SECURE BROADCAST FOR VEHICULAR COMMUNICATIONS

by

Anthony Melaragno
A Dissertation
Submitted to the
Graduate Faculty
of
George Mason University
In Partial fulfillment of
The Requirements for the Degree
of
Doctor of Philosophy
Information Technology

Committee:

_____	Dr. Duminda Wijesekera, Dissertation Director
_____	Dr. Paulo Costa, Committee Member
_____	Dr. Stephen Nash, Committee Member
_____	Dr. James Jones, Committee Member
_____	Dr. Stephen Nash, Senior Associate Dean
_____	Dr. Kenneth S. Ball, Dean, Volgenau School of Engineering

Date: _____ Summer 2016
George Mason University
Fairfax, VA

Secure Broadcast for Vehicular Communications

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

By

Anthony Melaragno
Master of Science
George Mason University, 2004
Bachelor of Science
The Pennsylvania State University, 1997

Director: Dr. Duminda Wijesekera, Professor
Department of Information Technology

Summer 2016
George Mason University
Fairfax, VA

Dedication

This dissertation is dedicated to my wife, Kim, daughter, Shirley, and my parents, Emidio and Maria. My wife and daughter have provided me with the love and support that allowed me the time to complete my degree. Additionally, my parents taught me the value of a strong work ethic and the value of education.

Acknowledgments

The accumulation of academic studies as well as my professional experiences have led to this work. Knowledge gained through course work and interactions with colleagues (both academically and professionally) have provided the support needed to finish this work.

I am grateful to my adviser, Dr. Duminda Wijesekera. He represents kindness, knowledge, and finally friendship that is beyond an advisory role. I am truly grateful to all of the weekends and late nights he has helped me. He and I have collaborated and have theorized and developed a road map for future technologies to safe guard infrastructure as well as transportation. I hope that our paths continue to cross in the future.

After completing my Masters of Science in Telecommunications, Dr. Jeremy Allnutt believed that I had the abilities to complete a PhD, for his confidence and encouragement, I am grateful.

I thank all of the members of my committee for their time and support. Dr. Paulo Costa has provided me additional guidance, both personal and academic, and I feel indebted to him as a friend and colleague. Dr. Nash and Dr. Jones provided insight and challenged me.

I owe a special thanks to the academic support staff at George Mason, including Lisa Nolder, then later, Jennifer Skorzawski-Ross, Susan Mueller, and Joyce Rose who provided me tireless guidance needed to navigate through the procedures and policies at George Mason University.

I would like to thank Mr. Mark Segal from L-3 Communications for his support, attending my PhD presentations, and showing me that there are caring people behind corporations.

I would like to thank the Department of Homeland Security for partially funding some of this research.

I would like to personally thank Mr. David Blackmore, and Dr. Mark Hartong from the Department of Transportation (DoT) Federal Railway Association (FRA). I appreciate Dr. Hartong's general curiosity and am inspired by his willingness to "geek" out" when discussing this technology.

The development of this technology did not occur in a bubble. There were many lively debates with fellow graduate students which led to the evolution of the technology. I and other graduate students have leveraged each other's research to theorize and build different aspects of the complex transportation grid. Specifically, I would like to thank Damindra Bandara for assisting numerous times in running tests and experiments.

I bent the ear of many friends and family who have assisted me throughout this process. Mathew Schuck listened to my excitement concerning the technology that I was developing, how I praised the merits of LaTeX, and how the technology that I was developing would become a "game changer" for the world. I am grateful to my brothers Mike and Fernando who discussed the world in terms of newer technologies and had a general curiosity of what I was attempting to do. My parents, Emidio and Maria, emulate what is best about immigrants who come to this country. I admire their will to always strive for success through

hard-work, dedication, belief in education and love for this country. Their example is what drives me and helps keep me humble.

Last but not least, I am thankful for the love and support I have received from Kim and Shirley. From the time Kim and I met as undergrads at The Pennsylvania State University, she knew it was my dream to get a PhD and has been my biggest champion, encouraging me to continue my degree. I love my wife and she has supported every aspect of the degree from providing me time to work on the degree while holding down the fort at home, to listening to my ideas, and to helping proofread my many papers, presentations, and dissertation. Shirley, I love you, and can not wait to spend more time with you.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ACK Acknowledgment

ACSESII Advanced Civil Speed Enforcement System II

AIS Automatic Identification System

ARA American Railway Association

BECO Back End Control Office

BER Bit Error Rate

CBTC Communication Based Train Control

C2 Command and Control

CE Cognitive Engine

CPU Central Processing Unit

CRC Cyclical Redundancy Check

DoS Denial of Service

DoT Department of Transportation

ERTMS European Traffic Management System

FCC Federal Communications Commission

FDR Failure Divergence Refinement

FILO First In Last Out

FRA Federal Railway Association

GAO General Accounting Office

GMT Greenwich Mean Time

GPS Global Positioning System

HMAC Hash Message Authentication Code

I2V Infrastructure to Vehicle

IDS Intrusion Detection System

I-ETMS Interoperable Electronic Train Management System

IV Initialization Vector

MAC Message Authentication Code

MGRS Military Grid Reference System

mph Miles Per Hour

NAK Not an Acknowledgment

NIST National Institute of Standards and Technology

NTP Network Time Protocol

OSI Open Systems Interconnection

OTAR Over The Air Rekey

PKI Public Key Infrastructure

pps pulse per second

PRF Pseudo Random Function

PTC Positive Train Controller

QPSK Quadrature Phase Shift Keying

RARE Radio and Radar Laboratory

RRIDS Rail Radio Intrusion Detection System

RRIDSDB Rail Radio Intrusion Detection System Database

RF Radio Frequency

SA Situational Awareness

SNR Signal to Noise Ratio

SDR Software Defined Radio

SHA Secure Hash Algorithm

TESLA Time Efficient Stream Loss-tolerant Authentication

USD United States Dollars

V2I Vehicle to Infrastructure

V2V Vehicle to Vehicle

WIU Wayside Interface Unit

Abstract

SECURE BROADCAST FOR VEHICULAR COMMUNICATIONS

Anthony Melaragno, PhD

George Mason University, 2016

Dissertation Director: Dr. Duminda Wijesekera

Intelligent transportation systems use radio beacons to broadcast infrastructure information that is used for safe navigation. Authenticity and Integrity of such messages are essential to ensure that recipients can safely make their safe navigational decisions. The emerging area of Communication Based Train Control (CBTC) is one system as applied to train communications. Trains rely on accurate and verified message broadcasts for vital information such as status of tracks, switches, highway crossings, or broken rail detectors, etc. This vital information allows trains to adjust speed and apply brakes ensuring their safe navigation. The problem my dissertation addresses is that due to the minimal bandwidth and the limited data integrity frames allocated to CBTC communications there exists potential attack vectors such as message replay, forgery, and corruption attacks. The aforementioned communication attack vectors will attempt at a minimum to disrupt communications and in the worst case, derail locomotives by providing false status information to the trains and underlying infrastructure. I address the attack vectors by introducing a cryptographic schema to ensure that data integrity is maintained throughout the communications process. I enhance security by developing a custom software intrusion detection system called RRIDS.

The combination of the cryptographic schema and RRIDS ensures that infrastructure communication is maintained for the safety of emerging rail communication infrastructures. My solution provides a prototype implementation that I experimentally validate. Additionally, RRIDS is verified using attacker tests which validate its contribution by differentiating and categorizing received radio signals from both legitimate and attacker radio sources as well as an alert mechanism.

Methods developed in this dissertation can be extended for other forms of intelligent transportation systems such as Vehicle to Infrastructure (V2I), Vehicle to Vehicle (V2V), etc., that are being advocated by the automobile community. This dissertation designs and prototypes the cryptographic solution and the Intrusion Detection System (IDS) system for CBTC systems. The effectiveness of the IDS system and the cryptographic schema is experimentally validated.

Chapter 1: Introduction

Vehicular navigation and control are becoming automated and autonomous with the expectation that vehicles and supporting infrastructure will have decision making intelligence for *safe* navigation (where safety is defined by a collection of application specific requirements), which includes accident avoidance. Such navigation will require communication between vehicles and infrastructure, as well as in some scenarios among vehicles. The underlying assumption is that communications are *secure* where the security depends on the navigational requirements as well as the ability to detect, mitigate, and avoid mal-actors. In case of communications from the supporting infrastructure to vehicle, *secure* broadcasting is needed to ensure the fulfillment of *safe* navigation with *security* as supporting requirements. The decision making intelligence requires the ability to securely broadcast messages and to detect if malicious actors are actively undermining the transportation communication system. The work that I present in this dissertation uses the cryptographic schema that I developed to ensure data integrity of vehicular and infrastructure message broadcast, and a specialized IDS in detecting malicious actors. The case study used to validate both the cryptographic schema and the IDS, referred to as RRIDS, is envisioned to apply to rail systems. However, my contributions can be extended to other transportation systems and infrastructures supporting aircrafts, ships, and automobiles .

A simple but useful example of public broadcasts controlling vehicular navigation is traffic signals. If a control system using *color light based* traffic signals were to be moved to an equivalent Radio Frequency (RF) communications, all vehicles would rely on the RF signal information for safe navigation decisions. All motorists would receive the information in a timely manner and act on the received RF based traffic signal. As this example shows, the broadcast traffic signals has no need of confidentiality requirements because all motorists depend on the color to base their navigational decisions when approaching an intersection.

In order to do so, the RF signals would preserve the semantics of light's color and the color itself (syntax of the light). In the realm of train control, the motorist is the train engineer, monitoring their in *cab signaling*. The signal indication represents the semantics of the light and the signal aspect represents the syntax. Consequently, the integrity of the RF signals must be preserved in order to ensure that the semantics of the situation (signal indication) is preserved at the recipient vehicles. Secondly, traffic signals are controlled by an authoritative source, such as a state department of transportation or city authorities. An unauthorized entity operating traffic signals would compromise vehicular safety and security. Therefore, authenticating the broadcast source of the signal matters. In the RF radio world, an attacker can over power the legitimate radio broadcast, transmit a forged message, thereby impersonating the authoritative broadcaster, thereby requiring the need of strong data integrity, data authentication, and intrusion detection for vehicular infrastructure security.

The color light based notification system is analogous to most forms of RF signaling in transportation, for example: surface transport modes, naval signaling, Automatic Identification System (AIS) signaling [4], and NextGen air traffic control [5]. In the previous example, the color of the signal conveys traffic condition semantics and since the Command and Control (C2) signals are bandwidth constrained and Situational Awareness (SA) is needed in near real time those constraints prevent the manipulation of data. Currently, if data confidentiality were available it would negatively affect the performance of the system to convey information to motorists because data encryption and decryption are processor intensive actions and would impact the ability to relay the information in a timely manner. Hence, encrypting the signal is impractical as well as potentially dangerous. Nevertheless, vehicles need the awareness that the radio signal originated from an authoritative source and not a rouge radio by the wayside. In the previously mentioned traffic signaling example, common to all transport modes is that the *beacons* have a data integrity requirement to ensure that the transmitted signal has not been manipulated by a mal-actor without the need of encrypting the information. The technology developed in my dissertation enables

communication devices to check the data integrity of a beacon's broadcast and provides a specialized intrusion detection system that detects attacks against such a system. My dissertation uses emerging rail infrastructure as a case study to address these concerns. The architecture and functional requirements of this case study are described in Section 1.1.

1.1 Communication Based Train Control and the Positive Train Control Network

Controlling trains has a long and developing history beginning in 1804 by Richard Trevithick. Control systems for trains, such as the European Traffic Management System (ERTMS) (Europe), Shinkansen (Japan), Advanced Civil Speed Enforcement System II (ACSESII) (Amtrak in USA), and Interoperable Electronic Train Management System (I-ETMS) (USA) use CBTC. Emerging Software Defined Radio (SDR) technologies are gaining acceptance and are beginning to become integrated into the rail infrastructure communication framework. The SDR communicates messages to control train movements and relay advisories that the controller takes into account to assist in navigation. The concept of a signaling network, mentioned previously, provides train movement messages in the form of signal aspects which informs the locomotive of speed limits, etc. Once received, the locomotive performs appropriate actions to ensure safe navigation. A separate network conveys environmental conditions, such as track status, flood, snow levels, and other potential issues such as objects that block the tracks (such as trains in the block, malfunctioning rail gates, switch positions, etc.)

CBTC networks that utilize SDR technologies may become subject to cyber security attacks. These attacks may entail sending signal aspect information which would cause a train to take unsafe actions such as proceeding through a region when the train is supposed to stop. Additionally, an attacker could send incorrect track status signals or jam signals not allowing critical messages to reach the locomotives control system. These attacks would compromise safe navigation of trains and possibly lead to accidents or endanger railroad

workers. As a solution, I developed RRIDS with the objective of detecting and deterring cyber attacks such as command replays, fabricating messages and message corruption attacks. RRIDS is a rail command specific IDS designed for locomotive and beacon communications security. RRIDS detects intrusions in near real time by leveraging underlying cryptographic attributes in the processing of legitimate communication devices.

The sample use case addressed in this dissertation pertains to the PTC [6] network to improve locomotive infrastructure security. This use case is illustrated in Figure 1.1 and is described as follows.

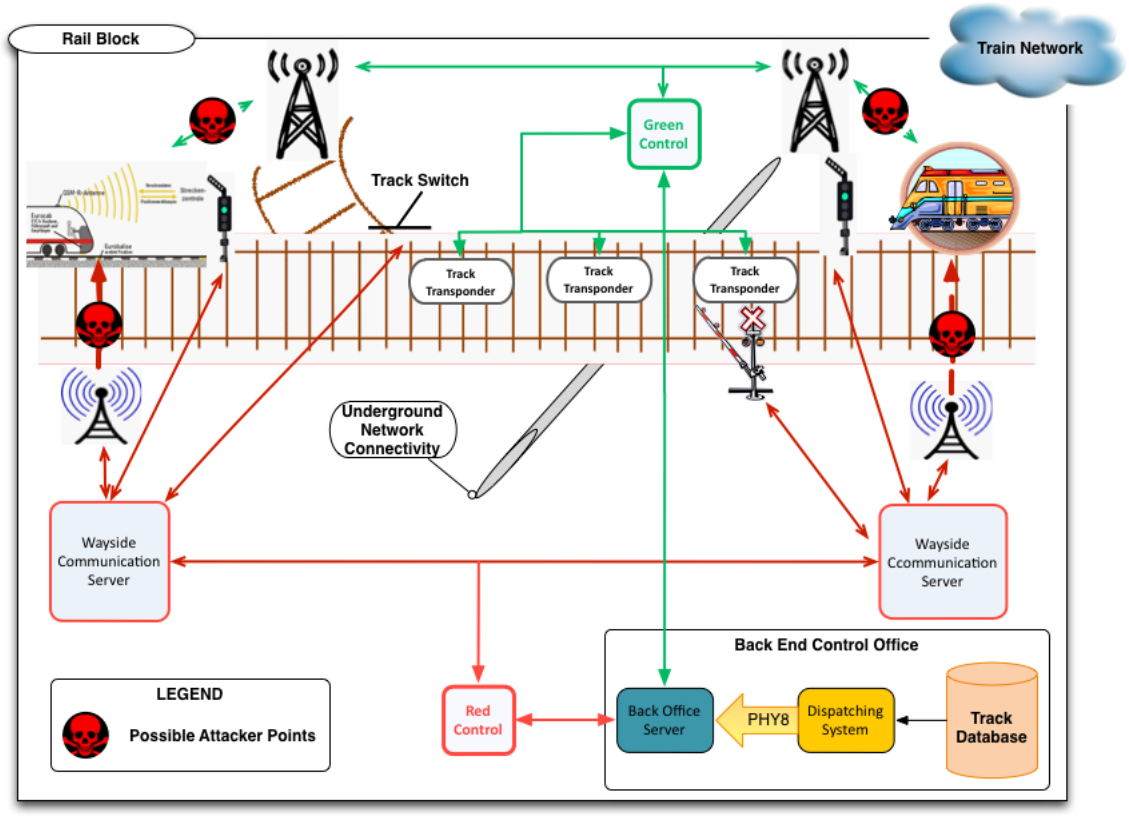


Figure 1.1: Possible Attack Vectors

The attacker use cases that I address can be envisioned as preventing possible attack vectors shown in Figure 1.1. The attacker, represented by the red skull and cross bones, performs the following actions: overpowers transmitters by preventing messages from being

received by the locomotive, forges messages and transmits the forged messages or copies and replays messages to the locomotive. My work aids in detecting and preventing such attacks.

Figure 1.3 pictorially represents the components used in proposed PTC (PTC)[6] systems that are to become operational by 2018 in the United States. Figure 1.3 illustrates a locomotive entering a rail block. Each infrastructural component illustrated in Figure 1.3 contributes to the overall safety of the system by providing information to the locomotive's control unit and the engineer. Figure 1.3[1] illustrates the communication equipment used for the C2 locomotive network notionally. The communication architecture of the PTC network comprises two distinct networks the signaling network and the WIU network. The signaling network is responsible for providing track warrants (the authorities for the train to enter a block of track), speed limits and other instructions to be used in safely navigating on the track segment (such as when to expect the next instructions). The WIU network provides radio beacons that broadcast rail infrastructural information such as the status of a switch, level-crossing, broken rail, flat wheels, hot wheels, snow, frozen rail, floods on rail and other track based information so that an incoming train can automatically initiate a fail safe operation to avoid the semantics of beacon message contents.

Locomotives are granted entry or exit to segments of track commonly referred to as (static) blocks. Prior to entering a block, a locomotive must receive a track warrant from the signaling network. The WIU beacons collect and relay information concerning track, switch alignment, and environmental conditions such as snow, flooding, etc. The beacon obtains this information from track mounted sensors, referred to as wayside devices, that informs if it is safe for the train to travel on the rail track. The wayside devices are physically connected to radios that broadcast the status information to oncoming trains. In addition, radios that transmit messages are connected to communications servers that reside in a *back office*.

The signaling networks for trains act like a combination of traffic lights and speed limit

[1]Provided by Bandara, D.

restrictions (currently on static boards on motorways such as I66 in Virginia) and construction related traffic diversions, advisories, lane closures and orange cones. The WIU network provides road advisories that communicate road conditions, including road closures which are done by local police stations, but emerging smart highways will provide this information using the Infrastructure to Vehicle (I2V) infrastructures for smart cars. Figure 1.2 [2] shows a traditional locomotive signal tower that will be replaced by the combination of signaling and WIU networks.

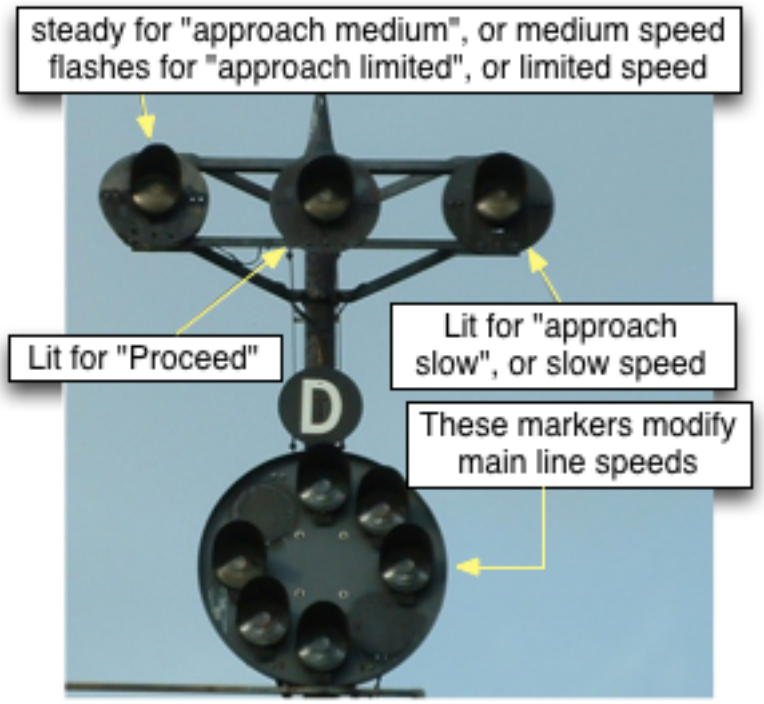


Figure 1.2: CSX Rail Network Signal for locomotive.

The light combinations shown in Figure 1.2 convey specific instructions to the railway engineer. The top three lights indicate the approach within the block and its speed within the main line. The six markers shown in the figure indicate the warrant to enter the block, the track occupancy, approach, and speed. Therefore, an approaching locomotive engineer knows if another locomotive is currently occupying the block, how to proceed into the

[2]Provided by: http://www.railroadsignals.us/signals/cpl/

block, and its speed. Appendix A describes Rail Signal Aspect for a complete illustration of signaling for the B&O Signal Aspect.

