Performance Evaluation of Secure Industrial Control System Design: A Railway Control System Case Study

Xenofon Koutsoukos, Himanshu Neema,
Goncalo Martins, Sajal Bhatia, Janos Sztipanovits
Institute for Software Integrated Systems
Vanderbilt University Nashville, TN, USA

Keith Stouffer, Chee Yee Tang, Richard Candell
National Institute of Standards and Technology
Gaithersburg, MD, USA

Abstract—**Industrial control systems (ICS) are composed of sensors, actuators, control processing units, and communication devices all interconnected to provide monitoring and control capabilities. Due to the integral role of the networking infrastructure, such systems are vulnerable to cyber attacks. In-depth consideration of security and resilience and their effects to system performance are very important. This paper focuses on railway control systems (RCS), an important and potentially vulnerable class of ICS, and presents a simulation integration platform that enables (1) Modeling and simulation including realistic models of cyber and physical components and their interactions, as well as operational scenarios that can be used for evaluations of cybersecurity risks and mitigation measures and (2) Evaluation of performance impact and security assessment of mitigation mechanisms focusing on authentication mechanisms and firewalls. The approach is demonstrated using simulation results from a realistic RCS case study.**

I. Introduction

The exponential growth of information and communication technologies over the last decade has given rise to their expansion in real-world computing applications involving physical processes. This expansion has led to the emergence of closed-loop systems involving strong integration and coordination of physical and cyber components, often referred to as cyber-physical systems (CPS). These systems are rapidly finding their way into various sectors of the economy, such as industrial control systems, transportation, healthcare, and critical infrastructure. Increasing dependence on CPS renders them critical, and in-turn demands them to be secure, robust, reliable, and trustworthy, but it also makes them very attractive targets for cyber attacks.

Because of these disruptive changes, physical systems can now be attacked through cyberspace and cyberspace can be attacked through physical means. While CPS research addresses the tight interaction between the physical and cyber parts from the performance point of view, in-depth consideration of security and resilience in an integrated manner is still in early stages. The complex nature of CPS, mainly due to tight coupling of cyber and physical phenomena, makes securing such systems a challenging problem. A multi-vector attack exploiting a combined set of vulnerabilities from individual components, none of which might pose a serious threat to the stand-alone component, can have damaging effects in the overall system.

Industrial control systems (ICS) are a specific class of CPS in the juncture of control systems and cyber systems. ICS are composed of sensors, actuators, control processing units, and communication devices all interconnected to provide monitoring and control capabilities. In contrast to traditional computing systems, ICS must perform their critical functions without interruption. This paper focuses on railway control systems (RCS), an important and potentially vulnerable class of ICS and CPS. Cybersecurity is vital for ensuring that these systems can provide their critical services without disruptions that may result in catastrophic damages.

The objectives of this work are to analyze the cybersecurity risks of RCS, propose mitigation mechanisms, and evaluate their effectiveness as well as their performance impact on system operations. We propose to achieve these goals by developing a simulation integration platform that enables (1) Modeling and simulation of RCS including realistic models of cyber and physical components and their interactions, as well as operational scenarios that can be used for evaluations of cybersecurity risks and mitigation measures and (2) Evaluation of performance impact and security assessment of mitigation mechanisms. The main innovation of our approach is that research processes and results are documented as executable software models, simulations, and generated data that support cybersecurity analysis and design in a quantifiable manner. It should be noted that RCS are treated as any other network critical infrastructure and hence the proposed approach can be directly applied to other classes of ICS.

The paper presents a simulation-based integration platform

The work at Vanderbilt is supported by NIST (70NANB13H169). No approval or endorsement of any commercial product by the National Institute of Standards and Technology is intended or implied. Certain commercial equipment, instruments, or materials are identified in this paper in order to facilitate understanding. Such identification does not imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose. This publication was prepared by United States Government employees as part of their official duties and is, therefore, a work of the U.S. Government and not subject to copyright.

for RCS in order to perform experiments and acquire measurements to characterize performance and impact of secure control system design. The developed simulation integration platform uses a modular approach to integrate two open-source simulators: OMNeT++ [1] and Train Director [2]. The integration is based on a software tool infrastructure developed at the Institute for Software Integrated Systems at Vanderbilt University called Command and Control Wind Tunnel (C2WT) [3] which enables large scale heterogeneous simulations.

The platform enables the evaluation of the performance impact of implementing security solutions, complying with the ICS cybersecurity standards. The communication model used is based on the Advanced Train Control System (ATCS) [4] and the implemented security solutions comply with ICS cybersecurity guidelines [5]. In addition, the platform allows the evaluation of the performance of these applied security solutions against cyber-attacks. Specifically, this paper focuses on the evaluation of authentication mechanisms and firewalls. Authentication mechanisms in RCS incur both computational and communication overhead. Although the computational overhead is typically very small in modern microprocessor architectures, the communication overhead can result in time delays that need to be taken into consideration in the system design. Firewalls can serve a central role in securing RCS against a variety of external attacks and depending on the implementation, they can incur negligible performance impact.

The rest of the paper is organized as follows. Section 2 presents the simulation integration platform, Section 3 describes RCS focusing on the ATCS standard, Section 4 describes the simulation of RCS, Section 5 presents the evaluation results for the performance impact of authentication mechanisms and firewalls, and Section 6 concludes the paper.

