Barcode Presentation, Key Management and Data Specification

Subject Ref: RSPS3001
Version: 02-03

Rail Settlement Plan Limited Registered Office, First Floor North, 1 Puddle Dock, London, EC4V 3DS
www.raildeliverygroup.com 020 7841 8000 Registered in England and Wales No. 03069042

Documentation Management

This documentation is published via the ASSIST website only.

The Version Control and Release Management of this documentation is managed by the Rail Delivery Group's Compliance Standards team (Compliance.Standards@raildeliverygroup.com).

To gain access to the latest documentation please visit the ASSIST website at www.rspaccreditation.org and request an account.

For accessibility purposes, Microsoft Word copies of this documentation are available if requested from RDG.

Copyright

Review Information

Version	Date	Author	Comments
02-02-A	06-Mar-2023	Matthew Pickman	The following JIRAs have been reviewed/addressed: STD-1548 Identification of sub-UTNs STD-1547 Section references wrong? STD-1538 Class code 9 STD-1537 Remove support for ITSO integration in RSPS3001 barcode STD-1534 Review Key Rollover Processes STD-1529 Change to barcode ticket & UTN STD-1506 Incorporate Bulletin RSPT0573 STD-1483 Sub UTN data STD-1455 Clarification to the PurchaseReferenceCode field STD-1392 Suggestion to store restriction codes on the barcode STD-1352 Errors in worked example STD-1336 Does the introduction of Flexi products impact on this document? STD-1254 Appendix to include rover (and packages) examples. STD-1240 Barcode key storage for additional devices/solutions
02-02-B	21-Apr-2023	Matthew Pickman	Address minor errors in appendices numbering
02-02-C	22-May-2023	Matthew Pickman	Address comments from External Review
02-02-D	06-June-2023	Matthew Pickman	The following JIRAs have been reviewed/addressed: STD-1656 Update RDG's registered/postal address
02-03	21-June-2023	Matthew Pickman	Version Issued

Release Control

The following personnel must formally approve the document prior to assigning a non-draft version number.

Organisation	Role	Name
RDG	Approval of Standards	Fares & Retail Delivery Group
RDG	Document Owner	James Wright
RDG	Subject Matter Expert	James Wright

Distribution

Organisation	Role	Name
TIS Suppliers	N/A	N/A
TOCs	N/A	N/A

Contents

Terms and Definitions

Term	Meaning
eTVD	Electronic Ticket Validation Database
ITSO	Standards and environment for interoperable smart ticketing schemes, or the media (Smartcard) holding tickets complying to these standards
IPE	ITSO Product Entity
IPEInstanceID	Unique ID for instance of an IPE
IPESeal	Digital Signature for IPE data structure
MAC	Message Authentication Code an ITSO field (not to be confused with MAC address a term used in computer networking)
EN1545	Standard for time and date fields used for ITSO, and adopted for the barcode data. ITSO documentation available at www.itso.org.uk
RFU	Reserved for Future Use
TIS	Ticket Issuing System
RDG	Rail Delivery Group
RSA	The Rivest-Shamir-Adleman cryptosystem for public-key encryption, named from the surnames of its inventors
SHA256	Secure Hashing Algorithm (256Bit) used to generate a hash for a message
UBI	Unique Barcode Identifier
UTN	Unique ETicket Number
URL	Uniform Resource Locator (commonly known as a website link or address)
XML	eXtensible Markup Language

1. Introduction

1.1 Purpose of document

1.1.1 This document is the RDG standard that defines the technical aspects of encoding rail ticketing information within a barcode.

1.1.2 This is applicable to any permanent product fulfilled as an 'mTicket', 'eTicket', 'Paper Roll Ticketing (PRT)' or 'sTicket'.

1.1.3 Adopting an industry standard is intended to support inter-availability between all interested stakeholders.

1.2 Background

1.2.1 Barcode fulfilment reduces retailing costs and aims to improve the customer experience.

1.2.2 The use of machine-readable barcodes provides an opportunity to:

- Introduce improved security checks to prove that the ticket is genuine;
- Automate on-train validation with a view to improving efficiency and accuracy;
- Obtain Management Information regarding travel patterns and train loadings;
- Protect against refund fraud.

1.2.3 RDG has developed this standard to ensure that ticketing which utilises barcode technology supports the retailing, fulfilment, validation of inter-available products, and that the validation equipment market is open to multiple suppliers.