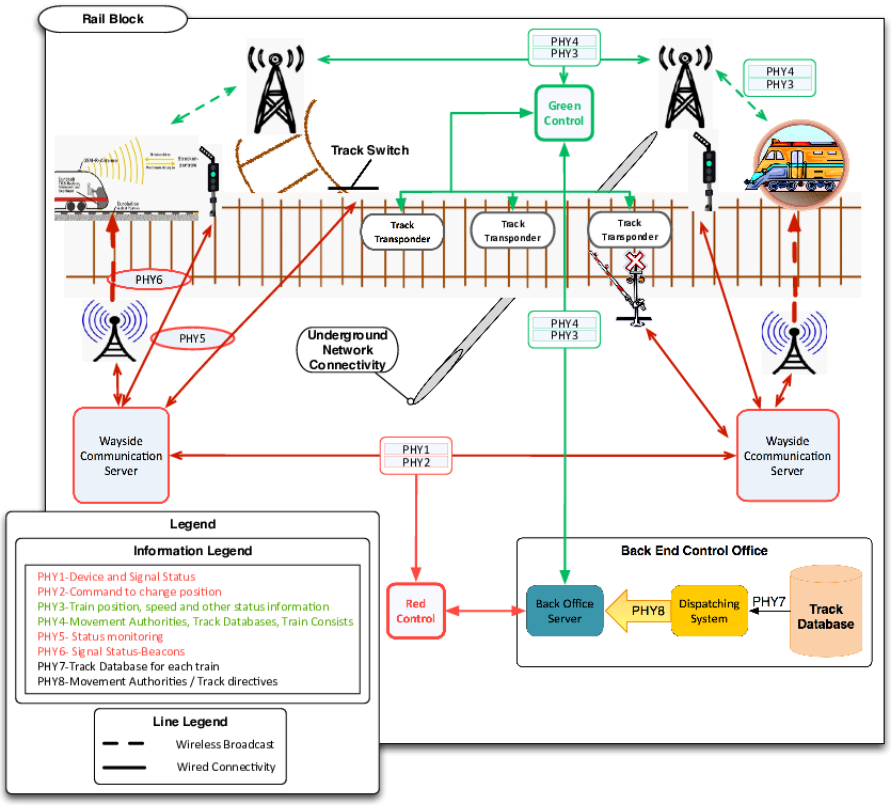


Figure 1.3: Overview Figure of WIU, PTC, Rail Network within a Block [1]

As shown in Figure 1.3, in the emerging PTC architecture, the Back End Control Office (BECO) communicates and coordinates the train's ingress into the block, movement, and egress out of the block. The coordination of movement of locomotives that operate within tracks is determined by the rail company that owns it and provide trackage rights (the permission to pass through) to trains owned by other companies, mandated by the Interstate Commerce Act. The movement coordination itself is governed by (mostly an automated) train scheduler that communicates with the BECO. The combination of the train scheduler and the BECO provides navigational guidance to trains from their own company and right of way to trains owned by other companies as mandated by the Interstate Commerce Act.

As a summary, the BECO communicates all decisions made by an automated train scheduler to the appropriate radios that in turn conveys these messages to appropriate locomotives. In Germany [7] two trains collided because of a scheduling accident. Essentially, scheduling, if used properly, would prevent two locomotives from occupying the same track at the same time.

When a train requests entry into a block, it generates a request, and uses its own PTC radio to transmit the request to radios in the signaling network (part of the infrastructure). The signaling network then relays the information to the back office which relays the information to the scheduler. The scheduler uses requests and a master track database and details of the requesting trains, such as their contents and consist (the physical configuration of the train, such as the number of engines, their distribution, number of compartments and their contents) and eventual destinations etc. The scheduler uses the aforementioned information and those of other trains already on the tracks to create a schedule for all trains that request entry. The scheduler then decides which train may enter the block and communicates that decision to the BECO. Then the back office server determines how to transmit the message to the requesting train using the signaling radio network. I will now decompose Figure 1.3 and describe it in terms of its functionality.

- **The Signaling Network**: The signaling network provides track warrants, speed limits and temporary travel restrictions to trains. The PTC controller, on the locomotive's lead engine, assesses if there is a potential of collision, and if the risk is high, it will automatically stop or slow down the train by directly communicating that requirement to the train controller and multiple break controllers. The information given out by the signaling network is used to prevent train collisions, provide inter-train distances and control rail traffic congestion.

- **WIU Network**: The WIU network conveys the wayside device status to incoming trains. Additionally, the WIU provides track condition and track advisory information to the train.

- **The BECO**: The BECO provides the main communication point between the scheduler and the two networks, shown in Figure 1.3, that directly communicates with the trains.

- **The PTC Controller**: The PTC controller on locomotives, referred to as the onboard unit, takes all information from WIU transmissions, signaling networks, and informs the locomotive operator of the mandated signals. If the operator does not react, the PTC controller directly instructs the train controller and brake controllers. Trains have multiple braking systems for redundancy that operate in multiple operational modes that directly follow the signaled operating mandate. Some example mandates are: to apply penalty braking in order to prevent signals passed at danger (i.e. stop the train prior to exceeding authority to move) or to enforce speed limits.

1.2 Contributions

As described earlier, the WIU network has to provide message integrity and authenticity but not confidentiality for messages conveyed to oncoming trains. The traditional way to ensure message integrity is to provide a keyed-Hash Message Authentication Code (HMAC) based on an initial seed that is known to the creator and verified by the receiver of the message. If the creator of the message is a beacon with a high frequency (say of 10 messages per second) that uses the same salt, the beacon itself provides many sample HMAC messages for potential attackers. Conversely, if a small number of messages use the same salt, then an attacker may not have multiple samples to infer the salt, or by the time the attacker derives the salt, the beacon would have started using another salt. Providing such a capability is my primary suggested solution, the details of which constitute a significant part of my dissertation.

The details of applying a solution to this problem in the PTC WIU network poses many challenges, and I address them as follows:

The first is that of creating as many keys for each broadcast. This solution has been

addressed in the past with limitations. The first such solution is Lamport's password schema that creates N passwords for any given integer N with the constraint that it is easy to create previous passwords from the current one, but not any passwords to be used in the future. Perrig and Tygar [2] enhanced this schema to a protocol called TESLA that provides data integrity and authentication for communication between bandwidth and resource constrained devices.

The main limitation in directly using Tesla for WIU beacon integrity is that the beacons do not have a pre-computed limit on transmissions. A second issue is the large number of beacons that exists in any railroad domain, and the challenge of providing the same service for all of them. A third challenge is to design a schema to share the multiple keys or key generation schema that work for every beacon belonging to one railroad. As a solution, I extended TESLA to generate much longer key sequences. Secondly, I also propose a key generation schema that would provide keying seeds to the whole collection of WIUs under the control of one railroad. In addition to changing salts, my algorithm will also randomly rotate integrity algorithms and provides a mechanism to detect communication traffic tampering.

Given that the proposed work contributes multiple services, I included them in a *cryptographic cognitive engine*. The cryptographic cognitive engine verifies data integrity of the received signal, detects potential jamming attacks. Thus my cryptographic cognitive engine provides cryptographic services required for the PTC radios as a module within the SDR. In order for my engine to report cryptographic violations, the cryptographic engine verifies the message content. If the content is consistent and the CRC is correct the engine then checks the integrity value. If the integrity value is improper then the system reports its findings to a higher-level cognitive engine, that I refer to as the master cognitive engine, for determination. The master cognitive engine then determines if the system is under attack or a false positive is reported. The consequences summarized in Table 1.1 are the intended objectives of the adversary in which it would lead to the partial or to the full compromise of the locomotive.

The master cognitive engine is being developed at the Radio and Radar Laboratory (RARE) lab at the George Mason University as a project funded by the FRA. The overall objective of the FRA project, which is beyond the scope of this work, is to provide a customized cognitive radio network to provide the safety and the security of an as yet to be completed PTC systems as a forward looking research prototype.

In my architecture, my cryptographic cognitive engine provides three innovative capabilities, which are the cryptographic cognitive engine (which changes hashing algorithms), salts, and a V2I based IDS. The layered architecture that I designed is a scalable system which includes the master cognitive engine that uses results from the IDS in detecting possible intruders, prevention of possible broadcast attacks, and alerting authorities.

Table 1.1: Notional Attack Vector Description

Attack Category	**Consequence**
Direct Attack: Cryptographic Compromise	
Message Forging	Complete Compromise
Reseeding	Attacker Reseeds with invalid seeds
Direct Attack: Denial of Service	
Random Message	Resource Overuse
Spectrum Jamming	No reception of signal
Side Channel	
GPS Jamming	Loss of clock synchronization
PTC Network	Invalid Signatures

1.3 Results

I have designed and implemented the cryptographic cognitive engine, the intrusion detection system, and the communications architecture to interface to software defined radios. I designed the overall architecture emphasizing scalability which includes a middleware layer, a cryptographic and IDS database with scalability and network-ability in mind. My implementation and the prototype testbed for experimentation utilizes four Ettus N210 SDR from George Mason University's RARE laboratory with partially implemented WIU [8]

protocol. My SDR integrates a GNU Radio software stack [9] with REDIS [10] to manage communications intelligently and an underlying MySQL database for cryptographic algorithm and salt persistence. The communication system sends wireless PTC messages to and from the radios. The following contributions are proposed for my dissertation:

- Develop a cryptographic cognitive engine with the cryptographic system to evaluate WIU messages
- Enhance TESLA to provide beaconing security for the WIU network broadcast protocol
- Create a specialized Intrusion Detection System for Rail
- Experimentally validate the effectiveness of the IDS system

1.4 Dissertation Organization

The dissertation is organized as follows:

- Chapter 2 Thesis: formulates the thesis problem and its proposed solution defended in my dissertation.
- Chapter 3 Related Work: describes the relevant work.
- Chapter 4 Design: describes the architecture and design supporting the thesis and the hypothesis experimentally validated in my dissertation.
- Chapter 5 Implementation: describes the software implementation.
- Chapter 6 Experimental Validation: describes the hypothesis testing and how they satisfy the hypotheses statements.
- Chapter 7 Conclusion: concluding statements

Chapter 2: Thesis

In the introduction, I described the need for specialized communications for V2I systems. The thesis problem that this work addresses applies to a use case for communications integrity for railway signaling and is as follows:

> **Thesis Problem Statement**
>
> *It is possible to enhance the currently published designs for communications RF radio networks to provide data integrity for infrastructural broadcasts and requests originating from beacons and locomotives in an environment that are susceptible to RF noise, radio clock skews, message forgery attacks, replay attacks, cryptographic compromises, and jamming without changing the packet formats and without compromising the real time communication and control safety requirements.*

My preliminary work showed that the WIU network can be compromised because of insufficient bandwidth and weakened Secure Hash Algorithm (SHA) algorithms [11–13]. These weaknesses would lead to attackers impersonating the PTC infrastructure components shown in Figure 1.3. Given this realization, I substantiate my thesis which solves the cryptographic weaknesses and provides an IDS designed to capture intrusions. I validate the effectiveness of my IDS experimentally. My contributions are experimentally validated and these validations are stated in the form of validating claims (stated as hypothesis) using a radio testbed.

Contributions:

1. **Specialized Cryptographic Derivation**: I created a cryptographic hash algorithm, salt derivation technique, and a supporting architecture. I use a cryptographic bootstrap process that takes two seed values in which one is used to derive hash algorithm's

selection and the other seed used to derive salts. My technique differs from Perrig's [2] in two fundamental ways; first, my cryptographic algorithm used to verify the hash is not held constant and secondly, the salt used to validate the message is never transmitted over the air, but derived at both ends. The approach minimizes potential leakage of information to potential threats.

2. **Cryptographic Cognitive Engine**: I created a cryptographic cognitive engine to provide cryptographic services. I experimentally measured the resource utilization and performance overhead and quantified the ability to withstand selected attacks and to detect attacks on my system using my IDS.

3. **Specialized IDS for Rail**: I created a specialized IDS called RRIDS which evaluates the integrity and the authenticity of received messages. The messages that are deemed to not satisfy the security objectives are categorized are stored in a database for later forensics analysis. In addition, a visual alert is generated for providing an immediate situational awareness.

The above contributions are evaluated by addressing the following 3 hypotheses under the conditions of computer clock synchronization using a Network Time Protocol (NTP) timing source, and a Global Positioning System (GPS) timing source using a 1 pulse per second (pps) reference signal for local radio oscillator synchronization. The hypotheses are validated under conditions of non timing clock skews, timing clock skews in environments with radio noise, and an active attacker:

Hypothesis 1 (**Cryptographic Security**)**.** *When communicating devices with self-derived cryptographic material do not reveal cryptographic details, such as salt or key disclosure time [2], it lessens the capability of the attacker to effectively create attacks such as message replay and message forgery in the presence known amount of fixed radio noise.*

Hypothesis 2 (**Cryptographic Performance**)**.** *Derivation and utilization of self-derived cryptographic material within the communicating device will not affect system performance*

for deriving, selecting, and calculating integrity values during communications in the presence of selected amounts of constant radio noise. Consequently my system does not violate time sensitive safety requirements of PTC.

Hypothesis 3 (**Intrusion Detection System**). *RRIDS provides controlling authorities information that allow them to act against detected attackers to enhance safety and security in the presence of selected amounts of constant radio noise.*

The hypotheses are validated using the results of a series of experiments shown in Table 2.1. Table 2.1 points to sections of my dissertation that describe the capabilities, contributions made, experiments run, and my experimental results.

Table 2.1: Experimental Validation

	Hypotheses and Experimental Validation Sections		
Contributions	**Cryptographic Security**	**Cryptographic Performance**	**Intrusion Detection System**
Specialized Cryptographic Derivation	Section 6.3	Section 6.3	Section 6.3 Section 6.4
Cryptographic Cognitive Engine	Section 6.3 Section 6.4	Section 6.3	Section 6.3 Section 6.4
Intrusion Detection	Section 6.3 Section 6.4	Section 6.3	Section 6.3 Section 6.4

Chapter 3: Related Work

As shown in Figure 1.3, rail infrastructure consists of both wired and wireless communication components. Hartong's work [14] described an infrastructure to support locomotive scheduling across block boundaries between railroads. Additionally, [14, 15] introduced the concept of a RF beaconing network to control locomotive speed and warrants into a rail block. [14, 15] was geared to improve rail safety through introducing the PTC network as described earlier as well as in Figure 1.3.

In [16] Ashwin Amanna, etal. introduced six SDR type of requirements:

1. **Ability to operate in a noisy environment**: This requirement is developed to mitigate noise by providing situational awareness which is used to allow the SDR to mitigate noise contributors such as other radios, natural sources etc.

2. **Ability to operate in the presence of jamming signal**: If intentional jamming occurs the objective of the requirement is to take actions and take a course of new actions.

3. **Ability to improve link performance**: In the case of degradation of communications performance, the cognitive radio will then take actions to improve the quality of communications by changing the tunable parameters such as power, channel selection, and modulation.

4. **Maintaining Connectivity**: If connectivity becomes an issue then the SDR will then take actions to react and make changes to it's parameters.

5. **Ability to inter-operate with another company's infrastructure**: Since the rail lines are owned by independent private entities interoperability is essential. The PTC technologies must be able to inter-operate within the 220 MHz spectrum.

6. **Conformance to Policy**: The SDR and resultant Cognitive Engine (CE) must conform to Federal Communications Commission (FCC) constraints with regards to communications a policy engine is necessary. The introduction of the a policy engine is needed to ensure that laws are not violated.

The work from [16] led to the development of the SDR and CE for rail to manage spectrum. A significant problem that is being addressed in PTC networks is spectrum availability. In [3, 3, 17–20] developed methods to dynamically manage spectrum allocated for the PTC infrastructure. In [3] we discuss an architecture that provides enhancements in both spectrum management and security of the PTC networks. The cognitive radio network proposed in [19] illustrates the advantage of using a cognitive radio network for spectrum management. The cognitive engine designed is responsible for ensuring spectrum availability, communications reliability, operational safety and security in the PTC network. Periodic Signal to Noise Ratio (SNR), and Bit Error Rate (BER) measurements are made and the cognitive engine uses those measurements to adjust to the RF environment. In [3, 3, 17–20] the authors attempt to solve the problem of spectrum availability and spectrum management but the authors do not attempt to detect and prevent active attacks. In [20], three constraints were considered in evaluating spectrum requirements for the PTC network: Capacity, Power, and Frequency Interference.

1. **Capacity**: Represents the maximum number of trains that can be accommodated at a control point.

2. **Power**: Represents the geographical distance between the locomotive and the radio so that communications is maintained, and additionally, provides a communications hand-off mechanism between the past train location and the next radio.

3. **Frequency Interference**: Minimization of co-channel and adjacent channel interference between the WIU and Signaling Network.

The aforementioned constraints described in [20] provides the underlying requirements to address limitations in bandwidth availability in a noisy RF environment. Bandara etal.

researched the impact of high speed rail at various speeds, and communication channel bandwidth availability, while taking into account transceiver overlap areas as shown in Table 3.1.

Table 3.1: Number of Handover Packets, Speed, and Overlapping Distance

Overlapping Area (miles)	**Number of Packets**	**Maximum Speed (mph)**	**Supported Data Rate (kbps)**
0.3	<50	400	16
0.2	50	379	32
0.2	200	190	64

Table 3.1 describes that as speed increases from 190 Miles Per Hour (mph) to 400 mph the data rate and number of available packets decrease. The analysis conducted in [20] supports the concept that noise contributors such as the Doppler effect will play a factor since the received signal will have shifted in frequency due to the increase in noise;therefore, the number of data packets that are able to be supported would decrease to relay information. Security will play a greater impact since the number of available packets has decreased. Given the limitations in supported data rate and speed described in Table 3.1, Hartong's block based movement authority [15] would need to be adapted to include another layer of restrictions due to bandwidth availability.

Since spectrum management and policy management have been studied extensively in [16, 18, 20–22] an aspect that was missing was security. Security which is introduced in [3] proposes a concept of a multi-tiered cognitive radio where a master cognitive engine would be responsible for coordinating all aspects of secure communications. Lowered tiered services, such as the cryptographic engine and a specialized IDS, would relay results to the master cognitive engine. The internal architecture [3] is described in great detail and introduces a publish-subscribe architecture that I created to relay internal communications decisions to other components of the system. The aforementioned architecture is developed to ensure safety, security, and efficient use of RF spectrum.

Looking beyond spectrum availability and management and into rail security is a common theme emerging in publications [23–26] is that a CBTC network would and has become targeted since the RF information can easily propagate within the reach of an attacker. As early as 2012, an attacker has scanned and attempted access to CBTC networks according to [26] and the referenced General Accounting Office (GAO) report. The GAO reported that 7 percent of the attacks were scans, probes, and attempted access, and 16 percent were unauthorized access. The current American Railway Association (ARA) specification [8] uses a truncated HMAC SHA-1. RRIDS, developed as a part of this dissertation, was created to detect, deter and prevent attackers of rail infrastructure.

Schneier [11, 12] postulated that SHA-1 will eventually be hacked which it was through the Freestart attack[13]. Freestart is an attack technique that specifically is designed to attack the SHA class of algorithms. [13] describes a methodology in cracking and producing collisions in SHA algorithms. [13] state that the current financial cost to find salt collisions is approximately $180,000 United States Dollars (USD). Improvements in technology combined with decreasing technological costs over time will enable Freestart to become a common attack tool. According to National Institute of Standards and Technology (NIST) [27] SHA-1 should no longer be trusted as a data integrity algorithm. As a solution to this problem, this dissertation extends the μTESLA [2] protocol to derive salts and algorithms dynamically, that would make broken hash keys useless in crafting a broadcast message.

Mitola's original intelligent radio concept was expanded to add the capability to detect malicious attackers which are working to undermine the safety of the PTC system. Dietzel etal. [28] describes a V2V IDS framework based on generalized subjective reasoning. The generalized subjective reasoning is a useful start but does not take advantage of known facts specific to the V2V protocol. In [28] the authors advocate that a general analysis of received network traffic is conducted by vehicles in the V2V network using a trust model that computes trust based on Equation 3.1.

$$o := (b, d, u) \tag{3.1}$$

The variables which comprise Equation 3.1 are as follows:

- the vehicle's IDS has a belief denoted as b, where b evaluates to $b \in [0, 1]$ is either honest or data is correct
- the vehicle's IDS has a disbelief denoted as d, where d evaluates to $d \in [0, 1]$ is either dishonest or the data is incorrect
- an uncertainty exits, $u \in [0, 1]$, which represents uncertainty in the results

Evaluating the trust of a vehicle based on V2V or V2I network traffic is useful, but it has potential drawbacks with regards to performance. A specification based IDS has inherent advantages over anomaly based IDS [28,29] in that it uses information specific to the protocol to determine if a node is behaving improperly. Known facts of the CBTC PTC protocol as well the cryptographic derivation techniques in enhanced derivation schema would aid in the determination of the attacker. RRIDS is introduced as a specialized IDS for PTC and leverages facts concerning the CBTC protocol as well as cryptographic derivation facts to determine and categorize the attacker. In [22] a description of a possible risk mitigation engine was introduced as a means for the PTC radio to ascertain the communication's environment. The risk mitigation engine in [22] begins to describe the concept of a threat actor that is not weather related. I take [22] and quantify the threat actor based on the message structure and cryptographic facts that can only be known to non threat actors.

Research at Oxford University has started to formally model rail signaling with models created in [30]. Failure Divergence Refinement (FDR) is used to model locomotives entering and performing actions within blocks. In the model [30], Simpson and Jacobs construct a beaconing network similar to that envisioned for the PTC network. The model incorporates detection devices, analogous to track sensors and the WIU, which relays information such as if the track segment is currently *occupied* (o) or *cleared* (c). In the model, the locomotive can take actions such as *controlled normal* (cn) or *controlled reverse* (cr). These actions correspond to locomotive movements within the block. The model can then be used to test the logical correctness of the interlocking control logic while the locomotive is moving

within the block. The underlying result in the model was to check the feasibility of FDR within a cloud environment to analyze a complex model. According to the paper [30] further research is needed to extend their model.

The consequence of not safeguarding rail infrastructure from cyber-attacks can lead to catastrophic accidents such as [31]. In Philadelphia [31] Amtrak 188 commuter train derailed causing 200 passenger injuries and 8 fatalities. According to [32], the accident would have been avoided if the PTC network was operational. According to the VOLPE Report [33] conducted that GPS can be attacked either by spoofing the GPS timing signal or by intentional or unintentional jamming. Therefore, a cognitive engine evaluating the health of communications and other metrics would provide situational awareness which can be acted upon to prevent a catastrophic incident. The technology could have notified the operator as well as the locomotive control system to safely slow down and take caution.

Chapter 4: Design of the Cryptographic Cognitive Engine and Intrusion Detection System

This chapter describes the design and architecture to solve the secure communications issues stated in Chapter 2. This architecture consists of two main components, a cryptographic cognitive engine and an intrusion detections system, RRIDS, designed to provide cryptographic services and an intrusion detection services. The following sections in this chapter are as follows:

- **Section 4.1:** describes the software architecture which was used to design the overall cognitive engine
- **Section 4.2:** describes the software architecture of the cryptographic cognitive engine
- **Section 4.3:** describes the software architecture of RRIDS
- **Section 4.4:** describes enhancements made to the communications protocol for WIU broadcasts

4.1 Cognitive Engine Design Architecture

The design of the cognitive engine is based on the concept of a layered architecture which loosely follows the layers in the Open Systems Interconnection (OSI) model. Each OSI layer provides a single objective.

