A Systematic Development of a Secure Architecture for The European Rail Traffic Management System

by

Richard James Thomas, BSc. (Hons), MSc.

A Thesis submitted to
University of Birmingham
for the Degree of
Doctor of Philosophy

School of Computer Science
College of Engineering & Physical Sciences
University of Birmingham
November 2018

Abstract

The European Rail Traffic Management System (ERTMS) is a new signalling scheme that is being implemented worldwide with the aim of improving interoperability and cross-border operation. It is also an example of an Industrial Control System, a safety-critical system which, in recent years, has been subject to a number of attacks and threats. In these systems, safety is the primary concern of the system designers, whilst security is sometimes an afterthought. It is therefore prudent to assure the security for current and future threats, which could affect the safe operation of the railway.

In this thesis, we present a systematic security analysis of parts of the ERTMS standard, firstly reviewing the security offered by the protocols used in ERTMS using the ProVerif tool. We will then assess the custom MAC algorithm used by the platform and identify issues that exist in each of the ERTMS protocol layers, and aim to propose solutions to those issues. We also identify a challenge presented by the introduction of ERTMS to National Infrastructure Managers surrounding key management, where we also propose a novel key management scheme, TRAKS, which reduces its complexity. We then define a holistic process for asset owners to carry out their own security assessments for their architectures and consider the unique challenges that are presented by Industrial Control Systems and how these can be mitigated to ensure security of these systems.

Drawing conclusions from these analyses, we introduce the notion of a 'secure architecture' and review the current compliance of ERTMS against this definition, identifying the changes required in order for it to have a secure architecture, both now and also in the future.

Acknowledgements

A fascination with trains is a fascination with journeys: they have a platform from which to depart, a clear direction of travel and a destination at the end. This particular adventure may not have always had a clear direction of travel, but it certainly brought many passengers on the journey. To say passengers, in fact, is to diminish the role of these individuals: to get to this destination, there are many without whose support I simply could not have 'gotten' this train to the platform on time.

I extend thanks to Chris Hankin and Ian Batten, my examiners, and to my Thesis Group, including Eike Ritter and Mark Ryan who, throughout my studies, have proffered wisdom and guidance which has ultimately shaped this Ph.D.

I would like to give special thanks to my Ph.D. (and Master's) Supervisor, Tom Chothia, where our countless hours thinking of new ideas and understanding hefty volumes of specification documents led to my work having a sense of impact. This was partly due to the SCEPTICS Project, where the collaboration in this project with John Easton, Joeri de Ruiter and Clive Roberts has reaffirmed my desire to stay in academia – there is still a lot more to do.

My thanks are also given to the people I've met during my time at Birmingham, who have provided support in different forms: Andreea-Ina Radu, Chris McMahon Stone, Jan van den Herrewegen, my officemates in Room 117, and my colleagues at the University. I cannot thank my friends Sam and Chris enough – even in the threat of derailment, you all have seen me through to the end.

Finally, to my family, I owe the greatest of thanks, for they have been on the journey with me from the very beginning. To June and Paul, my parents, Emma, my sister (and Ph.D. ahead of me – she does not allow me to forget) and Susie: your unconditional love, encouragement and patience has helped me get to this destination, even if patience was tested at the somewhat . . . arduous proofreading stage.

I said that a fascination with trains was a fascination with journeys: my grandparents John and Edna Butt set my heart on doing something with the railways – our times on the Severn Valley Railway, the bacon sandwiches and your stories live on, not just throughout this thesis, but in me as well.

Partial Funding for this Ph.D. and Research Project was made possible by the strategic partnership on railway data between Network Rail and the University of Birmingham and the Engineering Physical Sciences Research Council (EPSRC) under the SCEPTICS Project, Grant Number EP/M002845/1.

"Most men who have really lived have had, in some share, their great adventure.

This railway is mine."

James J. Hill, *Railway Pioneer*

Table of Abbreviations

3DES	Triple DES
AACS	Advanced Access Content System
API	Application Programming Interface
ATO	Automatic Train Operation
CBC	Cipher Block Chaining
CBTC	Communication Based Train Control
CIA	Confidentiality, Integrity and Availability
CNI	Critical National Infrastructure
COTS	Commercial off-the-shelf
CRC	Cyclic Redundancy Check
CSR	Cab Secure Radio
CVE	Common Vulnerabilities and Exposure
CVSS	Common Vulnerability Scoring System
DAS	Driver Advisory System
EB	EuroBalise
ERA	European Rail Agency
ERTMS	European Rail Traffic Management System
ETCS	European Train Control System
ETSI	The European Telecommunications Standards Institute
EVC	European Vital Computer
GPRS	General Packet Radio Service
GPU	Graphics Processing Unit
GSM	Global System for Mobile Communications
GSM-R	Global System for Mobile Communications for Railway
HSM	Hardware Security Module

HMI	Human Machine Interface
ICS	Industrial Control System
KMC	Key Management Centre
LEU	Lineside Electronic Unit
LRBG	Last Relevant Balise Group
LFSR	Linear Feedback Shift Register
MA	Movement Authority
MAC	Message Authentication Code
MMI	Man Machine Interface
NIS	Network and Information Security
NIST	The National Institute for Standards and Technology
OBU	On-Board Unit
OES	Operator of Essential Service
OT	Operational Technology
PKI	Public Key Infrastructure
PLC	Programmable Logic Controller
PRF	Pseudo-random Function
RBC	Radio Block Centre
RCM	Remote Condition Monitoring
ROC	Rail Operation Centre
SaF	Safety Feature
SaPDU	Safety Protocol Data Unit
SCADA	Supervisory Control and Data Acquisition
SDR	Software Defined Radio
SIL	Safety Integrity Level
TASS	Tilting Authorisation and Speed Supervision
TMSI	Temporary Mobile Subscriber Identity
WCML	West Coast Main Line

Contents

Chapter 1

Introduction

The United Kingdom (UK) rail network is undergoing a major transformation from legacy systems, some of which date back to the Victorian rail 'revolution', to the digitisation of more components from the track to the train itself. The objective of this transformation is to increase line capacity, improve punctuality, simplify maintenance and improve reliability whilst further developing interoperability to allow cross-border operations.

The new solution, which is in the process of being implemented, is the European Rail Traffic Management System (ERTMS). This platform comprises the European Train Control System (ETCS), a suite of protocols and standardised applications for in-cab signalling and the Global System for Mobile Communications for Railway (GSM-R) wireless communications protocol for train to trackside messaging.

With the increased digitisation of our railways, we must ask the question: "what threats may exist to the railway and what are the future risks that could arise?". ERTMS is a standard born out of a European Union (EU) Directive which has roots as early as 1997 [22]. We see, in other sectors, such as telecommunications that, if the security of a system is not continuously reviewed (for example, GSM's primary encryption cipher, A5/1 is now considered to be broken [100]), or the attack model is not evolved to consider today's attacker capabilities [41], it is

possible for attacks to have a higher likelihood of success. When we consider the rail network, systems are designed to fail safe. If an attacker had the ability to compromise a train, they have won the game where the safety of the train and rail network could also be subsequently compromised.

We therefore need to carry out a detailed analysis of the rail network from a security perspective to understand where lapses may have occurred and consider a factor that affects all Industrial Control System (ICS) environments – their inherent lifespan. ICS systems, compared to typical commodity equipment, have a much longer lifespan – in the order of decades, rather than years. This presents a problem to ICS owners when considering the security of their systems, where there is a gap between what was previously considered secure compared to what may now be insecure. It is also important to look at the future of these systems. Specifically, security decisions should be modular, so that if a vulnerability arises, there is an opportunity to address the exposure. This is where safety and security meet and are intrinsically linked.

The ERTMS standards have been developed by a large number of parties representing rail equipment vendors, Infrastructure Managers and the European Rail Agency (ERA). The standards are spread over 50 separate specification documents, totalling over 1,000 pages, with a particular focus on backwards compatibility and interoperability. With this volume of documentation, there is the risk of ambiguity in the specification, significant complexity and critical details (particularly for the security community) to become hidden in other technical details.

1.1 The Need for Assurance

As we have seen in widely-used code libraries, e.g. OpenSSL [149], for a stack of safety-critical protocols and applications, it is possible that any implementation of a new system could have inherent errors or weaknesses. Conversely, what if it was the underlying standard that contained and introduced those weaknesses? Any implementations of the standard would there-

fore be insecure. It is therefore essential that the standards and specifications which drive implementation by vendors provide a security assurance and any potential weaknesses are identified and addressed prior to implementation. In this section, we will highlight the contributions of this thesis to the development of a framework to formally analyse the standards and their relative security.

For protocols, techniques such as formal verification using mathematical modelling can be employed to verify the correctness of a protocol and ensure that there are no security weaknesses. In this thesis, we will show how formal verification can be used to analyse the compositional security of the EuroRadio and Application Layer train to trackside protocols against a given set of security properties. These properties should hold, especially for a safety-critical system where any potential exposure must be addressed.

However, having a protocol shown to be secure is only part of the assurance process. For example, whilst a protocol may be secure against replay attacks or prevents cryptographic keys from being leaked, it may be the case that the cryptography ultimately fails the security of the system. For systems with a long operational lifespan such as those used in Critical National Infrastructure (CNI) applications, cryptography may be used to provide message authentication. However, if the cipher itself is weak, the attacker has another, new, way of compromising the system itself. Cryptanalysis is an established means of assessing the security of ciphers and is used in this thesis to highlight weaknesses in the custom-defined Message Authentication Code (MAC) cipher used in the EuroRadio protocol.

Finding attack vectors and vulnerabilities in standards and specifications provides a level of assurance to owners, operators, vendors and regulatory bodies, enabling them to address any security issues. However, in some cases, whilst a viable threat may not be conceivable today, it is essential that the potential future exposure is also assessed and consideration given to the future security of the system. This thesis will analyse the way that cryptographic keys are handled in ERTMS and propose a new scheme that considers and addresses future threats

to the platform, in particular from attackers with quantum computing capabilities. As an example, which we will review in detail later in this thesis, post-quantum resistance is critical for new deployments of Supervisory Control and Data Acquisition (SCADA) systems, due to their long operational lifespan. Whilst today, an attacker could intercept traffic without having the quantum ability to break certain public-key schemes, in a post-quantum world, they would be able to recover keys with relative ease.

Conversely, we must assess the process used to install these cryptographic keys on equipment and its relative security. As mentioned previously, there are subtle differences between when ERTMS was initially ratified and subsequently adopted. The current offline ERTMS scheme is currently in the process of being replaced with an online, Public Key Infrastructure (PKI)-based scheme. However, as previously discussed, we need to ensure that it remains secure for the future by analysing and assessing potential areas of exposure including human factors in key management. Unlike protocols and cryptography, where implementation errors may occur, key management may also involve a human element, e.g. to transport keys, or carry out an authorisation. This element, therefore, needs to be equally assessed and improvements identified which minimise the possibility of poor practices becoming the norm or prevent social engineering attacks which are similarly of concern [101].

Finally, we should also consider the asset owners and determine ways that they can improve the security of their systems. One of the biggest threats to any architecture is where the asset owner simply does not have a comprehensive understanding of their system. There may be some knowledge siloed in the organisation or, through organisational change, the knowledge may no longer be held. For an adversary, this means that they could look for potential entry points which the asset owner is either unaware of, or the level of interconnectivity in the system has not been appraised. Again, if the adversary is able to gain a foothold into the system, they may be able to follow these unrealised connections between systems. With the introduction of the EU Network and Information Security (NIS) Directive, it is essential

for system owners to confidently assure the security of their systems, especially where the security models and adversary capabilities are constantly evolving.

All of these points pose the question: how can we confidently assure the security of our systems? Using Definition 1.1 below, we will try to answer this question in the forthcoming chapters of this thesis and demonstrate how vulnerabilities found in each part of the ERTMS stack mean that this definition does not hold. Through analysis, proposed improvements and mitigations, we are, however, able to demonstrate that we can provide a secure architecture for ERTMS. This Definition primarily concerns itself with the security of individual components but also considers the compositional security of an architecture.

Definition 1.1 (Secure Architecture of an Industrial Control System) *We consider an Industrial Control System (e.g. ERTMS) as having a 'secure architecture' if it has the following properties:*

- *Each component is verified for its security posture.*
- *Critical components (e.g. SIL3/SIL4) have a clearly defined security profile and measures have been taken to minimise the risk of exploitation.*
- *Protocols and cryptographic proposals are verified and the security assured for the lifespan of the platform.*
- *Messages and data are provided sufficient protection such that an attempt by an active attacker is always detected and the presence of a passive attacker does not compromise the security or safety of the system.*

Safety Integrity Levels (SILs) define risk thresholds for a given system, and the confidence placed in the safety performance of that system. SIL levels are extensively used in safety-critical applications, where four SIL Levels (further explained in Chapter 2) specify the likelihood of failure for a system.

1.2 Thesis Overview

In this thesis, we will 'dive down' the ERTMS stack from the perspective of the train and the Radio Block Centre (RBC), before returning back up the stack, reviewing the security of the standards from a holistic viewpoint and considering the existing and potential future threats to the platform. This thesis is structured as follows:

Chapter 2 Here, we outline the necessary technical background for ERTMS, its underlying protocols and architecture and the evaluation methods and tools that will be applied in this thesis. We will also review the EU Network and Information Security (NIS) Directive, its impact on system owners and the interplay between safety and security.

Chapter 3 From the overview of ERTMS and analysis techniques, we will review the existing literature, including previous and related work in this area of research and discuss the key challenges and considerations that exist in ICS and the rail sector. Whilst ERTMS security research is an evolving area, we will also look at related ICS sectors for inspiration and touch on real-world applications of analysis techniques, for example, in the financial sector.

Chapter 4 Building on the work from the previous chapter, we will start to 'dive down' the ERTMS stack, firstly analysing the protocols that are used for ERTMS train to trackside communications. The work presented in this chapter is an extension to a previously-submitted MSc. Thesis to the University of Birmingham, and is based on the publication:
A Formal Analysis of ERTMS Train to Trackside Communications, by the author, Tom Chothia and Joeri de Ruiter [51], presented at RSSRail in 2016.

The extensions to the MSc. Thesis include further analysis of the high-priority messages that are used within EuroRadio and a thorough analysis of the Application Layer protocol, including its use of timestamps.

Chapter 5 Chapter 4 only considers attacks against the EuroRadio and Application Layer protocols, where ProVerif assumes the use of cryptography to be perfect (i.e. it is implemented

correctly and has no vulnerabilities). However, this is not sufficient, as the assurance of the underlying cryptography must be established. In this chapter, we will review the EuroRadio MAC algorithm in detail and show how an attacker can leverage flaws in each of the train to trackside protocol layers to forge their own messages which would be accepted by the train. This chapter is based on the publication:
An Attack Against Message Authentication in the ERTMS Train to Trackside Communication Protocols by the author, Tom Chothia, Mihai Ordean and Joeri de Ruiter [41], presented at AsiaCCS 2017.

Chapter 6 What is subtly highlighted in Chapters 4 and 5 is that key management in ERTMS, required for the EuroRadio MAC and protocol to authenticate trains and RBCs, is also a challenge, with new proposals coming forward, as we observe from Chapter 3. In this chapter, we will review the existing ERTMS key management scheme, an offline, country-specific system and its proposed online successor, identifying potential improvements that could be made and relating these to Definition 1.1 by introducing the EuroBalise into the secure architecture. The work presented in this chapter is based on the publication:
TRAKS: A Universal Key Management Scheme for ERTMS by the author, Tom Chothia, Mihai Ordean and Joeri de Ruiter [133], presented at ACSAC 2017.

Chapter 7 Looking forward, given the issues identified in the previous chapters, we are now in a position to define a framework that allows asset owners to assess the overall security of their own systems. By using a model-based approach and the Common Vulnerability Scoring System (CVSS) framework, ICS operators can rationalise and reason about the security of their systems and identify potential weaknesses and opportunities for improvement. This chapter is based on the publication (currently under submission):
The SCEPTICS Tool for Threat Analysis in Industrial Control Systems by the author, Tom Chothia and Mihai Ordean.

Chapter 8 When considering the security of ICS, there are a number of obstacles which do not exist in other settings, specifically the design lifespan of systems and the requirement for continuous availability. In this chapter, we will look back at the marriage of security and safety and the future threats that could become possible in the lifetime of ERTMS. We will then consider one important, perhaps unique, challenge that ICS and widely-deployed systems present, namely where recommendations need to work alongside the existing standards before looking at future areas of work that can be explored.

Chapter 9 We will look back at Definition 1.1 and review how each chapter contributes to the development of a secure architecture for ERTMS and conclude this thesis.

Chapter 2

Background and Preliminaries

In this chapter, we will review, at a high level, the necessary technical details required for the remainder of this thesis, including a background of the current signalling system deployed in Great Britain, followed by an overview of ERTMS, its associated protocols and standards and the methods we can use to to assure the security of these components. For the purposes of this thesis and to directly relate to the standards, the United Kingdom refers to the scope of the Department for Transport (DfT), whilst Great Britain (GB) refers to England, Scotland and Wales only.

2.1 Existing Signalling in Great Britain

Currently, with the exception of sections of the Great Western main line, running from London to Wales and sections of Thameslink, the rail network signalling system in Great Britain is largely standardised to allow both passenger and freight services to operate on the same infrastructure. The Great Western main line operates a different system, known as Automatic Train Protection (ATP), an advanced system which supervises the train to ensure it does not pass a signal 'at danger' or exceed the permitted line speed. This system was introduced following the Hatfield rail accident, resulting in 4 fatalities and 70 casualties. Thameslink operates on a

combination of the European Rail Traffic Management System (ERTMS) in its 'core' and the conventional signalling system, track circuit block signalling, deployed throughout the rest of Great Britain.

In block signalling, the rail line is split into a number of blocks. A train cannot enter a block that is already occupied by another train, as a safety envelope is specified to ensure that a train is able to stop before it overruns into the next block when given a 'warning' aspect. These blocks are typically of variable length, dependent on a number of factors including line speed, proximity to stations, and other operational considerations. In Great Britain, a number of interconnected components are used to provide signalling and deliver a fully-managed and safe rail network.

2.1.1 Architecture

Compared to some deployments in Europe, the Great Britain signalling system has largely remained unchanged with lineside signals and other physical infrastructure providing 'on sight' signalling authorities. In contrast, for example in France and Germany, in-cab alternatives exist which allow for increased line speeds.

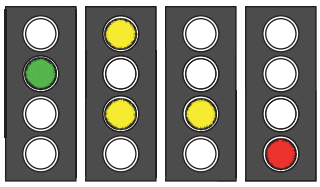


Figure 2.1: Example four-aspect signal, deployed in the Great Britain rail network, progressing from a 'proceed at line speed' (green) to a 'danger' (red) aspect.

Lineside signals are typically located to the side of the rail lines (shown in Figure 2.1) or mounted on gantries to display signalling 'aspects' to the driver, providing permission to proceed into the next supervised block. These signals can be supplemented with 'route indicators', informing the driver of the rail line or platform to which the train will be directed. Other as-

pect signals exist (e.g. three, two and single-aspect), which may be deployed on less utilised lines.

All signals of this type are connected to a fixed cable network through to a lineside cabinet, receiving information from a central control system or signal box determining which aspect to display.

As a train proceeds from one block to the next, it is considered to occupy that block and have cleared the previous block. In Great Britain, train occupancy can be determined in one of two ways, using track circuit detection or axle counters. Track circuit detection works by passing an electrical signal down one rail, where the train wheels will transfer the signal (due to their conductive nature) to the opposite rail. This signal is detected by a relay placed alongside the track, showing whether or not a section of track is occupied. It should be noted that whilst a line can be shown to be occupied, there is no way to identify where in the block a train is located, as only a positive or negative response is returned. Axle counters are devices directly attached to one of the rails, counting the number of axles passing through them. The way this technique works is that when a train moves into a new, unoccupied block, the number of axles on the train are counted in, and as it leaves a block, they are counted out. If the number of axles counted out equals the number of axles counted in, then the block is no longer deemed occupied, where another train may enter that section of track. In some areas of the UK, a combination of track circuit detection and axle counters are deployed, where both detection systems are complementary to each other.

2.1.2 Issues with the Existing Signalling System

As previously identified, the current signalling system deployed in Great Britain has remained largely unchanged in recent decades. Reliability, capacity and maintenance overheads are now a motivation to move to in-cab signalling solutions, including the European Rail Traffic Management System (ERTMS).

One of the main issues with lineside signals is the requirement for a driver to be able to see and process a signal aspect at speed. The West Coast Main Line (WCML) is considered to be one of the busiest and most congested rail lines in Europe [34], but relies on ageing infrastructure. The reliance on being able to 'read' signals at speed, limits the possible speed that trains can operate, restricting the capacity of the line. Another issue is associated with the required safety envelope to ensure that *any* train can stop safely within a block, leading to the term 'chasing signals', where drivers move from one red aspect to the next, introducing delays.

The reliability of this infrastructure is also another consideration where signal failures are a common occurrence.The replacement of this infrastructure with digital, in-cab solutions would not only reduce maintenance cost, but also improve the reliability of the signalling system due to the reduced number of components and exposed infrastructure.

2.2 European Rail Traffic Management System (ERTMS)

The European Rail Traffic Management System (ERTMS) is a European standard for 'next-generation' train traffic management and signalling, composed of the European Train Control System (ETCS) and GSM-R. Its primary aim is to improve interoperability for cross-border operation and optimise railway operations. A prime case-study for the benefits delivered by this standard is the Thalys PBKA train which has seven different signalling systems onboard to handle operations on different lines and country standards. Where ERTMS has not been deployed, existing signalling systems are used, including AWS/TPWS (Automatic Warning System/Train Protection Warning System) in Great Britain and KVM (Contrôle de Vitesse par Balises (Speed Control by Beacons)) in France.

ERTMS is currently being rolled out across Europe and, whilst it is a European standard, it is being actively deployed on high-speed lines across the world. At the end of 2014, over half

of some 80,000km of railways equipped with ERTMS were located in Asia [1]. One potential way to deploy ERTMS is through 'national deployment' where rolling stock and trackside infrastructure are upgraded on a large scale. An example of this is the pending East Coast Mainline deployment of ERTMS in Great Britain within the next decade.

In addition to the ratified standards, 'baseline' standards are 'controlled evolutions' [2] to the standards which are undergoing operational testing and, once ratified, become the new version of the standard.

2.2.1 Architecture

ERTMS has three operational levels, where ETCS Level 1 is the lowest, most basic level and ERTMS acts as an overlay to the national signalling system. Within this level, EuroBalises are responsible for delivering the signalling 'Movement Authority' issued by the Radio Block Centre (RBC) to the train (allowing it to proceed on the line).

A EuroBalise (EB) (also known as a balise) is a RFID-like device which is placed between the rails. In ETCS Levels 2 and 3, they are responsible for providing absolute location references to the train, in addition to track information, e.g. line speed and gradients. Optional messages, known as 'Packet 44', allow for national customisations e.g. default speeds and, in the UK, is being used for Tilting Authorisation and Speed Supervision (TASS), used on the Class 390 '*Pendolino*' and Class 221 '*Super Voyager*' to govern safe tilting actions on the West Coast Main Line (WCML). Balises are typically grouped into pairs (known as a 'balise group'). When a train passes over a 'balise group' it is able to determine its direction of travel and report its position to the RBC.

In ETCS Level 2, EuroBalises are used as location beacons, which inform the train of its current location. Movement Authorities (MA) are given via a train to trackside link and existing signals are now optional in this operational level. Finally, ETCS Level 3 is the most advanced

[1] `http://www.ertms.net`
[2] `https://ec.europa.eu/transport/modes/rail/ertms/general-information/faq_ertms_en`

operational level, removing the requirement for safety integrity monitoring through track-side circuitry and enabling moving block operation (i.e. a virtual block is created between trains based on the braking distance capabilities, rather than the fixed blocks between signals currently used in the UK rail network).

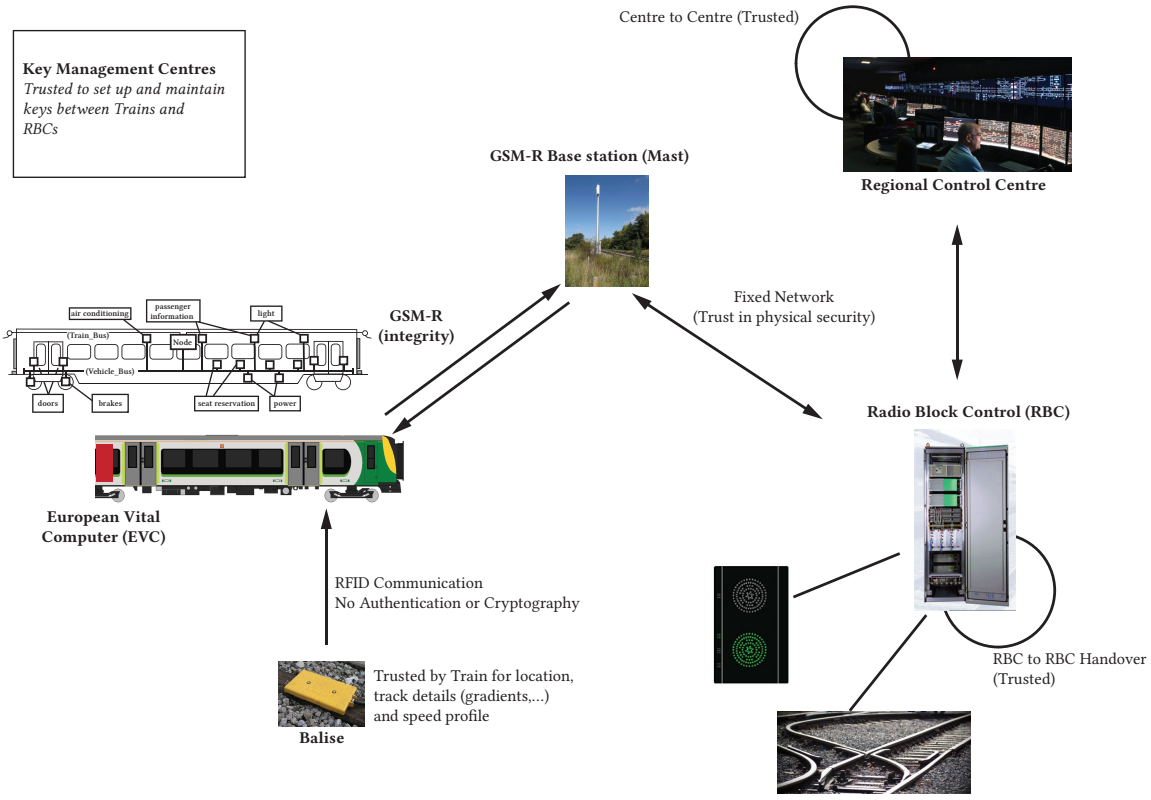


Figure 2.2: High-Level ERTMS System Architecture

Trains communicate with one or more RBCs during their journey, where the RBCs are responsible for issuing command and control messages to the train. RBCs typically cover a specific geographical area of approximately 70 kilometres [3]. RBCs acknowledge train location reports, verify the safe operation of the railway and also issue commands, known as a Movement Authority (MA) to the train, which defines the safe distance the train may travel and its maximum permissible speed. Each RBC is connected to a fixed network to hand over trains to the next RBC when a train leaves its area of responsibility.

RBCs may also be connected to a regional control centre (or Rail Operation Centre (ROC)), operated by the National Infrastructure Manager, where manual intervention actions may be taken, for example, in the event a train unit has failed. RBCs interface with the lineside signalling equipment in ETCS Levels 1 and 2 to cater for trains which are not ERTMS-ready and trackside circuitry, for example axle counters, to ensure that a block of line is truly clear and allows informed safety-critical decisions to be made by the RBC. In ETCS Level 3, however, train 'integrity' is managed solely onboard the train, where the RBC trusts the train's reported position with no external validation [71].

Finally, the Key Management Centre (KMC) is a nationally-operated system, handling cryptographic keys which are used to negotiate secure keys between trains and trackside equipment. Trains and RBCs are 'homed' to a specific country and each KMC will interface with peer country KMCs to obtain cryptographic keys to enable trains to operate across borders.

A high-level architecture of ERTMS is given in Figure 2.2 which shows how systems are interconnected, in addition to their respective roles and responsibilities.

2.2.2 Protocols

ERTMS relies on several protocol layers for communications between the train and trackside (Figure 2.3), where each layer provides some level of service and security features to the upper layers. As previously mentioned, ERTMS is formed of ETCS (the EuroRadio and Application Layers) and GSM-R, as shown in Figure 2.3.

Global System for Mobile Communications for Railway (GSM-R) This is the lowest layer for communications between trains and the trackside infrastructure specified in ERTMS [72, 73] and is an extended variant of the Global System for Mobile Communications (GSM) standard, operating internationally on a common frequency band. It includes enhanced functionality allowing, for example, emergency group calls, pre-emption and pre-defined short

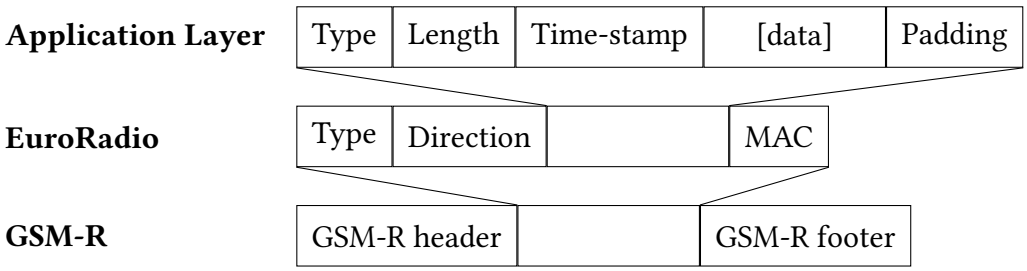


Figure 2.3: Overview of the different communication layers in ERTMS.

messages to be sent between the driver and signaller, e.g. 'standing at signal' [122]. An additional feature that GSM-R supports are 'shortcodes' which use location-based functions to route the call to the appropriate destination. An example of this is a shortcode which contacts the local signaller or RBC. Black-spots in the network may, however, arise from poor network coverage. Inside tunnels, radiating cables, known as 'leaky feeders', can be used to ensure that the train is continuously connected to the network.

EuroRadio This layer is situated above GSM-R, providing additional authentication and integrity protection for messages between the train and trackside [140] through the use of a Message Authentication Code (MAC). It also ensures that the train and trackside entities are genuine through a handshake protocol that takes place for every connection. Messages sent to the train are one of two priorities - 'normal' and 'emergency'. Normal-priority messages are sent with a MAC which provides an authenticity and integrity guarantee whilst emergency messages bypass authentication checks. The cipher used by EuroRadio is negotiated during the handshake as a Safety Feature (SaF). Currently, only one SaF is supported, based on the ISO-9797 specified MAC Algorithm 3 [140]. However, there have been reports of other 'ciphers', e.g. Cyclic Redundancy Check (CRC), being used [121] which only provide integrity checksums.

Application Layer This operational layer interfaces the European Vital Computer (EVC), located onboard the train which supervises its safe movement, with the lower layers (GSM-R and EuroRadio) and has additional checks for the timeliness of messages received and mes-

sage ordering, using relative timestamps. Pre-defined messages are specified in the structure, allowing system intercompatibility between vendors.

2.2.3 EuroRadio MAC

'Normal-priority' messages are authenticated by having a MAC appended to the end of the message. This cryptographically verifies the integrity and authenticity of the message. In ERTMS, the EuroRadio MAC is a Triple DES (3DES)-based scheme, loosely derived from ISO-9797 MAC Algorithm 3 [31]. The keys used for this MAC are negotiated by the train and RBC as part of the handshake process of the EuroRadio protocol.

2.2.4 Key Management

Cryptographic keys are managed and distributed by a national body (the KMC), typically the National Infrastructure Manager (e.g. Network Rail for Great Britain, ProRail in the Netherlands and ÖBB-Infrastruktur in Austria). KMCs interact with each other to submit and process key management requests for cross-border operation in addition to managing the keying estate in their home domain.

2.3 Analysis and Assurance Methods

Different analysis methods are applied, depending on the component or system being evaluated. In this section, we will explore some of these methods and how they provide security assurance. What is important to note is that these methods are complementary and should not be considered or used in isolation.

2.3.1 Formal Verification of Protocols

When protocols are being designed, especially those which have some critical function, whether it is to convey data between two given endpoints over a common interface (e.g. UDP) or to guarantee confidentiality of data for web-based services (e.g. TLS), it is important to verify

that the protocols are designed and implemented correctly. During the design stage, formal verification can highlight the unexpected behaviour of the protocol which, if implemented, could become an exploitable vector by a malicious adversary. These adversaries could model the Dolev-Yao attacker. For the purposes of this thesis, we will define the attacker as one who is active and is able to intercept, inject a new message, or replay an existing, possibly modified message into a communications channel between two or more participants. The Dolev-Yao attacker, however, cannot guess keys where encryption is used if the probability is negligible. They can only use keys they learn from a protocol run.

Formal analysis captures the protocol as a mathematical model, where it is for automated 'provers' such as ProVerif [16] and Tamarin [104] to prove some condition, for example, that a malicious adversary cannot interfere with a critical part of the protocol. Common flaws that may be identified during formal analysis can be determining sources of leakage where data, which should have been confidential, is leaked to the adversary which could have significant consequences. It should be noted that, whilst a protocol may be proven to be secure against a defined adversary, its implementation and cryptography (if present) may include errors which renders the assurance of the protocol null and void. Various techniques may be used to validate the implementation of the protocols from standards, ranging from verifying the code to fuzzing the implementation to determine if it maintains conformance to a given standard.

An example application of the formal verification of protocols can be observed in the ratification of the TLS 1.3 standard [46], which found a number of design flaws in the proposed standard using the Tamarin verifier. This work was further developed in 2017 to demonstrate a direct relationship between the Tamarin model and the specifications [45], highlighting how it is possible to map the protocol into a formal model.

2.3.2 Cryptographic Analysis

As previously identified, formal analysis of protocols assures only that the protocol behaves in an expected way and that it meets specific guarantees of security. What this type of analysis does not consider is the cryptographic design and how it might place the protocol at risk if it has known weaknesses. Cryptographic analyses concern themselves with the design of the algorithm and how it is used, where game-based approaches may be used to assure their design. That said, cryptographic algorithms may be proven secure against a current adversary but, with advances in computational capabilities, may allow the cryptography to be broken at a later time. However, unlike automated symbolic protocol verification, cryptographic analyses are typically manual processes.

Dowling et al., as an example of the application of cryptanalysis, carried out an analysis of the TLS 1.3 Handshake Protocol candidates, showing them to be cryptographically secure [53].

2.3.3 Modelling Complex Systems

Another complementary technique to assure the security of systems is achieved through modelling. Here, an abstracted architecture is created where we can explore the interconnected nature of these systems. We can then determine the data dependencies that exist in the system, following them to identify the flows of data and, of interest to a system owner, where attacks may take place and their impact on the system.

Yearworth et al., for example, model a corporate IT system, using system availability as a metric to determine the impact of attacks against the system [148].

2.4 Threat Models and Regulation

In the rail sector, an emphasis is placed on validating safety cases, which can include modelling the systems to identify the risks and hazards that may exist. This allows system owners

to be certain that they will not introduce something which can affect the safe operation of the railway. In security, whilst we can also model threats to our systems, we place an emphasis on what a malicious adversary could do, whereas the safety model considers all other eventualities.

The threat model is able to consider one or more attackers, each with their own individual set of capabilities, for example, intercepting traffic between the train and trackside or one who has a quantum computer with the ability to break public key cryptography. Within their 'arsenal', the adversaries are able to use *attack paths*, paths through the system they can take to reach their desired goal. These are typically used as part of *cyber kill chains* which includes the stages from reconnaissance to exploitation, to leverage weaknesses and vulnerabilities in connected systems to reach systems which may be externally isolated and not normally accessible by an external adversary. By following this 'chain', the attacker can use each system as a pivot point to traverse through the entire system until they reach their desired endpoint.

Using these models, the asset owner can determine whether the risk is acceptable in the context of other factors (i.e. their *risk appetite*, which may be influenced by their risk budget, which may specify a financial threshold at which the risk is acceptable) or whether mitigations should be made to reduce the level of risk to one that is acceptable, depending upon the type of risk and the likelihood of a risk occurring versus the cost to compensate or mitigate it.

Anderson outlines a step-by-step process for modelling such systems, the approaches to managing risk and provides a number of real-world worked examples for the various methodologies that can be applied [2].

2.4.1 The European Network and Information Security (NIS) Directive

The Network and Information Security (NIS) Directive in the European Union came into force in May 2018, aiming to improve the security of critical infrastructure. In this Directive, an Operator of Essential Service (OES) is required to report incidents to their respective National

Technical Authority (e.g. the National Cyber Security Centre for UK-based incidents) if certain thresholds are met. In the United Kingdom, the Department for Transport set out the NIS thresholds for the transport sector [52], where, for rail, all incidents must be reported when services are degraded and breach the defined threshold, *regardless of the root cause.*

2.4.2 Safety and Security Levels

Safety Integrity Levels (SILs) are used in safety-critical applications to define the risk thresholds that exist in a system and the confidence in the safety performance of that system [55]. These levels are extensively used in ICS deployments where safety-critical systems may be deemed SIL4 (the highest level of safety concern) and SIL0/1 is the lowest level (no safety concern) [61]. Each level has probability metrics based on the likelihood of failure. Security Levels (SL) can be seen as a security-focused evolution of SIL levels [23] where SL1, the lowest level, considers a casual attacker and SL4 captures a sophisticated attacker with high motivation and a comprehensive skillset. As Braband notes in [23], as there is no direct mapping between SIL and Security Levels, there may be disparity between the assigned levels. Bloomfield et al. consider 'capability levels' [19], similar to Braband's proposal, but try to map these levels directly to the SIL Levels.

Chapter 3

Related and Previous Work

As identified in Chapter 2, safety-critical systems, for example, those used in railway applications, are subjected to rigorous validation processes to assure that the system and its associated components are safe and, if an issue is identified, it is addressed to ensure that it does not have potentially serious implications. Whilst this assures the safe operation of the railway, it does not necessarily guarantee the security of that system and its components.

In this chapter, we will review the literature related to the assurance of the security of systems and the emerging research, specifically within the rail sector. As we focus on the safety-critical aspects of ERTMS and the implications of an attack, we will first review the existing tools and techniques that are applied to verify the safe operation of rail systems. We will then evaluate the existing techniques applied to assure the security of such systems, for example, formal protocol and cryptographic analysis of MAC algorithms.

Key management is a particular, perhaps, separate area, although intrinsically linked, that could have a significant impact if compromised. A number of proposals for its improvement have been suggested, which are used as inspiration in the development of TRAKS in Chapter 6.

In Chapter 2, we also noted that for both safety and security, there is an overlap in some of the techniques that can provide assurance, specifically risk and threat modelling. We will

also consider the security-focused aspects of modelling, used in Chapter 7, to overcome some of the gaps which were identified.

What is clear within this chapter is that security research within the rail sector and ERTMS is very new. In other sectors, however, the techniques that will be presented in this chapter are already well-established and extensively used in alternative security research fields.

3.1 Safety Verification and Modelling

As previously noted, the rail sector has a very mature verification process to assure its safe operation. In 2013, Fantechi surveys the past twenty-five years of the application of formal methods to verify rail systems [61], with one particular case study identified as 'one of the most fruitful' for formal methods – that of signalling. In this study, the *B Method* was applied to the Paris Metro Line 14, which identified a number of issues which were later addressed. This specific verification method now forms part of the EN50128 standard. In Fantechi's chronology, he assesses the evolution of formal modelling to include SAT-based checking. The benefit of this technique overcomes the issue of state explosion, a problem which arose when verifying interlocking systems and is now widely used in other industrial settings. Fantechi proceeds to consider the future of signalling, with ERTMS as a case study, noting the different efforts to validate the standards and principles (e.g. braking curves) and their interface with interlocking systems.

Another example of the use of formal verification was studied by Peterson in 1998 [111] on the Danish rail network. Previously, there was no way to verify the STERNOL-based interlocking system used by DSB, the national operator. The aim of Peterson's thesis was to assess how formal methods could be applied to verify the safety of the interlocking system, although various performance issues were also encountered which have subsequently been addressed.

3.1.1 Challenges of Safety Verification

Today, model-based checking continues to be used, both in the safety and security domain. With the increased digitisation of the rail network, the question of maintaining its safe operation now presents significant challenges in its verification, as Fantechi later identifies in 2016 [62]. In this work, Fantechi notes that, given the significant cost incurred in assessing safety requirements, assurance work is typically focused on components which are to conform to the highest SIL Level, Level 4, rather than a holistic approach which should cover all SIL Levels. However, due to the close interconnectivity of systems today, it is harder to segregate these components and, therefore, conduct independent analyses of these systems. For safety certification and simplicity, these components are now grouped under a common SIL Level, requiring significant development effort and cost for certification. Fantechi also offers a specific viewpoint on data-driven safety design methodologies which aim to simplify compliance with CENELEC EN50159, a communications safety standard used in rail applications. What Fantechi importantly notes is that security is an aspect which should now be considered to ensure safety, where safety and security are, today, intrinsically linked.

3.1.2 Safety Verification Today

The verification that components in a system work as designed and do not allow transitions into a potentially unsafe state is also well-established. Venit-Anunchai, for example, uses coloured petri nets to verify and formally prove nine properties, modelling the Thai signalling system [146]. In this work, issues were identified with this approach, specifically the risk of state explosion (an issue that Fantechi also notes [61]) and the time taken to generate the model state space. As we will observe in the following section, there is often an overlap between verification methodologies.

System designers, such as the RATP Group, have over twenty-five years of experience in verifying the safety of their systems [12], with the first verification of its automatic train

protection system of the RER A line in Paris. In this analysis, Benaissa et al. explore the PERF (Proof Executed over a Retro-engineered Formal model) approach, combining model-checking and induction proof techniques in a common tool. This approach allows formal verification of software (based on the code or a formal model), assumptions of the environment and safety requirements. This provides either a 'proof certificate', confirming the software meets the formal specification, or gives counter examples if a property does not hold. We will observe in the following section how various protocol verification tools can provide similar output in a different setting. Later work by Halchin et al. uses the PERF approach and applies it to the B method, a formal method based on logic and set theory [74]. The authors demonstrate how the B model is translated into a high-level language model which can then be verified. Another case study outlined by Comptier et al. applies Atelier B (a commercial implementation of the B method) to verify the Octys Communication Based Train Control (CBTC) system used by RATP on a number of lines [43], mapping each stage of the process to verify a component.

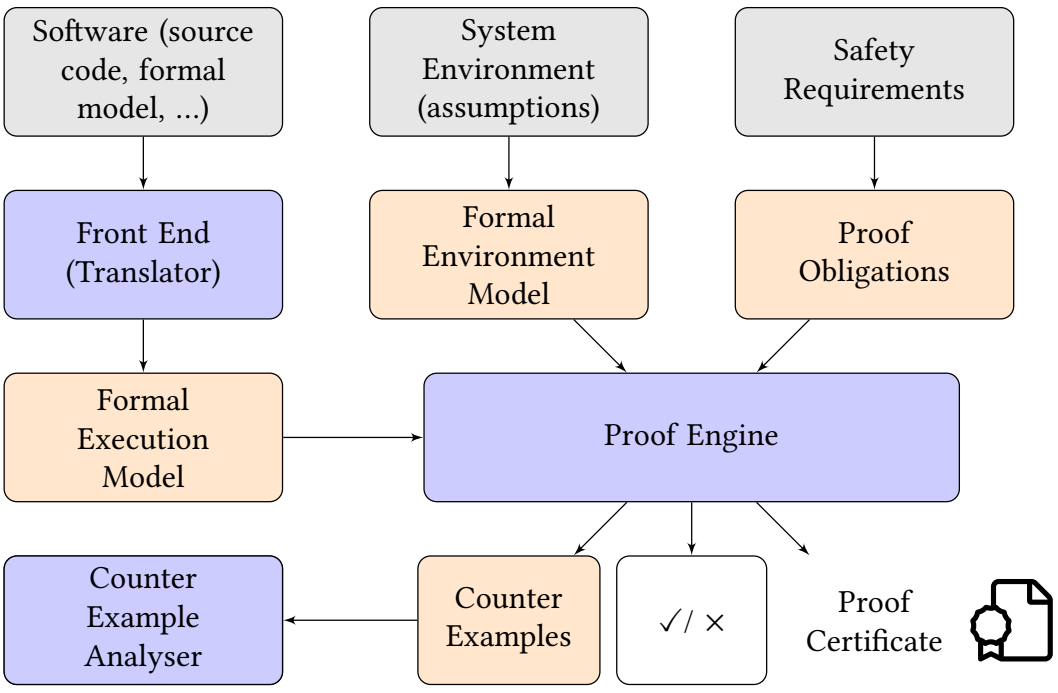


Figure 3.1: The PERF Approach Workflow [12].

An alternative verification tool is Systerel S3 prover, which takes a similar approach to PERF in how the safety specification and software developed are assessed, as noted by Breton and Fonteneau [27]. However, unlike PERF, where the software needs to be translated into a formal model, the S3 toolchain is able to translate code into the high-level language input required for the solver, providing a similar workflow to Halchin et al.

Moller et al. [106] develop their own modelling and verification approach, working from experience in Computer Science formal methods. Here, they apply a combination of the B Method and CSP (Communicating Sequential Processes) process algebra to specify communicating systems, creating the *CSP||B* approach. In their work, they consider the information flow of a signalling system to model the control table (determining how the interlocking system sets signals and routes), tracks and release tables, followed by safety verification of their model. In their analysis, they demonstrate how *CSP||B* only allows 'safe' movements of trains, using an external tool to verify this. What this work demonstrates is a union in formal methods between safety and Computer Science, which will be explored in the next section.

Finally, Song et al. apply 'reliability theory' in their analysis of ETCS Level 2 and highlight the implementation challenges that Infrastructure Managers and operators will face when the ETCS Level 3 standard is ratified [130]. The authors focus on the safety of European railways and propose a new type of Movement Authority, MA+, to bridge the interoperability gap between ETCS Levels 2 and 3. As part of this proposal, a safety analysis is carried out comparing MA+ to the current ETCS Level 2-specified Movement Authority. The authors note whilst the existing Movement Authority does not reveal the route and position of neighbouring trains and the state of the interlocking system, MA+ would provide this information. Using reliability theory, the authors consider the likelihood that a 'dangerous' event occurs and demonstrate that MA+ increases the safety and reliability of ETCS Level 2, whilst not affecting the performance of ERTMS.

3.2 Formal Modelling and Verification of Protocols

Protocols are widely used for a number of purposes, whether it is to relay data between two systems (e.g. TCP and UDP) or to provide secure transmission of data (e.g. TLS). These protocols, however, require formal assurance that they conform to their specification and, when faced with an adversary, those protocols which provide security services, do not contain flaws which can be exploited.

When assessing the security of the EuroRadio protocol, previous research work has attempted to provide assurance of its security, such as that presented by Esposito et al. [59] and Franekova et al. [67], using UML-based approaches. Alternative methods, for example, using the SPIN model checker by Zhang et al. [150] and petri nets by Hongjie et al. [79] have also focused on the EuroRadio protocol.

3.2.1 Existing Modelling of EuroRadio

Esposito et al. [59] carry out an analysis of the EuroRadio protocol at a high level from the perspective of the state machine that exists within EuroRadio. In this analysis, the protocol was modelled using UML to capture the state machine and protocol in their entirety, specifically verifying the behaviour of the state machine and how it handled unexpected messages being received or messages which did not conform to the standard (e.g. corrupted packets). This analysis, unlike using formal modelling tools, e.g. ProVerif, highlighted flaws, including the ability to force the protocol into a state of deadlock (i.e. the protocol is unable to proceed into any further state) and possible conflicts and inconsistencies within the standard. Their work, however, relates the EuroRadio specification to CENELEC EN50159, a standard which defines a threat/defence matrix to which safety-critical systems in rail should comply. However, this analysis does not clearly identify if EuroRadio successfully meets all of the requirements of the matrix.

