TRAKS: A Universal Key Management Scheme for ERTMS

Thomas, Richard; Ordean, Mihai; Chothia, Tom; De Ruiter, Joeri

DOI:
10.1145/3134600.3134631

Document Version
Peer reviewed version

Citation for published version (Harvard):
Thomas, R, Ordean, M, Chothia, T & De Ruiter, J 2017, TRAKS: A Universal Key Management Scheme for ERTMS. in *ACSAC 2017 Proceedings of the 33rd Annual Computer Security Applications Conference.* Association for Computing Machinery (ACM), pp. 327-338, 33rd Annual Computer Security Applications Conference (ACSAC 2017), Orlando, Florida, United States, 4/12/17. https://doi.org/10.1145/3134600.3134631

TRAKS: A Universal Key Management Scheme for ERTMS

Richard J. Thomas
University of Birmingham
Birmingham, UK
R.J.Thomas@cs.bham.ac.uk

Mihai Ordean
University of Birmingham
Birmingham, UK
M.Ordean@cs.bham.ac.uk

Tom Chothia
University of Birmingham
Birmingham, UK
T.P.Chothia@cs.bham.ac.uk

Joeri de Ruiter
Radboud University
Nijmegen, NL
joeri@cs.ru.nl

ABSTRACT

This paper presents a new Key Management and Distribution Scheme for use in the European Rail Traffic Management System (ERTMS). Its aim is to simplify key management and improve cross-border operations through hierarchical partitioning. The current scheme used in ERTMS involves the creation and distribution of 3DES keys to train and trackside entities, which are then used as part of the EuroRadio Protocol to provide message authentication. This results in the distribution of tens of thousands of keys using portable media, a prohibitively high burden on management and resourcing. We present a symmetric key solution, TRAKS, which has the benefit of being backwards compatible with the current ERTMS standard and being post-quantum secure. This new scheme reduces the number of cryptographic keys in circulation, and maintains the current security model. We achieve this by dynamically deriving unique keys from a shared secret, i.e. the line secret, which is combined with IDs of trains, and of signalling equipment. In addition to providing better key management, our scheme also adds authentication to the location data provided by EuroBalises.

CCS CONCEPTS

•Security and privacy → Key management; Hash functions and message authentication codes; Authorization; Mobile and wireless security;

1 INTRODUCTION

In any Industrial Control System (ICS) that has safety-critical functionality, it is important for any message to be authenticated in order to ensure that the message came from a genuine entity who had appropriate authorisation to send that message, and to detect malicious modifications by an attacker.

The European Rail Traffic Management System (ERTMS) is a safety-critical ICS which provides a suite of protocols used to deliver a modern train management and signalling platform[1]. This standard is designed with the intention to enable trains to interoperate across borders and optimise the running operation of railways. At present, the system is being rolled out across Europe and also on high-speed lines around the world. ERTMS is defined as a protocol stack formed of the following three layers: GSM-R [9], EuroRadio and the Application Layer Protocol. The EuroRadio and the Application Layer Protocol form ETCS, the European Train Control System [16]. GSM-R, a rail-specific variant of the GSM protocol, is used for communications between the train and trackside infrastructure such as radio block controllers (RBCs), i.e. the trackside components that manage trains in a geographical area. RBCs are responsible for issuing 'movement authorities', messages which permit a train to move a specific distance at a given speed, and managing safe train movement in a geographic region of approximately 70km. Trains periodically provide location updates and the RBC would respond with an updated movement authority. The EuroRadio protocol layer provides authentication and integrity of the communication using cryptographic MACs. Messages which have a valid MAC (or are from a carefully selected subset of messages that may be sent at a high priority and not requiring a MAC) are passed to the application layer. The Application Layer Protocol [6] is a stateful protocol that defines the ERTMS message standard and additionally implements checks that help prevent message replays.

EuroBalises are devices placed between the tracks, typically in groups of two or three, which are read by a train passing over them. The train trusts the EuroBalise to provide accurate location (rather than using GPS) and track profile data, which can include speed limits, gradients and tilt profiles. Currently, the balise data is validated using a CRC code, which is publicly known [21], and is only for error detection but does not provide any integrity protection.

The current ERTMS standard [18] states that key provisioning and management should be done based on geographical *domains* (e.g. Great Britain), where each domain has a Key Management Centre (KMC) which is responsible for key generation and management for that domain. Additionally, the KMC also defines procedures to install the keys on train on-board units (OBUs) and RBCs. Throughout the paper we will interchangeably use the terms *train* and *OBU* to refer to trains. The current procedure requires that keys for an OBU or from an RBC are generated by the KMCs following a request from a vendor (e.g. Siemens). After generation, the keys for

ACSAC 2017, San Juan, PR, USA

978-1-4503-5345-8/17/12…$15.00
DOI: 10.1145/3134600.3134631

[1] http://www.ertms.net

the requesting OBU or RBC are sent in the clear on portable media devices [19], to be installed.

This setup is highly inefficient; using portable media devices to move keys greatly increases the risk of compromise, especially due to the fact that keys on the device are stored in cleartext. Additionally, this makes deployment and management of keys difficult (i.e. in order to update a key for an RBC, an engineer needs to physically travel to the RBC's location in order to install the key on the portable media device). Informal discussions with rail systems managers have highlighted that insecure strategies like (i) provisioning all (OBU, RBC) key pairs to each OBU and RBC, or simply (ii) having KMCs extend the life of keys when they are due to expire are used in practice. Cross-border operation is also challenging as keys need to be shared between geographical domains that are managed by different KMCs. Under the current scheme, additional burden is placed on the foreign KMC operators (KMC owners who are outside of the 'home domain' that the train is registered). Whenever a new OBU is to operate in their domain, they are required to establish the appropriate keys to hand to the 'home' KMC, but they also have to send an engineer to each of their RBCs to install the necessary keys. In the case of a progressive national deployment, this burden is high.

ICS environments are also designed with a long lifespan as a key feature. Any solutions presented must consider issues such as post-quantum security, where quantum computers could render some encryption schemes insecure. It should, however, be noted that there are current efforts by NIST and ETSI to standardise post-quantum cryptography. The time required to standardise this, though, is past the intended start of ERTMS national deployment. For example in the United Kingdom, Crossrail and the East Coast mainline are expected to be complete by 2021.

In this paper, we give a formal definition of the security of the ERTMS key generation. We continue by proposing a new key management scheme for ERTMS comprised of a key generation scheme, which introduces line secrets to enforce an operational permissions model, and, when combined with the train ID and RBC ID produces a key which can be used in the EuroRadio protocol to negotiate a session key for message MACs, and its corresponding key distribution protocol. Our solution is *backwards-compatible* with the current scheme allowing infrastructure managers to progressively implement it into the standard.

1.1 Contributions

This paper proposes TRAKS (Train and RBC Authenticated Key Scheme), a key management scheme for ERTMS. Our main focus with TRAKS is to reduce the management overhead, i.e. physically transport keys on portable media devices, an inefficient process which has lead to insecure practices, while maintaining the same flexibility and security for the keys. As such, our contributions are as follows:

(1) **Unified ERTMS Key Management and Distribution Protocol.** The TRAKS key management scheme is designed to generate message authentication keys for communications between OBUs and RBCs, and for communications between OBU and balises.
(2) **Reduced deployment overhead.** The TRAKS scheme provides support for dynamic key generation using pseudo-random functions (PRFs) and a shared secret. The method of producing long-term keys removes the need to physically access RBC after deployment (RBCs are still provisioned with a secret during deployment).
(3) **Backwards compatible and post-quantum secure.** Our protocol is designed using symmetric key cryptography in a way that allows it to be fully backwards compatible with the current scheme used by ERTMS. This makes it possible to gradually roll out this scheme in existing systems. Additionally, TRAKS uses post-quantum secure PRFs to derive keys.
(4) **Security analysis.** To our knowledge, we provide the first security definition for the key generation protocol in ERTMS. We also prove that TRAKS is at least as secure as the current scheme while providing all additional benefits with respect to key management.