Figure 4.1 is segmented into three disctinct OSI like layers that illustrate an overview of the communications layers. These layers are used to communicate and relay events to and from the cognitive engine, RRIDS, transmitting the signal, or relaying the information to a subscribed entity through the middleware layer. Detailed decomposition of the CE

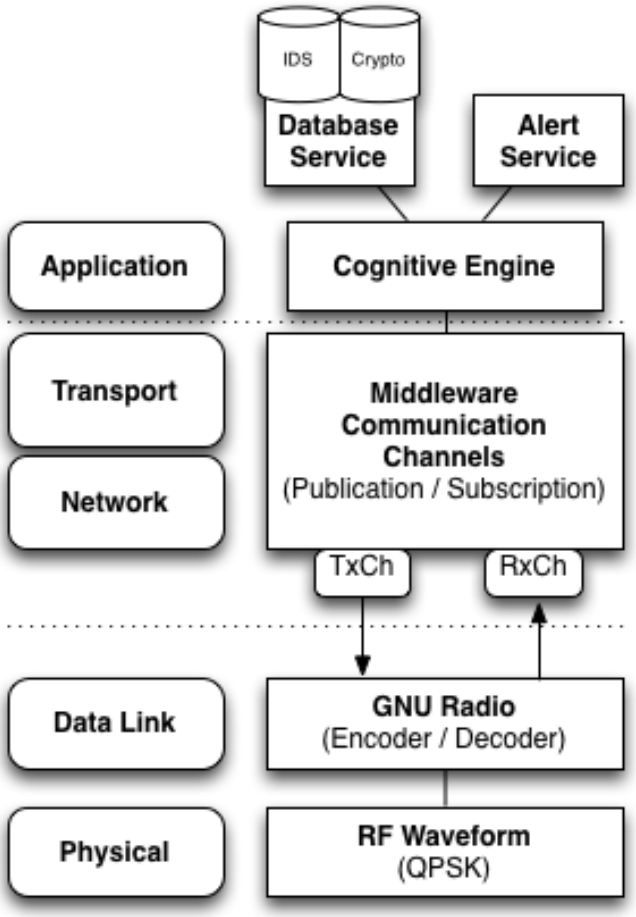


Figure 4.1: Overall Internal Communications Architecture

communications architecture is described in Table 4.1. The application layer, shown in Figure 4.1, contains a database service that is used to store the cryptographic material and results from the IDS. The alert service relays its decisions to the back end control office infrastructure. The Transport and Network layer consist of a middleware used to interface between the radio layer, the underlying RF radio, and connected subscribed services. The physical layer uses the underlying Quadrature Phase Shift Keying (QPSK) RF modulation and demodulation services.

Table 4.1: Cognitive Engine Communications Architecture description specific for the PTC as illustrated in Figure 4.1

Locomotive Transmission

1. **Application Layer:** Cognitive Engine

   (a) The locomotive selects to broadcast a `GetWIU` status message. The objective of the `GetWIU` status message is to turn the listening WIU beacon on to broadcast current status messages to the locomotive prior to it entering the block.

   (b) The CE makes an internal system call to retrieve its current time referenced to Greenwich Mean Time (GMT) and clock synchronized using NTP. The referenced time is then used to query the cryptographic database and retrieves the current hashing algorithm and salt.

   (c) The information is then framed to include the following information and represents a subset of the ARA [8] specification:

   - **Originating Time**: GMT time stamp to indicate the time that the message was created
   - **Message Type**: the type of message corresponding to the ARA [8] specification. I implemented two message from the specification `GetWIU` message (Type: 5201) and the corresponding broadcast message (Type: 5202) WIU broadcast.
   - **Beacon ID**: the intended beacon that needs to respond to the message or the transmitting beacon.
   - **CRC**: the CRC used to ensure that no transmission propagation errors effect the system.
   - **Hash**: the data integrity check corresponding to $hash = hash_{fxn,salt}(Message|TimeStamp)$ which uses the current hash function and salt for the timestamp.
   - **Sequence Number**: the message sequence number to aid in detecting replay attack.

2. **Transport** & **Network Layer:** Middleware Communications Channels

   (a) The framed message is sent to the REDIS Transmission Channel (TxCh).

3. **Data Link Layer:** GNU Radio

   (a) The framed message is sent to the REDIS Transmission Channel (TxCh) that is enqueued to GNU Radio services for transmission.

4. **Physical Layer:** RF Waveform

   (a) GNU Radio dequeues the information and then broadcasts the information using the QPSK waveform.

Table 4.2: Receiver's Cognitive Engine Communications Architecture description specific for the PTC as illustrated in Figure 4.1

WIU Reception
1. **Physical Layer:** RF Waveform (a) The radio synchronizes with the QPSK waveform and passes the information to GNU Radio to decode the signal. 2. **Data Link Layer:** GNU Radio (a) GNU Radio decodes the QPSK waveform and applies appropriate signaling processing filters and publishes the information to the REDIS RxCh. 3. **Network** & **Transportation Layer:** Middleware Communications Channels (a) RxCh receives the information and places it into a listening queue ready for the cognitive engine to act upon. 4. **Application Layer:** GNU Radio (a) The cognitive engine retrieves the received packet from the REDIS queue and passes the data frame to RRIDS to be processed by other modules. (b) RRIDS checks the CRC, Hash based on the frame originating time, the message type, hash, and sequence number and logs the results and infers the type of attacker based on its evaluation. All results are time stamped with GMT time.

The basis of the cryptography is Perrig's TESLA [2] protocol that extends Lamport's authentication schema. Lamport's schema uses a single key and the property of a strong one-way cryptographic function to derive additional keys. A strong one-way cryptographic function has the property that it is easy to compute forward but difficult to invert in a reasonable amount of time. I extended TESLA [2] for securing rail beacons. The comparison of the protocols and enhancements are as follows. Perrig's TESLA's algorithm is shown in Algorithm 1. The enhancements that I made to TESLA is shown in Algorithm 2.

The TESLA protocol shown in Algorithm 1 works as follows:

Line 1: Prior to messages being transmitted both parties divide the communication time into time intervals. The time interval is represented by t_{int}.

1. $S.$: Divides Total Communication Time into t_{int}
2. $S.$: Create salts using a strong one way hash function and assigns $K_i \forall t_{int}$
3. $R \rightarrow S : \{t_r\}$
4. $S.$: Calculates Δ
5. $S \rightarrow R : \{t_{int}|d\}$
6. $S \rightarrow R : \{\Delta|K_i\}$

Algorithm 1: A. Perrig [2] TESLA Approach

Line 2: The transmitter creates keys for each time interval, t_{int}.

Line 3 & Line 4: The transmitter measures the total time it would take for the receiver to respond to a bootstrap message which takes into account, propagation delay, processing delay, and outside interferes and is assigned to Δ. Δ is used in determining the key disclosure time.

Line 5: The transmitter sends the receiver the key disclosure time. The key disclosure time is used to indicate to the receiver when the key will be available to validate the previously sent messages.

Line 6: Messages are sent by the sender and the key is disclosed (by the sender) at predefined time periods.

After the key is disclosed for the known time interval, the transmitter moves to the next derived salt and the process is repeated. Eventually, all keys and salts will be exhausted and the communications bootstrap process would need to be initiated. In line 5, a key disclosure time is sent from the sender (S) to the receiver (R), the disclosure time represents the time in which the key will be sent to verify the previous set of messages transmitted. If the disclosure time or key is not received by the receiver then it could affect the overall validity of this approach. Further details are contained in [2] and Appendix D.
Contrasting my approach to that of TESLA, the main difference is that information, such as the disclosure time, time intervals, the key, and the salt in the bootstrap broadcast, is

1. $Secure \rightarrow S, R : \{MAC|IV \in (algo_{seed}, salt_{seed})|t_{int}\}$
2. $S., R.$: Divides Total Communication Time into t_{int}
3. $S., R : Salt, Algo \rightarrow t_{int}$
4. $S, R : Salt_{unused} \oplus Current_{salt}$
5. $S \rightarrow R : \{Message\}$

Algorithm 2: Cryptographic Cognitive Engine Enhancements to TESLA

never revealed. Essentially, the objective is not to reveal any information concerning salts being used thereby limiting the attacker to only the broadcasted information and minimizing their ability to perform cryptographic analysis. The approach that is described in Algorithm 2 provides enhanced security without disclosing K_i which I believe aids in safeguarding rail communications. Additionally, my solution has the capability for continuous key generation in case the locomotive is in the block longer than expected. The following is the decomposition of Algorithm (2):

Line 1 : The salt seeds are sent as part of the bootstrapping process by the back office to the transmitter (WIU) and the receiver (train).

Line 2: The total time for the day is subdivided into time intervals and is known only to the sender (S) and receiver (R) and sent through the secure back office.

Line 3: The algorithm and salt is valid for the time interval t_{int}.

Line 4: In the algorithm the system continuously samples salts and performs a bitwise xor of the salts to generate a continuation seed. This maybe needed in the case the locomotive is longer in the block then expected.

Line 5: At this point messages are being transmitted and verified with K_i. The sender (S) and the receiver (R) rotates integrity algorithms
and Keys K_i without disclosure.

Unlike in TESLA, there is no need for a disclosure time since both the sender (S) and

receiver (R) already have the disclosure time, time interval, algorithms, and salts which they generate. The generation of salts follows TESLA's approach but the verification of the messages are different since the sender (S) and receiver (R) contains all of the information that it needs.

My enhancements provide greater security without the need to use additional bandwidth and protocol synchronization since I take advantage of the underlying rail infrastructure. Additionally, my solution adds the benefit of random integrity algorithm selection in which adds another layer of protection. If the salt is compromised the algorithm used to validate the salt may not have been cracked.

Figure 4.2 provides a greater decomposition of the cognitive radio internal layered architecture. Currently, I have implemented a subset of the PTC protocol, the IDS, RxCh, TxCh, and the C2 channels within the application layer to communicate to GNU Radio. The security channel and location channel are additional research areas that is needed to be completed once the master cognitive engine is completed. The boxes shaded in gray in Figure 4.2 represent areas of interest outside the scope of this dissertation.

The following describes in greater detail the cognitive engine design shown in Figure 4.2:

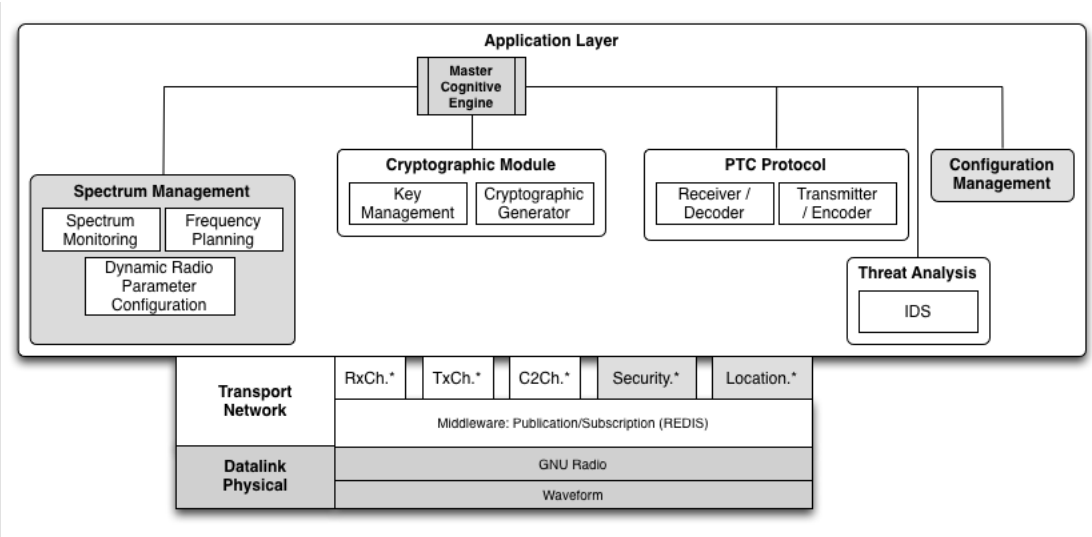


Figure 4.2: Cognitive Engine (gray shade represents other students research work which does not pertain directly to this dissertation.)

1. **Crypto Module**: The cryptographic module or crypto module is used to derive all cryptographic keys and then store them into an underlying database. As shown in Figure 4.1, all keys are stored in the underlying database for later retrieval and are time stamped referencing GMT as the cryptographic timing reference. The time reference is essential for all communications and cryptographic actions. GPS is used as the underlying radio time synchronization source, and NTP is used as the computer clock reference source. The components which make up the cryptographic module are as follows:

    - **Key Management**: The key management stores and manages the salts and algorithms in the database. Each cryptographic entry has an associated time stamp, selected algorithm, and salt for a specific start and stop time. The query and retrieval of cryptographic material for both the salt and the underlying hashing algorithm for time t and follow Equation 4.1:

    $$SaltAlgo(t) = query(t)_{t_{start} < t \leq t_{stop}}(CryptoDatabase) \tag{4.1}$$

    - **Cryptographic Generator**: Generates the cryptographic material based on my algorithms. The cryptographic material is then stored in the cryptographic database. Each cryptographic material is calculated with a time interval that consists of a start and stop time, a distinct hashing algorithm, and a salt which is then inserted into the *CryptoDatabse* as shown in equation 4.2:

    $$CryptoGen \xrightarrow{t_{start}, t_{stop}, algorithm, salt} (CryptoDatabase) \tag{4.2}$$

2. **PTC Protocol**: The PTC protocol consists of a message interrupter where once the RF message is received the data frames are decoded and transported through the middleware the receiver is responsible for translating the framed or unframed message as PTC messages.

- **Receiver/Decoder**: The receiver/decoder is responsible for taking the received messages and decoding it appropriately based on the ARA specification [8] and associated bit boundaries.
- **Transmitter/Encoder**: The transmitter/encoder is responsible applying the GMT timestamp, encodes the message and then prepares it for transmission.

3. **Threat Analysis**:
   - **IDS, RRIDS**: Represents the intrusion detection subsystem that is responsible for evaluating received messages and categorizes the received messages as either being from authorized transmitters, noise, or an attacker. RRIDS is further described in section 4.3.
4. **Middleware/RxCh, TxCh, C2Ch**: A publication and subscription service is responsible for receiving messages from the physical layer as well as intermodule communications. The middleware service uses the opensource REDIS [10] project. Essentially, it abstracts the radio from the application layer to allow for a logical separation of radio communications and cognitive engine logic.

I partially implement the Positive Train Controller's WIU broadcast protocol. The protocol subset I implemented consists of the three commands `GetWIUStatus`, the resulting `WIUStatus`, and `BeaconOn`. The `GetWIUStatus` message is initiated by the locomotive and the response message `WIUStatus` is returned by the WIU [8]. The internal communication architecture shown in Figure 4.1 illustrates messages being received, decoded and verified. After a message is received, the message is placed into a Receive Channel (RxCh). Services residing on the application layer subscribe to RxCh to evaluate the received message and perform appropriate actions in response to the received messages. The implementation of the internal communications is described in Section 5.1 and the middleware communications architecture is shown in Figure 5.1.

4.2 Cryptographic Cognitive Engine

The back office, illustrated in Figure 1.3 is used to securely distribute the IV seeds. Figure 4.3 illustrates the relationship between the WIU salt generation and its relationship with rail communications. Figure 4.3.a illustrates the salt generation from the securely provided seed and its subsequent generated salts for a given time interval for the day. Figure 4.3.b illustrates the representative messages that are used for the message beacons. Once the seeds are securely distributed, the derived salts are stored securely and used appropriately. In my architecture for secure rail communications, each WIU is assigned a unique seed for the day which is used to derive a salt S_n. The salt generation follows my previously discussed approach.

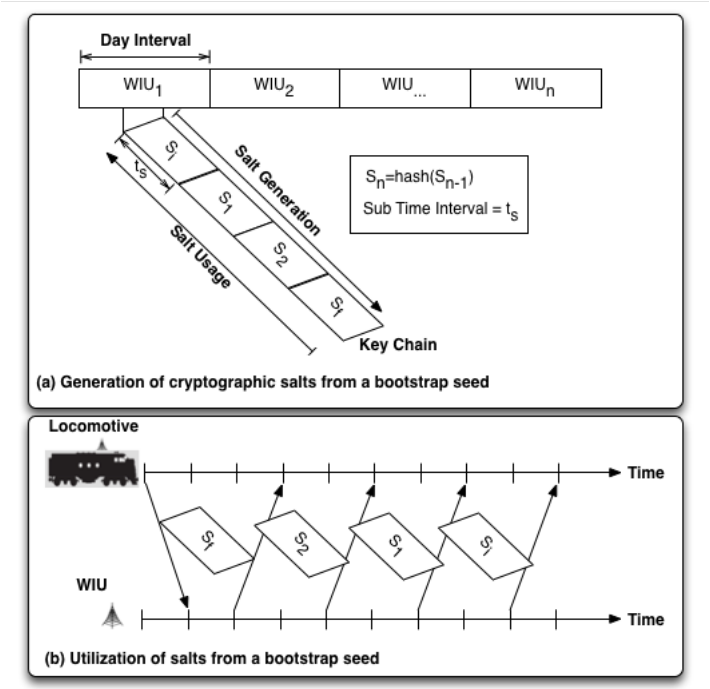


Figure 4.3: (a) Illustrates enhanced TESLA generation of cryptographic material, (b) Illustrates the utilization of the salts, algorithms with the WIU beacon.

The steps in communications are as follows in Figure 4.3.b:

Locomotive to WIU: The locomotive broadcasts a GetWIU message to the WIU. The WIU verifies the integrity of the message.

WIU to Locomotive: Once the integrity of the received message is verified the WIU then broadcasts 5 response messages corresponding to the ARA specification [8].

Locomotive: The locomotive then verifies the message based on knowledge of the WIU assigned seed and derived hash algorithm and salt.

Algorithm 3 describes how seeds are securely transferred to the locomotive and WIU from the back office. It is assumed that the back office has a secure network.

Definitions

B ≡ Back Office

L ≡ Locomotive

W ≡ WIU

h ≡ Integrity Algorithm Function

Pk ≡ Public Key

pk ≡ Private Key

t_s ≡ Time Stamp

Seed Distribution

1. $B \rightarrow L|W : \{CRC|h(seed_{list|t_{int}})|seed_{list}|t_{int}|t_s\}_{L|WPk}$
2. $L|W \rightarrow B : \{CRC|h(ack|nak|t_s)|ack|nak|t_s\}_{BPk}$

Algorithm 3: Seed Distribution

The following describes the communications between the back office, locomotive and WIU.

Line 1: Securely transfers the cryptographic material, including the hashing algorithm seed and salt seed to the locomotive and beacon using Public Key Infrastructure (PKI). The locomotive and WIU has a public and private key pair to securely receive its cryptographic material. The seeds represent day seeds for communications and respective time intervals, t_{int}, for hashing algorithms and salts.

Line 2: Either an Acknowledgment (ACK) message is received from both the locomotive and WIU in the reception of the seeding message or a Not an Acknowledgment (NAK) message is sent and received.

It is assumed that the back office will be responsible for ensuring that there are no repetitions in salt and algorithm pairs to avoid cryptographic collisions. Currently, in my schema the algorithm, salts, and seeds are provided securely to the underlying system. The cryptographic salt and algorithm derivation is performed by the cognitive engine located at the locomotive on its radio as well as at the WIU. Figure 4.4 illustrates the services and subcomponents used to provide services to the master cognitive engine. The model used in the production and usage of cryptographic material is a generator and a user. The production and the evaluation of a received message are conducted by the WIU Locomotive Process.

In Equation 4.3, salts are derived for the WIU and locomotive using enhanced μTESLA. Each salt is assigned a utilization start and stop time for day communications. Once the utilization time expires for both the WIU and locomotive, a new salt is queried from the cryptographic database and used for the current time. The usage of salts and algorithms follows a First In Last Out (FILO) queue where the final calculated salt is the first to be used to follow the previously introduced concept of a strong one-way cryptographic function.

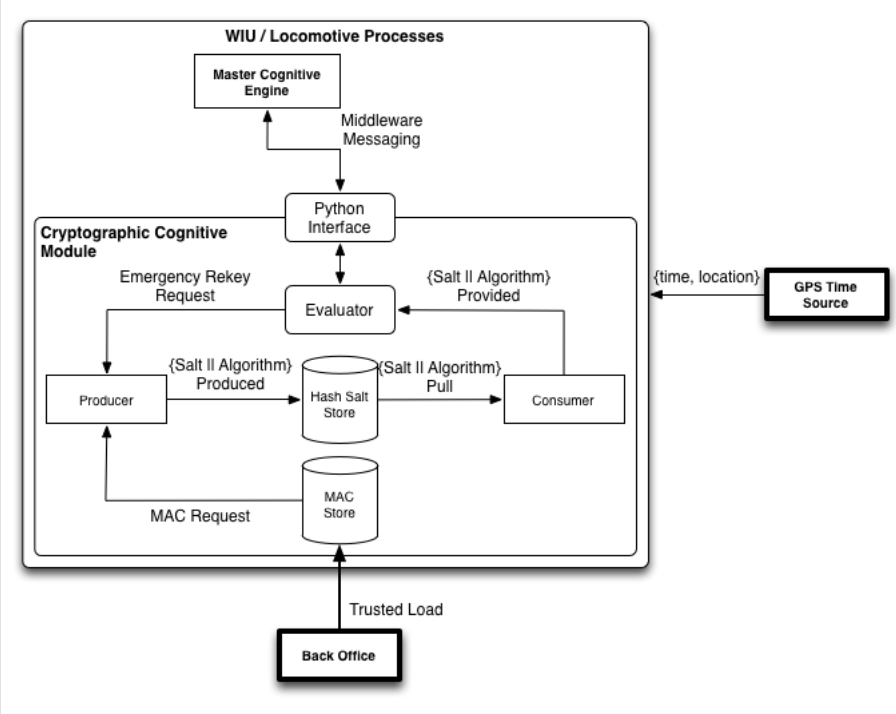


Figure 4.4: Cognitive Engine Cryptographic Module

$$salt_{list} = \forall(hash(seed_{salt})_{time} mod(hashAlgo_{total})) \tag{4.3}$$

Figure 4.3.b shows communications between the locomotive and WIU and illustrates the usage of salts over time. Algorithm 4 describes the sequence of communications between the locomotive and WIU. A single queue is used to illustrate the derivation of the seeds and insertion into a FILO queue. Adding material into the queue utilizes the concept of a strong one-way hash function based on Strong Preimage resistance with the approach [2] outlines.