II. Command and Control Wind Tunnel

A common problem with developing large-scale heterogeneous simulations is the complexity and effort required to integrate domain-specific simulation tools. Development challenges include how to integrate multiple simulation engines with varying semantics and how to integrate simulation models and manage the complex interactions between them. The High Level Architecture (HLA) provides the structural basis for simulation interoperability, distributed simulation, and is the standard technical architecture for heterogeneous simulations [6]. HLA provides application programming interfaces (APIs) that have helped to reduce the complexity of integrating multiple different simulation engines, but many challenges remain in such environments. As an example, HLA does not specify any tools to design or deploy a federation. It primarily standardizes runtime support for various tasks, such as coordinated time evolution, message passing, and shared object management. As a result, the HLA framework requires a significant amount of tedious and error-prone hand development integration code [3].

C2WT was developed to address the challenges present in the HLA framework [3]. C2WTis a graphical environment for designing and deploying heterogeneous simulation federations. Its primary contribution is to facilitate the rapid development of integration models, and to utilize these models throughout the lifecycle of the simulated environment. An integration model defines all the interactions between federated models and captures other design intent, such as simulation engine-specific parameters and deployment information. SIM uses the Generic Modeling Environment [7] and a custom Domain-Specific Modeling Language (DSML) for the definition of integration models. This language facilitates the easy capture of all of the design details for the simulation environment.

C2WT integration models follow the conceptual architecture depicted in Figure 1. A simulation environment is composed of multiple 'federates', each of which includes a simulation model, the engine upon which it executes, and some amount of specialized glue code to integrate the engine with the simulation bus. Both the engine configuration and the integration (or 'glue') code needed for each federate is highly dependent upon the role the federate plays in the environment, as well as the type of simulation engine being utilized. The main differences from HLA are the automatic generation of engine configurations, glue code to integrate the engine with the simulation bus, as well as scripts that allow the automation simulation execution and data collection. This integration enables a robust environment for users to rapidly define complex heterogeneous simulations.

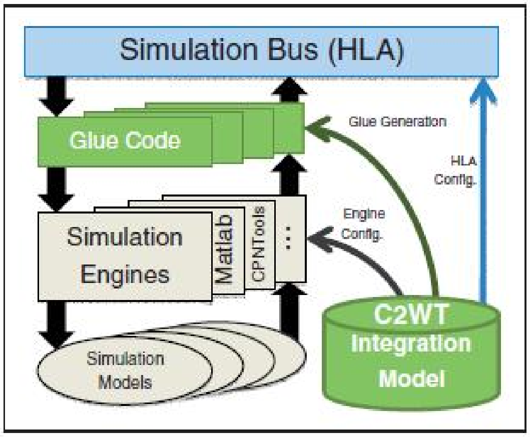


Fig. 1. C2WT Architecture

III. Railway Control Systems

The C2WT integration platform is used for simulation of RCS. The railroad network control infrastructure consists of the following main components:

Dispatch Center. The dispatch center (also known as Central Control Center) is a centralized control center for train management. It usually has a high bandwidth connection with the carrier network (e.g., MPLS/IP), but it could have any IP services.

Wayside Equipment. This is equipment located at the side of the track, such as signal controllers, switch circuit controllers, interlocking controllers, and various sensors for sending information back to dispatch center. Depending on the overall infrastructure, a variety of communication networks are used, such as cellular (e.g., GPRS), 900 MHz ATCS data network, and wired connection.

Locomotive Equipment. The locomotive equipment comprises of onboard equipment using a communication gateway to communicate to the base station. There may be different wireless networks such as 802.11 WiFi, cellular, 900 MHz ATCS data radio, and 160 MHz voice analog radio (individual locomotives may have a different mix).

Communication System. 900 MHz ATCS and 160 MHz voice radio are two commonly used wireless communication systems. These are legacy systems that have been used in the railroad infrastructure. WiFi and other cellular communication systems are IP based. Positive Train Control (PTC) is a more recent communication system in the US railroad infrastructure network, especially for the Class I railroads. This is a 220 MHz IP based communication network currently undergoing a large scale deployment. PTC would be used both for the onboard and wayside equipment communication, with WiFi (802.11x) reserved for terminals, yards, and other railroad facilities.

A. Advanced Train Control System (ATCS)

The Advanced Train Control System (ATCS) is an open standard that provides safe, cost efficient, and modular systems for wireless communication in railroads [8], [4]. ATCS is primarily used for monitoring and controlling signals and switches to manage the movement of trains [9]. ATCS provides compatibility of systems across railroads, modular growth path, vendor interoperability, and the ability to selectively choose capabilities and features based on specific needs.

ATCS comprises of five main systems as shown in Figure 2. Four of these systems are information gathering and processing systems – central dispatch system, on-board locomotive system, on-board work vehicle system, and field system – with the fifth system being the data communication system responsible for seamlessly interconnecting all the other systems.