Issues with the existing ERTMS scheme and related work are discussed in Section 2. We then provide a high-level overview and intuition of the current offline ERTMS scheme and TRAKS in Section 3. In Section 4, we provide a generic formalism of the offline ERTMS scheme, followed by our scheme, TRAKS, in Section 5. A security analysis is provided in Section 6. We discuss the key management lifecycle and distribution in Sections 7 and 8 respectively. Example applications of the scheme, showing its universal application in ERTMS, including EuroBalises (EBs) are presented in Section 9, and we conclude in Section 10.

2 BACKGROUND AND RELATED WORK

Currently, ERTMS Key Management is defined as two standards, one offline, and a proposed online scheme [19, 20]. Some past work has looked at ERTMS Key Management, for example in [7, 8], where ERTMS was taken as a case study in assessing the protection of critical infrastructure, with an introduction to a proposed ERTMS key management hierarchy, and study of the use of symmetric and public key cryptography and their applicability to the problem. They advocate symmetric key solutions over public key solutions. Pépin and Vigliotti [12] perform an analysis of ERTMS Key Management from a cryptographic perspective, identifying that the current scheme will fail when 3DES may be feasibly broken, but do not make any recommendations on how to amend the scheme other than a change in cryptographic algorithms.

Since the development of the offline scheme and the introduction of the ETCS Level 3 baselines, an online solution has been proposed in [20]. This involves the introduction of a PKI in which certificates and keys are issued to trains and RBCs, where they can then communicate with the Key Management Centre. This, however, relies on the assumption that GSM-R bandwidths are capable of these communications, and rely on the upgrade to GPRS support, which is not currently ratified for use in ERTMS. The move to an online scheme would be disruptive for infrastructure managers. The only benefit of this scheme it is that an engineer does not have to interact with the train or RBC after initial setup, with the exception of maintenance periods. Furthermore, the management of a large-scale PKI for cross-border operation would become a burden

as all trains and RBCs will be required to regularly contact all key management centres they are entitled to use to update their certificates and keys. With the limiting bandwidths available by GSM-R, this is not a feasible solution, and places significant overheads on the train owner and infrastructure manager, especially given the long lifespan of ERTMS will limit the cryptographic primitives that can be used. In a post-quantum world, however, with the use of public-key cryptography defined in [20], the scheme would also not be secure, as it relies on RSA public key encryption. Both the offline and PKI-based schemes retain the same key hierarchy, as presented in Table 1.

Different solutions for key management have been proposed in the literature. For example, Biswas proposes a solution with multiple two-party keys and one multi-party key, where base and extended keys are used to exchange keys across a large, static group [2]. Other schemes, such as the Advanced Access Control System (AACS) [10], use pseudo-random functions to generate symmetric keys which are used to derive the necessary keys to decrypt media content. These schemes are, however, not directly applicable to ERTMS, as groups cannot be statically set and they change over time. The AACS scheme is also establishes long-term keys for decryption only and not for the authentication of payloads. Alternative proposals are based on Diffie-Hellman, which is not post-quantum secure [11, 15].

Several proposals that have been made by the National Infrastructure Managers to specifically improve ERTMS key management [1, 3, 13], as the current schemes either lacked definition on key distribution[2] or were not suitable for use. In the case of ProRail and ÖBB, keys are not issued equally, where ProRail tendered a solution which allowed one train to share the same key with many RBCs, in addition to the one train sharing a key with exactly one RBC. For ÖBB, an online scheme was required to distribute keys with RBCs as no standard was available at the time. These solutions, however, whilst compliant with the current standard, should not be deemed secure, as the compromise of one train key would compromise its communications with all RBCs under that scheme. Furthermore, the other solutions advocate the use of public key encryption through its use of RSA, which is not post-quantum secure, with no ratified alternatives for use.

3 ERTMS KEY GENERATION OVERVIEW

Within the current implementation of ERTMS, long-term keys are installed on RBCs and OBUs, which are used to derive a session key using exchanged nonces through the EuroRadio protocol [18]. This session key is used as the MAC key to authenticate and provide integrity protection of transmitted messages. In this section, we provide an overview of ERTMS and TRAKS Key Generation, before formally defining these schemes in Section 4.

In the current ERTMS Key Management scheme, the infrastructure manager allocates a unique key for each RBC-OBU pair. This key is randomly generated by the Key Management Centre, and is encrypted and a MAC is computed for this. This setup is pending changes following a proposed modification which is part of the ETCS Level 3 baseline standard that introduces a PKI, which allows the trains and RBCs to communicate directly to the KMC over TCP/IP.

The existing and proposed parallel scheme carry significant operational overhead, and both present a security exposure due to the potential to extend the validity of keys. In our solution, we offer a fully backwards-compatible scheme, which allows the KMC to issue keys until such a time that the RBC may be updated to support dynamic key generation. This reduces the overhead on national KMC owners, improves interoperability for cross-border operation, and also reduces the requirement of the home infrastructure manager to go out into the field to install keys on the RBCs for each new entity introduced into the network. Furthermore, the scheme presented in this paper supports the authentication of data on EuroBalises (EBs), such as the current location data. Currently this data is trusted and assumed to be accurate and authentic, but is sent without any form of authentication.

Security Requirements. Under the existing key key management scheme, an implicit permissions model exists: a EuroRadio session cannot be established if there is no cryptographic relationship between two entities denoted by id and id'. This is because a key $km_{id,id'}$ does not exist between them. One benefit of this scheme is that a train cannot be given signalling commands in an area it is not authorised to operate in. An example attack which is prevented is GSM relaying, where the train traffic is redirected to a distant GSM Base Station. Without this permissions model (assuming all keys have been installed), the train would be given the command to move forward until it has reached a balise to provide its location. This could place the train in a potentially unsafe situation.

Informal Attacker Model. In our scheme, we investigate attackers that have the ability to insert keys, intercept keys between the KMC and Train, and intercept keys between the KMC and RBC.

If an attacker is able to install arbitrary keys onto either the train or RBC, it can create a state of disassociation between what the KMC and train/RBC believes is the current set of loaded keys. It is currently not clear in the ERTMS specifications what remediation should take place if a key is attempted to be installed and one already exists. In the case of key interception, once the transport keys, keys which are used to encrypt and provide integrity and authentication of payloads between train/RBC and KMC, are installed onto the train or RBC, the attacker is not able to establish the $km_{id,id'}$ keys unless they know the appropriate secret used to generate the keys. As the transport keys are installed in plaintext, an attacker who obtains the transport key at the point of manufacture/commissioning is able to intercept all keys between a train or RBC.

We therefore consider an attacker from the Dolev-Yao model, who operates in polynomial time, and is able to monitor communications between the KMC and any train and RBC. In particular, the attacker is also able to send arbitrary messages to any given train, RBC or KMC, and either delay, modify or delete messages being sent between participants. Any proposed solution has to consider the capabilities of such an attacker.

