Chapter 18

SIMULATION OF ADVANCED TRAIN CONTROL SYSTEMS

Paul Craven and Paul Oman

Abstract This paper describes an Advanced Train Control System (ATCS) simulation environment created using the Network Simulator 2 (ns-2) discrete event network simulation system. The ATCS model is verified using ATCS monitoring software, laboratory results and a comparison with a mathematical model of ATCS communications. The simulation results are useful in understanding ATCS communication characteristics and identifying protocol strengths, weaknesses, vulnerabilities and mitigation techniques. By setting up a suite of ns-2 scripts, an engineer can simulate hundreds of possible scenarios in the space of a few seconds to investigate failure modes and consequences.

Keywords: Railroads, wireless networks, ATCS, ns-2, simulation

1. Introduction

The transportation sector has been categorized as a critical infrastructure [1]. Any significant disruption in the movement of goods or people threatens the nation's defense as well as its economy [25, 26].

The North American surface transportation network is a complex interacting system designed to provide safety and efficiency, but it is vulnerable to accidents, malicious attacks and natural disasters [28]. Failures at a small number of "choke points" can impact highway, rail, air and maritime transportation throughout the network [21].

Given the importance of surface transportation, it is important to identify and protect the essential components of transportation systems and to ensure resiliency [19]. Simulating failures and analyzing their effects enables engineers to identify critical vulnerabilities, facilitating the construction of reliable, survivable systems [4, 5].

Please use the following format when citing this chapter:

Craven, P. and Oman, P., 2008, in IFIP International Federation for Information Processing, Volume 290; *Critical Infrastructure Protection II*, eds. Papa, M., Shenoi, S., (Boston: Springer), pp. 243–256.

On any given day, the United States surface transportation network moves eight million truckloads of freight across four million miles of roadway using 1.5 million railcars on 170,000 miles of track [21]. Containerization makes the highway and railway freight systems inseparable. Unfortunately, highway and railway communication control systems are vulnerable to malicious and nuisance cyber attacks [3, 7–9]. Railway control communications, in particular, have evolved in an obscure but benign environment where railroad enthusiasts often eavesdrop on control signals to track railcar movements [14].

The U.S. rail system uses communication protocols for tasks ranging from changing track switches to controlling locomotives [6]. Most rail communication protocols do not have simulation environments, requiring engineers to run expensive, time-consuming field experiments to analyze their behavior. For example, the U.S. rail system is currently testing the Positive Train Control (PTC) system designed to prevent train collisions, derailments and injuries to workers [11]. Several pilot projects have been launched, including the North American Joint PTC for the St. Louis-Chicago corridor that sought to implement PTC using the Advanced Train Control System (ATCS) protocol.

The ATCS is widely used in the rail industry for wireless data communications. However, because no simulation models are available for ATCS, its suitability can only be evaluated in physical environments. Unfortunately, time, labor, and safety considerations severely limit the extent of physical testing that can be performed on railroad systems.

Realistic computer simulations of ATCS could enable engineers to test PTC system configurations in the laboratory. The designs could be validated using a pilot project and further simulations could be run on other track segments. Such an approach would have significantly enhanced the results of the North American Joint PTC effort.

Simulation environments also offer effective and safe testbeds for analyzing vulnerabilities. In January 2008, a 14-year-old boy built a wireless device that enabled him to control switches for a tram system in Lodz, Poland [2]. He is suspected of causing four derailments, including one in which twelve people were injured. An ATCS simulation model would have enabled engineers to debug these and other problems in a laboratory environment.

This paper describes a network simulation model of the ATCS for wireless railroad controls. The ATCS is modeled using Network Simulator 2 (ns-2), a popular discrete event simulation system. The simulation model enables engineers to gain a better understanding of ATCS strengths and weaknesses as well as its vulnerabilities and the appropriate mitigation strategies.

