilience, and real-time control. The basic operations, such as propulsion, braking, and door-control, as well as the advanced operations, such as routing, interlocking switching, and collision avoidance, are increasingly automated and optimized by various control applications. To support the applications with growing complexity, the underlying communication system is also evolving from a simple, proprietary, and closed network to an interconnected network to handle more complicated and heavier communication traffic. For example, Figure 1 depicts three types of communication networks: on-board network (Ethernet-based train communication network), train-to-train network (optical sensing), and train-to-ground network (Balise and GSM network).

However, the communication technology transition also raises new challenges. First, transmission data grow significantly in volume and diversity due to the new functionalities introduced by control applications. As shown in the train communication network standard published by International Electrotechnical Commission, IEC-61375 (Kirrmann and Zuber 2001), the messages between the on-board control devices include but not limited to control messages, monitoring data, diagnosis information, driver-assistant information, and passenger information. Some of which are frequently generated with cycle time in milliseconds. Therefore, we need to investigate the means to achieve the required quality-of-service by managing the delay, jitter, bandwidth, and packet loss. Secondly, as pointed out in (Lopez and Aguado 2015), the transition from closed to open systems brings cyber-security issues as the attackers are able to apply existing sophisticated methods to compromise the system. Given the mission-critical nature of the railway system, it is extremely important to ensure their protection against cyber-attacks and operational errors, which could result in loss of life or in massive financial losses in a worst-case scenario.

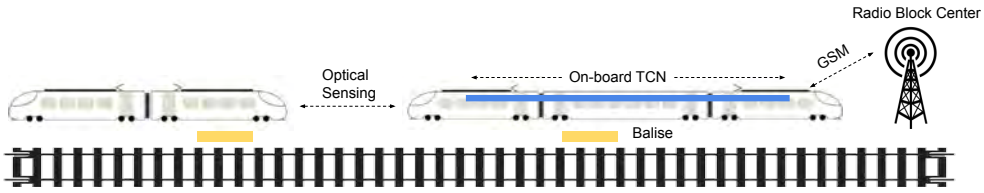


Figure 1: Railway communication networks.

The demanding requirements call for a testing platform to assist the research community to study the railway system regarding its cyber-security, resilience, and real-time control. Simulation is an important approach because of its flexibility, controllability, and lack of interference with real systems (e.g., avoid experimenting cyber-attacks on a train system). In addition, the scope of the next-generation railway system may make it infeasible to create a physical test system anywhere near its full scale.

In this work, we develop a parallel simulation platform, S3FTCN, for the on-board train communication networks (TCN). The on-board control network follows a two-layer architecture. Each train consist (i.e., a group of rail vehicles that make up a train unit) contains an Ethernet Consist Network (ECN) for transmitting message among devices within the network; and an upper-layer backbone network, called Ethernet Train Backbone (ETB), interconnects all the ECNs to enable a train-wide communication. There are two specific protocols for the TCN, one for topology discovery and IP configuration and the other for real-time data transmission. The scale of the network could be as large as 64 ECNs and 150-200 end-devices in each ECN, which contains thousands of traffic flows in the network. S3FTCN is composed of three layers to support the simulation of large-scale networks. The bottom layer contains a parallel discrete event simulation engine, S3F (Nicol, Jin, and Zheng 2011); the middle layer is a TCN simulator we developed according to the IEC-61375 standard (Kirrmann and Zuber 2001), including network topology, end-hosts, routing devices, and the specific network protocol stack; the top layer allows users to interact with the simulator by specifying the network and traffic configuration through a JSON file and reading/visualizing the statistical simulation outputs. To the best of our knowledge, S3FTCN is the first packet-level simulator for the Ethernet-based TCN with detailed models of the specific protocols. To evaluate the performance of S3FTCN, we conduct large-scale simulation experiments with different data transmission times and the number of flows while varying the number of timelines from 1 to 64 for exploring parallelism. The execution time and event rate are reported to demonstrate the scalability of S3FTCN for large-scale simulation experiments. To further illustrate the usage of S3FTCN, we present a case study to quantitatively analyze the impact of a VLAN attack in a TCN. Due to the misconfiguration of the VLAN in an ECN, an attacker is able to compromise a passenger device, from which launch a DoS attack with the double-tagging technique to intrude packets into the control network. The resulting packet delays in different network scenarios are displayed and analyzed using S3FTCN.

The rest of the paper is organized as follows. Section 2 presents the simulation framework, where Section 2.1 briefly describes the communication network that we modeled, and Section 2.2 describes the design and implementation of the simulation framework. Section 3 evaluate S3FTCN in terms of the parallel simulation performance. Sections 4 illustrates the VLAN attack case study. Section 5 presents the related work and Section 6 concludes the paper with future work.

2 S3FTCN SIMULATION FRAMEWORK

In this section, we first overview the on-board Train-Communication-Network including topology, protocols, and traffic profiles, and then present the design details of our simulation platform S3FTCN.