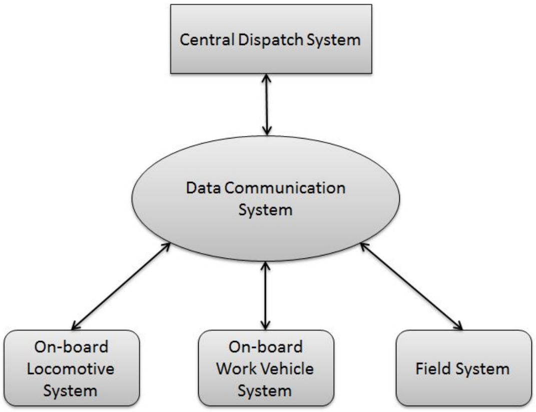

Central Dispatch System
Data Communication System
On-board Locomotive System
On-board Work Vehicle System
Field System

Fig. 2. ATCS Architecture Overview [4]

The Central Dispatch System is primarily responsible for managing the movement of trains throughout the railroad network with the aim to ensure safe operations without incurring delays. The On-board Locomotive System is responsible for providing automatic tracking and reporting of the vehicle as well as automated transmission of switch monitoring and control information via the Data Communication System. The On-board Work Vehicle System is responsible for providing the capability for track maintenance foremen to communicate with the Central Dispatch System via the Data Communication System. The Field System is used for monitoring and controlling wayside equipment such as switch and signal controllers, interlocking controllers, and various sensors.

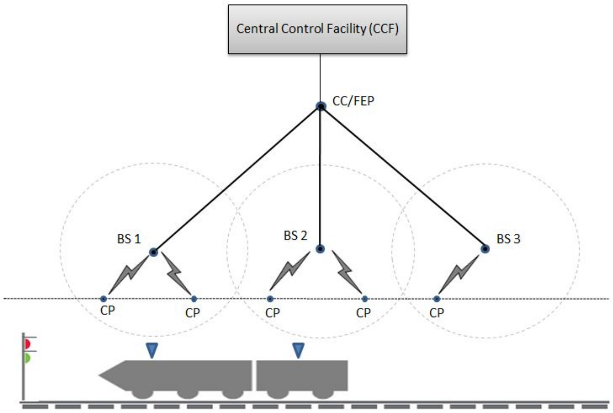


Fig. 3. Typical ATCS Network Architecture

A typical ATCS network architecture configuration consists of the main device types managed and controlled by the Central Control Facility (CCF) as shown in Figure 3. The Cluster Controller or Front End Processor (CC/FEP) coordinates all the ATCS traffic and is directly controlled by the CCF. Each CC/FEP is capable of handling multiple Base Stations (BSs) (also referred to as the Base Communication Packages BCPs). The communication link between CC/FEPs and the BSs is usually via high-speed wired lines. A key component of the ATCS architecture is the Control Point (CP), also referred to as the Mobile Communication Package (MCP), which is an interface for the on-board and wayside equipment (such as traffic lights and signals) to communicate to the CCF via BSs. Each BS serves a number of CPs. Communication between CPs and BSs usually employs full duplex wireless channels operating at different uplink and down frequencies. The communication from BSs to CPs usually operates at 935 MHz whereas the wireless channel uses 897 MHz to communicate from CPs to BSs. Typically, each CP is served by at least two BSs in order to ensure redundant communication paths in case of BS failure. Multiple BSs can receive data from a single CP which is then relayed to the CC and onto the CCF, while during the reverse path, the CC/FEP selects a BS to send the control or monitoring signal to the CP.

IV. RCS Simulation

For the assessment of security mechanisms in RCS, the SIM integrates two simulation tools: The network simulator OMNET++ [1] and the centralized traffic control simulator Train Director [2]. The integration allows the simulation of realistic scenarios in RCS that include cyber and physical phenomena as well as their interactions. Figure 5 depicts the infrastructure of an integrated simulation scenario. The railway layout used for the results in this paper is a modified version of the Oulu (OL) railway station in Finland available at [2].

A. *Computer Network Simulation*

OMNeT++ is a discrete event simulator that is widely used as a standard tool for studying protocols (for both wired and wireless networks), and modeling communication networks and distributed systems [1]. The simulation model is specified using an architecture description language called NED (Network Description). The language implements the desired communication model in terms of simple modules, compound modules, and a set of gates for handling the communication between these modules. The communication is governed by a set of customizable channel models and messages. The tool also provides a number of data gathering methods, including packet captures, for post-simulation analysis.

The ATCS network architecture described in Section III is mapped to a communication model shown in Figure 4. In the CCF, the controllers and servers (web and database) are connected to a Layer 2 (L2) network switch, which, in turn, is connected to the edge or gateway router. This router is connected to the gateway router of the Railroad Infrastructure Network via the Internet. Within this network, the first level L2 switch, representing the CC/FEP is connected to three BSs connected among themselves via L2 switches. The BSs are further connected wirelessly to the CPs that are responsible for controlling and monitoring the on-board and wayside equipment.