2. ATCS Overview

The ATCS is an open standard for wireless railroad data systems [6]. It is used for controlling signals, switches and monitoring trains over several thousand miles of rail in the United States [22, 30]. The ATCS specifications [1] span several volumes, covering everything from the dimensions of the hardware and electrical connectors to upper-level software protocols.

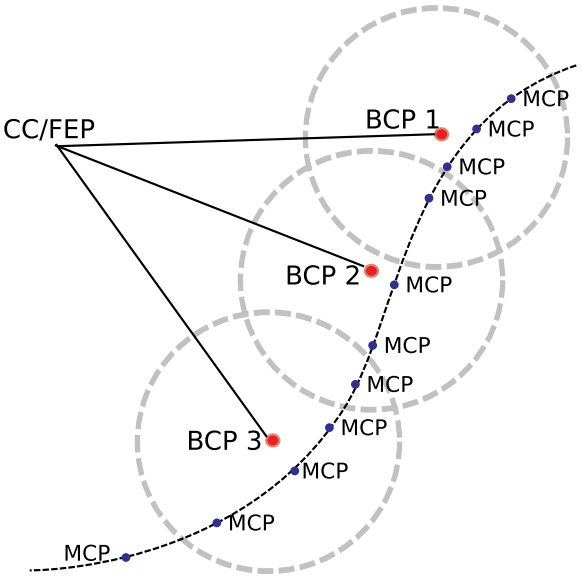


Figure 1. Example ATCS network.

A typical ATCS configuration involves three types of devices (Figure 1). The first is a cluster controller/front end processor (CC/FEP). The FEP coordinates all ATCS traffic and is directly controlled by a railway dispatch center. Several CCs may be connected to a FEP. The FEPs and CCs are usually combined into one unit (CC/FEP). Multiple CC/FEPs can be chained together using a specialized ATCS packet switch.

The second major device is the base communications package (BCP). Each CC coordinates traffic for multiple BCPs. Communications between CC/FEPs and BCPs usually employ wired lines. Typically, CC/FEPs handle around ten BCPs. While a CC/FEP can handle more BCPs, the effect of losing a CC/FEP is minimized by limiting the number of BCPs that connect to it.

The third type of device is a mobile communications package (MCP), which is an interface for equipment in locomotives and wayside devices (e.g., train signals, switches and track sensors). Locomotives use the BCP to send and receive information from wayside devices. Each BCP serves a number of MCPs. Communications between MCPs and BCPs is done wirelessly via full duplex using two separate frequencies. One frequency (typically around 897 MHz) is used for MCP to BCP communications. The other frequency (usually about 935 MHz) is for BCP to MCP communications. Other frequencies may be employed, depending on the licenses obtained by the railroad.

Usually, each MCP is serviced by at least two BCPs; this provides redundancy if there is a problem with a BCP or its communications. Also, MCPs are allowed to move between BCPs. Multiple BCPs usually receive transmissions

from an MCP, each BCP relays the data to the CC. When the CC has to send data to an MCP, it selects one BCP to transmit the data.

3. Network Simulator 2 Overview

Network Simulator 2 (ns-2) is a popular discrete event simulator for networking research [10]. Its flexible architecture supports 802.3, 802.11 and many other protocols, including new protocols that may be introduced as plug-ins. The ns-2 system handles protocols all the way from the physical layer (OSI Layer 1) to the application layer (OSI Layer 7).

Simulations in ns-2 have three major parts: C++ code for handling networking protocols, TCL-based specifications of a network setup, and code for data post-processing. Coding networking protocols such as TCP/IP and HTTP in C++ allows simulations to run fast. Most of the CPU time during a simulation is spent executing the protocol-handling code. C++ is also excellent for coding low-level bit manipulations required in network simulations. Networking code is usually written in C, a subset of C++; thus, the code is easily integrated and tested using ns-2.

Using the TCL scripting language to create nodes and network links allows quick changes to network setups. TCL also allows ns-2 to be used as a full-fledged development platform for simulations. In particular, a user can use TCL to create a suite of simulations to be run with just one command. The ns-2 system uses OTcl [29], an extension to TCL that supports object-oriented development.