2.1 On-Board Train-Communication-Network

Train Communication Network (TCN) offers communication and control services for the Train Control and Monitoring System (TCMS) including control functions such as propulsion, brakes, door-control, air conditioning, etc. To enable those functions, TCN needs to transmit control and monitoring messages between the controller/operator and the electronic devices deployed across the entire train. In addition, it communicates with the ground system and reports the train status to the remote control center and other wayside systems.

The safety-critical nature of TCMS imposes strict requirements of the communication network. TCN is designed with the following features and our simulation platform realized all of them, including (1) delay-guaranteed and ack-based reliable communication of its control and monitoring messages; (2) several fault-tolerant mechanisms, such as link/node redundancies, to handle unpredictable physical/cyber disruptions during train operations; and (3) automatic network (re-)configuration during initialization stage, i.e., when train consists are coupled/decoupled. The configurations include assigning global IPs to end-devices and establishing mappings between application URIs and their IP addresses. For clarification, two types of TCN exists in practice, the traditional bus-based MVB/WTB network and the Ethernet-based ECN/ETB network recently proposed in the IEC-61375 standard. We refer TCN technology to be the latter in this work. We develop the TCN standard in our simulator S3FTCN including its topologies, protocols, and traffic profiles with details are presented as follows.

Topology. A TCN network has a two-layer structure as depicted in Figure 2. The design supports the communication of dynamically joined/removed consist devices, as train consists are allowed to be coupled/decoupled during the train operation. Each consist contains one or more Ethernet consist networks (ECNs), which are connected to an Ethernet train backbone (ETB) network. The ECNs, like other local area networks, interconnect the end-devices including local controllers, sensors, and monitors, and use L2 switches to forward the traffic within the network. The possible topologies include linear bus, ring, or ladder. On the other hand, Ethernet train backbone nodes (ETBNs) are in charge of the communication between end-devices that are located at different consists. The ETBNs provide multiple functions performed by routers, DNS servers, and gateways, including (1) communicating with other ETBNs to construct the backbone network topology; (2) assigning global IP addresses to devices within its ECNs, and maintaining the mapping between the IP addresses and the URIs of the applications; and (3) routing all packets along the backbone and transmitting packets between local devices and external devices. ETBNs connect to each other to form a linear topology, and often there exists at least two communication links between peers for communication redundancy.

Protocols. To enable the backbone topology establishment, the TCN standard includes a layer 2 protocol named Train-Topology-Discovery-Protocol (TTDP) that resides on ETBNs' network interfaces. The basic procedure is that each ETBN sends messages to its neighbors in both directions, and notify the neighbors the number of peers on one's east side and west side; the receiver then constructs an ordered list of ETBNs based on the received information, and propagates the messages along with a cyclic redundancy check (CRC) value (calculated from the ETBN list) to its own neighbors; when all the received CRC values are equal, the topology information is converged among all ETBNs.

To guarantee the real-time delivery of the application messages, another protocol, Train Real-time Data Protocol (TRDP), is designed in the standard. TRDP is a layer 5 protocol on each end-device between the transport layer (i.e., UDP/TCP) and the application layer (i.e., train control and monitoring system). It abstracts the traffic among applications into two types: Process Data (PD) represents short periodical messages, such as train state information and sensor messages; while Message Data (MD) represents lengthy sporadic messages, such as diagnostic information and control information. For PD, the protocol takes the application data and the cycle time as inputs, and the sender transmits the packet to the receiver(s) with the cycle time to be a constant interval. The receiver(s) in turns starts a timeout session with the timeout value to be 1.3 times of the cycle time, beyond which data loss will be reported to the application. For MD, each transmission requires an acknowledgment message from the recipient to achieve reliable transmission.

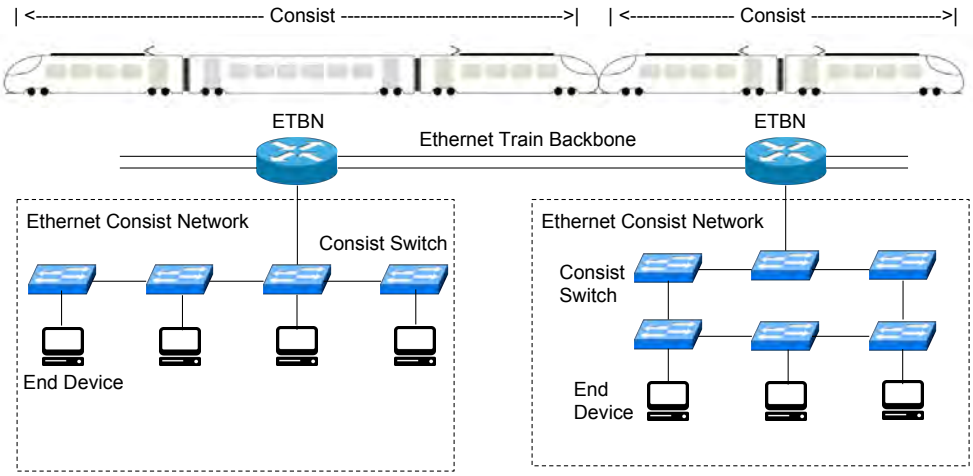


Figure 2: TCN topology.