ERTMS Parameters. In the ERTMS System Specifications [5], identity variables, known as `NID_C`, are assigned to specific 'zones' of operation, with varying scope, from an entire country to a specific

[2] The SUBSET-114 specifications were first released a number of years after SUBSET-037 was released

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	×	K-KMC_1	K-KMC_2

Figure 1: Current ERTMS Key Management Hierarchy [17], showing the specific keys used between ERTMS entitles in specific applications: session handshake, message authentication and encryption.

ERTMS Keys	Current	TRAKS
NID_C Secret	×	$knid\_c$
RBC Derivation Key	×	$km_{rid,null}$
Train Key	$K_{MAC_{rid,oid}}$	$km_{rid,oid}$
Balise Secret	×	km
Balise NID_C Area Key	×	$km_{\texttt{NID\_C},null}$
Balise MAC Key	×	$km_{\texttt{NID\_C},bgid}$

Figure 2: Proposed TRAKS Key Management Hierarchy. We introduce the new notation used throughout this paper as: a train with OBU ID 3, communicating with an RBC with ID 5 would have the key $km_{rid5,oid3}$. Balise MAC Keys are bound by the NID_C they are located in and their unique balise group ID, *bgid*.

line in a country. These are assigned by a central body, from a pool of available values. We leverage these variables to define a permissions model of where a train is allowed to operate, rather than relying on the direct pairing relationship that the current scheme uses. From this, the infrastructure manager derives a set of keys that are allocated to RBCs in that specific area identified by an NID_C. If the infrastructure manager is not in a position to support the full capabilities of dynamic key-derivation at RBC level, they can use the same functions to generate train keys for a RBC. This is done by computing the RBC key, and applying this to the set of train IDs authorised to operate within the NID_C. Train IDs and RBC IDs are allocated by the infrastructure manager from a central ERTMS ID database, *EDB*. The result of this process is a set of keys which are backwards-compatible with the existing ERTMS scheme. We provide an example application of TRAKS in Section 3.1.

For national infrastructure managers, this reduces the time taken to allow a train to cross borders, as the pre-requisite step to install keys on the RBC is no longer required, as the RBC is able to derive any key required based on the first EuroRadio message received by the train (where the train ID is provided). Under this model, we also enforce a permissions model, whereby, through partitioning of the rail network, we ensure that keys are not unnecessarily issued, where the operational zones are exactly defined and controlled. Trains may also be reallocated to new areas - in the current scheme, this would rely on an engineer going into the field to install the requisite keys on the train and RBCs, however, under our proposed scheme, TRAKS, this can be managed between the train and key management centre, with no requirement to interact with the RBC.

Current Key Distribution. Key distribution in ERTMS is currently a manual process, where keys are generated by the KMC and then distributed as required in response to a valid request. This set of keys is then accompanied with a separate MAC and encrypted using pre-shared symmetric keys referred to as *transport keys*. When a new train or RBC is added to the system, however, these transport keys, used to encrypt and authenticate payloads to and from the OBU, RBC and KMC, are installed in cleartext. The compromise of the transport media (e.g. USB stick or CD) would allow an attacker to forge their own valid messages which could compromise safety. Under TRAKS, we propose a new key distribution scheme, whereby trains and RBCs instead generate their own transport keys in such a way that an external attacker would require knowledge of the keys in order to obtain the keying material. In our scheme, we propose a shared key between the key management centre and approved vendors, which is used for vendor-KMC communications. Trains and RBCs have a burnt-in key pair, installed by the vendor which can be used for train/RBC-vendor communications.

In the following we define a framework for ERTMS key generation, outlining the functions run by the Key Management Centre. Using this outline definition, we can define a model of the current ERTMS Key Generation scheme in Algorithm 1 and TRAKS in Algorithm 2. Using these models, it is possible to prove the security of these mathematical models against an attacker, which gives us confidence that our proposed solution does not compromise the security of ERTMS.

Definition 3.1. An ERTMS key generation scheme KMAC = (SGen, INIT.ID, GEN.KMAC) over a ERTMS id 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 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 input a database *EDB* and a ERTMS entity type t 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 input two ids $id \in ID_t$ and $id' \in ID_{t'}$, and a secret $knid\_c$, and outputs the key $km_{id,id'}$ that is used to authenticate communication between id and id'.

3.1 TRAKS Application Example

We present a high-level hierarchy of ERTMS Key Management under TRAKS in Figure 3. Key management operations which allow trains to operate across borders remain unchanged, with a symmetric pair of keys shared between KMCs. From the current ERTMS scheme, we introduce an intermediary level, leveraging

NID_C, denoted in Figure 3 by $knid\_c_x$ where x is the identity of the region (NID_C) of operation. NID_C is an allocated ERTMS variable that represents a specific region or line. This has a unique secret key allocated per NID_C, representing a mapping from NID_C to its line secret.

For a given region or line, a number of RBCs may operate. Using the line secret $knid\_c_x$, and the RBC identity, the RBC keys are established. In the figure, we show two trains which are authorised to operate over two NID_C regions each. By applying Definition 3.1 and additionally supplying the identity of the train, the KMC allocates the keys required to the OBUs. For the RBCs, however, a single derivation function is required to establish the keys.

More concretely, let us consider an example national domain defined by Figure 3, with 4 NID_C regions and 6 RBCs. For each NID_C in this domain, which have NID_C values 1, 2, 3 and 4, the Key Management Centre will generate a line secret (based on a function which produces random output, e.g. using a Hardware Security Module) for the regions. It stores the line secret output $knid\_c_1$, ...,$knid\_c_4$. To provide RBC keys, the Key Management Centre will use the line secret for the given NID_C and compute the individual key for that RBC by combining the ID of the RBC and the line secret. We discuss the specifics of generating keys in Section 9. As an example, for RBC_2 and RBC_3, the Key Management Centre will use the line secret for NID_C_2, $knid\_c_2$, and compute the individual RBC keys based on the IDs of RBC_2 and RBC_3 (i.e. $rid2$ and $rid3$), producing $km_{rid2,null}$ and $km_{rid3,null}$, which would then be installed on the RBCs.

When train keys are required, the KMC will determine for which RBCs a key is required, and, for each of these RBCs, it will derive the corresponding RBC key, and output a key which is the result of applying the train ID to the RBC key using GEN.KMAC(rid, oid, $knid\_c_x$). This results in a key which uniquely pairs a train to a given RBC and can be installed on the train. When a train sends the first EuroRadio message, the RBC will use the ID of the train to compute the key it requires using the key it has installed by applying the train ID to the RBC key installed. As an example, for $Train_1$, with ID $oid1$, which operates in NID_Cs 1 and 2, it will receive all the necessary keys to communicate with the RBCs in that area, i.e. $km_{rid1,oid1}$, $km_{rid2,oid1}$ and $km_{rid3,oid1}$ will be installed on the train.

In the rest of this paper, we formalise the definition of this scheme and prove its correctness, and discuss how it can be implemented and used. In Section 5, we consider the security requirements and attacker model that TRAKS must work with, before defining the key lifecycle. We formally define the current ERTMS key management scheme in Section 4, presenting the TRAKS scheme in Section 5. We apply these formal definitions in Section 6 to prove that the current ERTMS Key Management scheme and proposed TRAKS scheme are secure.

4 OFFLINE ERTMS KEY GENERATION

In this section, the Offline ERTMS key generation scheme that is currently in use is presented in more detail.

The key provisioning process begins with the vendor (e.g. Siemens, Alstom and Bombardier) of an entity (e.g. OBU) making a formal request for an identity (ETCS ID), followed by a request for keys.