Network simulations can generate extremely large log files, which may have to be processed with summarization and analysis tools. Text processing tools such as grep, sed, awk and perl can be used for data summarization. ns-2 incorporates nam, a network animator program that displays animations of network traffic. Programs like gnuplot and Excel can be used to create graphs.

4. Modeling ATCS

The ATCS simulator is implemented in ns-2 using four main components: scheduler, event interface, handler interface and TCL/C++ language bindings. The scheduler is the heart of the ns-2 system. All the discrete events, which include packet transmissions, are passed to the scheduler along with the times they should be processed. The scheduler then examines each event in order and triggers a "handler" to process the event. Finally, the ATCS simulator binds its C++ code to the TCL scripting language using the same method as the rest of the ns-2 code.

Most of the C++ classes provided in the ATCS simulator are modeled after OSI layers. For example, a C++ class is available for modeling the physical layer, and subclasses are available for wireless and wired modems. Because many ATCS applications are low-level in nature, higher OSI layers (e.g., presentation layer) may be combined with other layers to simplify coding.

Other classes are employed to represent network nodes. These classes follow the "composite pattern" [12]. A class that represents a network node contains multiple OSI layer objects. The node class itself has the same handler interface as the OSI layers.

The next most common class in the ATCS simulator creates the adapter [12] between TCL and C++. The adapter class is static and is responsible for creating and binding a "shadow" object in C++ for each object of that type created in TCL.

While the C++ code models the internals of a working ATCS, specific networks are set up using TCL code. It takes just 25 lines of TCL code to model the communications between the three major types of ATCS nodes: MCP, BCP and CC. Using data files, or even random numbers, makes it easy to put together large simulations involving many nodes.

5. Wireless Medium Modeling

This section describes the wireless medium assumptions and characteristics modeled with the ATCS simulation code written for ns-2.

5.1 Propagation Models

The ATCS runs full-duplex communications with two frequencies, one for outgoing data from the BCP and one for incoming data. The two frequencies are handled independently by the simulator. Propagation can be calculated based on free space loss on a flat plane or by using the Longley-Rice model, which takes elevation into account. By default, the network is assumed to be on a flat plane and the coordinates for each node are in the latitude-longitude format. The received signal strength is calculated using free space loss and the user can account for antenna gain and cable loss.

The Longley-Rice propagation model can be used if the simulation requires terrain effects to be taken into account. This model requires digital elevation models (DEMs) [27] and involves more computational overhead than the free space loss model. The free space loss and Longley-Rice propagation models are discussed in detail later in this paper.

5.2 Transmitter-Receiver Distance

The code for the wireless medium relays copies of each transmitted packet to the radio receivers listening on the corresponding frequency. The time taken for the signal to arrive depends on the distance to the receiver. The distance between the transmitter and receiver is calculated according to their latitudes and longitudes using an adaptation of the Haversine Formula:

$$\Delta lat = lat_2 - lat_1 \quad (1)$$

$$\Delta long = long_2 - long_1 \quad (2)$$

The distance between the transmitter and receiver (in meters) is computed using the following equations:

$$a = sin^2\left(\frac{\Delta lat}{2}\right) + cos(lat_1)cos(lat_2)sin^2\left(\frac{\Delta lon}{2}\right) \quad (3)$$

$$b = 2 \cdot atan2(\frac{\sqrt{a}}{\sqrt{1-a}}) \quad (4)$$

$$d = Rb \quad (5)$$

where lat_i is the latitude of point i (radians), $long_i$ is the longitude of point i (radians), $atan2(a, b)$ is the arctangent of a/b, R is the radius of the Earth ($\approx$ 6,371 km), a is the square of half of the straight-line distance, b is the great circle distance (radians), and d is the distance between the transmitter and receiver (km).

Whether the receiver can actually receive and decode the packet depends on how much power the signal has upon arrival. This is where the user has the option to use the free space loss propagation model or the Longley-Rice propagation model.