Table 1: Scale and traffic profile of TCN.

	Maximum	Average
Number of EDs per ECN	500	150 - 200
Number of switches per ECN	20	6 - 8
Number of ECNs	64	2 - 6
PD packet size	1424 Bytes	
MD packet size	65459 Bytes	

Network Scale. The scale of the TCN network and the user traffic profile are illustrated in Table 1. The number of end-device in each ECN cannot exceed 500, and on average there are about 150 to 200 devices; the devices are connected by at most 20 consist switches and 6-8 switches on average. The supported number of ECNs is bounded by the global IP definition. An 8-bit field is defined to indicate which ECN the device is located, thus resulting in $2^8 = 64$ available ECNs within one TCN. The packet size of Process Data is limited to 1424 bytes in order to keep the data frame within 1500 bytes (because the TRDP header is 48 bytes, the UDP header is 8 bytes, and the IP header is 20 bytes). A Message Data has a maximum 65459-byte application layer payload because the size of an IP packet is limited to 65535 bytes.

2.2 S3FTCN Design and Implementation

The design architecture of S3FTCN is illustrated in Figure 3. S3FTCN is composed of three layers. The bottom layer is the S3F parallel simulation engine to support large-scale simulation; the middle layer contains the TCN simulation models; and the top layer provides an interface to allow users to configure experiments, visualize statistical results, and perform output analysis.

S3F is a parallel simulation engine that supports modular construction of simulation models in a way that potential parallelism can be easily identified and exploited (Nicol, Jin, and Zheng 2011). In S3F, a simulation is essentially the interactions among a number of entity objects. Each entity is aligned to a timeline, and each timeline contains an event list. We execute the event lists to advance the simulation. Within one timeline, no synchronization is required to simulate the interactions of co-aligned entities. S3F supports parallel simulation by allowing multiple timelines running simultaneously, and the timelines have to be carefully synchronized to guarantee global causality.

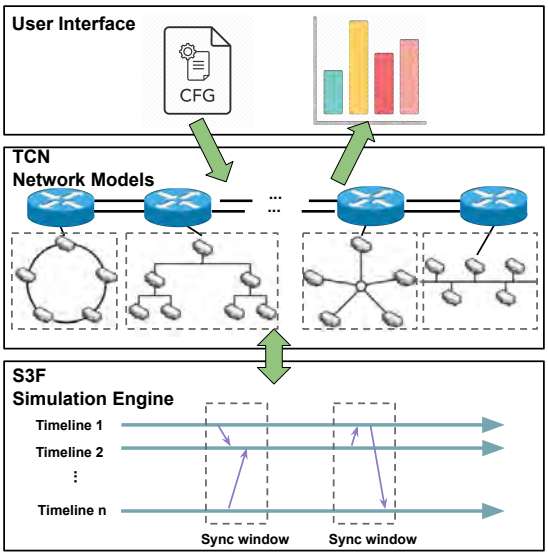


Figure 3: S3FTCN architecture.

We synchronize timelines using barrier synchronization. The main simulation thread spawns a number of timelines at the beginning. During the simulation experiment, we continuously compute synchronization windows, within which all timelines are safe to execute concurrently without being affected by other timelines. The synchronization mechanism is built around explicitly expressed delays across entities aligned to different timelines. In each synchronization window, the timelines coordinately determine the next synchronization window (e.g., the minimum value of the first event's timestamp plus the lookahead value among all the timelines). At the end of a synchronization window, cross-timeline events are processed and pushed to the event queue in other timelines and S3F then proceeds to the next synchronization window.

The middle layer provides sophisticated TCN models with protocols, topologies and traffic profiles described in the previous section. We model EDs, ETBNs, and consist switches as entities and communication links as channels among entities. We develop the TCN models with detailed TCN protocols (i.e., TTDP and TRDP) to support high fidelity analysis.

We allow users to easily specify parameters for simulation experimentation. Users can define the number of consists, ECNs, and EDs in a JSON configuration file. Users also can specify the details of traffic including the volume and distribution of global/local traffic flows, and PD/MD traffic flows. S3FTCN is capable to output user-defined statistics, such as per-packet or per-flow throughput/latency/drop rate data, to support output analysis.

Both of the simulation kernel and the TCN models are implemented in C++ using only the standard STL library. The use interface such as configuration file processing, stats collection, and visualization, are implemented in Python.

3 PERFORMANCE EVALUATION

S3FTCN is designed to support parallel simulation for large-scale networks. In this section, we conduct performance evaluation by reporting the execution time and event rate for simulation experiments.