Each of the parameters that are shown in Figure 4.5 and are as follows:

1. κ: Derived key used for the predefined communications period.

2. α: The integrity algorithm selected for the period of communications.

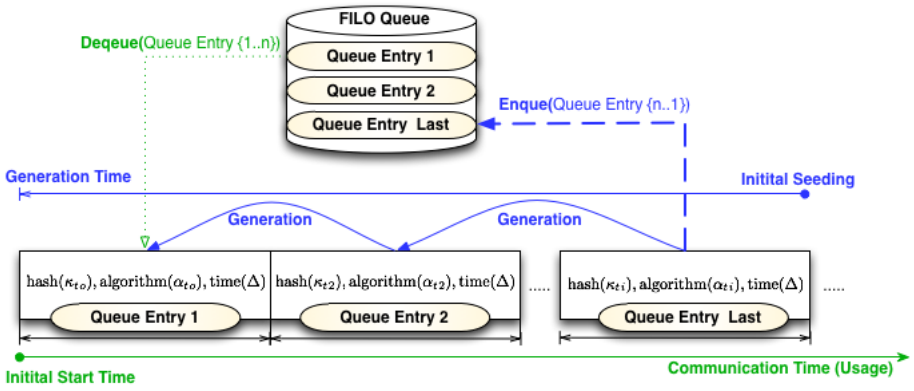


Figure 4.5: μTESLA Architecture Key Generation and Usage

3. Time(Δ): Represents the cumulative delays (transport, queuing and processing).

Each element in the queue follows Equation 4.11 and is represented as:

$$\Phi(t) = \epsilon(t) \in \lambda \tag{4.4}$$

$$\epsilon(t) \equiv \alpha(t_{o-i}), \kappa(s, t_{o-i}), t_{\Delta} \tag{4.5}$$

In Equation 4.4, Φ, represents a generator function which is used to produce time based cryptographic material for a specific event ϵ and key-algorithm chain λ for all ϵ and is expressed as Equation 4.6.

$$\forall\lambda(t_{total}) : \epsilon(t) \tag{4.6}$$

$$Seed_{new} = F(\forall(hash(seed_i \oplus seed_{i-1} \oplus hash_i))) \tag{4.7}$$

In Equation 4.7 the function F is used to generate a MAC value used for the communication time period based on derived seeds from the final seed to the initial seed to derive a new Message Authentication Code (MAC). The seeds as well as the result of the hash

is XORed ($\oplus$) and hashed. The XOR function spreads the distribution uniformly thereby adding randomness additional hashing of the derived MAC provides additional randomness.

Preconditions

1. The locomotive and WIU have been securely seeded by the back office.
2. The clocks are indirectly synchronized with a timing source such as GPS.

Definitions

L $\equiv$ Locomotive

W $\equiv$ WIU

h $\equiv$ Integrity Algorithm Function

$t_s \equiv$ Time Stamp

P $\equiv h(Message|time_{stamp})|Message|time_{stamp}$

Communications Interaction

1. $L \rightarrow W : \{CRC(P_{getwiu}))|P_{getwiu}(t|P)_{salt|algo|t}$
   The locomotive sends a request for beacon status at time t_i with the final derived salt for the day period.
2. $W \rightarrow L : \{CRC(P_{wiustatus}))|P_{wiustatus}(t|P)_{salt|algo|t_j}$
   The WIU response is a WIU status message with salt and algorithm associated for that current time period within the day period.

Algorithm 4: Rail and WIU Communications

As time progresses, salts and algorithms are used until the end of the day is reached. Once the end of the day is reached, new seeds and salts are provided by the back office for the rail block. Therefore, as critical as securely seeding the locomotive and WIU, a common time reference, such as GPS, is also needed to keep all communication devices synchronized to allow for seed usage, integrity checks, and hashing.

When the message is validated by verifying the CRC, the IV is made by the Cryptographic Cognitive Engine and the results are published to the Security Channel and RRIDS. Prior to entering the block at a scheduled time, the Locomotive begins to generate the salt

```
Input:  CRC: crc, Salt: s_i, true hash: hash_i(M ⊕ s_i)
Data: CryptoContainer:(Algoithm, Salt, Time)
Result: CRC Error, IV Error, VALID
while Receiving Messages do
    1. Start Producer Thread
       while Producing do
        if Over The Air Rekey (OTAR) Request then
            MAC = OTAR_MAC
        end
        time = get current time
        if firstHash then
            result = hash(MAC)
            alogrithm = (result ⊕ MAC)% Total Algorithms
            Queue.enqueue(result,time, algorithm)
        end
       end
    2. Start Consumer Thread while Consuming do
        CryptoContainer = Queue.dequeue()
        NewSeed = hash(CrypotContainer.Salt ⊕ MAC)
        if Queue.length < 50% then
            Start Producing on a secondary Queue
        end
        if CurrentQueue == Completed then
            PrimaryQueue = SecondaryQueue
        end
       end
    if hash and salt are correct then
        return Publish Security VALID
    else
        if CRC Invalid then
            return Publish Security CRCError
        else
            return Publish Security IVError
        end
    end
end
```

Algorithm 5: Producing Cryptographic Material

and algorithm pairs. The WIU will have already generated its day seeds. When a locomotive enters a region it sends a beacon request and uses the integrity value for the specific time needed to find the appropriate salt and hash algorithm needed to verify the integrity of the message. Figure 4.6 illustrates the communication sequence between the Locomotive and Beacon.

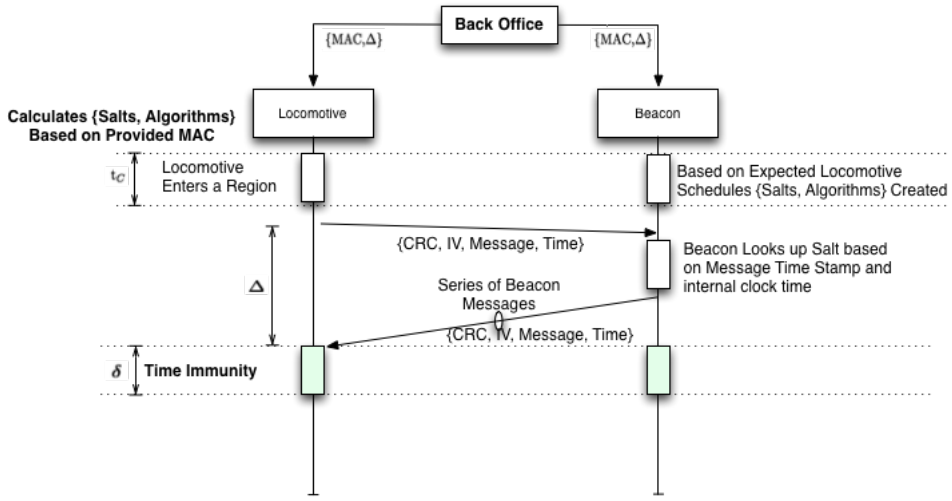


Figure 4.6: Cognitive Engine Cryptographic Sequence Diagram

There are two forms of clock synchronization, *Direct* and *Indirect* time synchronization [2]. Because the locomotive and PTC networks are not physically connected by cable, GPS is used as an indirect time synchronization source as explained in Appendix D.0.2. Additionally, because the oscillator internal to a computer is imperfect, its internal clock will drift over time [2]. Consequently, GPS is used as a reference time source to correct for the internal clock drift which would effect cryptographic synchronization. As illustrated in the sequence diagram shown in Figure 4.6, there is a value of Δ which represents the cumulative delay between the Locomotive and the Beacon. Δ is an empirical measurement made by the locomotive and the beacon radios which represent the total round trip time, which includes queuing, processing, and communication response. The Back Office provides

an estimate of Δ to all locomotives as an initial estimate prior to entering a region. δ is an additional safety factor based on a potential vulnerability identified in [1] in using the TESLA protocol.

4.3 Design of the RRIDS System

The purpose of RRIDS is to determine if and when a forgery and message replay attack occurs. RRIDS purpose is to provide detection and notification services. The subcomponent evaluates the CRC and the integrity value. Each evaluation results in a message being sent to the master cognitive engine, leaving the master cognitive engine to determine the required reactions. Illustrated in Figure 4.7, RRIDS is a passive IDS, once messages are received and evaluated, alert actions are taken. RRIDS performs three types of evaluations to evaluate CRC Error, Replay Attacker, and Forgery Attacker stated in Chapter 3. When RRIDS completes its evaluation, RRIDS logs its results in the RRIDS database and concurrently an alert message is generated and published to the middleware layer. Figure 4.7 illustrates the underlying communications between RRIDS, the IDS database, and the cryptographic store.

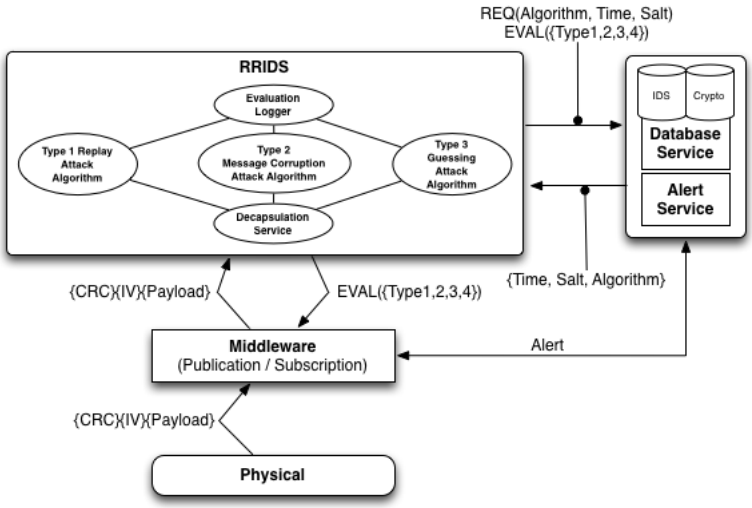


Figure 4.7: RRIDS Software Design and Architecture

The underlying cryptographic design and architecture in RRIDS uses a form of TESLA [2] which is enhanced for RRIDS and applied for CBTC systems. Enhanced TESLA is a subset of μTESLA in which it provides authentication mechanisms based on the originating seed and resulting salt derivations. Integrity algorithms, and cryptographic salts, derived in enhanced TESLA provides authentication and improved data integrity by linking derived salts, and algorithms to the initial seed provided to the WIU and locomotive within the rail block. The back office, illustrated in Figure 1.3 is used to securely distribute the IV seeds.

4.3.1 CRC Error (Type 1) Detection

The CRC Error detection follows the standard algorithm to calculate transmission error for received messages. The message content and its hash value is evaluated and reported appropriately.

Equation 4.8 is used to determine if the message has been corrupted during transport or has been corrupted by an attacker.

$$\begin{aligned} &MessageCorruption : (CRC = INVALID) \\ &\&(hash = IN|VALID)\&Current|OldTimestamp \\ &\&(M = Corrupted) \end{aligned} \tag{4.8}$$

In the case of CRC errors, the message is checked for corruption and then logged without any further inspection unless the CRC is Valid and the resulting hash is invalid.

4.3.2 Corrupted Message (Type 2) Attack Detection

The message corruption attack stated in Section 1 is one in which messages are captured by the attacker, corrupted then re-transmitted to either the locomotive or WIU. Figure 4.8 illustrates the message sequence processing within RRIDS. Algorithm 6 describes the currently implemented detection algorithm.

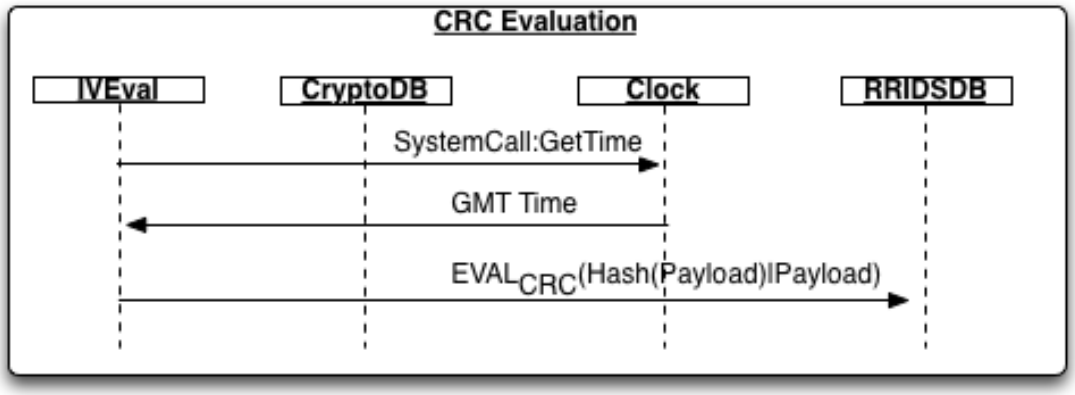


Figure 4.8: CRC Message Corruption Sequence Diagram

Equation 4.8 shows the logic used to detect a message corruption attack. RRIDS checks the CRC, the hash, the timestamp, and syntactical correctness of the received message. If the CRC is evaluated to be invalid, the hash, timestamp, or message may have been corrupted. The rate of message corruption could lead one to infer that a Denial of Service (DoS) attack is being conducted.

4.3.3 Replay (Type 3) Attack Detection

The replay attack stated in Section 1 is one in which messages are captured by the attacker and then re-transmitted to either the locomotive or WIU. Figure 4.9 shows the message communications sequences for evaluating a replay attack and described in Algorithm 7.

Equation (4.9) shows the logic used to detect a replay attack. RRIDS checks the CRC, the hash, the timestamp, and syntactical correctness of the received message. The hash is verified based on the salt and algorithm derived using the approach described in Section 4.2.

Preconditions

1. Timing source is accurate
2. Seeds distributed, boot strap process completed
3. Message received

Processes

IVEval ≡ Integrity Value Evaluation Process

CryptoDB ≡ Cryptographic Database Process

Clock ≡ System Clock

RDB ≡ Rail Radio Intrusion Detection System Database (RRIDSDB)

RED ≡ REDIS

R ≡ RRIDS

Q ≡ Query

$t_s \equiv$ Time Stamp

P $\equiv h(Message|time_{stamp})|Message|time_{stamp}$

Sequence of Internal Interactions

1. $RED \rightarrow R : \{P\}$
   Message P is received through the middleware layer from REDIS (RED) and passed to RRIDS (R) for evaluation.
2. $IVEval \rightarrow Clock : \{time()\}$
   GMT time is requested by IVEval to get the current time and for later usage.
3. $Clock \rightarrow IVEval : \{GMT_{currentTime}\}$
   The clock returns the current GMT time.
4. $IVEval \rightarrow RDB : \{EvalResult\}$
   The message evaluation uses the current system time, and the message time stamp is evaluated. The results of CRC and message, time stamp is then sent to RRIDSDB.

Algorithm 6: Message Corruption Sequence Description

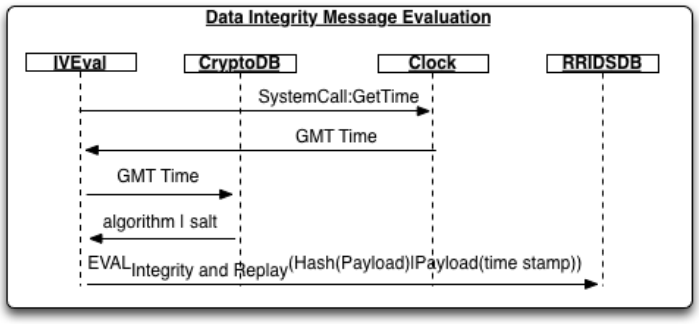


Figure 4.9: Replay Process Sequence Diagram

$$ReplayDetected : (CRC = VALID)\&(hash = INVALID) \\ \&M_{OldTimestamp}\&(M = SyntacticallyCorrect) \tag{4.9}$$

Equation 4.9 is used as a basic detection algorithm and executed within RRIDS. Detection is based on the premise that if the hash is invalid and the message has an old time stamp then it is labeled as a potential replay attack. The resulting detection is then committed into the RRIDSDB.

4.3.4 Forgery (Type 4) Attack Detection

The message guessing attack stated in Section 1 is one in which messages originated by the attacker is transmitted to either the locomotive or the WIU. Figure 4.10 illustrates the message sequence processing within RRIDS. Algorithm 8 describes the guessing detection algorithm.

Equation 4.10 shows the logic used to detect a guessing attack. RRIDS checks the CRC, the hash, the time stamp, and syntactical correctness of the received message. The basis for the attack detection is that if the CRC is correct, the timestamp is current and the message

Preconditions

1. Timing source is accurate
2. Seeds distributed, boot strap process completed
3. Message received

Processes

IVEval ≡ Integrity Value Evaluation Process

CryptoDB ≡ Cryptographic Database Process

Clock ≡ System Clock

RDB ≡ RRIDSDB

RED ≡ REDIS

R ≡ RRIDS

Q ≡ Query

$t_s \equiv$ Time Stamp

P $\equiv h(Message|time_{stamp})|Message|time_{stamp}$

Sequence of Internal Interactions

1. $RED \rightarrow R : \{P\}$
   Message P is received through the middleware layer from REDIS (RED) and passed to RRIDS (R) for evaluation.
2. $IVEval \rightarrow Clock : \{time()\}$
   GMT time is requested by IVEval to get the current time and for later usage.
3. $Clock \rightarrow IVEval : \{GMT_{currentTime}\}$
   The clock returns the current GMT time.
4. $IVEval \rightarrow CryptoDB : \{Q_{GMTTime}(salt|algo)\}$
   The cryptographic salt and algorithm is provided to IVEval.
5. $IVEval \rightarrow RDB : \{EvalResult\}$
   The evaluation results is provided to RRIDSDB based on the parameters.

Algorithm 7: Replay Evaluation Process Sequence Description

is syntactically correct and the hash is incorrect RRIDS qualifies the attack as a forgery attempt. Equation 4.10 to deduce that the system is under a forgery or guessing attack.

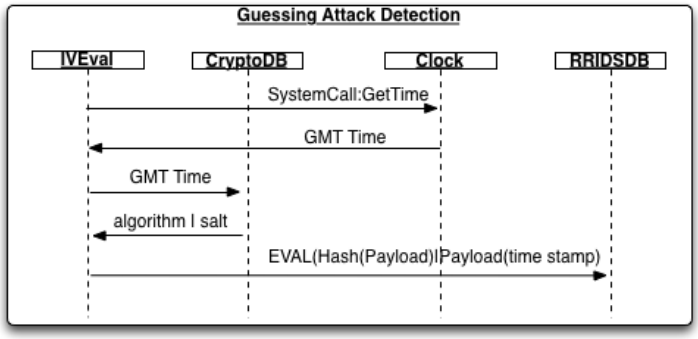


Figure 4.10: Guessing Detection Sequence Diagram

$$MessageCorruption : CRC = VALID$$

$$\&IV = INVALID \tag{4.10}$$

$$\&CurrentTimestamp\&M = SyntactticallyCorrect$$

4.4 The Secure Protocol for WIU Beaconning

Secure WIU beaconing protocol extends the TESLA protocol in two ways. First, it provides changing cryptographic salts used in communications. Second, extended TESLA adapts the integrity algorithm in a manner which is difficult for the adversary to predict. TESLA [2] and my extension to it are contrasted in Table 4.3.

The salts and algorithms are used for a specified time period then expired for the next time period which is known to both the locomotive and the beacon. The generation of the salts follows Algorithm 5 for both the locomotive and WIU. The locomotive and WIU generates independently an integrity algorithm and salt based on the IV.

Preconditions

1. Timing source is accurate
2. Seeds distributed, boot strap process completed
3. Message received

Processes

IVEval ≡ Integrity Value Evaluation Process

CryptoDB ≡ Cryptographic Database Process

Clock ≡ System Clock

RDB ≡ RRIDSDB

RED ≡ REDIS

R ≡ RRIDS

Q ≡ Query

$t_s \equiv$ Time Stamp

P ≡ $h(Message|time_{stamp})|Message|time_{stamp}$

Sequence of Internal Interactions

1. $RED \rightarrow R : \{P\}$
   Message P is received through the middleware layer from REDIS (RED) and passed to RRIDS (R) for evaluation.
2. $IVEval \rightarrow Clock : \{time()\}$
   GMT time is requested by IVEval to get the current time and for later usage.
3. $Clock \rightarrow IVEval : \{GMT_{currentTime}\}$
   The clock returns the current GMT time.
4. $IVEval \rightarrow CryptoDB : \{Q_{cryptodb}(GMT_{currentTime})\}$
   IVEval requests cryptographic material for the current time from the cryptographic database.
5. $IVEval \rightarrow CryptoDB : \{Q_{GMTTime}(salt|algo)\}$
   The cryptographic salt and algorithm is provided to IVEval.
6. $IVEval \rightarrow RDB : \{EvalResult\}$
   The evaluation results is provided to RRIDSDB based on the parameters. Evaluation is based on equation 4.10.

Algorithm 8: Guessing Detection Sequence Description

Input: Algorithm : IDS, CRC: crc, Salt: s_i, true hash: $hash_i(M \oplus s_i)$, ReceivedTime, Δ, δ
Data: CRC, IV, Message: M
Result: CRC Error, Spoof Attempt, Forge Attempt,SuspiciousMessage, VALID
while *Receiving Messages* **do**
  if $ReceivedTime > (\Delta + \delta)$ **then**
    return SuspiciousMessage
  end
  if *(CRC == INVALID)* **then**
    return *CRC Error*
  end
  if *(CRC == VALID) & (IV == INVALID) & (M==Syntactically Correct)* **then**
    return *Forgery Attempt*
  end
  if *CRC & IV == VALID* **then**
    if *(CRC == VALID) & (IV ==* $VALID_{Previous\ Salt}$*)& (M==Syntactically Correct)* **then**
      return *Spoof Attempt*
    end
    if *(CRC == VALID) & (IV == VALID) &* $M_{Old\ Timestamp}$ *& (M==Syntactically Correct)* **then**
      return *Replay Attempt*
    end
    return *VALID*
  end
end

Algorithm 9: IDS Determination

Table 4.3: TESLA vs. Extended TESLA

Property	**TESLA**	**Extended TESLA**
Low computation overhead for the generation and authenticity verification of received messages	✓	✓
Low communications overhead	✓	✓
Limited data buffering due to device resource constraints	✓	✓
Robustness to packet loss	✓	✓
Scales to large broadcast for multiple receivers	✓	✓
Adaptive Algorithms		✓
Time Attack Immunity		✓

4.4.1 Attacking TESLA

According to [1], a vulnerability that lies in TESLA is disrupting the indirect time synchronization during the bootstrap process between communicating devices. TESLA uses a time disclosure process to disclose Δ to communicating devices. Disruption as well as publishing a fictitious time would provide an avenue of attack. The attacks range from a DoS attack to publishing an attacker generated MAC. Figure 4.11 outlines a process to attack TESLA.