UML is one other possible way to model the protocol, where Zhang et al. [150] transform

Threats	Defences							
	Sequence Number	Timestamp	Timeout	Source/Destination Identifier	Feedback Message	Identification Procedure	Safety Code	Cryptographic Techniques
Repetition	•	•						
Deletion	•							
Insertion	•			• (with Source Identifier)	• (Application Dependent)	• (Application Dependent)		
Re-sequence	•	•						
Corruption							•	•
Delay		•	•					
Masquerade					• (Application Dependent)	• (Application Dependent)		•

Figure 3.2: Threat/Defence Matrix as defined in EN50159 [29].

the protocol into a model which can be verified using the SPIN verifier. This model expresses the protocol as Interface Automata, a different approach, which does not require the input/output arguments that other tools require but, instead, works with 'assumptions'. In this work, UML complements the Interface Automata in defining a number of properties to which the protocol must conform. These properties are similar to that of Esposito et al. in [59], for example, deadlock conditions, whilst also including extensions, such as mandatory transitions and consistency checks. The tool found one trace which led to a deadlock. However, neither this, nor [59] provide any recommendations that would resolve these deadlocks. In reality, these would be considered unlikely due to the state transitions required to achieve a deadlock not being typical and would not affect the safe operation of the railway.

In 2012, a petri net-based analysis of the EuroRadio protocol was proposed by Lijie et al. [97] taking a statistical approach, a different perspective from the safety modelling discussed earlier in this chapter, to determine the impact of lost packets on safety. The authors verify EuroRadio for 'liveness properties', for example, ensuring that all states have a transition, do not lead to a deadlock and all transitions are reachable in the state machine. In 2013, this approach was also used by Hongjie et al. [79] to determine the results of using coloured petri nets to assess the EuroRadio protocol. This model contains an extension for timeouts, which Zhang et al. in [150] and Esposito et al. in [59] do not capture and more closely matches the true EuroRadio protocol. Whilst this analysis captures the performance impact and likelihood of protocol failure, it does not provide assurance from a security perspective.

A similar approach, where a UML model and petri nets are used in an alternative technique,

is described by Lijie et al. in 2016 [96]. Specifically, the authors assess the protocol but from a safety perspective for timeliness using uncertainty analysis. Here, a model of the EuroRadio protocol is analysed to determine the likelihood that the protocol may induce an unsafe state, where a connection cannot be established and the data received is not timely. The results of this work, by their definition, shows that EuroRadio is a safe protocol but question its validity as the data received may not be timely. As we will note in Chapter 4, timeliness and the delay of messages is an important property to inspect, as this can affect the safe integrity of the train.

In comparison, Hongyu et al. [117] use the PRISM tool to assess EuroRadio probabilistically. In this work, they also assess EuroRadio from a timing perspective for the safe reception of messages and the impact of message loss. This work, however, does not consider any potential security threats.

At a high-level, Lopez and Aguado identify some issues within the EuroRadio protocol, for example, a lack of sequence numbers (or timestamps) being used within the EuroRadio handshake protocol [99]. The authors note that this would allow replay attacks during the handshake. Under the state-machine as specified, there should be no transition based on a replayed message unless there was an error in the protocol implementation. The authors also identify a risk of message 'flooding' as each message MAC has to be verified prior to being handed up to the application-level protocol, creating a performance bottleneck. As Bloomfield et al. state in [18], this would not impact safety, but would force the train to fail-safe and stop, potentially causing disruption to the network.

What all these analyses highlight is that, whilst protocols may be shown to be safe, a lack of formal security verification may lead to exposures in all implementations of the standard. As security threats can affect the safe integrity of the protocol, it is essential that there is some verification from a security perspective. However, a weakness in all these analyses is the lack of a thorough assessment of the EuroRadio and Application Layer Protocols from a security perspective, using EN50159 as a driver to determine whether the two protocols, when used

together, provide sufficient resistance to attacks.

3.2.2 Protocol Verification Outside of ERTMS

Outside of ERTMS, formal verification of protocols are used extensively in other applications, for example, in assuring the new TLS 1.3 standard [45], finding flaws in ePassports [42] and analysing the EMV standard used for card payments [50]. de Ruiter models the EMV protocol [50], used for card payments in the EU and slowly being adopted worldwide, by assessing its guarantees to authenticate the card, customer and transaction. In this analysis, de Ruiter demonstrates flaws in the EMV protocol, identifying its susceptibility to man-in-the-middle attacks and that the terminal is unable to verify the authenticity of the transaction. The author later employs an alternative technique, specifically fuzzing, to assess the state machines of a number of TLS implementations. Fuzzing is a complementary analysis technique to formal verification where implementations should be assessed for conformity to the standards. If the implementation does not conform to the specification, there may be additional states which could lead to an exploitable vulnerability. As an example, McMahon Stone et al. fuzz the 4-way handshake protocol, used in Wi-Fi, on a number of devices [103]. This analysis identified a number of additional states or transitions which are not part of the protocol specification and, again, can lead to exploits such as downgrade attacks. However, whilst a model could be proven correct, it does not rule out implementation-based vulnerabilities which could be found through these complementary methods.

Similar to the work conducted in the rail sector [61] to verify a model and automatically generate code conforming to a given specification, reference implementations of protocols can also be generated. JavaSPI is an example of a tool which can formally verify the protocol implementation using ProVerif [8], but this again does not negate vulnerabilities introduced during implementation.

3.3 Cryptographic Analysis of MACs

The analysis of encryption schemes is a mature area of research with a significant volume of literature available. As highlighted in Chapter 2, the MAC used in ERTMS is a modified version of the ISO-9797-1 MAC Algorithm 3 [31]. This algorithm, as specified in the ISO standard, has been analysed previously by Mitchell [105] and Preneel et al. [116, 75]. Although more directly-related to the EuroRadio MAC, Pepin and Vigliotti [109] completed an analysis of ERTMS key distribution, which uses this algorithm, whilst Franekova et al. [67] carry out a specific analysis of the EuroRadio MAC.

3.3.1 Cryptanalysis of ISO-9797 MAC Algorithm Three

The analysis of the ISO-9797 specified MAC algorithm by Mitchell [105] and Preneel et al. [116, 75] found a collision-based key recovery attack, where $2^{32.5}$ known plaintext-MAC pairs and $3 \cdot 2^{56}$ offline operations were required to obtain the key. One caveat of this research is its focus on the MAC as specified in the standard. As previously stated, the EuroRadio MAC algorithm is based on the ISO-9797 MAC Algorithm 3, with a modification to the final transformation. This, therefore, means that the attacks described by the authors, whilst useful at the time, are no longer appropriate due to the change from a double DES transformation with two distinct keys to a 3DES transformation with three distinct keys.

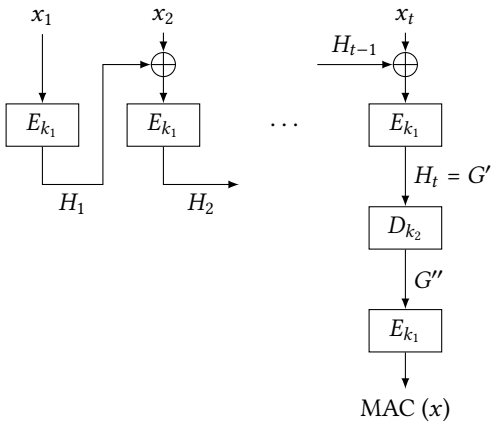


Figure 3.3: ISO-9797 MAC Algorithm 3, as reviewed by Preneel et al. [116].

Considering the final transformations between both MAC versions, Smart, however, provides proofs (with increased time and space complexity of 2^{64} respectively) that a two-key 3DES transformation can be broken using a chosen-plaintext attack [129]. In this attack, a chosen-plaintext attack oracle is used to look for key pairs which match at each stage of the computation and candidate keys (as pairs) are then returned. Smart concludes with a subtle hint regarding the lifespan of 3DES, where it is 'just not as secure as one would expect', given its keylength of 168 bits.

3.3.2 Analysis of the EuroRadio MAC and Key Distribution Scheme

Franekova et al., on the other hand, assess the EuroRadio MAC and key encryption scheme [66, 67] but focus on the 3DES component of the algorithm. Any attacks found would, therefore, be limited to key distribution and one part of the EuroRadio handshake. In their former work, the authors describe the key management hierarchy before identifying three potential attacks which are more efficient than a brute-force search, in particular, differential cryptanalysis, linear cryptanalysis and the 'Davies Attack', a known-plaintext attack [66]. They note that the best type of cryptanalysis for DES is linear cryptanalysis, where 2^{43} *known* plaintexts are required with a time complexity in the region of $2^{39} - -2^{42}$ to recover the key. Whilst this has a noticeably higher known-plaintext requirement than the attack found by Mitchell [105] and Preneel et al. [116, 75], it requires less time. The authors then consider a birthday attack, a similar approach to Mitchell and Preneel et al. to reduce the keyspace. In the latter work by Franekova et al. [67], the authors use UML to define this attack, with similar results of possible collisions to their previous work. As example collisions are not detailed in either contributions, some additional validation may be required.

In their analysis of the ETCS component of ERTMS, Lopez and Aguado note the limited lifespan of 3DES, and that the risk of a birthday attack arises from two sources, the extended use of the same KMAC for multiple trains and weak random number generators [99]. The

latter risk is critical for the EuroRadio handshake, as both the train and RBC exchange nonces which, when combined, derive the session key between the train and trackside. If the random number generator is predictable, it reduces the attack complexity significantly. The authors also review the potential alternatives, albeit at a high-level without in-depth analysis.

Pépin and Vigliotti [109] analyse the key distribution mechanism used in ERTMS and assess its resilience against standard attacks against the 3DES algorithm. They specifically use a *related-key attack* and quantify the cost to the attacker, in terms of expense, resources and capabilities to break the 3DES encryption, used in ERTMS key management. The attack they present is, however, considered currently impractical due to the inherent financial cost, estimated to be in the order of millions of dollars. It also focuses on the 3DES scheme applied to key distribution, where there would be significantly fewer keys involved compared to EuroRadio session keys.

3.3.3 Cryptanalysis Fundamentals and Wider Applications

Smart outlines three fundamental properties that a hashing/MAC function should satisfy [129]:

- Preimage Resistance – given a hash/MAC, the adversary should only be able to recover the original input message with negligible probability.
- Second Preimage Resistance – given a hash/MAC, the adversary should only be able to recover an input message that results in that hash/MAC with negligible probability.
- Collision Resistance – it should be sufficiently hard to find two messages which result in the same hash/MAC.

Using these properties, it is possible to determine the security of a given scheme, demonstrating appropriate security games which would validate if the algorithm satisfies the given properties. Smart also provides insight on the generation of fresh keys from a given, long-term, symmetric key, where protocols such as Needham-Schroeder and Kerberos are given as examples, before providing an intuition on two analysis methods which can be used, game-based

and symbolic.

Looking outside the EuroRadio MAC and into the related research field of authentication and encryption of payloads, Karstensen applies a more involved cryptanalysis technique of brute-force recovery on the A5/1 cipher, used by GSM and GSM-R for encryption of data between the base station and mobile device [87]. The brute-force attack involves capturing plaintext/ciphertext pairs and generating a rainbow table by running the A5/1 cipher (as it is a Linear Feedback Shift Register (LFSR)-based scheme) and then extracting potential candidate keys to decrypt the traffic. Lu et al. apply a related time-memory trade-off attack [100], also using rainbow tables, reducing the attack time from the previously attained hours and minutes to 9 seconds, with a pre-computation time of 55 days on Commercial off-the-shelf (COTS) hardware and an 81% likelihood of success.

Barkan et al. have also analysed the A5 suite of ciphers [10], focusing in particular on the A5/2 cipher, an export variant of the A5/1 scheme. The authors use a known-plaintext attack to recover the A5 key, reducing the previously attained attack time for A5/2 of 6 seconds to less than a second, requiring 2 GSM frames of data which are 2^{11} frames apart (approximately 6 seconds). They then propose a ciphertext-only attack requiring just 8 frames of data, recovering the key in milliseconds. What is important from the results is that the attack also works on General Packet Radio Service (GPRS) networks, planned to be implemented into ERTMS to relieve the capacity limitations imposed by GSM-R. Known-plaintext attacks are, therefore, common to all three. It is also important to note that GSM and GSM-R broadcast packets are sent both as plaintext and ciphertext (due to the time-based nature of the protocol), allowing an adversary to carry out these types of attacks.

3.4 Key Management for ERTMS

Key Management is also a mature area of research. However, when considering post-quantum solutions, it is still an emerging field. As we will discuss in Chapter 6, alternative key manage-

ment schemes have been explored for ERTMS. In this section, we will assess their shortcomings and also review the literature for potential schemes from other sectors that could have been applied to ERTMS.

3.4.1 Existing ERTMS Key Management Analyses and Proposals

In 2012, Franekova et al. [68] consider two approaches for an online key management solution. The authors propose a new key management hierarchy, essentially turning the scheme into a solution based on public key cryptography, introducing public/private key pairs. Their work, however, specifically focuses on the technical implementation of their solution, for example, the length of keys and algorithms to be used, rather than exploring their proposed architecture in more detail. For the purposes of integration into any system, the lack of clarity regarding the specific public key cryptography used could lead to the loss of forward secrecy. If one entity is compromised, its private key would be divulged to the attacker, presenting longer-term issues, as the key is not fresh and all future session keys would be compromised.

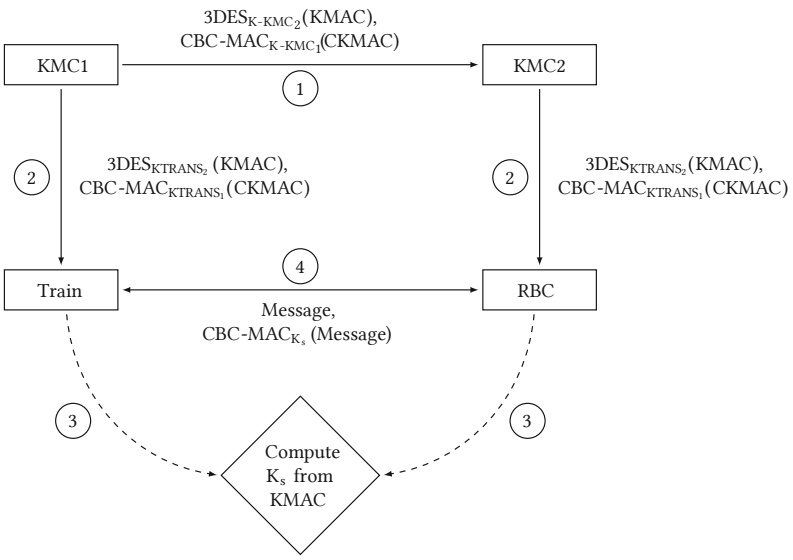


Figure 3.4: The current ERTMS Key Distribution Scheme, assessed by Pépin and Vigliotti [109].

A whitepaper presented by Brandenburg et al. [24] takes an industrial view on the state of key management in ERTMS and identifies some of its associated issues, e.g. cross-border

operation. Its solutions, however, are not particularly secure, as spreadsheets and text files are used without any clearly-defined protection. What the whitepaper does highlight is that there is a need for a more efficient solution whilst also demonstrating the problems associated with the proposed online scheme.

Existing proposals to improve ERTMS key management have been largely based on the existing offline scheme and extended such that the back-end KMC infrastructure is accessed via online communications by trains and RBCs. Fuloria et al. in 2010 [70] use ERTMS as a case study for protecting communications within critical infrastructure applications. The authors introduce another hierarchy for ERTMS key management, deeming it 'both efficient and robust', but this relates solely to offline key management. They propose a similar key management hierarchy, used in smart-grid applications, applying this into an online scheme for ERTMS, introducing a Public Key Infrastructure (PKI) with a supporting central key management infrastructure. In a further extension to this work by the authors in 2011 [69], considerations are given to when asymmetric cryptography should be applied, compared to symmetric cryptography. Here, they consider the potential threat vectors using a similar model for ERTMS [19, 21].

Similar to industry-led proposals such as [24], the Danish Rail Board [9] defined their own set of security goals and prerequisites for an online scheme. Their approach was to limit the keyspace such that only one train has a single KMAC installed to simplify revocation, a potentially dangerous proposal as Lopez and Aguardo suggest [99]. As we will see in Chapter 6, this is an unusual approach, where the authors advocate the replacement of the KTRANS key with public key cryptography, based on the online version of the ERTMS scheme [141].

3.4.2 Alternative Key Management Approaches

If we look at key management as a general concept, it may be possible to identify and adopt a different approach for ERTMS. Boyd and Mathuria [21] provide a succinct overview of the

problem surrounding key management, particularly where more than one party is involved. As trains may communicate with many RBCs and vice versa, this is highly relevant. In their analysis, the authors demonstrate how example protocols based on the Diffie-Hellman Key Exchange Protocol do not provide a means of authentication. It does, however, highlight how some of these protocols may be used to effectively share cryptographic keys between more than one party. One potential solution that could have been applied in Chapter 6 is the two-party Diffie-Hellman scheme. However, this is not suitable as it does not support the notion of a permissions model and does not provide any guarantees of authenticity.

In their proposal, Burmester and Desmedt [33] argue that a generalised Diffie-Hellman protocol could guarantee key authenticity, provided that specific values in the exchange were authenticated. This proposal was considered insufficient by Just et al. [84], where a TLS-style handshake, outlined by the authors, provides authentication using a signed hash of all messages sent as part of the protocol. If applied to EuroRadio, this would additionally demonstrate that no values were 'tainted' during the handshake. This signed hash, however, would be particularly expensive to compute, especially for the RBC as it would have to temporarily store all the messages received and sent prior to computing the signed hash. An alternative that could have been considered is to sign each message. This proposal also has a limitation, specifically the bandwidth of GSM-R, where message transmission latency would be impacted.

Biswas further extends the notion of Diffie-Hellman based key exchanges with multiple two-party keys and a single multi-party key [13]. In this proposal, 'base' and 'extended' keys are used to provide an efficient key exchange technique for large, static groups. Base keys, four initial keys which can be combined through multiplicative operations to derive extended keys are initially defined, which can then be used to provide as many keys as required. Considering its applicability to ERTMS, parties can either be static or dynamic, depending on where the system is analysed. The current key management scheme [142] is largely static: once a key 'relationship' is established between a train and RBC, it will only concern those two specific

entities. As ERTMS is moving towards a dynamic scheme, where trains can communicate with a particular RBC but may also be transferred out of the original 'approved' region, the proposal by Biswas still remains partially applicable, albeit with some caveats. One example is that once keys are allocated, their revocation is difficult. A further limitation, highlighted by Boyd and Mathuria, is a limitation to the efficacy of the solution, as messages are not authenticated [21].

Two new methods for key agreement are proposed by Ateniese et al. [6, 7] which extend the GDH.2 protocol by Steiner et al. [132] to provide an authenticated key agreement solution. One major flaw in these proposed solutions is that it is impossible to identify if there are any other participants in a protocol run, such as an adversary. Attacks which render this scheme insecure were found by Pereira et al. [110], showing that messages could be modified or replayed, ultimately compromising the session key. Bresson et al. assess other modifications to GDH.2, introducing the identities of all participants, sending them as part of the message flow [26, 25]. Here, the authors ensure that each flow is signed with the long-term key of the sender. This solution is more appropriate for authenticating the general Diffie-Hellman key exchange between multiple parties, as all entities are made aware of each other and can identify dishonest parties that should not be participating in the session. One critical aspect that we need to consider, especially for Diffie-Hellman-based schemes, which rely on the difficulty to factor large numbers, is the post-quantum resistance of any proposal. As a result, none of the proposals above would be considered post-quantum resistant and secure for the future.

Instead of directly using Diffie-Hellman, an alternative approach for key management can be achieved through the use of identity-based conference key protocols. All parties are assigned a public key based on their identity which, in the case of ERTMS, is the ETCS ID. Messages between parties are encrypted using this public key, where a central authority holds the appropriate private key, releasing it to the entity upon successful authentication. Koyama et al. use Diffie-Hellman to define a protocol which agrees a key between a number of entities [90] and considers a number of potential attacks, including impersonation and interception

by a passive and active adversary and, of interest when considering post-quantum resistance, the difficulty to factor large numbers and overcome the discrete logarithm problem. In their proposal, they use a star architecture, requiring one principal to have all messages exchanged between it and all other participating entities. This, however, introduces additional computational costs but has the benefit of *implicit* key authentication guarantees. In this scheme, we rely on a trusted third party to generate the private key which corresponds to the specific entity. Moreover, as the scheme uses the identities of the participants as the public key, if no 'limiting' factor is appended to the identity, the private key is perpetual, consequently making key revocation difficult and potentially increasing the likelihood that an adversary recovers the private key. As Schneier states in 2007, 'nothing proposed so far is both practical and secure' [126] when considering the current state-of-the-art of identity-based public key cryptography proposals. This still remains a problem today.

Many of the previously mentioned protocols rely on the Diffie-Hellman protocol but none provide sufficient measures that limit the 'reach' of entities or introduce a permissions model. For example, a train that is only authorised to operate in the West Midlands region should only be able to interact with a RBC located within the West Midlands. This presents a limitation where some secret value would need to be shared between parties to introduce this 'permissions' concept.

Alternative proposals which do not use Diffie-Hellman include one, for example, by Tzeng et al., where their protocol can be completed in a single, synchronous round [137]. The only information that has to be known prior to a session commencing is the session ID. In the context of EuroRadio, the train could announce a randomly-generated session ID which would start the key agreement process. In this proposal, a third party, e.g. the railway line as an entity, would be required to participate in the protocol run. This is not currently possible, as a number of modifications would be required for all entities to support this permissions property and the announcement of a session ID to EuroRadio.

3.4.3 Key Management in SCADA Applications

If we look at some of the solutions proposed for SCADA applications, Piètre-Cambacédès and Sitbon [113] identify where cryptography is best applied in a relatively straightforward process noting, however, that managing the keys is a difficult task. They consider different key management strategies, one of which is the 'key-server based' solution, the closest-matching architecture for the offline ERTMS scheme. A comparison of the available strategies is outlined, where symmetric keying systems are deemed more attractive than PKI-based schemes, confirming that the proposals reviewed earlier in this section may not be a suitable approach. The authors raise the issue of keylength and the limited computational power available, suggesting that a risk-based decision on suitable schemes should be made by the operator. Of particular interest from this work are the case studies outlined by the authors, where it appears that the rail sector is not the only area which has industry-led work to improve key management. For example, DNP3, a protocol used in power grids had its key management standardised through an industry user group. This provided a two-level keying scheme with one long-term key shared between entities. One of the key points the authors note is that, when considering SCADA systems, some trade-offs are required, especially when considering key management, as no single solution assessed was suitable for ICS applications.

As previously stated, the research area surrounding key management is relatively mature. However, when we consider the design of ERTMS, whilst there is no suitable solution that we can draw upon from the current 'state-of-the-art' solutions, we may use these to form a basis for a new proposal. Key management in rail additionally presents a unique constraint through its complexity and requirement for cross-domain interoperability. As many solutions fail to support these requirements we cannot, therefore, propose solutions which would otherwise require significant changes to the underlying standards.

3.5 Risk and Threat Modelling

Modelling of complex systems is a process that is prevalent in a number of areas, with its application now progressing to information and operational systems security. As compliance to the European Union Network and Information Security (NIS) Directive is now a legal requirement, a number of processes, tools and methodologies have been developed for threat and risk modelling, allowing system owners to review the security of their infrastructure. Anderson notes that threat modelling is also a critical part of designing systems [2].

3.5.1 Modelling of ICS Systems

A framework for vulnerability detection in ERTMS is proposed by Arsuaga et al. [5] which takes a 'lessons learnt' approach as its motivation and identifies existing vulnerabilities in the ERTMS standards. The authors identify vulnerabilities by simulating a minimal train and RBC implementation using open source software. Using this attack-focused framework, they aim to find potential vulnerabilities by recovering traffic sent between the train and RBC through a man-in-the-middle attack. This work, however, only considers the existing vulnerabilities within the ERTMS standards, specifically the A5/1 encryption keys used by GSM-R and the related-key attack found by Pépin and Vigliotti [109] to verify whether an attack was successful. It does not, however, offer any scalability or scope to assess the security of other components used within ERTMS.

For ICS systems, the STRIDE (Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service, Elevation of privilege) framework is applied by Khan et al. [88] to a small synchronous island model. This framework has five key steps: reducing the system into its underlying components, creating a data-flow diagram of the system, analysis of threats to this diagram and the identification and subsequent mitigation of vulnerabilities. From this methodology, the consequences of a threat are identified, with components matched to those threats in each of the STRIDE phases. From the outset, this framework carries out a 'snapshot-

in-time' analysis of a system, but does not explore the propagative effect of vulnerabilities across the system where an exploitation at a given node affects neighbouring nodes, spreading through the system.

An alternative way of modelling system architectures is proposed by Fielder et al. [64] which uses particle swarm optimisation in a system model to identify methods that provide the best defences (with a measurable impact) that can be applied to the system. In this work, the authors carry out simulations to support the development of security strategies, modelling attackers and defenders as separate classes of agent, each with their own set of profiles and resources. The results of this work provide an insight into the attack and defence strategies that can be applied to complex systems to deliver optimal security, within a specified budget. Whilst Fielder et al. use Netlogo 'agents' to represent the input, Lodderstedt et al. [98] use a UML-like language to express the input, where their SecureUML tool models an access control system and generates a supporting infrastructure that conforms to the specifications provided.

Agent and UML-based threat modelling is one potential approach an asset owner may use. Alternatively, graph-based risk modelling may be used which has been previously applied to cyberphysical systems. Santini et al. [125] apply graph-based evidence theory as a way to express risk in complex, interconnected systems. What is key here is that the authors allude there must be some union between domain and subject-specific knowledge, providing a contextual point of reference to support the development and maintenance of the model. Lautier et al. also apply graph theory in an alternative context, specifically the finance sector [93], to assess systemic risk to financial markets, where their framework is able to search for, and find, the most probable and shortest path that would result in stock fluctuations. However, in cyberphysical systems, an adversary could chain together a set of vulnerabilities with less resistance to attack, albeit following a longer path, which the approach by Lautier et al. would not find.

Existing work has also investigated the integration of Common Vulnerability Scoring Sys-

tem (CVSS) metrics to attack graphs. However, there are differing approaches which do not necessarily work in a probabilistic model. The CVSS metric is defined as a 'snapshot-in-time' view of the current system but, beyond this, interdependencies between components are not captured, which would not provide a rich analysis. As an example, Cheng et al. provide one such insight to transform a CVSS score into a probabilistic metric of an attack being successful [40]. As we will see in Chapter 7, by tracking these dependencies, the CVSS probability values across attack paths can be calculated.

Looking at risk and threat modelling from a different context, Ioannidis et al. use modelling for security operations [82] to determine the trade-offs for patch management and the optimal strategy for patching without affecting system security. This approach is also used by Beautement et al. [11], adapting the security requirements of a system to determine appropriate strategies. Trade-offs are decisions that ultimately may have to be made as part of the modelling process when potential issues are identified. Anderson also considers trade-offs [2] and the balance between the financial investment to mitigate or reduce the risk against its likelihood and impact.

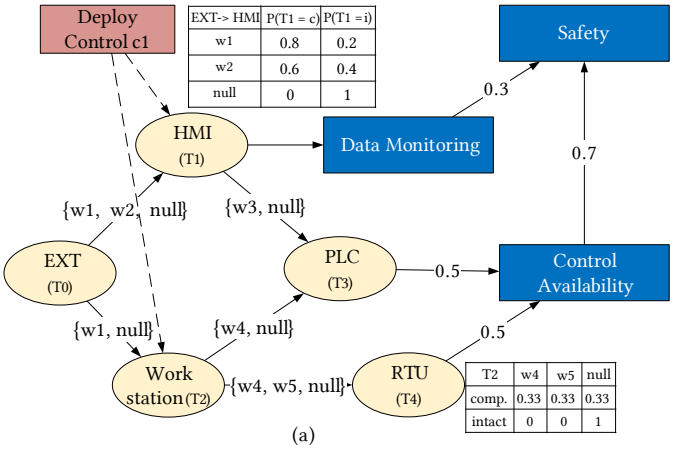
An alternative perspective for determining the appropriate action to take (if the asset owner has an established and well-defined attacker model) are Attack Trees [2]. Given a set of adversary goals, the asset owner can define the steps that the attacker would have to take (factoring in the expense and effort required) to progress through the tree. As an example, a node on the tree could be a perimeter firewall, which may cost $1,000 of time, effort and expense to breach. Attack-Defence Trees are similar, but contain an extension where the tree is supplemented with 'defender' nodes, each with their own defined costs and contextual information as Kordy et al. describe [89]. The outputs of the trees allow the asset owner to balance the 'cost' of attack against the cost of implementing defences. Byres et al. assess the use of Attack Trees when used in SCADA Systems [36], in particular identifying the possibility of becoming too focused on the resolution of a potential vulnerable node, when the relative risk is low.

Example trees are outlined by the authors which are then assessed for the relative difficulty for an attack to succeed, its severity and the likelihood that the adversary would be detected.

The SPARKS Horizon 2020 Project applies Attack Trees in a risk assessment for smart-grids [80] where they further extend Attack-Defence Trees to include vulnerabilities that may exist within a given architecture. The authors identify an issue with existing Attack Trees, as asset owners can become too detailed and the tree, as a result, becomes very large and mainly redundant. In this analysis, vulnerabilities can exist as general threats (e.g. no authentication), or link directly to a Common Vulnerabilities and Exposure (CVE) that affects that component. The Project notes that when designing the tree, it is important to have a deep understanding of the attacks that may arise and also that their methodology defines a process to establish 'good' attack graphs. In this methodology, by capturing the vulnerabilities that exist in a given architecture, the asset owner will become aware of the real threats that exist to their system, in contrast to the hypothetical threats that may exist.

Both of these approaches rely on the system owner having prior knowledge of the potential attack vectors that could exist in their system architectures, potentially requiring many iterative trees, with the additional risk of omitting vectors (and goals). Unlike Attack Trees, a model-based approach views the system architecture as a whole, rather than siloed components. Instead of quantifying security as a cost metric, which is purely an estimate and may be incorrect, we can, instead, use security profiles. These allow the asset owner to pose questions such as 'Given this asset, what can reach it?', similar to the goal-based approach that is taken by Attack-Defence Trees. Shostack reviews a number of the previously mentioned processes, including STRIDE and Attack Trees [128] and how they can be managed and addressed, in particular, dealing with the potential type of trade-offs that may arise, e.g whether the risk should be accepted, mitigated, avoided or transferred. The author also appraises the value of modelling systems and the potential threats posed, providing a summary of the tools and methodologies that are available.

For safety-critical systems, risk assessments, as identified earlier in this chapter, are undertaken using safety modelling. These assessments define the possible hazardous events and scenarios that may lead to a threat to life. From this, causal and consequential analysis is carried out, evaluating the risk and defining those actions necessary to mitigate, reduce or remove the identified risk factor. This approach is primarily safety-driven but can be adapted for security. However, this would suffer from the same flaws that exist in Attack-Defence Trees, where a definitive metric is needed to quantify the security impact and investment required to compromise an asset. Matulevičius offers an alternative approach through the ISSRM security metric [102], which factors in the likelihood and vulnerability level (the relative ease to exploit). We could argue that the vulnerability level influences the likelihood but, on the other hand, there may be more attractive targets for an adversary, where the vulnerability level would be a more useful metric than the likelihood when analysing a system. Matulevičius continues to argue that, when modelling a system, the capabilities of the 'threat agent' need to match the real-world, similar to the method applied by Fielder et al. [64].



Weakness	Description	Location	Exploitability
w1	Internet Malware	T1, T2	0.8
w2	Removable Drive Malware	T1	0.6
w3	DoS Attacks	T3	0.7
w4	Buffer overflow	T3, T4	0.8
w5	Man-in-the-middle	T4	0.6

(b)

Control	Control Descriptions	Combat
c1	anti-virus software	w1
c2	removable media check/disable	w2
c3	IDS and Firewalls	w3
c4	data validation & software security analysis	w4
c5	fine-grained access controls & integrity check	w5

(c)

Figure 3.5: Bayesian ICS Threat Model [95], showing the graph network (a), threats (b) and controls that can be placed (c).

Li and Hankin extend the work of Fielder et al. [95] to define a metric that measures a system's tolerance to 'zero-day' attacks. The authors use Bayesian networks to model a simple ICS environment and look at the effects of applying controls (e.g. deploying firewalls

or anti-virus solutions) to the network (shown in Figure 3.5). In their model, they assess the progressive effect of zero-days with time to determine which combination of controls provides maximal defence against such vulnerabilities. This approach is similar to that of HYRIM [80], where a combination of both approaches could be effective. Again, similar to some of the other approaches outlined in this section, one aspect that the asset owner has to carefully control is the threshold metric, as its accuracy is critical.

In the United Kingdom, the National Cyber Security Centre (NCSC) introduced the Cyber Assessment Framework (CAF) [1] as a way to bridge the gap for asset owners to assess their level of compliance with the NIS Directive. The CAF framework is broken down into 'NIS objectives', aligned with the The National Institute for Standards and Technology (NIST) framework, where asset owners evidence 'indicators of good practice', to *protect*, *detect* and *respond* for NIS compliance. This is unlike some other standards, e.g. the PCI-DSS standard which governs the management of payment card data, where asset owners can 'check off' their compliance to the standard, compared to CAF which is evidence-based, similar to ISO 27001. The framework, however, requires a level of cybersecurity knowledge to be able to confidently assess compliance with each objective. One limitation of CAF is that no automated tool exists to demonstrate efficacy and compliance to the principles, which would otherwise allow faster, more accurate assessments.

Mann details the human aspects of social engineering from an industry perspective [101]. This is another consideration when determining the threats that exist within an organisation and how assets may not have a computational vulnerability, but one exploitable through human intervention. Whilst his mapping technique was applied to enterprise data, a similar approach can be used to consider the human element in ICS environments. This is a factor which has been previously highlighted, but there is no simple method to model human behaviour. Beautement et al., however, define a method to formally model this human element [11], as

[1] https://www.ncsc.gov.uk/guidance/nis-directive-cyber-assessment-framework

part of a wider analysis on the trade-off of memory stick security. Similarly, Caulfield et al. assess the human element and model it formally [38], using a similar agent-based approach to Fielder et al. [64].

3.5.2 Risk Analyses of ERTMS

ERTMS-specific analyses have been previously conducted by KPMG and Bloomfield et al. [91, 18]. The KPMG report focused on the key management scheme, highlighting that key installation is vendor-specific and how the key hierarchy is defined. The analysis by Bloomfield et al. focuses on components e.g. the EuroBalise and potential vectors, identifying where there is a lack of authentication in the architecture. Their analysis is given at a high-level and does not provide any details of the specific issues identified.

A process was outlined in a later publication by Bloomfield et al. [19] where a methodology is defined for carrying out a risk assessment of ERTMS. In this methodology, the authors consider the trust relationships that exist in the architecture and the potential attack strategies an adversary may use. In their risk assessment, a number of outcomes that should be avoided are outlined which, today, would constitute a NIS violation if the threshold was exceeded. A 'lessons learnt' retrospective review of the work, undertaken in the UK, assessing the risk of a national deployment, is outlined by the authors, where the impact assessment, given as attacker capability levels, closely matches that of Braband [23]. One exclusion from the scope of their assessment is information leakage which, itself, should be considered a focus of assessment. As the authors note, information that could be potentially leaked may include specific details about hazardous cargo which, at the hands of an attacker, could have severe consequences. What is highlighted by the authors is that there is a need for a holistic analysis of architectures, both externally to identify relationships between systems and also internally, with in-depth assessments of the components used.

An alternative approach for conducting high-level security assessments is defined by Evans

et al. [60], outlining a process which can be applied in the rail industry to identify potential risks prior to commencing an expert security assessment. In this process, the authors outline three steps as part of the assessment: *identify*, *describe* and *assess*. In the first two phases, the rail asset owner specifies their critical assets and priorities, before gathering information about the interdependencies and interactions that take place within the system under assessment. One potential issue that may arise in these phases is that they may lack sufficient detail which would be captured in the final step, when an expert assessment is conducted. Both this approach, and the evidence-based CAF framework, however, are wholly different and manual processes, where this particular approach would allow asset owners to establish some baseline for security which they can improve upon.

In the wider field of SCADA, Igure et al. argue that the open standards, which ICS systems are based upon, are part of the problem [81]. The authors outline the potential attack vectors that could exist in SCADA architectures, additionally highlighting the challenges in mitigating potential threats due to the currently existing technology constraints, for example, limited computational power, a factor that was identified by Pièrre-Cambacédès and Sitbon [113]. Another assessment of SCADA threats was conducted by Cárdenas et al. [37], providing a chronology of SCADA attacks, similar to the safety-focused review outlined by Fantechi [61]. The authors note similar challenges to secure SCADA systems to those identified by Igure et al., but also quantify the difficulties in identifying when an attack is taking place. Their work also considers the complexity of patching and resolving vulnerabilities, highlighting a worst-case scenario from the power sector. An attempt is made to formalise this into a framework but, as the authors note, as there are no case studies available, it is difficult to appraise its capabilities.

3.6 Chapter Summary

In this chapter, it is clear that the rail sector has been primarily safety-focused, albeit with increased efforts to assess and improve the security of the sector. However, the security schemes introduced to date lack any form of future-proofing, where previous work shown in this chapter has identified attacks or technical issues in the standards which should be resolved prior to mass deployments of ERTMS.

Providing sufficient security, however, is not a sector-specific issue, where a similar conclusion can be drawn from other ICS applications. As we have observed in this chapter, asset owners may not understand the intricate detail of their architectures, such that they may be unaware of the vulnerabilities that exist.

Through the application and assessment of the methods reviewed in this chapter, we are able to develop and establish a baseline of the current security of a given architecture which allows us to conduct detailed security assessments to identify vulnerabilities in existing systems, using formal verification of protocols and cryptanalysis and security assessments. We can then develop methodologies which can be used in our assessment of emerging threats and exploitations.

Part I

Diving down the ERTMS Stack

Chapter 4

Formal Verification of ERTMS Protocols

In this chapter, we will examine a formal analysis of the train to trackside communications used in ERTMS and, in particular, the EuroRadio protocol. As part of this analysis, we will set a number of security goals that the protocols should conform to and make recommendations that will ensure the future security of these protocol layers, as part of the secure architecture for ERTMS.

4.1 Motivation

The European Train Control System (ETCS) component of ERTMS is composed of a suite of protocols and standards. For example, the EuroRadio protocol is used to ensure that messages exchanged between train and trackside entities are genuine and have not been forged or modified by an attacker. It is also used to manage the effective handover of trains from one area of control to another. For the UK rail network, which is a largely analogue, semi-automatic system, the transformation to a wholly-digitised, fully supervised system, may expose it to threats not previously deemed possible.

During the deployment of a new system, it is prudent to carry out appropriate analysis to ensure that the replacement systems protect the underlying infrastructure and operational

rolling stock from what were previously considered as 'impossible' attacks. As ERTMS is a major, safety-critical system, it is essential to ensure that trains and Radio Block Centres (RBCs) cannot be externally influenced into an unpredictable, potentially unsafe state, or be coerced by an attacker into sending out unsafe instructions.

Currently, attacks are typically perceived by the public to be some exploited flaw or weakness in vendor systems, rather than potential weaknesses in the underlying standards. For ERTMS, the standards underpin all vendor equipment where the protocols, cryptography and application messages provide consistency and interoperability between proprietary vendor systems, e.g. interlocking systems. We commonly think of exploits being the result of backdoors or through precise trigger conditions being met which put the attacker in a privileged position. Whilst this may be the case, when new systems are implemented, it is critical that the standards, from a security perspective, are assured prior to implementation. This ensures that vulnerabilities are more likely to be due to incorrect implementation or error on a vendor's part [134, 135].

Automated protocol analysis is one way in which we can assure the security of protocols although, as discussed in Chapter 2, it does not guarantee the total security of the system. This, when coupled with analysis of the cryptographic primitives used by the protocol can, however, show that, from the perspective of the standards, they are secure from the outset.

4.2 Contributions

In this work, we carry out a formal analysis of the EuroRadio protocol and parts of the higher-level Application Layer protocol, using the applied pi-calculus [1] and ProVerif analysis tool [14, 17]. Prior to this research, there was no known formal verification of the suite of protocols used in ERTMS, as most previous research has primarily focused on correctness issues [18] or assessing the general security and possible attacks that are sector-specific [19].

As part of this analysis, we explore the Application Layer timestamps in ProVerif, illus-

trating that, whilst the replaying of messages is not possible, the Application Layer itself does not prevent messages from being dropped in transmission. From the perspective of EuroRadio and the Application Layer, we then assess the resilience of the standards against a number of requirements to which, we believe, any safety-critical protocol should comply.

We will finally consider in this chapter the potential issues that arise from the use of these protocols and how they can be remediated in future iterations of the standard.

4.3 Background

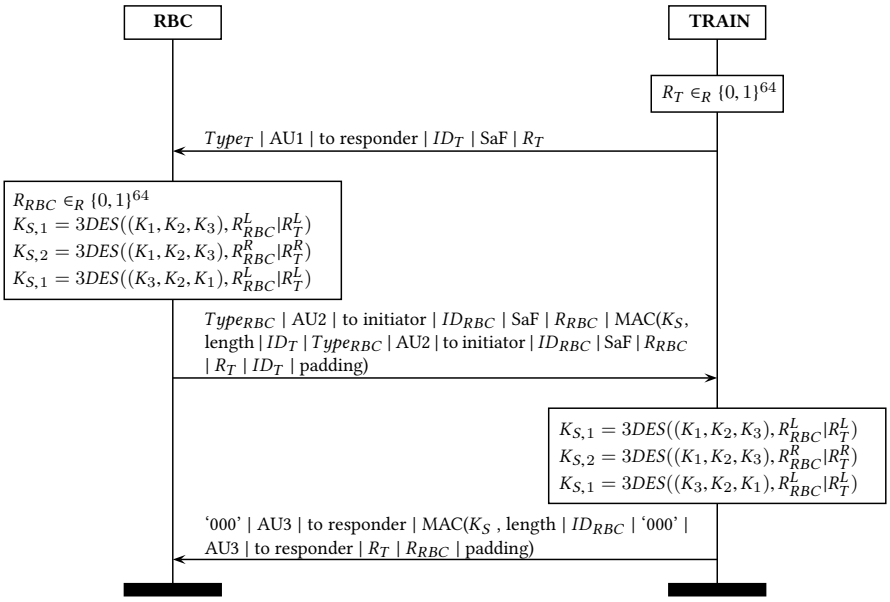
In this section, we will outline the purpose of the EuroRadio protocol and Application Layer in ERTMS and provide an intuition about the requirements imposed on these layers.

4.3.1 The EuroRadio Protocol

When a train is due to start its journey, it will contact the RBC responsible for the area in which the train is located. Once a data-link has been set up over the GSM-R connection between the train and RBC, a handshake protocol (EuroRadio) will take place. The main aim of this protocol is to verify the identity of each party and the authenticity of messages sent between train and trackside. During this handshake, the train and RBC broadcast their identities, establish a session key and negotiate a Safety Feature (SaF), the cipher to be used for message authentication.

In Figure 4.1, we outline the various steps taken by each party to authenticate themselves before any Application Messages are sent. Once authenticated, messages may be sent between train and trackside. We will then examine, in detail, each step of this protocol and its function.

Prior to a EuroRadio handshake taking place, there are a number of pre-requisites, specifically that all entities are allocated with an ETCS Identifier, centrally allocated by the European Rail Agency (ERA) and that a pre-shared 3DES master key, KMAC, has been created and distributed between the two entities. This 3DES key is unique per train-RBC pairing and is used



Key	Description
AU_x	Authentication Message x
ID_x	ETCS Entity ID of party x
R_x	Nonce generated by party x
SaF	Safety Feature selected
To Initiator/Responder	Flag to indicate in which direction the message is being sent
$Type_x$	ETCS Entity ID type (e.g. RBC or train)

Figure 4.1: The EuroRadio Handshake Authentication Protocol. The RBC and Train share a unique, symmetric key, KMAC. R^L and R^R are used to indicate the leftmost and rightmost 32 bits respectively of a given nonce. The KMAC is a 3DES key, composed of three single DES keys: KMAC = (K_1, K_2, K_3).

to authenticate handshake messages and negotiate the session key for future messages, as part of this specific ERTMS session.

At the start of the handshake, the train sends the first message, generating a 64-bit nonce (R_T) and sends this to the RBC. This message will contain the type of ERTMS entity, followed by AU1, a direction flag, the ERTMS Identity of the train (allocated by the ERA), the proposed SaF and its nonce.

The AU Safety Protocol Data Unit (SaPDU) is a collection of flags which determine the

progress of the EuroRadio handshake. AU1 is the first authentication message, followed by AU2 and finally AU3, where the handshake is complete and data may be sent. At each stage of the protocol, validation takes place, e.g. at AU1, the RBC requires a shared KMAC key between itself and the train. If it does not have a key, then the protocol will enter an error state and the EuroRadio connection is terminated. The same response occurs if there is a mismatch in the MAC received compared to the data sent. The direction flag is a binary value, indicating whether the message is being sent to a recipient or the original sender (relative to the entity sending the message).

If the RBC has a key corresponding to the received train ID, it will then generate its own nonce and carry out a key derivation step, taking its 64-bit nonce, the train's 64-bit nonce and generate three keys, based on halves of each nonce, where the product is KSMAC, a session key used to MAC all future messages. Once the key has been generated, the RBC will respond with its ETCS entity type, the AU2 SaPDU, direction (to the train), its ETCS identity, the SaF to use, its nonce and then a MAC over the message, using the derived session key.

Upon receipt of this message, the train will derive the session key using the same process that the RBC followed, verify the MAC of the received message and respond with an AU3 message, confirming the completion of the handshake and allowing Application Messages to be sent to and from the RBC.

4.3.2 Application Message Layer

Sitting above the EuroRadio layer, the Application Layer is responsible for ensuring timely receipt of messages and delivery to the onboard European Vital Computer (EVC). Within this layer is a 32-bit counter-style timestamp, measured in milliseconds, relative to the train's clock. The RBC is expected to maintain multiple clocks, one per active session, relative to that of the train. Messages are passed up from the Application Layer, if and only if, the timestamp on the message received is greater than that of the last received message. Any time greater than the

last received message will, therefore, be accepted.

Assumptions are placed on the underlying EuroRadio layer that messages have been appropriately authenticated with the only exception being 'High-Priority' messages, where a flag in the DT (Data) SaPDU indicates the priority of the message, either normal-priority or high-priority. Only a subset of messages (currently Emergency Stop) may be sent with this flag. This forces the EuroRadio MAC checks to be bypassed, ensuring that the Application Layer receives the message as soon as possible. The justification for this will be questioned later in this chapter.

4.4 Formal Modelling in ProVerif

In this section, we will introduce the ProVerif automated verifier which is used to carry out the analysis of EuroRadio and part of the Application Layer protocols.

ProVerif leverages the applied-pi calculus, which allows us to specify processes capable of receiving input and generating some output and, more importantly, can run in parallel and replicate. Running processes in parallel reduces the time required to analyse protocols, whilst replication allows new processes to be created as required. Alternative tools could also be used for this purpose, including Tamarin [104] and Scyther [47].