5.3 Power Loss

The ATCS model estimates the power loss during transmission between two nodes using the free space loss or Longley-Rice propagation models. The models incorporate parameters for antenna gain and signal loss from the cable and connectors.

Free Space Loss Propagation Free space loss [13] assumes that there are no obstructions between the two nodes and that the signal is not reflected (that would cause it to propagate further). Free space loss has been shown to provide a reasonable approximation for distance-dependent loss [24]. However, the model does not account for losses due to terrain obstructions.

The loss, l, is computed as:

$$l = \left(\frac{4\pi d}{\lambda}\right)^2 \quad (6)$$

where d is the distance between the nodes and λ is the wavelength of the signal.

The source signal spreads out in a sphere whose radius is the distance between the transmitter and the receiver; this gives the $4\pi d$ term in the equation. The longer the wavelength, the larger the antenna and the more signal that is captured; this gives the $\frac{1}{\lambda}$ term in the equation. Also, the signal strength drops off in proportion to the square of both terms. Since the radio channel is specified in terms of its frequency rather than wavelength, substituting for $\lambda = c/f$, yields:

$$l = \left(\frac{4\pi df}{c}\right)^2. \tag{7}$$

Since d is in meters, substituting the value for the speed of light $c \approx 3 \cdot 10^8$ m/s yields:

$$l = \left(\frac{4\pi df}{3 \cdot 10^8}\right)^2. \tag{8}$$

Taking the log_{10} of both sides to obtain values in dB, the equation reduces to:

$$\begin{aligned} l_{dB} &= 10\log_{10}\left(\frac{4\pi df}{3 \cdot 10^8}\right)^2 && (9)\\ &= 20\log_{10}\left(\frac{4\pi}{3 \cdot 10^8}\right) + 20\log_{10}(f) + 20\log_{10}(d) && (10)\\ &\approx -147.56 + 20\log_{10}(f) + 20\log_{10}(d). && (11) \end{aligned}$$

The ATCS simulation model uses Equation 11 to compute the free space loss.

Longley-Rice Propagation The Longley-Rice propagation model was first published in 1967 [20] and has since undergone several revisions [17]. The latest algorithm and model equations were published by Hufford [15].

The original Longley-Rice algorithm was written in FORTRAN. This algorithm was translated to Microsoft-specific C code by the Institute of Telecommunication Sciences (ITS) [16], and was subsequently modified by Magliacane [18] for use in a Unix environment. Magliacane's code is distributed under the GNU Public License (GPL). It incorporates DEMs, which are available free-of-charge from the United States Geological Survey [27]. A useful feature of the code is that it enables users to create signal propagation maps.

To implement Longley-Rice propagation in the ATCS ns-2 simulator, portions of the ITS code and Magliacane code were adapted and encapsulated in a C++ class, and then modified to interface with TCL and the data structures used by the simulator. This enables the ATCS simulator to use the Longley-Rice model by simply changing the name of the default propagation method, and having the DEMs loaded into the current working directory.

The results of an ATCS simulation can be saved in comma-delimited or packet-dump formats. The comma-delimited format is handled by spreadsheets and text processing tools (e.g., grep, awk and perl). The packet-dump format can be loaded directly into ATCSMon, the ATCS packet-inspection tool.

6. ATCS Simulation Model Validation

It is important to test the ATCS code for ns-2 to ensure that it provides accurate simulations. This section describes three validation procedures used to

test the simulation model: (i) running simulation data on an ATCS monitoring tool, (ii) comparing simulation data with laboratory data, and (iii) comparing simulation data with data from a mathematical model.

6.1 ATCSMon-Based Validation

The ATCSMon tool decodes ATCS radio traffic and interprets packets. It is similar to Ethereal, which is used in TCP/IP networks.

In order to test that the ATCS ns-2 simulation produces valid packets, support was added to save packet traces in the ATCSMon format. The simulated packets were compared with real packets to verify that they were similar in content and timing. The ATCSMon visualization feature helped confirm that the simulation data was "realistic."