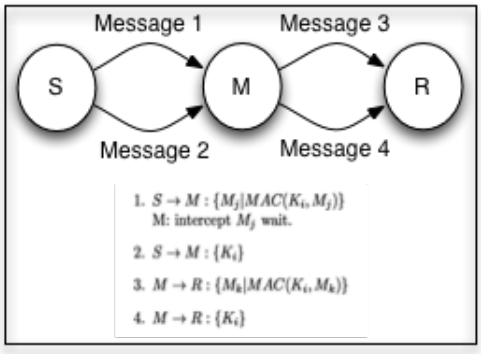


Figure 4.11: Attacking TESLA Time Synchronization [1]

The following describes a man in the middle attack described by [1] and shown in Figure 4.11. S and R represents Sender and Receiver communications nodes. M is a malicious node subverting communications.

1. M decodes the MAC as well as the disclosure time. M delays the response to $R \rightarrow S$.
2. S sends a message to M with the communication key K_i assuming that it is communicating to R.
3. M uses the MAC disclosed by S as well as a crafted message M_k
4. M sends R a validation of the key, K_i it received previously from S

According to [1], malicious node M is part of the TESLA network and forging messages to M. The delay that M introduces gives it an opportunity to act as a man in the middle

and thereby attack R.

Unlike TESLA, my solution does not reveal the disclosure or the bootstrap process. The disclosure time is not part of my protocol, essentially, the total processing, queuing and path loss is used to determine how long it takes for the beacon or locomotive to receive, calculate the integrity value, and start transmission. The bootstrap process is carried out on the PTC secure signaling network. The solution will prevent the attacker from taking advantage of the aforementioned vulnerabilities.

4.5 Synchronization

Synchronized communications are essential for both TESLA and physical layer communications [2]. I use indirect time synchronization based on a common clock reference as described in Section D.0.2. The time Δ takes into account transmitter and receiver internal clock drift.

Figure 4.12 shows a message being broadcast by the WIU to any locomotive that can hear the WIU radio. Once clocks are indirectly synchronized, the locomotive begins communicating with the WIU and calculates Δ. The Δ is used as the worst case delay and is the queuing, processing, and propagation delay.

The locomotive infers trust because the initial seed provided to the WIU and the locomotive uses the trusted source of the back office. Event and time correlation [34, p. 38] is integrated into the cryptographic cognitive engine; details of the architecture and integration are described in Chapters 4 and 5.

Clock and event synchronization are important for time-critical secure communications. If the derived salts are not used in the correct order and time then communications between devices will not occur. Therefore, clock and event synchronization is imperative for proper cryptographic synchronization. Figure 4.13 pictorially show the relationship between time and communication events.

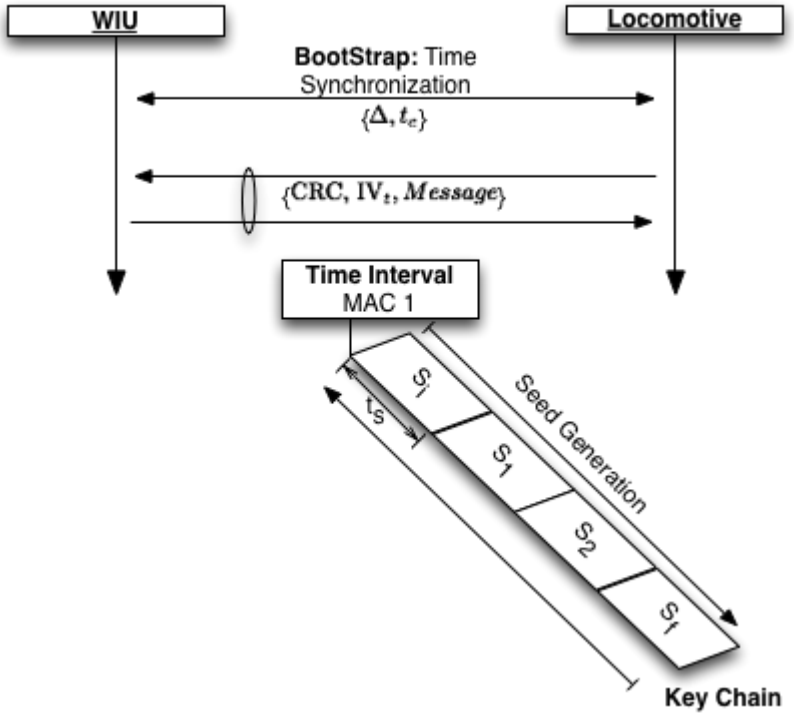


Figure 4.12: Sequence diagram of message exchanges to establish trust

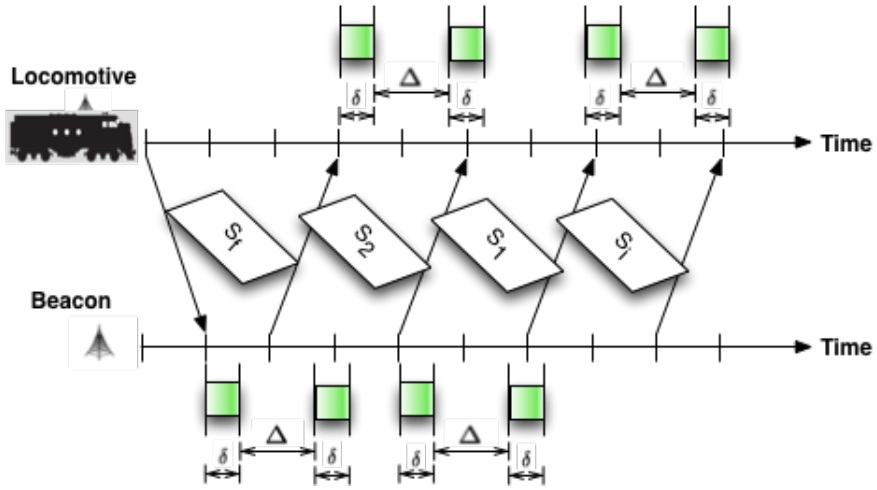


Figure 4.13: Pictorial relationship between time and communication events derived from [2].

$$t_{com} = 2 \cdot \delta + \Delta \tag{4.11}$$

In the protocol, RRIDS, that I developed the locomotive then uses the salt based on the initial transmission time. If the beacon switches to the next salt S_2 it would not impact communications because both salts are valid for a time period as illustrated in Figure 4.13. Additionally, the cryptographic database aids in ensuring that the previous salt and hashing algorithms are available.

$$\forall \lambda : \epsilon(t) \tag{4.12}$$

In Equation 4.11 salt, algorithm, and time are tied together as time events and are illustrated in Figure 4.14. If the total amount of time expires beyond the expected time as well as the cumulative delay then the salt, and algorithm will have expired and not used.

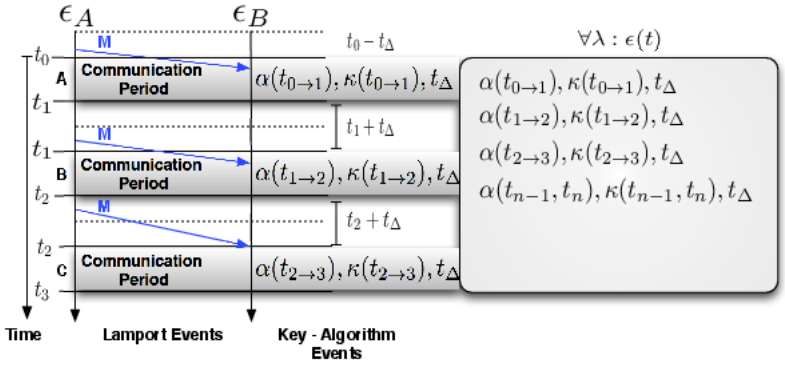


Figure 4.14: Event and key-algorithm chain relationship.

In Figure 4.14 ϵ_A and ϵ_B represent communicating devices. They can be decomposed as an Alice and Bob communications sequence.

Line 1: At time $t_{(0\to1)}$: Alice sends to Bob a message M, within a time delta, t_Δ with

Definitions

ϵ_A: Communication Device A - Alice

ϵ_B: Communication Device B - Bob

α: Algorithm

κ: Key or Salt

t_Δ: Time Interval

Alice and Bob Communications

1. $\epsilon_A \rightarrow \epsilon_B : M_{\alpha,\kappa} : t_{(0\rightarrow 1)} : t_\Delta$
2. $\epsilon_A \rightarrow \epsilon_B : M_{\alpha,\kappa} : t_{(1\rightarrow 2)} : t_\Delta$
3. $\epsilon_A \rightarrow \epsilon_B : M_{\alpha,\kappa} : t_{(2\rightarrow n)} : t_\Delta$

Algorithm 10: Event and Key Relationship Protocol Description

a designated precomputed algorithm, and salt.

Line 2: At time $t_{(1\rightarrow 2)}$: Alice sends to Bob a message M, within a time delta, t_Δ with a designated precomputed algorithm, and salt. The previous time, $t_{(0\rightarrow 1)}$, algorithm and key expired and a new algorithm and salt is utilized.

Line 3: At time $t_{(2\rightarrow n)}$: Alice sends to Bob a message M, within a time delta, t_Δ with a designated precomputed algorithm, and salt to the n^{th} message for complete communications.

Each event ϵ acts as an ordered set based on time in which the receiver is expecting to receive either a current event or a past event which can be decoded with an associated α, κ in a given t_Δ. Table 4.4 shows the ordering where ϵ_A is a transmitter and ϵ_B is a receiver. The cryptographic key and algorithm period is defined as the duration in time. Additionally, $\Phi(t)$ is a time-based function which pseudo-randomly chooses the cryptographic algorithm $\alpha(t)$, and cryptographic algorithm $\kappa(t)$ for a specific time period $((t_n \rightarrow t_{n-1}) \pm t_\Delta)$. Table 4.4 orders the time segments based on time and illustrates the logical ordering of events.

Table 4.4: Cryptographic event ordering

Time Segment	**Time**	ϵ_A	**Order**	ϵ_B	$\Phi(t)$
A	$0 \rightarrow 1$	ϵ_0	$<$	ϵ_1	$\alpha(t_{0\rightarrow 1}), \kappa(t_{0\rightarrow 1}), t_\Delta$
B	$1 \rightarrow 2$	ϵ_1	$<$	ϵ_2	$\alpha(t_{1\rightarrow 2}), \kappa(t_{1\rightarrow 2}), t_\Delta$
C	$2 \rightarrow 3$	ϵ_2	$<$	ϵ_3	$\alpha(t_{2\rightarrow 3}), \kappa(t_{2\rightarrow 3}), t_\Delta$

The cryptographic events and their logical ordering for ϵ_A and ϵ_B in Table 4.4 are based on logical ordering with the expectation that the event ϵ is ordered based on time. The ordering is based on the exception that each ϵ has a hard timing deadline. If an event is missed it will have no impact on the communication. The selection of the cryptographic algorithm and the generation of cryptographic keying material, α and κ still continue for the given communication time period. Each time segment in Table 4.4 has an associated key-algorithm chain λ which is comprised by $\Phi(t)$. Another perspective is illustrated in Figure 4.14 that combines all events where blue lines represent messages being transmitted from ϵ_A to ϵ_B. The message being sent in a given time period are generated using Equation 4.13.

$$\{M\}\{\alpha(t_{period}), \kappa(t_{period})\} \tag{4.13}$$

Chapter 5: Implementation

5.1 Internal Cognitive Engine Communications

This chapter describes the underlying internal communications architecture first and the implementation of the cryptographic engine thereafter. The internal communications use a publish and subscribe architecture illustrated in Figure 4.7 using an open source software library referred to as REDIS [10]. The cryptographic cognitive engine illustrated in Figure 4.2 relays its decisions to other subscribed components through a general Security channel using Security.KeyManagement, Security.KeyManagement.Distro, and Security.ThreatEval subchannels. The illustration in Figure 4.2 shows the regular grammar notation .*. The publication subscription architecture shown in Figure 5.1 shows the interconnections of middleware to the cognitive engine using REDIS.

The decomposition of the channels and its inter-linkage to the master cognitive engine and acts as the internal communications architecture of the overall cognitive engine.

The regular grammar notation allows for a greater level of granularity in providing specific processes to listen on channels needed for communications. The receive channel looks at only two types of messages with respect to the ARA specification [8]. The `GetWIUStatus` and `BeaconOn` messages. I use the Locomotive ID, CRC, IV, and message type. The distinguishing factor is the Message Type for the [8] protocol.

- **RxCh.***: The receive channel receives ARA[8] messages. REDIS utilizes regular expressions to further sub-categorize channel information so that specific processes can subscribe and fuse data based on the specification. Every received message is assigned a MessageID upon arrival and associated with the received data. The MessageID association of a transmission source to the geolocation of the emitter provides another

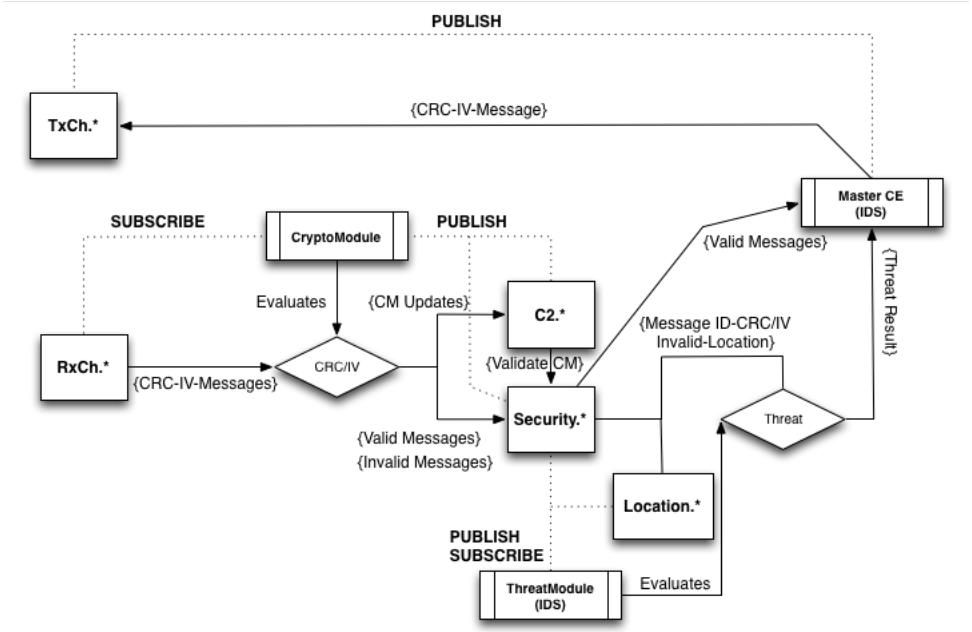


Figure 5.1: Publication Subscription overall architecture [3]

attribute for determining if the system is under attack. Figure 5.2 illustrates an overview of the messages specific to the communications channel. I implemented the following subset of the ARA specification [8].

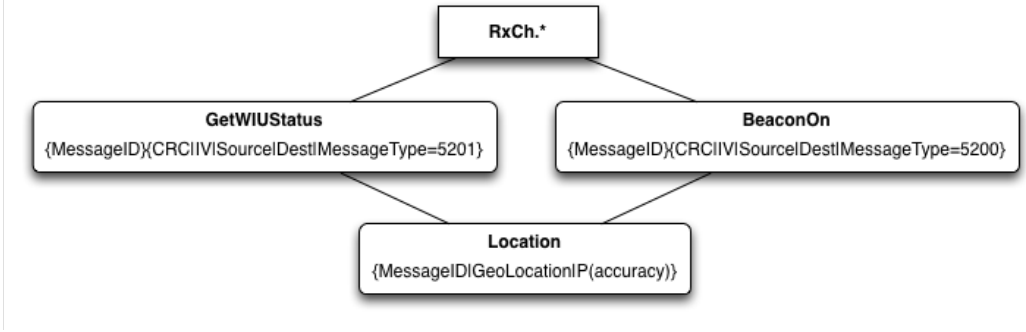


Figure 5.2: Receiver Message Hierarchy and Messages

`GetWIUStatus`: WIU sub-channel used to send a request for status information from a named WIU [8, pp.19-20].

`BeaconOn`: A WIU sub-channel used to request a beacon to be turned on so that the WIU radio beacon at a specified interval [8, pp.20-21].

Location: The geolocation channel communicates the geolocation and the accuracy of the geolocation calculation estimate. This information is fused using the Message Identifier (ID).

- $\{MessageID|GeoLocation|P(accuracy)\}$:
  - **MessageID**: is a random string which represents a key associated with the received message.
  - **GeoLocation**: standard latitude, longitude, elevation notation or Military Grid Reference System (MGRS).
  - $P(accuracy)$: accuracy of the geolocation.

- **TxCh.***: When a message is decoded, a transmit response message is created and sent to the locomotive. The subset of the messages are illustrated in Figure 5.3.

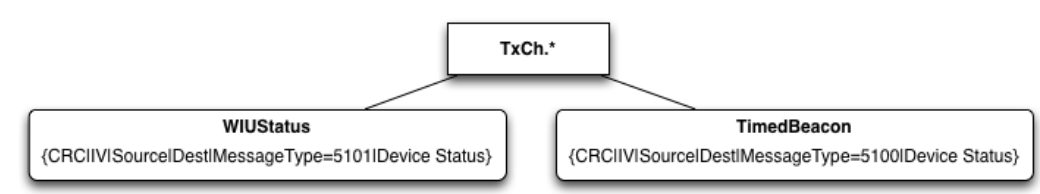


Figure 5.3: Transmitter Message Hierarchy and Messages

  - `WIUStatus`: Relays current operational / health status of the WIU beacon [8, pp.17-19]
  - `TimedBeacon`: Relays information of beaconing [8, pp.16-17]

- **C2.***: C2 messages are relayed to the signaling network

- **Security.***: The security channel communicates its determination of RRIDS to the master cognitive engine. The message that it supports are the following:

  1. **Normal**: A normal expected message
  2. **CRC Error**: Message was corrupted in transmission message fields are improper
  3. **Forgery Attempt**: Message was forged since the CRC is valid as well as the data fields
  4. **Replay Attack**: Message was previously replayed

  This component is to ensure timely delivery of messages to subscribed components. Figure 5.4 shows the decomposition of the message structure.

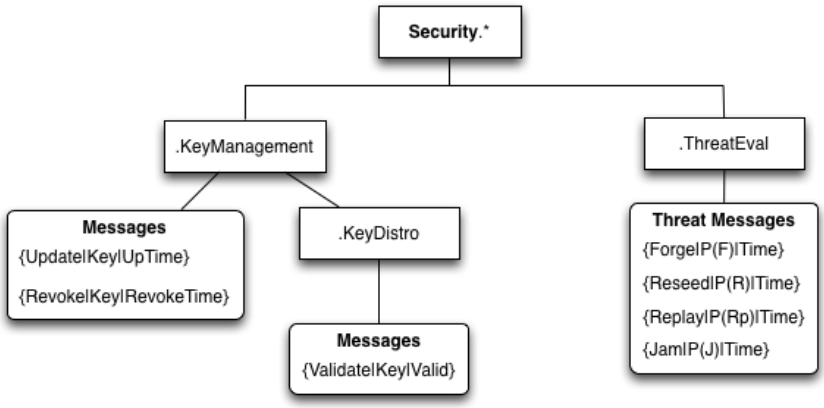


Figure 5.4: Security Message Hierarchy

  - **KeyManagement**: The key management channel provides key management updates as well as seeds that are available for communications to all subscribed processes. The decomposition of the messages are as follows:

    * $\{Update|Key|UpTime\}$

      Update: Command to update the key.

      Key: Key use for future seed generation.

UpTime: The update time when the key is supposed to switch over.

* $\{Revoke|Key|RevokeTime\}$:

Revoke:Revoke the Key and Key-chain of interest

Key: Key identifier for the key of interest

Revoke Time: Indicated future time to revoke the key and related key chain

- **KeyManagement.KeyDistro**: The key distribution channel is used to inform the system of a newly created seed
- **ThreatEval**: communicates the threat to subscribed processes. The general components of the message structure is {Threat|Probability of Threat|Threat Time}. Once published, the master cognitive engine will evaluate the probability and make a decision based on the calculated probability.

  $\{Forge|P(F)|Time\}$: The probability that the message is a forged message. Essentially the security component will evaluate the current IV with previous seeds used in communications.

  $\{Reseed|P(R)|Time\}$: If a reseed request is fictitious based on external confirmation from the signaling network.

  $\{Replay|P(R_p)|Time\}$: Indicates if the message is a previously replayed message by comparing it to previous seeds as well as previously transmitted messages.

  $\{SuspiciousMessage|P(S)|Time\}$: Indicates if the message is suspicious based on the difference between the received time and the expected time.

Chapter 6: Experimental Validation

In order to determine the effectiveness of the cryptographic hash generation and RRIDS in categorizing attacks (namely CRC, Corruption, Replay and Forgery attacks) I conducted experiments with and without radio noise in the presence of an attacker and an artificially generated controlled radio noise. Experiments without noise were used to measure memory, Central Processing Unit (CPU), and threading efficiency. Experiments with noise tested the capability to detect attacks with differing RF noise levels, clock mis-synchronizations in the presence or absence of active attackers. These experiments were conducted to validate the hypotheses stated in Chapter 2, restated here.