The applied-pi calculus (as shown in Figure 4.2) allows us to declare new, private, names within processes which can be used to simulate, for example, a private channel or nonce. The functions provided by the calculus can also be used to model a range of cryptographic primitives, namely MACs, encryption, signatures and key generation. These, however, are abstractions from any implementations of cryptography, where they are assumed to be cryptographically perfect and conform, in their entirety, to the specification. In this analysis, we focus only on the verification of the protocol, rather than any known existing weaknesses or exploits as the direct result of the cryptographic schemes used. Consequently, we can provide assurance of the protocols, as defined in the standard.

$M, N ::=$	terms
x, y, z	variables
a, b, c, k, s	names
$f(M_1, \ldots, M_n)$	constructor/deconstructor application
$P, Q ::=$	processes
0	nil
out $(M, N).P$	output N on channel M
in $(M, x).P$	input x from channel M
$P \mid Q$	parallel composition
$!\,P$	replication
new $a.P$	create new name
if $x = M$ then P else Q	term evaluation (if statement)
$event(x)$	execute event x

Figure 4.2: ProVerif syntax [17], an application of the classic Applied Pi Calculus syntax.

In ProVerif, we can use the applied-pi "`let`" statement to express "`if`" statements and conditional inputs to branch, based on the input received. We can also force explicit binding for input to a specific variable to match using "=". As an example, given a channel c, we can ensure that a variable a transmitted is exactly the same value as a using the statement $in(c, = a)$. If we receive something that is not equal to a, then the process will terminate. In Figures 4.3 and 4.4, flags such as `AU1`, `AU2` and direction are explicitly checked in this way.

The calculus also supports methods to report back the current state at key points in the process, through 'Events'. These can be placed at any point in the model but are used at critical points, where they can be parameterised with variables held within the model. In the case of ProVerif, the tool is able to provide guarantees for soundness where, if no attack is found, the protocol is correct. The tool, however, is not complete, as it could potentially return false attacks [17]. Events, therefore, provide a means for manual verification of possible attacks returned by the tool.

ProVerif supports several types of queries which enable us to check the security properties of a given protocol. The most basic queries check for secrecy in a protocol, i.e. if an attacker is able to learn specific values which are not meant to be openly communicated, for example,

```
1  let Train =
2    (* Set up a new session for the model *)
3    (* Create a fresh session identifier used to link different events in the model *)
4   new session;
5   (* Get the identity of the RBC the train wants to communicate with *)
6   in(id, rbc_etcs_id);
7   (* Start of the actual authentication protocol *)
8   (* TCONN.request  Au1 SaPDU *)
9   new trainNonce;
10  event trainStartSession(rbc_etcs_id, train_etcs_id, trainNonce, SAF);
11  out(c, (TRAIN_ETCS_ID_TYPE, AU1, DF_SEND, train_etcs_id, SAF, trainNonce));
12  (* TCONN.confirmation  Au2 SaPDU *)
13  in(c, (=RBC_ETCS_ID_TYPE, =AU2, =DF_RESP, in_rbc_etcs_id, rbcSaF, rbcNonce, inMAC));
14  (* Generate the session key *)
15  let trainKS = genSessionKey(trainNonce, rbcNonce, getKey(in_rbc_etcs_id, train_etcs_id)) in
16  (* Output encrypted secret to check secrecy of keys *)
17  out(c, encrypt(SECRET, trainKS));
18  out(c, encrypt(SECRET, getKey(in_rbc_etcs_id, train_etcs_id)));
19  (* Verify whether the received MAC is correct *)
20  if inMAC = mac(trainKS, ((PAYLOAD_LENGTH, train_etcs_id, RBC_ETCS_ID_TYPE, AU2,
        DF_RESP, in_rbc_etcs_id, rbcSaF), rbcNonce, trainNonce, train_etcs_id)) then
21  (* TDATA.request  Au3 SaPDU *)
22  event trainFinishSession(in_rbc_etcs_id, train_etcs_id, trainNonce, rbcSaF, rbcNonce, trainKS);
23  out(c,(ZEROS, AU3, DF_SEND, mac(trainKS, (PAYLOAD_LENGTH, train_etcs_id, ZEROS, AU3,
        DF_SEND, trainNonce, rbcNonce))))
```

Figure 4.3: The ProVerif model of a train carrying out the EuroRadio Handshake (the calling party).

cryptographic keys. Another type of query that ProVerif supports is correspondence assertions which can be used to verify whether, in the case of a particular event being executed (i.e. reached), another prior event was executed. A typical application of this could be to connect an input/output operation between two processes. Two correspondence assertions, non-injective and injective, can be verified within the ProVerif tool. As an example, let us consider the subtle differences between these assertions. Non-injective queries, e.g. `ev:event1(vars) ==> ev:event2(vars)`, hold if `event2` was executed at *some* point before `event1`. Injective assertions, however, e.g. `evinj:event1(vars) ==> evinj:event2(vars)` hold, if and only if, for every execution of `event1`, there was a *unique* execution of `event2`.

4.4.1 Modelling the EuroRadio Protocol

As EuroRadio only involves two parties (the train and RBC), it is fairly straightforward to map the protocol into its constituent processes. In Figures 4.3 and 4.4, we see the result of transforming the protocol described in Figure 4.1 into a ProVerif model. We then proceed to analyse the normal-priority messages sent between the train and RBC for each process, in Figures 4.5 and 4.6 respectively. A further model was created to analyse high-priority messages and demonstrate how it is possible for an attacker to bypass the authentication and timestamp checks, which are included in Appendix A.

```
1  let RBC =
2    (* Set up a new session for the model *)
3    (* Initialise with the RBC ID *)
4    in(id, rbc_etcs_id);
5    (* Start of the actual authentication protocol *)
6    (* TCONN.indication  Au1 SaPDU *)
7    new rbcNonce;
8    in(c, (sent_ETCS_ID_TYPE, =AU1, =DF_SEND, in_train_etcs_id, trainSaF, trainNonce));
9    event rbcStartSession(rbc_etcs_id, in_train_etcs_id, rbcNonce, trainSaF, trainNonce);
10   (* Generate the session key *)
11   let rbcKS = genSessionKey(trainNonce, rbcNonce, getKey(rbc_etcs_id, in_train_etcs_id)) in
12   (* Output encrypted secret to check secrecy of keys *)
13   out(c, encrypt(SECRET, rbcKS));
14   out(c, encrypt(SECRET, getKey(rbc_etcs_id, in_train_etcs_id)));
15   (* TCONN.response  Au2 SaPDU *)
16   out(c, (RBC_ETCS_ID_TYPE, AU2, DF_RESP, rbc_etcs_id, trainSaF, rbcNonce, mac(rbcKS, ((
         PAYLOAD_LENGTH, in_train_etcs_id, RBC_ETCS_ID_TYPE, AU2, DF_RESP, rbc_etcs_id,
         trainSaF), rbcNonce, trainNonce, in_train_etcs_id))));
17   (* AU3 SaPDU *)
18   in(c,(=ZEROS, =AU3, =DF_SEND, inMAC));
19   (* Verify whether the received MAC is correct *)
20   if inMAC = mac(rbcKS, (PAYLOAD_LENGTH, in_train_etcs_id, ZEROS, AU3, DF_SEND ,
         trainNonce, rbcNonce)) then
21   event rbcFinishSession(rbc_etcs_id, in_train_etcs_id, rbcNonce, trainSaF, trainNonce, rbcKS)
```

Figure 4.4: The ProVerif model of the RBC in the EuroRadio protocol (the called party).

The expressive nature of the syntax for ProVerif allows us to define the processes run by the verifier in a number of ways. In the model presented in Figure 4.3, we create and initialise

```
(* Send three messages from the train to the RBC *)
new time;
let msg1 = (DT, time, MESSAGE_1) in
event DataSent1(session, msg1);
out(c, (msg1, mac(trainKS, msg1)));
let msg2 = (DT, inc(time), MESSAGE_2) in
event DataSent2(session, msg2);
out(c, (msg2, mac(trainKS, msg2)));
let msg3 = (DT, inc(inc(time)), MESSAGE_3) in
event DataSent3(session, msg3);
out(c, (msg3, mac(trainKS, msg3)))
```

Figure 4.5: ProVerif model of the Application Layer sending normal priority messages.

both instances for the RBC and train as replicating processes, through the '!' command. These may be nested, i.e. giving us an arbitrary number of train and RBC processes which can be run in parallel. This ensures that the verifier can carry out a thorough examination of the protocol and, from the perspective of an attacker in ProVerif, allows it to reuse variables it has observed from previous protocol runs. Given a set of properties defined within the model, we can assess whether these hold using ProVerif and, if not, the tool will provide a trace if a possible attack is found.

In Figures 4.3 and 4.4, we need to include an additional variable that is not part of the EuroRadio protocol, the `session`. This will be used during verification to show the reordering and replaying of messages so that we can observe the session from which these messages originated. During the setup process, we also note that the train and RBC processes are sent the ID of the RBC (`in(id, rbc_etcs_id)`). This allows us to assert that the train knows the identity of the RBC to which it should be connecting. We then directly model the EuroRadio specifications from the start, where the nonces and identities of the parties are exchanged, with the inclusion of the session key derivation step. In the next step, we generate and output a secret value, encrypted with the negotiated session key. We validate the confidentiality of this secret value to verify whether the attacker is able to establish the session key, which should not be possible. At each stage of message reception we verify the MAC appended to

messages before proceeding. This simulates the process within EuroRadio and the expected behaviour when EuroRadio is implemented by vendors. Once the AU3 SaPDU has been sent by the train, the EuroRadio link is established. From here, we can take one of two routes. Firstly, we can either apply the model to normal-priority messages, or, alternatively, model high-priority messages.

```
1  (* Receive messages from the train *)
2  in(c, ((=DT, timeA, msgA), macA));
3  (* Check the MAC of the received message *)
4  if macA = mac(rbcKS, (DT, timeA, msgA)) then
5  event DataReceived1((DT, timeA, msgA));
6  in(c, ((=DT, timeB, msgB), macB));
7  (* Check the MAC and timestamp of the received message *)
8  if macB = mac(rbcKS, (DT, timeB, msgB)) then
9  if greater: timeB, timeA then
10 event DataReceived2((DT, timeB, msgB));
11 event MessagesReceived2((DT, timeA, msgA), (DT, timeB, msgB));
12 in(c, ((=DT, timeC, msgC), macC));
13 (* Check the MAC and timestamp of the received message *)
14 if macC = mac(rbcKS, (DT, timeC, msgC)) then
15 if greater: timeC, timeB then
16 event DataReceived3((DT, timeC, msgC));
17 event MessagesReceived3((DT, timeA, msgA), (DT, timeB, msgB), (DT, timeC, msgC))
```

Figure 4.6: ProVerif model of the Application Layer, receiving normal priority messages.

In Figures 4.5 and 4.6, we observe the Application Messages sent through EuroRadio, where MESSAGE_X can represent any message sent between a train and RBC, including the application of timestamps. Once a EuroRadio session has been established, we generate a value for the timestamp and proceed to use it when sending messages. Each time a message is sent, we use a light-weight notion of time, which we will review shortly. Upon receipt of the message from the train, the RBC will verify the MAC and then the timestamps of the message. If the timestamp in this message is greater than that of the previous received message, the message will be accepted and execute the appropriate ProVerif event to indicate that it was received in the context of that session. When the event is executed, we include the session identifier to ensure that an attacker cannot combine messages from different sessions.

```
1 data inc/1.
2 pred greater/2.
3 clauses
4   greater: inc(x),x;
5   greater: x,y > greater: inc(x),y.
```

Figure 4.7: ProVerif model which captures counter-style timestamps. Timestamps are incremented using the *inc* predicate and we can compare two timestamps using the *greater* predicate.

4.4.2 Modelling Counter-style Timestamps in ProVerif

ProVerif and the applied-pi calculus do not support notions of timestamps. However, for the validation of the Application Layer component of the protocol, we need to assure its behaviour from a formal analysis perspective. To support the validation of timestamps, we have a minimal notion of time added to the model. In the Application Layer, we check whether the timestamp on a given message is greater than that of the previous message. In the model, we instantiate the time by using a relative train-borne counter. Time can increase using this model, enabling us to compare different timestamps based on the same initial timestamp. This means that we have no notion for how much time an action may take but, in the context of the model, this proves to be adequate.

In Figure 4.7, we define this notion of time, where a timestamp can be incremented using the *inc* operator and two timestamps can be compared using the predicate *greater*.

4.5 Assessing the Security Goals of Protocols

Using ProVerif, we can verify whether the protocol preserves the secrecy of keys and that an attacker cannot disrupt the handshake and key agreement process. Whilst these checks are standard ProVerif queries, we also want to ensure that the protocol meets expectations that would have an impact on safety and security if they were not held. It should not be possible for an attacker to insert, reorder, replay or delete messages without being noticed. We can consider ways that these checks can be carried out, specifically by making the train send three

messages to the RBC and tagging them with a particular event. Additionally, events are used to tag the messages sent by the train, their particular order and the resultant three messages received by the RBC and their order. From this analysis, we can check for the insertion, re-ordering, replaying or deletion of messages using queries on these events. The results of this analysis (corresponding to EN50159) are summarised in Figure 4.8. However, this analysis goes beyond EN50159 and includes verifying the secrecy of keys, agreement on a shared key, mutual authentication and the analysis of high-priority messages.

For each of the properties, an intuition is given for the possible impact if that security goal was breached in some way. We will then explore appropriate recommendations and their feasibility and any issues identified using this model-based approach in Section 4.6.

Threat	EuroRadio	Application Layer
Repetition	×	✓
Deletion	×	×
Insertion	✓	×
Re-sequencing	×	×
Corruption	✓	✓
Delay	×	×
Masquerade	✓	×

Figure 4.8: Analysis Summary as a Threat/Defence Matrix corresponding to EN50159 [29].

4.5.1 Secrecy of Keys

As EuroRadio is a safety-critical component for train to trackside communications, it is imperative that the cryptographic keys used to provide authenticity and trust to the platform cannot be recovered by an active attacker, through flaws in the protocol for both the long-term train-RBC key, *km* and the session key, *ksmac*. We can check this by creating a new, private value, 'SECRET', encrypting it using both the long-term and session key and publicly broadcasting this over the channel to which the attacker has full access. If the attacker can learn the value 'SECRET', it means that the keys were recovered or learnt by them. In ProVerif, this can be expressed by using the query `attacker:SECRET`, which verifies if the attacker is

able to establish a given value – in this case, SECRET. Private values are not disclosed to the attacker and 'SECRET' is only output on the public communication channel encrypted using the long-term and session keys. Therefore, if the attacker was able to learn the value, 'SECRET', then one of the keys must have been compromised.

Running this ProVerif model, we find that the attacker cannot learn the value 'SECRET', confirming that the EuroRadio protocol succeeds in its main goal of keeping the cryptographic keys secure from an active, Dolev-Yao attacker, who is interfering with the protocol, as they cannot guess the keys. The theorem proving method of ProVerif further assures us that this holds for an unlimited number of runs of the protocol.

If ProVerif produced an attack trace proving otherwise, such that 'SECRET' was able to be recovered, then the attacker also has knowledge of the shared session key. In the context of EuroRadio, this key provides authenticity and integrity for all future messages exchanged between the train and RBC. Moreover, most of these messages are safety-informed decisions, e.g. Movement Authorities where, if an attacker had access to the key used in the EuroRadio MAC, they would then be able to reliably forge messages with valid, trusted, MACs which would be accepted by the train. Some additional technical caveats exist if this was possible, for example, upon receipt of a message, the train would acknowledge it to the RBC [140]. To prevent the RBC being made aware of the attack, the GSM-R data connection could be temporarily jammed by an adversary.

4.5.2 Agreement on a Shared Session Key

EuroRadio provides a means to negotiate a shared session key such that, even if the attackers cannot learn the initial shared key or session key, they may still attempt to interfere with the key establishment process. If successful, a session key would not be able to be negotiated, leading to possible denial of service and the EuroRadio link never successfully becoming established between a train and RBC. In ProVerif, we use the following injective correspondence

assertions:

```
evinj:trainFinishWithKey(ks) ==> evinj:rbcUsing(ks)
evinj:RBCFinishWithKey(ks) ==> evinj:trainUsing(ks)
```

These queries will only hold if, whenever a train believes it has successfully completed the EuroRadio handshake protocol, having established the session key, *ks*, there was then a single RBC that has also run the protocol and believes the established key is also *ks* and vice versa. ProVerif returns a confirmation that these queries hold, establishing that EuroRadio succeeds in its second major security goal by successfully negotiating and setting up a shared key between the train and RBC.

If an attacker was able to affect the shared session key, the accompanying EuroRadio MACs between the train and RBC would not be equivalent when the MACs are verified. Under the current standard [138], the MACs are rejected and the connection between the train and RBC may be terminated [140] on the first indication that an error has occurred. Furthermore, this would lead to a possible denial of service where an attacker, intervening in sessions, for example, at Clapham Junction, the UK's busiest rail junction, could lead to significant delays which would propagate throughout the network.

4.5.3 Mutual Authentication – Agreement on All Shared Values

During the EuroRadio handshake, critical values determine the negotiated session key, what MAC cipher to use (SaF) [1] and the identities of the train and RBC. In the model, we extend the queries to investigate the malleability of these values to ensure that, at the point the train and RBC have completed the handshake, the values that are agreed and the train and RBC are aware of, remain unchanged.

The model is further extended with the following queries:

[1] Currently, only one cipher is allowed (with the exception of CRC used in Italy). However, in future baseline standards, this may change, based on nationally-defined requirements.

```
evinj:trainFinishSession(rbc_id,train_id,train_nonce,saf,rbc_nonce,ks) ==>
      evinj:rbcStartSession(rbc_id,train_id,rbc_nonce,saf,train_nonce)

evinj:rbcFinishSession(rbc_id,train_id,rbc_nonce,saf,train_nonce,ks) ==>
      evinj:trainStartSession(rbc_id,train_id,train_nonce,saf)
```

We observe that the first correspondence assertion holds – when a train confirms it has finished the handshake, it has agreed the same values as the RBC. However, the second assertion does not hold, where ProVerif is able to provide an attack trace [1]. In this trace, we observe that the attacker was able to redirect the initial message sent by the train to a different RBC. This is possible because the train does not explicitly verify the identity of the RBC, nor is there any remediation if it is shown to be incorrect. At the same time, it is possible for the MAC cipher (SaF) to be modified in-flight by the attacker, where a less secure cipher could be forced upon both entities.

Following on from the RBC example given above, the threat presented here is that a train which has started its 'mission' may not have an accurate location reference (e.g. in the case of stabled or recovered units). As a result, the RBC may issue a 'staff-supervision' authority, where the train is allowed to be moved forward to the next balise at low speed until it is able to report an authoritative location. In the event that the train is redirected to a RBC that does not have authority for that region of track, it is possible that the train could be placed in a conflicting section of line with another train.

4.5.4 Ability to Insert Attack Messages

Once the handshake has been completed, we want to ensure that it is not possible for an attacker to arbitrarily insert their own messages to the train/RBC. In the ProVerif model, we use the `DataSent'i'(m)` event to express that some message, `m`, was the `i`-th message sent

[1] Attack traces and full models are available at `http://research.rjthomas.io/rssrail2016`.

by the train and received by the event. Using this event, we can check if an attacker has been able to insert their own message into the protocol where, for all messages `m` received by the RBC, they must have been sent by the train as one of three messages. The queries are shown below:

```
ev:DataReceived1(m) ==>
 (ev:DataSent1(s2, m) | ev:DataSent2(s2, m) | ev:DataSent3(s2, m))
ev:DataReceived2(m) ==>
 (ev:DataSent1(s2, m) | ev:DataSent2(s2, m) | ev:DataSent3(s2, m))
ev:DataReceived3(m) ==>
 (ev:DataSent1(s2, m) | ev:DataSent2(s2, m) | ev:DataSent3(s2, m))
```

Here, we observe that for each of the messages received, they arrived in some order but only for those exact three sent by the RBC. ProVerif confirms that this goal holds and the attacker cannot arbitrarily insert their own messages.

The insertion of arbitrary messages by an attacker presents an interesting issue, specifically that the EVC onboard the train would be required to process each message. An attacker could use this as a way of carrying out a denial of service on the train/RBC by storming them with a large volume of messages. Those messages do not need to carry a valid MAC, as EuroRadio will still attempt to verify the MAC and accept/reject the message. If a train cannot handle the volume of messages, it will stop [18]. For a RBC, however, many trains would be forced to stop, leading to significant disruption, considering the area of control for which RBCs are responsible.

4.5.5 Ability to Replay Messages

Whilst we are able to ensure that an attacker cannot arbitrarily insert their own messages into the EuroRadio stack, they may still be in a position to capture valid messages between the train and RBC and subsequently replay them. The receiver would be tricked into thinking

these messages are new and, as they carry a valid MAC, it would rely on the Application Layer to detect this type of attack. We further develop this possibility by modifying the queries for the insertion to be injective, i.e., for each `DataReceived` event, there must have been a *single, unique* `DataSent` event:

```
evinj:DataReceived1(m) ==>
 (evinj:DataSent1(s,m) | evinj:DataSent2(s,m) | evinj:DataSent3(s,m))
evinj:DataReceived2(m) ==>
 (evinj:DataSent1(s,m) | evinj:DataSent2(s,m) | evinj:DataSent3(s,m))
evinj:DataReceived3(m) ==>
 (evinj:DataSent1(s,m) | evinj:DataSent2(s,m) | evinj:DataSent3(s,m))
```

ProVerif shows no possible attack traces, with all correspondence assertions holding, meaning that the attacker is unable to replay messages. This is due to the counter-style timestamps having a requirement that the timestamp of the message received must be greater than that of the last message. Replayed messages will be allowed through the EuroRadio MAC validation but will be detected at the Application Layer when the timestamps are validated.

If an attack was to be found, it would mean that specific messages, e.g. a 'Revoke Emergency Stop' order would be invaluable to an attacker. In the event where an emergency stop message has been previously issued by the RBC, the attacker could simply replay the revocation message, which may have serious implications such as allowing a train to request a new Movement Authority when it is unsafe to do so. Again, as identified previously, the attacker could still attempt other attacks, e.g. denial of service, as the MAC would be valid, but would prevent other messages being received as a queuing mechanism is in use [140].

4.5.6 Ability to Reorder Messages

An alternative means by which an attacker could affect the normal operation of a train is by reordering messages, e.g. swapping the order of a 'Movement Authority' and 'Reduce Move-

ment Authority' message. Given no additional messages are required and they are unique, the previous two correspondence assertions (insertion and replay) do not attest EuroRadio and the Application Layer's ability to prevent arbitrary reordering of messages.

As a result, we require an alternative event and query to be used to verify the precise ordering of messages. This is given in the `MessagesReceived3(m1, m2, m3)` event, which holds if messages `m1`, `m2` and `m3` were received in that exact order. Using this event, we can build an injective correspondence assertion as follows:

```
evinj:MessagesReceived3(m1, m2, m3) ==>
(evinj:DataSent1(s,m1) & evinj:DataSent2(s,m2) & evinj:DataSent3(s,m3))
```

When proving the model, ProVerif shows this assertion holds. However, the attacker may still be able to prevent messages from being received and we will shortly assess how this is possible. An attacker could, therefore, block one of the messages from being received and reorder the remaining two (i.e. `MessagesReceived3` is never executed). Consequently, we also need to consider this possibility:

```
evinj:MessagesReceived2(m1, m2) ==>
  ((evinj:DataSent1(s, m1) & evinj:DataSent2(s, m2)) |
   (evinj:DataSent1(s, m1) & evinj:DataSent3(s, m2)) |
   (evinj:DataSent2(s, m1) & evinj:DataSent3(s, m2)))
```

In this event, we explicitly check the possible ordering of two messages and the corresponding event which generated these messages. Specifically, were the two messages received the first, second, or third output? The assertion considers the possible permutations where one could have been blocked, whilst the remaining two would have been successfully received. This correspondence assertion holds, confirming that, for three messages sent, they could not be reordered by an attacker.

As alluded to in the previous assertion, an adversary could try to force a train enter an unpredictable state by reordering messages, e.g. if a MA was issued and then reduced, reordering these two messages could have the effect of providing a MA that extends beyond a safe limit. Conversely, if two emergency stop messages were issued, e.g. the first being conditional, followed by an unconditional emergency stop (potentially as part of a 'Stop all Trains' command from the Rail Operation Centre (ROC)), a revocation message sent and the messages subsequently reordered, to which stop message does that revocation apply? However, this analysis shows that reordering is not possible.

4.5.7 Ability to Delete Messages without the Receiver Knowing

We have established, thus far, that it is not possible for an attacker to insert, reorder or replay messages. Blocking the successful reception of a message, however, is equally important and we need to ensure that there is a way of knowing if a message was not received. It should be the case that, for two messages, `m1` and `m2`, `m2` should not be accepted if `m1` was never received.

In the model, we can check this property using the following correspondence assertions:

```
evinj:DataReceived1(m) ==> evinj:DataSent1(s, m)
evinj:DataReceived2(m) ==> evinj:DataSent2(s, m)
evinj:DataReceived3(m) ==> evinj:DataSent3(s, m)
```

Here, for three messages, we ensure that for each of those messages, upon receipt, there must have been a corresponding send event. These assertions, however, include the additional verification to determine whether reordering and also the ability to delete messages was possible. As previously demonstrated, reordering is not possible. This set of correspondence assertions will, therefore, only hold if deletion is impossible, failing if a message can be successfully blocked/deleted, by an attacker, from being received.

Using ProVerif, we find that these correspondence assertions fail to hold where, given the counter-style timestamps must be greater than that of the last received message, there are no

simple means that can be employed by the recipient to detect the absence of a message. This can be partially mitigated by using acknowledgements and timeouts, as we will discuss in the following sections.

Deletion of messages have the potential to impede the safe operation of the railway. For example, if an emergency stop message was blocked by an attacker, the train would not receive the message and could be travelling into an unsafe situation.

4.5.8 Analysis of Emergency Messages

As outlined earlier, emergency stop messages may be sent without a corresponding MAC, where they bypass the EuroRadio layer and are immediately handed to the Application Layer protocol. As this behaviour is not typically found in some protocols, each test is re-run in a second model to assess the effect of the lack of a MAC on emergency stop messages. As previously found, the secrecy of keys, authentication of parties and the mutual agreement on shared values continue to hold, meaning that an attacker could not simply pretend to be a train or RBC, with communications only possible between a valid train and RBC.

That said, we now find that an attacker is able to insert, delete, reorder and replay messages successfully, with the correspondence assertions failing to hold, as the MAC is no longer verified. As a result, once a EuroRadio session has been set up, it is now possible for the attacker to insert an emergency stop message, which would be accepted by the train.

4.6 Discussion and Recommendations

In this section, we will discuss the possible solutions to address the issues identified from the analysis undertaken in this chapter.

4.6.1 Insertion of High-Priority Messages

When executing the second model, which carried out an analysis of the high-priority messages, we find that it is possible for an attacker to insert arbitrary messages as no protection

is offered to this class of message. It should be noted that only two messages can be sent as high-priority, a conditional and unconditional emergency stop. As a result, anyone who is able to access the EuroRadio communications layer has the ability to insert an unconditional emergency stop message into the stream, which would trigger the train to immediately engage its emergency brakes, without any driver intervention. Whilst this might not directly lead to an incident involving trains colliding, it has a disruptive capability, such as having crew and rolling stock displaced. This type of disruption has a more profound impact on the network, especially if the emergency stop was to be carefully timed, for example, when a train is in a known GSM-R blackspot with no reception. In this case, the RBC would not be aware of the emergency stop as it cannot communicate with the train and consequently unable to revoke the emergency stop. In the UK, a set of procedures exist [120] for the driver to follow until GSM-R coverage is available again. In the event of a train carrying out an emergency stop in this situation, it would take significantly longer to recover operations to the timetable after an emergency stop, which could propagate delays throughout the national network. It is important to note that for this attack to be successful, the adversary would need to be close to the target train and can only stop that particular train.

To prevent an attacker being able to send unauthorised messages, high-priority messages should be authenticated using MACs, as is the case with normal-priority messages. They would still have the same priority given to them by the EuroRadio layer as is the case currently when a high-priority message is received. This proposal presents us with a dilemma concerning the motivations for excluding a MAC in high-priority messages and whether this is still appropriate. One concern could have been that if the keys were corrupted at some point, either by the train or RBC, it should still be possible to fall back to voice communication (as is currently the case in most systems). The benefits of using a MAC in high-priority situations *would*, however, prevent the misuse of these messages by an external attacker, preventing them from successfully injecting messages in the communications between a train

and RBC. This, however, would not prevent an attacker causing disruptions by other means, e.g. jamming the GSM-R communications channel or physically damaging the infrastructure where simpler, but effective, attacks exist.

4.6.2 Deletion of Messages

As previously identified, the EuroRadio protocol does not protect against the deletion of messages and would need to be handled by the Application Layer. The use of counter-style timestamps also offer no protection against the deletion of messages. In ERTMS, the message sender has the ability to request acknowledgements upon receipt of the message, although this is not the default for all messages and has to be explicitly requested. We also note that recipients have no way of knowing if a message was deleted in-flight and, therefore, not received and not detected by the Application Layer. In the most extreme case, where an attacker prevents the reception (i.e. deletion) of emergency stop messages, a train may compromise the safe operation of the railway, entering a stretch of track which could conflict with the movement of another train.

It is difficult to prevent the total deletion of messages, as an attacker could jam all communications between two parties, but it is possible to detect the deletion of single messages. A simple solution would be to add a counter to all messages sent. If the recipient notes that the counter has skipped between two messages, then a message must have been missed and may be reported back to the sender, in a `NACK`-style response that exists in TCP. This proposed solution would require changes to the current Application Layer specification to amend both the message format (adding the counter) and the procedures on how to deal with the situation where a missed message is identified. A counter is more effective against delay of messages, although enforcing an acknowledgement for all messages would also be a suitable short-term solution.

Currently, the ERTMS specifications include a mechanism to cater for a situation should a

jamming attack take place, where an emergency stop is triggered to bring the train to a stop if no messages are received within a given timeout period. This timeout and the action to be taken if the timeout is reached are defined as part of nationally-set parameters for ERTMS. These parameters are `T_NVCONTACT` and `M_NVCONTACT` respectively. The possible actions are to trigger the normal 'service' brakes, trip the onboard systems including the immediate application of the emergency brakes, or take no action. The default, as set out in SUBSET-026 of the ERTMS standards is 'no reaction' with an infinite amount of time [118], i.e. there is no timeout for the safe reception of messages. This measure may result in problems within GSM-R blackspots, i.e if a train spends too much time in a blackspot (e.g. due to poor weather, causing the train to run at a lower speed) the brakes could be automatically triggered. Consequently, the standard provides methods to inform the train of GSM-R blackspots, meaning that the proposal of effectively using timeouts, and the appropriate actions to take, should not conflict with the existing standards.

4.6.3 Disagreement over RBC Identity and Safety Feature

One area that the EuroRadio specifications do not specifically cover is ensuring that when a EuroRadio session is started by a train, the RBC with which it establishes its communications is not only genuine, but is also the correct one with responsibility for the area in which the train is operating. When a train tries to set up a EuroRadio session, it is not always aware of the identity of the RBC with which it will be communicating. As previously identified, the specific case we consider here is a train that is not aware of its location where, at the 'start of mission' the train may invalidate the RBC identity and phone number. This may occur when a train is recovered following a breakdown or its state is lost following a system reboot. The specifications [73] allow the train to reuse the last RBC identity and phone number. It also allows the use of the EIRENE shortcodes to use location-based addressing to contact the most appropriate RBC for the area the train is connected to, via GSM-R. This assumes that

the attacker cannot compromise the network-level routing in the GSM-R backend which we consider to be an involved, high-risk and expensive task. Finally, the driver can enter the RBC number manually. The latter two considerations would allow the connection to a RBC which may not be situated in the area to which the train should be connected. Either through malicious means or human error there is, therefore, the possibility of an unsafe command being issued to the train.

When the train knows the identity of the RBC with which it is to communicate, there is no remediation offered in the specifications if the identity presented during the EuroRadio handshake differs from that which the train expects, nor whether this should be verified. To remove this possibility of ambiguity, we would recommend that the specifications consider this eventuality, where the RBC identity must be verified, if it is known and, if the validation fails, the connection should be terminated.

The Safety Feature (SaF) also presents a similar issue, where the attacker has the ability to influence its negotiation, given the first message is sent in plaintext and fundamentally underpins the MAC cipher to be used. The train selects the SaF in the first message and the RBC responds with confirmation of the SaF to be used. However, the standards do not consider the possibility that the train and RBC Safety Features mismatch. As highlighted earlier in this chapter, only one SaF is currently supported but we expect that, for future revisions and baseline standards, new SaFs may be supported and it is therefore essential to add in this verification. It should be enforced that the selected SaF is either equal to, or more secure than, what was sent by the initiator of the EuroRadio handshake. From here, two questions may be presented: (1) how do you rank MAC algorithms and (2) what if the train/RBC does not support the proposed SaF?

Naïvely, we could maintain a list of MAC algorithms in the standards documents, with Cyclic Redundancy Check (CRC) and ciphers known to be vulnerable and weak (e.g. DES and RC4) ranked lower, with AES-MAC and known, strong, ciphers ranked higher. It is expected

that the ERTMS standards provide a 'minimum' set of ciphers that are to be supported by all entities, with possible extensions by National Infrastructure Managers, who may introduce newer, more secure schemes. In this case, an alternative approach can be taken from TLS, where a list of supported ciphers is provided by the sender and the recipient simply chooses one from that list, where there must be at least one supported cipher shared between the train and RBC.

4.7 Chapter Summary

In this chapter, we have carried out a security analysis of the EuroRadio protocol and elements of the Application Layer protocol used in ERTMS. As part of this, we developed a novel way of representing the counter-style timestamps used in the standards and new correspondence assertions to test for message insertion, deletion, reordering and replay.

From this analysis, we find that EuroRadio defends the security of its key and authenticates the parties involved against an active Dolev-Yao attacker. It does, however, have weaknesses which should be addressed to improve its resilience to meet the security properties mentioned earlier, including message deletion and the insertion of emergency stop messages by an attacker. The relevance of these findings are discussed in the previous section of this chapter.

The model itself currently only considers the sending of exactly three messages which is sufficient to prove the security of the EuroRadio protocol. However, an extended model which captures the interleaving of normal-priority and high-priority messages could be created. This can be achieved by including the priority flag in the model with appropriate changes to assertions to capture the security requirements for any number of messages. Whilst this analysis found that EuroRadio and the Application Layer protocols do not offer any protection against the insertion of high-priority messages, we should note that the insertion of messages into a GSM-R data stream is currently technically difficult and alternative, more physical, attacks may have more impact in the short-term whilst this capability is developed.

One other issue presented by the ProVerif model is that it assumes that the cryptography used is perfect. However, as we will find in Chapter 5, this is not the case.

Chapter 5

Cryptographic Analysis of the ERTMS MAC

In this chapter, we build on the work from Chapter 4, in which we observed that one of the gaps in that analysis arose from ProVerif's assumption of 'perfect' cryptography, with no observable weaknesses or vulnerabilities. We, therefore, 'close' this gap by carrying out a cryptographic analysis of the MAC used in EuroRadio, a custom cipher based on an ISO standard [31]. From this, we identify a weakness which, whilst not currently exploitable, has the potential to be when used in high-bandwidth applications.

5.1 Motivation

As highlighted in Chapters 2 and 4, ERTMS is built up from a stack of three core protocols and layers: GSM-R, EuroRadio and an Application-Level protocol layer (see Figure 2.3), where each layer is relied upon to provide specific security and safety guarantees to the layer above.

The MAC algorithm used in EuroRadio itself is based on the ISO-9797 MAC Algorithm 3 [31, 140], a DES-based Cipher Block Chaining (CBC) cipher which is modified from the standard. As a result, any assurances and formal proofs, if any, that were conducted on the algorithm, as defined in the standard, may no longer hold and requires separate assurance. Of particular interest to us is the change at the end of the cipher from two DES encryption rounds

to three, i.e. the cipher itself uses a single DES transformation for all but the last block, where this final round has been modified. The justification for moving from using just 2 keys for the final round is to mitigate the possibility for weaknesses to be exploited. EuroRadio, therefore, uses 3 distinct keys in the final Triple DES (3DES) round instead.

Without this assurance, it is possible that the cipher could suffer from an unknown, or unaccounted for, weakness and an attacker would then be able to formulate new attacks which undermine the security of the protocols which rely on these ciphers. EuroRadio is the next line of defence after GSM-R, which already has existing weaknesses, namely the A5/1 encryption cipher, used to provide confidentiality for the data sent between the base station and endpoint [100]. EuroRadio has a significant responsibility in providing the secure and safe operation of ERTMS, as it guarantees the authenticity and integrity of messages to and from trains and RBCs. If an attacker leveraged a cryptographic weakness in the EuroRadio MAC, they would then be able to forge their own valid messages, which could have serious consequences.

5.2 Contributions

Here, we carry out a cryptographic analysis of the MAC cipher used in the EuroRadio protocol, building on previous work which has analysed the standardised version of the ISO-9797 MAC [105, 116, 115]. This work found a collision-based key recovery attack, requiring $2^{32.5}$ known plaintext-MAC pairs, coupled with $3 \cdot 2^{56}$ offline operations.

As EuroRadio uses three distinct keys instead of the two-key 3DES transformation in ISO-9797, these attacks should become less effective. In this chapter, we will consider the cryptography offered by each layer in the ERTMS stack (GSM-R, EuroRadio and the Application Layer protocol) and find an attack that allows an adversary to forge messages at the EuroRadio layer, leveraging several limitations and weaknesses within the EuroRadio and Application Layer protocols. This chaining of attacks shows how an attacker can use weaknesses in the entire stack to succeed.

We will finally assess what possible mitigations and remedial actions can be implemented, taking into account the original design decisions and the impact of these proposals.

5.3 ERTMS Train to Trackside Technical Overview

In this section, we will provide an intuition of each component, the cryptography and message formatting used in ERTMS, which we will then use in this chapter to formulate our attack. This intuition is more technical than that presented in Chapter 4, when considering the cryptography used.

5.3.1 GSM-R Communications Layer

The GSM-R communications protocol sits at the lowest layer of the ERTMS stack and is used to handle communications between the train and trackside infrastructure [72, 73]. It is important to note, however, that this communication medium is only used for railway operations (e.g train to RBC and train to signaller communications) and not for, say, reading balises.

GSM-R is based on the GSM mobile communications standard [72], but leverages different frequency ranges, based on national spectrum availability, typically consolidated to a common set of ranges across nations. More importantly, GSM-R has additional functionality included, for example, support for multi-party communications between drivers and signallers (group calls), emergency calling and priority-based pre-emption. Here, specific types of calls/messages are allocated priorities, where active calls/transmissions may be terminated if a higher priority call/message is received. A classic example of where pre-emption may be used is if a driver is currently engaged in a call, but another driver engages the emergency group call. As a result, all calls in the local area will be terminated as emergency calls carry the highest priority [122]. For ERTMS, however, all communications take place at the highest priority for the data modem calls and cannot be pre-empted.

Compared to GSM, which has a cell-based network layout, GSM-R uses an alternative cell

layout, where base stations are instead situated alongside, or near to, railway lines, with overlaps provided to ensure redundancy in the event of a cell failure. In Great Britain, a nationwide deployment of GSM-R was completed by the Infrastructure Manager, Network Rail, in 2014, as part of a programme to retire its outdated predecessor, Cab Secure Radio (CSR). GSM-R data support was rolled out in 2010 on the Cambrian Line in Wales as part of the deployment of ERTMS Level 2, the first deployment of its kind in Great Britain.

Whilst progressive, the cryptography used is the same as the current public GSM network, where both employ the use of the A5 suite of cryptographic ciphers, specifically A5/1, a stream cipher, based on Linear Feedback Shift Registers (LFSRs). Alternative ciphers are available, such as A5/2, an export version of the A5/1 cipher, and A5/3, a block cipher. At this time, it is understood that only A5/1 is deployed on the railways, although the infrastructure is capable of supporting all three ciphers. These ciphers are used to encrypt the communications that take place between 'mobile stations' (i.e. trains and handheld devices) and a GSM-R base station, providing confidentiality during transmission. Of particular interest is the fact that mobile handsets and devices are authenticated to join the network, though the base station never authenticates itself to the handset.

5.3.2 EuroRadio

The EuroRadio protocol layer sits between the GSM-R and Application Layer protocols where its main function is to provide authenticity and integrity verification for messages sent and received between the train and trackside infrastructure. It provides guarantees that messages sent and received are authentic, through the use of applying cryptographic Message Authentication Codes (MACs). EuroRadio relies on GSM-R to provide guarantees of confidentiality through encryption of communications between the train and base station, where EuroRadio has provisions for algorithm negotiation for the MAC during the handshake. It should be recognised, however, that only one MAC algorithm is supported.

The MAC message authentication algorithm used in EuroRadio is based on the ISO-9797-1 MAC Algorithm 3 (also known as ANSI X9.19 Optional Procedure 1) [4, 140, 31] and is used to compute the MAC for a given message using a combination of DES and Triple DES (3DES) ciphers, as shown in Figure 5.1. This algorithm works by dividing a message into n blocks of 64-bits each, padded appropriately, which are then fed as the input into a CBC circuit which uses DES encryption for the first $n-1$ blocks of the EuroRadio message payload, where the final block is subject to a 3DES transformation. The MAC is computed with a 40-bit prefix, formed of the message length and the ETCS ID of the recipient, followed by the message being sent and some optional padding to ensure it is a suitable length for MAC computation [140].

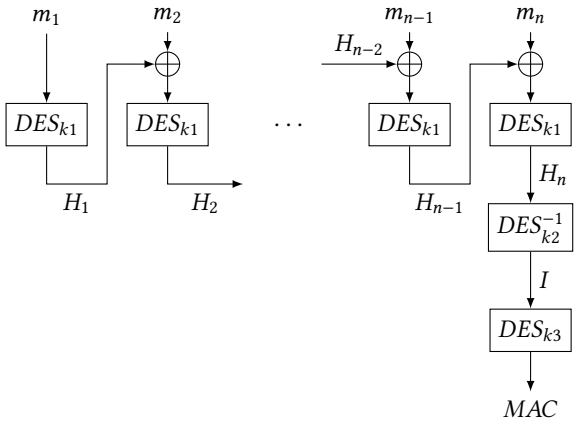


Figure 5.1: The EuroRadio MAC Algorithm. This cipher is made up of of $n-1$ DES rounds, followed by a final 3DES round.

In EuroRadio, the MAC algorithm uses three distinct DES keys for the final 3DES transformation, compared to that specified in the ISO standard, which uses only two keys for the same transformation. The padding method, *Padding Method 1*, is the same for both specifications and is used to ensure the message is the correct length before a MAC computation takes place, such that the data length is a multiple of the block size. Should the message length be equal to a multiple of the block size, no padding is added.

As we noted in Chapter 4, the EuroRadio handshake establishes session keys which are shared between the train and RBC. This key is used by EuroRadio's MAC algorithm to provide authenticity and integrity validation checks.

5.3.3 Application Layer Protocol

Once the train to trackside communication link has been established and both the train and RBC have mutually authenticated each other, data may now be exchanged via the Application Layer Protocol. This protocol and accompanying standards [58] are used to specify the messages sent between the train and trackside and how they are to be defined.

All messages at this layer contain a header, set appropriately dependent on whether the message was generated by the train or RBC, in addition to a timestamp (see Chapters 2 and 4 for a more detailed overview of how the timestamp is managed).

Typically, messages sent from the train to the RBC have the following format:

msg type	msg length	timestamp	train ID	pos. report	other (opt.)	padding (opt.)

A message sent from the RBC to the train will, instead, have this format:

msg type	msg length	timestamp	ack.	LRBG ID	other (opt.)	padding (opt.)

Common to both messages is that they contain their respective type, length and the relative timestamp. The difference between them is that the train will report its current position and, dependent on the message being sent, additional data, as prescribed in SUBSET-026 [58]. The position report sent by the train contains the distance and direction of travel relative to the Last Relevant Balise Group (LRBG), an upper and lower level based on the tolerance of the onboard systems and the location accuracy of the LRBG, with part of this data provided by the balise located between the rails. To ensure the full byte length and that the message length is a multiple of the block size for the EuroRadio MAC, optional padding of '`0`' may be added to the end of the message.

5.4 Attacking EuroRadio to Forge Valid Control Messages

In this section, we will review the roadmap of an attack against EuroRadio, leveraging known vulnerabilities against GSM's cryptography. Whilst the focus of this chapter is a cryptographic analysis of the EuroRadio MAC, in reality an attacker may need to additionally overcome the security provided by GSM-R. This could be achieved using Commercial off-the-shelf (COTS) equipment, for example, the $100 HackRF Software Defined Radio (SDR) and an NVIDIA GTX Titan X GPU for $1,400 to recover the A5/1 key.

5.4.1 Obtaining and Decrypting GSM-R Communications between Train and Trackside

Given that GSM-R uses the same cryptographic primitives and mechanisms as GSM to prevent message interception and malicious modification, any vulnerability that exists against the current GSM standard may also be exploitable within GSM-R. As an example, a jamming attack against GSM would also be effective in GSM-R, with the further complication that it would have the capability of affecting all ERTMS operations within its effective range.

A more sophisticated attack would be to directly target an individual train, with the potential prospect of the interception of communications to eavesdrop, or insert messages into the communications channel. By inserting messages, an attacker could, for example, instruct a train to stop for an indeterminate period of time or even coerce a train to enter an unsafe situation, such as allowing two trains to occupy the same block of track. In GSM, messages sent only provide authentication that the communicating party has possession of the A5 key for encryption and not that the message being conveyed is actually from a genuine, honest party. Again, as stated in Chapter 4, simpler, physical, attacks could be used to stop a train.

Existing work has largely assessed the security of the GSM encryption scheme. In 2015, it was shown that the A5/1 cipher used in both GSM and GSM-R could be broken in as little

as nine seconds using COTS hardware [100]. This work used a time-memory trade-off attack, with a probability of success of 81%. This statistic, however, does not reflect the two months preparation time and the reconnaissance work involved. Within this two month set-up period, lookup tables were produced. As these tables are reusable, once the set-up period has been completed, the attack can be repeatedly run in a short time period. An alternative form of attack is the use of rainbow tables, a pre-computed lookup table of plaintext/ciphertext pairs, used by Nohl et al. [131, 87, 85], although requiring some 1.6TB of storage capacity. Whilst this was previously considered to be a barrier, with the advances in hardware, storage capacity is now relatively inexpensive.

As previously mentioned, an export variant of the A5/1 cipher, A5/2 exists. This was also shown to be weak and, by having a reduced keylength, allows near real-time attacks, breaking the cipher in less than one second [10, 112]. Barkan et al. proposed a means to man-in-the-middle the GSM protocol, forcing it to fall back to the A5/2 cipher, reducing the overheads (e.g. financial and effort) and the time required to break the cipher. In the context of A5/1, this reduction may make the difference between the attack being successful or not. Once an attacker has established the A5 key, they can then start to decrypt the messages being sent over the link. In the case of GSM-R, this applies to ERTMS train control messages.

Breaking the A5 cryptographic keys is just one example of the various alternative methods of attack, such as using so-called 'IMSI Catchers' [48]. These work by tricking mobile devices into connecting to a dishonest GSM base station, masquerading as a genuine network. This is possible as the base station does not authenticate itself to the mobile device. For such an attack to succeed, the attacker needs to learn the Temporary Mobile Subscriber Identity (TMSI) of the victim train, in the '*Identification*' phase. In some countries, including Great Britain, running timetables and open data Application Programming Interfaces (APIs) exist to provide real-time train position data. However, the TMSI and some ERTMS values (for example the train ID) are not publicly available. In the second phase, '*Camping*', the attacker simply captures

traffic between the train and trackside infrastructure, where an attempt will be made to *recover* the A5/1 key. If successful, they then can control all traffic to and from the train and RBC. It is worth noting here that interception of the downlink (from base station to train) is well-documented, however, due to the timeslot-based nature of GSM, capturing the uplink (from train to base station) remains an open research question, as the adversary would have to 'latch' onto the particular timeslot a train is using.