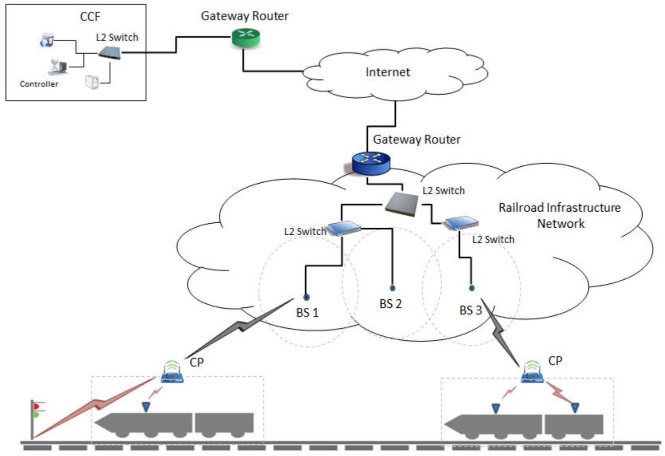


Fig. 4. Preliminary Communication Model

This model is developed in OMNeT++ and includes one instance of each of the three key devices in the ATCS architecture discussed above (Figure 5). The TDoperator node represents the CCF connected to the edge router (ccfGwRouter) via an L2 switch (ccfSwitch) using standard Ethernet (ethline) as the communication link. The gateway router of the CCF is further connected to the railway infrastructure network via the rwnGwRouter. The railway infrastructure network is connected to a single BS (baseStation) via high speed fiber optic links (fiberline). The base station connects to the CP (controlPoint) via the ATCS link operating at 900 MHz monitoring and controlling the wayside equipment (traffic lights and switches) via standard WiFi links operating at 2.4 GHz. In this initial model, the Internet link connecting the CCF network to the railway infrastructure network is omitted, however, it can be included in a large scale railway infrastructure model.

B. *Railway Simulation*

Train Director is a clone of Train Dispatcher, a software simulating a traffic controller for railroads [2]. Train Director simulates the work of a real-life dispatcher working as a Centralized Traffic Control (CTC) and controlling the movement of a number of trains. The key task in Train Director is to direct trains to their final destination by controlling switches and signals. Penalties are imposed for incorrect or inefficient operations such as incorrect destination and late arrivals respectively.

Train Director comprises of four key elements: tracks, signals, trains, and itineraries. Each of these elements has certain associated parameters and functions associated with them. The parameters are user defined and given at design time. Functions associated with the elements are performed based on the occurrence of specific events during the simulation. The Train Director simulator can function as a server, allowing other software to communicate externally with the simulator. In this mode, a socket connection is used to receive commands from external programs and is used for the simulation of the RCS case study.

V. Evaluation of Secure RCS Design

Our objective is to evaluate the performance impact of security mechanisms that comply with the ICS cybersecurity standards. ISA/IEC-62443 is a series of standards and technical reports that define procedures for securing IACS (Industrial Automation and Control Systems) against cyber-attacks [10]. In addition, NIST Special Publication 800-82 – Guide to ICS Security is used to identify security mechanisms for RCS [5]. This paper focuses on ISA-TR62443-3-1 which reports suitable security technologies for IACS security [10]. In particular the paper considers authentication mechanisms to secure the communication links and firewalls to filter external unauthorized messages.

A. *Hash-Based Message Authentication*

Authentication can be used to protect messages in the networking infrastructure against integrity attacks. However, the