Hypothesis 1 Cryptographic Security

When communicating devices with self-derived cryptographic material do not reveal cryptographic details, such as salt or key disclosure time [2], it lessens the capability of the attacker to effectively create attacks such as message replay and message forgery in the presence known amount of fixed radio noise.

Hypothesis 2 Cryptographic Performance

Derivation and utilization of self-derived cryptographic material within the communicating device will not affect system performance for deriving, selecting, and calculating integrity values during communications in the presence of selected amounts of constant radio noise. Consequently my system does not violate time sensitive safety requirements of PTC.

Hypothesis 3 Intrusion Detection System

RRIDS provides controlling authorities information that allow them to act against detected attackers to enhance safety and security in the presence of selected amounts of constant radio noise.

6.1 Experimental Setup

Two distinct experimental setups were used. Both used an anechoic test chamber to isolate RF noise and Ettus N210 [35] radios. The test suite used for experiments without noise is shown in Figure 6.1. It has two Ettus N210 radios, one acts as a locomotive and the other acts as the beacon. The test suite also contains custom logging software modules, common to both experiments so that logging will not degrade the experiments performance. The logging information is stored in MySQL so that it can be later queried and analyzed.

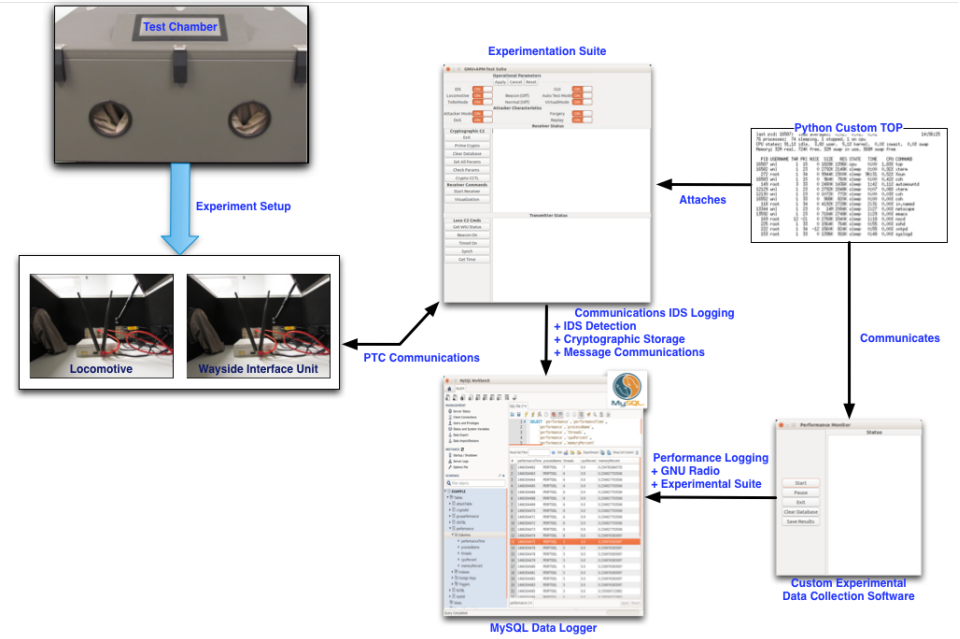


Figure 6.1: The two radio, no noise test suite.

The experimental setup shown in Figure 6.2 is used for tests involving experiments with controlled noise. The test suite is run in an anechoic test chamber to isolate RF noise, four Ettus N210 [35] radios and the previously mentioned performance logging software.

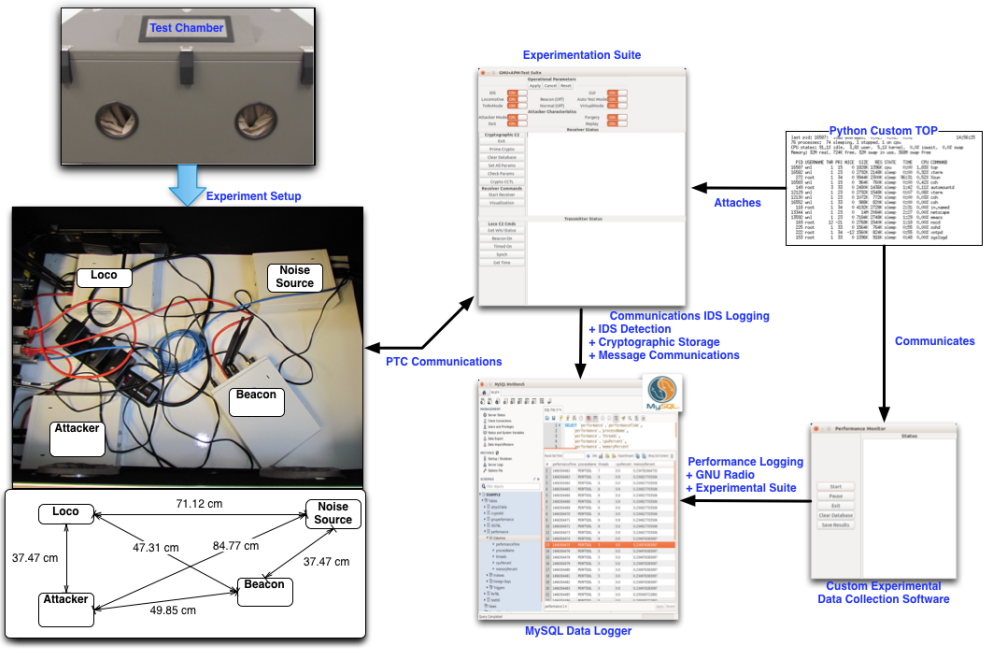


Figure 6.2: Experimental hardware and software used to test the effectiveness of RRIDS. The Ettus radios are placed in the chamber during testing.

Statistical irregularities were accounted for by running each experiment three independent times. Common to all tests was exiting the application, verifying that all processes associated with GNU Radio, Python, and C were killed, and restarting the experimentation suite and the performance logging software.

6.2 Experimental Overview

Experiments without noise are described in Section 6.3 and are used to validate performance of the system under varying attacks, clock synchronization conditions (clock skew), and changing cryptographic time periods. The following tests were used to validate the hypotheses:

Experiment Without Noise

- **Section 6.3.1 Normal Operations**: Experimental and performance results describing the normal communications between a WIU and locomotive.
- **Section 6.3.2 Replay Detection**: Experimental and performance results describing Replay Attacks being conducted against the WIU.
- **Section 6.3.3 Forgery Detection**: Experimental and performance results describing RRIDS capability to detect an attacker guessing at a cryptographic salt to create WIU messages.
- **Section 6.3.4 Normal Varying Cryptographic Rollover Period**: Experimental and performance results describing RRIDS capability to *roll over* cryptographic time boundaries as salts are expiring.

The objectives of experiments with noise are different from those without noise and are described in Section 6.4. The experiments in that section are used to validate the capability of the system to detect an attacker in the presence of RF interference. The following tests are used to validate my hypotheses.

Experiment With Noise

- **Section 6.4.1 Replay Attack With Noise**: Experiment testing the replay attack with noise present in the environment.
- **Section 6.4.2 Replay Attack With No Noise**: Experiment testing the replay attack with no noise present in the environment.
- **Section 6.4.3 Forgery Attack With Noise**: Experimental and performance results describing Replay Attacks being conducted against the WIU.

6.3 Experiments Without Noise

Experiments without noise test the system without the presence of RF interference. Time intervals for the cryptographic parameters and the time interval between cryptographic roll over periods are identically set for the transmitter and receiver. Once the cryptographic time parameters were set, RRIDS then generated the cryptographic material. The conducted experiments had the locomotive and WIU transmit and receive approximately 500 `GetWIUStatus` messages [8] at a rate of one message every 10 seconds. In the experiment RRIDS software was installed on the target WIU. According to the ARA specification [8], when the WIU validates a received locomotive message, the WIU transmits five status messages for track and/or block conditions. This set of experiments did not make any distinction between different types of WIU messages.

6.3.1 Normal Operations

Normal operations are defined as broadcasts that do not involve an attacker or an RF interference source. The cryptographic algorithm and salt expiration period is set to approximately 36 minutes. The hypotheses that are verified in this section are Hypothesis 1, Hypothesis 2, and Hypothesis 3.

Experimental Procedure

The cryptographic material was created as described in Section 4.2. The keys are generated for a total duration of 1 day and the time is segmented into 40 time segments. The time segments are equivalent to 2160 seconds or 36 minutes between the cryptographic roll over event. The transmitter and receiver cryptographic key chain generation are started approximately at the same time. The experiment was run independently three times to gather experimental sample data as well as to detect experimental anomalies.

Experimental Output

The results shown in Table 6.1 are summarized as follows:

- RRIDS positively detects the expected message >∼93% in all the three independent experimental runs,
- The false positive detection rate is <∼7%

The detection pattern illustrated for RRIDS shown in Figure 6.3 plots the three experiments and 500 samples per experiment in the time domain.

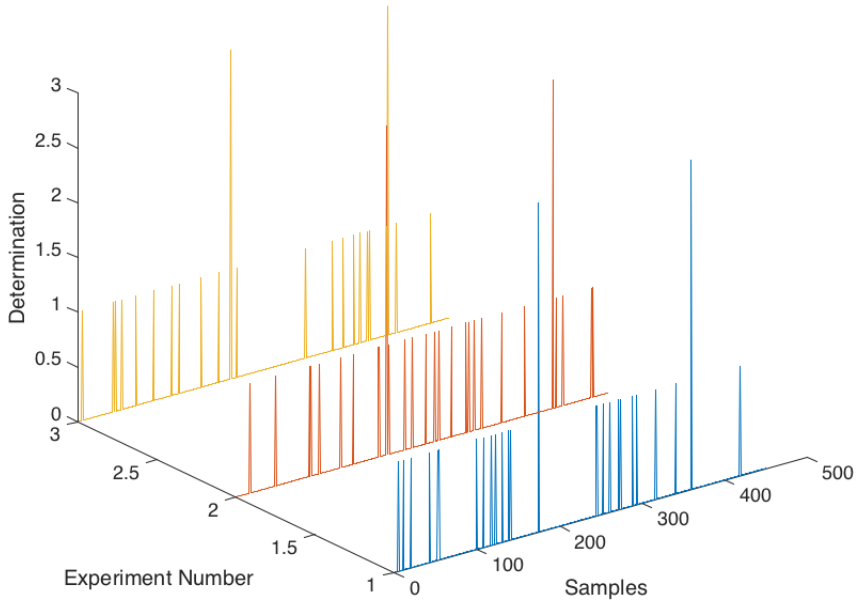


Figure 6.3: IDS determination: Normal Operation with a Cryptographic Interval set to 2160 seconds (36 minutes)

Validation of Hypothesis

In experiment 6.3.1 normal `GetWIU` status messages are broadcast from the locomotive to the WIU beacon. The WIU beacon responds with a `BeaconOn` response message. This set of experiments validate the following set of hypotheses:

> Hypothesis 1 is validated that no leakage of cryptographic material occurs during the transmission of locomotive or WIU messages.

Hypothesis 2 was validated with the system performance staying consistent at approximately 0.9% CPU utilization worst case and 0.6% memory utilization worst case.

Hypothesis 3 was validated as shown in Table 6.2 and Table 6.1. The false positive detection rate was low at approximately 6.2% for normal non attacker in a RF non noise environment.

6.3.2 Replay Detection

As defined in Section 3, a replay attack is one in which a previous message is copied and rebroadcast. The objective of this test case is to validate RRIDS detection capability during a replay attack within a non RF noise environment. The cryptographic algorithm and salt expiration period is set to approximately 36 minutes. This section describes validating Hypothesis 1, Hypothesis 2, and Hypothesis 3.

Experimental Procedure

Due to limitations in the test setup, the locomotive acts as the attacker; therefore, the attacker did not overpower the radio first. The first message is copied and then replayed 500 times. The experimental configuration is identical to the normal operations test case described in Section 6.3.1. After 500 messages were sent, it was decided to only run one experiment due to the aforementioned limitations. The replay transmission rate is 1 replayed message every 10 seconds. The experiment was conducted for three independent runs and the results are as follows.

Experimental Output

The results that were captured in the experiment showed that RRIDS captured all replay attacks. A transmitted message was captured and replayed for 500 times after being captured.

- RRIDS positively detects a replay attack 100% of the time

- RRIDS no anomalies were detected due to a simple setup of just two radios

Validation of the Hypothesis

In experiment 6.3.2, a normal `GetWIU` status messages are broadcasted from the locomotive to the WIU beacon. The locomotive then stores the message and rebroadcasts it continuously to behave as a replay attacker this is due to initial limitations in the two radio setup shown in Figure 6.1. The WIU beacon does not respond and logs the information. This experiment validates the following set of hypotheses:

> Hypothesis 1 is validated that no leakage of cryptographic material occurs during the transmission of locomotive or WIU messages.
>
> Hypothesis 2 was validated with the system performance staying consistent at approximately 0.9% CPU utilization worst case and 0.6% memory utilization worst case.
>
> Hypothesis 3 was verified as shown in Table 6.2 as well as in Table 6.1 that the false positive detection rate at 100% for a replay attacker in a non RF noise environment.

6.3.3 Forgery Detection

As defined in Section 3, a forgery attack is one in which an attacker has the capability to forge a message based on seeds that it will use to derive the cryptographic material. The objective of this test case is to validate RRIDS detection capability during a replay attack within a non RF noise environment. The cryptographic algorithm and salt expiration period is set to approximately 36 minutes. This section attempts to validate hypothesis 1, hypothesis 2, and hypothesis 3.

Experimental Procedure

In the experiment, three independent tests were conducted and results are plotted in Figure 6.4. The results in Figure 6.4 illustrate that RRIDS detects the attacker as either a guessing attacker or mis-categorized as a replay attacker.

Experimental Output

Table 6.1 summarizes the collected data as the following:

- RRIDS positively detects a random guess attack ∼93% of the time
- RRIDS mis-categorizes a replay attack ∼16% of the time

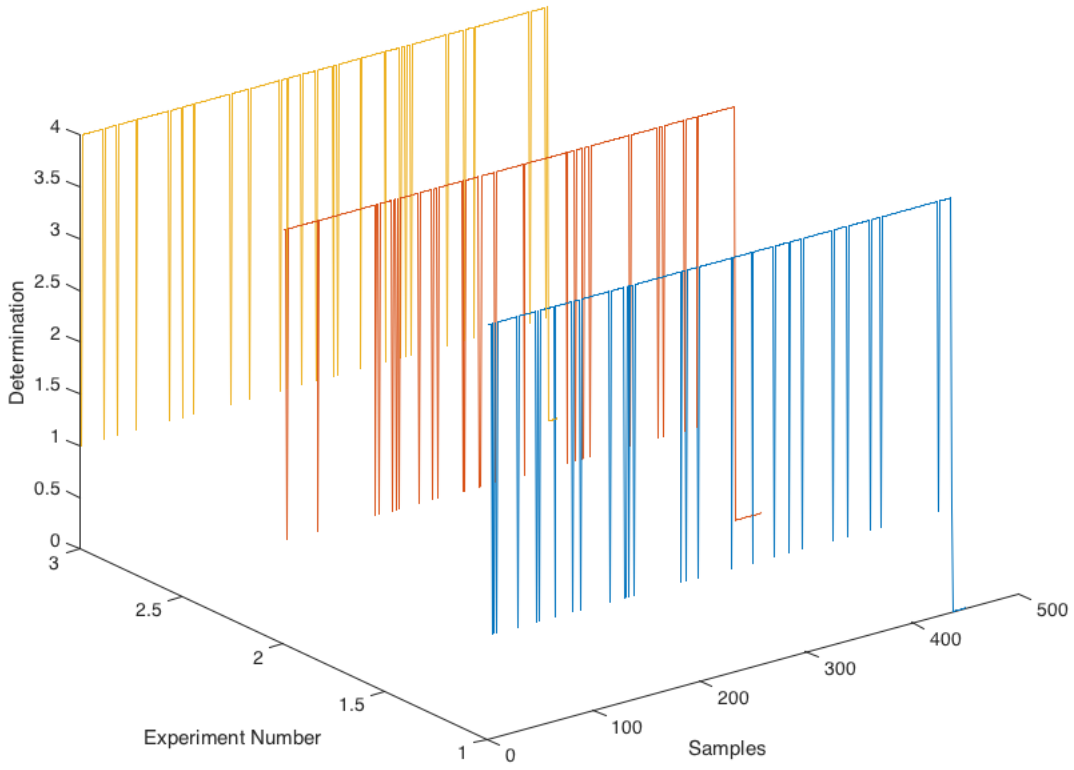


Figure 6.4: IDS determination: Random IV Seed Attack, Cryptographic Interval set to 2160 seconds (3.6 minutes)

Validation of the Hypothesis

In experiment 6.3.3 normal `GetWIU` status messages are forged and sent to the WIU beacon. The forgery locomotive creates its own salts and algorithms based on the same cryptographic generation schema as the legitimate locomotive and beacon. Due to limitations in the test setup, the locomotive acted as the forger as shown in Figure 6.1. The WIU beacon does

not respond but rather logs the message. This experiment validates the following set of hypotheses:

Hypothesis 1 is validated that no leakage of cryptographic material occurs during the transmission of locomotive or WIU messages.

Hypothesis 2 was validated with the system performance staying consistent at approximately 0.9% CPU utilization worst case and 0.6% memory utilization worst case.

Hypothesis 3 was verified as shown in Table 6.2 and Table 6.1 where the false positive detection rate at approximately 6.9% for a forgery attacker in a non RF noise environment.

6.3.4 Normal Varying Cryptographic Rollover Period

The objective of this test is to verify Hypothesis 3 under varying salt expiration time periods. The cryptographic time interval shrunk from 36 minutes to 3.6 minutes.

Experimental Procedure

The cryptographic roll over the period was set to 216 seconds or 3.6 minutes, three independent experimental runs were conducted. The experiment consisted of transmitting and receiving approximately 500 messages at 1 message every 10 seconds. The cryptographic roll over periods was then shortened until the bit error rate increased. Figure 6.5 plots the effects of decreasing the roll over period to 216 seconds were the false positive detection rate is increased.

Experimental Output

Table 6.1 summarizes data collected.

- RRIDS average detection rate was dropped to an average ∼88% from 36 minutes to 3.6 minutes.

- The CRC error rate remained consistent at approximately 20 CRC errors where as the false positive rate increased an order of magnitude when the time interval shrunk.

It was later discovered that having a clock reference point was important and that both the locomotive and beacon were not time synchronized. The experimental data lead to the conclusion that the false positive rate was dependent on initialization time, specifically, the key generation start time in the transmitter and the receiver.

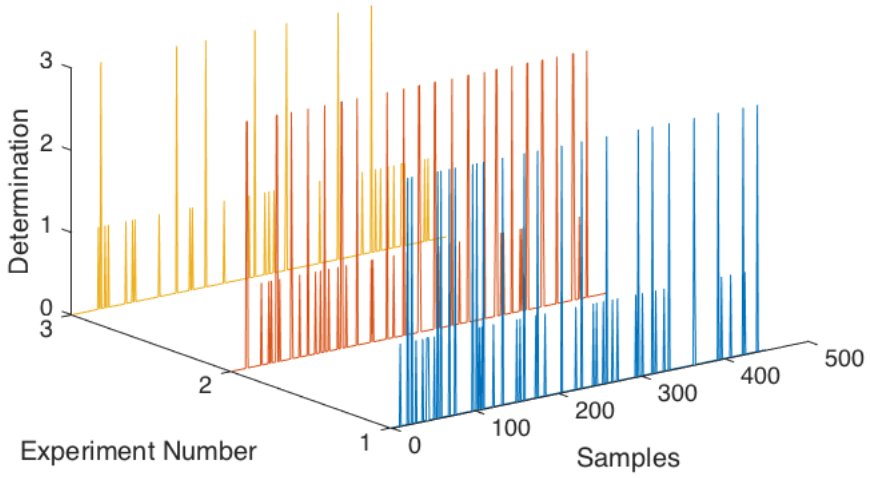


Figure 6.5: IDS determination:Normal Operation with a Cryptographic Interval set to 216 seconds (3.6 minutes)

Validation of the Hypothesis

In experiment 6.3.4 a normal `GetWIU` status messages is sent. When the test completed the time interval was shrunk from 36 minutes to 3.6 minutes.

Hypothesis 3 was verified as shown in Table 6.2 and Table 6.1 that the false positive detection rate increased from approximately 6.2% to 13.2%.

Table 6.1: IDS determination

Test scenario	**Determination count**				**Detection Percentage**	**False Positives**
	0	**1**	**2**	**3**		
Normal 2160 sec Test 1	394	24	0	2	93.8%	6.2%
Normal 2160 sec Test 2	417	26	0	2	93.8%	6.2%
Normal 2160 sec Test 3	414	21	0	2	94.5%	5.5%
Normal 216 sec Test 1	391	28	0	22	88.6%	11.4%
Normal 216 sec Test 2	376	26	0	32	86.8%	13.2%
Normal 216 sec Test 3	396	27	0	17	90.0%	10%
Random Guess Test 1	0	30	0	408	93.1%	6.9%
Random Guess Test 2	0	26	0	399	93.8%	6.2%
Random Guess Test 3	0	25	0	416	94.3%	5.7%
Replay	1	0	463	0	100%	0%
CRC Corrupt	0	442	0	0	100%	0%

6.3.5 Conclusions Experiment Without Noise

The results that were obtained in the experiments lead to the conclusion that RRIDS accuracy are related to time synchronization. In the experiment, an accurate time source was not available and time synchronization was seconds off. Decreasing the time segment intervals between cryptographic material lead to higher false positives. The experiment reducing the cryptographic utilization time from 2160 to 216 seconds, stated in Table 6.1 showed a direct dependency on accuracy. A reference time and an accurate timing source are needed for both cryptographic key generation and RRIDS. Overall, RRIDS functionality was able to detect the attackers outlined in this paper.

The current RRIDS system is designed to detect three types of cryptographic attacks: replay attacks, message corruption attacks, and hash guessing attacks. Preliminary results show that RRIDS system can detect replay attacks, and message corruption attacks with 100% accuracy. Hash guessing attacks were detected 100% as an attack, but approximately 6% are mis-classified as CRC corruption attacks.