3.1 Experimental Setup

With reference to the configuration parameters presented in Section 2.1, we constructed a backbone network with 64 ETBNs and each connects to an ECN. The parameters are are derived based on the IEC-61875

standard and the consultation with railway industry experts. We set up 100 EDs for each ECN and connect each ED to an L2 switch, which formed a linear topology. We randomly generated traffic flows across all EDs. We set the local/global traffic flow ratio to be 1:1 and varied the number flows as 500, 1000, and 1500. The user traffic transmission started at the 10th second in simulation time and ended at the 20th, 30th, and 40th second. We intentionally waited for 10 seconds before starting the traffic transmission so that the ETBNs could complete the global IP assignment using the TTDP protocol to enable cross-ECN communication. Although we configure the simulation to end at the 50th second, the simulation may terminate earlier since the event list could be empty after the last PD/MD packet was transmitted. On each end-device, the PD traffic was generated with a period of 5 milliseconds and the PD packet size was set to be 1424 bytes, and the transmission interval of the MD traffic follows an exponential distribution with a mean value of 10 milliseconds, and the MD packet size was 65459 bytes. The bandwidth of each network interface was set to be 1 Gb/s. We summarize the experimental parameters in Table 2. The experiments were conducted on a Dell KOI server with 40 CPU cores (2.60 GHz each). Each experiment was repeated 10 times.

Table 2: Summary of Experimental Parameters.

Number of ECNs	Number of EDs per ECN	Simulation Time	Traffic Transmission Time	Number of Flows
64	100	50 sec	10/20/30 sec	500/1000/1500
PD Cycle Time	PD Packet Size	MD Interval	MD Packet Size	Bandwidth
5 ms	1424 bytes	10 ms	65459 bytes	1 Gb/s

3.2 Experimental Results

We first plot the total packets sent/received/dropped in each experiment in Figure 4 to present the simulation workload of our experiments. While the number of flows is fixed to 500, with the increasing traffic transmission time, the total number of generated packets increases from around 570K to 1.15M and then 1.73M, while the drop rates are around 1.5%, as shown in Figure 4(a). Similar results are shown with the increasing number of flows (traffic time fixed at 20 seconds) in Figure 4(b), with the number of generated packets being 570K, 1.15M, and 1.71M and drop rates increase to 1.5%, 5%, to 7%, respectively. Also, we notice that the standard deviations of the 10 repeated trials are very small.

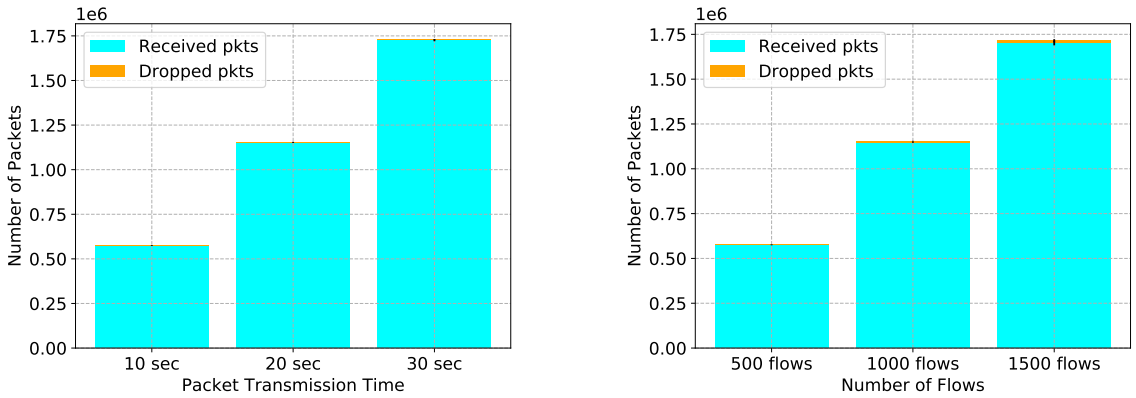
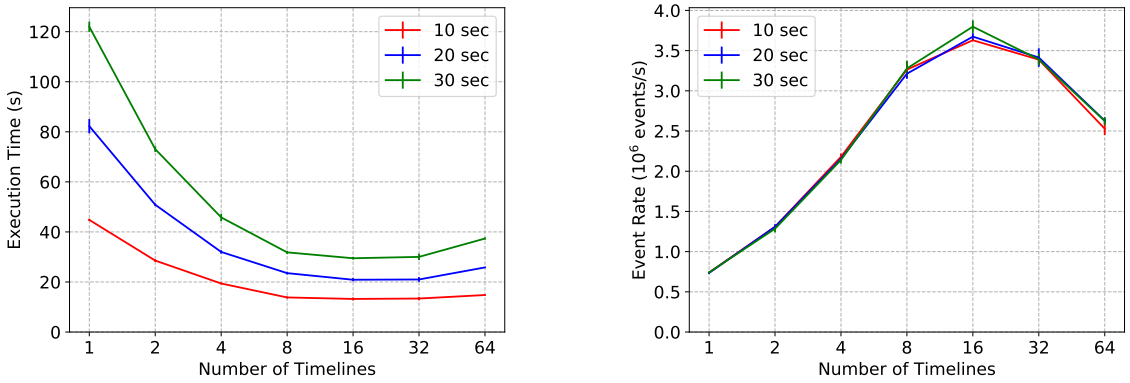


Figure 4: Number of packets generated with varying transmission time and traffic flows.