5.4.2 Recovery of k_1 from the EuroRadio MAC

Given that the attacker has now recovered the A5/1 key, they can observe messages between the train and RBC and, furthermore, look for two unique messages which have the same EuroRadio MAC.

As the current EuroRadio MAC generation and verification scheme depends on the DES and 3DES encryption ciphers, the way that the DES cipher is used in the EuroRadio MAC generation makes it possible for collisions to occur due to the small block sizes (i.e. 64-bits) [116]. In a typical, well-designed scheme, we would not necessarily suggest that the existence of a collision implies a vulnerability. This, however, when coupled with 'interesting' functionality provided by the Application Layer protocol (as outlined in Section 5.6) and flaws in the MAC (due to the short keylength of DES) allows us to not only recover the key through brute-force attacks on k_1 (from Section 5.5) but also forge messages.

5.4.3 Forging a Malicious Movement Authority Message

In 3DES, the recovery of a single key does not normally compromise the rest of the scheme or provide a means for further attack. That said, as EuroRadio uses a combination of DES and 3DES, we can successfully use a single recovered key to produce a false, malicious, payload which has the same MAC as an observed genuine message, using just the first key, k_1.

Messages sent from the RBC to the train can include optional packets, outside of the 'required' components. In ERTMS, a free-text text message can be sent which is then displayed

to the driver, the conditions for the message to be shown being specified by the RBC. Using this message, we can use random bytes and take advantage of an unattainable condition to construct a valid, correctly-formatted Application Layer message which would be accepted and acted upon by the train. As the conditions for the message to be displayed to the driver would never be met, they would be unaware of any messages being received by the train.

5.5 Recovering EuroRadio Keys

In this section, we will review one method whereby a malicious actor could recover k_1, one of the three DES keys used by the EuroRadio MAC algorithm. As we have identified earlier in this chapter, this attack is possible by leveraging the small block size of the DES and 3DES ciphers, where the attacker simply has to wait for a collision to take place between MACs. Whilst recovery of the key is only one, potentially significant part of the attack, the final building block for the attack uses exploitable components from the Application Layer, which we will investigate as part of Section 5.6 using the key recovered in this section.

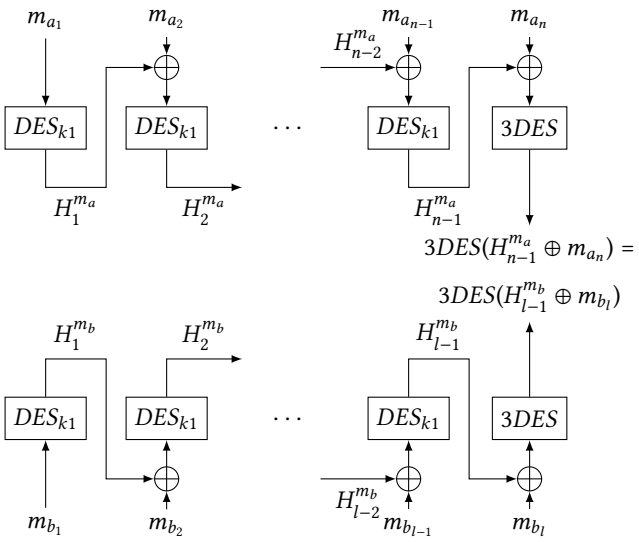


Figure 5.2: Recovery of the k_1 DES key through MAC collisions, where two unique, independent messages under the same EuroRadio session keys have the same MAC.

As identified from Figure 5.1, a single key is used for the single DES rounds which may

be bruteforced, making it possible for two, unique, messages to have the same corresponding MAC. As the final 3DES transformation is deterministic, it follows that the input to the 3DES block must have been identical for those two messages. Using these two limitations, we are able to recover k_1 as described further in this section. The EuroRadio MAC uses a CBC-like mode to compute messages from the Application Layer, using the following process:

A given message, m, generated by the Application Layer, is split into 64-bit blocks $m = (m_1, m_2, ..., m_n)$. If the final block, m_n, is not 64-bits long, it is padded with `0`s (defined by Padding Method 1 from ISO-9797 [31]) to allow it to be encrypted appropriately. For each block $m_i, i \in \{1, ..., n-1\}$, the MAC cipher block H_i is then computed:

$$H_i = DES_{k_1}(H_{i-1} \oplus m_i) \tag{5.1}$$

In the first block, m_1, $H_0 = 0$, where $Hi - 1 \oplus m_i$ is the result of an XOR operation on H_{i-1} and m_i. The final transformation, used for m_n, is the result of a 3DES encryption which produces our final MAC, that is $MAC(m)$:

$$MAC(m) = 3DES_{k_1,k_2,k_3}(H_{n-1} \oplus m_n) \tag{5.2}$$

For this attack to succeed, as previously highlighted, two unique messages are required which share the same MAC. The likelihood of observing at least one MAC collision occurring between two message/MAC pairs for N possible values is given below [49]. We set N to be 2^{64} for DES and 3DES. A diagram providing an intuition about this collision is given in Figure 5.2.

$$\begin{aligned} P &= 1 - \prod_{i=1}^{M-1} \left(1 - \frac{i}{N}\right) \\ &\approx 1 - e^{-M(M-1)/(2N)} \\ &\approx \left(1 - \frac{1}{N}\right)^{M(M-1)/2} \end{aligned} \tag{5.3}$$

For safety-critical systems, such as those employed in rail, it should be the case that a 1% probability of compromise is significant. Reversing Equation 5.3 where $P = 0.01$, the number of message/MAC pairs to be observed by an attacker to achieve a collision is $M = 6.1 \cdot 10^8$. Whilst this appears to be a high number of message/MAC pairs that need to be captured, we will rationalise the feasibility of such a data collection in Section 5.8.

For two unique messages, m_a and m_b, which share the same MAC (i.e a collision exists) and the number of blocks contained in each message respectively is n and l, we already know that the input to the 3DES transformation in the MAC algorithm (see Figure 5.2) must have been the same. Conversely, it follows that $H_{n-1}^{m_a}$ and $H_{l-1}^{m_b}$ are identical:

$$H_{n-1}^{m_a} \oplus m_{a_n} = H_{l-1}^{m_b} \oplus m_{b_l} \tag{5.4}$$

Using this property of the MAC algorithm, we are able to carry out a brute-force search for k_1, as both the plaintext messages m_a and m_b are known to the attacker. Using these messages, the attacker can compute potential candidates for $H_{n-1}^{m_a}$ and $H_{l-1}^{m_b}$, for all possible keys k_1'. When Equation 5.4 holds, we then have a possible candidate key k_1. An overview of the time and resources required by an attacker to compute these possible keys is given in Section 5.8.2.

To provide a concrete idea of how this attack works, let us consider an example with two, three-block messages.

Example. Let $n = l = 3$ for messages m_a and m_b. Using Equation 5.4, $H_2^{m_a}$ and $H_2^{m_b}$ can be computed. Expanding this to the underlying transformations, i.e. $DES_{k_1}(H_1^{m_a} \oplus m_{a_2}) \oplus m_{a_3} = DES_{k_1}(H_1^{m_b} \oplus m_{b_2}) \oplus m_{b_3}$, this results in the following final expansion:

$$\begin{aligned} &DES_{k_1}(DES_{k_1}(m_{a_1}) \oplus m_{a_2}) \oplus m_{a_3} \\ &= DES_{k_1}(DES_{k_1}(m_{b_1}) \oplus m_{b_2}) \oplus m_{b_3} \end{aligned} \tag{5.5}$$

As all blocks $m_{a_{1,\ldots,3}}$ and $m_{b_{1,\ldots,3}}$ are known to the attacker, the only unknown that remains is k_1. Using these blocks, however, we can use the brute-force guessing attack to recover this and forge our own messages with valid MACs.

5.6 Forging Valid, Malicious, Train Control Messages

Once an attacker has recovered k_1, they have only completed part of the 'puzzle'. They can attempt to use this to create a malicious message, but without the remaining two keys, k_2 and k_3, it is not possible to resolve the final 3DES transformation that takes place at the end of the MAC algorithm. As we have previously alluded, we require flaws that exist in all three layers of the ERTMS stack to allow this attack to succeed. So far, we have access to the communications channel via the cryptographic weakness of the A5/1 GSM encryption cipher and can now recover k_1 from the EuroRadio MAC. The remainder of this section will look at how we can manipulate application-level messages to produce messages which have identical MACs to one previously observed. Using this vector, we only require the recovered key, k_1.

Key to this section is, however, the assumption that we have observed two messages, $m_v = (m_{v_1}, \ldots, m_{v_n})$ and m'_v which have the same corresponding MAC, allowing us to recover the first key, k_1.

It is possible to create a malicious, forged, message, m_f if, and only if, we ensure that the valid message is not identical to the malicious message, i.e. $m_f = (m_{f_1}, \ldots, m_{f_l}) \neq m_v$ and, using Equation 5.2, the MAC for m_f, $MAC(m_f)$ is identical to the valid message MAC ($MAC(m_v)$) if the following holds:

$$H_{n-1}^{m_v} \oplus m_{v_n} = H_{l-1}^{m_f} \oplus m_{f_l} \tag{5.6}$$

As we know k_1, we are in a position to compute all of the H intermediate MAC values for each message for all blocks, with the exception of the last block (using Equation 5.1) and we

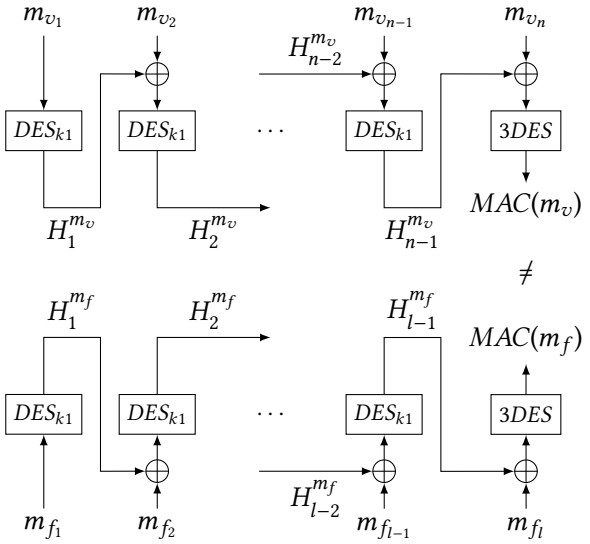


Figure 5.3: Ideally, for two given EuroRadio messages, m_v and m_f, the corresponding MACs for both should not match.

are then able to compute $H_{n-1}^{m_v} \oplus m_{v_n}$ and $H_{l-1}^{m_f} \oplus m_{f_l}$. One pertinent detail lies here, specifically that there is a very high likelihood that these two messages will not have an identical H intermediate value up to the penultimate block (see Figure 5.3).

For an attacker to be able to ensure that they can reliably produce valid MACs for their payloads, the message m_{f_l} can be extended with an additional block, $m_{f+1_{l+1}}$ (see Figure 5.4):

$$m_{f_{l+1}} = m_{v_n} \oplus H_{n-1}^{m_v} \oplus H_l^{m_f} \tag{5.7}$$

By adding this additional block, we can force the input of the 3DES transformation in the MAC computation to be the same for the new, forged message as the old, observed message. Even if we do not possess k_2 and k_3, we are still, therefore, able to ensure that the MAC for this message is the same as m_v. The additional block, however, has to be crafted by the attacker to contain some random data which, due to the flexibility of the Application Layer message standard, is unlikely to be verified. Using this approach, we can craft a message such that we can achieve a higher level of control over it and ensure the message conforms to the current

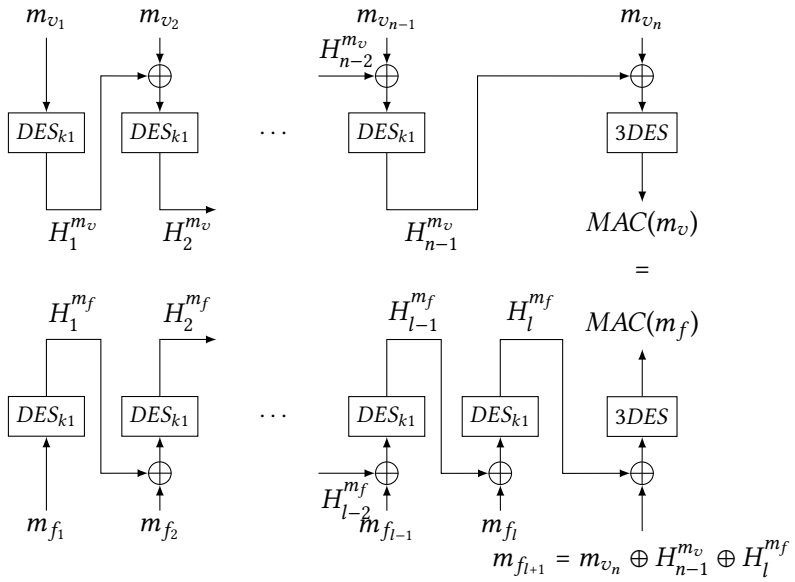


Figure 5.4: The forged message m_f can have the same MAC as m_v through the addition of the block $m_{f_{l+1}} = m_{v_n} \oplus H_{n-1}^{m_v} \oplus H_l^{m_f}$.

ERTMS standard.

In SUBSET-026 [144], the standard that governs application-level messages, there is an additional feature that allows optional packets to be sent in addition to those packets that are mandatory for a given message. A plaintext message can be sent in this packet, where the limit is 255 characters, intended to be displayed on the driver's console (Man Machine Interface (MMI)). As a first step, we would want the random data included in this block ($m_{f_{l+1}}$) to be the text message to be displayed within this packet. This packet, however, allows condition-based events, which determine if and when the message should be displayed to the driver. These conditions, however, can be specified such that the message, most likely, will never be displayed.

Core to this plaintext message is that there is a requirement for the data to be encoded to the ISO 8859-1 encoding standard, which includes ASCII characters [30]. It is not explicit within the standards whether the encoding is to be validated, or, if and when it is displayed to the driver at all. However, for the purposes of this chapter and to demonstrate that it is

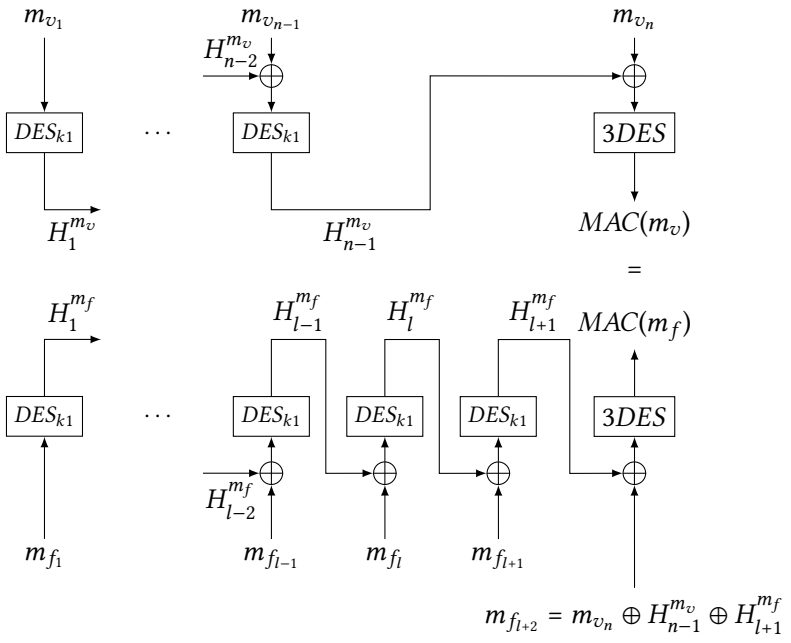


Figure 5.5: For the Plain Text Message to be correctly encoded, we add two blocks, $m_{f_{l+1}}$ and $m_{f_{l+2}}$, both of which we control.

the standard and not its implementation at fault, we will assume that the encoding is checked when a message is received. Now, all that remains is for us to use these building blocks to construct a valid, forged, message which includes this valid-encoded text message.

As previously mentioned, the attacker now has control over the last two full blocks as input to the MAC computation. In these blocks, we can therefore include a 16 character text message within the attack message (see Figure 5.5). The start of the message is denoted as before – $m_f = (m_{f_1}, \ldots, m_{f_l})$. We add two additional blocks, those that form the actual text messages, to be appended to blocks $m_{f_{l+1}}$ and $m_{f_{l+2}}$. These two blocks are sufficient to provide flexibility in the plaintext to ensure they conform to the ISO 8859-1 standard.

We first need to compute the input up to the 3DES transformation block for the original message ($H_{n-1}^{m_v} \oplus m_{v_n}$) and the intermediate MAC of the fixed part of the forged message ($H_l^{m_f}$) using our recovered key, k_1. We then proceed to randomly generate the first half of the plaintext text message in the correct encoding, placing it into block $m_{f_{l+1}}$. For the last block,

$m_{f_{l+2}}$, we compute the value such that m_f has the same MAC as the original message, using Equation 5.7, and $H_{l+1}^{m_f} = DES_{k1}(m_{f_{l+1}} \oplus H_l^{m_f})$:

$$m_{f_{l+2}} = m_{v_n} \oplus H_{n-1}^{m_v} \oplus H_{l+1}^{m_f}$$

It is important to verify that $m_{f_{l+2}}$ is a valid ISO 8859-1 encoded message. If it is not, then we start over again by generating another random first half of the text message $m_{f_{l+1}}$, and determine if this is a valid candidate for a correctly formatted block for $m_{f_{l+2}}$. Once $m_{f_{l+2}}$ is correctly encoded, we have our forged message. In Section 5.7, we provide an example of a valid forged Movement Authority message.

It is also important to evaluate the probability of finding a correctly encoded $m_{f_{l+2}}$. For this, we assume a uniform distribution, which should be the case, given the nature of the DES encryption used to compute $H_{l+1}^{m_f}$. ISO 8859-1 does not allow 65-byte values. We can then calculate the probability that a random string of 8 bytes is correctly encoded. As we assume DES to be a pseudo-random function, on average we need to carry out 10 attempts to find a correctly-encoded block, with 50 attempts providing a 99% success rate. Limiting ourselves to just ASCII-encoding, the probability drops to $\left(\frac{95}{256}\right)^8 \approx 0.04\%$, whilst requiring, on average, 2,780 attempts to find a correctly-encoded ASCII block.

5.7 End-to-End Process of Forging a Message

Following the work outlined in the previous section, we will now reaffirm this with a practical example, detailing such an attack against ERTMS.

One example message an attacker may wish to send to a train, which would have a debilitating effect, is the Movement Authority (MA) message, which is coupled with the Plain Text Message (PTM) packet, using the format given in Table 5.2. The Movement Authority in this message to the train changes a number of parameters, for example allowing the train to travel

for the next 328 kilometres at 200 km/h. The values in this crafted message are such that, through the inclusion of the optional variables, we can align the text message exactly with the last two message blocks used in the computation of the MAC. The MAC algorithm adds a 40-bit prefix and the message header (75-bits), supplemented by the MA packet (113-bits) and the Plain Text Message. The PTM without the actual message (92-bits) is 280-bits. This fixed component of the message is exactly contained within five message blocks, so the text to be displayed will commence at the start of the sixth block.

In order to generate an example of a MAC collision to use in this proof-of-concept, a program was developed in Java to generate a set of ERTMS messages. To ensure that this program matched real-world conditions, we can use the Acknowledgement Message (Message 146 in the standard). This is a message sent by the train to the RBC and is one of the most common messages, as the train will typically acknowledge most normal-priority messages sent to it (the RBC has to request an acknowledgement in the first instance). Not necessarily every message sent by the train, however, will result in the RBC carrying out some action. An example of such an Acknowledgement Message is given below in Table 5.1.

Variable	**Length (bits)**	**Description**	**Example**
Message 146 (Acknowledgement)			
NID_MESSAGE	8	Message type	146 (Acknowledgement)
L_MESSAGE	10	Length of message (bytes)	10
T_TRAIN	32	Train timestamp	53088208
NID_ENGINE	24	Train ID	1
T_TRAIN	32	Timestamp being acknowledged	53088178

Table 5.1: Example of an Acknowledgement Message that is sent by a train.

Within the acknowledgement message is the current timestamp, relative to the train's onboard clock and the timestamp of the message being acknowledged. As we recall from Chapter 4, there is no notion of sequence numbers in ERTMS messages, where a counter-style timestamp is all that encapsulates and ensures timely receipt of data. We could argue that, as

Variable	Length (bits)	Description	Example
Message 3 (Movement Authority)			
NID_MESSAGE	8	Message type	3 (Movement Authority)
L_MESSAGE	10	Length of message (bytes)	51
T_TRAIN	32	Train timestamp	1327095428
M_ACK	1	Acknowledgement required	0 (Acknowledgement not required)
NID_LRBG	24	ID of Last Relevant Balise Group	1
Packet 15 (Movement Authority)			
NID_PACKET	8	Packet ID	15 (Movement Authority)
Q_DIR	2	Direction	2 (Both directions)
L_PACKET	13	Length of packet (bits)	113
Q_SCALE	2	Scale used for definition of resolution	2 (10m scale)
V_EMA	7	Maximum speed	40 (200km/h)
T_EMA	10	Validity time	1023 (unlimited)
N_ITER	5	Number of iterations	0 (No iterations)
L_ENDSECTION	15	Length of section in MA	5000 (50000m)
Q_SECTION-TIMER	1	Section timeout Qualifier	0
Q_ENDTIMER	1	Timer for end section in MA qualifier	0 (No information)
Q_DANGERPOINT	1	Indicates whether a danger point exists or release speed is to be specified	0 (No information)
Q_OVERLAP	1	Indicates whether overlap exists or release speed is to be specified	1 (Overlap information to follow)
D_STARTOL	15	Distance from overlap timer start to end of MA	0
T_OL	10	Validity period for overlap	0
D_OL	15	Distance from the end of the MA to end of overlap	0
V_RELEASEOL	7	Release speed for overlap	126 (Use calculated onboard speed)
Packet 72 (Plain Text Message)			
NID_PACKET	8	Packet ID	72 (Plain Text Message)
Q_DIR	2	Direction	0 (Reverse)
L_PACKET	13	Length of packet (bits)	220
Q_SCALE	2	Scale used for definition of resolution	2 (10m)
Q_TEXTCLASS	2	Class of Message to be displayed	0 (Auxiliary)
Q_TEXTDISPLAY	1	Display message if one/all events fulfilled (start/end events relation)	0 (as soon as one event fulfilled)
D_TEXTDISPLAY	15	Distance at which text is displayed (start event)	32767 (327670m)
M_MODE-TEXTDISPLAY	4	Operating mode for text display (start event)	9 (System Failure)
M_LEVEL-TEXTDISPLAY	3	Operating level for text display (start event)	0 (Level 0)
L_TEXTDISPLAY	15	Length the text is to be displayed for (end event)	0 (0m)
T_TEXTDISPLAY	10	Time the text is to be displayed for (end event)	0 (0 seconds)
M_MODE-TEXTDISPLAY	4	Operating mode for text display (end event)	9 (System Failure)
M_LEVEL-TEXTDISPLAY	3	Operating level for text display (end event)	0 (Level 0)
Q_TEXTCONFIRM	2	Confirmation required	0 (Not required)
L_TEXT	8	Length of text message	16 (16 chars)
X_TEXT	variable	Contents of text message	...

Table 5.2: An example of a forged message which could be sent to a train. This message contains a Movement Authority and Plain Text Message packet, both containing forged values.

timestamps are limited to 32-bits in length, wrap-around might be an issue. However, this is an accepted characteristic of ERTMS and is therefore not considered a problem. Nevertheless, we should model this in the proof-of-concept to ensure this reflects real-world conditions.

In the proof-of-concept, we can compute the MAC value for a range of acknowledgement messages, using the same fixed set of keys, looking for collisions. The only values that are dynamic and subsequently parameterised, are the timestamps, where a fixed offset between the timestamp being acknowledged and 'current' time was introduced. To allow us to simulate reality, the offsets were set to intervals of 10, 20 and 30ms respectively. This would result in the simulated train having the ability to create and reply to a message received with a corresponding acknowledgement set by that offset limit. As previously mentioned, all other parameters (e.g. the train ID) remain static, to simulate one train acknowledging all messages sent to it.

Running the program, approximately 12.9 billion ($3 \cdot (2^{32} - 1)$) candidates were generated, which, under Equation 5.3, should provide a probability of collision of 99.9999%. Fewer candidate MACs could have been generated and, if a collision was not found, the program could simply be re-run. The program would simply output plaintext/MAC pairs, using the `uniq` and `sort` UNIX utilities to find collisions. Running the program continually over two days, 8 collisions were found (given below with their corresponding MAC) running on a system with an Intel® Xeon E5-2420 CPU, running at 2.2GHz. For example, the following two messages resulted in the same intermediate MAC (H_n), `80B7557F31566DBB` for k_1 = `01020407080B0D0E`:

`00120000020A9203A2105E0480000062105DFD0000000000`

`00120000020A9203AAE360078000006AE360000000000000`

Details of the additional collisions can be found in Appendix B.

To allow us to verify our candidate keys, a total of six DES encryptions (three for each

message to calculate the inputs to the 3DES transformation) are required. From the literature, highlighted in Chapter 3, the fastest DES cracker currently in known existence is the RIVYERA, cracking some 292 billion candidate keys per second [1], taking just under one day to recover a single DES key. The cost of such a system is, however, approximately $140,000.

These two factors are limiting to an attacker, where time is of the essence and cost can be prohibitive. Alternatively, an adversary could leverage high-performance cloud-based computing solutions, for example Amazon's EC2 or Microsoft's Azure Platforms. Within this research, the cost and speed of cracking DES keys was also evaluated, considering where it may be more attractive for an attacker to use scalable platforms which charge per hour, rather than incurring a large initial startup cost. Amazon offers an EC2 cloud computing instance which was, at the time of publication in 2017, fitted with 16 NVIDIA Tesla K80 GPUs (`p2.16xlarge`), with the state-of-the-art password cracker, `hashcat` [2] used to benchmark the number of DES operations per second. Today, this instance is now deprecated and has been replaced with the `p3.16xlarge`, which boasts 8 NVIDIA V100 GPUs, which include additional floating point units for computation. As a result, this may further reduce the time and cost required to break a single key.

Using cloud-based services allows almost infinite scalability and thus, scaling the number of instances used to crack the DES key, it is possible to brute-force k_1 within 30 minutes for a cost of $48,960. This provides the attacker with a trade-off between cost and time. At further expense, they can obtain the key in a smaller window, but if they do not need the key as urgently, they can use fewer instances. By using this method, we obtain a feasible window to recover the key, unlike that previously obtained using dedicated crackers like RIVYERA and the EFF's cracker, COPACOBANA. Further information on the full breakdown of the costs and benchmarking output using a `p2.16xlarge` instance can be found in Appendix C.

[1] `https://www.voltage.com/technology/rivyera-from-sciengines/`
[2] `https://hashcat.net`

Let us now focus on what this speed advantage means to an attacker. Using the collision found between two messages, a forged message with the same MAC can be constructed. A Python script was written which, given some k_1, will find a valid text message. This script provides flexibility in searching for valid text messages conforming to the ISO 8859-1 encoding, or purely ASCII text. Using the messages given in this section, we can create a valid text message, ASCII-encoded, on an Intel Core i7-4550U CPU in 0.209s, resulting in the below text message:

```
Z|1MB%<w*RRf)8n/
```

The corresponding ERTMS message that would be sent is:

```
030cd3c677a100000021f01c651ff809c4080000000007e4801
b90fffd2000000120105a7c314d42253c772a52526629386e2f
```

This message is constructed without the need to recover k_2 and k_3 and has the same intermediary MAC as the acknowledgement messages (`80B7557F31566DBB`), therefore providing the same final MAC. When received by the train, it would pass both the EuroRadio MAC validation and the Application Layer timestamp checks. The message above is broken down in Table 5.2.

There is one caveat to the attacker, which is to ensure that the attack 'holds'. When the attack is deployed, the train will be compelled to respond to the RBC with an acknowledgement. If the RBC receives an acknowledgement for a message it has not sent, it might trigger an emergency stop or changes made to the Movement Authority that would overrule the attacker's desired objective. An attacker would, therefore, need to block both the upstream and downstream GSM-R links to and from the train to ensure that the command is not detected by the RBC. Additionally, if the RBC detects, through the interlocking and track circuitry, two trains occupying the same section of track, the attacker would have to prevent the reception of an emergency stop or amended MA to the train. One side effect of such jamming is that it

would hamper GSM-R's voice capabilities as the same frequency ranges are shared – only the timeslots differ. This would prevent a driver contacting the signalling centre and vice versa, meaning that the train is essentially a 'ghost', unreachable by the defined processes stated in the Rulebooks. If an attacker had managed to compromise the RBC or GSM-R base station, a simpler course of action could be taken. This could be as simple as dropping those replies or any other messages, which are irrelevant, negating the effects of the attack.

5.8 Means of Data Capture for Attack

The attacker, at this point, can now find collisions for two given messages, recover k_1 and now craft their own attack message which has a corresponding, valid, MAC. Data capture is the only factor that is now pivotal to this attack, where they need to capture data sent to and from the train. How much data, however, is a consideration that provides a foundation for the criticality of this attack and the ease by which an attacker can perform it. Let us consider how much data needs to be captured and also the likelihood of such an attack being successful.

What we have established so far is that the attacker needs to capture train control data to find a collision. One possible way is that they target a specific, single, train and capture all the GSM-R data between the train and base station. An alternative, perhaps more ambitious means, would be to monitor the network infrastructure which links RBCs, base stations and Rail Operation Centres (ROCs) together. Capturing data here would be far more effective in providing a larger set of data to the attacker, increasing the likelihood of finding a collision, but this would be much more difficult to compromise. We will consider both types of attack vectors in detail and, to ensure we set the boundaries of what is an acceptable and reasonable probability threshold, let us consider a 1% likelihood of an attacker, having the ability to affect a train, as being unacceptable.

5.8.1 Monitoring of a Single Targeted Train

For this type of attack to be successful, an attacker would, most likely, need to be travelling on the train or have a remotely-connected device on the unit which can monitor the GSM-R data stream, following the appropriate GSM-R handovers between base stations. As we have already noted, the GSM protocol and its encryption ciphers have been shown to be weak and, given sufficient time for pre-computation [100], the attacker can easily decrypt the stream in real-time.

From Section 5.5, the chance of observing a collision at this threshold requires $6.1 \cdot 10^8$ messages. To provide some perspective, the average message length sent between a train and RBC is 32 bytes, therefore requiring 19.5GB of data to be captured. Assuming a train does not send and receive at the full possible bandwidth of GSM-R (14Kbps), but a modest 10Kbps, to obtain 19.5GB of data (the amount of 32 byte messages to exceed the threshold), the train would need to send messages constantly over a period of 23 days. Of particular interest is that the specifications do not explicitly mention key rotations or the limitations on key lifetimes if a train never leaves a single RBC zone. The only time the key is renegotiated is at startup or when communications take place with a new RBC. In reality, it is unlikely for a train to retain the same session key for such a long period for this type of attack to become feasible.

5.8.2 Monitoring Multiple Sessions

A more attractive option which the attacker could consider, would be to tap into the wired connections between the GSM-R base stations and the railway communications backbone, allowing them to monitor a potentially large number of trains at the same time. These cables and infrastructure are typically buried in the ground or carried overhead alongside the railway lines. This method of attack would, therefore, involve significant effort, risk and time, although it is technically possible. GSM-R base stations are, however, situated in open spaces, unlike the cell-based public networks, such that unauthorised access may not be detected as quickly,

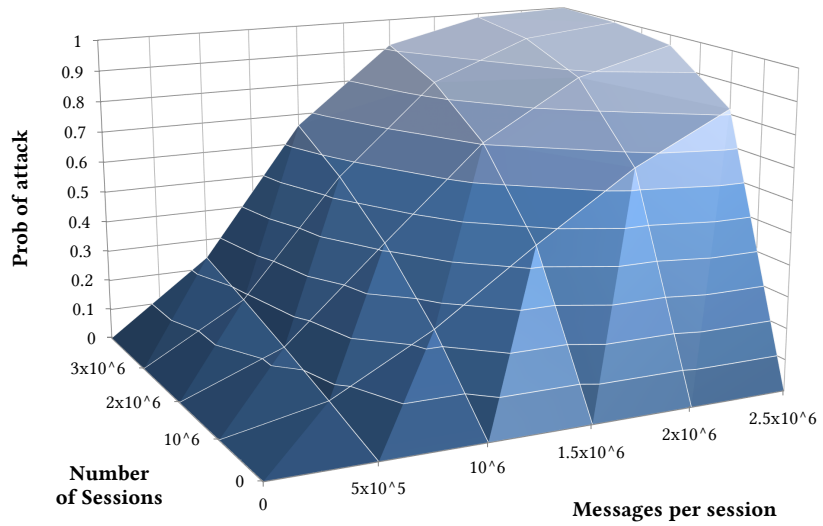


Figure 5.6: Graph showing the attack probabilities for the number of messages and sessions that are observed.

if at all. In comparison, an 'inside' attacker could achieve the same goal more easily by having direct access to the underlying infrastructure.

To establish the probability of finding one or more collisions in S sessions, we need to adapt Equation 5.3. Each session contains M messages, which gives us the probability of collision P as follows:

$$\begin{aligned} P &= 1 - \prod_{i=1}^{M-1} \left(1 - \frac{i}{N}\right)^S \\ &\approx 1 - e^{-M(M-1)\cdot S/(2N)} \end{aligned} \tag{5.8}$$

A range of the possible values is given in Figure 5.6. To provide a concrete intuition, let us consider Great Britain, which currently has approximately 4,000 trains in service each day [107], all of which communicate at a theoretical maximum bandwidth of 10Kbps, sessions which last for 10 hours and an average message size of 32 bytes. Using these values, we can

establish the number of messages per session and the number of sessions an attacker would need to monitor. Using Equation 5.8, for an attacker to have a 1% chance of finding a collision and having a successful attack, this would take approximately 45 days. However, to increase the probability of an observed collision to 50%, it would require 8 years. The attack, whilst theoretically possible, would be difficult, prohibitively expensive and time-consuming to carry out and succeed.

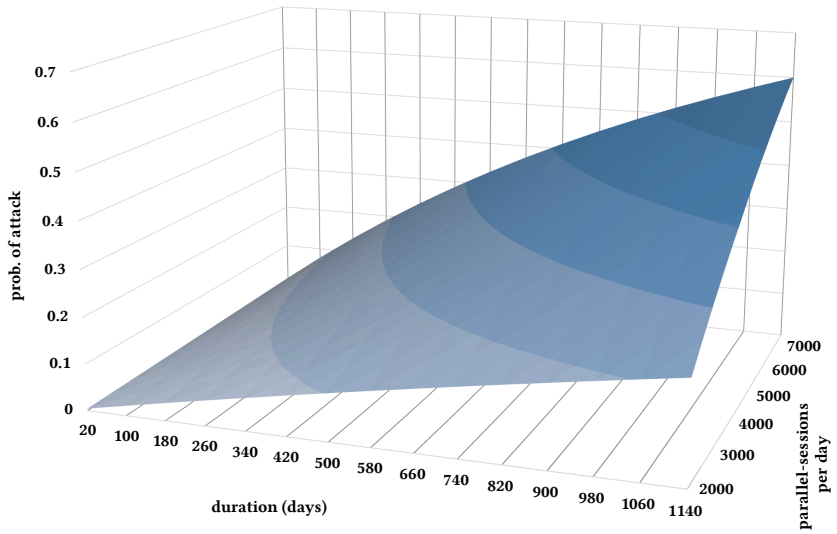


Figure 5.7: Attack probabilities over the number of days and sessions per day.

These figures are, however, only theoretical estimates of typical usage. High-speed trains, which traverse between RBC zones of control in a short space of time will have shorter sessions compared to local trains, which may not leave a RBC's control area, resulting in longer sessions. Data speeds also fluctuate based on network capacity and equipment, influencing the bandwidth speed. What is clear, however, is that with more trains being introduced into service, this landscape may change, resulting in increasing opportunities for an attacker. In Figures 5.7 and 5.8, we see how the likelihood of success for the attacker varies as the estimates and variables change. In Fig. 5.7, the potential of an attack being successful grows for an attacker who waits and is able to observe messages in a densely populated environment of

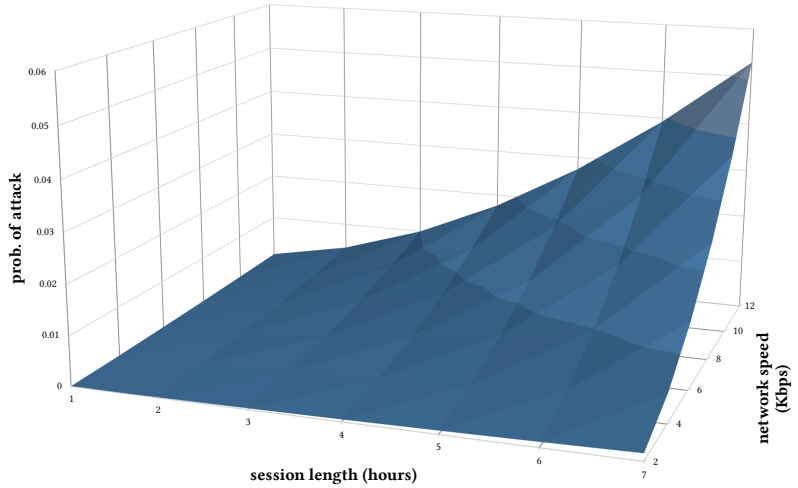


Figure 5.8: Probability of attack taking into account the session length and bandwidth of the connection.

trains. Fig. 5.8 assumes an attack space of 45 days and shows how the likelihood of an attack becoming successful changes with the session length and data speed. For short sessions and low speed, the cipher itself is safe to use. However, as the quantity of data sent and bandwidth increases, this rapidly becomes a problem. Defining the threshold at which safety becomes compromised will be reviewed in this chapter.

As discussed in Section 5.4.1, although capturing data sent over GSM is not a new concept, for completeness we should consider how, in reality, the attacker may deploy it. In real-world conditions, the HackRF Software Defined Radio (SDR) was used, with the `airprobe` and `gnuradio` (see Figure 5.9) Linux tools to capture traffic on the Cambrian Line, an ERTMS/ETCS Level 2-fitted regional line on the North Wales coastline (captures of which are shown in Figure 5.10). Other software-defined radios and alternative tools, for example OsmocomBB and BladeRF may also be used. Firstly, the operating frequency of the base station must be identified. The `kalibrate` tool scans for GSM-R base stations, where `airprobe` can then capture traffic on that frequency, decoding it into Control (broadcast) Channel traffic and encrypted

traffic. The capture files can be passed as input to `kraken`, which will recover the A5/1 key. `kraken` requires the Temporary Mobile Subscriber Identity (TMSI) of the entity, which is exposed in the 'Paging Request' (Figure 5.10) messages, which then allows full recovery of the

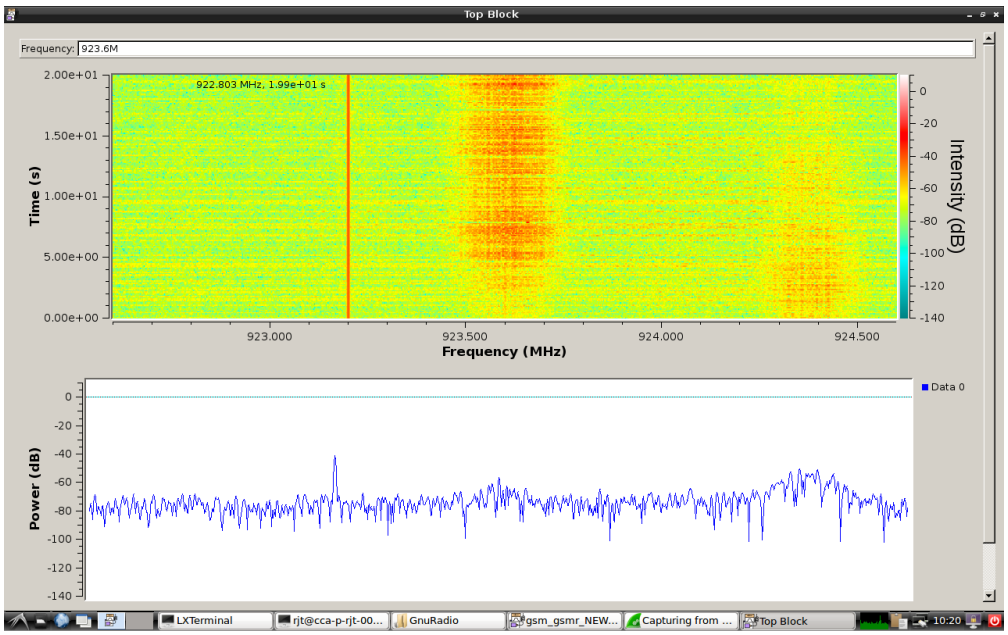


Figure 5.9: RF Waterfall Graph taken on the Cambrian Line, using `gnuradio`. Potential GSM-R data (red areas) frequency ranges are shown between 923.6MHz and 924.4MHz.

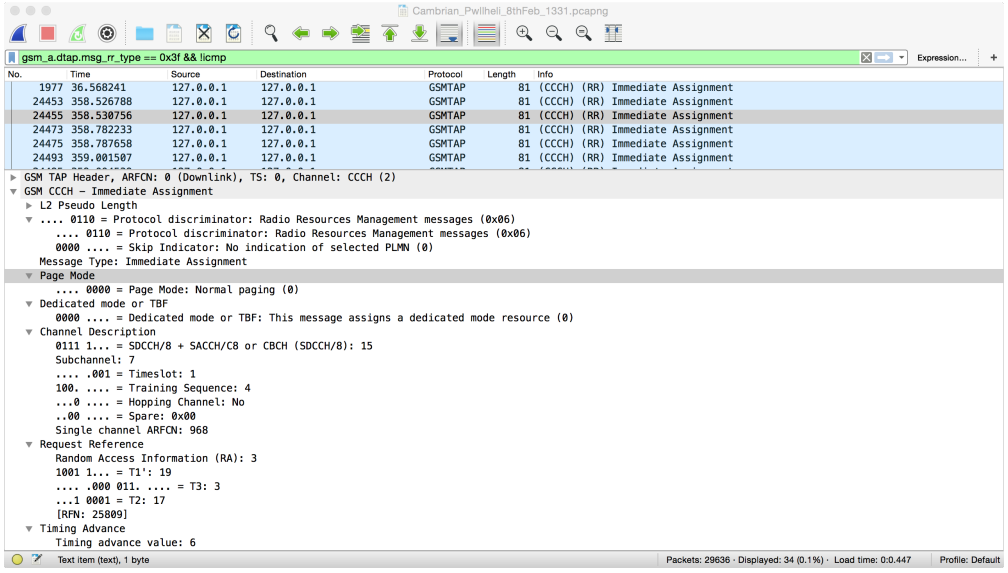


Figure 5.10: Wireshark Capture of GSM-R packets. Here, we can recover the TMSI value of the train and pass it as input to `kraken`.

key.

It is worth highlighting that this is just one method by which traffic can be captured. An alternative way would be to impersonate a base station and relay traffic to a genuine base station, which can be achieved through open-source GSM base station stacks, for example OpenBTS, OsmoBTS and Yate. A figure showing each ERTMS layer and the corresponding attack (as described in this chapter) is given in Figure 5.11.

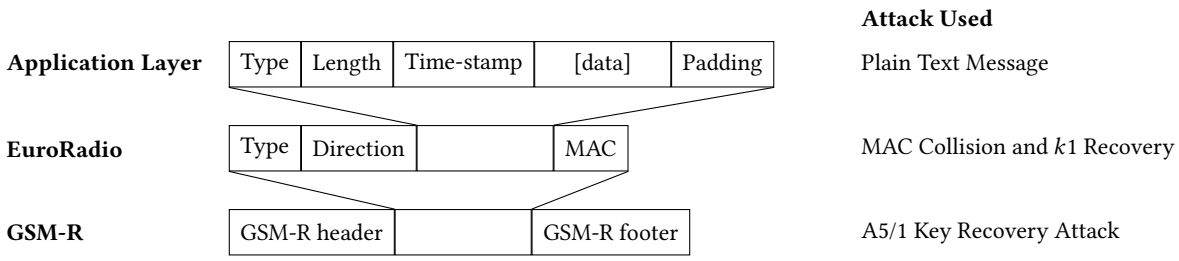


Figure 5.11: Overview of the ERTMS communication layers and corresponding attacks from this chapter.

5.9 Mitigations for the Future of EuroRadio

Whilst, based on the current assumptions that we have made in this chapter, the likelihood of an attack being successful is relatively small, we must still consider ways in which the EuroRadio MAC algorithm can be made more secure. Where DES was fairly new in 1997 and there was no alternative, fast, secure competitor when EuroRadio was first proposed [22], these arguments simply no longer hold now that hardware-accelerated ciphers, for example AES, are now widely available. In this section, we will explore the possible improvements that can be made to further secure ERTMS train to trackside communications for the future, whilst also protecting users who may deploy the EuroRadio MAC in alternative applications.

To modify a pre-existing standard, there is a cost in terms of expense, time and effort to the Infrastructure Manager, operators and vendors to implement these changes, which might present a barrier to such proposals being deployed. The following recommendations below,

therefore, are ranked in order of the highest compatibility against the current standards (i.e. little change is required and interoperability is maintained), with low implementation cost through to those solutions which require significant changes to the standard, with associated higher implementation costs:

- Restricting the use of the EuroRadio MAC from high-bandwidth applications.
- Forcing session keys to be rotated on a regular basis by limiting the lifetime of a session to ensure the probability of finding a collision is $P \leqslant 10^{-6}$.
- Expanding the Safety Feature (SaF) list of allowed algorithms to a set that are more secure.
- Reducing the EuroRadio and Application Layers into a single, combined, secure message sending/receiving protocol layer directly interfacing between GSM-R and the European Vital Computer (EVC).

As previously stated, EuroRadio is considered safe for the currently-defined ERTMS standards for command and control applications against an attacker targeting a single train. However, EuroRadio's MAC algorithm should not, in future, be considered for use in high-bandwidth applications, e.g. streaming data services used for remote condition monitoring, as the time required to obtain a collision (Figure 5.7) reduces, increasing the likelihood of session key recovery. Similarly, the algorithm should not be used in other applications as newer, faster and more secure alternatives exist, which we will discuss later. As these alternatives can be implemented at the design-stage for new systems, the cost is considered relatively insignificant and has a 'no change' mentality, whilst maintaining intercompatibility.

One other suggestion, as we observe in Figure 5.8, is that if a session period is sufficiently long, then the probability of a successful attack increases. For trains which constantly operate in the same area of control (e.g. commuter trains), sessions could be torn down on a regular basis, forcing the session key to be changed. As discussed in Section 5.5, assuming a collision probability $P = 1\%$, a total number of messages, M, would have to be very large – $M = 6.1 \cdot 10^8$.