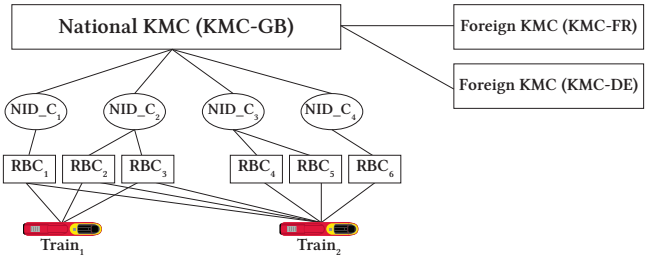

Figure 3: TRAKS Key Hierarchy for ERTMS. TRAKS is composed of four layers: (1) the national infrastructure for the 'home domain', which is responsible for liaising with foreign KMCs. (2) Geographic regions within a country, known as NID_C. (3) RBCs, responsible for command and control messages to trains. (4) Trains, which operate across one or more NID_C regions.

The KMC then responds by issuing the two transport keys, kt_1 and kt_2, used for authentication and encryption respectively. The two keys are sent in the clear on portable media devices [19], to be installed.

On successful installation of kt_1 and kt_2, the KMC will generate unique keys for each train and RBC pair with identified through their IDs oid and rid respectively. In order to authenticate messages exchanged between them, each entity receives a collection of these keys. We will denote the set of keys with which OBUs and RBCs are provisioned as KM. KM may be initially empty (in the case of a new train or RBC), but it keys will be added as the train is authorised to communicate with relevant RBCs.

We provide an algorithm for the GEN.KMAC(id, id',knid_c) function from Definition 3.1 in Algorithm 1. This algorithm takes as input two IDs, id and id' corresponding to the two entities which are authorised to communicate with one-another and returns a randomly generated 3DES key to be used for message authentication in the EuroRadio protocol. For example, the key between a train with oid and an RBC with rid is: $km_{rid,oid} \leftarrow SGen(1^\lambda)$.

Algorithm 1: Offline ERTMS key generation

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

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

5 TRAKS - A UNIFIED ERTMS KEY MANAGEMENT SCHEME

In this section we propose TRAKS, our efficient key generation scheme for ERTMS. Our scheme uses the existing partitioning of the rail network into individual zones of control, denoted by a national identifier, which is announced to the train by EuroBalises. Using this partitioning, we enforce a permissions model which ensures that trains are not able to operate outside their agreed areas

of operation. TRAKS improves on the offline ERTMS scheme by allowing the key generation to be dynamic for RBCs thus improving the interoperability and reducing operational complexity for cross-border operations, while maintaining static key provisioning to trains. This provides maximum backwards compatibility with the current standards and practices.

In addition to the reduced management overheads, our scheme also provides authentication to messages both between OBUs and RBCs, and between OBUs and EuroBalises (EBs) making it universally applicable to any rail entity.

5.1 Secret Generation

In the rail network for a given country, there will be one or more regions. Each is identified through a public value, $\texttt{NID\_C}_i$ for the region with NID_C value i. For each $\texttt{NID\_C}_i$, the KMC will generate a random $knid\_c_i$ secret. As such, multiple secrets can be produced by the KMC to partition the rail network into geographic zones, and/or to establish trust between entity types, e.g. keys between OBUs and RBCs are generated using $knid\_c_1$ and keys between OBUs and RBCs are generated using $knid\_c_2$ for zone $\texttt{NID\_C}_1$ and $\texttt{NID\_C}_2$ respectively'.

Similarly to the offline ERTMS scheme the TRACKS shared secret is generated using:

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

where $\text{SGen}(1^\lambda)$ is a pseudo-random number generator (PRNG) with the security parameter λ. Unlike the offline ERTMS scheme, however, this secret is never given directly to OBUs or RBCs. It is instead used together with IDs to generate the message authentication keys. This approach greatly enhances the usability of the scheme by reducing the overall management overhead (i.e. secret key material storage, distribution and disposal). In the following, we will detail how we use this secret to generate the authentication keys for each ERTMS entity.

5.2 Key Generation

Algorithm 2: TRAKS key generation

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

1 **function** GEN.KMAC(id, id', s)
  /* for computing keys using $s = knid\_c$ */
2 **if** $id \neq null$ **then**
3 $km_{id,id'} \leftarrow PRF(id, s)$;
4 **if** $id' \neq null$ **then**
5 $km_{id,id'} \leftarrow PRF(id', km_{id,id'})$;
  /* for 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** $km_{id,id'}$

Keys in TRAKS are generated using Algorithm 2, using $knid\_c$ and the IDs of the communicating ERTMS entities. Algorithm 2 can be used to generate both (1) static keys which can be used to directly authenticate messages between two entities with identities id, and id' and (2) dynamic keys which can be combined by the holder with any id' to derive a static key. We continue with concrete descriptions of how keys are generated for OBUs, RBCs and EBs using TRAKS.

OBU Key Generation. OBU keys are static keys which are entirely computed by the KMC. Similarly to the offline ERTMS scheme, when a train identified by ID oid is authorised to operate on a specific line it is provisioned with a set of keys $KM = \{km_{rid1,oid}, km_{rid2,oid}, \ldots, km_{ridn,oid}\}$ where $rid1, \ldots, ridn$ are the IDs of the RBCs which control the line.

Each key $km_{ridi,oid}$ is computed using GEN.KMAC($ridi, oid, knid\_c$) detailed in Algorithm 2. First the secret $knid\_c$ is combined with the RBC's ID $ridi$ using a PRF to generate an intermediate pseudo-random value (line 3). This intermediate value is subsequently combined using the same PRF function with the oid to create the final $km_{ridi,oid}$ key (line 5).

The PRF can be any secure, non-malleable function, which is proven secure against length-extension attacks, such as HMAC-SHA-256 or AES-CMAC. It is easy to see that until now, our scheme is fully backwards-compatible, as there are no changes to how the keys are managed by the train.

RBC Key Generation. In TRAKS, keys that are provisioned to RBCs are dynamic. This means that RBCs are able to produce keys and communicate with any train for which they have an OBU ID. Under the current ERTMS implementation, trains are required to broadcast their oid in plaintext as part of the EuroRadio handshake when communicating with a RBC. This enables TRAKS to seamlessly replace the existing ERTMS scheme with only minor modifications to the RBCs internal programming.

Algorithm 2 is also used to generate RBC keys. Unlike in the case of the trains where OBUs are provisioned with a set of keys, the RBCs only have one key, $km_{rid,null}$, computed by running GEN.KMAC($rid, null, knid\_c$), where rid is the RBC's own ID. The value returned is the intermediate value computed in line 3.

RBCs can use $km_{rid,null}$ by running GEN.KMAC($null, oid, km_{rid,null}$) for any broadcasted oid to compute the message authentication key $km_{rid,oid}$.

It is easy to observe that generating key $km_{rid,oid}$ based on the $knid\_c$ secret can be used to enforce the permissions model explicitly. Any OBU and any RBC which are able to complete the EuroRadio handshake protocol must have been explicitly approved to operate in a given NID_C, as opposed to the current scheme, where this is implicit and therefore could be violated.

EuroBalise Key Generation. Another significant benefit of TRAKS is that it can be used without modifications to generate authentication keys for any pair of ERTMS entities. Here, we will try to exemplify this using EuroBalises (EB), however the scheme should work with other, possibly not yet developed, rail entities. Currently, EuroBalises offer no cryptographic protection of their payloads, and are implicitly trusted by the train for accuracy and validity.