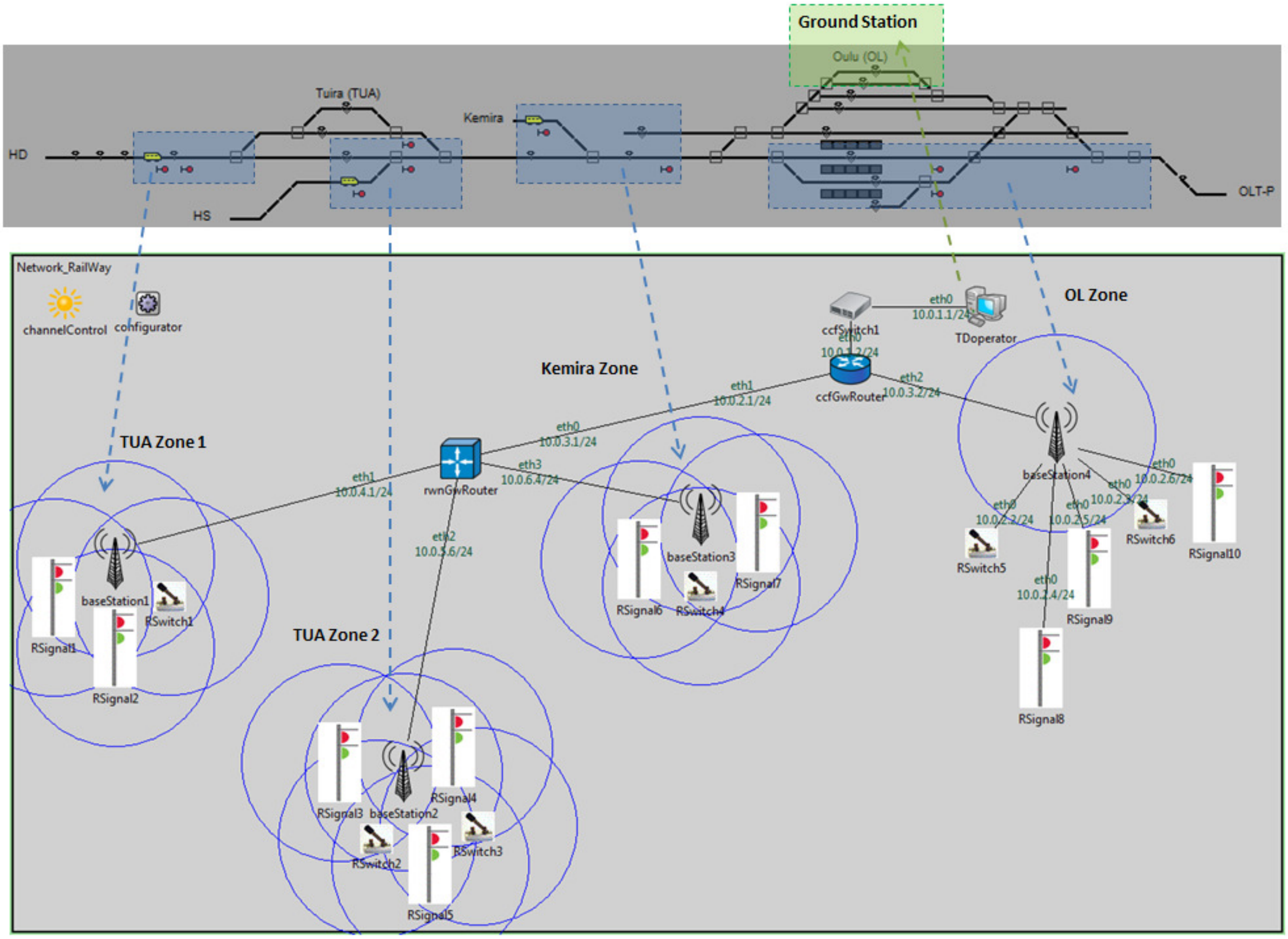


Fig. 5. Simulation Scenario

computational and communication overhead of authentication mechanisms may impact the performance of RCS. The objective is to perform a comprehensive evaluation of the computation and communication overhead of the implementation of authentication mechanisms based on the simulation integration platform. We consider a keyed-Hashed Message Authentication Code (HMAC) [11] to protect the integrity of a message. The first goal is to measure the computational overhead on the sender and receiver nodes (e.g., TDoperator and RSignal1) due to implementation of the authentication mechanism. HMAC generates additional information that needs to be attached to the original message (tag). The second goal is to measure the communication overhead on the network as a result of adding information on the message desired to transmit.

HMAC generates a tag by combining a cryptographic hash function with a secret cryptographic key. The tag is appended and transmitted with the original message (Figure 6). The cryptographic hash-functions should be one-way and collision resistant. It is computationally infeasible to find a message which corresponds to a given message digest, or to find two different messages which produce the same message digest. Any change to a message in transit will, with very high probability, result in a different message digest, and the signature will fail to verify. The strength of the HMAC depends upon the cryptographic strength of the underlying hash function, the size of its hash output, and on the size and quality of the key [11]. In this paper, three cryptographic hash functions are implemented and evaluated: ***SHA-1*** (a 160-bit hash function), ***SHA-2*** (SHA-256 hash function with 32-bit words, and ***SHA-3*** (Keccak hash function that supports the same hash lengths as SHA-2, but its internal structure is significantly different from the rest of the SHA family [12].) For all the hash functions, a secret cryptographic key with 64 bytes is used. The unique tag message authentication code generated by the hash-algorithms simultaneously verify the data integrity and the authentication of a message. Sender and receiver share the same key. The extra message tag overhead in bytes introduced is dependent on the message tag generated by the cryptographic hash-function used in HMAC. For SHA-1 the message tag is 20 bytes and for SHA-2 and SHA-3 the message tag is 32 bytes.

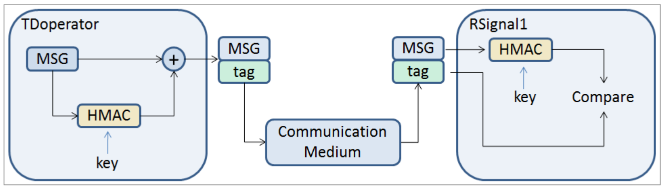


Fig. 6. Message Authentication Scheme

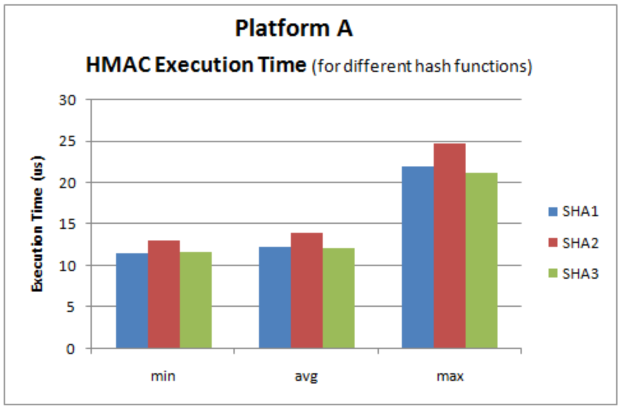


Fig. 7. HMAC Kernel Execution Time for Different Hash Functions

1) Computational Overhead: The simulation integration platform is not able to provide measurements for estimation of the computation overhead since its purpose is not to simulate hardware or microprocessor performance. However, if empirical measurements of the execution times are available, they can be incorporated in the simulation to evaluate the overall system performance. In order to acquire the empirical measurements, we implement and test HMAC in two platforms that represent possible hardware or microprocessor configurations in RCS.

- **Platform A:** Self-contained unit (IBX-530W) that includes a processor (1.6 GHz Intel Atom processor) with 1GB of memory and 512 MB of cache, and a real time operating system based on RTLinux and Ubuntu (Linux kernel 2.6.24-24-rt);
- **Platform B:** Single board unit (Trimslice2) with a CPU based on the NVIDIA Tegra2 SoC - a dual core 1 GHz ARM Cortex-A9 CPU with 1 GB of RAM, and an operating system based on Ubuntu 12.04 (Linux kernel 3.1.10-l4t.r16.02).