6.2 Laboratory-Data-Based Validation

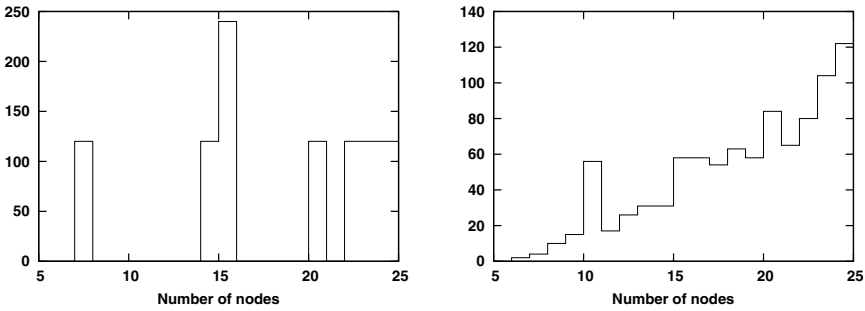
This validation study used test data from an ATCS laboratory maintained by Wabtec Corporation. The laboratory setup included several MCPs, one BCP and one FEP/CC; ATCSMon was used to capture data. An identical setup was created virtually using ns-2. The data was identical, except for a CRC code and three bytes used to hold session information. The virtual model was adjusted to handle these data fields.

Packet timing was also compared. In this case, the results were similar, but not exact. Rather than transmitting status data at exactly 60-second intervals, the MCPs would drift a few tenths of a second during each cycle. The difference is significant when large numbers of MCPs are employed. Packet collisions occur when two stations transmit at exactly the same time (every 60 seconds). However, fewer collisions occur when the 60-second cycle drifts a little, This difference is seen more clearly when data from the ns-2 model is compared with ATCS data generated using a mathematical model of ATCS throughput.

6.3 Mathematical-Model-Based Validation

The Sharif-Furman mathematical model of ATCS throughput [23] was used in the study. According to the Sharif-Furman model, the throughput should change smoothly depending on the number of MCP nodes that have traffic. The greater the number of MCPs, the greater the number of packet collisions, which reduces the effective throughput.

We tested this characteristic with multiple simulations of the ATCS ns-2 model. All the MCPs were set the same distance from the BCP, and support for "busy-bit" traffic control was turned off. Simulation runs were performed for networks with five MCPs up to 25 MCPs; each simulation modeled one hour of traffic. The simulations were run, summarized and graphed in one step using TCL scripts. The total computation time required to create twenty hours of simulated ATCS network traffic was less than five seconds.



(a) 60 second interval.

(b) 59-61 second interval.

Figure 2. Corrupted packets vs. nodes.

The first group of simulation results did not match the mathematical model. The main reason is that the Sharif-Furman model assumes that packets are scheduled according to a Poisson distribution (i.e., they arrive independently); however, ATCS traffic does not follow a Poisson distribution. Acknowledgement packets are sent upon packet arrival, and status information is usually sent once per minute. This results in a mean number of packets over a time interval that does not equal the variance, violating one of the criteria for a Poisson distribution. Evidence of this can be seen in Figure 2(a), where the number of packet collisions in one hour is plotted against the number of nodes. Instead of a smooth increase in packet collisions, the collisions jump in multiples of 120. Nevertheless, the Sharif-Furman model provides an upper bound for randomly-scheduled ATCS packets.

The simulations show that packet collisions occur when each node transmits status information once a minute and two nodes on the network are set to transmit at exactly the same time. Thus, two packets are marked as corrupted due to each collision. Because the time to retry is somewhat random, the second attempt at sending the packets usually works. However, as discussed earlier, laboratory tests show that most MCPs do not transmit exactly once per minute. Thus, if the simulation is changed to randomize the interval between 59 and 61 seconds, the results are notably different, as shown in Figure 2(b).