We now evaluate the simulation performance. We repeat the two sets of experiments mentioned above by varying the number of timelines ranging from 1 to 64. The parallel simulation experiments produce the same simulation results as the sequential simulation experiments (i.e., with one timeline) regardless of the number of timelines as the conservative synchronization correctly preserve the global causality.

To study the performance of parallel simulation, we plot the simulation execution time and the event rate under the network scenarios with various traffic transmission time in Figure 5. We observe that for all experiments, the execution time keeps decreasing as the number of timelines grows from 1 to 16. For example, in the 30-second transmission time case, the execution time is over 120 seconds with one timeline and falls below 30 seconds with 16 timelines, which is about 4x speedup introduced by parallel simulation. The results further justify the motivation of using parallel simulation to speed up simulation without losing fidelity, and ideally achieving real-time simulation. When the number of timelines further grows to 32 and 64, the execution time however increases. It is because the increasing number of timelines reduces the length of the synchronization window and thus more frequent synchronization is required during the simulation experiments. As a result, the increasing synchronization overhead reduces the benefit of parallel simulation. Figure 5(b) plots the event rates. The results match well with the corresponding execution time results in all three cases. As the number of timelines increases, the event rate keeps increasing and saturates at the peak value being 3.79M events/s with 16 timelines, and then drops with the further increase of the timelines.



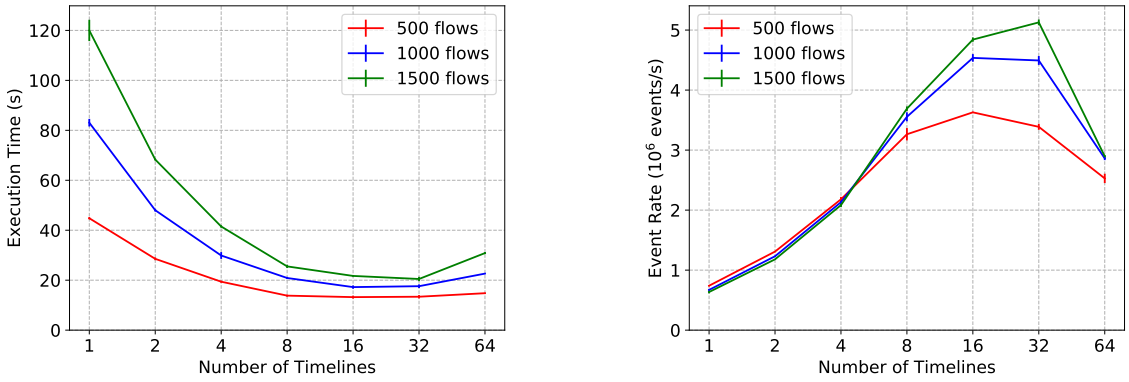
(a) Execution Time

(b) Event Rate

Figure 5: Simulation performance with the increasing packet transmission time.

In the next set of experiments, we set the packet transmission time to be 10 seconds and varied the number of flows as 500, 1000 and 1500. The average execution time and event rates over ten runs and their standard deviations are plotted in Figure 6. For the single timeline case, the execution time increases proportionally to the number of flows as shown in Figure 6(a). As the number of timelines increases, the execute time keeps reducing whose behavior is similar to ones in the first set of experiments. Another observation in Figure 6(a) is that we saved more execution time in the 1500-flow case than the 500-flow case as the number of timelines grows up to 16. It is because the 1500-flow case generates more simulation load per timeline and most events are processed within one timeline. In other words, each simulation thread has more independent work to process during a synchronization window and render the synchronization overhead less significant, and thus enhance the parallel simulation performance. The results of event rates are shown in Figure 6(b). When the number of timelines is small (≤ 4), the results are very close among 500-, 1000- and 1500-flow cases. As the number of timeline increases, the 1500-flow case achieves a higher event rate up to 5.1 million events/sec with 32 timelines, while the other two cases reach the peak

performance with 16 timelines. It further indicates that the synchronization overhead may increase as the number of timelines grows, and the performance is highly related to the simulation load on each timeline.



(a) Execution Time (b) Event Rate

Figure 6: Simulation performance with the increasing number of traffic flows.