The process of generating the unique authentication keys for balises is similar to the one used to generate the keys for the OBU-RBC pairs.

The KMC generates a balise specific shared secret as $kbls \leftarrow \mathsf{SGen}(1^\lambda)$ which ensures that the shared keys between EBs and OBUs are completely separate from the ones shared between OBUs and RBCs and are generated by each the KMC responsible for each country. However, unlike messages sent by trains or RBCs, messages sent by EBs are fixed. Furthermore EuroBalises are unable to perform any cryptographic operations. As such, when using TRAKS between OBUs and EBs, the balises will be provisioned with their fixed messages concatenated with the corresponding MAC. The MAC will be computed using a static authentication key generated by running GEN.KMAC(NID_C, *bgid*i, *kbls*) for each balise group *bgid*i under the control of NID_C. A balise group is a collection of balises concentrated in a common geographical area, for example, deployed along a rail line. We recommend the use of groups of balises instead of single balises to ease deployment. However, we note that the scheme would work similarly if balise group IDs would be replaced with balise IDs. More details are provided in Section 9.2.

Following key provisioning to a balise, the corresponding keys for NID_C, which are used for provisioning OBUs, can be computed as follows:

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

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

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

6 SECURITY ANALYSIS

In this section, we formally discuss the security of ERTMS key generation using a game-based approach.

We begin by stating that the attacker is given access to all IDs that can be generated from EDB. The attacker wins if it is able to generate a valid key $km_{id,id'}$ for any pair (id, id') for the types t and t', where $id \in \mathsf{INIT.ID}(EDB, t)$ and $id' \in \mathsf{INIT.ID}(EDB, t')$. We define the ERTMS KMAC security with a game played between a challenger C and an adversary $\mathcal{A}$ as follows.

Definition 6.1 (Key indistinguishably from random). Let KMAC = (SGen, INIT.ID, GEN.KMAC) be a scheme over a database EDB with security parameter λ, t and t' are entity types with $t \neq t'$, and $b \in \{0, 1\}$. We consider $Exp^b_{\mathcal{A}}(\mathsf{KMAC})$ (see Fig. 4), a probabilistic experiment played between an adversary $\mathcal{A}$ and a challenger C consisting of:

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

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

$ID \leftarrow \{i | i \in \mathsf{INIT.ID}(EDB, t)\}$
$ID' \leftarrow \{i | i \in \mathsf{INIT.ID}(EDB, t')\}$
$s \xleftarrow{R} \mathsf{SGen}(1^\lambda)$
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
$b' \leftarrow \mathcal{A}\left((km_{id,id'})_{id \in ID, id' \in ID'}, ID, ID'\right)$
return b'

Figure 4: ERTMS key derivation security game for an adversary $\mathcal{A}$.

The advantage of an 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 the advantage is negligible in λ.

In the following, we will show that both the current ERTMS scheme KMAC_{ERTMS}, and our proposed scheme KMAC_{TRAKS}, produce keys that are indistinguishable from random.

Key Indistinguishably for KMAC_{ERTMS}. It is easy to observe that the advantage of the adversary is based on its ability to distinguish the output produced by GEN.KMAC from random. However, the protocol specification (see Algorithm 1) shows that the key $km_{id,id'}$ for any pair (id, id') is already randomly sampled. Thus, the advantage of $\mathcal{A}$ to distinguish $km_{id,id'}$ from random, $|\Pr[Exp^0_{\mathcal{A}}(\mathsf{KMAC}_{ERTMS}) = 1] - \Pr[Exp^1_{\mathcal{A}}(\mathsf{KMAC}_{ERTMS}) = 1]|$, is negligible in λ.

Key Indistinguishably for KMAC_{TRAKS}. Based on the protocol specification (presented in Algorithm 2) there are three cases for GEN.KMAC.

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

(1) An intermediary key is generated by running the PRF 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 run that also takes as input id'. If the PRF is secure then this output will also be indistinguishable from random.

Case 3. If identifier id' is provided together with the secret $km_{id,null}$ as inputs to GEN.KMAC then the key $km_{id,id'}$ is generated as in the *Case 2.2* above, i.e. the value $km_{id,null}$ is used as an input, together with id' to a PRF. As $km_{id,null}$ is indistinguishable from random (see *Case 1*), if this PRF is also secure then the output will be indistinguishable from random.

Thus, the advantage of $\mathcal{A}$ to distinguish any $km_{id,id'}$ for TRAKS, $|\Pr[Exp^0_{\mathcal{A}}(\text{KMAC}_{TRAKS}) = 1] - \Pr[Exp^1_{\mathcal{A}}(\text{KMAC}_{TRAKS}) = 1]|$ will also be negligible in λ for any pair (id, id').

7 THE TRAKS KEY MANAGEMENT LIFECYCLE

In this section, we outline the key lifecycle for TRAKS, beginning with how the keys will be used and the changes required to support TRAKS. We then look at other aspects, for example key revocation and disposal of RBCs and trains.

Key Usage. Under our proposal, the RBC will be in an improved position, whereby it does not require individual keys to be assigned to it for a particular train. It is now able to derive its own symmetric keys based on some public information broadcast by the train, namely the ETCS ID. As a result, an RBC only requires a single key to be installed on it, which it uses to derive the keys which will correspond to those of the train. In the event that the RBC software does not support dynamic key generation, it may proceed using the same set of keys that would be allocated to trains for that RBC ID. Within the EuroRadio protocol, no changes are required to the core protocol that takes place, other than the way that the lookup of the key takes place. Currently, the RBC will use its ETCS ID and look for a corresponding entry that pairs the RBC to the Train ETCS ID and vice versa for the train. If an entry exists, it will use that key, otherwise, the EuroRadio protocol will fail as no key could be found. For our proposal, the RBC instead uses the $km_{rid,null}$ key, to derive the key which it can then use to compute a symmetric session key for the MAC authentication in EuroRadio.

For the train, when keys are allocated, the KMC will look up a set of the RBCs within the NID_Cs specified as authorised, and carry out the computation of $km_{rid,oid}$ for each RBC. As with the current scheme [19], the train will receive a set of keys, which, upon the receipt of the the first response message from the RBC in the EuroRadio handshake (the AU2 message), can select the appropriate $km_{rid,oid}$ key to use.

Revocation. In this scheme, trains may have their keys revoked using the 'DELETE_KEY' command, issued by the KMC, and likewise for the RBC. A blacklist, which would be made available for entities to download during maintenance or stabling would inform the train of RBCs it may no longer engage with, and vice versa for the RBC. However, if there is cause to 'reintroduce' that particular ETCS ID, there is no solution but either to generate a new $knid\_c$, which is an intensive process to rekey all existing entities, or change the ETCS ID of the affected entity. This, however, should be considered a very rare event. Validity of keys would be governed under the validity of $knid\_c$, ensuring that keys are refreshed on a regular, enforced basis. In the event that a train or RBC are to be reintroduced, a new ETCS ID should be introduced, as we consider all keys issued to the original ID to be compromised and should not be eligible for reuse.

Disposal and End-of-Life. Keys should have a set lifetime, which is subject to further discussion and scoping. However, at the end of a key's validity, a 'DELETE_KEY' command should be issued to the entity to remove the key, and a confirmation received that the key has been deleted. At the point of end-of-life, in order to prevent cryptographic material being leaked, a 'DELETE_ALL_KEYS' command should be issued by the KMC, where the entity confirms receipt of the command, prior to deleting its kt keys. During this process, the KMC should retain a copy of the keys, but mark them as invalid, to prevent any keys pertaining to that entity from being issued. If the entity operates across borders, the appropriate Key Management Authorities responsible for cross-border keys should be requested to initiate their appropriate 'DELETE_ALL_KEYS' command. After this point, the entity in question may be disposed of.