This probability, however, is much higher than that typically used in safety analysis, $P = 10^{-6}$ [147], measured in the space of a year. Using Equation 5.8, the number of messages per session for some given fixed probability P can be expressed as:

$$M \approx \sqrt{\frac{2N}{S} \cdot ln\left(\frac{1}{1-P}\right)} \tag{5.9}$$

As an example, with $N = 2^{64}$ and $S = 1,825,000$ sessions and assuming an attacker monitors 5,000 sessions per day for an entire year, the number of messages per session which would result in a collision probability of $P = 10^{-6}$ is $M = 4,496$. Once this number of messages in a session has been reached (attainable via a counter mechanism as proposed in Chapter 4), the session should be terminated and the keys renegotiated. This requires a few minor changes to be made throughout the standards, where there is some flexibility for implementation on either the RBC or the train, as there is already provision for the RBC and train to terminate sessions, as required, for other purposes. The RBC managing session message sequence numbers is a better proposal, as there are fewer RBCs deployed, compared to trains, and this solution can be implemented and tested as a software update in a shorter timeframe, compared to updating trains in depots which would be a more substantial work programme.

Core to the attack presented in this chapter is the EuroRadio MAC, which relies on the security of the DES and 3DES ciphers. In the case of DES, it has previously been shown to be vulnerable, whilst 3DES currently does not have a full key recovery attack. Current estimates show that in the case of 3DES, it may be possible to brute-force keys and recover them by 2030 [1]. Longer term solutions which mitigate this attack, for example, changing the supported MAC algorithms as part of the Safety Feature negotiated during the EuroRadio handshake would require significant changes to the standards, software and, in some cases, reissuing keys where the keys stored are too short. Additionally, Infrastructure Managers cannot simply update the derivation key on every train and RBC, as it would be an 'all or nothing' approach

[1] `https://www.keylength.com/en/3`

at the same time, to support these changes. Economically and logistically, this would have a significant infrastructure deployment and implementation cost. As ERTMS is an evolving standard, new Safety Features should be phased in as part of new baseline standards which are then backported to the lower level implementations during ratification.

Alternative MAC schemes exist and, in this section, AES and HMAC-based MAC schemes are also considered, where efficiency and, more importantly, longevity are a consideration. Any proposed MAC changes should be quantum-resistant to prevent key recovery in a 'post-quantum' world. A study of possible alternative MAC schemes was also undertaken, using the same collision detection code, but with different MAC algorithms. Where the key size was too short for the proposed alternatives, the same prefix was used and the keys extended with distinctly different bits. The computations were timed to measure their relative performance against the current EuroRadio MAC, and the results given in Table 5.3. We include a 'patched' version of the EuroRadio MAC algorithm that has a 3DES transformation in the first block, which would prevent this collision attack.

MAC Algorithm	Avg. time (ns)	Time impact vs. EuroRadio MAC (%)	QCR
EuroRadio MAC (current)	10276.89	-	×
3DES Patch	13155.26	28% slower	×
AES-256-CBC MAC	8589.98	12% faster	✓
HMAC-SHA-256	4558.64	55% faster	✓

Table 5.3: Assessment of the performance impact of using different MAC Algorithms. These Algorithms are also assessed for Quantum Computer Resistance (QCR).

This table demonstrates that, using a DES-based cipher, has a significant effect on the time to generate a corresponding MAC compared to the suggested alternatives. It should be noted, however, that these theoretical performance improvements may not directly map to the real world in terms of functionality improvements. To give perspective to what these improve-

ments achieve in terms of time, the train's stopping distance is not significantly affected by changing the MAC algorithm. Under the current scheme, a train travelling at 200km/h would travel 0.05cm in the time taken to validate the MAC. The improvement in the time to compute a MAC, therefore, has a minimal effect in terms of the physical distance travelled.

Finally, if the opportunity arose that the ERTMS specifications allowed the EuroRadio and Application Layer protocols to be merged, each with its own set of individual defences, they may be combined into a unified layer providing the same guarantees as before, specifically authenticity, integrity and replay protection. This, though, is the most prohibitive change, requiring significant changes to the standards and subsequent impacts on the supply chain and ERTMS users.

5.10 Chapter Summary

In this chapter, we have reviewed and analysed the EuroRadio MAC algorithm, used to secure communications between trains and trackside systems by providing guarantees of the authenticity and integrity of messages. In the wider context of ERTMS, EuroRadio also has safety-critical responsibilities and any security flaws could have an impact on the safe operation of railways. By assessing each layer, it is possible to leverage weaknesses that exist within each layer to develop this novel key recovery attack and exploit cryptographic weaknesses that exist in EuroRadio. Combining this with a weakness that exists in the Application Layer protocol, the attacker could then reliably forge their own Movement Authority messages with corresponding valid MACs. These would be accepted by both the EuroRadio and Application Layers and accepted by the train, potentially placing it in an unsafe situation.

Finding an attack vector is only one concern, as all of the recommended solutions to minimise exposure require changes to be made, in varying degrees, to the standards. Some require minimal changes which leverage existing functionality in ERTMS, for example, forcing the renegotiation of session keys after a specified time period, whilst others involve signifi-

cant change, such as consolidating the EuroRadio and Application Layers into one, cohesive, security-service protocol. What has been made evident by the work presented in this chapter and Chapter 4 is the need to consider an evolving attacker model and their evolving capabilities. This will form a key part of influencing proposed improvements to the ERTMS standards as we will explore in Chapter 6.

Part II

Defining a Secure Architecture for ERTMS and Rail

Chapter 6

Key Management for the Future of ERTMS

This thesis has, thus far, shown that the ERTMS standards have remained largely unchanged in terms of security and not been updated to reflect the capability of today's adversaries. The absence of formal assurance of both protocols and cryptography highlights significant issues in a standard that is currently deployed internationally. In Chapter 4, we explored the application of cryptography in ERTMS to negotiate a secure session key and validate that each party was authentic whilst, in Chapter 5, weaknesses were found in the cryptography. What has not yet been explored, however, is how these cryptographic keys are deployed and maintained and consider what changes can be made to secure the ERTMS architecture.

Key management in ERTMS is entirely 'home-grown', specified in its own set of standards [142, 143], as explored in the previous two chapters, and has not been subjected to sufficient, detailed, analysis to provide assurances for the future. In this chapter we will, therefore, complete this analysis of key management in ERTMS, identify its issues and weaknesses and propose improvements which guarantee a secure architecture for the platform, especially given its anticipated lifespan.

6.1 Motivation

As we reviewed in previous chapters, the Application Layer provides little assurance of the authenticity of messages. Rather, its only purpose is to ensure messages are correctly formatted and received in a timely, ordered fashion. The EuroRadio layer provides these guarantees to validate the message sender and its integrity through a MAC, as explored in Chapter 5. How this MAC is keyed depends on the EuroRadio handshake protocol, as analysed in Chapter 4, which authenticates the train and RBC using a key only they share. It then uses that key and two exchanged nonces to derive a session key, which is then used to generate the MAC for messages sent between the train and RBC. This MAC, as demonstrated in this thesis, has been shown to be weak. The fact that the same base cipher (3DES) is also core to the ERTMS key management process is also of concern.

A number of issues exist within the current form of the key management standard. In particular, there is a significant responsibility imposed on the Infrastructure Managers to provision and maintain cryptographic keys both at home and across borders. To alleviate part of this burden, a proposed online version of the standard is under consideration but this does not consider the lifespan of ERTMS as a platform. The reliance on public-key cryptography in this online proposal, whilst considered secure now, will not be in the future when faced with a quantum-capable adversary.

Industrial Control System (ICS) environments are designed with the purpose of having a long lifespan as a core fundamental feature. Therefore, any solutions proposed for these systems should take into account issues such as post-quantum security, where quantum computers could render some of the current encryption schemes in use, for example, RSA public-key cryptography, insecure. Whilst there are efforts by The National Institute for Standards and Technology (NIST) and The European Telecommunications Standards Institute (ETSI) to standardise post-quantum public-key cryptography, the timescales we understand to deliver the

recommended standards and ratify them exceeds the intended start of Great Britain's national deployment of ERTMS. However, progressive deployments, for example, Crossrail in London and the East Coast Main Line are expected to be ERTMS-ready by 2021. Parts of the Thameslink capacity upgrade also included the deployment of ERTMS Level 2 with Automatic Train Operation (ATO) in 2017 where Network Rail, the UK Rail Infrastructure Manager, is also taking the opportunity to carry out major resignalling programmes by deploying ERTMS.

As key management is a core part of the secure architecture for ERTMS, if the management of keys is insecure, or is unable to cope with the future capabilities of adversaries in its lifespan, it is essential to consider solutions which could prevent future exploitation. These solutions would, however, for the purposes of integration by industry bodies, have a number of specific requirements placed upon them. Without meeting them, any proposals would have limited acceptance and there is the likelihood of a 'do-nothing' approach.

6.2 Contributions

In this chapter, we will review the existing ERTMS key management scheme and its proposed online variant, highlighting the potential issues and the lack of scalability offered by both standards. One fundamental contribution of the work covered in this chapter is a new proposed key management scheme, *TRAKS*, the Train and RBC Authenticated Key Scheme, which aims to reduce the complexity of managing cryptographic keys within both geographic domains and cross-border operations, reducing the risk of introducing potentially insecure practices.

From the outset, TRAKS has been formulated to enable the secure generation of message authentication keys between train and trackside. As its application is unified in ERTMS, it can also be applied to other environments, such as the authentication of balise payloads. Using dynamic key generation, Pseudo-random Functions (PRFs) and a shared secret would result in a significant reduction in the overhead of deploying cryptographic keys. Specifically, RBCs would be given a single, long-term, key for derivation, rather than a potentially large number

of keys for trains with which they *may* interact.

We have previously mentioned the motivation for this work and, in Chapter 1, we noted that the lifespan of ERTMS is significantly longer than most systems – in the order of 30-50 years. As a result, any critical elements which are part of the secure architecture of the platform need to either provide security for the future or allow flexibility, such that there is interchangeability of components to ensure future security. Backwards-compatibility is also a necessity to minimise the implementation overheads for operators who have currently deployed ERTMS. TRAKS uses symmetric cryptography, designed such that it is fully backwards-compatible with the current ERTMS scheme, allowing phased deployment. It also, through the use of post-quantum secure PRFs, introduces longevity to the scheme, where we demonstrate that TRAKS is at least as secure as the current standard. However, TRAKS provides the additional benefits of post-quantum security, scalability, extensibility and improved cross-border operation, all of which are limitations or omissions of the current ERTMS scheme.

In this chapter, we will provide an intuition into the current key management scheme, how it operates and its shortcomings and create a generic formalism to compare to TRAKS. Using this formalism, we will assess the security of both schemes, review the key management lifecycle and distribution and complete case studies of applications for TRAKS within ERTMS.

6.3 Background

Currently, the ERTMS standard [140] requires key management and provisioning to be carried out based on geographic *domains*, where each domain has its own Key Management Centre (KMC), responsible for the generation and management of cryptographic keys for all ERTMS entities in that domain. The KMC also defines the procedures and policies by which keys are installed on trains (also known as On-Board Units (OBUs)) and RBCs. Throughout this chapter, the terms *train* and *OBU* will be used interchangeably, so that we have a direct reference to the standards, such that comparisons between the ERTMS schemes and TRAKS may more easily

be drawn.

For a new OBU or RBC to be introduced to the network, the keys for these entities are generated by the KMCs following a key management request, typically from the vendor (e.g. Siemens, Bombardier or Alstom) and are then installed (in the clear) on the 'requesting' OBU/RBC through portable media (e.g. USB storage) [142]. This process is, however, inefficient and insecure. The use of portable media significantly increases the risk of compromise of the cryptographic keys, thus removing the need for the attacker to recover the first key used in the MAC algorithm to forge their own message and MAC as shown in Chapter 5. The transport keys (the keys installed on the train when 'born') are installed in plaintext and future keys, whilst encrypted, are not protected. Additionally, the storage devices on which the keys are stored may not offer protection against unauthorised access to that data. The provisioning and management of ERTMS keys is also inefficient, for example, updating a key on a RBC would require physical access by an engineer to install the key from the portable media.

In researching the real-world application of ERTMS key management, informal discussions with Rail Infrastructure Managers highlighted that, due to financial and logistical constraints, insecure strategies may become commonplace. Typical examples found were provisioning all (OBU, RBC) key pairs to each OBU and RBC (essentially removing the unique pairing between a train and RBC) or, alternatively, using the KMC to extend the lifetime of keys when they are due to expire, given the effort required to replace keys nationwide. In Great Britain, RSSB guidance suggests that keys may have their lifetimes infinitely extended [123], a potentially insecure practice. Cross-border operation of trains is also impacted, as the train keys need to be shared across geographic domains, managed by different KMCs. For these foreign KMC operators (the KMC operators who are outside of the 'home domain' in which a train is registered), there is a significant burden placed on them by the current version of the standard. For each new OBU to operate in their domain, they have to carry out the key provisioning process and also distribute them to the 'home' KMC for installation. What is subtle in the standards

[142] is that the foreign KMC operator will need to generate keys for that train and dispatch an engineer to install keys on every RBC with which the train may need to communicate. As a typical example, let us consider the Thalys PBKA analogy from Chapter 1. This train passes through France, Belgium, Germany and the Netherlands, thus requiring the involvement of four KMC operators, with four sets of key provisioning and distribution activities taking place for a single train to operate. If, say, the United Kingdom was to operate an increased number of trains through to mainland Europe, this burden would increase. The current scheme also impedes cross-border operation through the logistical burden of provisioning keys by the foreign KMC operator.

Additionally, the current scheme relies on the security of 3DES which, as we have considered in Chapter 5, is only recommended for use through to 2030 [1]. With the lifespan of ERTMS expected to go well beyond 2030 it should, therefore, be replaced with an alternative scheme which will be secure throughout its lifespan. We also note, in the case of the EuroBalise, that Cyclic Redundancy Check (CRC) encoding is used to validate the integrity, but not the authenticity of payloads, presented by the balise [139]. A unified solution which brings balises into the secure architecture would 'close the loop', given the level of trust that is placed on the balise.

ERTMS key management is defined in two standards, one offline [142] and a proposed online scheme [143] which forms part of the developing Baseline Level 3 standards. Introduced in the online variant is a Public Key Infrastructure (PKI), where certificates and keys are issued to trains and RBCs, which can then be used to communicate with the KMC. This places assumptions on GSM-R (for trains) that it has sufficient capacity and bandwidth to handle trains polling for new and updated keys and, to meet a requirement of the PKI variant, the Infrastructure Manager has rolled out GPRS support across the entire network. Importantly, support for GPRS over GSM-R is not currently widely available, nor is it a feasible solution.

[1] https://www.keylength.com/en/3/

This change would not only be disruptive to operations, but its management for cross-border operation would also place additional load on operators through the requirement for a regular polling and update cycle.

The only difference we can see between the offline and proposed schemes is that engineers would not need to interface with trains and RBCs after their initial commissioning, with the exception of out-of-band maintenance. As the key hierarchy remains the same, the proposed online solution, using RSA certificates, is one that would be insecure in the future, in a post-quantum world.

As discussed in depth in Chapter 3, we observed how a number of solutions have been proposed by both industry and academia (e.g. the Advanced Access Content System (AACS) scheme) but have limited applicability in the real world, placing functional requirements first and foremost. Security, consequently, takes a less important role to alleviate the strains on the Infrastructure Managers.

In this chapter, we will further consider the industry requirements and what would be necessary to include key management in the secure architecture for ERTMS and propose a new solution, TRAKS, for the future.

6.4 Overview of ERTMS and TRAKS Key Management and Generation

In this section, we will analyse how key management and generation takes place in ERTMS, introducing common terms and language which we will then use to compare the existing standards to TRAKS.

6.4.1 ERTMS Key Management

As discussed in Chapters 4 and 5, long-term keys are installed on RBCs and trains (OBUs) and are used to derive session keys through exchanged nonces during the EuroRadio handshake

protocol [140]. This negotiated session key is then used in all future messages between the OBU and RBC as a MAC key to ensure the authenticity and integrity of messages between the train and trackside. These unique long-term keys are allocated by the Infrastructure Manager for each OBU-RBC pair and are randomly generated by the KMC which then encrypts and MACs these keys (Figure 6.1). However, this process, as previously highlighted, is due to change under proposed modifications to the standard as part of ETCS Level 3. The main change is that a PKI is introduced, allowing the trains and RBCs to communicate directly to the KMC over TCP/IP. In the case of RBCs, this is achieved via a pre-existing fixed network, where trains are expected to use GPRS over their GSM-R datalinks to poll for changes.

ERTMS Entities	**Handshake**	**MAC**	**Encryption**
OBU ↔ RBC	K_{MAC}	KS_{MAC}	×
RBC ↔ RBC	K_{MAC}	KS_{MAC}	×
KMC ↔ RBC/OBU	×	kt_1	kt_2
KMC ↔ KMC	×	$\mathrm{K\text{-}KMC}_1$	$\mathrm{K\text{-}KMC}_2$

Figure 6.1: The current ERTMS Key Management Hierarchy [141], showing the specific keys (in the notation used in this chapter) that are used between ERTMS entities for specific applications, including EuroRadio Handshakes, Message Authentication and Encryption.

ERTMS Keys	**Current**	**TRAKS**
NID_C (Line) Secret	×	$knid\_c$
RBC Derivation Key	×	$km_{rid,null}$
Train Key	$\mathrm{K}_{MAC_{rid,oid}}$	$km_{rid,oid}$
Balise Secret	×	km
Balise NID_C Area Key	×	$km_{\mathtt{NID\_C},null}$
Balise MAC Key	×	$km_{\mathtt{NID\_C},bgid}$

Figure 6.2: The proposed Key Management Hierarchy under TRAKS. As an example, a train with OBU ID 7, communicating with a RBC with ID 3 would have the key $km_{rid3,oid7}$. Balise MAC keys are bound by the NID_C in which they are located and their unique balise group ID, *bgid*.

6.4.2 Issues with the ERTMS Schemes and Comparison with TRAKS

Both the existing offline and proposed online ERTMS schemes (including those implemented by National Infrastructure Managers outside of the current standard) carry significant operational overheads and present a security exposure given the potential to arbitrarily extend the lifespan and validity of the key. In TRAKS, backwards-compatibility is maintained, allowing the KMC to issue keys until such a time that the RBC can be updated to support dynamic key generation. This reduces the operational overheads on national KMC operators, enabling more efficient cross-border operations and interoperability. It also reduces the requirement of home and foreign Infrastructure Managers to deploy engineers into the field, visiting each and every RBC to install keys for each new OBU commissioned.

As established in the previous section, EuroBalises (EBs) do not have any means of authenticating their payload and can only guarantee their integrity, which are two distinctly different sets of guarantees. Balises are implicitly trusted to provide accurate location and data references but, without a trust anchor to guarantee this, it would be possible for an attacker to impersonate such a device. TRAKS, however, as described in this chapter, can be extended to include balises within its scope. Figure 6.2 highlights the key differences between the current ERTMS scheme and TRAKS.

Let us now consider some of the 'building blocks', requirements, definitions and security principles that will be applied and factor in the development of TRAKS:

Security Requirements. A review of the existing key management scheme in ERTMS shows that an implicit permissions model exists, where a EuroRadio session cannot be established if there is no cryptographic relationship between two entities, id and id'. This arises because no key exists ($km_{id,id'}$) between these two entities. Contextually, having no cryptographic relationship and requiring some form of approval to operate in a given area can be seen as a benefit, as it means that a train cannot carry out a EuroRadio handshake with a RBC with

which it is not authorised to communicate. As stated in Chapter 4, one attack EuroRadio cannot prevent is relay attacks, where the train's GSM-R connection is re-routed to a RBC located outside of the region. Without this permissions model (assuming for operational simplicity that all keys have been installed), the train would be able to complete the EuroRadio handshake and be given the command to move forward until it has reached the next balise and is able to provide its location. The effect of this could place the train in a potentially unsafe situation.

Informal Attacker Model. In the development of TRAKS, we investigate the impact of attackers who have capabilities including the insertion of keys, interception of keys between the KMC and train, and also the interception of keys between the KMC and RBC.

If an attacker was able to insert their own keys onto a train or RBC, a state of dissociation could be created between what the KMC asserts should be installed on the train and RBC as the set of installed keys, and what is actually installed on the train and RBC. Currently, it is not clear in the ERTMS standards how entities handle the event where an attacker might try to replace an existing key with their own, using the insertion commands.

Key interception is also a significant concern. Once 'transport keys', a pair of keys to authenticate and encrypt the OBU-RBC keys in transit from the KMC to the train and RBC, have been installed, the attacker is unable to recover and establish the $km_{id,id'}$ keys unless they know the appropriate encryption and authentication keys. Whilst this appears to be a sensible scheme, as the transport keys are transported and installed on the train and RBCs in plaintext, an attacker intercepting those transport keys during commissioning would be able to intercept all future keys successfully.

In order to assess the security of both the existing and proposed schemes, in addition to TRAKS, we must therefore consider an attacker from the Dolev-Yao model, one who operates in polynomial time and is able to observe all communications between the KMC and any train and RBC. Additionally, the attacker can also send arbitrary messages to any party, delaying,

modifying or deleting messages from being sent and received between parties. Any proposed solution for such a critical part of ERTMS should consider and address the capabilities of such an attacker.

ERTMS System Parameters. In the ERTMS System Specifications [56], identity variables, known as NID_C, are assigned to specific 'zones' of operation, which have varying scopes. These scopes can be as granular as a particular line in a country to the entire country itself. The European Rail Agency (ERA) centrally assigns the values for these variables from a pool of available values [56]. Within TRAKS, we can leverage these variables to define a permissions model of where a train can operate, rather than relying on the direct pairing relationship between the train and RBC that the current scheme employs, implicitly giving us the desired permissions model.

As we will explain in further detail later in this chapter, the Infrastructure Manager is responsible for generating a set of derived keys which are allocated to the RBCs that operate in a specific area, identified by a specific, unique, NID_C. What we must consider, however, is that it may take considerable time to phase in the full capability of dynamic generation and derivation of keys on a national basis for RBCs. To simplify this issue, the Infrastructure Manager could carry out the same functions that the RBC would, specifically computing the RBC key and then applying it to the set of train IDs that are authorised to operate in that given NID_C. For completeness, the RBC and train IDs are allocated by the Infrastructure Manager from an ERTMS identity database for that domain, hereafter known as *EDB*. Through the pre-computation of these keys, the Infrastructure Manager has a set of keys which are backwards-compatible with the existing ERTMS scheme, but are TRAKS-ready. In Section 6.4.3, we will consider a high-level hierarchy under TRAKS.

One of the benefits of using TRAKS is that, for National Infrastructure Managers, the time taken to provision the necessary keys for cross-border operation is significantly reduced, as the pre-requisite step to install keys onto all RBCs with which the train would interact would

no longer be required. This is because, in the first message of the EuroRadio handshake, the train announces its identity to the RBC, which it can then use to derive the appropriate key required to proceed. Through this, we enforce our permissions model through partitioning of the rail network. Keys are not, therefore, unnecessarily issued and can be managed in a more controlled manner, with operational zones precisely defined and supervised. One further flaw that exists in the current scheme arises when train fleets are moved across franchises and zones (e.g. the Class 172 *Bombardier Turbostar* units currently operated by Transport for London on the London Overground will soon move to the new West Midlands Railway Franchise) which would require a new provisioning run for each train, with an engineer visiting all RBCs and trains in the new area to install the new requisite keys. Under TRAKS, however, this is simply managed between the KMC and the train, without any requirement for interaction with the RBC, as it can simply derive the appropriate key to use. Additionally, this enables key management operations to be allocated to maintenance periods, for example when the train is stabled or in a depot.

Defining the Current ERTMS Key Distribution Scheme. The current method of key distribution in ERTMS is a manual process, where keys are generated by the KMC and then distributed, as required, in response to a valid request. These keys are then accompanied with a separate MAC and encrypted using pre-shared symmetric keys, known as *transport keys*. It is important to note, however, that during the initial commissioning of a train or RBC (i.e. where no transport keys are installed), these keys between the KMC, train and RBC are in cleartext. Should the transport media be compromised (whether it is a USB storage device, CD or smartcard) it would therefore allow an attacker to intercept and record the keys and all future keys issued by the KMC. The impact of this, as previously highlighted in this chapter, is that the attacker would no longer need to recover $k1$ using the attack presented in Chapter 5, as they have full control of all keys and can forge their own valid MACs for a specific train without having to wait for a MAC collision. Moreover, under the current scheme, if they target a RBC

during commissioning, they would have the ability to control a potentially large number of trains. To address this, by leveraging TRAKS, we can consider a new key distribution scheme which removes this attack vector. Trains and RBCs would, instead, be allowed to generate their own transport keys in such a way that an attacker would require knowledge of the keys in order to obtain any keying material. Furthermore, as we will see in Section 6.9, a new level of control is also introduced in TRAKS, where a shared key exists between the KMC and approved vendors, used only for vendor-KMC interactions. The expectation here is that trains and RBCs will have their own burnt-in key pair, installed by the vendor, which can be used for train/RBC-vendor commissioning.

Prior to comparing the current ERTMS scheme to TRAKS, we need to define a framework for ERTMS key generation, outlining the functions performed by the KMC. Using this outline definition, we can create a model of the current scheme (Algorithm 1) and TRAKS (Algorithm 2). Using these, we are able to prove the security of these mathematical models against an attacker, giving us confidence that the proposed solution in TRAKS does not compromise the security of ERTMS.

Definition 6.1 (ERTMS Generation Scheme) *An ERTMS key generation scheme* KMAC = (SGen, INIT.ID, GEN.KMAC) *with an ERTMS Identity database* EDB *is a tuple of three polynomial-time algorithms:*

$knid\_c \leftarrow \mathsf{SGen}(1^{\lambda})$: is a probabilistic key generation algorithm run by the KMC. It takes, as its input, a security parameter λ and outputs a random value, s, of length λ.

$ID_t \leftarrow \mathsf{INIT.ID}(EDB, t)$: is a deterministic algorithm run by the KMC to retrieve a set of ids from an ERTMS database. It takes, as its input, a database EDB and an ERTMS entity type t (e.g. train/RBC) and outputs the set of ids corresponding to t. In the current ERTMS Standard, $t \in \{OBU, RBC, EB\}$.

$km_{id,id'} \leftarrow \mathsf{GEN.KMAC}(id, id', knid\_c)$: is an algorithm run by the KMC to generate a MAC

derivation key. It takes, as its input, two ids $id \in ID_t$ and $id' \in ID_{t'}$, and a secret $knid\_c$, and outputs the key $km_{id,id'}$ which is used to authenticate communications between id and id'.

6.4.3 TRAKS Application Worked Example

Let us now consider a high-level hierarchy of ERTMS key management under TRAKS as shown in Figure 6.3. For key management operations that span across borders (e.g. to allow a train in the *GB* domain, managed by Network Rail, to operate in the *FR* domain, managed by SNCF Réseau), the operations required to provision keys remain unchanged, where a symmetric pair of keys, one for encryption and the other for authenticating payloads, is shared between KMCs. What is different, when comparing this to the existing ERTMS scheme, is that an intermediary level is introduced, leveraging `NID_C`, where keys at this level are denoted, using Figure 6.3, as $knid\_c_x$, where x is the identity value of the region (i.e. `NID_C`) for operations. As a reminder, `NID_C` is an allocated ERTMS variable that differs in granularity, either referencing a specific line, region, or entire country. For each `NID_C`, a unique secret key is allocated, mapping the `NID_C` to its given line secret.

In any given line or region, there may be more than one RBC – in Figure 6.3, as an example, the region represented as $\texttt{NID\_C}_1$ has only one RBC, but $\texttt{NID\_C}_2$ has two RBCs in its area of scope. Using the line secret, $knid\_c_x$ and the RBC identity, the RBC keys can be established. In this Figure, we have two trains, each of which are authorised to operate over two `NID_C` regions. Using Definition 6.1, we input the identities of the two trains and the KMC will allocate the keys required for those OBUs. For the RBC, however, only the single derivation function is required to establish the keys.

As a detailed example, let us consider the national domain defined by Figure 6.3. This domain has 4 `NID_C` regions and 6 RBCs. For each `NID_C` in this domain, which have allocated `NID_C` identities of 1, 2, 3 and 4 respectively, the KMC will generate a line secret (based on

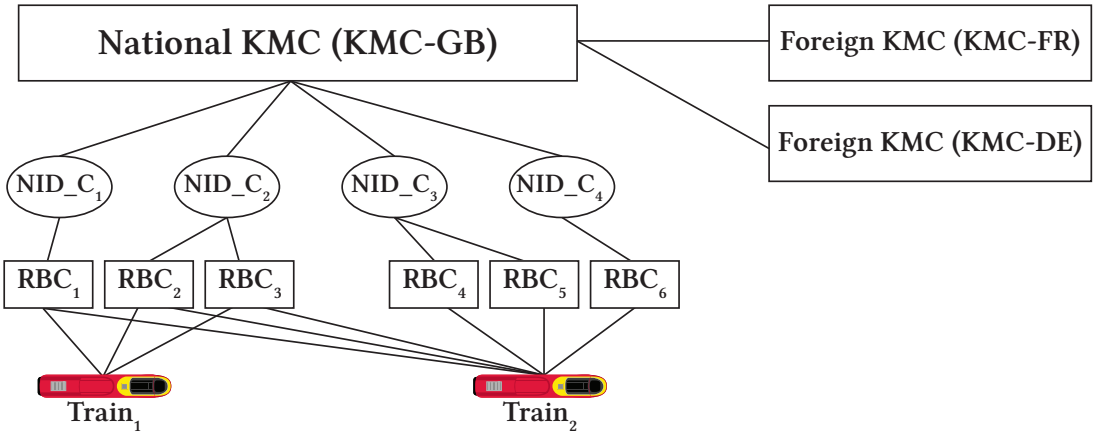


Figure 6.3: Proposed TRAKS Key Hierarchy for ERTMS. TRAKS is composed of four layers: (1) the National Key Management Centre for the 'home domain', responsible for liaising with foreign KMCs for cross-border operations. (2) Geographic regions within a country, identified as NID_C. (3) RBCs, responsible for sending command and control messages to trains. (4) Trains, which may operate across one or more NID_C regions.

a function which produces random output, for example, using a Hardware Security Module (HSM)), for each region. The KMC then stores the outputs of this function (i.e. the line secret) $knid\_c_1, \ldots, knid\_c_4$. For the RBCs in each region, the KMC will use the line secret for that given NID_C and compute the individual key for that RBC by combining the ID of the RBC with the line secret. The specific detail of how these keys are computed is given in Section 6.10.

For RBC_2 and RBC_3 as shown in Figure 6.3, the KMC will use the line secret for NID_C_2, that is $knidc_2$, and compute the individual RBC keys based on the identities of RBC_2 and RBC_3, $rid2$ and $rid3$ respectively, producing two keys, $km_{rid2,null}$ and $km_{rid3,null}$. These are the keys that will be installed on each RBC respectively and allow it to carry out the appropriate key derivations.

Where train keys are requested, the KMC will determine the specific RBCs for which a key is required and, for each of these RBCs, will derive the corresponding RBC key, outputting a key which is the result of applying the train ID to the RBC key using $\mathrm{GEN.KMAC}(rid, oid, knid\_c_x)$. The result of this process is a key which uniquely pairs a specific OBU to a RBC, for installation on the OBU. During the EuroRadio handshake, the first

message contains the train ID, which the RBC uses to compute the key it requires by taking its own RBC key and applying the train ID to it. As an example, for Train_1, with identity $oid1$, operating in `NID_C`s 1 and 2, the KMC will allocate all the necessary keys it requires to communicate with the RBCs in that area. Specifically, the train will have $km_{rid1,oid1}$, $km_{rid2,oid1}$ and $km_{rid3,oid1}$ installed.

For the rest of this chapter, we will formalise the definition of the TRAKS scheme and its suitability, highlighting its applicability and possible areas of implementation. In Section 6.5, we will formally define the current ERTMS key management scheme, before defining the TRAKS scheme in Section 6.6. Here, we will review the security requirements and attacker model that TRAKS must consider, before defining the key lifecycle. Using these formal definitions, we can prove the security of the current key management scheme compared to TRAKS.

6.5 Offline ERTMS Key Generation

In this section, we will review the current, offline, ERTMS key generation scheme in more detail. The key provisioning process during commissioning of systems begins with the vendor/system builder (e.g. Siemens, Alstom or Bombardier) of an ERTMS entity (e.g. OBU/RBC) making a formal request for an identity (ETCS ID) to be allocated from the central pool of values held by the ERA. This is followed by a request to the KMC for the appropriate keying material to be established.

The KMC, upon receipt of the request, will generate and issue two transport keys, kt_1 and kt_2, used for authentication and encryption respectively. These keys are then returned to the vendor for installation, in the clear, on portable media devices [142] on the intended entity.

When the transport keys, kt_1 and kt_2, have been successfully installed, the KMC will then proceed to generate unique keys for each train and RBC pair (as defined in the request from the vendor), identified by their identities oid and rid respectively. To ensure messages can be successfully authenticated within the EuroRadio protocol, each entity will receive a collection

of these keys. In the ERTMS standard, these are referred to as *KMAC* keys. The set of keys provisioned to both the OBUs and RBCs will be denoted as KM. KM, itself, can be initially empty (in the case of a new train or RBC being 'born'), but will have keys added later as the train is authorised to communicate with the relevant, appropriate, RBCs and vice-versa.

We can consider an algorithm that captures this generation process for the GEN.KMAC(id, id',knid_c) function from Definition 6.1, as shown in Algorithm 1. In this algorithm, two inputs are given, id and id', which correspond to the identities of the two entities authorised to communicate with one another. The output of this algorithm is $km_{id,id'}$, a randomly-generated 3DES key which is used for message and party authentication in the first steps of the EuroRadio protocol. As an example, for a train with ID oid and a RBC with ID rid, the key generated would be $km_{rid,oid} \leftarrow SGen(1^{\lambda})$.

Algorithm 1: Offline ERTMS Key Generation

Input: id, id'
Output: $km_{id,id'}$

```
1 function GEN.KMAC(id, id', null)
2     km_{id,id'} ← SGen(1^λ)
3     return km_{id,id'}
```

This algorithm shows that, under ideal conditions, as specified in the standard, for any given pair of identities, a completely random key should be generated. Let us now consider a scheme which is more dynamic and intelligent.

6.6 TRAKS - A Unified ERTMS Key Management Scheme

In the previous section, we observed how keys were statically generated and that, in larger quantities, this places a significant maintenance burden on the Infrastructure Manager. In this section, we will explore the TRAKS proposal, a more efficient key generation scheme for ERTMS. In this scheme, we can leverage the existing partitioning of the rail network into individual control zones, denoted by a national identifier. This national identifier is already

announced to the train by EuroBalises as part of the location report. Under this partitioning scheme, an explicit permissions model is enforced, ensuring that trains cannot start EuroRadio sessions and operate outside of their agreed areas of operation. TRAKS builds on the offline ERTMS scheme by allowing dynamic key generation for RBCs, removing the obstacles that impede cross-border interoperability and maintaining the notion of static key provisioning for trains. The benefit of TRAKS is maximum backwards compatibility with the current standards and key management practices, allowing a phased-in introduction.

In this section, we will also highlight how EuroBalises can effectively be introduced into this notion of a secure architecture, by extending the OBU-RBC authentication scheme to include OBU-EuroBalise authentication. TRAKS is therefore a universally applicable scheme for any rail entity.

6.6.1 Line Secret Generation

As highlighted in Figure 6.3, for a given country, it is possible for there to be one or more defined regions of operation. Each region is identified through a public value, $\texttt{NID\_C}_i$ for a region with NID_C value i. As a real-world example, the Cambrian Line in Wales (190km) has a NID_C of 1, whereas a country such as the United Arab Emirates (UAE) has a single NID_C, 882, covering all its 1,200km of track [56]. For each given $\texttt{NID\_C}_i$, the KMC will generate a random secret, knid_c_i. The KMC can, therefore, generate multiple secrets in order to partition the rail network into the defined geographic zones. Alternatively, it can also be used to establish trust between entity types, e.g. keys between OBUs and RBCs in the zone represented by $\texttt{NID\_C}_1$ would be generated using knid_c_1 and for zone $\texttt{NID\_C}_2$, knid_c_2 would be used for key generation.

The generation of the TRAKS shared line secret is similar to the current ERTMS key generation process, where

$$knid\_c_i \leftarrow \text{SGen}(1^\lambda)$$

and SGen(1^λ) is a pseudo-random number generator (PRNG) with the security parameter λ. However, unlike the offline ERTMS scheme, this secret key is never revealed outside of the KMC to the OBUs or RBCs. This secret is, instead, used together with the identities of the appropriate parties (e.g. RBCs and OBUs) to generate the message authentication keys. By taking this approach, we significantly enhance the usability of the scheme and reduce the overall management overhead (i.e. secret key material storage, distribution and disposal). In reality, the storage saving is in the order of megabytes whilst the effect of key accessibility has a more profound and significant benefit.

Let us now define, in detail, how this secret is used to generate the authentication keys for each ERTMS entity.

6.6.2 TRAKS Key Generation

In Algorithm 2, we observe how keys are generated under the TRAKS scheme, using *knid_c* and the identities of the communicating ERTMS entities.

Algorithm 2: TRAKS Key Generation

Input: id, id', s
Output: $km_{id,id'}$

1 **function** GEN.KMAC(id, id', s)
  /* computing keys using $s = knid\_c$ (line secret) */
2 **if** $id \neq null$ **then**
3 $km_{id,id'} \leftarrow PRF(id, s)$;
  /* computing train keys for a particular RBC */
4 **if** $id' \neq null$ **then**
5 $km_{id,id'} \leftarrow PRF(id', km_{id,id'})$;
  /* computing OBU-RBC keys using $s = km_{rid,null}$ */
6 **else if** $id = null$ **then**
7 $km_{id,id'} \leftarrow PRF(id', s)$;
8 **return** $km_{id,id'}$

This algorithm can be used to generate both static keys, which can be used to directly authenticate a set of messages between two entities, with identities id and id', and also generate

dynamic keys which can be combined by the respective entities with any given id' to derive a static key. We will now apply these concepts directly using the TRAKS scheme to show how keys can be generated for OBUs, RBCs and EBs.

OBU (Train) Key Generation. OBU keys are static and are generated by the KMC. Similar to the offline ERTMS scheme, the train, identified by its identity, oid, is authorised to operate on a specific line when provisioned with a set of keys $KM = \{km_{rid1,oid}, km_{rid2,oid}, \ldots, km_{ridn,oid}\}$, where $rid1, \ldots, ridn$ are the identities of the RBCs responsible for that area of control.

Each $km_{ridi,oid}$ key is computed using the GEN.KMAC($ridi, oid, knid\_c$) function, as detailed in Algorithm 2. In the first step of the key generation process, the line secret key, $knid\_c$ is combined with the RBC's identity, $ridi$ using a PRF which generates an intermediary pseudo-random value (as shown in line 3 of the algorithm). Using this intermediary key, we subsequently combine it with the OBU identity, oid, using the same PRF as given in line 5 of the algorithm, to produce the final $km_{ridi,oid}$ key.

The explicit PRF used has specifically been omitted from this recommendation as any secure, non-malleable function may be used, provided it has been proven secure against length-extension attacks. Example functions which meet this criteria include HMAC-SHA-256 and AES-CMAC, a view also shared by Ferguson et al. [63]. It should be noted that, up to this point, the TRAKS scheme has been shown to be fully backwards-compatible with the existing scheme, as key management onboard the train remains unchanged. We will now explore the extensions offered by TRAKS which significantly improve ERTMS key management.

RBC Key Generation. Under TRAKS, the way train keys are managed remains broadly similar to the existing scheme. However, for RBCs, their keys are dynamically provisioned compared to trains. This means that RBCs are able to produce the necessary keys for the EuroRadio handshake with any train, upon receipt of their OBU identity. This identity is provided to the RBC during the first EuroRadio handshake message, where the train is required

to announce its *oid* to the RBC. By taking this approach, TRAKS can seamlessly replace the existing ERTMS scheme with only minor software changes to the RBC to enable support for dynamic key generation.

As shown in Algorithm 2, we are also able to generate RBC keys. However, unlike a train, which is provisioned with a set of keys, the RBCs will only be provided with a single key, $km_{rid,null}$. This key is generated by the KMC via the GEN.KMAC($rid, null, knid\_c$) function, where rid is the identity of the specific RBC. This is the same key generated in line 3 of the algorithm which is used as an intermediary key for OBU keys. For any broadcast *oid* in the first EuroRadio message, the RBC can use the $km_{rid,null}$ key to generate the message authentication key $km_{rid,oid}$ through the GEN.KMAC($null, oid, km_{rid,null}$) function.

From this, we can observe that the generated $km_{rid,oid}$ key, based on the line secret key knid_c, can be used to enforce an *explicit* permissions model for operations. Any OBU and RBC capable of completing the EuroRadio handshake must have been explicitly approved to operate in a given NID_C. In comparison, the current scheme defines this implicitly (i.e. keys which uniquely pair a train to a RBC exist, but may be provisioned on a large scale), which may be violated with no known means of prevention (e.g. relaying).

EuroBalise Key Generation. As previously highlighted in this chapter, TRAKS improves the current key management scheme by adopting capabilities available today without modifications to generate these authentication keys. As a scheme, it offers flexibility, allowing authentication between other entities, such as EuroBalises. It is worth noting, however, that, whilst EuroBalises are given as an example in this section of the extended functionality through the application of TRAKS, this notion of universal key management can be extended to other, possibly not yet developed/ratified, entities.

Currently, EuroBalises offer no cryptographic means of protecting their payloads and guaranteeing authenticity and are implicitly trusted by the train for the accuracy and validity of this data [139]. One critical packet the EuroBalise can announce is the so-called 'Packet 44',

used to provide specific information about nationally-defined applications outside of ERTMS. As an example, in Great Britain, one active use for Packet 44 is the Tilting Authorisation and Speed Supervision (TASS) beacon [119], used on the West Coast Main Line (WCML) as a means of authority to allow trains to tilt on sections of track. It is therefore possible for an attacker to attempt to impersonate a real balise and change the parameters of the TASS authority as there is no cryptographic mechanism that would prevent this.

Under TRAKS, the process for generating the unique authentication keys for a balise is similar to that used to generate the keys for the OBU-RBC pairs. The KMC generates a balise-specific shared secret, $kbls \leftarrow \mathsf{SGen}(1^{\lambda})$, ensuring that the shared keys between EuroBalises and the OBUs are completely separate from the ones shared between OBUs and RBCs, used within EuroRadio. Furthermore, the keys are generated by each KMC responsible for the country in which the balises are located. As EuroBalises do not have any computational capabilities, they are unable to carry out any operations on their data, as their payloads are fixed. To overcome this restriction, the EuroBalise payload can be provisioned with their fixed messages concatenated with the corresponding MAC, which is then read and validated by the train. The MAC itself is computed using a static authentication key, generated through the $\mathsf{GEN.KMAC}(\mathtt{NID\_C}, bgid\mathrm{i}, kbls)$ function for every balise group, $bgid$i, located within some NID_C. A balise group is a collection of balises which are concentrated in a specific geographic area, typically in pairs, communicating the same 'telegram'. To simplify the deployment of EuroBalise MACs, balise group IDs are used instead of the individual balise identity. Alternatively, the balise group ID could be replaced with the balise ID. This is covered in more detail in Section 6.10. Once a key has been allocated for balises, the corresponding keys for NID_C, used to provision OBUs, can be computed as follows:

$$km_{\mathtt{NID\_C},null} = \mathsf{GEN.KMAC}(\mathtt{NID\_C}, null, kbls).$$

This allows trains to use $km_{\texttt{NID\_C},null}$ to generate the keys required corresponding to any balise group created using *kbls* and NID_C, by computing the specific balise group key as:

$$km_{\texttt{NID\_C},bgid} = \mathsf{GEN.KMAC}(null, bgid, km_{\texttt{NID\_C},null}).$$

6.7 Security Analysis of ERTMS Key Generation and TRAKS

One omission from the ERTMS guidance and documentation is a security analysis of the existing and proposed key management schemes. In this section, we will formally review and discuss the security of ERTMS key generation using a game-based approach. Such games are used to formalise the security of the cipher, demonstrating the probability in which an adversary can query a black box and determine which 'world' it is in. Cryptographic games are a strong requirement in assuring their security, although failing the game does not necessarily constitute a real attack.

In this game, as formally described in Definition 6.2, we will begin by allowing the attacker to have access to all identities which can be generated from EDB, the ERTMS Identity Database. The attacker wins the game if they are able to generate some valid key $km_{id,id'}$ for any pair (id, id') for ERTMS entity types t and t', where $id \in \mathsf{INIT.ID}(EDB, t)$ and $id' \in \mathsf{INIT.ID}(EDB, t')$. We do this to prevent an attacker from being able to incorrectly generate RBC-RBC and OBU-OBU keys which are not valid, constraining the domain to one that is realistic.

Definition 6.2 (Key Indistinguishably from Random) *Let* KMAC = (SGen, INIT.ID, GEN.KMAC) *be a scheme over a database EDB with security parameter λ, where t and t' are entity types with $t \neq t'$, and $b \in \{0, 1\}$. We consider $Exp^{b}_{\mathcal{A}}(\mathsf{KMAC})$ (see Figure 6.4), a probabilistic experiment played between an adversary $\mathcal{A}$ and a challenger C consisting of four steps:*

1. *Get IDs.* C runs INIT.ID for types t and t' to generate the sets ID and ID'.

$Exp^{b}_{\mathcal{A}}(\mathsf{KMAC})$

$ID \leftarrow \{i | i \in \mathsf{INIT.ID}(EDB, t)\}$
$ID' \leftarrow \{i | i \in \mathsf{INIT.ID}(EDB, t')\}$
$s \xleftarrow{R} \mathsf{SGen}(1^{\lambda})$
for $id \in ID, id' \in ID'$ **do** :
  if $(id, id') \neq (last(ID), last(ID'))$:
    $km_{id,id'} \leftarrow \mathsf{GEN.KMAC}(id, id', s)$
  endif
endfor
if $b = 0$:
  $km_{last(ID),last(ID')} \leftarrow \mathsf{GEN.KMAC}(last(ID), last(ID'), s)$
else :
  $km_{last(ID),last(ID')} \xleftarrow{R} \mathcal{K}$
endif
$b' \leftarrow \mathcal{A}((km_{id,id'})_{id \in ID, id' \in ID'}, ID, ID')$
return b'

Figure 6.4: Security Game for ERTMS Key Derivation with an adversary $\mathcal{A}$.

2. *Generate keys.* C generates a new random secret s and uses it to generate unique keys by running $\mathsf{GEN.KMAC}(id, id', s)$ for all pairs $(id, id') \in ID \times ID'$ except for $(last(ID), last(ID'))$. The function $last(X)$ returns the last element from a set X.
3. *Challenge.* If $b = 0$ then the last key is generated as
   $km_{last(ID),last(ID')} \leftarrow \mathsf{GEN.KMAC}(last(ID), last(ID'), s)$.
   If $b = 1$ then $km_{last(ID),last(ID')}$ is sampled randomly from the keyspace $\mathcal{K}$.
4. *Guess.* $\mathcal{A}$ is given access to all the identifiers in ID and ID' and to all the generated keys $km_{id,id'}$, where $(id, id') \in ID \times ID'$, and computes a guess $b' \in \{0, 1\}$. The output of the experiment is b'.

The advantage of the adversary $\mathcal{A}$ against the security of the keys generated in ERTMS is defined as:

$$Adv_{\mathcal{A}} = \left| \Pr\left[Exp^{0}_{\mathcal{A}}(\mathsf{KMAC}) = 1 \right] - \Pr\left[Exp^{1}_{\mathcal{A}}(\mathsf{KMAC}) = 1 \right] \right|$$

The key is indistinguishable from random if, and only if, the advantage is negligible in λ.

Let us now compare the current ERTMS scheme (KMAC_{ERTMS}) and TRAKS (KMAC_{TRAKS}).