4 CASE STUDY: ANALYSIS OF A VLAN ATTACK IN TCN

4.1 Double-tagging Attack

Besides the control systems, there exist other communication networks in the railway system during the train operation, such as the passenger entertainment system and Internet connection service. According to the IEC-61375 standard, the control network and passenger network may share the same underlying ECN infrastructure and be logically separated using the VLAN technology. Figure 7 illustrates such an architecture with two VLANs in one consist network. VLAN1 is used for the control network (i.e., TCN) and VLAN0 for the passenger network to stream entertainment information. The two networks connect to different backbones (i.e., ETB1 and ETB0) and they can only communicate to the end-devices within their own VLANs. For example, host h1 is able to send packets to h2 through h2's MAC address acquired from the ARP protocol. But if h1 wants to communicate to h3, its packets are forwarded to the router in ETB1. The packets will then be dropped since the control network is designed to be closed. Similarly, h3 cannot access h1 or h2 without going through the corresponding router as well.

However, the shared network devices introduce potential vulnerabilities, which may render the control devices to be accessed by malicious devices on the other VLAN. For example, a well-known exploit named "double-tagging attack" can be applied to this scenario if the networks are not carefully configured. Normally, when the sender and receiver are connected to different switches, say h1 and h2 in this case, the ports connecting two switches (s1 and s2) are trunk ports, which are in charge of adding VLAN tags to packets and identifying the destination VLAN that the packets belong to based on the tags. However, there is a special VLAN called "native VLAN", where the packets traveling through trunk ports have no tags. Assume the native VLAN is VLAN0 in our example, then h3 is able to launch the double-tagging attack with its packets forged to have two tags: the outer tag has VLAN0 and the inner tag has VLAN1, while the destination MAC address is set to be the broadcast address. When s1 receives the packets, it checks the outer tag and strips the tag before forwarding to s2 as it is the native VLAN. Then s2 gets the packet with a VLAN tag being VLAN1, then forward it to the two hosts in VLAN1. In this way, the packets are sent across VLANs without going through the router. A common way to avoid this attack in practice is to avoid assigning any hosts to the native VLAN.

Although the attack is well-known, it is still possible to launch the attack due to the fact that (1) vendors often keep the default settings of the switch ports which are assigned to the native VLAN for the easy configuration, and the native VLAN has well-known default values (e.g., Cisco switches has native VLAN to be 1); and (2) the network operator may pay more attention to the control network to give it a separate VLAN ID and leave the default native VLAN for the passenger network, whose end-devices are easier to be compromised as the devices might be connected to other public networks. Therefore, the double-tagging attack finds its way to inject packets into the closed control network, which enable a DoS attack to harm the availability of the critical control system.

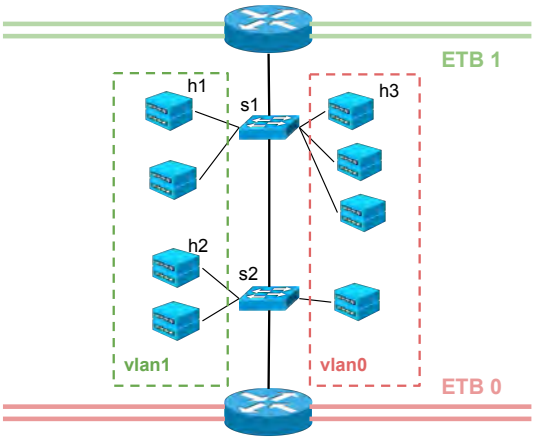


Figure 7: VLAN configuration.

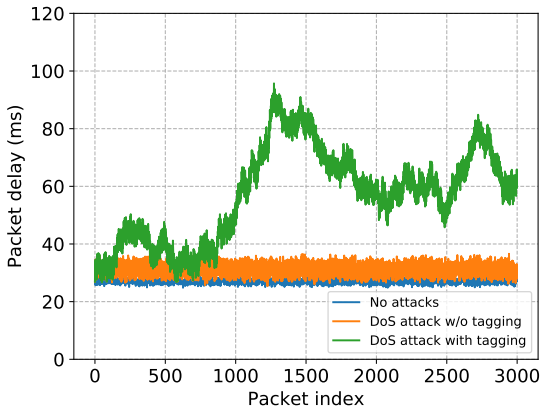


Figure 8: Packet delay under VLAN attack.

4.2 Simulation Experimental Results

To study the potential impact of the double-tagging attack on a TCN, we implemented the attack in S3FTCN and conducted simulation experiments. Assume the data traffic from h1 to h2 is constant and periodical with cycle time being 10 ms and packet size being 1000 bytes. Meanwhile, h3 sends the double-tagged packets with the same cycle time and packet size. VLAN1 is the control network and VLAN0 is the passenger network. The packets within VLAN1 has a high priority, which will be put in a priority queue during the transmission. The parameters are summarized in Table 3.

Table 3: Summary of Case Study Parameters.