1.2.4 RDG has developed this standard by considering:

- The data required for on-train ticket checking;
- The data required for ticket checks by a gate;
- The size of barcode that can be displayed on the screen of a mobile phone;
- Specification of existing rail ticket printers;
- Recommended levels of error correction;
- Feedback from implementers of barcode ticketing

1.3 Related Documents

Reference	Title	Usage
RSPS1024	Code of Practice - Interoperable Barcode Ticketing	Sets out the key aspects of the responsibilities and operational requirements for Interoperable Barcode Ticketing to support all parties' compliance with the terms of their licence.
RSPS2000	Barcodes in Rail Retailing	Provides details of the set of 'Barcode Types' utilised in rail retailing, basic technical information and references to related documentation.
RSPS3007	TIS Accreditation Requirements – Barcode Ticketing	Presents the accreditation requirements for the barcode component used in the fulfilment of National Rail tickets.
RSPS3013	Ticketing Specification – mTicket	Defines the approved layout to be used as a means of barcode fulfilment to a mobile phone device that includes visual security features.
RSPS3019	Ticketing Specification – Paper Roll Ticket	Sets out the specification relating to the fulfilment of rail products to Paper Roll Ticket (PRT).
RSPS3030	Ticketing Specification – eTicket	Sets out the specification relating to the fulfilment of rail products as an eTicket, to be displayed on a customer device and/or printed by the customer, and as a layout supported by various applications and operating systems.
RSPS3035	Ticketing Specification – sTicket	Defines the layout and acceptable delivery of sTicket fulfilment.
RSPS5043	eTVD Messaging Specification	Defines the required interface communication between various validation databases.
RSPS5045	Fares and Associated Data Feed Interface Specification	Describes in detail the Data Feed for the extraction of Fares information from the Data Transformation and Distribution Service (DTD).

2. Overview

2.1 Symbology

2.1.1 The ‘Aztec’ barcode symbology has been chosen for use on rail tickets as these barcodes have the best performance when used for paper and mobile phone tickets.

2.1.2 In line with general guidelines defined by most of the suppliers of barcode generation software, the error correction level should be set to provide the greatest level of protection within the limits of the maximum size defined above. This should be 23%, but may be higher, as long as the higher value does not cause the Aztec code to exceed 8 layers. Note that in practice, a higher level is normally achieved than that specified, as additional check characters are automatically applied to fill the layer.

2.2 Data Capacity

2.2.1 Currently only barcode ‘Type 06’ and ‘Type 08’ are accredited by RDG.

Both contain 233 characters, the first 15 forming a unique identity, and the remaining 218 contain the encrypted data. This assumes that no *additional data blocks* are present.

2.2.2 Type 04 and Type 11 are smaller 4-layer, 27 x 27 pixel Aztec codes containing 42 bytes of data. Due to their limited capacity, the objectives described in section 1.2.2 cannot be met and these types are no longer accredited by RSP. Barcodes in use in the rail industry of this type are encoded at the discretion of the issuing TOC.

2.3 Presentation Size & Print Quality

2.3.1 The physical size of an Aztec code is determined by the number of *Layers*. Payloads defined in this specification should be rendered as either 7 layers, being a 45 x 45 pixel square, or 8 layers, being 49 x 49.

2.3.2 To ensure a consistent physical size on printed media, each pixel should be a 0.5mm square, such that the resulting code fits within a 25mm (or 1 inch) square.

2.3.3 For non-printed media, the barcode size is defined in the relevant Ticketing Specification for its implementation.

2.4 Availability

2.4.1 Each fulfilment method that includes the rendering of a Type 06 barcode is controlled individually in RCS. This is because different characteristics apply to each fulfilment method, which may affect their suitability to a ticket type or flow.

2.5 Security

2.5.1 The current security measure for Type 06 is to encrypt the data with 1024 Bit RSA encryption and use an SHA256 hash. Please see section 3.3.3 for more details.

2.6 UniqueETicketNumber

2.6.1 The UniqueETicketNumber (UTN) uniquely identifies the ticket and must be displayed visually on barcode tickets.

2.6.2 The UTN (11 characters) is comprised of the IssuingSystemID (2 characters) and IssuingSystemUniqueETicketNumber (9 characters). Implementers should note that a different IssuingSystemID (and encryption key pair) must be used for the test and production environment.

2.6.3 The UTN is used as the barcode ticket identifier for the storage and distribution of scanning records. The best practice guidance is that each ticket should be uniquely identified by its UTN alone, especially if it is an inter-available ticket. Implementers should note that there may be instances where multiple barcodes are used for a given ticket, encoded with the same UTN.