Figure 6.7 shows the CPU utilized by the RRIDS for different test cases. As shown in the figure, the CPU utilization is mostly zero where there are spikes of 0.9%. The spike is due to when the processor calculates the salts and algorithms needed prior to communicating. Additionally, the replay test shows occasional utilization of 1.8% CPU. The memory utilization for all the test cases is approximately 0.6%. In normal operation there is a slight increase in memory utilization and in all other cases the memory utilization remains constant. Figure 6.6 and Figure 6.7 show memory and CPU utilization of RRIDS over time.

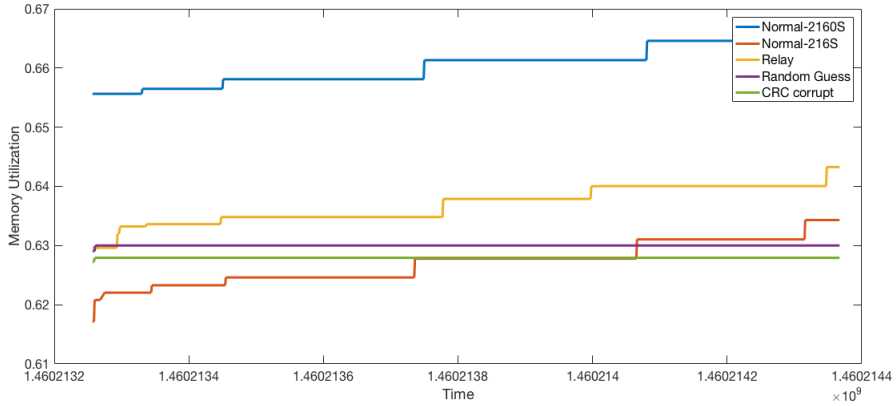


Figure 6.6: Memory utilization by RRIDS during the experiments

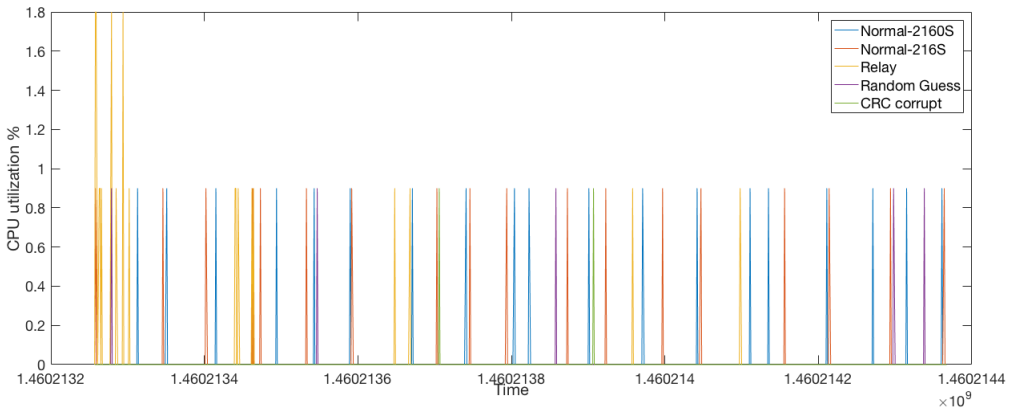


Figure 6.7: CPU utilization by RRIDS during the experiments

Table 6.2: CPU and memory utilization

	Memory utilization(%)		CPU utilization(%)	
	Mean	Standard Deviation	Mean	Standard Deviation
Normal operation -cryptographic rollover period 2160S	0.6605	0.0031	0.0171	0.1229
Replay Attack	0.6369	0.0029	0.0225	0.1620
CRC corruption	0.6279	3.7909e-05	0.0018	0.0402
Random Guess	0.6300	5.9377e-05	0.0045	0.0635
Normal operation -cryptographic rollover period 216S	0.6325	0.0034	0.0171	0.1229

6.4 Experiments With Controlled Noise

Experiments with controlled noise were designed to test the capability of the cryptographic generation schema and RRIDS under controlled constant RF interferences. The time intervals between cryptographic roll over are identical for the locomotive and beacon. Conducted experiments had the locomotive send a `GetWIUStatus` message to the beacon. The beacon in response would send a `BeaconOn` message with the corresponding status. The duration of the test lasted 30 minutes and the locomotive sent messages with a rate of one message every 10 seconds according to the ARA specification [8]. The WIU validates the message and responds at a broadcast rate of one message every 5 seconds. The experiment did not make a distinction between different kinds of WIU messages and focuses solely on the locomotive. The test suite consisted of Figure 6.2 and incorporated a noise generator, an attacker, a locomotive, and a beacon.

6.4.1 Replay Attack With Noise

A replay attacker with noise is defined as one in which both the attacker and noise are present. The cryptographic algorithm and salt are set and the time interval is set to approximately 36 minutes. If the attacker does not have a cryptographic algorithm or salt, it will capture the message and then rebroadcast the captured message. The attacker radio shown in Figure 6.2 overpowers the locomotive radio so that it can synchronize with the receiver. The hypotheses that are verified in this section are Hypothesis 1 and Hypothesis 3.

Experimental Procedure

The cryptographic material was created as described in Section 4.2. The keys are generated for a total duration of 1 day and the time is segmented into 40 time segments. The time segments are equivalent to 2160 seconds or 36 minutes between the cryptographic roll over event. The transmitter and receiver cryptographic key chain generation are started approximately at the same time. The experiment was run independently three times to gather experimental sample data as well as to detect experimental anomalies. The replay attacker

sends a total of 100 replayed messages to the WIU.

Experimental Output

- As shown in Figure 6.8 in the presence of a noise environment the replay attacker was positively identified with 99% accuracy with one lost packet.

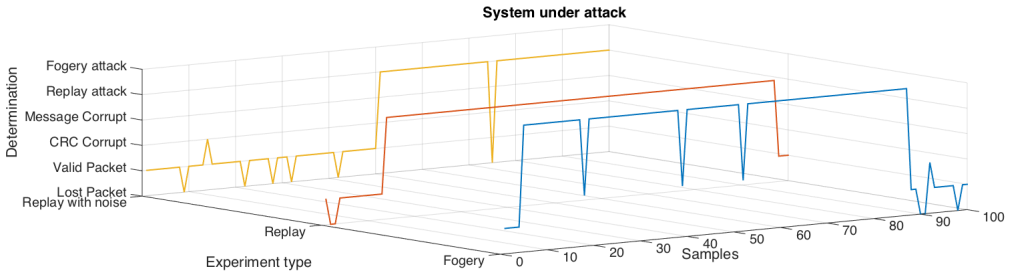


Figure 6.8: Noise based experimentation

Validation of Hypothesis

In experiment 6.4.1, normal `GetWIU` status messages are broadcast from the locomotive to the WIU beacon. The replay attacker overpowers both the locomotive and noise and then synchronizes with the WIU and begins to transmit a captured message. The performance was characterized in Section 6.3 and there was no noticeable difference between a nonnoise environment and a RF environment only Hypothesis 1 and Hypothesis 3 were repeated with calibrated constant noise.

> Hypothesis 1 is validated that no leakage of cryptographic material occurs during the transmission of locomotive or WIU messages.
>
> Hypothesis 3 was verified as shown in Figure 6.8 showing that when the replay attacker synchronizes with the WIU 99% of the replayed messages were properly categorized with one dropped packet.

6.4.2 Replay Attack With No Noise

A replay attacker without noise is defined as one in which both the attacker and other communicating devices are present. The cryptographic algorithm and salt are set and the time interval is set to approximately 36 minutes. The attacker does not have a cryptographic algorithm or salt. Hence the attacker captures the message and rebroadcasts the captured message. The attacker radio shown in Figure 6.2 overpowers the locomotive radio so that it can synchronize with the receiver. The objective of this section is to validate Hypothesis 1 and Hypothesis 3.

Experimental Procedure

The cryptographic material was created as described in Section 4.2. The keys are generated for a total duration of 1 day and the time is segmented into 40 time segments. The time segments are equivalent to 2160 seconds or 36 minutes between the cryptographic roll over event. The transmitter and receiver start generating the cryptographic key chain approximately at the same time. The experiment was run independently three times to gather experimental sample data as well as to detect experimental anomalies. The replay attacker sends a total of 100 replayed messages to the WIU.

Experimental Output

- Figure 6.8 shows the presence of a non noise environment where the replay attacker was positively identified with 100% accuracy with no lost packets.

Validation of Hypothesis

In experiment 6.4.2 a normal `GetWIU` status messages are broadcasted from the locomotive to the WIU beacon. The replay attacker overpowers the locomotive and synchronizes with the WIU and begins to transmit a captured message. Section 6.3 characterizes performance and I did not observe any difference between a nonnoise environment and an RF environment. Hypothesis 1 and Hypothesis 3 were repeated under noise and I analyzed the results to

validate my hypothesis.

Hypothesis 1 is validated that no leakage of cryptographic material occurs during the transmission of locomotive or WIU messages.

Hypothesis 3 was verified as shown in Figure 6.8 showing that when the replay attacker synchronized with the WIU 100% of the replayed messages were properly categorized with one dropped packet.

6.4.3 Forgery Attack With Noise

As described, a forgery attack is one in which the attacker creates a message with the goal to activate and control the WIU. The cryptographic algorithm and salt are set and the time interval is set to approximately 36 minutes. The attacker is assumed to not know a cryptographic algorithm or salt. The attacker radio shown in Figure 6.2 overpowers the locomotive radio so that it can synchronize with the receiver. This set of experiments attempted to validate Hypothesis 1 and Hypothesis 3.

Experimental Procedure

The cryptographic material was created as described in Section 4.2. The keys are generated for a total duration of 1 day and the time is segmented into 40 time segments. The time segments are equivalent to 2160 seconds or 36 minutes between the cryptographic roll over event. The transmitter's and receiver's cryptographic key chain generation starts approximately at the same time. The experiment was run independently three times to gather experimental sample data as well as to detect experimental anomalies. The forgery attacker sends a total of 100 forged messages to the WIU.

Experimental Output

- As shown in Figure 6.8, in the presence of a noise environment the forgery attacker was positively identified with 97% accuracy

- As shown in Figure 6.8, in the presence of a noise environment the forgery attacker was inaccurately mis-categorized 3% of the time as a CRC corrupted packet

6.4.4 Validating Hypotheses

In experiment 6.4.3, a normal `GetWIU` status messages is broadcast from the locomotive to the WIU beacon. The forgery attacker overpowers both the locomotive and noise and then synchronizes with the WIU and begins to transmit crafted messages. Because there was no noticeable difference in performance (as defined in section 6.3) between a non noise environment and an RF environment only Hypothesis 1, and Hypothesis 3 were repeated under varying levels of constant noise.

Hypothesis 1 is validated that no leakage of cryptographic material occurs during the transmission of locomotive or WIU messages.

Hypothesis 3 was verified as shown in Figure 6.8 showing that when the forgery attacker synchronizes with the WIU 97% of the replayed messages were properly categorized with one dropped packet.

6.4.5 Conclusions Drawn from Experiment With Noise

The results that were obtained in the experiments led to the conclusion that regardless if attacker messages are sent in the presence of noise or in the anechoic chamber RRIDS detection rate was approximately 90% accurate given the two types of attackers and a 3% mis-classified as a CRC corruption. In the experiment, the known time interval for the locomotive and WIU was set to 36 minutes. I found that 36 minutes or greater was an ideal cryptographic interval time where greater expiration increases the false positive rate. The lost packet in the replay with noise experiment shown in Figure 6.8 was most likely due to anomalies during experiments.

Chapter 7: Conclusion

The objective of my dissertation work was to provide a secure beaconing framework for emerging intelligent transportation systems [36], where the beacons emitted by infrastructure are used to guide navigation. Due to the differences in application specific details, I chose to use the I-ETMS PTC system as the focus of my study. I-ETMS is the system chosen by the US Freight systems to be operational starting in 2020.

I have provided a possible solution to this problem within the rail network by designing a custom cryptographic generation schema and an intrusion detection system specialized for railway communications. I have also prototyped my system using a laboratory based software defined radio system. Using software defined radios and RADARs is an emerging trend in industrial control systems. I have also demonstrated that my prototyped system satisfies the design objectives by carrying out tests in a variety of RF noise environments as well as the incorporation of attackers. The systems I developed perform well, utilizing minimal CPU, and memory resources to less than 1 percent in both categories. In the presence or absence of an attacker and under calibrated constant RF noise conditions, my prototype system correctly identifies transmitted information.

My research can be extended to contributions in radio forensics, networked enhanced intrusion detection systems for vehicular infrastructure (where a part of the IDS participating in the detection comes from static infrastructures and the other come from the vehicle fleet under service from a radio tower) and real time intrusion detection in radio and RADAR networks. In radio forensics, the information stored in the IDS can be mined in such a way to determine the state of the radio given the received information. RF layer detection would need to be made before the middleware stack and that detection would then be reported to the middleware and propagates to the back-end database. Communicating and reporting intrusion detection events to a network of self reporting systems would provide back-end

office with real-time situational awareness of the communications environment, including the RF environment. The forensics and networked intrusion detection system rely on the real time nature of communications and a lower layer's real time intrusion detection system.

The research areas identified could also be applied to autonomous vehicles. These vehicles will depend on V2V and I2V infrastructures; therefore, their navigational safety will depend on having a secure broadcast system and IDS.

Authored Works

A. Melaragno, D. Bandara, A. Fewell, and D. Wijesekera,*Rail radio intrusion detection system (RRIDS) for communications based control (CBTC)*, IEEE International Conference on Intelligent Rail Transportation, p. 9, August 2016.

D. Bandara, A. Melaragno, D. Wijesekera, and P. Costa, *A Case Study of Cognitive Radio Networks: Secure Spectrum Management for Positive Train Control.* Springer Science, 2016.

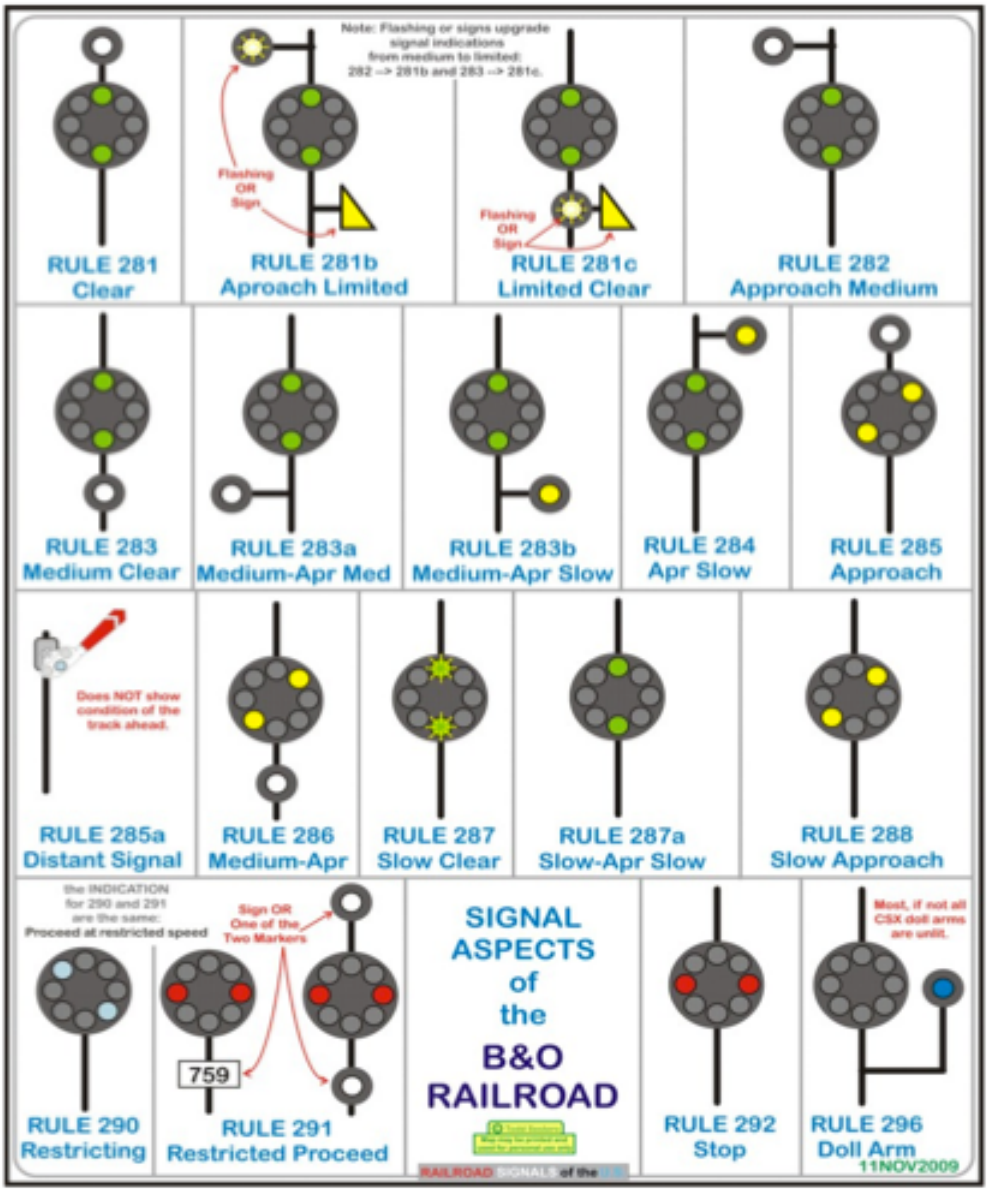
D. Bandara, A. Abadie, T. Melaragno, and D. Wijesekara, *Providing wireless bandwidth for high-speed rail operations*, Procedia Technology, vol. 16, pp. 186 191, 2014.

D. Bandara, A. Melaragno, D. Wijesekera, and P. Costa, *Multi-tiered cognitive radio network for positive train control operations multi-tiered cognitive radio network for positive train control operations multi-tiered cognitive radio network for positive train control operations*, Joint Rail Conference, p. 10, April 2016.

A. Melaragno, D. Bandara, D. Wijesekera, J. B. Michael, *Securing the ZigBee Protocol in the Smart Grid* , Computer, vol. 45, no. 4, pp. 92-94, April, 2012

Appendices

Chapter A: Rail Signal Aspect



Chapter B: Attacker Advantage

Knowns and Assumptions

- Seeds are passed securely to the generator
- There is a temporal component to the communication process which is assumed to be a Markov chain
- Salt is held constant
- Salt length is s^l
- Message length is M^l
- Total Algorithms is A^t

Table B.1: Algorithm Selection

Pseudorandom (PRP)	**True Random**

Input: Algorithms : A, Selection Seed : s_d, Salt$_{guess}$: $s_{guess} \in S$, true hash: t_h
Data: Message : M
Result: Algorithm : α, otherwise : !Found
foreach $i = 1..\alpha_i \in A$ **do**
  $hash \leftarrow \alpha_i(M \oplus (s_{guess} + 1))$
  if *hash equals* t_h **then**
  return α_i
  end
  return *!Found*
end
	Input: Private Algorithm Seed: s_{selected}
	Data: *seed*; A^t
	Result: Hashing Algorithm : α_i
	foreach $i = 1..\alpha_i \in A$ **do**
	  $\alpha_i \leftarrow \frac{1}{A^t}$
	  return α_i
	end

The advantage of the attacker in finding the algorithm is specified in Equation 2.1.

$$ADV(A)_{\alpha}^{PRP} = Pr[REAL_{PRP}^{A} \rightarrow 1] - Pr[RAND_{\{0..n\}^{\text{algorithms}}}^{A} \rightarrow 1] \tag{2.1}$$

I separate the proof into two sections: the left side and the right side. I look at all messages that would result in finding the hash algorithm including collisions. I assume that the algorithm selection process of α_i is derived using a one way hash. Therefore α_i can be restated as Equation 2.2.

$$\alpha_i = hash(s_d \oplus i) mod A^t \tag{2.2}$$

In Equation 2.2 i is the hash algorithm sequence counter and α_i is a selector to the specific hash algorithm therefore it will lie in the range from $\{0 \ldots t\}$ algorithms.

Conceptually α_i can be thought of as a first order Markov chain where the time duration between generation and usage can be set where the reverse direction is computationally infeasible to invert due to the properties of a one-way cryptographic function as shown in Equation (2.3).

$$ADV_f(I) = Pr_{x \leftarrow \{0,1\}^n}[I(1^n, f(x)) \in f^{-1}(f(x))] = negl(n) \tag{2.3}$$

Equation 2.3 indicates that the probability inverting selection seed used to calculate the selection of the algorithm is negligible.

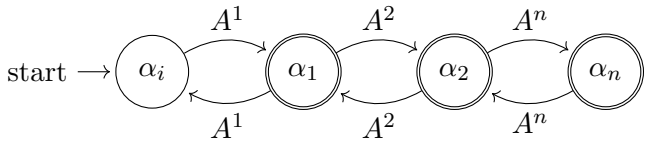


Figure B.1: One way First Order Markov Chain for Key Derivation

Left Side Equation 2.3 state that the advantage is negligible and approaches 0. Consequently, the attacker will be forced into guessing, resulting in the value $\frac{1}{A^t}$.

Proof. **Left Side** $Pr[REAL^A_{PRP} \rightarrow 1]$:

$$ADV_f(I) = Pr_{x \leftarrow \{0,1\}^n}[I(1^n, f(x)) \in f^{-1}(f(x))] = negl(n)$$

Because the modulus operator provides algorithm selection as a uniform distribution, it does not reveal information to the adversary:

$$Pr[REAL^A_{PRP} \rightarrow 1] \leftarrow negl \approx \frac{1}{A^t} \qquad \square$$

Proof. **Right Side** $Pr[RAND^A_{\{0..n\}^{\text{algorithms}}} \rightarrow 1]$:

$$Pr[RAND^A_{\{0..n\}^{\text{algorithms}}} \rightarrow 1] = \frac{1}{A^t} \qquad \square$$

The advantage for the attacker is expressed as:

$$ADV(A)^{PRP}_{\alpha} = |Left - Right|$$

$$ADV(A)^{PRP}_{\alpha} = |\frac{1}{A^t}_{\text{one time use}} - \frac{1}{A^t}| \approx negl \tag{2.4}$$

Chapter C: Formal Modeling

I developed a part of a high-level formal model for the current PTC specification [8]. The results of the formal model is shown in Figure C.1 and Figure C.2. Appendix C shows my modeling of the protocol. The communications between the actors and the attacker can be interpreted as the Locomotive sends a request message to the WIU. The WIU responds appropriately but the salt has been compromised and the attacker begins to forge messages and broadcasts it to the locomotive.