6.7.1 Key Indistinguishability for KMAC_{ERTMS}

This chapter previously established how the current scheme (Algorithm 1) generates a random key for each OBU-RBC pair. Clearly, the advantage to the adversary is based on their ability to distinguish the output from the GEN.KMAC function from random. As the key $km_{id,id'}$ produced under the current scheme for any pair (id, id') is randomly sampled, the advantage of $\mathcal{A}$ to distinguish $km_{id,id'}$ from random is
$|\mathsf{Pr}[Exp^0_{\mathcal{A}}(\mathsf{KMAC}_{ERTMS}) = 1] - \mathsf{Pr}[Exp^1_{\mathcal{A}}(\mathsf{KMAC}_{ERTMS}) = 1]|$, which is negligible in λ.

6.7.2 Key Indistinguishability for KMAC_{TRAKS}

As we recall from Algorithm 2, there are three possible cases for the function GEN.KMAC to follow. Let us consider each case in detail:

Case 1. If GEN.KMAC is only provided with a single ERTMS entity identifier in the form of id, the resultant key, $km_{id,null}$ is generated using the secret s. With the assumption that the PRF used in this computation produces output that is indistinguishable from random, then the advantage for the adversary $\mathcal{A}$ to distinguish which 'world' they are in (i.e. $b = b'$) is negligible in λ.

Case 2. If two ERTMS entity identifiers id and id' are provided as inputs to the GEN.KMAC function, then the key $km_{id,id'}$ is generated as follows:

1. An intermediary key is generated by the PRF function with the secret s and input id. As shown in *Case 1*, this is indistinguishable from random.
2. The previously generated intermediary key is used as a secret in a second PRF computation that additionally takes, as its input, id'. If the PRF is secure, this input will then also be indistinguishable from random.

Case 3. If the identifier id' is provided with the secret $km_{id,null}$ as inputs to the GEN.KMAC function, then the key $km_{id,id'}$ is generated, as shown in *Case 2.2.* above, i.e. the value $km_{id,null}$

is used as an input to the function, together with id' in the PRF. As $km_{id,null}$ is indistinguishable from random (from *Case 1*), if the PRF used in this computation is also secure, then the output is also indistinguishable from random.

From these three cases, the advantage to the attacker $\mathcal{A}$ to distinguish any $km_{id,id'}$ key for the TRAKS scheme, i.e. $|\Pr[Exp^0_{\mathcal{A}}(\mathsf{KMAC}_{TRAKS}) = 1] - \Pr[Exp^1_{\mathcal{A}}(\mathsf{KMAC}_{TRAKS}) = 1]|$, will also be negligible in λ for *any* pair (id, id').

From this section, we have now proven the security of the current (offline) ERTMS scheme and compare it to TRAKS.

6.8 Managing the Key Lifecycle under TRAKS

Now that we have formally defined TRAKS and have demonstrated its security against an adversary, let us now consider its key lifecycle. Specifically, we will review how the keys will be used and the changes required to support its rollout on a larger scale. As part of this analysis, we will also consider additional aspects, including how keys can be revoked under TRAKS and what actions and processes should be followed when trains and RBCs have reached the end of their usable life.

Key Usage. Under TRAKS, the RBC no longer has a set of keys installed on it which, under the current ERTMS scheme, would have placed it in a vulnerable position. This improves its security posterity, as it can now derive its own symmetric keys based on public information broadcast by the train, the OBU's ETCS ID. As the keys are derived, only a single key has to be installed on the RBC, which it uses to derive the keys which correspond to those held by the train for communications. Moreover, this limits the need for less-secure practices to exist which have attempted to address the limitations of the existing scheme. Where the RBC does not have the capabilities of supporting dynamic key generation (i.e. it has not been updated, or the Infrastructure Manager has not enabled this functionality), it can still continue to operate using the same set of keys that would have been allocated to trains, bound to that RBC's

ID. The benefit here is that no changes are required to the EuroRadio protocol design. The only changes necessary would be in the way that keys are looked up. The RBC performs a lookup using its own ETCS ID and then finds a corresponding entry that pairs the RBC to the presented OBU ETCS ID and vice-versa, for the train to locate the key required for the EuroRadio handshake. If the lookup is successful and a key entry is found, then it will use that key. However, if no matching key is found, the EuroRadio protocol will trigger an error and fail as there is no key available for the RBC and OBU to proceed with the handshake.

As an example, under TRAKS, the RBC will use the $km_{rid,null}$ key to derive the key used to compute a symmetric session key for MAC authentication within the EuroRadio protocol. For the train, during commissioning, the KMC will carry out a lookup of the set of RBCs within the `NID_C`s specified in which the train is authorised to operate and will carry out the computation of the $km_{rid,oid}$ key for each RBC in that set. Similar to the current process [142], the train will simply receive a set of keys, which, upon the receipt of the first response message from the RBC in the EuroRadio handshake (the `AU2` Safety Protocol Data Unit (SaPDU)), will select the appropriate $km_{rid,oid}$ key to use.

Revocation of Keys. The existing standard [141] provides a set of messages that can be issued by the KMC to the OBU/RBC which essentially revokes keys from that entity, through the '`DELETE_KEY`' command. During maintenance and stabling, a blacklist, made available by the KMC for entities to download, would inform the train of RBCs with which it cannot engage and, similarly for a RBC, a set of trains that are no longer permitted in the RBC's area of control. In the event that there is a need to 'reintroduce' that *particular* ETCS ID into operation, there is no straightforward, comparable solution similar to the current ERTMS scheme, available under TRAKS that retains the existing ID other than to generate a new $knid\_c$, an intensive and disruptive process which would require all existing entities to be rekeyed. A more effective and simpler alternative to the existing scheme is, however, available under TRAKS which would allow the ETCS ID of the affected ERTMS entity to be changed. The validity of keys

(including their lifespan) would be governed under the validity of *knid_c*, ensuring that keys are refreshed on a regular, enforced basis. If we were to reintroduce a train or RBC into the network, a new ETCS ID should be allocated to that entity, as we would consider all keys allocated to the original ID to be potentially compromised and, thus, not eligible for reuse.

Disposal and End-of-Life. All keys for safety-critical applications should have a set, enforced, lifespan, ratified by national bodies and scoped for the appropriate risk 'budget' linked to that lifespan. At the end of a key's lifespan and validity, a 'DELETE_KEY' command should be issued to the entity to remove the key and a confirmation received by the KMC that the key has been deleted. During decommissioning, however, in order to protect the integrity of ERTMS by preventing cryptographic material being leaked, a 'DELETE_ALL_KEYS' command would be issued by the KMC, where the entity confirms receipt of the command, deleting all *km* keys, prior to deleting its *kt* transport keys. During this process, the KMC should retain an auditable copy of the keys, marked as 'invalid', to prevent any keys corresponding to that entity from being reissued. In the case that an entity operates across borders, the responsible Key Management Authorities would be requested by the 'home' operator to initiate their appropriate 'DELETE_ALL_KEYS' command. At this point, the entity in question may be disposed of safely.

6.8.1 Specific Changes and Discussion

So far in this section, we have demonstrated how TRAKS has been engineered to be backwards-compatible, such that Infrastructure Managers are able to immediately move to this scheme without any changes to the way that the EuroRadio standard operates. All changes can be performed within the software, with most changes being made at the KMC or RBC interfaces. Considering the strategic direction of the ERTMS key management standards, TRAKS is additionally able to operate in both offline and online situations.

In the offline scheme, the RBC requires a single initialisation and installation of its deriva-

tion and *kt* keys, compared to a visit by an engineer each time a train is commissioned or introduced to the network. Trains, however, at the point of manufacture, can have their keys installed at the same time as commissioning, whilst updates to the key database onboard can take place when the train is stabled or undergoing maintenance. If a train was 'moved' to a different operational region, for example the Class 323 units moving from the current Northern Franchise to the West Midlands, then the blacklist for the 'previous' RBC should be updated to contain that train unless it was going to continue operating in that region. Similarly, a 'DELETE_ALL_KEYS' command would be issued if the train is no longer operating in a given area, specifically to prevent the relay attack that we have explored in Chapters 4 and 5, as a train should not be allowed to make a journey for which it does not have the appropriate keying material *or* operational approval to proceed.

For online communications, when a RBC comes online for the first time and has carried out the appropriate TLS handshake and client authentication with the KMC, the KMC will only need to issue the necessary derivation key and appropriate constraints that apply to that key, as set out in the SUBSET-038 Standard [141]. For a new train, however, the process is simply an online version of the offline scheme. However, instead of physical media, an online connection is used. This solution provides the additional protection that, in the event that the KMC goes offline, the entities will retain sufficient keying material that would allow them to complete the EuroRadio handshake if, and only if, they were initially approved to operate by the KMC in that geographic area.

We also need to consider the case of existing, operational trains. Under the existing, offline, ERTMS scheme, each train has a unique key shared with the appropriate RBC for 'approved routes'. Operationally, this is a challenging concept. During the process of a national deployment under TRAKS, the line secret needs to be established and the appropriate derivation keys generated and distributed. This means that when RBCs are introduced to the network, no additional keys are required to be installed for the RBC to communicate with a new train

and, likewise, a train can immediately start communicating with any RBC. However, once a national deployment is complete, the process of adding additional keys is both costly and logistically complex. This can be overcome with the KMC having the ability to 'pre-compute' the keys required for the train in preparation for a switchover, where the RBC would now be expected to carry out the appropriate derivations.

For existing lines and deployed entities, for example the Cambrian Line in Wales, which pre-dates Network Rail's national deployment of ERTMS, the keys in use can remain valid until the validity period expires, when a replacement key would be issued based on the TRAKS specification.

In summary, the identified changes to the EuroRadio protocol (in software) and underlying key management infrastructure proposed in TRAKS are:

- Modifications to the way that a RBC looks up keys: the RBC will not perform a lookup, rather, it will carry out the appropriate key generation computation (from Algorithm 2) to derive the key that will allow it to communicate with the train. Trains will remain statically-keyed, with no internal changes required.
- Establishment of 'line secrets' that correspond to a particular `NID_C`, and appropriately generating $km_{rid,oid}$ keys for RBCs and trains based on the TRAKS principle.

6.9 Key Distribution under TRAKS

As previously highlighted, in the current offline ERTMS scheme for distributing keys to trains and RBCs, the transport key is issued by the KMC in the clear, with the requirement on the transport medium to be 'trusted', ranging from USB storage devices to CDs [142]. During a large-scale provisioning of keys, for example, a national deployment of ERTMS, there will be a significant number of transport keys being issued, with the risk that multiple transport keys could be stored on the same media as other transport keys [1]. Given that the transport keys are

[1] In SUBSET-114, the RBC and OBU will select a specific file from the media for its keying payloads, typically bound using their ETCS ID [142].

stored on the media in the clear, there is a potential that an attacker could simply intercept the transport keys during installation on the entities.

If the attacker was able to intercept the transport keys, they would have access to all future (encrypted) keys that are issued by the KMC when they are installed on the train or RBC. When considering a train, this would mean that the attacker could establish the session key, with relative ease, by observing the EuroRadio handshake between the train and RBC. For a RBC, however, an attacker could 'spoof' arbitrary EuroRadio sessions with valid MACs. The impact is that it would create a state of confusion for the RBC or, even worse, send arbitrary messages to a train with a valid MAC, as the attacker has access to the nonces exchanged. Consequently, the MAC would be valid and accepted. Given this threat, the current scheme is, therefore, insecure. In the following section, we will consider a new way of distributing keys for ERTMS as a set of steps.

6.9.1 Point of Manufacture/Commissioning

Instead of requiring the KMC to generate the transport keys for trains and RBCs, the train/RBC will create its own pair of transport keys kt_1 and kt_2, using a hardware security module and exported using a pair of symmetric keys, kv_1 and kv_2, burnt into the module by the vendor or operator to encrypt and MAC the payload containing the transport keys and the ERTMS identity of the train/RBC. Here, a requirement is placed on the vendor that they have an attestable secure supply chain and the keys are securely generated and handled in transit. If the vendor keys were to become compromised, it would be the responsibility of the vendor to provide a mechanism to update all trains/RBCs built by them with new, trusted keys.

6.9.2 Key Requisition from National Infrastructure Managers

Upon receipt of the transport keys, the vendor/operator will generate a request to the Key Management Centre Operator (typically the Infrastructure Manager) for the appropriate keys to be provisioned and installed. The transport keys generated by the train or RBC would be

provided as part of this request. The request itself is encrypted and also authenticated using a MAC with a symmetric pair of keys issued by the KMC to 'approved' vendors and operators, k_{v,kmc_1} and k_{v,kmc_2} (similar to the existing scheme shown in Figure 6.1). Given that a vendor may be responsible for manufacturing trains and building systems that are used in different countries, it is their responsibility to determine the appropriate countries for which keys are required (based on the operator's needs) and submit the keys to those identified Infrastructure Managers.

6.9.3 KMC Processing of Key Management Request

When a request for keys has been received by the KMC from the vendor, it will verify the nature of the request, based on nationally-agreed procedures. If valid, it will generate the necessary $km_{rid,oid}$ keys based on the request, encrypted and accompanied by a MAC, keyed using the transport keys provided, and stored as part of the request. The KMC would then retain the transport keys for all future communications if the request was from the commissioning of an entity. This part of the scheme accounts for two possible cases. One, where keys can be dynamically generated (e.g. for RBCs) and another, where keys are statically used.

- **Ability to Dynamically Generate Keys.** The Key Management Centre computes the 'RBC derivation key', i.e. $km_{rid,null}$ from Algorithm 2, with no identity parameter provided. It will then encrypt and MAC this key using the transport keys, before issuing the resultant payload to the vendor/operator.
- **Static Key Usage.** The Key Management Centre will identify the keys required, generating the set of keys based on the `NID_C`s provided in the request. This is achieved through repeated runs of Algorithm 2 for each RBC identified in that given `NID_C` with the appropriate $km_{rid,null}$ key. These keys are encrypted and accompanied by a MAC over the payload using the transport keys held by the KMC, before being issued to the vendor/operator.

6.9.4 Key Installation

The vendor/operator installs the payload of *km* keys received from the KMC directly onto the train/RBC, which will then verify the MAC and encrypt the keys before installing/performing the key maintenance instruction as directed by the KMC. Within this process, the train/RBC will generate a response to the KMC confirming that the key management action has been completed (i.e. the keys are installed '`KEYS_INSTALLED`'). Additionally, this includes a hash of the key database installed on that entity, which is encrypted and then accompanied with a MAC using the transport keys, which are then provided to the KMC. We add a hash of the entity's database to ensure that its expected state matches that which the KMC expects, to ensure that no keys have been omitted/included which were not part of the request.

6.9.5 Vendor Commissioning Confirmation

The response generated by the train/RBC is provided to the KMC via the vendor/operator who will verify the payload using its own transport keys held for that ETCS entity and compares the local key database hash corresponding to that entity. If there is a mismatch between the hashes, the protocol will require the issuance of keys to be carried out again.

An example issuance protocol is given in Figure 6.5 for a new train, highlighting the keys in circulation, how they are used and the responsibilities of each entity.

6.9.6 Considerations for the TRAKS Distribution Scheme

Under TRAKS, the process of revoking keys is considered to be an extremely rare event, compared to the regular revocation of public keys as we see for online website certificates. In the event that a train key was compromised, the Infrastructure Manager would be required to allocate a new ETCS ID to the specific train and provision new keys. A similar process would also be followed for the RBC. The rare intervention by an engineer at this stage with the affected entity is considered to be an acceptable overhead. One further issue with the current

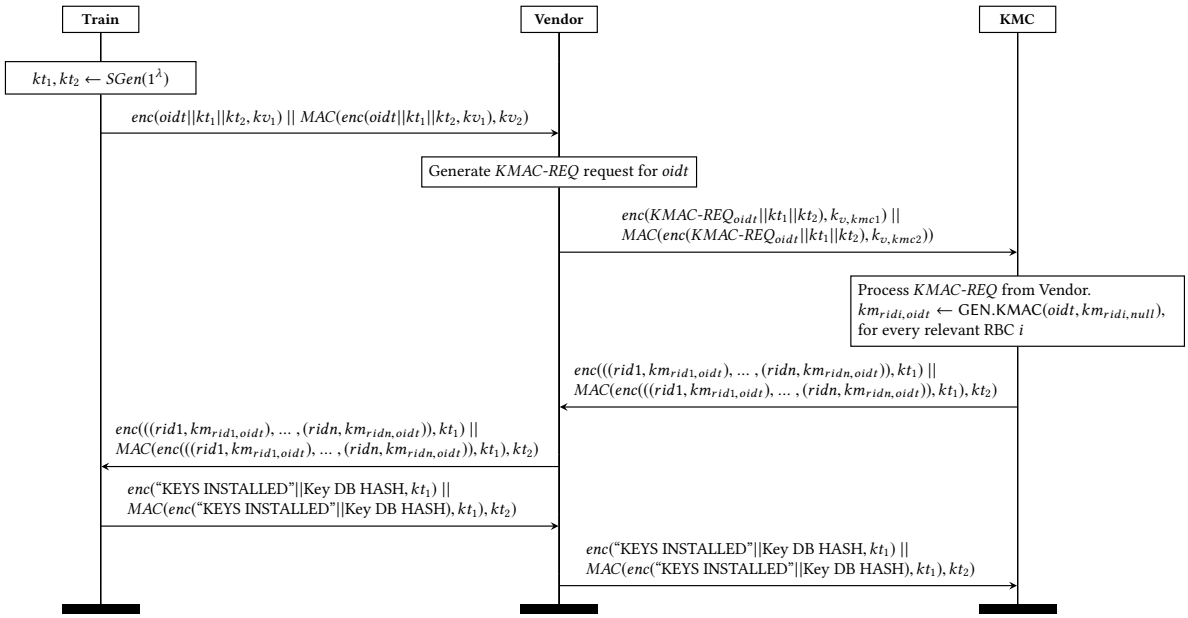


Figure 6.5: TRAKS Key Issuance Protocol for a new train t. The train generates its own transport keys prior to encrypting and computing a MAC using keys kv_1 and kv_2 respectively. These two keys are supplied to the vendor as part of the keying request to the KMC. Upon receipt, the KMC will process the request, generating the necessary keys for the train, returning them, encrypted, to the vendor for installation on the train. After installation, the train submits a checksum of the key database to ensure that the keys held by the train/RBC match what the KMC believes should be installed. For new RBCs, a similar process is carried out, as outlined in this section. An online version may be used, meaning that the vendor is only involved during commissioning.

ERTMS specifications and National Standards Authorities is that, for example, in Great Britain, recommendations by the Rail Safety Standards Board [123] do not quantitatively recommend the validity of ERTMS keys, beyond “the most time the key is required". TRAKS, therefore, includes regular rekeying intervals to prevent attacks, such as those presented in Chapter 5. Rekeying should be performed during maintenance periods, where trains can have keys safely installed at the depot. When rekeying or revocation takes place, the KMC updates the blacklist as required. The introduction of a blacklist, maintained and distributed by the KMC is not considered to impose a burden on the Infrastructure Managers and train owners.

Already connected to a fixed network for the purpose of RBC-RBC handovers, RBCs can

move to online communication with the KMC, where the keys can be managed online at regular intervals. However, due to the restrictions imposed by GSM-R and the need for safe key maintenance as discussed in this chapter, it is recommended that train key maintenance should take place offline at the depot.

6.10 Applying TRAKS in a Secure Architecture

In this section, we will review applications where TRAKS can be used and the considerations when implementing the scheme. For EuroRadio, we will discuss how TRAKS may be used, before describing in detail how EuroBalises, trusted for location reports and speed/track profile data, can be protected using a cryptographic MAC supplied by TRAKS. Finally, we will consider a wider application of TRAKS in an Industrial Control System (ICS) setting.

6.10.1 EuroRadio MAC Keying

For use within the EuroRadio handshake protocol, there are some implementation considerations that must be taken into account.

Core to the TRAKS framework is the use of a PRF. For a National Infrastructure Manager, this could be a HMAC function, for example HMAC-SHA-256, which is believed to be post-quantum secure. For key management operations, when keys are in transit, a similar keying mechanism to the current ERTMS scheme would be used between the KMC and approved organisations. As an example, for TRAKS, this should be limited to train operators and vendors, to ensure that the security of keying material is never compromised. As the current scheme relies on 3DES keys, the first 168 bits can be used in place of the 3DES keys, until support for the full 256-bit key length is implemented.

The blacklist should be cryptographically signed or accompanied with a MAC, which, when retrieved on a regular basis can ensure that, in the event of revocation, RBCs and trains do not interact with otherwise possibly compromised infrastructure.

6.10.2 EuroBalise Payload Security

As previously identified, EuroBalise data is protected with a simple CRC [139]. This would allow an attacker, with relative ease, to define their own balise payload, potentially increasing line speeds or creating 'phantom' blocks, which overlays two sets of balise groups. The importance of the balise in this architecture is that it is the sole source of data for location and track speed/profile information, trusted by the train to be accurate and used for position reports to the RBC. This data is then used by the RBC for safe supervision of the network and safety-critical decision making.

Generally, 'read-off' components are secure in providing broadcast information in plaintext when used in conjunction with some verified component, or if there is external validation of their data. However, when used as part of a trusted architecture in making safety decisions, it is not acceptable to supply only a checksum to assure the integrity of the data presented. The only protection offered through this method is error detection which would not defend against an attacker who is capable of manipulating the data presented by a balise, or emulating a balise, such that these payloads can be arbitrarily forged. It is essential, therefore, that the payloads presented by balises should be authenticated. With the exception of ETCS Level 1, balise payloads are static and can be signed or accompanied with a MAC. In ETCS Level 1, the balise is connected to a Lineside Electronic Unit (LEU) and uses EuroRadio payloads to relay dynamic signalling information to the OBU. The TRAKS framework can, therefore, be used to define a key management scheme for 'per-balise' MAC keys in ETCS Levels 2 and 3, ensuring that the compromise of one balise-specific key does not allow an attacker to impersonate other balises in the rail network.

At the core of TRAKS, the KMC would generate a national balise secret, e.g. for Great Britain, the key km_{GB} would be used to generate all subkeys which are linked to the `NID_C` for a given line. The output of this process is $km_{\texttt{NID\_C},bgid}$, which would be installed on all OBUs allowed to operate within a given `NID_C`, where *bgid* is the identity of the balise group.

As a train passes over the balise, it reads the telegrams presented to it, derives the MAC key using the broadcast balise group ID, and verifies the MAC. Should the MAC fail verification, an internal 'violation' is recorded, so that the Infrastructure Manager can attend to the potentially faulty balise. In the event that it was, in fact, the train at fault, where erroneous 'violations' were recorded, this issue would be expected to be referred back to the vendor for reactive maintenance.

Under TRAKS, a further requirement is added to ensure that all OBU units must have some trusted execution environment installed, for example, Intel SGX or ARM TrustZone. This is necessary to prevent the set of $km_{\texttt{NID\_C},bgid}$ keys from being extracted from the OBU and to ensure that only authorised cryptographic operations are performed using the installed keys. Balises, however, do not have the $km_{\texttt{NID\_C},bgid}$ installed on them at any time – this specific key is only made available to the balise programming/encoding units during installation and maintenance periods. The only data made available to the balise during programming is the *(plaintext, MAC)* payload. As a result, no engineering changes are required to be made to the balise, as it is not required to carry out any computations.

As balise keys are always derived and never stored on any system, the balise-specific keys provide defence-in-depth, ensuring that the compromise of one balise MAC key does not compromise other balises, as the MAC key is specific only to that particular balise. It is worth noting, however, that other possible forms of attack against a balise (e.g. shuffling and removal) are not prevented through this change. We will explore this further in Chapter 8.

An alternative solution to appending a MAC to the payload presented in a balise telegram is to implement a protocol where trains and balises may communicate with each other and a MAC is generated in real-time. In SUBSET-085 [145], there is a requirement for the balise group to be read and provide three telegrams to a train travelling at 500km/h. If an interactive authentication protocol between the train and the balise group was to be implemented, this would no longer be possible, due to the time constraints for a balise to be read.

Providing a static payload appended with a MAC is, therefore, a more appropriate and sustainable proposal which is more efficient and less prone to errors in transmission. Assuming that the time taken to read and process an 'unauthenticated'/CRC-encoded 1023-bit balise telegram is t milliseconds (i.e. 1.81ms for a train running at 500km/h, assuming no read failures, average conditions and data rates), the impact of adding TRAKS authentication would increase this time by only 6%. This effectively means that the impact is only 1.92ms or 26.6cm of travel for the train to read, derive the balise MAC key and validate the balise payload. This could be further reduced with hardware developments, although this would have marginal benefits for, perhaps, significant cost.

One possible caveat of moving to a MAC is that the current CRC and other data used as part of the integrity verification process accounts for 110-bits of the total balise payload. Replacing this with a 128-bit MAC would reduce the available balise data payload by only 18-bits. For the majority of balises deployed, this should not pose a problem, given that Packet 44, used to send non-ERTMS specific messages to the train from a balise, as defined by the Infrastructure Manager, typically would not fill all of the 1023 bits available for the balise payload.

6.10.3 Wider Applications of TRAKS in ICS PLC Environments

As a framework, TRAKS can also be applied to wider ICS environments, where there are a number of Programmable Logic Controllers (PLCs) in ICS/SCADA settings which may communicate with one another, and the communication protocols may not be protected. As an example, MODBUS and PROFIBUS, two widely-deployed protocols in industrial control systems, due to age, do not offer any cryptographic protection of their payloads [77], exposing them to an attacker who can 'sniff' or arbitrarily inject their own packets. Allowing an attacker to carry out this man-in-the-middle attack would affect the overall operation of the system, with the potential to create an unsafe situation.

A simple solution to this type of attack would be to send messages with a MAC attached

between PLCs where there is some attestation that the message is authentic and has not been tampered with in-flight. For an ICS owner, however, using a single key for all devices across the operational network achieves little security as the keys could be extracted by alternative means. Partitioning the system would allow specific Operational Technology (OT) 'zones' to be defined, analogous to the `NID_C` variable used in ERTMS to identify regions and lines. Here, a centralised key per zone could be defined and installed on the appropriate hardware responsible for that zone. A derivation step would then be included in the PLC logic to allow a per-device key to be used in the actual communications. This would also allow the owner to identify the source of other attacks taking place on their network, where an attacker could leverage a vulnerability on that PLC to coerce it into a specific state.

6.11 Chapter Summary

In this chapter, we have defined and detailed a new key management solution which can be applied to a number of ICS environments. By leveraging proven cryptographic techniques, we are able to create and achieve an interoperable, backwards-compatible solution which can be deployed for ERTMS. This scheme reduces the National Infrastructure Manager's overheads, whilst considering future threats that would effect the current ERTMS scheme by delivering post-quantum security. Whilst this chapter has primarily focussed on the EuroRadio protocol, we have also demonstrated its ability to work as a framework to include EuroBalises in the secure architecture of ERTMS, ensuring safety through security, where both function cooperatively and not independently. Through the application of a partitioning principle to ERTMS with TRAKS, we are able to establish a key distribution scheme which maintains the same level of security offered by the current set of standards, whilst delivering significant benefits to ICS owners and operators.

It is important to note from this chapter that, as the current scheme dates back to 1997, the key management burden is only now being experienced by Infrastructure Managers, as

the number of ERTMS deployments increases. New proposals are being introduced, but they only consider partial solutions to an existing problem. These solutions do not take into account future threats or the potential exposures that are being introduced, for example, a lack of post-quantum security, allowing an attacker to capture traffic now and break the cryptography when the capability arrives – something well within the lifespan of ERTMS. As a result, within TRAKS, we considered this particular problem and designed a solution whilst providing formal security guarantees, an assurance which is not addressed in the proposed online ERTMS scheme.

From this and previous chapters, the analyses carried out have highlighted how the standards require regular review. However, for many operators and system owners, this is not a simple process. Many may lack the necessary expertise and understanding and the physical deployments may also be vulnerable, whether it is due to outdated software, or unappreciated interconnectivity. Using the processes developed in this thesis, we will consider the 'big picture' of system architectures and identify a process to enable asset owners to assess the security of their architectures.

Chapter 7

Modelling Threats to Rail

This thesis has assessed the ERTMS standards in detail, firstly diving down the stack of protocols used to deliver in-cab signalling. As part of this analysis, whilst we were able to provide some levels of assurance for those protocols, we also found flaws which, when chained together, provide a valid attack vector and the opportunity for a malicious adversary to exploit. We then proceeded to return up the stack, considering what the future threats *could* be for ERTMS, defining a solution which provides future security to current systems and a flexible framework which can be used in other Industrial Control System (ICS) environments outside of train-to-trackside signalling. We evaluated, as an example, the EuroBalise and, through our notion of what a secure architecture should be for an ICS, closed the gap which now allows balises to be trusted and secured.

System architectures, however, vary and, until this point, we have focused primarily on a reference ERTMS architecture, abstracted the ERTMS interfaces and also the components included onboard a train. This, however, does not take into account the diverse nature of deployments and the interconnected nature of systems. In this chapter, we will explore a methodology which allows operators to consider the overall security of their system architectures, through the use of a new modelling tool which abstracts notions of security for engineers.

This abstracts the methodology applied in this thesis of taking a starting node and following the various specification documents to understand the security relationship between components to provide a 'first steps' tool for asset owners to understand the security dynamics of their architecture.

7.1 Motivation

With the introduction of the EU Network and Information Security (NIS) Directive in 2018, ICS system owners are required to be able to rationalise and assure the security of their systems, so that they are compliant with the Directive and have made efforts to minimise risk as far as possible. One issue that exists for many asset owners is that when the standards and products were designed and built, the primary emphasis was to provide functionality and safety, whilst security was considered less of a priority. As these systems have a much longer lifespan (in the range of decades) compared to commodity systems, ICS deployments may conform to older standards. For example in ERTMS, the GSM-R A5/1 cipher dates back to 1987 [85] and EuroRadio, proposed in 1997 [22] no longer provide sufficient security today. Vulnerabilities in these ciphers are now being realised, where the capabilities of adversaries have significantly improved, at lower cost and effort.

Modelling tools are commonly used by engineers when they are validating their safety cases for new systems and to rationalise assumptions made within the design of the systems. The same technique can be applied to consider the security of a given architecture. Tools are available today which model the security of a system but they do not allow the asset owner to visualise how an attack could propagate through their system. Moreover, they require the asset owner to formulate the risk profile for their systems which can be prone to error, due to a misunderstanding or domain and subject-relevant knowledge not being captured as part of profiling the system, e.g. using the Altran 'REVEAL' methodology [76].

A minimal security assessment of assets should aim to address the following questions:

- What is the set of assets we want to protect?
- Who are our attackers and what are the threats to our assets?
- How likely is an attack against this asset which has been identified as requiring protection?
- What would be the consequences if this asset was compromised?
- What is the cost of preventing this asset from being compromised/minimising the risk of compromise?

Risk analysis is a prime focus in this chapter. However, this is very subjective, as an intricate understanding of individual components in a system is required, especially when these components are interconnected into a network. Asset owners, when presented with the potential risk of a threat to safety-critical components, may deem the risk to be unacceptable, regardless of the level of risk – the mere presence of any threat to the safe operation of a system could, for example, endanger human lives. The evaluation of risk, probability and impact needs to be carefully balanced, given that a high risk may have a negligible consequence. As an example, let us consider a national infrastructure which has a vulnerable communications protocol running between nodes. An attacker who is able to compromise this protocol could potentially have unrestricted access to the nodes. From the perspective of a train using GSM-R, there would be the potential for a disruptive effect by an attacker overloading the EuroRadio layer with messages that cannot be processed, either due to poor formatting or an incorrect MAC, which could trigger the session to be terminated. Conversely, an air-gapped interface between the train control systems and, say, a passenger seat reservation system would have little impact on the safe operation of the train but would have a disruptive effect on the passengers. These risks should be reliably modelled and captured to ensure that they are understood and their interface with the specified NIS thresholds are identified.

In the United Kingdom, for the transport sector and its Operator of Essential Services (OESs), the Department for Transport [52] thresholds set for rail-related incidents only con-

sider how many services were affected or delays imposed, rather than take account of the root cause. In the case of a train being targeted at Clapham Junction, an attacker could jam the GSM-R reception, which could paralyse part of the network for an indefinite amount of time. Under the defined thresholds, this would be considered as a NIS-reportable incident. Conversely, a power failure which causes 20% of services to be cancelled would also need to be reported under these thresholds, irrespective of the root cause. In addition to the NIS Directive, some rail asset owners are implementing the ISO/IEC 62443 Standard [32], an organisational-based security standard. However, the implementation of both ISO/IEC 62443 and the NIS Directive present a large knowledge, experience and skills gap. One risk that may arise during implementation is that the operator simply does not understand the risks that exist in their systems, nor may they understand the level of connectivity in those architectures. As an example, the Stuxnet malware leveraged the fact that 'air-gapped' systems, in reality, had data transferred between them using USB media, allowing the malware to propagate and have such a destabilising effect on the system [35, 92]. More recently, the BlackEnergy [88] variant achieved a similar outcome by using the interconnected nature of operational systems to IT management networks, affecting the Ukraine power grid in 2015 [54, 94]. Spearphishing has now become the attack vector of choice for many adversaries [86], allowing them to gain access to the management system and use it as a 'pivot point' to gain access and tamper with ICS components, for example, Human Machine Interfaces (HMIs).

These are a few examples of threats which were identified in ICS deployments or where the interconnectivity of systems could have the potential of introducing risk to the overall operation of the system. Methodologies such as that presented in [60] provide a high-level means of considering the threats to rail networks by defining a manual process to identify the security risks to infrastructure. Given the architecture of a system, it does not, however, provide the asset owner with an automated method of identifying the cybersecurity risks to their infrastructure.

7.2 Contributions

In this chapter, we will explore a new tool that allows asset owners and engineers who are not security experts to assess the security of their systems. This is achieved by modelling components of a system as a graph of connected systems. Here, the dataflows between these components are also captured and the transformation of this data from input to output (e.g. a balise's location payload being converted into authenticated data by the European Vital Computer (EVC) for submission to the RBC) is used in its analysis. The tool allows the asset owners to find paths across the model between the assets and judge the compositional security, in addition to enabling them to simulate possible attackers, their capabilities and the entry points they may use.

As previously identified, leaving the calculation of a risk metric to the asset owner can sometimes be difficult, with the potential for errors to arise. Furthermore, if the metric is not standardised, there is no way to compare systems and their risk profiles. Thus, the SCEPTICS tool presented in this chapter leverages the Common Vulnerability Scoring System (CVSS) metric to define the security profile for a given component or link and, based on this, derive the likelihood of an attack on that asset being successful. To determine the security of a group of elements, the inclusion-exclusion principle is used to compute the probability for a given path. Given that the asset owner is now able to quantify the security of their systems, they can compare the security of similar systems, especially from a regulatory perspective.

What this tool provides is the ability to conduct assessments that convey contextual meaning to the asset owner, allowing them to visualise the improvements being made. For non-security experts, they are able to rationalise the security of their architectures without a detailed knowledge of the vulnerabilities that exist and better understand the implications to the security of the system when modifications are made or, for example, if firmware updates are not promptly applied. This, though, does not remove the need for expert security assess-

ments, as these may highlight details which cannot be captured through the asset owner's lack of detailed security knowledge.

As part of the analysis, 'interesting' components or those which have a critical function are highlighted to the asset owner which, if compromised, may have a higher impact on the overall security of the system under assessment. This allows them to identify and prioritise improvements which would reduce their overall exposure. Aside from allowing asset owners to assess the *current* security posterity of their systems, the tool also allows them to experiment and assess the impact of the different strategies and changes that could be made to improve the overall security of the architecture before any financial investments or modifications to the system have been made. Using this approach allows the asset owner to identify the strategies which are the most cost effective, whilst delivering the most appropriate security improvements to the overall system.

7.3 The SCEPTICS Modelling Tool

In this section, we will appraise the SCEPTICS tool in detail, starting with the formal grammar the tool uses to reason with its input, followed by the probabilistic computations made by the tool. We will then discuss how the input values for the probabilistic calculations are determined, prior to an overview of the specific implementation and design details for the tool.

7.3.1 Input Grammar

Three specific inputs to the tool are required: a *system graph* which describes the individual components, their linking between each other and the respective data flows between components, an *adversarial model*, which describes the capabilities of an attacker and, finally, a *list of assets* which the attacker may target or use to enter the system. From these three inputs, the tool is able to identify the paths which are most likely to be taken by the attacker and determine the most vulnerable components in the system. Looking forward, the tool then al-

lows the asset owner to make changes to their model and re-run the tool to assess the effect of their proposed mitigations. The type of analysis and level of detail is controlled through the adversarial model and the list of assets given to the tool. More formally, the grammar of the inputs to the tool can be presented as:

$$\begin{aligned} \langle \text{system graph} \rangle &\models \mathit{List}\langle \text{node} \rangle\ \mathit{List}\langle \text{edge} \rangle \\ \langle \text{assets} \rangle &\models \mathit{List}\langle \text{asset} \rangle \\ \langle \text{adversarial model} \rangle &\models \mathit{List}\langle \text{adversary} \rangle \end{aligned}$$

For each of these inputs, let us consider them individually:

7.3.1.1 The System Graph

The system graph is a directed graph which describes the architecture of the system under assessment, where individual components are modelled as nodes and connections between them are given as edges in the graph. As an example, from Figure 7.1, *GSM*, *WiFi* and *GPS* can be expressed as nodes in the model with *Car BUS* additionally being modelled as a node. The *Car BUS* is shown differently to the other nodes as it shows how the asset owner can define components in their system as granular or generic, as required, where, for example, the *Car BUS* is exploded into its constituent components as shown later in Figure 7.2. Here, they can express the shared security properties of that node or, alternatively, be more specific if more

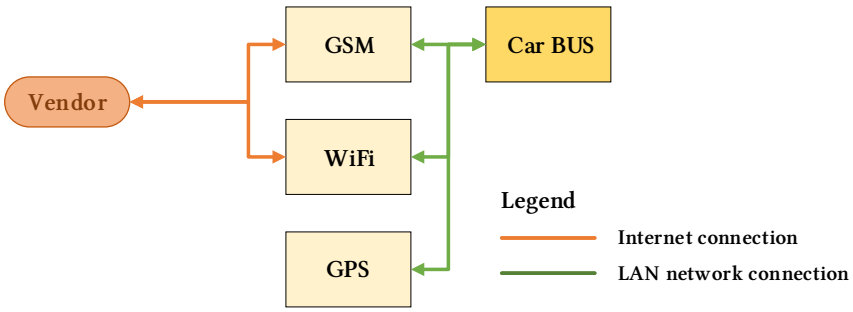


Figure 7.1: Reference Example Model based on Figure 7.2.

detail is known about that component which could affect the overall security of the model.

For each node, a *node identifier* is given with a descriptive name and a list of labels which describe the supported *data types* for that particular component. Specifically, the *data types* can refer to standardised types of data, whether it is a protocol (e.g. MODBUS/TCP/UDP) or some custom-defined type (e.g. geographical positioning data, unauthenticated data). Nodes also include *bridges* entries which describe the capabilities of that node to convert data from one type to another. Finally, each node has a data profile, *DP*, which has a *data type*, *link type* (similar to data type but describes the physical transmission method, e.g. electrical or Ethernet) and a corresponding CVSS profile, *CP*, which describes the security characteristics of that node. Further details of this profile are discussed later in this chapter. Formally, the grammar to describe a node can be given as:

$$
\begin{aligned}
\langle \text{node} \rangle &\models \texttt{id:}\langle \text{node id} \rangle\ \texttt{name:}\langle \text{name} \rangle\ \texttt{data\_types:}\mathit{List}\langle \text{data type} \rangle \\
&\quad\ \texttt{bridges:}\mathit{List}\langle \text{bridge} \rangle\ \texttt{data\_profile:}\langle \text{DP} \rangle \\
\langle \text{bridge} \rangle &\models \langle \text{data type} \rangle : \langle \text{data type} \rangle \\
\langle \text{DP} \rangle &\models \texttt{data\_type:}\langle \text{data type} \rangle\ \texttt{link\_type:}\langle \text{link type} \rangle\ \texttt{CP:}\mathit{List}\langle \text{CP} \rangle \\
&\quad\ \mid \texttt{data\_type:}\langle \text{data type} \rangle\ \texttt{CP:}\langle \text{CP} \rangle \\
&\quad\ \mid \texttt{link\_type:}\langle \text{link type} \rangle\ \texttt{CP:}\langle \text{CP} \rangle \\
&\quad\ \mid \texttt{CP:}\langle \text{CP} \rangle
\end{aligned}
$$

Edges in the graph, i.e. connections between nodes in the graph, have a similar grammar to nodes. Specifically, they can have one or more *DP* associated data profiles, where an edge is uniquely defined on a 'per-link' basis:

$$
\langle \text{edge} \rangle \models \texttt{id:}\langle \text{node id} \rangle\ \texttt{id:}\langle \text{node id} \rangle \texttt{data\_profile:}\langle \text{DP} \rangle
$$

Using Figure 7.1, we can derive an example input grammar as:

node1 ⊨ `id:`nGSM `name:`GSM `data_types:`[LAN net. con., WAN net. con.]
`bridge:`[bridge1] `data_profile:`dpn1

bridge1 ⊨ LAN net. con.:WAN net. con.

dpn1 ⊨ `CP:`0.381

node2 ⊨ `id:`nCB `name:`Car BUS `data_types:`[LAN net. con.]
`data_profile:`dpn2

dpn2 ⊨ `CP:`0.972

edge1 ⊨ `id:`nGSM `id:`nCB `data_profile:`dpe1

dpe1 ⊨ `data_type:`LAN net. con. `data_link:`Ethernet `CP:`0.200

7.3.1.2 Adversarial Model

The second input to the tool is the adversarial model, which captures the capabilities of an adversary to the system. This component is important as it means that the tool will only consider attackers who genuinely present a threat and should be evaluated. Each adversary has a set of *entry nodes* from which they may start, with the optional inclusion of being able to exploit specific *data types* and *link types*. The supporting grammar to describe an adversary is expressed as:

⟨adversary⟩ ⊨ `entry_nodes:`*List*⟨entry nodes⟩ `data_types:`*List*⟨data types⟩
`link_types:`*List*⟨link types⟩

An example of such an adversarial model as input to the tool is given below:

adversary1 ⊨ `entry_nodes:`[ε] `data_types:`[RF connection]
`link_types:`[Wireless]

The adversary defined above is able to compromise any connections that use *RF connections* or *Wireless* link types. Referring to Figure 7.1, this would allow the adversary to use the *WiFi*

and *GSM Nodes* as both possess a wireless interface and may be used as entry points to the system. Thus, an equivalent notation for the adversary can be given as:

$$\text{adversary1} \quad \models \quad \texttt{entry\_nodes:}[\text{WiFi, GSM}]\ \texttt{data\_types:}[\varepsilon]\ \texttt{link\_types:}[\varepsilon]$$

7.3.1.3 Asset List

The final input to the tool is the list of assets which are under assessment in the context of the entire model, specifically those assets which might be targeted by an adversary. A rail asset owner, for example, may want to understand the ways in which the *Car BUS* onboard a train may be compromised, where the corresponding asset would be the *Car BUS*. The attack space can also be constrained by the inclusion of specific *data types* handled by that node. This results in the following grammar to describe an asset:

$$\langle\text{asset}\rangle \quad \models \quad \texttt{id:}\langle\text{node id}\rangle\ \texttt{data\_types:}\mathit{List}\langle\text{data types}\rangle$$

From Figure 7.1, a concrete example of two assets, one unrestricted and another restricted, are respectively given below:

$$\begin{aligned} \text{asset1} \quad &\models \quad \texttt{id:nCB} \\ \text{asset2} \quad &\models \quad \texttt{id:nCB}\ \texttt{data\_types:}[\text{LAN net. con.}] \end{aligned}$$

7.3.2 CVSS Security Profiles

Defining the interplay, especially the security of system components, can have significant implications on how attack vectors (i.e. the attack paths found by the tool) are ranked and the relationships between them are found. Requiring asset owners and modellers to specify the security characteristics of their systems (especially when it must be expressed as a single value) is unreasonable, as a lack of security expertise can lead to an under/overestimation of the reality of the security of their assets. Existing tools allow a targeted analysis of these

systems but, as described in Chapter 3, still require the asset owner to determine these values, involving a high degree of manual effort.

To overcome this problem, we can integrate the CVSS framework [1] into the SCEPTICS tool and use it to uniformly generate and assign security (*CP*) profiles to nodes and edges. The CVSS metric is suitable for this task as it represents a standard way of determining a score that expresses the severity of a Common Vulnerabilities and Exposure (CVE). The CVSS framework also allows the inclusion of information related to the severity of a vulnerability and other contextual information. For example, it can include the 'environmental impact', i.e. how does a vulnerability for this given node affect neighbours directly connected to it? An alternative metric that can be included is the 'temporal impact' which assesses whether a remedy is available for the vulnerability and how close it is to be successfully exploited, allowing for a more consistent granularity of the profiles.

A CVSS vector is formed of three parts: the *base score*, *temporal score* and *environmental score*, each of which are independent but, together, form the final, overall, score. The *base score* is defined by the vector used to attack the component, e.g. via a network connection or physical access, the complexity of the attack, the privileges required and whether the user has to perform any actions which would otherwise render the attack unsuccessful or force another path to be taken. The Confidentiality, Integrity and Availability (CIA) 'triad' is taken into account as part of this score, as is the case when the attacker is in a position of elevated privileges through the compromise of that component. The *temporal score* concerns itself with the 'here and now' security of the component, i.e. if there are any exploits 'in the wild' against that component, or if there are theoretical but not yet practical attacks. The score also factors how these exploits may be remediated, if a solution exists, and how confident we, as the assessor are, in the reports and literature about the specific exploitability of that component. Finally, the *environmental score* looks beyond the component and evaluates how the compromise of

[1] `https://www.first.org/cvss`

that component could affect the neighbouring, interconnected, components, what reliance and guarantees are expected of the component and the impact on these guarantees if it was compromised.

Generally, CVSS vectors are expressed as numerical values, $i \in 0, \ldots, 10$, using the equations defined in the CVSS standard [65] (included for reference in Appendix D). However, the tool uses probabilistic computations on the model to rank and compute the security values for groups of components, where the CVSS values, CP must be expressed in terms of $CP \in [0, 1]$. The tool itself only requires the probability of an attack being successful (i.e. its exploitability), so the *base score* of the CVSS value is transformed as follows:

$$CP = \frac{ESC - 0.121}{3.887} \tag{7.1}$$

ESC is the exploitability sub-score component of the CVSS base value which, itself, is defined as:

$$\begin{aligned} ESC = 8.22 \times AttackVector \times AttackComplexity \times \\ \times PrivilegeRequired \times UserInteraction \end{aligned} \tag{7.2}$$

For the purposes of our calculations, we do not need to consider the impact on the CIA requirements on the component as we only use the ESC component of the base score. As explained earlier, this value alone does not accurately reflect the likelihood of a successful attempt to exploit a component. The ESC value is therefore scaled using Equation 7.1 to derive the final probabilistic value, which is then used by the tool in its calculations.

CVSS Profile Example. Let us use the GSM component, as shown in Figure 7.1, as a concrete example of how we may determine the CVSS profile for a node or link in the model. GSM-R, as we recall from previous chapters, is used to provide train to trackside communications and, for ERTMS, a data link for in-cab signalling. As the weak A5/1 encryption scheme is used, this affects some of its guarantees for the confidentiality and integrity of messages.