The platforms provide sufficient hardware and software resources for empirical evaluation of the computational overhead. All the software is running at the kernel space managed by a RTLinux scheduler, guaranteeing real time execution. RTLinux provides a crypto library that allows implementation of HMAC in a simple manner. Although these platforms may provide more resources than typical devices already deployed in RCS, they are representative of microcontrollers that are currently available for railway systems.

In order to evaluate the computational overhead, the HMAC execution time is used as the evaluation metric. The minimum, maximum, and average execution time for each hash function (SHA-1, SHA-2 and SHA-3) for a packet size of 60 bytes plus the respective message tag was measured. Figure 7 summarizes the measurements for Platform A. The execution times in Platform B are similar but SHA-3 is not implemented due to kernel incompatibilities. From the experiments, the maximum HMAC execution time is 25 μs for Platform A and can be incorporated in the simulation integration platform.

2) Communication Overhead: Our objective is to measure the communication overhead due to the the authentication hash tag. We first analyze theoretically the expected delay, and then we compare with the simulation results. Computer networks introduce delays between hosts. There are various types of delays in networks that occur due to various factors shown in Figure 8:

- **Processing Delay** (t_{proc}) is the time taken by the hardware or microprocessor to access information in a packet. This time can include for example, overhead in accessing packet header information, bit error calculation, and/or compute encryption or decryption algorithms;
- **Access Time** (t_{acc}) is the time that a packet has to wait before it can be transmitted over the link. Normally, this delay is related with the Medium Access Control (MAC) protocol used to access the transmission medium;
- **Transmission Delay** (t_{trans}) is usually caused by the data rate of the link. It is the time taken to push all the packet bits on to the link. For example, if the data rate of the link is 100 Mbps (12.5 MBytes) and the packet size is 100 Bytes, then $t_{trans} = 100/(12500000) = 8\mu s$
- **Propagation Delay** (t_{prog}) is the time taken by the first bit of the packet to reach the receiver. It can be calculated by dividing the distance between two nodes and the speed of the propagation of the link.

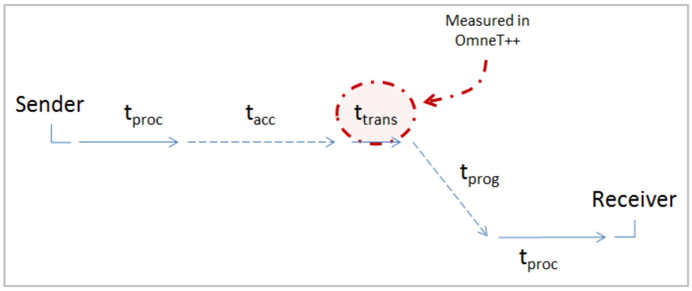


Fig. 8. Network Delays

We consider a 100 Mbits/s communication channel for wired environments and a 54 Mbits/s for wireless networks. The overhead on the frame size due to the generated hash tag is 20 or 32 bytes (depending on the hash function used). For the results in this section, we consider a tag size of 20 bytes (e.g., SHA-1). Table I shows the theoretical estimate for t_{trans} for one communication link, wired and wireless, and for frame sizes of 100 and 120 bytes.

OMNeT++ simulations can optionally create an event log file, which records simulation events such as: message cre-

TABLE I
THEORETICAL t_{trans} FOR ONE COMMUNICATION LINK

	t_{trans}	
	Wired	**Wireless**
100 Bytes	8 μs	14.8 μs
120 Bytes	9.6 μs	17.8 μs
Overhead	**1.6** μ**s**	**3.0** μ**s**

ations and deletions, event scheduling and cancellations, message sends and packet transmissions, and other information. For simplicity and to better illustrate how the communication overhead is estimated, we consider the communication between TDoperator and RSignal1 and RSignal10 (Figure 5) and we calculate the transmission delay (t_{trans}). The results are summarized in Table II. The table also summarizes the differences between the theoretical and the simulated transmission delay (t_{trans}). Despite the effort in running the experiments in a deterministic way, there is a slight difference between the theoretical and the simulated results. The simulated results present a small increase on the t_{trans} which is due to two facts: the sockets' header overhead is not considered in the theoretical calculations, nor is the access time delay (t_{acc}) from the wireless links.

TABLE II
THEORETICAL AND SIMULATED t_{trans} RESULTS

	t_{trans}			
	RSignal 10		**RSignal 1**	
	Theor.	**Meas.**	**Theor.**	**Meas.**
100 Bytes	32 μs	37 μs	46.8 μs	65 μs
120 Bytes	38 μs	41 μs	56.2 μs	70 μs
Overhead	**6.4** μ**s**	**4** μ**s**	**9.4** μ**s**	**5** μ**s**

In conclusion, both the communication and computational overhead are in the order of 10 μs for the example considered in this paper. This additional delay is relatively small and does not affect the physical components of the RCS.

B. Firewalls

Firewalls are an integral part of the defense mechanism for protecting RCS systems from a wide variety of external attacks. In order to simulate firewalls in the proposed simulation integration platform, a network filtering module is implemented at the edge router. The network topology is shown in Figure 9. The firewall module performs packet filtering based on a combination of source and destination IP addresses. The simulations include two types of attacks: internal and external. In the case of internal attacks, a denial-of-service (DoS) attack is simulated on the TDOperator node. In the case of external attacks, a rogue operator node (MaliciousTDOperator) is integrated into the network topology for sending various malign signals to disrupt the normal operation of the RCS.