Transmission Time	Cycle Time	Packet Size	Bandwidth	Buffer Size
30 sec	10 ms	1000 bytes	1 Mbits/s	400 KB

The experimental results are shown in Figure 8. We compared the end-to-end delay of each packet sent from h1 to h2 with the following three cases: (1) no DoS attack packets, (2) h3 sending DoS packets without double-tagging, and (3) h3 sending DoS packets with double tagging. In all three cases, h1 generates 3000 packets with no packet drops. This is because whenever the network interface starts transmitting a new packet, the trunk port of s1 will first check the existence of the data packet and then transmit the DoS packets only if the queue is empty. Therefore, without double-tagging, the impact on the data traffic is limited as we see the average delay increases from 28 ms to 35 ms in the first two cases. The increasing delay is introduced when a data packet arrives and the network interface sends a DoS packet. Since the scheduler is non-preemptive, the user packet needs to wait for the previous transmission to complete. However, when double tags are added to the DoS packet, the packet is enqueued in the output port from s2 to h2 with the same priority. As a result, we observe the delay of data packets increases up to 92 ms. The fluctuation of the delay is caused by the priority queueing mentioned above at s1. The double-tagging

attack originated from the outside devices is able to significantly harm the real-time requirement of critical control messages.

5 RELATED WORKS

5.1 Modeling and Simulation of Railway Communication Networks

Researchers have developed modeling and simulation tools for various railway communication networks. (Cho, Lee, Lee, Kim, and Kim 2001) designed and implemented a network simulator for the bus-based TCN, which is the previous generation of the Ethernet-based TCN that we modeled, and use it to investigate the performance characteristics of the network. Similarly, (Liu, Hou, and Fu 2011) developed a CAN-based on-board network simulator for the train diagnosis system, which was considered as an alternative bus technology of TCN. The tool was used to study the relationship between Bit Error Rates (BERs) and the throughput achieved in the network. (Pinedo, Aguado, Lopez, and Astorga 2015) built a modeling and simulation tool for the European Railway Train Management System (ERTMS), which is a part of the train-to-ground communication infrastructure as the counterpart of our on-board network. The main components of the simulator are the models of the control applications and the underlying wireless communication technologies (i.e., GSM and LTE). The tool was designed to study the cyber-impact on the railway operations. (Aguado, Jacob, Berbineau, Astorga, and Toledo 2009) focuses on the simulation of wireless communication. The authors built the control application (Automatic Control Service) on top of a general-purpose network simulator, OPNET Modeler. Additionally, (Unterhuber, Sand, Fiebig, and Siebler 2018) focuses on the low-level statistical modeling of the wireless train-to-train communication including a path loss model.

5.2 Cyber-security in Railway Networks

There are limited research works studying the cyber-security aspect of the railway communication networks, as the traditional railway communication networks are often considered as an isolated infrastructure. (Chen, Schmittner, Ma, Temple, Dong, Jones, and Sanders 2014) proposed a security analysis framework that combines the modeling tools for attack method analysis (e.g., attack graphs) and consequence analysis, e.g., failure mode and effects analysis (FMVEA). It aims to establish a unified framework for evaluating cyber-security impact on urban railway systems. A similar framework was proposed in (Kohli 2016) to identify the cyber-threats to specific railway assets. It employed a diamond model to identify the vulnerabilities and a FMVEA model to assess the impact. A comprehensive description of the security analysis of rail transit environment was written in (SS-CC 2015) as a white paper from the American Public Transportation Association (APTA). In this work, detailed procedures for security zone identification, entry point analysis, attack modeling and counter-measure analysis of the railway systems are illustrated.

Both (Lopez and Aguado 2015) and (Bloomfield, Bendele, Bishop, Stroud, and Tonks 2016) analyzed the vulnerabilities of the ERTMS system. The former mainly focused on issues in the specifications. For example, the choice of obsolete encryption schemes and the key distribution techniques may lead to potential exploits. The latter conducted security analysis at three levels, i.e., vulnerability identification of specifications, high-level risk assessment of a working system, and detailed risk assessment of the on-board systems. (Teo, Tran, Lakshminarayana, Temple, Chen, Tan, and Yau 2016) studied the impact of a specific DoS attack on the Shenzhen metro system using simulation. A train motion simulator and a traction power flow simulator are integrated to produce the outputs in terms of train movements and passenger flows with and without the attack. The authors directly changed the operational states as inputs without incorporating detailed models of cyber-components in the simulation.

6 CONCLUSION AND FUTURE WORK

In this work, we designed and implemented a simulation platform named S3FTCN to study the TCN network. The specific network topology and communication protocol stack are accurately modeled based on the IEC-61375 standard. The underlying parallel simulation engine supports large-scale network simulation. Using S3FTCN, we investigate a potential security exploit to a TCN system's vulnerability. We simulate a VLAN attack to evaluate the impact on the end-to-end message delay. In the future, we will further improve the parallelism of S3FTCN by studying lookahead for efficient cross-timeline synchronization. Additionally, we will extend S3FTCN to support wireless communication models in the railway networks and conduct a comprehensive cyber-security study.