We can define the CVSS vector in the model for GSM nodes as:

$$CVSS_{GSM}{=}AV{:}A/AC{:}H/PR{:}N/UI{:}N/S{:}C/C{:}H/I{:}H/A{:}H/E{:}F/RL{:}U/RC{:}C/CR{:}L/$$
$$IR{:}H/AR{:}H/MAV{:}N/MAC{:}H/MPR{:}N/MUI{:}N/MS{:}X/MC{:}L/MI{:}H/MA{:}H$$

This vector describes the parameters of the CVSS metric, for example, the requirement that the attacker has 'adjacent access' (`AV:A`), using a component of the stack (radio) to gain access to the endpoints. That said, whilst gaining access to the downlink (from base station to mobile station) is relatively simple, affecting the uplink (mobile station to base station) is complex due to the time-sensitive nature of GSM timeslots (`AC:H`), coupled with the need to determine the exact timeslot and frequency used by the train.

For the remaining parts of the vector, let us consider highlights; for the temporal score, we acknowledged that a functional means to compromise the downlink exists (`E:F`) as rainbow tables for the A5/1 cipher are available. It is currently unclear what the remedial action would be (`RL:U`) as the cipher is used worldwide for GSM-R and may require significant reimplementation and cost. We also place confidence in this vector (`RC:C`) based on tutorials and the attack roadmap given in Chapter 5, demonstrating how easy it is to capture data from the downlink. A full list of these parameters are available in Appendix D.1.

By applying Equation 7.2 and the mappings from Appendix D.1, we can compute the exploitability subscore $ESC = 1.6$. Using Equation 7.1, we can compute the corresponding CVSS profile (CP) for the GSM component ($CVSS_{GSM}$) as $CP_{GSM} = 0.3805$. What this value indicates is the probability that an attacker *can* compromise and affect the GSM link which, when factoring in the environmental impact, would have a propagative effect on dependent, interconnected, systems.

During the computation over the model, the tool's attack paths consider all of these dependent systems, demonstrating how an attacker can leverage one, possibly weak, component

to have maximal impact. An example attack path (with nodes included from Figure 7.2) could take the attacker from the GSM system (antennae, modem and equipment) to the EVC, where a denial of service attack would lead to the train entering a fail-safe condition and stopping. Similarly, injecting into the GSM stream would allow the attacker to go from the GSM node, through the GSM trackside infrastructure to the RBC, where the RBC could be given a false report or have messages replayed, placing it in an unpredictable state.

Allowing Asset Owners to Custom-Define Security Profiles Whilst using CVSS as a framework to determine input values assists asset owners and reduces the likelihood of errors and inconsistent security profiles being applied to components within the model, it is not a replacement for an expert security analysis. The scores may not always match what can be discovered through an assessment of domain and subject-specific knowledge [76], where some small, but critical detail, cannot be simply captured in the CVSS vector. To overcome this, the asset owner can override the security profiles with correctly-formatted values provided by them, i.e. $CP \in [0, 1]$.

A beneficial side effect of this specific tool feature also allows the modeller to run simulations to determine those options which provide the most effective security solution. This will be discussed later in this chapter.

7.3.3 Requisite Definitions and Formal Computational Model

We will now consider how the attack vectors are modelled and the method used to compute the values which determine the security of these vectors.

Firstly, we need to define the system graph input more formally as a directed graph, $G(N, E)$, with N as the set of all nodes (the components contained within the model) and E, the set of edges (connections) contained within the model (the directed Graph G). The probability space, $(\Omega, \mathcal{F}, P)$, is used where the sample space, Ω, is the powerset of edges in $G(N, E)$, i.e. $\Omega = \mathbb{P}(E)$. The event space, $\mathcal{F}$, is the powerset of the sample space $\mathcal{F} = \mathbb{P}(\Omega)$, where P

is a probability function $P : \mathcal{F} \rightarrow [0, 1]$, which tells us the probability of a given edge being exploitable by an adversary (i.e. it exists in the graph). From this formalism, we can state that an *attack vector*, starting from a given component in the graph, n_s, targeting a specific asset n_e through the graph $G(N, E)$, can be defined as:

Definition 7.3 (Graph Path) *A path ($\rho(n_s, n_e)$) in the graph $G(N, E)$ is a sequence of unique edges $e_i \in E, i = 0 \ldots n$, where the start node, n_s is e_0, the end of the path is given by the end node n_e as e_n and, for each pair of consecutive edges (e_i, e_{i+1}), the end node of e_i is equal to the starting node of e_{i+1}.*

A path from the node e_s to e_n represents an attack which starts at e_s, following links and passing through intermediary nodes, reaching e_n. The end of the path (e_n) is an asset that an attack may take place, where the start of the path (e_s) is the entry node that an adversary uses to launch their attack, where multiple paths may exist between e_s and e_n.
In order to verify that the paths in a graph are valid (i.e. that they exist), we can define an event that the path exists from the event space as follows:

Definition 7.4 (Event that Path ρ Exists) *For a given path, ρ, the Event that the Path exists is $\varepsilon_\rho = \{o \mid o \in \Omega \wedge \rho \subseteq o\}$.*

Now, let us establish the probability for a given path in the Graph.

Lemma 7.5 *Given a path ρ, the probability that the corresponding path exists occuring ε_ρ is*

$$P(\varepsilon_\rho) = \prod_{e_i \in \rho} P(e_i).$$

Proof. The probability of an event ε_ρ occurring can be given by the probability of all the edges of ε_ρ, i.e. those that are part of the path ρ and those that do not exist in ρ, i.e. $P(\varepsilon_\rho) = \prod_{e_i \in \rho} P(e_i) \cdot \prod_{\overline{e}_j \notin \rho} P(\overline{e}_j)$, where $e_i, \overline{e}_j \in \varepsilon_\rho$ are the edges that form part of ε_ρ, and not of ε_ρ respectively.

ε_ρ, however, contains all possible combinations of edges $\overline{e}_j$ that are not part of ρ, i.e. $\prod_{\overline{e}_j \notin \rho} P(\overline{e}_j) = 1$. Therefore, $P(\varepsilon_\rho) = \prod_{e_i \in \rho} P(e_i)$. □

We finally need to introduce the notion of the inclusion-exclusion principle whereby:

Lemma 7.6 *The probability of a set of events $\mathcal{P}$ can be described by the existence of n paths ρ_i, computed as:*

$$\begin{aligned} P(\mathcal{P}) &= P(\bigcup_{i=1}^{n} \rho_i) \\ &= \sum_{k=1}^{n} (1)^{k+1} \cdot \sum_{\substack{\varepsilon_{\rho_i} \subseteq \mathcal{P} \\ |\varepsilon_{\rho_i}|=k}} P\left(\cap\ \varepsilon_{\rho_i}\right) \end{aligned}$$

Proof. Lemma 7.6 can be observed as a probabilistic application of the inclusion-exclusion principle [28]. □

7.3.4 Tool Design and Implementation

We will now review how the SCEPTICS tool was implemented, highlighting some of the key design decisions and optimisations.

Discovery of Attack Vectors As previously highlighted, the input to the tool is a directed graph such that, for a given system transformed into a Graph $G(N, E)$, the possible probabilistic attack vectors of this graph will be the paths from $G(N, E)$. Searching for the paths is achieved through the breadth-first search algorithm [44], a fairly simple, but efficient, search algorithm with a worst-case complexity of $O|N|$, where the cardinality of N represents the number of nodes that are present in the graph.

Transforming the Graph and Route Discovery System architectures, especially those in IT/ICS environments are typically complex, highly-connected graphs, which contain multiple layers of information (e.g. the protocols used, the physical medium and contextual information about the data), where it is sometimes difficult to conduct accurate and meaningful security assessments. As an example, considering Figure 7.1, we would have to decompose the model

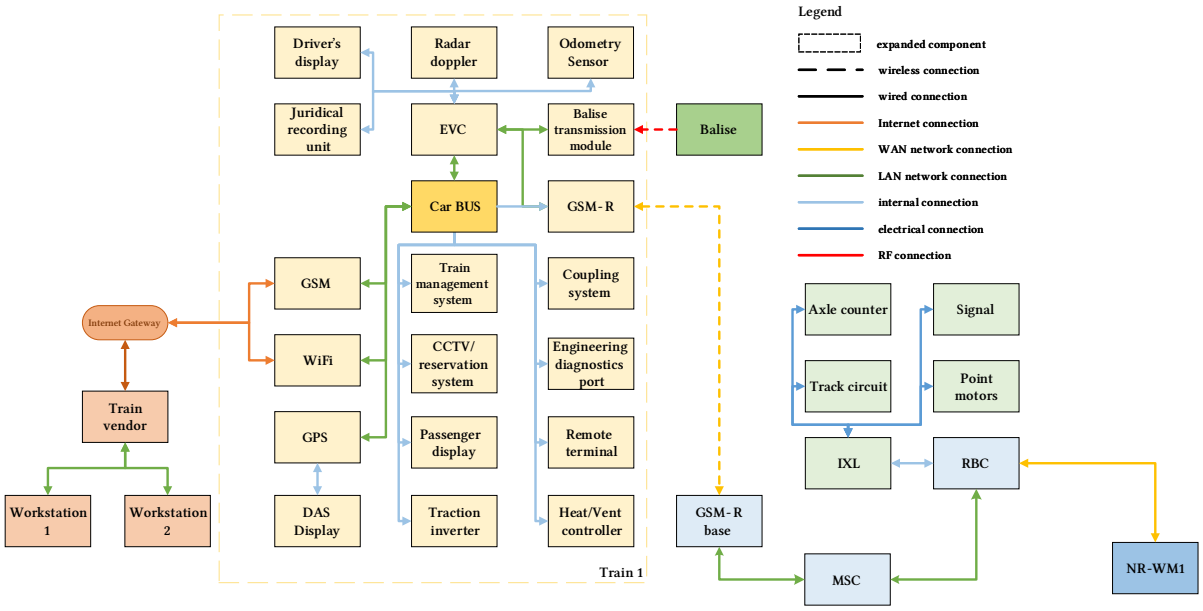


Figure 7.2: Model of an example reference ERTMS implementation, with the train systems exploded into constituent components.

into twelve independent graphs (comprised of the two medium types: wired and wireless and the six connections) and apply the security computations for each of these graphs to obtain an accurate analysis. Whilst this method could be used to establish the state of security for a system, it would not capture the highly-connected nature of the system and its interdependencies, whereas the SCEPTICS tool would capture this effectively in a single model. We will see how it achieves this in Section 7.5.3.

The SCEPTICS tool uses a layer-aware breadth-first search, supporting routing-like functionality, where nodes are appended to a path if, and only if, they both support a specific *data type* and are connected by an edge that also supports that *data type*. To capture the interplay between these layers, the *bridge* elements of the nodes are used, which essentially describe the capabilities of that node to convert data from one layer (i.e. the *data type*) to another. A path, therefore, can be formed of several shorter paths, all belonging to different layers, provided that they share a common bridging node that allows them to traverse these layers. In Figure 7.2, we can see this more clearly. Here, the path `[GSM-R]->[GSM-R base]->[MSC]` uses a

combination of wireless (dotted line) and wired (solid line) connections and is valid only if the intermediary node, `[GSM-R base]`, bridges the two connection types. Similarly, we can use the connection type to determine whether the path is valid (i.e. the *WAN network connection* (yellow, dotted line) and the *LAN network connection* (green, solid line) has the `[GSM-R base]` node defined as a bridge between those two connection types).

7.4 Analysis Profiles

The SCEPTICS tool enables asset owners to carry out a number of tasks and analyses to assure the security of their architectures. The principal type of analysis that the tool executes is the discovery of all paths to key assets that have been defined by the asset owner. In addition, the tool enables alternative analyses including:

'All Roads Lead to' Analysis. Given that some assets in the graph are considered 'bastions' which *must* be protected from an adversary, the tool will first search for, and assess, paths which terminate at those assets. A list of those paths, where the likelihood of an attack being successful exceeds the owner-defined threshold, is then ranked and returned. In Figure 7.2, we observe that the RBC is directly connected to a number of data sources which are used in its safety-critical decision-making process. In addition to finding these paths, the tool would also find the paths from indirectly-connected systems which may have a more pronounced impact if the RBC was compromised, than the effect of the compromise of a directly-connected system.

Patient-Zero Analysis. As an asset owner, we may also want to consider the environmental impact to neighbouring systems should a particular node become compromised. Given a specific starting node, the tool will search for paths which again exceed some given threshold. What an asset owner can learn through this particular analysis is how attacks can propagate through their systems, especially in ways that had not previously identified or appreciated.

Testing New Strategies. As previously identified, by allowing the asset owner to define their own security values to nodes and edges, instead of using the CVSS framework, a further

benefit of the tool is that it enables asset owners to model and assess new strategies which better inform them of the way that they can review the security of their system architectures. Asset owners can then either amend the CVSS vectors or the probability values directly, to determine the most effective areas to implement changes and develop future priorities for investment.

The intrinsic value of the tool is in the way in which it discovers the paths and carries out probabilistic analyses, using the inclusion-exclusion principle, to assess the propagation of attacks in a given model. This path discovery enables the asset owner, in particular, to understand all the 'via' steps an attacker might take to circumvent the security of the most direct routes to gain a foothold into a critical asset.

7.5 Applying the SCEPTICS Tool to ERTMS

In this section, we will apply the SCEPTICS tool to the ERTMS architecture shown in Figure 7.2. ERTMS is one example of an ICS which is highly interconnected with a range of protocols and standards to maintain its safe operation at all times. The model in this figure is based on the reference ERTMS standard and a train architecture which has been developed from vendor technical reports [114, 108]. Here, a range of nodes have been exploded to demonstrate the level of granularity that the asset owner can assess. ERTMS was chosen instead of, for example, an onboard train network, as the standards are public, not closed and not proprietary, allowing a more thorough profiling of the security of the architecture.

To demonstrate the level of granularity, a number of assets were specified as input to the model that are available to the adversary, including the *Balise*, *Workstation1*, *Workstation2* and *GPS* assets. To determine the exploitability of some of these assets, the targets for the attacks by the adversary were set to be the *EVC*, *RBC*, *Car BUS*, *DAS Display* and *Passenger Display*.

7.5.1 Selection of Adversary Entry Points

The attacker entry points were specifically selected due to the range of differing interdependencies that exist in the rail network. As an example, the EuroBalise is implicitly trusted by the train for accurate location and track profile data, where the data is passed to the RBC to make safety-critical decisions that ultimately determine the train movements. In the event that the data presented to the RBC was incorrect, without an independent source to verify it, it would be possible for it to issue an overlapping Movement Authority, potentially placing the train in the same space as another. Similarly, for other parameters provided by the balise, such as speed limits and, in the UK, tilting constraints, if any of these parameters were compromised, the train could be placed in an unsafe situation due to the limited validation of the data.

Remote Condition Monitoring (RCM) is a diagnostic tool used by train vendors to carry out real-time monitoring of trains within a fleet. The data that the train produces is sent via the RCM system to the vendor, enabling preventative and predictive maintenance to be preplanned when the train returns to the depot, whilst also allowing the remote triage of vehicle faults whilst the train is in service. If compromised by an attacker, the train may be prematurely removed from service. If the vendor is additionally capable of making remote changes to a vehicle, the attacker could compromise the safe operation of the train. With access to the vendor's systems, the adversary could easily carry out reconnaissance of the onboard proprietary and Commercial off-the-shelf (COTS) equipment and search for vulnerabilities to exploit. In the model, we consider two engineer workstations located at the vendor's site as possible entry points for the attacker, where *Workstation1* has been infiltrated and compromised, whereas *Workstation2* has not been affected.

Finally, we assess the security of the on-board GPS which is only used for supplementary services, e.g. transferring location information to the passenger information displays and for automated announcements. If the GPS was affected, say, through spoofing, this could result in passenger disruption (e.g. swapping seat reservations or convincing passengers to disembark

at a station which is not their ultimate destination). Whilst GPS does not form part of any safety-critical process in ERTMS, it has applications in the driver's cab, for example the Driver Advisory System (DAS). This is used by train operators to manage train and driver performance. This component, represented in the model as the *DAS Display*, can be used by the train driver to determine if the train is running to schedule. If the GPS was affected, it might influence the driver to undertake an alternative driving style which could have a knock-on effect on following trains and route (track path) planning.

7.5.2 Defining the Target Assets

For the analysis of the model to be meaningful, target assets were chosen as systems which either have high safety requirements or systems that, if compromised, have the potential for significant disruption. A prime example of a target asset is the EVC which supervises the train operations. If a balise provided inaccurate information, the EVC would relay this to the RBC, which could return an unsafe decision. As the EVC is responsible for train safety supervision, the malicious data from the balise could, for example, lead to an exaggerated permissible line speed being presented to the driver, with potentially serious consequences.

The Car BUS is another target of interest to an adversary. This runs along the length of the train and, in modern rolling stock, carries a range of data (e.g. power, braking and door operation commands). If an adversary was able to insert arbitrary data into the bus, they would be able to control a number of train functions.

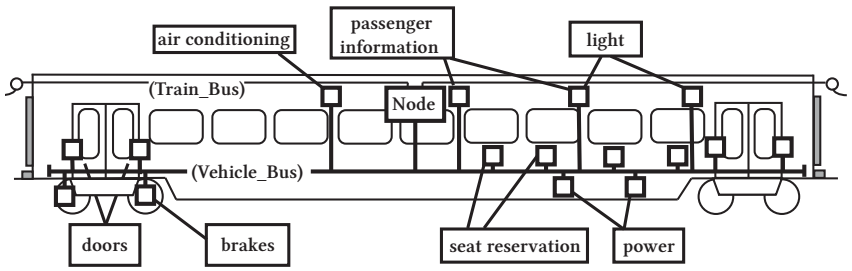


Figure 7.3: Train Car BUS based on [83].

Whilst the DAS and passenger information displays have no safety-requirement and are considered SIL0 systems, used only as aids for drivers and passengers, they still have the ability to cause disruption to passengers and the rail network. The RBC, however, defines the safety-critical decisions through Movement Authorities. Compromising this would seriously impact the safe operation of the railway.

7.5.3 Results

Running the SCEPTICS tool using the model shown in Figure 7.2, with the assets and entry points available to the adversary as discussed in this section, we obtain the following results:

```
Adversary a1:
[Balise]->[Balise transmission module]->[EVC]:0.32736
[Balise]->[Balise transmission module]->[EVC]->[GSM-R]->[GSM-R base]->[MSC]->[RBC]:0.00016
[Balise]->[Balise transmission module]->[EVC]->[GSM-R]->[GSM-R base]->[MSC]->[RBC]->[MSC]->
    ->[GSM-R base]->[GSM-R]->[EVC]:0.32747
[SecureBalise]->[Balise transmission module]->[EVC]:0.06591
[SecureBalise]->[Balise transmission module]->[EVC]->[GSM-R]->[GSM-R base]->[MSC]->
    ->[RBC]:0.00006
[SecureBalise]->[Balise transmission module]->[EVC]->[GSM-R]->[GSM-R base]->[MSC]->
    ->[RBC]->[MSC]->[GSM-R base]->[GSM-R]->[EVC]:0.06597

Adversary a2:
[Workstation1]->[Train vendor]->[Internet Gateway]->[WiFi]->[Car BUS]:0.13456
[Workstation1]->[Train vendor]->[Internet Gateway]->[GSM]->[Car BUS]:0.03275
[Workstation2]->[Train vendor]->[Internet Gateway]->[WiFi]->[Car BUS]:0.0313
[Workstation2]->[Train vendor]->[Internet Gateway]->[GSM]->[Car BUS]:0.00762

Adversary a3:
[GPS]->[DAS Display]:0.97222
[GPS]->[Car BUS]->[Passenger display]: 0.9452
[GPS]->[Car BUS]->[EVC]->[GSM-R]->[GSM-R base]->[MSC]->[RBC]:0.00085
```

As a reminder, the probabilities shown in these results represent the likelihood that the chain may be successfully exploited, i.e. the adversary may successfully breach the starting node all the way through the system to the 'critical asset'. A high probability returned by the tool would indicate that an attack is more likely to succeed than not and, as a result, should be considered a priority for improvement by asset owners.

From this run we observe that, for Adversary $a1$, an adversary who uses the balise as their entry point, has a probability of 0.32736 of being successful in affecting the EVC, potentially accepting invalid line speeds or train location data. It should be noted, however, that the RBC will receive this data with a probability of 0.00016, but the return path from the RBC, where decisions are made, have a much higher probability. This is because the balise data is read into the EVC and reported to the RBC where it will convert (i.e. bridge) the location into command data, returning it to the EVC. Existing tools would terminate their analysis at the RBC as they are unable to support nodes with multiple datatypes and conversions, whereas this tool is able to find this path by following *bridges*.

Adversary $a2$, however, provides some interesting results for review. In the case of the compromised workstation, $Workstation1$, it predictably has a higher probability of affecting the Car BUS than the 'secure' workstation, $Workstation2$. What is not captured here is a malicious employee using $Workstation2$, who could remotely manage the train, highlighting the need for the implementation of appropriate security practices to minimise exposure.

Finally, Adversary $a3$ demonstrates how systems which have high reliance on the data offered by GPS, a system which has known, successful exploits [127], are affected more than systems which do not have any reliance on GPS, such as the RBC. The on-board systems may be affected (e.g. the passenger information displays) but, as the RBC does not rely on GPS data, we note that the likelihood of the RBC being affected is negligible. This can be likened to an attack which uses the Car BUS to exploit some other vulnerability to reach the RBC.

7.5.4 Allowing Asset Owners to Test Security Strategies

As we identified in the previous section in this chapter, the balise has a high likelihood of coercing the RBC to carry out some form of action. A probability of 10^{-6} in the space of a year is generally considered to be the safety limit, where safety-critical systems should not exceed this likelihood [147]. The value of preventing a fatality (VPF) in the rail sector is defined as £1,946,000[1], which could motivate decisions to improve safety and, in this context of this chapter, security. The use of this tool allows asset owners to simulate and prototype improvements to component security, prior to carrying out potentially costly proof-of-concept exercises. As an example, if we were to replace the *Balise* from Figure 7.2 with a new type of balise, *SecureBalise*, capable of adding a keyed MAC to its payload, we observe the improvements delivered through the simple addition of a MAC to the balise payload. What this MAC achieves is that it prevents an attacker from creating their own malicious payload as they do not know the balise-derived MAC key (as outlined in Chapter 6), which reduces the overall path likelihood of success from 0.32736 (reaching the EVC only) to 0.06591. This demonstrates how an asset owner can improve the overall system security and reduce their exposure through minor changes. It should be noted, however, that alternative changes, for example modifying balises to communicate in a similar way to ETCS Level 1 (where the balise is essentially connected to the RBC) could be tested in this model, but in reality, would be impractical and incur significant expense to deploy.

7.6 Tool Discussion

The SCEPTICS tool leverages the CVSS framework to provide a contextual and, more importantly, a verifiable and repeatable means to calculate the probability of an asset becoming compromised. However, it is important to note that the results it provides are a 'snapshot-

[1] https://www.rssb.co.uk/risk-analysis-and-safety-reporting/risk-analysis/taking-safe-decisions/taking-safe-decisions-safety-related-cba

in-time' view of the system under assessment from the time the model has been created. The model should be developed as part of an iterative process where as many insights, subject and domain-specific knowledge and information as possible, are captured over time. It is essential that the model is constantly updated and revised when assets are changed and introduced to the architecture, or where modifications/mitigations have been made which may affect the exposure of the identified high-risk assets.

From the perspective of NIS compliance, the SCEPTICS tool provides a new capability to asset owners that allows them to assess the security of their infrastructure, whilst being able to reason, with relative ease, its posterity from an adversary. One further benefit the tool delivers is its ability to find intrinsic links which may not have been clear to the asset owner, or simply not appreciated, which would be of value to the adversary.

The tool does, however, rely on the competency and skillset of the asset owner to be able to derive accurate and correct CVSS values and maintain its currency. As a 'first-steps' practice to understand the security of a given architecture, the tool allows the asset owner to carry out assessments of their infrastructure, which can, in future applications, validate identified improvements.

7.7 Chapter Summary

In this chapter, we have explored a means for asset owners to assess the security of their infrastructures and carry out a 'first-steps' remediation through the SCEPTICS tool and methodology. This is achieved through a probabilistic analysis, path discovery and application of the CVSS framework, to provide insights that are not easy to extract from complex ICS architectures. Whilst this chapter has primarily focused on applying the tool to ERTMS, its generic view of components and interconnectivity means that it can be applied to other sectors (e.g. pure-IT infrastructures) where asset owners can confidently reason about the security of their systems, whilst delivering assurances. The analyses offered by the tool are, therefore, non-

exhaustive and allow asset owners to set the level of granularity within the model to more precise detail, which is beneficial for systems built with high levels of interconnectivity and dependency.

Future work and enhancements to the tool are reviewed in Chapter 8, where we will consider the benefits of the tool and how other considerations, which might factor in decision making by the asset owner, can be implemented in the tool. We also need to understand some of the barriers to improvements, due to the need for intercompatibility, and how these barriers may be overcome to provide future security assurances for ERTMS.

Part III

Looking to the Future and Closing Statements

Chapter 8

Balancing Safety and Security

The value of carrying out security analysis on the ERTMS standard has been demonstrated throughout this thesis and how the notion of a 'secure architecture' can be developed to identify components in the architecture which can impact its security. From Definition 1.1 given in Chapter 1, we have now established that, with minor changes to the standards, future developments can be assured for both safety and security, whilst also considering threats that may become possible. In this chapter, we will discuss how this is specifically achieved, in addition to other challenges that exist, which may become future work.

8.1 Assessing and Assuring the Secure Architecture for ERTMS

As identified in this thesis, an analysis of the standards from both a safety *and* security perspective should provide the appropriate assurances that there are no weaknesses which may present a threat to the safe operation of the system. As we recall from Definition 1.1, we set out a number of criteria to which components in an architecture must conform. Let us now consider how each chapter in this thesis has contributed towards a secure architecture for ERTMS.

In Chapter 4, we assessed the security of two protocols used in ERTMS, EuroRadio and the Application Layer, which carry out safety-critical functions and identified some potential exploit vectors which could undermine the integrity of ERTMS.

The analysis of the EuroRadio MAC algorithm in Chapter 5 highlighted how previous justifications for choosing a 3DES-based scheme no longer hold. In the future, it would therefore be possible to compromise the security of the MAC, allowing an adversary to affect the safe operation of the railway. Finally, in Chapter 6, we used this Definition to consider future threats that may apply to any system, specifically from a post-quantum attacker who could have undermined the proposed ERTMS online key management scheme, an issue that is addressed by introducing TRAKS. Through TRAKS, weaknesses were identified in the integrity of ERTMS where the balise, which is currently unprotected, can have significant consequences on the safe operation of the railway should it be compromised.

Focusing on specific aspects of the standards, in the wider context of ERTMS, through modelling, we were able to assess the entire architecture as part of a 'bigger picture', where it was possible to identify potential risks that would affect its compliance to the Definition. By modelling the system, as outlined in Chapter 7, we can consider this 'big picture' and analyse, at a high-level, the security of the overall architecture of any system.

What is evident from this analysis is that, in its development, there was a primary focus on ensuring the safety of ERTMS and that its introduction did not present any risk that had not already been addressed in the systems it replaced. Security had been considered, but not to the same extent as safety. As the security landscape is constantly evolving, it is essential that regular reviews are carried out to assure the continued security of the platform, compared to safety case assurances which are generally not reviewed as regularly. However, it is important to state that safety and security are intrinsically linked where, through guarantees of the security of a system, we can provide an assurance towards its safety.

Furthermore, this thesis has primarily considered a reference architecture of ERTMS. As

real-world deployments may not necessarily follow the reference architecture, it is essential that all deployments are assured to prevent exploitation through implementation errors.

8.2 Future Threats to ERTMS

The ERTMS standards include safety-critical and interoperability requirements, but are not regularly updated to address any changes in the security landscape.

Any changes to the standards may require significant effort to retest and to ensure that they continue to provide safety assurances. A frequent review of the standards from both a safety *and* security perspective should, therefore, be considered a necessity and assessments should be carried out as part of the introduction of new baseline standards.

Looking to the future, the gap between the standards and current security practices presents an even greater concern, given that the evolving deployment of ERTMS also increases the attack surface available to an adversary. When we consider the NIS Directive, it is ambiguous where responsibility lies should an attack be leveraged against the standards. As a result, any breaches to the thresholds, irrespective of the root cause, currently become the responsibility of the operators. If an attack was successfully carried out, the operators must report and manage the incident. Using the example of a denial of service attack by replaying messages, causing a train to stop, for example, at Clapham Junction, it would be the responsibility of the train operator to report and manage the incident, even if it was a consequence of a flaw in the underlying standards.

This also raises concerns for the wider ICS sector who, believing in the security provided by compliance to their industry standards, may be similarly exposed due to inherent flaws in the underlying standards. This is particularly an issue for older protocols which, when developed, could not have forseen the future security landscape.

Other forms of exposure may be introduced to ERTMS deployments through the specific implementations used, e.g. where additional or vendor-specific functionality has been im-

plemented outside of the standard. Vendor assurances must, therefore, be provided which additionally benefits compliance to the NIS Directive.

In these implementations and in the development of future standards, one attractive cost-effective solution available to system designers may be to reuse components which have already been assured, but applying them in subtly different areas which may introduce an unappraised risk. One example may be the reuse of the EuroRadio MAC algorithm to validate messages sent between the RBC and Rail Operation Centre (ROC), where a higher volume of messages might allow an adversary to recover the key in a smaller timeframe, as shown in Figure 5.6 in Chapter 5. By reusing an algorithm shown, in Chapter 5, to have flaws, there is an increased risk of an exploit being successful.

8.3 The Need for Interoperability and Intercompatibility

Throughout this thesis, where issues were found in the standards, a number of solutions were proposed. In Chapter 5, we could have suggested a merger of the EuroRadio and Application Layers as a primary recommendation. This, however, would have affected intercompatibility with the existing standards, where we must consider existing deployments and the disruptive effect of any changes to the standards. As a result, whilst we could aim to merge these two layers, an alternative and more cost-effective solution, e.g. adding alternative Safety Features, could be applied which maintains compatibility with the current standards, or requires minor changes which ensure backwards compatibility. This is a more appropriate proposal as it can be implemented without any conflict with the existing standards.

During the development of TRAKS discussed in Chapter 6, we could, again, have proposed a completely new key management architecture for ERTMS. However, as large-scale deployment is already in progress and, given the timescale to certify and ratify any proposed changes to the standards, its applicability would be significantly reduced. From informal discussions with asset owners who have active ERTMS deployments, the introduction of major changes to

the standards should be kept to a minimum, due to the inherent cost of upgrading and replacing equipment which conforms to the existing versions of the standard. In any event, there is also limited ability to deviate from the standards, as Bloomfield et al. note [19].

ERTMS, as a collection of standards, recognises that, as changes are made, they will require potentially significant re-engineering for each incremental version. These versions could, however, include changes which significantly improve the security of the deployment. The current standards require the past two versions (versions $n-2$ and $n-1$) relative to the current version to be supported [57]. The standard, SUBSET-104, however, does not specify the timeframes on the National Infrastructure Managers to upgrade to a newer version (e.g. version $n-2$ to $n-1$). If we consider commodity systems, this approach would be unacceptable, where patching is carried out on a regular basis. ERTMS, however, is not a commodity system. It requires rigorous safety validation which, due to its dynamic nature, can delay the implementation and upgrade to conform to the latest standards [136].

8.4 Limitations and Methodology Review

One issue which has not been addressed in this thesis are any limitations of the analyses conducted. Whilst the standards may be assured for security and safety, the implementation can be the source of weaknesses. These weaknesses can be introduced at any point in the supply chain or through lack of maintenance. For example, a coding error when defining the EuroRadio protocol may introduce unintended state transitions which the standard does not support, or malware similar to that of Stuxnet and BlackEnergy may remain unpatched. Similarly, a system which conformed exactly to the specification when built, but uses a library (e.g. OpenSSL), could introduce a newly-discovered vulnerability which, if not patched, would provide another, potentially easier, vector which the adversary could use as a pivot point.

Assuring the security of these systems and identifying ways to prevent stagnation is, therefore, also critical. Considering the NIS Directive, there is now a requirement for both vendors

and operators to ensure compliance and that their systems do not have any potential exposures. Failure to meet the requirements of the Directive could subsequently result in a financial penalty. Assuring these implementations means that we must have confidence in what can otherwise be considered as 'black boxes', which have the potential to be exploited with possibly serious consequences.

8.4.1 Limitations of the Research

From Chapter 4, we assured the EuroRadio protocol but not the implementation of the RBC-RBC handover protocol. This protocol is not defined in as much detail as EuroRadio, but has been analysed from a liveness perspective [39]. Under ERTMS, there should be an intercompatible protocol between vendors but its lack of detail means that further analysis is required. The modelling captured in this chapter only considers three messages of a specific priority being sent between the train and RBC. Further work could assess a more realistic model and consider an arbitrary number of messages between the train and trackside, some normal-priority, others high-priority. This modelling also does not consider denial of service attacks, where alternative assurance methods could be applied. In Chapter 5, a cryptography-focused variant of ProVerif, CryptoVerif [15] could have been applied to complement the model of the EuroRadio and Application Protocols as an alternative.

As previously identified in Chapter 6, we can provide authentication of the payloads presented by balises to a train. What this does not protect against, however, is a physical attacker who either removes a group of balises (which might cause a train to become stranded if its Movement Authority depends on passing over that group), or 'shuffles' a set of balises. The adversary could also use the tolerances in distance measurements between groups of balises to potentially, on a large scale, create their own overlapping block. The RBC (in ETCS Level 3) would believe that no safety margins are being violated but, by creating a 'ghost' block, two trains could be made to occupy the same piece of track. A further consequence is that the

RBC could be convinced that a set of points is now clear, which could either cause derailment or, even worse, a collision. Currently, the standards specify the inclusion of the 'last relevant balise group' in a balise telegram. However, it is not clear how these identities are specified, e.g. when crossing a set of points. By chaining the balises together, it is possible to anticipate the identities of the next balises, preventing an adversary from switching in a balise which has a different ID.

In Definition 1.1, we noted that protocols should be verified for security. In Chapter 6, the proposed TRAKS Key Distribution Protocol could be modelled using ProVerif and other tools to assure the security of the keys exchanged between the train, vendor and KMC. One issue here is that the without an understanding of the internal vendor processes for how such cryptographic keys are handled, abstractions of the vendor processes would need to be made. This, however, means that full assurance cannot be given as the vendor could have poor key handling practices which would undermine the scheme.

Another part of the standard that has not yet been explored is the management of Packet 44 and what trust boundaries can be crossed if this was compromised. As previously stated, in Great Britain, Packet 44 is currently used for Tilting Authorisation and Speed Supervision (TASS) telegrams, authorising the train to tilt on the West Coast Main Line. In 2014, Network Rail [20] proposed using Packet 44 for other applications, for example, Automatic Train Operation (ATO) on the Thameslink route. Thameslink's 'core' is fitted with ERTMS Level 2, with ATO, enabling an increase in line capacity. In November 2017, it was confirmed that the scope of Packet 44 was extended for ATO operations on the Siemens Class 700 rolling stock for Thameslink [78]. However, the mismanagement or incorrect implementation of Packet 44, for example, invalid calculations of variables, could have potentially serious consequences. For tilting authorities, RSSB provides the requisite calculations for 'Enhanced Permissible Speed', higher line speeds at which a tilting train can travel [124], but with limited verification onboard the train, this may be an aspect that warrants further research.

Asset owners may have varying priorities about where investment and improvements are made to their systems. The SCEPTICS tool from Chapter 7 provides a 'first steps' approach to reasoning the security of architectures, but there are further enhancements which could be made to increase its effectiveness. Some asset owners who are implementing ISO/IEC 62443 are using Security Levels to determine those sensitive components of their systems. Here, the tool could highlight where possible thresholds could be breached. Another consideration is that, when a given path has a high likelihood of being successfully exploited, it may be costly to mitigate or resolve. If we consider the econometrics of implementing changes, we may find that a cheaper solution, applied on a different node, may have a greater impact at lower cost.

Definition 1.1 is also not exhaustive and further case studies by applying it to alternative ICS environments would further benefit its efficacy. By taking the 'lessons learnt' from this thesis and applying them to other sectors, we can provide a framework which can be used by ICS asset owners. This is particularly relevant when we consider the impact that the NIS Directive has already had in forcing asset owners to consider new approaches to implement its requirements.

8.4.2 Methodology Review

The approach of reviewing the set of standards that form ERTMS and mapping them out as a system architecture proved key to the analyses conducted in this thesis, where a detailed understanding of the subtle links between systems is critical. There is, however, the risk of becoming too focused on specific aspects of the standards without considering the wider impact on neighbouring systems. By mapping out these systems, this risk is reduced as all analyses can be carried out in context of the overall system and understanding the wider consequences of any issues found. Through the detailed understanding of the standards, conflicts and subtle differences between standards documents can be identified and further investigated from a security perspective.

As previously mentioned in Chapter 2, a number of different analysis techniques exist to analyse various aspects of the standards. As an example in Chapter 4, an alternative toolset, e.g. TAMARIN, could have been used to provide similar results.

One of the issues that arose during the development of the ProVerif model in Chapter 4 was non-termination due to the initial modelling of timestamps and infinite messages between the train and RBC. This was resolved through the creation of the lightweight notion of time and modelling only three messages exchanged between the train and RBC. This proved sufficient to provide security assurances of the EuroRadio and Application Layer protocols.

8.5 Future Research Directions

Whilst this thesis has primarily focused on the standards and system architectures, there is a reliance on the supply chain to correctly implement the standards. Under the NIS Directive, the responsibility for compliance lies with the asset owner, who may typically consider these components to be black boxes. It is often difficult for asset owners to verify the security of these systems, as the source code and system designs are not typically available. Therefore, an approach similar to that developed in this thesis with a more practical emphasis can be undertaken to consider the security of industrial (and rail) components. This would allow asset owners to ease the challenge of assurance to achieve NIS compliance.

A number of analysis techniques can be surveyed and compared, allowing ICS owners to review potential threats that their systems could face and provide guidance on the most appropriate and effective methods that can be applied. As highlighted in Chapter 7, a knowledge and skills gap exists when considering the security of systems. For most asset owners who lack this understanding, we can look at how existing vulnerabilities to ICS systems arose, whether it was through implementation or lack of clarity in the standards. By applying the methodology developed in this thesis and reviewing methods to assure the security of these black boxes, we can obtain an understanding of the link between standards and implementation.

Whilst this thesis has primarily considered ERTMS, further analyses on Communication Based Train Control (CBTC)-based systems is also warranted. Where ERTMS is specified as a set of open standards for implementation by vendors, CBTC-based systems can be wholly proprietary, particularly due to their target market for metro and light-rail operators, such as Transport for London. In ERTMS, only vendor-supplied systems, for example the RBC and Key Management Centre may contain proprietary software. Only the external interfaces (e.g. train to RBC and RBC handover protocols) must conform to a common specification. As an example, a programming interface for a balise can be proprietary, provided the balise transmits a valid telegram to a train passing over it.

Another area which should be further explored is the security of onboard systems, particularly at the threshold between the ERTMS systems and the proprietary, on-train, systems. Here, there is an explicit trust boundary placed between two sets of hardware. Investigation into the adversary's capabilities, should one side of the trust boundary become compromised, would be of further interest.

Exploring the link between safety and security is also an area that should be researched further. Striking the right balance is essential, where further analysis can identify and define solutions to maintain safe and secure systems for the future.

Finally, modelling and considering vendor influences on standards is also of interest, as the rail supply chain can have a large influence on the development of standards. Modelling the process for the creation of, and changes to, a standard would also be valuable in understanding the influence a vendor may have on the full set of standards.

Chapter 9

Conclusion

At the start of this thesis, we established a definition of a secure architecture to which an industrial control system should conform. Taking the ERTMS standards as a primary focus of this thesis, we find that this definition does not hold when the standards are presented with a malicious adversary. However, using the recommendations proposed in this thesis, it is possible to ensure that the ERTMS architecture is secure.

When we consider the EuroRadio and Application Layers, other assurance work to date has revolved around the safety offered by these protocols and very little in the way of security analysis. This ultimately led to the accepted approach of 'when in doubt, stop the train', where the network fails safe. Through the formal analysis of the EuroRadio and Application Layers, we find that, by applying the definition of a secure architecture, these protocols do not hold. Specifically, due to the way that the EuroRadio layer allows unauthenticated emergency stop messages to be sent, an adversary who is able to inject into the GSM-R data stream would have the potential for mass disruption on the network. When evaluating the possible methods of addressing these issues, it is essential that the safety of the railway is not compromised, whilst considering a balance of proposed security improvements to the existing standard.

From the analysis above, we were introduced to the MAC scheme used by EuroRadio to

authenticate messages between the train and trackside, a custom 3DES-based scheme which was first proposed in 1997. As it was not previously subject to detailed security assurance, in the analysis of the scheme, we identified a collision-based attack such that, if an attacker could capture sufficient data, it would be possible to forge valid train control messages. The design limitations of 1997 are no longer a barrier with the advances in technology and cryptography available today. Alternative schemes could, therefore, be employed which do not introduce the weaknesses identified in this thesis.

We then considered how key management takes place in ERTMS, introducing a new post-quantum key management scheme, TRAKS, which allowed other components, for example the EuroBalise, to be introduced into the secure architecture. As previously highlighted, the operational burden that the current scheme places on National Infrastructure Managers and operators is significant, leading to poor practices which undermine the security of the platform. Whilst there are improvements in its proposed, online, successor, it is still not capable of resisting a post-quantum adversary.

Finally, in light of the challenges faced by asset owners through the implementation of the NIS Directive, where limited resource may be available to assess system security, we developed a graph-based modelling approach to carry out initial security assessments of system architectures. The value of these assessments allows the asset owner to understand the interconnectivity of their systems and the true security of their architectures without requiring expert security knowledge.

By applying these various analysis and assurance methods, a number of recommendations were made to improve the future security of ERTMS, which are summarised in Appendix F.

Whilst all the analyses presented in this thesis provides a level of assurance from the perspective of the standards, vulnerabilities may still exist in their implementations. Complementary, alternative assessments (e.g. penetration testing and fuzzing) should, therefore, also be conducted to provide an end-to-end (from standard to operational use) assurance.

A recurring theme of this thesis are the barriers to change that exist in ICS deployments. When security issues are identified, their risk would be assessed and any solutions would have to be compatible with the existing standards to provide backwards compatibility and not impact the safe operation of the railway. We also had to consider that any proposed solutions must remain viable for the lifespan of ERTMS. We have also observed how the standards can lack currency. This will ultimately impact implementations – if the standard is not secure, how can the implementations be secure? Since 2007, when SUBSET-026, which defined the system specification of ERTMS was first released, very little appears to have changed to improve the security of the platform.

We have previously established that safety and security are intrinsically linked. They are no longer considered independent – security influences safety and, without sufficient security assessments, safety cannot be assured.

It is, therefore, imperative that whilst safety is core to the design process of safety-critical standards, security must be given equal priority.

Appendices

Appendix A

Formal Verification of the High-Priority EuroRadio Message

A.1 High-Priority Message Model (Train)

```
1  let Train =
2    new session;
3    (* Receive an RBC id to communicate with *)
4    in(id, rbc_etcs_id);
5    (* TCONN.request  Au1 SaPDU *)
6    new trainNonce;
7    event trainStartSession(rbc_etcs_id, train_etcs_id, trainNonce, SAF);
8    out(c, (TRAIN_ETCS_ID_TYPE, AU1, DF_SEND, train_etcs_id, SAF, trainNonce));
9    (* TCONN.confirmation  Au2 SaPDU *)
10   in(c, (=RBC_ETCS_ID_TYPE, =AU2, =DF_RESP, in_rbc_etcs_id, rbcSaF, rbcNonce, inMAC));
11   (* Generate the session key *)
12   let trainKS = genSessionKey(trainNonce, rbcNonce, getKey(in_rbc_etcs_id, train_etcs_id)) in
13   (* Output encrypted secret to check secrecy of keys *)
14   out(c, encrypt(SECRET, trainKS));
15   out(c, encrypt(SECRET, getKey(in_rbc_etcs_id, train_etcs_id)));
16   if inMAC = mac(trainKS, ((PAYLOAD_LENGTH, train_etcs_id, RBC_ETCS_ID_TYPE, AU2,
         DF_RESP, in_rbc_etcs_id, rbcSaF), rbcNonce, trainNonce, train_etcs_id)) then
17   (* TDATA.request  Au3 SaPDU *)
18   event trainFinishSession(in_rbc_etcs_id, train_etcs_id, trainNonce, rbcSaF, rbcNonce, trainKS);
19   out(c,(ZEROS, AU3, DF_SEND, mac(trainKS, (PAYLOAD_LENGTH, train_etcs_id, ZEROS, AU3,
         DF_SEND, trainNonce, rbcNonce))));
20
```

Figure A.1: The ProVerif model of High-Priority Messages in EuroRadio from the perspective of a train.

A.2 High-Priority Message Model (RBC)

```
1  let RBC =
2    (* Get an RBC identiy *)
3    in(id, rbc_etcs_id);
4    (* TCONN.indication  Au1 SaPDU *)
5    new rbcNonce;
6    in(c, (sent_ETCS_ID_TYPE, =AU1, =DF_SEND, in_train_etcs_id, trainSaF, trainNonce));
7    event rbcStartSession(rbc_etcs_id, in_train_etcs_id, rbcNonce, trainSaF, trainNonce);
8    (* Generate the session key *)
9    let rbcKS = genSessionKey(trainNonce, rbcNonce, getKey(rbc_etcs_id, in_train_etcs_id)) in
10   (* Output encrypted secret to check secrecy of keys *)
11   out(c, encrypt(SECRET, rbcKS));
12   out(c, encrypt(SECRET, getKey(rbc_etcs_id, in_train_etcs_id)));
13   (* TCONN.response  Au2 SaPDU *)
14   out(c, (RBC_ETCS_ID_TYPE, AU2, DF_RESP, rbc_etcs_id, trainSaF, rbcNonce, mac(rbcKS, ((
         PAYLOAD_LENGTH, in_train_etcs_id, RBC_ETCS_ID_TYPE, AU2, DF_RESP, rbc_etcs_id,
         trainSaF), rbcNonce, trainNonce, in_train_etcs_id))));
15   (* AU3 SaPDU *)
16   in(c,(=ZEROS, =AU3, =DF_SEND, inMAC));
17   if inMAC = mac(rbcKS, (PAYLOAD_LENGTH, in_train_etcs_id, ZEROS, AU3, DF_SEND ,
         trainNonce, rbcNonce)) then
18   event rbcFinishSession(rbc_etcs_id, in_train_etcs_id, rbcNonce, trainSaF, trainNonce, rbcKS);
19
```

Figure A.2: The ProVerif model of High-Priority Messages in EuroRadio from the perspective of a RBC.

Appendix B

EuroRadio MAC Collisions

Following the analysis of the EuroRadio MAC and its susceptibility to collisions, 8 separate collisions were found in the Acknowledgement Message (Message 146) that is sent by a train in response to a message sent by the RBC. This collision analysis only took into account the intermediate MAC value (i.e. H_n), which is sufficient for collision detection, given the deterministic nature of the 3DES transformation in the final block. The MAC and corresponding two messages that resulted in that collision are given in the following table.