7.1 Discussion

The solution presented in this section has been engineered to be backwards-compatible, such that infrastructure managers may immediately begin to shift to this alternative solution with no changes to the internal way in which EuroRadio operates. All changes here may be performed in software, with most of the code changes lying on the Key Management Centre and RBC interfaces.

That said, our solution can work in both offline and online situations, where in the offline mode, the RBC requires one initialisation and installation of its derivation and kt keys, compared to a visit by an engineer each time a train is introduced to the network and requires 'approval' to communicate with this particular RBC. For trains, however, the keys at the point of manufacture may be installed at the same time, whilst updates may take place whilst the train is stabled and undergoing maintenance. If a train were to be 'moved' and made operational in a different geographic region[3], then the blacklist for the 'previous' RBCs should be updated to contain that train unless it was going to continue operating in that region. Likewise, a 'DELETE_ALL_KEYS' command should be issued if the train will no longer operate in a given area.

In the online method, when an RBC comes online for the first time and has carried out the appropriate TLS handshake and client authentication with the KMC, the KMC will simply issue the derivation key and appropriate constraints, as set out in SUBSET-038 [17]. For a new train, however, the process simply is an online version of the offline system, where instead of physical media being used, an online connection is used. Overall, this solution provides the additional protection that if the Key Management Centre went offline, the entities still retain sufficient keying material to carry out the EuroRadio Handshake. A train should not be allowed to make a journey for which it does not have the appropriate keying material to proceed. This prevents a situation where the GSM connection between the train and RBC is instead redirected to an RBC in another area, where an unsafe instruction may be issued.

We should also consider the case of existing trains which are currently operational. Under the existing offline model, each train has a unique key shared with the appropriate RBC on its 'allowed routes'. This provides an challenging operational concept, where

[3]For example, the West Midlands franchise in the United Kingdom is likely to receive 17 of the former Northern franchise British Rail Class 323 electrical multiple units.

during the process of national deployment, the line secret may be set up and appropriate derivation keys established. This means that as RBCs are introduced, no additional keys need to be introduced to trains, and they may immediately be in a position to communicate with the RBC. However, once a national deployment is complete, the process of adding additional keys is expensive, and there are few storage benefits to having few keys. Thus, the KMC may be able to 'precompute' the keys for the train, where the RBC is still required to carry out the appropriate calculations.

For existing lines and entities, their keys will remain valid until the validity period expires, in which course, a replacement key will be issued based on this specification.

Identified changes to the EuroRadio Protocol (in software) and underlying infrastructure are:

- Modification of the way that an RBC looks up keys: it will not perform a lookup, rather it will simply carry out the appropriate key generation computation (as per Algorithm 2) for the key that will allow it to communicate with the train. Trains will continue to be statically keyed, with no changes required.
- Establishing 'line secrets' that correspond to a specific NID_C and appropriately generating $km_{rid,oid}$ keys for RBCs and trains based on this principle.

8 KEY DISTRIBUTION

In the current ERTMS distribution scheme, the transport key is issued by the KMC in the clear, with a requirement on the intermediary to use trusted media (ranging from CD to USB storage). During national deployment, there will be a significant number of transport keys being issued, with the risk that multiple transport key payloads will be stored on the same media[4]. As the transport key is installed in the clear, an attacker could intercept the transport keys at the point of installation.

Once the attacker has the transport keys, they are able to intercept the encrypted keys from the Key Management Centre when they are installed on the train or RBC. For a train, this means that the attacker can determine the session key with relative ease if they observe the EuroRadio handshake protocol between the train and an RBC. In the case of an RBC, however, this means that an attacker could initiate arbitrary EuroRadio sessions with valid MACs, or even worse, send arbitrary messages to a train with a valid MAC, once the nonces have been obtained which the train would accept. The current scheme is not secure due to this threat, and we propose a new key distribution scheme as follows:

Manufacture/Commissioning. The train/RBC creates its own pair of transport keys, kt_1 and kt_2, using a hardware security module, 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 or RBC. We place a requirement that the vendor has a secure supply chain and the keys are securely generated and handled. If the vendor keys are compromised, new trusted keys would need to be installed onto every train/RBC from that vendor through their update mechanism.

[4]The RBC and OBU selects a specific file from the media for its payloads, typically bound using its ETCS ID[19].

Key Requesting from Infrastructure Manager. The vendor/operator generates a request to the national infrastructure manager (KMC) for the appropriate keys to be installed, providing the transport keys generated by the train or RBC. This request is encrypted and MACed with a pair of symmetric keys issued by the KMC to the 'approved' vendor or operator, $k_{v,kmc1}$ and $k_{v,kmc2}$. As a vendor may be responsible for manufacturing trains and RBCs for use in different countries, they determine the appropriate countries that keys are required for and submit requests to the appropriate infrastructure managers.

KMC Processes Request. The KMC verifies the request, as per nationally agreed procedures and, if valid, generates the appropriate $km_{rid,oid}$ keys as required, encrypted and accompanied by a MAC using the transport keys given and stored as part of the request. The scheme accounts for two cases, dynamic key generation ability and static key usage. For each case:

- **Ability to dynamically generate keys.** The Key Management Centre computes a 'master derivation key', i.e. $km_{rid,null}$ as per Algorithm 2, with no additional identity parameter included. It then encrypts and generates a MAC using the transport keys, before issuing the result to the vendor/operator.
- **Static key usage.** The Key Management Centre identifies the keys required, and generates a set of keys based on the NID_Cs provided in the request. This is done through repeated runs of Algorithm 2, with the appropriate $km_{rid,null}$ for the set of RBCs located in the appropriate NID_C that the train requires keys to authenticate to them. These keys are encrypted then MACed using the transport keys, prior to issuing them to the vendor/operator.

Key Installation. The vendor/operator then installs the payload of km keys onto the train/RBC where it will verify the MAC and decrypt the keys before installing/performing the maintenance instruction as required. As part of this, the train or RBC will generate a response to confirm the keys are installed, "KEYS INSTALLED", which includes a hash of the database of keys installed on that entity, and encrypts then MACs the response with the transport keys shared between the train/RBC and KMC.

Vendor Commissioning Confirmation. The vendor/operator provides the response to the KMC which verifies it using the transport keys and compares its local key database hash (corresponding to the train/RBC) to ensure that it matches the one in the response. If there is a mismatch, the protocol will require the issuance of keys to take place again.

We provide an example issuance protocol for a new train under the TRAKS scheme in Figure 5.