Disabled Firewall and External Attack: The first simulation is performed without any firewalls, i.e., without "enabling" the filtering module in the edge router. Since no network packet filtering is implemented, the malicious packets from the MaliciousTDOperator node are able to penetrate into the network and disrupt the system operation. The results from this simulation are visualized with trains deviating from their schedules and arriving late at their desired destinations as seen in the scheduler part of Figure 10.

Enabled Firewall and External Attack: In this simulation, the packet filtering module is enabled in the edge router. As a result, the MaliciousTDOperator packets are filtered and prevented from entering the control systems' network. The simulation results confirm that all trains arrive on time at their respective destinations (figures are omitted due to length limitations).

Enabled Firewall Enable and Internal/External Attacks: The aim of this simulation is to demonstrate that although firewalls are the first line of defense and can protect a control system network from external attacks, they are not sufficient dealing with other cyber attacks such as those originating from inside or trusted sources. A DoS attack affects the TDOperator node and the simulation includes the effect of the DoS attack, which means that the TDOperator node does not respond to external attacks. The effect of the attack is that trains are forced to wait at non-scheduled stations (figures are omitted due to length limitations).

VI. CONCLUSIONS

The objective of this work is to evaluate the performance impact of implementing security mechanisms in RCS using a simulation integration platform. The platform is comprised by two open-source simulators (OMNeT++ and Train Director) and an infrastructure (C2WT) for deploying heterogeneous simulators.

A comprehensive evaluation of the computation and communication overhead of the implementation of an authentication mechanism is presented and simulated. To obtain realistic measurements for the computational overhead, the authentication mechanisms selected (HMAC SHA-1, SHA-2, and SHA-3) were tested in two different platforms. These platforms were selected to represent possible hardware or microprocessor configurations. The algorithms are implemented at the kernel space managed by a RTLinux scheduler, guaranteeing real time execution. The execution times are negligible and do not affect the overall system performance. For the communication overhead, we estimated analytically and we empirically measured the additional delay due to the adding authentication tags. The communication overhead is also negligible and does not affect the overall system behavior.

In addition, we simulated the effect of firewalls. A network filtering module was included in the simulated network system. The firewall simulations include two types of attacks, internal and external attacks. The aim is to demonstrate that even though firewalls are the first line of defense and can protect a control system network from external attacks, they are not sufficient for dealing with other cyber attacks such as those originating from internal or trusted sources.

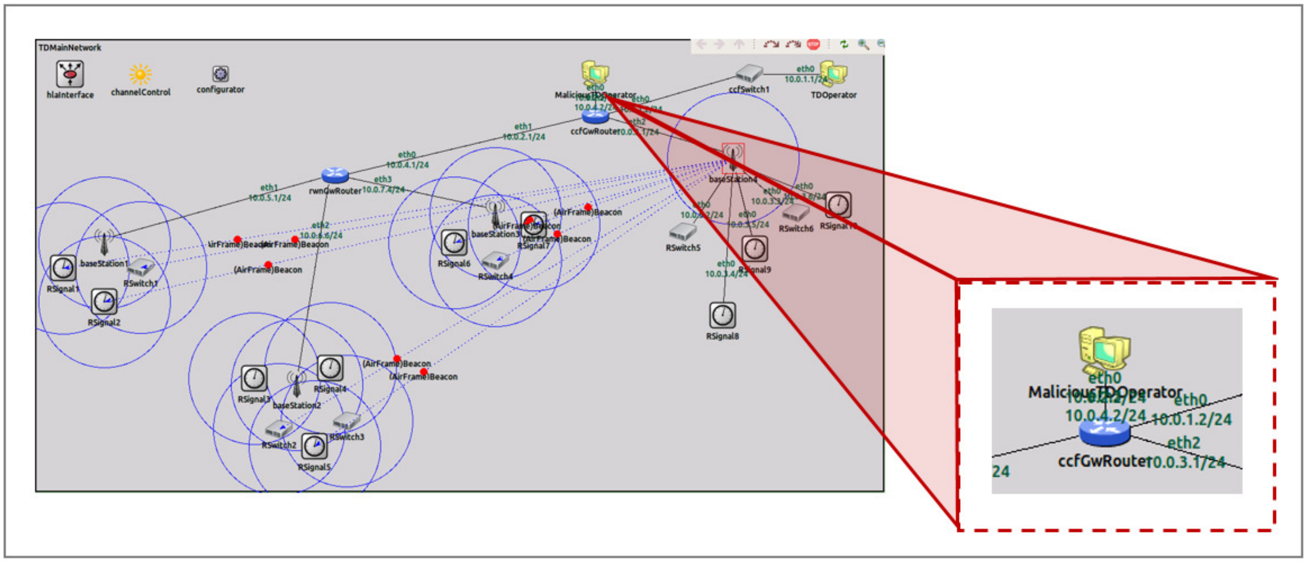
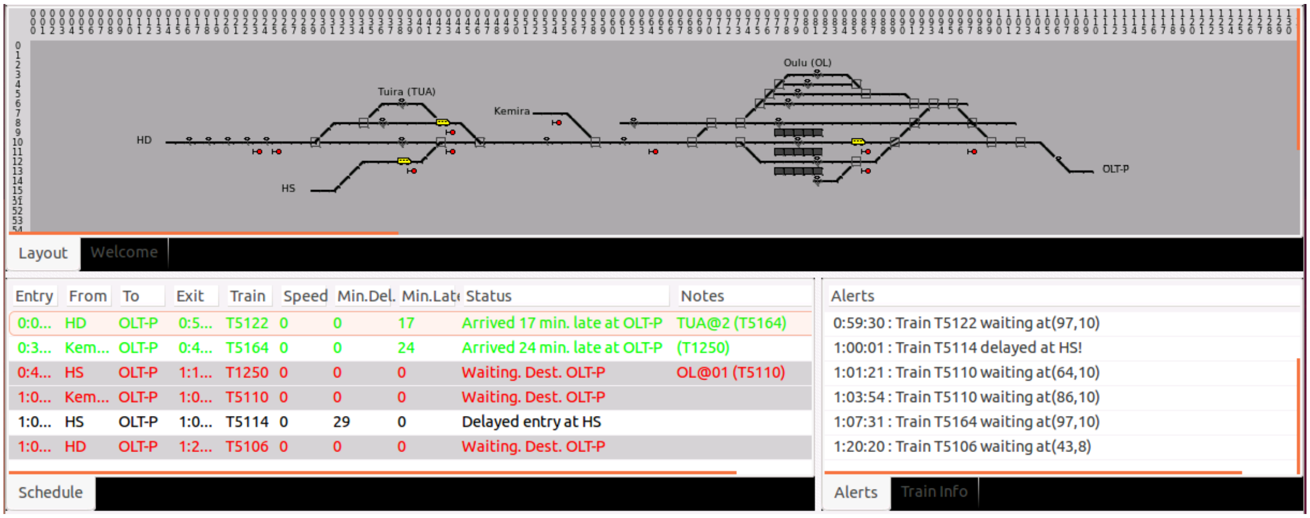