Sharif and Furman expressed their results in terms of the throughput that could be expected in a given system and provide a graph of the predicted throughput with 251 byte packets. They used the following equation to compute the throughput:

$$S = \frac{\bar{U}}{\bar{B} + \bar{I}} \tag{12}$$

where $\bar{U}$ is the average time spent on successful packets, $\bar{B}$ is the average busy time and $\bar{I}$ is average idle time (all in seconds).

On the other hand, we use the following equation to calculate the throughput for an ATCS model simulation:

$$S = \frac{l_{bits} \cdot n}{t \cdot r} \tag{13}$$

where l_{bits} is the packet length (bits), n is the number of packets successfully received, t is the time (seconds) and r is the data rate (bits per second).

The simulated environment used for validation incorporated ten MCP nodes and one BCP node. The results show that the ATCS simulator replicates the conditions imposed by Sharif-Furman. Figure 3(a) compares the Sharif-Furman model results with those obtained with the ATCS simulator. Each point in the ATCS simulation results represents an average of 20 runs, each 10 minutes long.

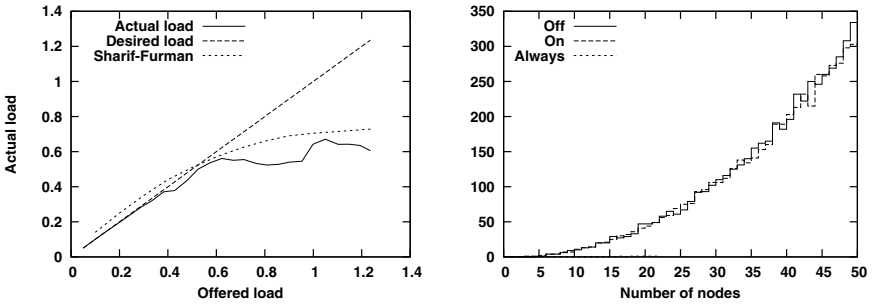
The system load was computed by dividing the bits sent per second divided by the capacity of the data line. In this case, a load of 1.0 implies that ten MCP nodes generate 1,200 bits to transmit every second on a 1200 baud line. Loads greater than 1.0 indicate more data is sent at a faster rate than the baud rate of data transmission. Simulated loads were increased in increments of 0.05 with busy-bits being transmitted. Note that the 251-byte packets used by Sharif-Furman were much larger than the ATCS packets, which were less than 40 bytes.

The Sharif-Furman results show that the predicted throughput is slightly larger than the data offered load for many of the lower load values. In other words, more data was received than was transmitted; but it is not clear from the Sharif-Furman paper what this extra data represents. Consequently, it is difficult to tell the point at which packets start getting dropped for the Sharif-Furman model. This is important because, in most real-time systems, networks must be configured so that dropped packets are very rare. In the case of the ATCS simulation, packets start getting dropped at a load of 0.2 for a packet length of 251 bytes.

In general, the simulation results are slightly lower than the results predicted by the Sharif-Furman model. This is expected because the Sharif-Furman model provides the "ideal" upper bound – due to the fact that packets are generated at random intervals. The ATCS simulation, on the other hand, generates packets at intervals that are mostly periodic in nature, increasing the chance of packet collisions. ATCS packets do not follow the Poisson distribution expected by the Sharif-Furman model; this is reflected in lower throughput for the simulated model.

7. Modeling Network Loads

The ns-2 ATCS simulator can also be used to model new situations. For example, support for ATCS traffic control can be set to one of three states: (i) BCP busy-bits can be turned off, (ii) BCP transmits only when it has traffic to send, or (iii) BCP always transmits. Figure 3(b) shows the average



(a) Predicted/simulated throughput.

(b) Corrupted packets vs. nodes.

Figure 3. Simulation results.

number of corrupted packets for simulation runs of one hour with the BCP set to each of the three possible states. For each node count in the graph, twenty simulations were run and the results averaged. Each MCP node transmits a status message over an interval between 59 and 61 seconds. The CC does not send any control messages to the MCPs. The only packets received by the MCPs are acknowledgement packets.