8.1 Considerations

In TRAKS, revocation is considered to be an exceptionally rare event, compared to the regular revocation process we see in public-key revocation with certificates. In the event that a key becomes compromised on a train, the infrastructure manager would be required to allocate a new ETCS ID to the train in question and provide new keys. For an RBC, the same process would take place, where

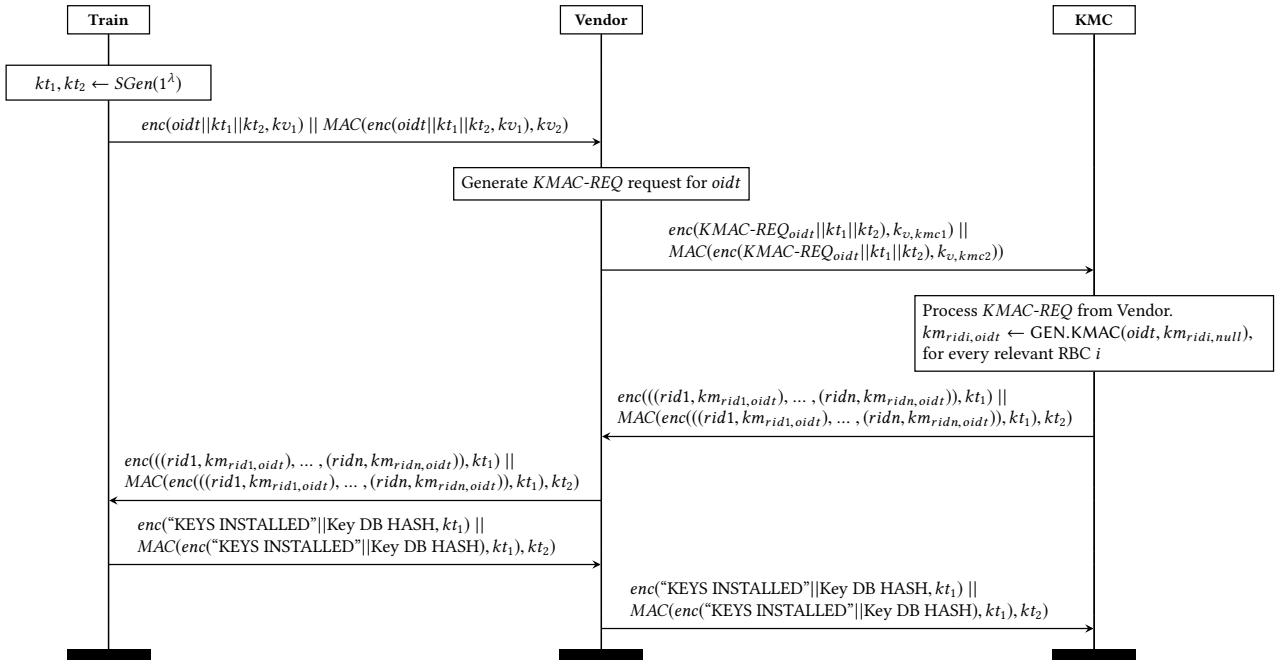

Figure 5: TRAKS Key Issuance Protocol for a new train t. The train is responsible for generating its own transport keys before encrypting and calculating a MAC using keys kv_1 and kv_2 respectively. The payload is submitted to the vendor as part of the key request to the KMC. The KMC processes the request, generating the necessary keys, where they are then installed on the train. The train is required to submit a checksum of the key database to ensure that the keys held on the train/RBC match what the KMC asserts should be installed. A similar process is run for new RBCs, as described in Section 8.

an engineer interacting with the affected unit is an acceptable overhead. That said, the current ERTMS specification and guidance from National Standards Bodies, for example RSSB [14] does not recommend quantitatively how long keys should be valid for, beyond 'the most time the key is required'. As part of TRAKS, we recommend a regular rekeying interval to prevent attacks, such as those presented in [4]. Rekeying can be performed during maintenance intervals, with trains at depots to ensure safe installation of keying material. We do not see the blacklist, maintained and distributed by the KMC, to impose considerable burdens on infrastructure managers and train owners.

For RBCs, which are already connected to a network to allow intercommunication with other RBCs, it may be possible to move the RBCs to communicate with the KMC online, and at regular intervals, whereas for trains, the maintenance would happen offline at the depot.

9 APPLICATIONS OF TRAKS IN A SECURE ARCHITECTURE

In this section, we present applications where TRAKS may be used and the considerations when implementing the scheme. We demonstrate how TRAKS may be used in EuroRadio, before showing how EuroBalises, which are trusted for location and speed/track profile data can be protected with a cryptographic MAC, before highlighting other applications outside of ERTMS.

9.1 EuroRadio MAC Keying

In this subsection, we will discuss the implementation considerations of the TRAKS scheme for the EuroRadio handshake protocol.

We require the use of a pseudo-random function, *PRF*, in the TRAKS framework. For a National Infrastructure Manager, this could be an HMAC function, such as HMAC-SHA-256, which is believed to be post-quantum secure. For key management operations, where the keys are in transit, a similar keying mechanism should be used between the KMC and approved organisations. For example with vendors and train operators, to ensure that the security of the keying material is never compromised. Given the current scheme relies on 3DES keys, a subset of the 256-bit output can be used as the 3DES key, until support for 256-bit keys is implemented.

We also recommend the use of a blacklist, which should be signed or have an accompanying MAC which can be retrieved on a regular basis, to ensure that, in the event of revocation, RBCs and trains do not interact with possibly compromised infrastructure.

9.2 EuroBalise Payload Security

Currently, the data held on EuroBalises is protected with a CRC [21], which allows an attacker to define their own balise payload, increasing line speeds or creating a 'virtual' block which overlays two sets of balise groups (BGs). This data is trusted by the train to be accurate and is used in position reports to the RBC, which in turn is used for the safe supervision of the network.

Generally, 'read-off' components are secure in providing broadcast information in plaintext when used in conjunction with a verified component. However, when used as part of a trusted network to make safety-critical decisions, it is not acceptable to have only add a checksum to the data. This only provides error-detection and does not defend against an attacker who is capable of either manipulating or emulating a balise, as they can be arbitrarily forged. Therefore, the payloads on the balises should be authenticated. Balise payloads, with the exception of ETCS Level 1 deployments, are static in data, which can therefore be signed or be accompanied with a MAC. We use the TRAKS framework to define a Key Management Scheme, which provides a 'per-balise' MAC Key, ensuring that the compromise of one balise-specific key does not enable the attacker to impersonate other balises in the network.

At the root of TRAKS, the Key Management Centre generates a national balise secret, e.g. for Great Britain, km_{GB}, which is used to generate all subkeys which are linked to the NID_C of the line. This produces $km_{\texttt{NID\_C},bgid}$ which is installed on all OBUs allowed to operate within a given NID_C. When the train passses over the balise and reads the telegrams from it, it can derive the MAC Key using the Balise Group ID, and verify the MAC. In the case the MAC fails verification, a 'violation' is recorded, so that the infrastructure manager can attend to the faulty balise. In the event that the train is at fault, recording erroneous 'violations', it is expected that issue is referred to the vendor for reactive maintenance.

Under this scheme, we add the requirement that all OBU units must have a trusted execution environment installed, for example Intel SGX and ARM TrustZone. This is necessary to prevent that the set of $km_{\texttt{NID\_C},bgid}$ keys can be extracted from the OBU and to ensure only authorised cryptographic operations can be performed using the keys. Balises do not have the $km_{\texttt{NID\_C},bgid}$ key installed on them at any time - the key is made available to the encoding units during maintenance periods, with the balise only containing the (plaintext, MAC) payload. No changes are required to the balise, as it does not carry out any computation.

While balise keys are always derived and never stored on any system, balise-specific keys provide defence in depth. This ensures that the compromise of one balise MAC does not compromise other balises, as the MAC key is specific to that particular balise. It should be noted, however, that EuroBalises can still be moved.