ACKNOWLEDGMENTS

This work is partly sponsored by the Chinese-American Railway Transportation Joint Research Center at University of Illinois at Urbana-Champaign and CRRC Zhuzhou Institute Co., Ltd.

REFERENCES

Aguado, M., E. Jacob, M. Berbineau, J. Astorga, and N. Toledo. 2009. "Simulation Framework for Performance Evaluation of Broadband Communication Architectures for Next Generation Railway Communication Services". In *2009 9th International Conference on Intelligent Transport Systems Telecommunications,(ITST)*, 453–457. IEEE.

Bloomfield, R., M. Bendele, P. Bishop, R. Stroud, and S. Tonks. 2016. "The Risk Assessment of ERTMS-based Railway Systems from a Cyber Security Perspective: Methodology and lessons learned". In *International Conference on Reliability, Safety, and Security of Railway Systems*, 3–19. Springer.

Chen, B., C. Schmittner, Z. Ma, W. G. Temple, X. Dong, D. L. Jones, and W. H. Sanders. 2014. "Security Analysis of Urban Railway Systems: the Need for a Cyber-Physical Perspective". In *International Conference on Computer Safety, Reliability, and Security*, 277–290. Springer.

Cho, C.-H., J.-D. Lee, J.-H. Lee, K.-H. Kim, and Y.-J. Kim. 2001. "Design of the Train Network Simulator Based on Train Communication Network". In *ISIE 2001. 2001 IEEE International Symposium on Industrial Electronics Proceedings (Cat. No. 01TH8570)*, Volume 1, 343–347. IEEE.

Kirrmann, H., and P. A. Zuber. 2001, March. "The IEC/IEEE Train Communication Network". *IEEE Micro* 21(2):81–92.

Kohli, S. 2016. "Developing Cyber Security Asset Management Framework for UK Rail". In *2016 International Conference On Cyber Situational Awareness, Data Analytics And Assessment (CyberSA)*, 1–6. IEEE.

Liu, Z., Y. Hou, and W. Fu. 2011. "Communication Simulation of On-board Diagnosis Network in High-speed Maglev Trains". *Journal of Modern Transportation* 19(4):240–246.

Lopez, I., and M. Aguado. 2015. "Cyber Security Analysis of the European Train Control System". *IEEE Communications Magazine* 53(10):110–116.

Nicol, D. M., D. Jin, and Y. Zheng. 2011. "S3F: The Scalable Simulation Framework Revisited". In *Proceedings of the 2011 Winter Simulation Conference (WSC)*, edited by S. Jain, R. Creasey, J. Himmelspach, K. P. White, and M. C. Fu, 3283–3294. Institute of Electrical and Electronics Engineers.

Pinedo, C., M. Aguado, I. Lopez, and J. Astorga. 2015. "Modelling and Simulation of ERTMS for Current and Future Mobile Technologies". *International Journal of Vehicular Technology* 2015.

SS-CC, A. 2015. "Securing Control and Communications Systems in Rail Transit Environments". *Washington DC: American Public Transportation Association*.

Teo, Z.-T., B. A. N. Tran, S. Lakshminarayana, W. G. Temple, B. Chen, R. Tan, and D. K. Yau. 2016. "SecureRails: Towards an Open Simulation Platform for Analyzing Cyber-Physical Attacks in Railways". In *2016 IEEE Region 10 Conference (TENCON)*, 95–98. IEEE.

Unterhuber, P., S. Sand, U.-C. Fiebig, and B. Siebler. 2018. "Path Loss Models for Train-to-Train Communications in Typical High Speed Railway Environments". *IET Microwaves, Antennas & Propagation* 12(4):492–500.

AUTHOR BIOGRAPHIES

XIN LIU is a Ph.D. candidate in Computer Science major at Illinois Institute of Technology. His research interests include discrete-event simulation, modeling and simulation of computer networks, and network security and resilience. His email address is xliu125@hawk.iit.edu.

DONG (KEVIN) JIN is an Associate Professor in the Computer Science Department at the Illinois Institute of Technology. He holds a Ph.D. degree in Electrical and Computer Engineering from the University of Illinois at Urbana-Champaign. His

research interests include simulation modeling and analysis, trustworthy cyber-physical critical infrastructures, software-defined networking, and cyber-security. His email address is dong.jin@iit.edu.

TAIRAN ZHANG received his B.S degree from China University of Mining and Technology, Xuzhou, China in 2005, the M.S. degree from East China Normal University, Shanghai, China in 2008, and the Ph.D. degree in optical communication form Shanghai Jiaotong University, Shanghai, China in 2015. He is currently working for CRRC Zhuzhou Electric Locomotive Research Institute Co. LTD, Zhuzhou, China. His research interests include the modeling and experimental study of wireless optical communications and train network simulation platform research for rail transit. His email address is zhangtr@csrzic.com.