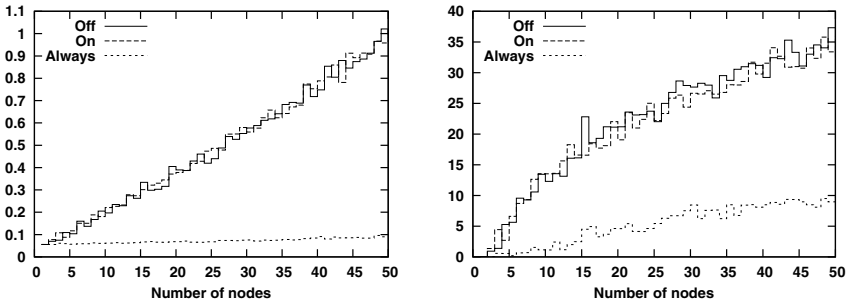
The results in Figure 3(b) show that always transmitting a busy-bit for traffic control significantly impacts the number of corrupt packets. Transmitting the busy-bit only when the BCP needs to transmit data does not significantly impact the number of corrupted packets. However, it should be noted that this is the worst-case scenario because the CC usually sends control messages to the MCPs on a periodic basis and this increases the number of busy-bits sent. The frequency with which control messages are sent depends on the scenario. It is easy to run simulations that model the different scenarios.

Figure 4(a) shows the average times for delivering ATCS packets (over 20 simulation runs). Packets are not counted unless they arrive. the average time increases as more nodes are added to a network. Most packets continue to arrive quickly, but packet collisions and retries drive up the average. Note that transmitting a busy-bit improves the average packet time fairly significantly.

Figure 4(b) shows the average maximum times for deliver packets (over 20 simulation runs). Knowing the expected maximum time for packet delivery is important in real-time applications. Note that this time increases quickly even when only a few nodes added to the system.

8. Conclusions

The ATCS ns-2 simulation system is a valuable tool for modeling ATCS networks and investigating their behavior in a virtual environment. By setting up a suite of ns-2 scripts, an engineer can simulate hundreds of possible scenarios



(a) Av. time to deliver packets.

(b) Av. max time to deliver packets.

Figure 4. Average packet times.

in the space of a few seconds. The simulation results are useful for understanding ATCS communication characteristics, identifying protocol strengths, weaknesses and vulnerabilities, and developing mitigation techniques. The fidelity of the ATCS simulation model is validated by test results using ATCSMon monitoring software, laboratory data and the Sharif-Furman mathematical model.

References

[1] Association of American Railroads, System Architecture: ATCS Specification 100 (Revision 4.0), Washington, DC (www.atcsmon.com/100_4_0.htm), 1995.

[2] G. Baker, Schoolboy hacks into city's tram system, *The Daily Telegraph*, January 11, 2008.

[3] M. Benke, On the Survivability of Transportation Systems, M.S. Thesis, Computer Science Department, University of Idaho, Moscow, Idaho, 2005.

[4] M. Benke, A. Abdel-Rahim, P. Oman and B. Johnson, Case study of a survivability analysis of a small intelligent transportation system, *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1944, pp. 98–106, 2006.

[5] S. Butapati, R. Kalidindi, M. Benke, A. Abdel-Rahim, P. Oman and B. Johnson, A case study of critical point identification in the physical and communication layers of a small urban transportation system, *International Journal of Critical Infrastructures*, vol. 2(4), pp. 319–330, 2006.

[6] P. Craven, A brief look at railroad communication vulnerabilities, *Proceedings of the Seventh International IEEE Conference on Intelligent Transportation Systems*, pp. 245–249, 2004.

[7] P. Craven, Security of RCL wireless railway communications, *Proceedings of the IEEE Conference on Control Applications*, pp. 711–715, 2005.

[8] P. Craven and S. Craven, Security of ATCS wireless railway communications, *Proceedings of the ASME/IEEE Joint Rail Conference*, pp. 227–238, 2005.