2.6.4 For instance, return tickets are to be implemented as two barcodes, each with the same UTN, but with different Coupon Types for the Outward and Return directions of travel. Hence, for the instance of a return ticket, each barcode generated will have a Unique Barcode ID comprised of the combination of the UTN and the CouponType (Outward/Return) for the ticket. To cope with these Return coupons a gate or scanner system must allow for the use of the same UTN in each direction. Another example is where a sub-UTN is used (e.g. for sTicket fulfilment) and a new barcode is issued for the same product but with different VersionNumber and/or Checksum values.

2.6.5 Unique Barcode ID (UBI) uniquely defines the barcode according to the formula:

Unique Barcode ID (UBI) = (UTN) + (CouponType as letter)

2.6.6 CouponType maps to a letter as follows: 0=S (Single), 1=N (Season), 2=O (Outward), 3=R (Return).

2.6.7 Note that even though two barcodes may have the same UTN and CouponType, the data inside the barcode, and any optional data after the end of the sealed ticket data may be different. Therefore, it is important for a gate or scanner to interpret the data presented in the barcode each time it is seen, and to send the whole barcode payload back to the original issuer, as they may be tracking the different data inside and outside the payload.

2.7 Barcode Encoding Principles

2.7.1 The expected encoding for each type of product is recorded in the table below.

2.7.2 Detailed encoding information for products can be found in the ‘Type 06 Encoding Information’ spreadsheet as Appendix A of this document.

Product Type	Example	Barcode Media	Number of Barcodes Issued	Validity Period of each Barcode	Travel Permitted by each Barcode	Number of UTNs
Advance	Advance Single	PRT, mTicket, eTicket	One	Same as the product	One Journey	One
Single	Anytime Single	PRT, mTicket, eTicket	One	Same as the product	One Journey	One
Return	Anytime Return	PRT, mTicket, eTicket	One per direction of travel	Same as the product, for that direction	One Journey	One
Unlimited Use (up to 1 month)	Weekly Season	PRT, eTicket	One	Same as the product	Unlimited	One
		sTicket	Many	3 hours (Coupon)	One Journey	One

Product Type	Example	Barcode Media	Number of Barcodes Issued	Validity Period of each Barcode	Travel Permitted by each Barcode	Number of UTNs
				1 day (Backup Coupon)		
Unlimited Use (more than 1 month)	Annual Season	sTicket	Many	3 hours (Coupon) 1 day (Backup Coupon)	One Journey	One
Limited Use (by Days)	Flexi Rover – 3 days in 7	PRT	One	Same as the product	Unlimited	One
		sTicket	Many	3 hours (Coupon) 1 day (Backup Coupon)	One Journey	One
Limited Use (by Journeys)	Carnet	PRT, mTicket, eTicket	One per Journey	Same as the product	One Journey*	One per Journey
Group Product	Duo	PRT, mTicket, eTicket	One per passenger, per direction of travel	Same as the product	One Journey	One per passenger
Non travel product	Supplement	eTicket	One	Same as the travel ticket	Not valid for travel	One

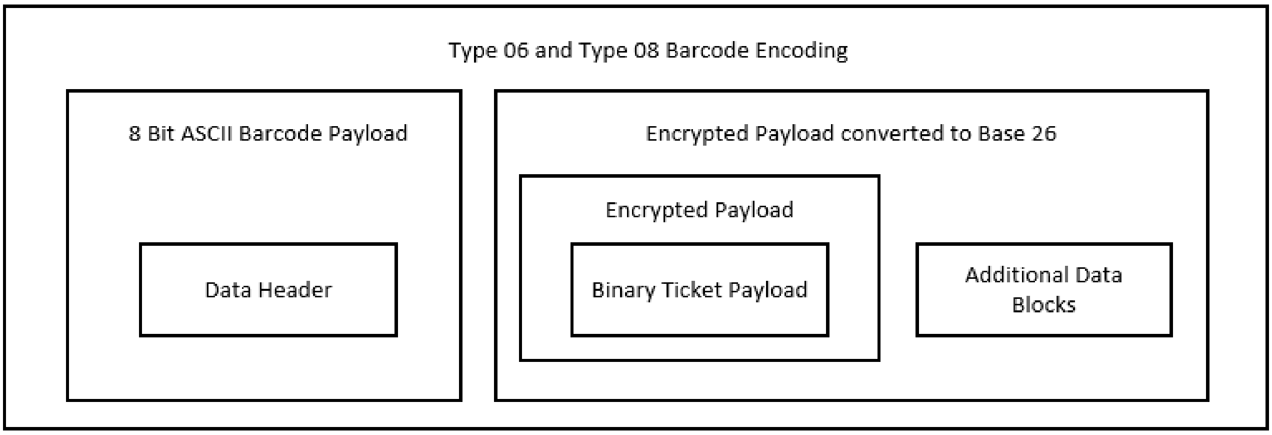
*Note: Barcode validation logic applies sequence rules to the outward and return directions of a return product, for example expecting an outward coupon to be used before a return coupon.