1. $Locomotive \rightarrow Beacon : TimeStamp, PTCMessage, sha1\{TimeStamp, PTCMessage\}$
2. $Beacon \rightarrow Locomotive : TimeStamp, WIU\_Response, sha1\{TimeStamp, WIU_Response\}$

The attack in Figure C.1 is executed against the specification and specifically against the lack of authentication as defined in the WIU protocol. Messages and requests are not validated on the basis of origination which would lead to a possible attack vector.

I use Figure C.1 for this discussion. It is common convention in formal methods to describe the actors as Bob, Alice, and Eve. The Scyther tool assigns the Locomotive (L) to Bob, the Beacon (B) to Alice. The variables and their assignments are as follows:

- $L \rightarrow Bob$: Bob is assigned the role as the locomotive
- $B \rightarrow Alice$: Alice is assigned the role of the WIU Beacon
- Fresh Ta, PTCM : represents a Time Stamp and a PTC message
- SHA1 represent the hash function used as defined by the specification
- Y: defined as the logical result of the hash function

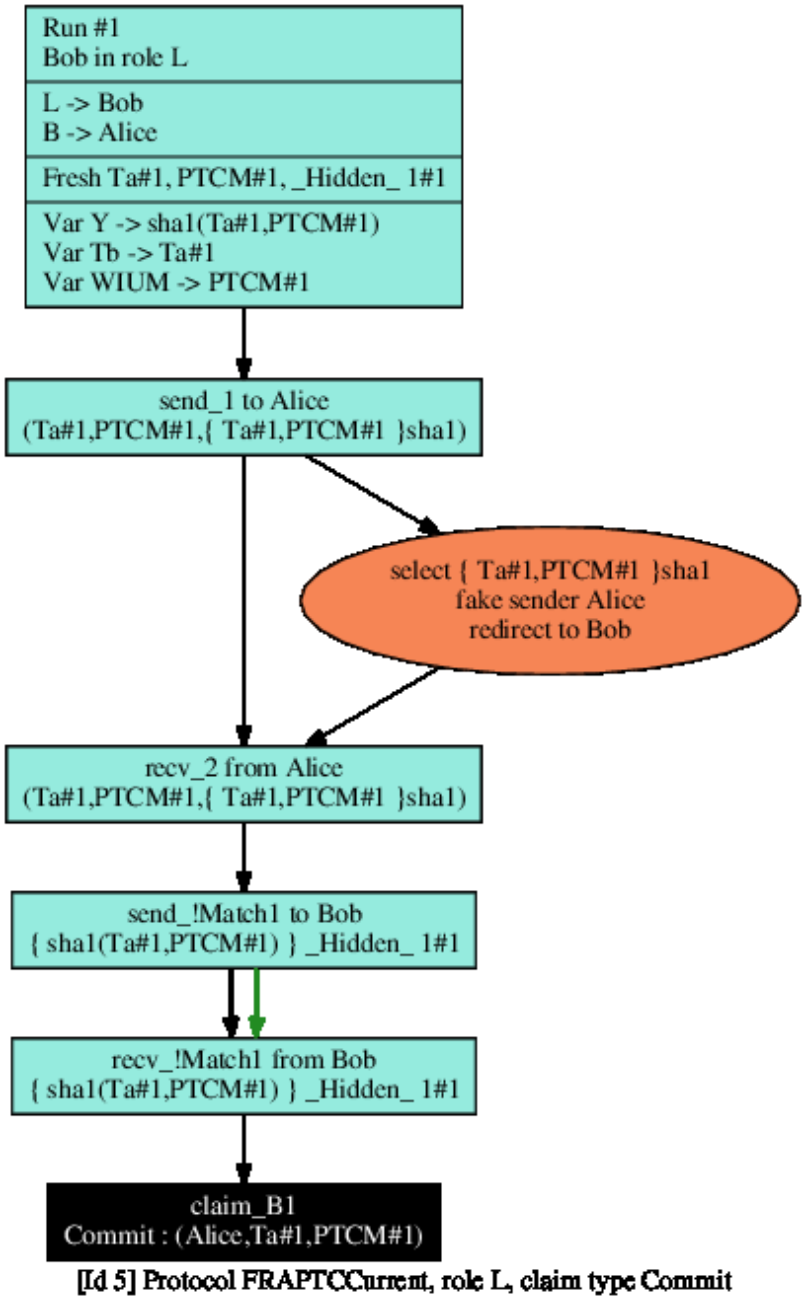


Figure C.1: Locomotive Bob to Beacon Alice Hash Attack Against the Locomotive WIU

Bob initiates communications by sending a message to the beacon, Alice. The intruder (imposter of Alice) redirects the message to Bob, thereby impersonating Alice as the beacon. Likewise, in Figure C.2, Alice's imposter can author fake messages and send it to Bob.

Alice has the capability to create a message and hash messages without detection because confidentiality is not part of the specification [8].

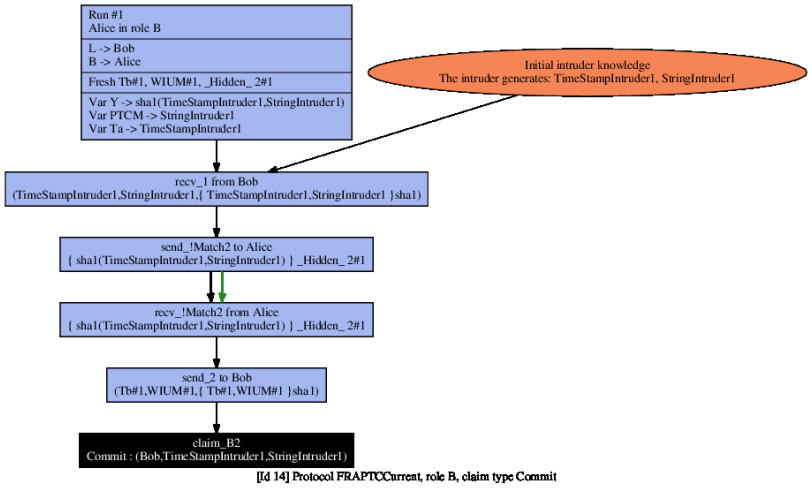


Figure C.2: Fake Beacon Alice to Locomotive Bob Messages Impersonation WIU

Chapter D: TESLA Introduction

The TESLA protocol [2] has three distinct processes to establish communications [2, pp.41]: The Sender Setup, the Receiver Setup, and finally Sending authenticated packets. The following is from [2, pp.41]:

Table D.1: TESLA Protocol Process

1. **Sender Setup**: the sender splits the total communication period into distinct time intervals t_{int}.
   (a) The sender assigns a key per each time interval based on a Pseudo Random Function (PRF).
   (b) The sender randomly selects a Master Key,X_i, and derives a key for the interval i: $K_i = F_X(i)$ and creates a hash tree over all the keys.
   (c) The sender discloses K_i at the end of time interval i for message verification.
2. **Receiver Setup**: The receiver need to synchronize its clock to the senders clock so that the time intervals t_{int} align. Additionally, the receiver clock synchronization knows the maximum clock synchronization error (*disclosure time*) Δ.
3. **Sending Authenticated Packets**: The sender sends a message M within the time intervals t_{int} as:
   (a) $S \rightarrow * : MAC(K_j, M), M, m_j^{`}, ..., m_j^{``}, K_j$

   Once all the messages are received the individual message is verified in step 3a by each K_j hash in the correct order.

The original TESLA uses two distinct time periods for communications. Figure D.1 illustrates the time periods: a bootstrap time period which is used to establish the cumulative delay Δ and the communications period. The MAC is never disclosed and is considered a hidden fact only known to the sender and the receiver the Key though K_j is disclosed at the end at Step 3a in Table D.1.

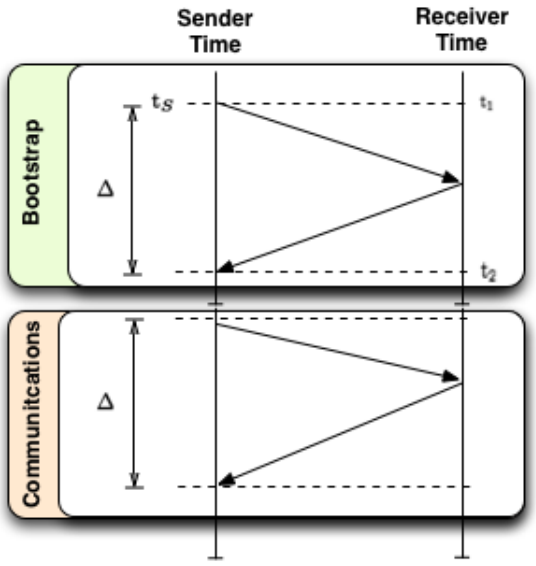


Figure D.1: TESLA Sequence Diagram

During the communications time period, messages are sent from the sender to the receiver. The sender expects that the worst case delay for the receiver to broadcast messages is within Δ. TESLA [2, pp.30] has two distinct requirements:

- Loose time synchronization through via a common time source
- Message pre-buffering

The requirements of TESLA and our enhanced TESLA are essentially the same for time synchronization, authentication and integrity. Messages can then be easily verified for integrity and authentication. TESLA assumes that time is asymmetric and that the receiver is loosely time synchronized with the sender. Time error measurements are made to take into account cumulative delays due to propagation, queuing, and processing delays and is referred to as Δ. The summary of the approach is as follows:

- Time is split up into equal uniform duration time intervals. Each time interval has an associated one way key chain. Each key element is assigned to one time interval. The

one way key chain is derived using Strong Permiage resistance that is computationally infeasible to reverse. A disclosure time is published to the receiver indicating the duration of using the seed.

- A MAC is attached to each packet by the sender and the MAC is computed over the entirety of the data packet. The time interval determines which corresponding one way chain to use for the packet.
- At each receiver the packets are buffered if the receiver can determine that the MAC corresponds to the key since the MAC produced uses the private key in its production.
- If the packet, MAC, and disclosure key are correct then the receiver will then use the packet since the key used corresponds to the proper time interval and key corresponding to the time interval.

Figure D.2 explains the generation of the key chain for a time interval. All time intervals are of equal duration, but within each time interval has sub time intervals in which a specific seed are valid. The sender and the receiver agree on the effective seed for each sub-time intervals. In order to do so, the sender and receiver needs to synchronize their clocks.

Even if packets are lost in communications, the receiver can determine the currently used seed due to the loose time synchronization maintained between the transmitter and receiver.

Unlike TESLA [2, pp.31] my approach is loss tolerant and the receiver does not need to authenticate all received packets once the salts are generated.

Transmitter Setup and Receiver Bootstrapping

Setting up the transmitter and having it communicate to a receiver follows the following process. In Figure D.2, the transmitter begins by dividing the total transmission time into time intervals t_{int}. The transmitter starts at Time interval 1 and creates a key chain from $S_i \ldots S_f$ each of the seeds are then associated to the sub-time intervals and can be thought

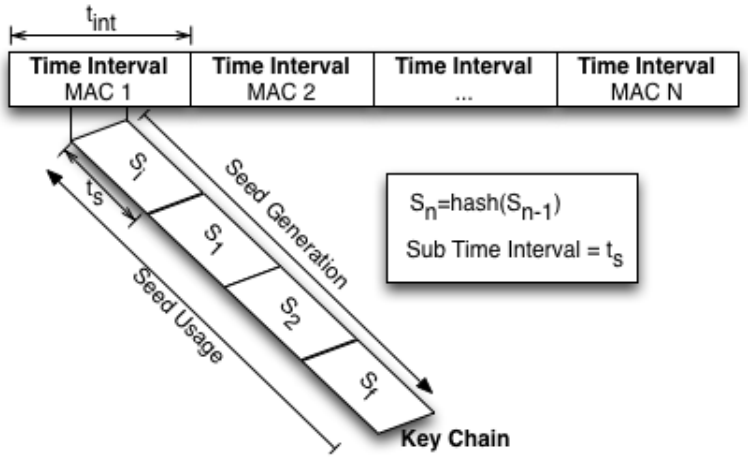


Figure D.2: Introduction to Tesla Seed Generation

of as $t_{int} = 1, t_s \exists \{S_i, S_1, S_2, S_f\}$. The transmitter when initially sends a message it starts with the final salt and uses the key chain in *reverse*. Figure D.2 illustrates the direction of Seed Generation and Seed Usage. Equations 4.1, 4.2, and 4.3 describe the seed generation mathematically According to [2, pp.32], Equation 4.1 is used to generate key or salt sequence that uses a strong permiage resistance and one way hash function.

$$K_i = F(K_{i+1}) \tag{4.1}$$

Additionally, [2, pp.32], uses a pseudo-random generator to construct a one-way function. The one way function referenced in Equation 4.2 is used to generate the seeds used in the key chain.

$$F : F(k) = f_k(0) \tag{4.2}$$

Perrig [2, pp.32] concludes that Equations 4.1 and 4.2 reduce to Equation 4.3. Perrig [2, pp.32] says that given any time interval i, a seed can be derived from the MAC.

$$K_i = F^{N-i}(K_N) \tag{4.3}$$

Perrig concludes that Equation 4.3 summarizes the capability to reference a time interval and derive the necessary key needed for both the time interval and sub-time interval. Because time is the main component for synchronization dependency, transmission, propagation, queuing and processing delays are measured and used to a correction value called Δ [2, pp.32-33] during the transmitter and receiver bootstrap processes.

According to [2, pp. 32], TESLA offers three types of time synchronization methods, direct, indirect, and delayed. Time synchronization requires that the receivers know the upper bound Δ for the receiver. An assumption made in TESLA [2, pp.40] is that relative clock drift between the sender and receiver is negligible. The assumption holds for PTC because we use GPS governed clocks in the WIU network.

D.0.1 Direct Time Synchronization

Direct time synchronization as defined by [2, pp. 40] is where the receiver and transmitter are time synchronized using a physical clock an example a shared network time server would be considered direct time source. Because the transmitter and receiver are clock synchronized, Δ is known precisely. Direct Time Synchronization is used by hardwired computer networks [2].

D.0.2 Indirect Time Synchronization

According to [2, pp.43], Indirect Time Synchronization is the process that the transmitter and receiver are independently synchronized by a time reference. Independent clocks such as GPS is used as a time synchronization reference source. Because the internal clock within a receiver can drift due to oscillator errors, a maximum synchronization error is introduced by [2] to provide a value to the synchronization error Δ_{SC}. According to [2] an upper bound is calculated by the receiver as Equation 4.4.

$$t_s \leq t_r - t_R + t_c + \Delta_{SC} \tag{4.4}$$

The transmitter periodically sends Δ_{SC} to the receiver. I modified Indirect Time Synchronization for PTC communications, because locomotives and the PTC network utilizes GPS as a time synchronization source. Perrig [2, pp.45] defines disclosure delay as the amount of time that is needed to have the key disclosed to the communicating device that is used for seeding. In my case, the locomotive and the beacon are reloaded with keys prior to travel; therefore, this is not an issue.

Limitations of TESLA

The limitations of TESLA is the seed starvation and the re-initiation of a bootstrap process [2] for continuous communications. In the PTC network, re-initiation of a bootstrap process would pose a serious issue with continuous communications. The possibility of a man in the middle attacker during the bootstrap process has been described in [1].

Bibliography

[1] D. J. Scott, "Relying on time synchronization for security in ad hoc networks," in *Proceedings of the 43rd Annual Southeast Regional Conference - Volume 2*, ser. ACM-SE 43. New York, NY, USA: ACM, 2005, pp. 87–91. [Online]. Available: http://doi.acm.org/10.1145/1167253.1167273

[2] A. Perrig and T. J.D., *Secure Broadcast Communication in Wired and Wireless Networks*. Assinippi Park, Norwell Massachusetts 020601 USA: Kluwer Academic Publishers, 2002, vol. 1.

[3] D. Bandara, A. Melaragno, D. Wijesekera, and P. Costa, "Multi-tiered cognitive radio network for positive train control operations multi-tiered cognitive radio network for positive train control operations multi-tiered cognitive radio network for positive train control operations," *Joint Rail Conference*, p. 10, April 2016.

[4] U. S. C. Guard. (2016, 02) Automatic identification system. [Online]. Available: http://www.navcen.uscg.gov/?pageName=AISworks

[5] (2016, 02). [Online]. Available: https://www.faa.gov/nextgen/

[6] J. council on transit wireless communications, "Positive Train Control-White Paper," May 2012.

[7] (2016, 07). [Online]. Available: http://www.cnn.com/2016/02/16/europe/germany-train-collision/

[8] A. of American Railroads, *Interoperable Train Control Wayside Interface Unit Requirements*. Association of American Railroads, 2010, vol. Version 1.2.3 02/24/2010, no. Issue of 2010.

[9] gnu radio (www.gnuradio.org), "Gnu radio is a free open-source software development toolkit that provides signal processing blocks to implement software radios."

[10] redis.io, "Redis is an open source (bsd licensed), in-memory data structure store, used as database, cache and message broker."

[11] B. Schneier, "Sha-1 broken."

[12] K. Bhargavan and G. Leurent, "Transcript collision attacks: Breaking authentication in tls, ike, and ssh," *Internet Society*, vol. ISBN 1-891562-41-X, 2016.

[13] M. Stevens, P. Karpman, and T. Peyrin, "Freestart collision for full sha-1," Cryptology ePrint Archive, Report 2015/967, 2015, http://eprint.iacr.org/.

[14] M. W. Hartong, "Secure communications based train control (cbtc) operations," Ph.D. dissertation, George Mason University, 2009.

[15] M. Hartong, "Secure communications based train control (CBTC) operations," Ph.D. dissertation, George Mason University, Fairfax, Virginia, Sep. 2009.

[16] A. Amanna, M. Gadhiok, M. J. Price, J. H. Reed, W. P. Siriwongpairat, and T. K. Himsoon, "Rail-CR: Cognitive Radio for Enhanced Railway Communication." ASME, 2010, pp. 467–473.

[17] D. Bandara, A. Abadie, T. Melaragno, and D. Wijesekara, "Providing wireless bandwidth for high-speed rail operations," *Procedia Technology*, vol. 16, pp. 186 – 191, 2014.

[18] A. Abadie, "Combining Operational and Spectrum Characteristics to Form a Risk Model for Positive Train Control Communications," Ph.D. dissertation, George Mason University, 2014.

[19] D. Bandara, A. Melaragno, D. Wijesekera, and P. Costa, *A Case Study of Cognitive Radio Networks: Secure Spectrum Management for Positive Train Control.* Springer Science, 2016.

[20] K. R. D. S. Bandara, A. Abadie, and D. Wijesekara, "Cell Planning for High-Speed Train Operations in USA." ASME, Mar. 2015, p. V001T03A007.

[21] D. Bandara, A. Abadie, T. Melaragno, and D. Wijesekara, "Providing wireless bandwidth for high-speed rail operations," *Procedia Technology*, vol. 16, pp. 186–191, 2014.

[22] A. Abadie, D. Bandara, and D. Wijesekera, "A Composite Risk Model for Railroad Operations Utilizing Positive Train Control (PTC)." ASME, Apr. 2014, p. V001T06A004. [Online]. Available: http://proceedings.asmedigitalcollection.asme.org/proceeding.aspx?doi=10.1115/JRC2014-3730

[23] April 2015. [Online]. Available: http://thehackernews.com/2015/04/hacking-train-crash.html

[24] BBC, "Train-switching technology 'poses hacking threat'."

[25] gatomalo, "Hacking the rail network."

[26] R. A. K. Manoj K. Jha, "Evaluating cyber attacks in rail transit," *Information Resources Management Association*, 2014.

[27] L. Chen, "Nist comments on cryptanalytic attacks on sha-1."

[28] S. Dietzel, R. van der Heijden, H. Decke, and F. Kargl, "A flexible, subjective logic-based framework for misbehavior detection in v2v networks," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014 IEEE 15th International Symposium on a*, June 2014, pp. 1–6.

[29] W. Bamberger, J. Schlittenlacher, and K. Diepold, "A trust model for intervehicular communication based on belief theory," in *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, Aug 2010, pp. 73–80.

[30] A. C. Simpson and J. Jacobs, "On the cloud-enabled refinement checking of railway signalling interlockings," in *Proceedings of the 2nd International Workshop on Formal Techniques for Safety-Critical Systems (FTSCS 2013)*, ser. Communications in Computer and Information Science, vol. 419, 2014, pp. 195–211.

[31] May. [Online]. Available: http://www.wsj.com/articles/deadly-train-wreck-in-philadelphia-leaves-disastrous-mess-1431499608

[32] D. A. Graham, "'preventable tragedy': Amtrak 188 and the case for positive train control."

[33] J. A. V. N. T. S. Center, "Vulnerability assessment of the transportation infrastructure relying on the global positioning system," John A. Volpe National Transportation Systems Center, Tech. Rep., 2001.

[34] J. Wright and M. Manic, "Time synchronization in hierarchical tesla wireless sensor networks," in *Resilient Control Systems, 2009. ISRCS '09. 2nd International Symposium on*, Aug 2009, pp. 36–39.

[35] Wikipedia, "Universal software radio peripheral," 2014. [Online]. Available: $https://en.wikipedia.org/wiki/Universal\_Software\_Radio\_Peripheral$

[36] A. Melaragno, D. Bandara, A. Fewell, and D. Wijesekera, "Rail radio intrusion detection system (rrids) for communications based control (cbtc)," *IEEE International Conference on Intelligent Rail Transportation*, p. 9, August 2016.

Curriculum Vitae

Anthony Melaragno is a senior software and systems engineer who is currently working on a variety of hardware and software projects. He has extensive knowledge in software development for both real time and non real time systems. He has contributed to securing United States infrastructure for rail safety. He received a Masters of Science in Telecommunications from George Mason University in 2004, as well as a Bachelors of Science in Electrical Engineering from The Pennsylvania State University in 1997.