Intermediate MAC (H_n)	Plaintexts
365CA0E4D4901E85	00120000020A9203A2105E0480000062105DFF8000000000 00120000020A9203AAE360078000006AE360028000000000
410F1B9C2C09E958	00120000020A9203970598C5C00000570598C34000000000 00120000020A9203B04EA8D7C00000704EA8D54000000000
4BBDFBABD9757A38	00120000020A9203A9D9B5FDC0000069D9B5FB4000000000 00120000020A9203AC38CEEA8000006C38CEE58000000000
7A3D01D36BE88B21	00120000020A920385CCD6F280000045CCD6EB0000000000 00120000020A920386E4CFBCC0000046E4CFB7C000000000
80B7557F31566DBB	00120000020A9203A2105E0480000062105DFD0000000000 00120000020A9203AAE360078000006AE360000000000000
A7A3AD4FA4C6D433	00120000020A9203A16580E0400000616580DDC000000000 00120000020A9203A34C8FAF400000634C8FAA4000000000
BE23849D77705C72	00120000020A920398952D5AC0000058952D534000000000 00120000020A9203B553FC648000007553FC5D0000000000
F813AED5FE3D445F	00120000020A9203A16580E0400000616580DB4000000000 00120000020A9203A34C8FAF400000634C8FACC000000000

Table B.1: Pairs of messages which result in the same MAC under key k_1 = 01020407080B0D0E

Appendix C

MAC Collision Benchmarking

To understand the real-world and current capabilities of an adversary, an Amazon EC2 instance was used to perform the `hashcat` benchmarking, using a `p2.16xlarge` instance. As mentioned in Chapter 5, these instances are now deprecated in favour of more powerful instances, which may reduce the time, effort and cost required to carry out the attack. These instances are designed for high-performance Graphics Processing Unit (GPU) computation [2], costing $14.40 per hour to use. Each instance comes with 64 vCPUs, 734GB of local RAM, and, for the purposes of this experiment, a 8GB SSD-backed storage facility, available on the same network as the instance to reduce latency.

Each `p2` EC2 instance, at the time, was fitted with an NVIDIA Tesla K80 GPU, which has 5,000 CUDA cores, 24GB of GDDR5 RAM, and, for the `p2.16xlarge` instance used for this experiment, 16 K80 GPUs were available. At the time of benchmarking, the latest NVIDIA GPU drivers, the Amazon Linux image and `hashcat` source code was compiled and installed, optimised for this system.

`hashcat` is optimised for OpenCL, a framework that allows GPUs to be leveraged for computation-heavy processes where, using the system GPUs, we obtain the following results.

`hashcat` Benchmarking Results

`hashcat` offers a benchmarking mode, allowing it to state the number of hashes, or values it is able to produce per second. The argument set and results are broken down:

- `-m 1500` : Message Type: descrypt, DES(Unix), Traditional DES
- `-b` : Benchmark Mode
- `-w 4` : Workload Profile 4 – Extreme
- `--powertune-enable` : Enable automatic power tuning option on GPU

The EuroRadio MAC algorithm is not available to `hashcat`, where the closest family to the simple DES algorithm available is the *descrypt* message type. This message type, in reality, takes a 56-bit key as a password and a 64-bit zeroed data input block and encrypts this block 25 times, where the hash is the output of the process. For the output speed stated from the `p2.16xlarge` instance, we *gain* a 25x factor improvement, due to the *descrypt* function carrying out 25 rounds of DES encryptions.

[2] `https://aws.amazon.com/ec2/details/`

```
$ hashcat -m 1500 -b -w 4 --powertune-enable
hashcat (v3.10) starting in benchmark-mode...

OpenCL Platform #1: NVIDIA Corporation
======================================
- Device #1: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #2: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #3: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #4: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #5: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #6: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #7: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #8: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #9: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #10: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #11: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #12: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #13: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #14: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #15: Tesla K80, 2859/11439 MB allocatable, 13MCU
- Device #16: Tesla K80, 2859/11439 MB allocatable, 13MCU

Hashtype: descrypt, DES(Unix), Traditional DES

Speed.Dev.#1:   176.5 MH/s (482.39ms)
Speed.Dev.#2:   174.6 MH/s (482.68ms)
Speed.Dev.#3:   176.2 MH/s (483.11ms)
Speed.Dev.#4:   175.4 MH/s (485.09ms)
Speed.Dev.#5:   174.4 MH/s (483.27ms)
Speed.Dev.#6:   175.3 MH/s (480.53ms)
Speed.Dev.#7:   175.9 MH/s (483.79ms)
Speed.Dev.#8:   175.9 MH/s (483.84ms)
Speed.Dev.#9:   175.1 MH/s (481.23ms)
Speed.Dev.#10:   177.5 MH/s (479.46ms)
Speed.Dev.#11:   177.4 MH/s (479.99ms)
Speed.Dev.#12:   174.8 MH/s (486.82ms)
Speed.Dev.#13:   177.6 MH/s (484.14ms)
Speed.Dev.#14:   175.8 MH/s (484.19ms)
Speed.Dev.#15:   176.8 MH/s (481.54ms)
Speed.Dev.#16:   175.0 MH/s (481.54ms)
Speed.Dev.#*.:  2814.1 MH/s
```

Finally, we can estimate the cost of breaking the EuroRadio MAC using these instances. Each `p2.16xlarge` instance is capable of producing 2,814,100,000 'outputs' per second. As identified earlier, this involves 25 DES rounds, where the per-DES round speed is actually 70,352,500,000 'outputs' per second. The time taken to break DES using a single instance can

be calculated using the below equation:

$$\frac{2^{56}}{(70,352,500,000 * 60 * 60)} \approx 284\ hours$$

If we were to use, say, 400 `p2.16xlarge` instances, the time required would reduce to $\sim$ 42 hours.

For EuroRadio, where at least 6 DES encryptions are used for a 32 byte message (where 3 DES encryptions are used, totalling 6 to test the key), the equation shown below gives us a time of 30 minutes, using 3,400 EC2 instances in parallel.

$$\frac{6 \cdot 2^{56}}{(70,352,500,000 * 60 * 3400)} \approx 30\ mins$$

As a `p2.16xlarge` instance, at the time, cost $14.40 per hour to rent, using 3,400 instances for 30 minutes to break the EuroRadio MAC would be as little as $48,960.

Appendix D

CVSS Framework Equations

The expressiveness of the CVSS framework allows us to define the security of systems in a meaningful way. In the SCEPTICS tool, it is applied to describe the security of a given component or link. The vector defined by CVSS is transformed into an overall score. We will describe how its function and values may be determined.

The base score is derived as a function of two sub-score equations, which relate to the impact of compromise and the exploitability of the system under assessment. The impact subscore (known as *ISC*) has two definitions – the definition applied depends on whether the scope of the exploit changes as it progresses through the system or, alternatively, where the scope does not place the adversary in a more 'privileged' position. In the former case, the score is defined as $ISC = 6.42 \cdot ISC_{Base}$, where ISC_{Base} is the impact base subscore, defined by Equation D.2. In the latter case, if scope changes, the score is derived in a different way, i.e.

$$ISC = 7.52 \times (ISC_{Base} - 0.029) - 3.25 \times (ISC_{Base} - 0.02)^{15} \tag{D.1}$$

$$ISC_{Base} = 1 - ((1 - Impact_{Confidentiality}) \times (1 - Impact_{Integrity}) \times (1 - Impact_{Availability})) \tag{D.2}$$

The exploitability sub-score (*ESC*) is more straightforward to compute, as defined in Equation D.3. The variables used in this equation and their numeric mappings are given in [65].

$$ESC = 8.22 \times AttackVector \times AttackComplexity \times PrivilegeRequired \times UserInteraction \tag{D.3}$$

In the event that the *ISC* metric is 0, then the base score is 0, otherwise, it is determined through Equation D.4. In this Equation, the function $rup(x)$ rounds its input x up by one decimal place. As an example, $rup(2.42) = 2.5$, $rup(2.9) = 2.9$.

$$BaseScore = \begin{cases} rup(min(1.08 \times (ISC+ESC), 10)), & \textit{IF scope changed} \\ rup(min((ISC+ESC), 10)), & \textit{IF scope unchanged} \end{cases} \tag{D.4}$$

D.1 CVSS Value Conversion

Metric	Metric Value	Numerical Values
(Modified) Attack Vector	Network	0.85
	Adjacent Network	0.62
	Local	0.55
	Physical	0.2
(Modified) Attack Complexity	Low	0.77
	High	0.44
(Modified) Privilege Required	None	0.85
	Low	0.62 (0.68 if Scope/ Modified Scope is changed)
	High	0.27 (0.50 if Scope/ Modified Scope is changed)
(Modified) User Interaction	None	0.85
	Required	0.62

Table D.1: CVSS Metric Values used by the SCEPTICS Tool

Appendix E

SCEPTICS Tool Input XML Definition

E.1 Adversary Definition

```
1  <adversaries>
2    <adversary id="a1">
3      <entry_nodes>
4        <!AND operation>
5        <node>SecureBalise</node>
6      </entry_nodes>
7      <data_types>
8        <type>any</type>
9      </data_types>
10       <link_types>
11         <type>shortrange wireless</type>
12         <!OR operation>
13         <type>longrange wireless</type>
14       </link_types>
15   </adversary>
16 </adversaries>
```

Figure E.1: Input XML Definition for an Adversary, who has as their entry point, a component in the model named *SecureBalise*, and can use their capabilities on specific edge types.

E.2 Asset Definition

```
<assets>
  <asset>
    <node>
      <id>nRBC</id>
    </node>
    <data_types>
      <type>data</type>
    </data_types>
  </asset>
</assets>
```

Figure E.2: Input XML Definition for an Asset (*RBC*) which is a target node, and subject to patient-zero and reachability analysis, with specific profiles and datatypes of interest.

Appendix F

Thesis Recommendations

In this Appendix, a summary list of recommendations made in this thesis are included below for ease of reference.

F.1 Chapter 4: Modelling of the EuroRadio and Application Layer Protocols

Inclusion of a MAC in EuroRadio High-Priority Messages The previous justifications for omitting a MAC on high-priority messages, specifically where speed of computation was of the essence, is no longer valid. Advances in computational power mean that the speed now taken to compute a MAC is no longer a barrier. The inclusion of a MAC would prevent an adversary who is able to inject into the GSM-R data stream from sending arbitrary emergency stop messages without knowing the shared MAC session key.

Addition of a Counter to Messages As the Application Layer relies on timestamps to ensure timely receipt of messages and to honour the latest message received, it is possible for messages to be blocked from reception. By moving towards a sequence counter-like scheme where each message to and from the train and RBC are sequentially numbered, it is possible to determine if a message has not been successfully received, in a TCP-like manner and then appropriately handled.

Verification of the Safety Feature (SaF) in the EuroRadio Handshake As the first message from the train is sent in the clear, it is possible for an adversary to intercept and modify the proposed Safety Feature to one that is considered less secure. By ranking Safety Feature security, if the train, upon receipt of the `AU2` message from the RBC, is instructed to use a different Safety Feature to that sent by the train in the initial message (`AU1`), the train can determine whether or not to proceed with the handshake if a weaker cipher is proposed by the RBC.

F.2 Chapter 5: EuroRadio MAC Cipher

Replacement of the EuroRadio MAC with a Modern and Secure Alternative The existing MAC for EuroRadio was proposed in 1997 [22] with the requirement that a determined adversary would need 'sufficient technical, financial and human resources' to compromise the

scheme. As highlighted in Chapter 5, 3DES is only recommended for use up to 2030. Alternatives should, therefore, be considered which offer sufficient security throughout the anticipated operational lifespan of ERTMS systems. Alternative MAC ciphers, specifically those understood to be quantum-resistant should be considered and made available for use as Safety Features prior to this date. Where train and RBC keys do not have the sufficient length for new ciphers, new keys should be issued that meet the required length, where a portion of the new key can be used for backwards compatibility.

F.3 Chapter 6: ERTMS Key Management

Replacement of the Offline and Online ERTMS Key Management Schemes with TRAKS One critical issue with the existing ERTMS offline key management scheme is its lack of scalability and the significant burden on National Infrastructure Managers during and after national deployments of ERTMS. Whilst the online scheme attempts to alleviate this burden, its reliance on RSA certificates makes the scheme insecure against a post-quantum adversary, where we have to consider the possibility of such adversarial capabilities in the next 10-15 years. The TRAKS Key Management Scheme, whilst being designed to be backwards-compatible with the existing offline scheme, reducing the operational burden on Infrastructure Managers to introduce new trains into their geographic domain, also achieves post-quantum security through its use of post-quantum secure Pseudo-random Functions (PRFs).

Replacement of the Key Issuance Protocol used by Infrastructure Managers Presently, for a new train or RBC, transport keys are issued in plaintext and installed on the train and RBC. An internal adversary could therefore intercept these keys and be able to decrypt all future key management directives issued by the KMC to the train and RBC. Using technologies available today, e.g. the TPM, ARM TrustZone and Intel SGX, the trains and RBCs can securely generate and store their own keys with the vendor simply acting as a communication link to the KMC, thus removing this exposure.

Include Authentication for Trusted Devices EuroBalises (as shown in Chapter 7) can influence decisions made by the RBC. The addition of a MAC, which is verified by a train, would prevent an adversary from attempting to impersonate a balise or reprogramming one without the necessary keys. Such keys can be issued using the TRAKS scheme through the hierarchy defined in Chapter 6.

List of References

[1] M. Abadi and C. Fournet, "Mobile values, new names, and secure communication," in *Symposium on Principles of Programming Languages (POPL)*, 2001.

[2] R. J. Anderson, *Security engineering: a guide to building dependable distributed systems*. John Wiley & Sons, 2010.

[3] Ansaldo STS Group, "Product portfolio and ERTMS/RTCS projects of Ansaldo Segnalamento Ferroviario," 2008. [Online]. Available: http://old.fel.zcu.cz/Data/documents/sem_de_2008/AnsaldoSTS_08.pdf

[4] ANSI, "ANSI X9.19:1998 Financial Institution Retail Message Authentication," ANSI, Tech. Rep., 1998.

[5] I. Arsuaga, N. Toledo, I. Lopez, and M. Aguado, "A Framework for Vulnerability Detection in European Train Control Railway Communications," *Security and Communication Networks*, vol. 2018, 2018. [Online]. Available: https://doi.org/10.1155/2018/5634181

[6] G. Ateniese, M. Steiner, and G. Tsudik, "Authenticated group key agreement and friends," in *Proceedings of the 5th ACM conference on Computer and communications security*. ACM, 1998, pp. 17–26.

[7] G. Ateniese, M. Steiner, and G. Tsudik, "New multiparty authentication services and key agreement protocols," *IEEE journal on selected areas in communications*, vol. 18, no. 4, pp. 628–639, 2000.

[8] M. Avalle, A. Pironti, R. Sisto, and D. Pozza, "The Java SPI framework for security protocol implementation," in *Availability, Reliability and Security (ARES), 2011 Sixth International Conference on*. IEEE, 2011, pp. 746–751.

[9] Banedanmark, "Fjernbane Infrastructure East/West BAFO Tender Document / Appendix 3.1 – Att 11 Online Key Management Concept," Banedanmark, Tech. Rep., 2011. [Online]. Available: http://uk.bane.dk/db/filarkiv/9893/SP-12-041120-103.0111App_3.1_Attachment_11.docx

[10] E. Barkan, E. Biham, and N. Keller, "Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication," in *Advances in Cryptology - CRYPTO 2003*, ser. Lecture Notes in Computer Science, D. Boneh, Ed. Springer Berlin Heidelberg, 2003, vol. 2729, pp. 600–616. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-45146-4_35

[11] A. Beautement, R. Coles, J. Griffin, C. Ioannidis, B. Monahan, D. Pym, A. Sasse, and M. Wonham, "Modelling the human and technological costs and benefits of USB memory stick security," in *Managing Information Risk and the Economics of Security*. Springer, 2009, pp. 141–163.

[12] N. Benaissa, D. Bonvoisin, A. Feliachi, and J. Ordioni, "The PERF Approach for Formal Verification," in *Reliability, Safety, and Security of Railway Systems. Modelling, Analysis, Verification, and Certification*, T. Lecomte, R. Pinger, and A. Romanovsky, Eds. Cham: Springer International Publishing, 2016, pp. 203–214.

[13] G. Biswas, "Diffie-Hellman technique: extended to multiple two-party keys and one multi-party key," *IET Information Security*, vol. 2, no. 1, pp. 12–18, 2008.

[14] B. Blanchet, "An Efficient Cryptographic Protocol Verifier Based on Prolog Rules," in *Computer Security Foundations Workshop (CSFW)*. IEEE, 2001, pp. 82–96.

[15] B. Blanchet, "A computationally sound automatic prover for cryptographic protocols," in *Workshop on the link between formal and computational models*, Paris, France, Jun. 2005.

[16] B. Blanchet and V. Cheval, "ProVerif: Cryptographic protocol verifier in the formal model," 2015. [Online]. Available: http://prosecco.gforge.inria.fr/personal/bblanche/proverif/

[17] B. Blanchet, B. Smyth, and V. Cheval, "ProVerif 1.88: Automatic Cryptographic Protocol Verifier, User Manual and Tutorial," 2013.

[18] R. Bloomfield, R. Bloomfield, I. Gashi, and R. Stroud, *Computer Safety, Reliability, and Security: SAFECOMP 2012 Workshops: Sassur, ASCoMS, DESEC4LCCI, ERCIM/EWICS, IWDE, Magdeburg, Germany, September 25-28, 2012. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, ch. How Secure Is ERTMS?, pp. 247–258. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-33675-1_22

[19] R. Bloomfield, M. Bendele, P. Bishop, R. Stroud, and S. Tonks, *The Risk Assessment of ERTMS-Based Railway Systems from a Cyber Security Perspective: Methodology and Lessons Learned*. Cham: Springer International Publishing, 2016, pp. 3–19. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-33951-1_1

[20] P. D. Booth, "Application of ATO/DAS to Thameslink," 2014. [Online]. Available: http://www.irse.org/knowledge/publicdocuments/08_IRSE_ATO_Seminar_Oct_2014_v5_[Compatibility_Mode].pdf

[21] C. Boyd and A. Mathuria, *Protocols for authentication and key establishment*, ser. Information security and cryptography. Berlin, Heidelberg, New York (N. Y.): Springer, 2003. [Online]. Available: http://opac.inria.fr/record=b1100919

[22] J. Braband, "Safety and Security Requirements for an Advanced Train Control System," in *Safe Comp 97*. Springer, 1997, pp. 111–122.

[23] J. Braband, "Cyber security in railways: Quo vadis?" in *Reliability, Safety, and Security of Railway Systems. Modelling, Analysis, Verification, and Certification*, A. Fantechi,

T. Lecomte, and A. Romanovsky, Eds. Cham: Springer International Publishing, 2017, pp. 3–14.

[24] D. Brandenburg, E. Grünberger, and M. Grandits, "Effective ETCS Key Management with KEY.connect," SIGNAL + DRAHT, Tech. Rep., 2010. [Online]. Available: http://www.tuev-sued.de/uploads/images/1347442379238610830293/article-keyconnect.pdf

[25] E. Bresson, O. Chevassut, and D. Pointcheval, "Dynamic group Diffie-Hellman key exchange under standard assumptions," in *International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2002, pp. 321–336.

[26] E. Bresson, O. Chevassut, D. Pointcheval, and J.-J. Quisquater, "Provably authenticated group Diffie-Hellman key exchange," in *Proceedings of the 8th ACM conference on Computer and Communications Security*. ACM, 2001, pp. 255–264.

[27] N. Breton and Y. Fonteneau, "S3: Proving the safety of critical systems," in *Reliability, Safety, and Security of Railway Systems. Modelling, Analysis, Verification, and Certification*, T. Lecomte, R. Pinger, and A. Romanovsky, Eds. Cham: Springer International Publishing, 2016, pp. 231–242.

[28] R. A. Brualdi, *Introductory Combinatorics (5th ed.)*. Prentice-Hall, 2010.

[29] BSI, "BS EN 50159:2010 - Railway Applications – Communication , signalling and processing systems – Safety-related communication in transmission systems," Tech. Rep., 2010. [Online]. Available: https://bsol.bsigroup.com/Bibliographic/BibliographicInfoData/000000000030202175

[30] BSI, "BS ISO/IEC 8859-1:1998 – Information technology. 8-bit single-byte coded graphic character sets. Latin alphabet No. 1," 2011. [Online]. Available: https://bsol.bsigroup.com/Bibliographic/BibliographicInfoData/000000000001302578

[31] BSI, "BS ISO/IEC 9797-1:2011 – Information technology. Security techniques. Message authentication codes (MACs). Mechanisms using a block cipher," 2011. [Online]. Available: https://bsol.bsigroup.com/Bibliographic/BibliographicInfoData/000000000030105371

[32] BSI, "BS EN IEC 62443-4-1:2018 – Security for industrial automation and control systems – Secure product development lifecycle requirements," Tech. Rep., 2018. [Online]. Available: https://bsol.bsigroup.com/Bibliographic/BibliographicInfoData/000000000030321856

[33] M. Burmester and Y. Desmedt, "A secure and efficient conference key distribution system," in *Workshop on the Theory and Application of of Cryptographic Techniques*. Springer, 1994, pp. 275–286.

[34] L. Butcher, "Railways: West Coast Main Line," Available: www.parliament.uk/briefing-papers/SN00364.pdf, 2010, online.

[35] E. Byres, "The Air Gap: SCADA's Enduring Security Myth," *Commun. ACM*, vol. 56, no. 8, pp. 29–31, Aug. 2013. [Online]. Available: http://doi.acm.org/10.1145/2492007.2492018

[36] E. J. Byres, M. Franz, and D. Miller, "The use of attack trees in assessing vulnerabilities in SCADA systems," in *in IEEE Conf. International Infrastructure Survivability Workshop (IISW '04). Institute for Electrical and Electronics Engineers*, 2004.

[37] A. A. Cárdenas, S. Amin, and S. Sastry, "Research Challenges for the Security of Control Systems." in *HotSec*, 2008.

[38] T. Caulfield, D. Pym, and J. Williams, "Compositional security modelling," in *International Conference on Human Aspects of Information Security, Privacy, and Trust*. Springer, 2014, pp. 233–245.

[39] M. Chai and B.-H. Schlingloff, "Monitoring systems with extended live sequence charts," in *International Conference on Runtime Verification*. Springer, 2014, pp. 48–63.

[40] P. Cheng, L. Wang, S. Jajodia, and A. Singhal, "Aggregating CVSS base scores for semantics-rich network security metrics," in *Reliable Distributed Systems (SRDS), 2012 IEEE 31st Symposium on*. IEEE, 2012, pp. 31–40.

[41] T. Chothia, M. Ordean, J. de Ruiter, and R. J. Thomas, "An Attack Against Message Authentication in the ERTMS Train to Trackside Communication Protocols," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ser. ASIA CCS '17. New York, NY, USA: ACM, 2017, pp. 743–756. [Online]. Available: http://doi.acm.org/10.1145/3052973.3053027

[42] T. Chothia and V. Smirnov, "A Traceability Attack against e-Passports," *Financial Cryptography and Data Security*, vol. 6052, pp. 20–34, 2010. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-14577-3_5

[43] M. Comptier, D. Deharbe, J. M. Perez, L. Mussat, T. Pierre, and D. Sabatier, "Safety Analysis of a CBTC System: A Rigorous Approach with Event-B," in *Reliability, Safety, and Security of Railway Systems. Modelling, Analysis, Verification, and Certification*, A. Fantechi, T. Lecomte, and A. Romanovsky, Eds. Cham: Springer International Publishing, 2017, pp. 148–159.

[44] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, "Introduction to Algorithms Second Edition," 2001.

[45] C. Cremers, M. Horvat, J. Hoyland, S. Scott, and T. van der Merwe, "A Comprehensive Symbolic Analysis of TLS 1.3," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: ACM, 2017, pp. 1773–1788. [Online]. Available: http://doi.acm.org/10.1145/3133956.3134063

[46] C. Cremers, M. Horvat, S. Scott, and T. van der Merwe, "Automated Analysis and Verification of TLS 1.3: 0-RTT, Resumption and Delayed Authentication," in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2016, pp. 470–485.

[47] C. J. Cremers, "The Scyther tool: Verification, falsification, and analysis of security protocols," in *International Conference on Computer Aided Verification*. Springer, 2008, pp. 414–418.

[48] A. Dabrowski, N. Pianta, T. Klepp, M. Mulazzani, and E. Weippl, "IMSI-catch Me if You Can: IMSI-catcher-catchers," in *Proceedings of the 30th Annual Computer Security Applications Conference*, ser. ACSAC '14. New York, NY, USA: ACM, 2014, pp. 246–255. [Online]. Available: http://doi.acm.org/10.1145/2664243.2664272

[49] A. DasGupta, "The matching, birthday and the strong birthday problem: a contemporary review," *Journal of Statistical Planning and Inference*, vol. 130, no. 1, pp. 377–389, 2005.

[50] J. de Ruiter, "Lessons learned in the analysis of the EMV and TLS security protocols," Ph.D. dissertation, Radboud University Nijmegen, 2015.

[51] J. de Ruiter, R. J. Thomas, and T. Chothia, "A Formal Security Analysis of ERTMS Train to Trackside Protocols," in *Reliability, Safety and Security of Railway Systems: Modelling, Analysis, Verification and Certification. International Conference, Paris, France, June 28-30, 2016, Proceedings*, ser. Lecture Notes in Computer Science, A. R. Thierry Lecomte, Ralf Pinger, Ed., 2016.

[52] Department for Transport, "Implementation of the NIS Directive," 2018. [Online]. Available: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/707969/implementation-of-the-nis-directive-dft-guidance.pdf

[53] B. Dowling, M. Fischlin, F. Günther, and D. Stebila, "A cryptographic analysis of the tls 1.3 handshake protocol candidates," in *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '15. New York, NY, USA: ACM, 2015, pp. 1197–1210. [Online]. Available: http://doi.acm.org/10.1145/2810103.2813653

[54] Dragos Inc., "CRASHOVERRIDE: Analysis of the Threat to Electric Grid Operators," 2017. [Online]. Available: https://dragos.com/blog/crashoverride/CrashOverride-01.pdf

[55] Y. Dutuit, F. Innal, A. Rauzy, and J.-P. Signoret, "Probabilistic assessments in relationship with safety integrity levels by using Fault Trees," *Reliability Engineering & System Safety*, vol. 93, no. 12, pp. 1867 – 1876, 2008, 17th European Safety and Reliability Conference. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0951832008001099

[56] ERA, "Assignment of Values to ETCS Variables–v1.25," European Union Agency for Railways, Tech. Rep., 2015. [Online]. Available: https://www.era.europa.eu/sites/default/files/activities/docs/ertms_040001_etcs_variables_values_en.pdf

[57] ERA, "SUBSET-104: ETCS System Version Management," European Union Agency for Railways, Tech. Rep., 2015. [Online]. Available: https://www.era.europa.eu/filebrowser/download/525_en

[58] ERA, "SUBSET-026: System requirements specification, version 3.6.0," European Union Agency for Railways, Tech. Rep., 2016. [Online]. Available: https://www.era.europa.eu/filebrowser/download/493_en

[59] R. Esposito, A. Lazzaro, P. Marmo, and A. Sanseviero, "Formal verification of ERTMS EuroRadio safety critical protocol," *Proceedings 4th symposium on Formal Methods for Railway Operation and Control Systems (FORMS'03)*, 2003. [Online]. Available: http://www.math.unipd.it/~tullio/CS/Dispense_2005/Articolo-3.pdf

[60] R. Evans, J. Easton, and C. Roberts, "SCEPTICS: A Systematic Evaluation Process for Threats to Industrial Control Systems." World Congress on Railway Research, 2016.

[61] A. Fantechi, "Twenty-Five Years of Formal Methods and Railways: What Next?" in *Software Engineering and Formal Methods*, S. Counsell and M. Núñez, Eds. Cham: Springer International Publishing, 2014, pp. 167–183.

[62] A. Fantechi, "Formal Techniques for a Data-Driven Certification of Advanced Railway Signalling Systems," in *Critical Systems: Formal Methods and Automated Verification*, M. H. ter Beek, S. Gnesi, and A. Knapp, Eds. Cham: Springer International Publishing, 2016, pp. 231–245.

[63] N. Ferguson, B. Schneier, and T. Kohno, *Cryptography engineering: design principles and practical applications*. John Wiley & Sons, 2011.

[64] A. Fielder, T. Li, and C. Hankin, "Defense-in-depth vs. Critical Component Defense for Industrial Control Systems," in *4th International Symposium for ICS & SCADA Cyber Security Research 2016, ICS-CSR 2016, 23 - 25 August 2016, Queen's Belfast University, UK*, ser. Workshops in Computing, T. Brandstetter and H. Janicke, Eds. BCS, 2016. [Online]. Available: http://ewic.bcs.org/content/ConWebDoc/56472

[65] FIRST.Org, Inc., "Common Vulnerability Scoring System v3.0: Specification Document," 2016. [Online]. Available: https://www.first.org/cvss/cvss-v30-specification-v1.8.pdf

[66] M. Franekova and P. Chrtiansky, "Key Management System in ETCS," *Transport System Telematics*, 2009. [Online]. Available: http://yadda.icm.edu.pl/baztech/download/import/contents/BSL7-0037-0002-httpwww_bg_utp_edu_plartarchives20of20transport20system20telematics2008-20112009-022012.pdf

[67] M. Franekova, K. Rastocny, A. Janota, and P. Chrtiansky, "Safety Analysis of Cryptography Mechanisms used in GSM for Railway," *International Journal of Engineering*, vol. 11, no. 1, pp. 207–212, 2011. [Online]. Available: http://annals.fih.upt.ro/pdf-full/2011/ANNALS-2011-1-34.pdf

[68] M. Franekova and M. Výrostko, *Telematics in the Transport Environment: 12th International Conference on Transport Systems Telematics, TST 2012, Katowice-Ustroń, Poland, October 10–13, 2012. Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, ch. Approaches to a Solution of Key Management System for Cryptography Communications within Railway Applications, pp. 301–313. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-34050-5_34

[69] S. Fuloria, R. Anderson, F. Alvarez, and K. McGrath, "Key management for substations: Symmetric keys, public keys or no keys?" in *Power Systems Conference and Exposition (PSCE), 2011 IEEE/PES*. IEEE, 2011, pp. 1–6.

[70] S. Fuloria, R. Anderson, K. McGrath, K. Hansen, and F. Alvarez, "The protection of substation communications," in *Proceedings of SCADA Security Scientific Symposium*, 2010.

[71] N. Furness, H. van Houten, L. Arenas, and M. Bartholomeus, "ERTMS Level 3: the game-changer," *IRSE News View*, vol. 232, p. 232, 2017.

[72] GSM-R Functional Group, "EIRENE Functional Requirements Specification, version 7.4.0," Tech. Rep., 2014. [Online]. Available: https://uic.org/IMG/pdf/p0028d003.4r0.4-7.4.0.pdf

[73] GSM-R Functional Group, "EIRENE System Requirements Specification, version 15.4.0," Tech. Rep., 2014. [Online]. Available: http://www.era.europa.eu/Document-Register/Documents/P0028D004.3r0.5-15.4.0.pdf

[74] A. Halchin, A. Feliachi, N. K. Singh, Y. Ait-Ameur, and J. Ordioni, "B-PERFect," in *Reliability, Safety, and Security of Railway Systems. Modelling, Analysis, Verification, and Certification*, A. Fantechi, T. Lecomte, and A. Romanovsky, Eds. Cham: Springer International Publishing, 2017, pp. 160–172.

[75] H. Handschuh and B. Preneel, "Minding your MAC algorithms," *Information Security Bulletin*, vol. 9, no. 6, pp. 213–221, 2004.

[76] A. Hawthorn, "A Proven Approach to Requirements Engineering – The Why, What and How of REVEAL," 2017. [Online]. Available: https://conferences.ncl.ac.uk/media/sites/conferencewebsites/rssrail/Andrew%20Hawthorn%20-%20ALTRAN%20-%20A%20Proven%20Approach%20to%20Requirements%20Engineering.pdf

[77] G. Hayes and K. El-Khatib, "Securing modbus transactions using hash-based message authentication codes and stream transmission control protocol," in *2013 Third International Conference on Communications and Information Technology (ICCIT)*, June 2013, pp. 179–184.

[78] J. Hayes, "ATO over ETCS – Achieving Metro Headways on the Mainline," 2017. [Online]. Available: https://www.era.europa.eu/sites/default/files/events-news/docs/ccrcc_2017_l2_ato_etcs_network_rail_en.pdf

[79] L. Hongjie, C. Lijie, and N. Bin, "Petri net-based analysis of the safety communication protocol," *Telkomnika*, vol. 11, no. 10, pp. 6034–6041, 2013. [Online]. Available: http://www.iaescore.com/journals/index.php/IJEECS/article/view/2736/3850

[80] M. Hutle, "SPARKS threat analysis using Attack Trees and Semantic Threat Graphs," 2015. [Online]. Available: https://hyrim.net/wp-content/uploads/2015/11/10_SPARKS_Threat_Analysis_Using_Attack_Trees_and_Semantic_Threat_Graphs.pdf

[81] V. M. Igure, S. A. Laughter, and R. D. Williams, "Security issues in SCADA networks," *Computers & Security*, vol. 25, no. 7, pp. 498 – 506, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404806000514

[82] C. Ioannidis, D. Pym, and J. Williams, "Information security trade-offs and optimal patching policies," *European Journal of Operational Research*, vol. 216, no. 2, pp. 434–444, 2012.

[83] X. Iturbe, J. Jiménez, A. Zuloaga, J. Lázaro, and J. L. Martín, "The Train Communication Network: Standardization goes aboard," in *2010 IEEE International Conference on Industrial Technology*, March 2010, pp. 1667–1672.

[84] M. Just and S. Vaudenay, "Authenticated multi-party key agreement," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 1996, pp. 36–49.

[85] M. Kalenderi, D. Pnevmatikatos, I. Papaefstathiou, and C. Manifavas, "Breaking the GSM A5/1 cryptography algorithm with rainbow tables and high-end FPGAS," in *22nd International Conference on Field Programmable Logic and Applications (FPL)*. IEEE, 2012, pp. 747–753.

[86] B. Kang, P. Maynard, K. McLaughlin, S. Sezer, F. Andrén, C. Seitl, F. Kupzog, and T. Strasser, "Investigating cyber-physical attacks against IEC 61850 photovoltaic inverter installations," in *2015 IEEE 20th Conference on Emerging Technologies Factory Automation (ETFA)*, Sept 2015, pp. 1–8.

[87] L. Karstensen, "GSM A5/1 rainbow tables in Oslo, Norway," 2015. [Online]. Available: https://lassekarstensen.wordpress.com/2013/08/08/gsm-a51-rainbow-tables-in-oslo-norway/

[88] R. Khan, K. McLaughlin, D. Laverty, and S. Sezer, "STRIDE-based threat modeling for cyber-physical systems," in *Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), 2017 IEEE PES*. IEEE, 2017, pp. 1–6.

[89] B. Kordy, S. Mauw, S. Radomirović, and P. Schweitzer, "Foundations of attack–defense trees," in *Formal Aspects of Security and Trust*, P. Degano, S. Etalle, and J. Guttman, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 80–95.

[90] K. Koyama and K. Ohta, "Identity-based conference key distribution systems," in *Conference on the Theory and Application of Cryptographic Techniques*. Springer, 1987, pp. 175–184.

[91] KPMG, "IT Security Threat identification, Risk Analysis and Recommendations," 2013. [Online]. Available: http://www.ertms.be/pdf/IT_Security_Threat_identification.pdf

[92] R. Langner, "Stuxnet: Dissecting a Cyberwarfare Weapon," *IEEE Security Privacy*, vol. 9, no. 3, pp. 49–51, May 2011.

[93] D. Lautier and F. Raynaud, "Systemic Risk and Complex Systems: A Graph-Theory Analysis," in *Econophysics of Systemic Risk and Network Dynamics*. Springer, 2013, pp. 19–37.

[94] R. M. Lee, M. J. Assante, and T. Conway, "Analysis of the cyber attack on the Ukrainian power grid," *Electricity Information Sharing and Analysis Center (E-ISAC)*, 2016.

[95] T. Li and C. Hankin, "Effective defence against zero-day exploits using bayesian networks," in *Critical Information Infrastructures Security*, G. Havarneanu, R. Setola, H. Nassopoulos, and S. Wolthusen, Eds. Cham: Springer International Publishing, 2017, pp. 123–136.

[96] C. Lijie, H. Yinxia, Z. Tianshi, and Z. Jian, "A decision-based analysis for the safety communication protocol in train control system," in *2016 SAI Computing Conference (SAI)*, July 2016, pp. 649–654.

[97] C. Lijie, T. Tao, Z. Xianqiong, and E. Schnieder, "Verification of the safety communication protocol in train control system using colored Petri net," *Reliability Engineering & System Safety*, vol. 100, pp. 8 – 18, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0951832011002730

[98] T. Lodderstedt, D. Basin, and J. Doser, "SecureUML: A UML-based modeling language for model-driven security," in *International Conference on the Unified Modeling Language*. Springer, 2002, pp. 426–441.

[99] I. Lopez and M. Aguado, "Cyber security analysis of the European train control system," *IEEE Communications Magazine*, vol. 53, no. 10, pp. 110–116, October 2015.

[100] J. Lu, Z. Li, and M. Henricksen, "Time-memory trade-off attack on the GSM A5/1 stream cipher using commodity GPGPU," in *13th International Conference on Applied Cryptography and Network Security (ACNS 2015)*, 2015.

[101] I. Mann, *Hacking the human: social engineering techniques and security countermeasures*. Routledge, 2017.

[102] R. Matulevičius, *Fundamentals of Secure System Modelling*. Cham: Springer International Publishing, 2017. [Online]. Available: https://doi.org/10.1007/978-3-319-61717-6

[103] C. McMahon Stone, T. Chothia, and J. de Ruiter, "Extending automated protocol state learning for the 802.11 4-way handshake," in *Computer Security*, J. Lopez, J. Zhou, and M. Soriano, Eds. Cham: Springer International Publishing, 2018, pp. 325–345.

[104] S. Meier, B. Schmidt, C. Cremers, and D. Basin, "The TAMARIN Prover for the Symbolic Analysis of Security Protocols," in *Computer Aided Verification*, N. Sharygina and H. Veith, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 696–701.

[105] C. J. Mitchell, "Key recovery attack on ANSI retail MAC," *Electronics Letters*, vol. 39, no. 4, pp. 361–362, 2003.

[106] F. Moller, H. N. Nguyen, M. Roggenbach, S. Schneider, and H. Treharne, "Railway modelling in CSP||B: the double junction case study," *Electronic Communications of the EASST*, vol. 53, 2012.

[107] Office of Road and Rail, "Rail infrastructure, assets and environmental 2016-17 Annual Statistical Release," 2017, online. [Online]. Available: http://orr.gov.uk/__data/assets/pdf_file/0008/25838/rail-infrastructure-assets-environmental-2016-17.pdf

[108] M. R. Orsman, "The Victoria Line upgrade: From concept to full operation," in *IET Professional Development Course on Railway Signalling and Control Systems (RSCS 2012)*, May 2012, pp. 191–204.

[109] F. Pépin and M. G. Vigliotti, *Risk Assessment of the 3Des in ERTMS*. Cham: Springer International Publishing, 2016, pp. 79–92. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-33951-1_6

[110] O. Pereira and J. J. Quisquater, "A security analysis of the cliques protocols suites," in *Computer Security Foundations Workshop, 2001. Proceedings. 14th IEEE*, 2001, pp. 73–81.

[111] J. L. Petersen, "Mathematical Methods for validating Railway Interlocking Systems," Ph.D. dissertation, Technical University Denmark, 1998. [Online]. Available: http://www2.compute.dtu.dk/~dibj/racosy/jlpthesis.ps

[112] S. Petrovic and A. Fúster-Sabater, "Cryptanalysis of the A5/2 Algorithm," Cryptology ePrint Archive, Report 2000/052, 2000. [Online]. Available: https://eprint.iacr.org/2000/052.pdf

[113] L. Piètre-Cambacédès and P. Sitbon, "Cryptographic Key Management for SCADA Systems-Issues and Perspectives," in *2008 International Conference on Information Security and Assurance (ISA 2008)*, April 2008, pp. 156–161.

[114] K. Prendergast, "Condition monitoring on the Class 390 Pendolino," in *2008 4th IET International Conference on Railway Condition Monitoring*, June 2008, pp. 1–6.

[115] B. Preneel and P. van Oorschot, "On the security of iterated message authentication codes," *Information Theory, IEEE Transactions on*, vol. 45, no. 1, pp. 188–199, Jan 1999.

[116] B. Preneel and P. C. van Oorschot, "Key recovery attack on ANSI X9. 19 retail MAC," *Electronics Letters*, vol. 32, no. 17, pp. 1568–1569, 1996.

[117] H. Quan, H. Zhao, and G. Zhou, "Analysis on EURORADIO safety critical protocol by probabilistic model checking," in *2013 IEEE International Conference on Intelligent Rail Transportation Proceedings*, Aug 2013, pp. 75–78.

[118] Rail Safety and Standards Board, "RIS-0708-CCS - ERTMS/ETCS National Values," 2017. [Online]. Available: https://www.rssb.co.uk/rgs/standards/ris-0708-ccs%20iss%201.pdf

[119] Rail Safety and Standards Board, "The Management of Packet 44 Applications," 2017. [Online]. Available: https://www.rssb.co.uk/rgs/standards/RIS-0784-CCS%20Iss%201.pdf

[120] Rail Safety and Standards Board, "GE/RT8000/TW5 - Preparation and movement of trains: Defective or isolated vehicles and on-train equipment," 2018. [Online]. Available: https://www.rssb.co.uk/rgs/rulebooks/GERM8000-master-module%20Iss%201.pdf

[121] Rete Ferroviaria Italiana, "Protocollo Vitale Standard," 2009. [Online]. Available: http://www.gare.italferr.it/cms-file/allegati/gare-italferr/PA-1146_ProtocolloVitaleStandard-RFI-DT-revA.pdf

[122] RSSB, "GSM-R User Procedures, issue 7.1," Tech. Rep., 2015. [Online]. Available: http://www.rssb.co.uk/Library/improving-industry-performance/2015-gsm-r-cab-mobile-user-procedures-ns-gsm-r-ops-0514.pdf

[123] RSSB, "Rail Industry Standard RIS-0743-CCS – ERTMS Key Management," Tech. Rep., 2017. [Online]. Available: https://www.rssb.co.uk/rgs/standards/RIS-0743-CCS%20Iss1.pdf

[124] RSSB, "Calculation of Enhanced Permissible Speeds for Tilting Trains," Tech. Rep., 2018. [Online]. Available: https://www.rssb.co.uk/rgs/standards/RIS-7704-INS%20Iss%201.pdf

[125] R. Santini, C. Foglietta, and S. Panzieri, "A graph-based evidence theory for assessing risk," in *Information Fusion (Fusion), 2015 18th International Conference on.* IEEE, 2015, pp. 1467–1474.

[126] B. Schneier, *Applied cryptography: protocols, algorithms, and source code in C.* John Wiley & Sons, 2007.

[127] D. P. Shepard, T. E. Humphreys, and A. A. Fansler, "Evaluation of the vulnerability of phasor measurement units to gps spoofing attacks," *International Journal of Critical Infrastructure Protection*, vol. 5, no. 3, pp. 146 – 153, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1874548212000480

[128] A. Shostack, *Threat Modeling: Designing for Security*, 1st ed. Wiley Publishing, 2014.

[129] N. P. Smart, *Cryptography Made Simple*. Cham: Springer International Publishing, 2016. [Online]. Available: https://doi.org/10.1007/978-3-319-21936-3

[130] H. Song, H. Liu, and E. Schnieder, "A train-centric communication-based new movement authority proposal for etcs-2," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2018.

[131] SR Labs, "Decrypting GSM phone calls," 2010. [Online]. Available: https://srlabs.de/decrypting_gsm/

[132] M. Steiner, G. Tsudik, and M. Waidner, "Diffie-Hellman key distribution extended to group communication," in *Proceedings of the 3rd ACM conference on Computer and communications security*. ACM, 1996, pp. 31–37.

[133] R. J. Thomas, M. Ordean, T. Chothia, and J. de Ruiter, "TRAKS: A Universal Key Management Scheme for ERTMS," in *Proceedings of the 33rd Annual Conference on Computer Security Applications*, ser. ACSAC '17. New York, NY, USA: ACM, 2017.

[134] S. L. Thomas and A. Francillon, "Backdoors: Definition, Deniability & Detection," in *Proceedings of the 21st International Symposium on Research in Attacks, Intrusions and Defenses, ser. RAID*, vol. 18, 2018.

[135] S. L. Thomas, "Backdoor Detection Systems for Embedded Devices," Ph.D. dissertation, University of Birmingham, 2018.

[136] TÜV SÜD Rail GmbH, "Approval of ERTMS in a reality of ever changing software," 2014. [Online]. Available: https://www.tuev-sued.de/uploads/images/14171761820482248805 31/tuev-sued-approval-of-ertms-in-a-reality-of-ever-changing.pdf

[137] W.-G. Tzeng and Z.-J. Tzeng, "Round-efficient conference key agreement protocols with provable security," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2000, pp. 614–627.

[138] UNISIG, "ERTMS EuroRadio Test cases - Safety Layer," 2015. [Online]. Available: https://www.era.europa.eu/filebrowser/download/518_en

[139] UNISIG, "SUBSET-036 - FFFIS for Eurobalise, version 3.1.0," European Union Agency for Railways, Tech. Rep., 2015. [Online]. Available: https://www.era.europa.eu/filebrowser/download/498_en

[140] UNISIG, "SUBSET-037 - EuroRadio FIS, version 3.2.0," European Union Agency for Railways, Tech. Rep., 2015. [Online]. Available: https://www.era.europa.eu/filebrowser/download/499_en

[141] UNISIG, "SUBSET-038 - Off-line key management FIS, version 3.1.0," European Union Agency for Railways, Tech. Rep., 2015. [Online]. Available: https://www.era.europa.eu/filebrowser/download/500_en

[142] UNISIG, "SUBSET-114 - KMC-ETCS entity off-line KM FIS, version 1.1.0," European Union Agency for Railways, Tech. Rep., 2015. [Online]. Available: https://www.era.europa.eu/filebrowser/download/541_en

[143] UNISIG, "SUBSET-137 - KMC-ETCS entity on-line KM FIS, version 1.0.0," European Union Agency for Railways, Tech. Rep., 2015. [Online]. Available: https://www.era.europa.eu/filebrowser/download/542_en

[144] UNISIG, "SUBSET-026-7: System Requirements Specification – Chapter 7 - ERTMS/ETCS language," 2016. [Online]. Available: https://www.era.europa.eu/filebrowser/download/493_en

[145] UNISIG, "Test Specification for EuroBalise FFFIS," 2016. [Online]. Available: https://www.era.europa.eu/filebrowser/download/519_en

[146] S. Vanit-Anunchai, "Application of Coloured Petri Nets in Modelling and Simulating a Railway Signalling System," in *Critical Systems: Formal Methods and Automated Verification*, M. H. ter Beek, S. Gnesi, and A. Knapp, Eds. Cham: Springer International Publishing, 2016, pp. 214–230.

[147] J. Wolff, "What is the Value of Preventing a Fatality?" in *Risk: Philosophical Perspectives*, T. Lewens, Ed. Routledge, 2007.

[148] M. Yearworth, B. Monahan, and D. Pym, "Predictive modelling for security operations economics," in *Workshop on the Economics of Securing the Information Infrastructure (WESII)*, vol. 23, 2006, p. 24.

[149] S. Yilek, E. Rescorla, H. Shacham, B. Enright, and S. Savage, "When Private Keys Are Public: Results from the 2008 Debian OpenSSL Vulnerability," in *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '09. New York, NY, USA: ACM, 2009, pp. 15–27. [Online]. Available: http://doi.acm.org/10.1145/1644893.1644896

[150] Y. Zhang, T. Tang, K. Li, J. M. Mera, L. Zhu, L. Zhao, and T. Xu, "Formal verification of safety protocol in train control system," *Science China Technological Sciences*, vol. 54, no. 11, pp. 3078–3090, 2011. [Online]. Available: http://dx.doi.org/10.1007/s11431-011-4562-2