Fig. 9. Firewall and Malicious Node Network Placement



Entry	From	To	Exit	Train	Speed	Min.Del.	Min.Late	Status	Notes
0:0...	HD	OLT-P	0:5...	T5122	0	0	17	Arrived 17 min. late at OLT-P	TUA@2 (T5164)
0:3...	Kem...	OLT-P	0:4...	T5164	0	0	24	Arrived 24 min. late at OLT-P	(T1250)
0:4...	HS	OLT-P	1:1...	T1250	0	0	0	Waiting. Dest. OLT-P	OL@01 (T5110)
1:0...	Kem...	OLT-P	1:0...	T5110	0	0	0	Waiting. Dest. OLT-P	
1:0...	HS	OLT-P	1:0...	T5114	0	29	0	Delayed entry at HS	
1:0...	HD	OLT-P	1:2...	T5106	0	0	0	Waiting. Dest. OLT-P	

Alerts
0:59:30 : Train T5122 waiting at(97,10)
1:00:01 : Train T5114 delayed at HS!
1:01:21 : Train T5110 waiting at(64,10)
1:03:54 : Train T5110 waiting at(86,10)
1:07:31 : Train T5164 waiting at(97,10)
1:20:20 : Train T5106 waiting at(43,8)

Fig. 10. Disabled Firewall and MaliciousTDOperator Attack

References

[1] A. Varga and R. Hornig, "An overview of the omnet++ simulation environment," in *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, p. 60.

[2] BackerStreet. Train director. http://www.backerstreet.com/traindir/en/trdireng.php. [Online]. Available: http://www.backerstreet.com/traindir/en/trdireng.php

[3] G. Hemingway, H. Neema, H. Nine, J. Sztipanovits, and G. Karsai, "Rapid synthesis of high-level architecture-based heterogeneous simulation: a model-based integration approach," *Simulation: Transactions of the Society for Modeling and Simulation International, pp. 16, March*, 2011.

[4] *System Architecture: ATCS Specification 100 (Revision 4.0 – http://www.atcsmon.com/)*, Association of American Railroads, 1995.

[5] K. Stouffer, J. Falco, and K. Scarfone, "Guide to industrial control systems (ICS) security," NIST Special Publication, Rev. 2, 800-82, 2015.

[6] D. J. S. Dahmann and M. K. L. Morse, "High level architecture for simulation: An update," *IEEE Proceedings on the 2nd International Workshop on Distributed Interactive Simulation and Real-Time Applications*, 1998.

[7] A. Ledeczi, M. Maroti, A. Bakay, G. Karsai, J. Garrett, C. Thomason, G. Nordstrom, J. Sprinkle, and P. Volgyesi, "The generic modeling environment," in *Workshop on Intelligent Signal Processing, Budapest, Hungary*, vol. 17, 2001, p. 1.

[8] P. V. Craven, "A brief look at railroad communication vulnerabilities," in *Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on*. IEEE, 2004, pp. 245–249.

[9] D. Williams, B. Metzger, and G. Richardson, "Spec 200 radio code line ducting–Cause and effect," in *Proceedings of the American Railway Engineering and Maintenance-of-Way Association Conference*, 2001.

[10] *ISA/IEC-62443 – Security of Industrial Automation and Control Systems*, International Electrotechnical Commission (IEC), 2013.

[11] W. Stallings, *Cryptography and Network Security: Principles and Practices*. 5th Edition, Prentice-Hall Press, 2010.

[12] H. Gilbert and H. Handschuh, "Security analysis of SHA-256 and sisters," 2004.