An alternative solution would be to implement a protocol where the train and balises are able to communicate in which MAC are computed in real-time. However, when considering the speed considerations at which balises are read, in excess of 300km/h, where three 'telegrams' from the balise group must be successfully read, implementing such a protocol would be difficult and an interactive authentication protocol would be infeasible. Therefore, providing a static payload with a MAC is far more efficient and less prone to errors in transmission. Assuming reading an unauthenticated/CRC-encoded 1023-bit balise message takes t milliseconds to read and process (e.g. 1.81ms for a train running at 500km/h, with no read failures, average conditions and data rates), adding TRAKS authentication would only increase this time by 6%. This effectively means that the impact is 1.92ms or 26.6cm by the time the train has read, derived the balise MAC key, and validated the MAC, which can be further mitigated with better hardware. That said, the benefits are considerable with minimal cost in time.

Currently, the CRC as well as other data used in integrity verification uses 110 bits of the total balise payload. If we replace this data with a 128-bit MAC, the available balise payload is reduced by only 18 bits. For the majority of balises ,this should not pose a problem, as Packet 44, used to send non-ERTMS messages from a balise to the train, is defined by the infrastructure manager, and typically will not fill all 1023 bits of balise payload.

9.3 ICS PLC Environments

The TRAKS framework may also be applied to other ICS environments, where a number of programmable logic controllers (PLC) devices in ICS/SCADA environments may communicate with each other and the communication protocol is not protected. This is, for example, the case with MODBUS/PROFIBUS. This lack of protection allows an attacker to carry out a man-in-the-middle attack and affect the operation of the system, potentially putting it into an unsafe situation.

A simple solution to this attack vector is to apply MACs to each message sent by a PLC. However, for an ICS owner, putting one key across the network achieves little security, where the keys could be extracted. Therefore, a partitioning of the system would allow specific 'zones' of control to be drawn, analogous to the NID_C variable that is used in ERTMS to define regions and lines. With this, a centralised key could be defined per zone and installed onto the appropriate hardware. A derivation step is then included on the controllers which would allow a per-device key to be used in the actual communications.

10 CONCLUSION

In this paper, we have presented a new key management solution which may be applied to a number of industrial control system environments. Using proven cryptographic techniques, we achieve an interoperable, backwards-compatible solution that can be used in ERTMS. It reduces management overheads for National Infrastructure Managers, and delivers post-quantum security. This scheme has further applications beyond EuroRadio, including EuroBalises, to ensure safety through security. By applying a partitioned system principle to ERTMS, we have been able to develop a key distribution scheme which maintains the same level of security in the system, whilst delivering significant benefits to the ICS owners and operators.

Acknowledgements. We would like to thank the UK's National Cyber Security Centre (NCSC) and the Birmingham Centre for Rail Research and Education (BCRRE) for their helpful discussion on technical and standardisation aspects. Funding for this paper was provided by the UK's Centre for the Protection of National Infrastructure (CPNI) and Engineering and Physical Sciences Research Council (EPSRC) via the SCEPTICS: A SystematiC Evaluation Process for Threats to Industrial Control Systems project.

REFERENCES

[1] banedanmark. 2011. *Fjernbane Infrastructure Easst/West BAFO Tender Document / Appendix 3.1 – Att 11 Online Key Management Concept.* Technical Report. Banedanmark. http://uk.bane.dk/db/filarkiv/9893/SP-12-041120-103.0111App_3.1_Attachment_11.docx

[2] GP Biswas. 2008. Diffie-Hellman technique: extended to multiple two-party keys and one multi-party key. *IET Information Security* 2, 1 (2008), 12–18.

[3] David Brandenburg, Erick Grünberger, and Manfred Grandits. 2010. *Effective ETCS Key Management with KEY.connect.* Technical Report. SIGNAL + DRAHT. http://www.tuev-sued.de/uploads/images/1347442379238610830293/article-keyconnect.pdf

[4] Tom Chothia, Mihai Ordean, Joeri de Ruiter, and Richard J. Thomas. 2017. 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 (ASIA CCS '17)*. ACM, New York, NY, USA, 743–756. https://doi.org/10.1145/3052973.3053027

[5] ERA. 2015. *Assignment of Values to ETCS Variables–v1.21.* Technical Report. European Union Agency for Railways. http://www.era.europa.eu/Document-Register/Documents/ERA_ERTMS_040001_v1.21.pdf

[6] ERA. 2015. *SUBSET-026: System Requirements Specification, version 3.5.0.* Technical Report. European Union Agency for Railways. http://www.era.europa.eu/Document-Register/Documents/SUBSET-026%20v350.zip

[7] Shailendra Fuloria, Ross Anderson, Fernando Alvarez, and Kevin McGrath. 2011. Key management for substations: Symmetric keys, public keys or no keys?. In *Power Systems Conference and Exposition (PSCE), 2011 IEEE/PES*. IEEE, 1–6.

[8] Shailendra Fuloria, Ross Anderson, Kevin McGrath, Kai Hansen, and Fernando Alvarez. 2010. The protection of substation communications. In *Proceedings of SCADA Security Scientific Symposium*.

[9] GSM-R Functional Group. 2014. *EIRENE System Requirements Specification, version 15.4.0.* Technical Report. European Union Agency for Railways. http://www.era.europa.eu/Document-Register/Documents/P0028D004.3r0.5-15.4.0.pdf

[10] Kevin Henry, Jiayuan Sui, and Ge Zhong. 2007. An Overview of the Advanced Access Content System (AACS).

[11] Kenji Koyama and Kazuo Ohta. 1987. Identity-based conference key distribution systems. In *Conference on the Theory and Application of Cryptographic Techniques*. Springer, 175–184.

[12] Florent Pépin and Maria Grazia Vigliotti. 2016. *Risk Assessment of the 3Des in ERTMS.* Springer International Publishing, Cham, 79–92. https://doi.org/10.1007/978-3-319-33951-1_6

[13] Prorail. 2012. *Gebruiksvoorschrift – ERTMS Key Management.* Technical Report. Prorail. https://www.prorail.nl/sites/default/files/gebruiksvoorschrift_ects_keymanagement.pdf

[14] RSSB. 2017. *Rail Industry Standard RIS-0743-CCS – ERTMS Key Management.* Technical Report. https://www.rssb.co.uk/rgs/standards/RIS-0743-CCS%20Iss1.pdf

[15] Wen-Guey Tzeng and Zhi-Jia Tzeng. 2000. Round-efficient conference key agreement protocols with provable security. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 614–627.

[16] UIC. 2015. UIC ERTMS Projects - UIC. (2015). http://www.uic.org/spip.php?article383

[17] UNISIG. 2012. *SUBSET-038 - Off-line Key Management FIS, version 3.0.0.* Technical Report. European Union Agency for Railways. http://www.era.europa.eu/Document-Register/Documents/Set-2-Index011-SUBSET-038%20v300.pdf

[18] UNISIG. 2015. *SUBSET-037 - EuroRadio FIS, version 3.2.0.* Technical Report. European Union Agency for Railways. http://www.era.europa.eu/Document-Register/Documents/SUBSET-037%20v320.pdf

[19] UNISIG. 2015. *SUBSET-114 - KMC-ETCS Entity Off-line KM FIS, version 1.1.0.* Technical Report. European Union Agency for Railways. http://www.era.europa.eu/Document-Register/Documents/Set-2-Index079-SUBSET-114%20v100.pdf

[20] UNISIG. 2015. *SUBSET-137 - KMC-ETCS Entity On-line KM FIS, version 1.0.0.* Technical Report. European Union Agency for Railways. http://www.era.europa.eu/Document-Register/Documents/SUBSET-137%20v100.pdf

[21] UNISIG. 2016. *SUBSET-036 - FFFIS for Eurobalise, version 3.1.0.* Technical Report. European Union Agency for Railways. http://www.era.europa.eu/Document-Register/Documents/SUBSET-036%20v310.pdf