[9] P. Craven and S. Craven, Security of railway EOT systems, *Proceedings of the ASME/IEEE Joint Rail Conference*, pp. 199–204, 2005.

[10] K. Fall and K. Varadhan, The Network Simulator – ns-2 (nsnam.isi.edu/nsnam/index.php/Main_Page).

[11] Federal Railroad Administration, Positive train control overview, Department of Transportation, Washington, DC (www.fra.dot.gov/us/content/1265).

[12] E. Gamma, R. Helm, R. Johnson and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, Boston, Massachusetts, 1995.

[13] D. Green and A. Obaidat, An accurate line of sight propagation performance model for ad-hoc 802.11 wireless LAN (WLAN) devices, *Proceedings of the IEEE International Conference on Communications*, vol. 5, pp. 3424–3428, 2002.

[14] D. Houy, ATCS monitor for Windows (www.atcsmon.com).

[15] G. Hufford, The ITS Irregular Terrain Model (version 1.2.2), Institute for Telecommunication Sciences, National Telecommunications and Information Administration, Boulder, Colorado (flattop.its.bldrdoc.gov/itm/itm_alg.pdf), 1999.

[16] Institute for Telecommunication Sciences, C++ code for Longley-Rice propagation, National Telecommunications and Information Administration, Boulder, Colorado (flattop.its.bldrdoc.gov/itm.html), 2007.

[17] A. Longley and P. Rice, Prediction of Tropospheric Radio Transmission Loss over Irregular Terrain: A Computer Method, ESSA Technical Report ERL 79-ITS 67, Institute for Telecommunication Sciences, National Telecommunications and Information Administration, Boulder, Colorado (www.its.bldrdoc.gov/pub/essa/essa_erl_79-its_67), 1968.

[18] J. Magliacane, Splat! RF signal propagation, loss and terrain analysis tool (www.qsl.net/kd2bd/splat.html).

[19] N. Mead, R. Ellison, R. Linger, T. Longstaff and J. McHugh, Survivable Network Analysis Method, Technical Report CMU/SEI-2000-TR-013, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, 2000.

[20] P. Rice, A. Longley, K. Norton and A. Barsis, Transmission Loss Prediction for Tropospheric Communication Circuits, Volumes I and II, Technical Note 101, Institute for Telecommunication Sciences, National Telecommunications and Information Administration, Boulder, Colorado (www.its.bldrdoc.gov/pub/ntia-rpt/tn101), 1967.

[21] L. Ritter, J. Barrett and R. Wilson, *Securing Global Transportation Networks*, McGraw-Hill, New York, 2006.

[22] Securities and Exchange Commission, Norfolk Southern Corporation: Form 10-K (Fiscal year ending December 31, 2003), Washington, DC (www.secinfo.com/dwDMq.11q.htm), 2004.

[23] H. Sharif and E. Furman, Analytical model for ATCS inbound RF channel throughput, *Proceedings of the Forty-First IEEE Vehicular Technology Conference*, pp. 885–892, 1991.

[24] D. Smith, *Digital Transmission Systems*, Kluwer Academic Publishers, Norwell, Massachusetts, 2003.

[25] T. Smith, The impact of highway infrastructure on economic performance, *Public Roads Magazine*, vol. 57(4), pp. 8–14, 1994.

[26] The White House, The National Strategy for the Physical Protection of Critical Infrastructures and Key Assets, Washington, DC (www.whitehouse.gov/pcipb/physical_strategy.pdf), 2003.

[27] United States Geological Survey, 1:250,000-scale digital elevation model (DEM), Reston, Virginia (edcftp.cr.usgs.gov/pub/data).

[28] J. Waite, M. Benke, N. Nguyen, M. Phillips, S. Melton, P. Oman, A. Abdel-Rahim and B. Johnson, A combined approach to ITS vulnerability and survivability analyses, *Proceedings of the Seventh IEEE Conference on Intelligent Transportation Systems*, pp. 262–267, 2004.

[29] D. Wetherall, OTcl (otcl-tclcl.sourceforge.net/otcl).

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