2.8 Barcode Checking Tools

2.8.1 A Type 06 or Type 08 barcode payload can be pasted into the ASSIST search box to generate a report on the barcode’s content.

3. Data Encoding

3.1.1 The diagram below shows the overview data encoding for Type 06 and Type 08 barcodes:


Type 06 and Type 08 Barcode Encoding
8 Bit ASCII Barcode Payload
Data Header
Encrypted Payload converted to Base 26
Encrypted Payload
Binary Ticket Payload
Additional Data Blocks

3.2 Data Header

3.2.1 Refer to the encoding specified in sections 4 and 5 for Type 06 and Type 08 barcodes respectively.

3.3 Binary Ticket Payload

3.3.1 Data Content

3.3.1.1 The data to be encoded within rail ticket barcodes (before RSA Encryption and encoding to Base26) is specified within the separate spreadsheet referred to as either Appendix A (for Type 06) or Appendix B (for Type 08) in this document.

3.3.1.2 All alpha characters should be stored as uppercase characters.

3.3.1.3 The data items included in the specification have been specified to reflect validation and authentication requirements. It is assumed that any further ticket details required for customer support functions such as refunds will be obtainable from the original product records retained by the issuing system.

3.3.2 6Bit encoding

3.3.2.1 All the current alphanumeric fields must be represented as 6Bit to store case insensitive letters and numbers using the Digital Equipment Corporation 6Bit character set.

3.3.2.2 This is simply the ASCII character codes from 32 to 95 coded as 0 to 63;

- To convert from 6Bit to ASCII, add 32 to the character value
- To convert from ASCII to 6Bit, subtract 32 from the ASCII value

3.3.2.3 Appendix E provides a 6Bit character table.

3.3.3 SHA256 Hash

3.3.3.1 The data that encodes the attributes of the ticket is protected by a SHA256 hash. The 108 data bytes are applied to the SHA256 algorithm as a raw byte string, resulting in a 32 byte binary hash. The most significant 8 bytes of this hash are appended to the 108 bytes of data, and the resulting 116 byte value forms the input to the encryption process.

3.4 Encrypted Payload

3.4.1 The security solution to prevent alteration to the data content is the use of RSA encryption of the data content.

3.4.2 The Binary Ticket Payload data is encrypted using the TIS private key, using 1024Bit RSA PKCS#1v1.5. (This is a freely available public standard). This protects a payload of up to 116 bytes, or 928Bits, creating a 128 byte or 1024Bit encrypted output. The RSA encryption algorithm adds a header of length 12 bytes to the payload before encryption to improve security. For **decryption** purposes, this header should be removed to get the correct payload. The format of this header is:

0x00 [10 non-zero bytes which are padding] 0x00 [content]. This is often done automatically by encryption/decryption software.

3.4.3 The purpose of applying the RSA encryption is to digitally sign the payload, and not to hide it. Some libraries may not provide the ability to encrypt under the private key, the more normal usual function being to encrypt under the public key, in order to hide the content from all but the holder of the private key. This type of digital signing is similar in principal to that described in the public standard PKCS#7, but without the hashing mechanism.

3.4.4 This encrypted output cannot be modified by anyone not holding the private key but can be read and validated by anyone who has the freely available public key. The use of asymmetric encryption simplifies the key distribution to other participants and protects the secrecy of the private key used to generate genuine tickets.

3.5 Additional Data Blocks

3.5.1 Additional data can be added after the encrypted portion of the payload, and can be ignored by most systems, but should be stored and reported back to the issuing system in the eTVD records. Scanning systems should not fail to successfully process a barcode that exceeds the expected length but should attempt to decode and decrypt the earlier parts of the barcode if they are present.[1]

3.5.2 Because gate throughput speed is of paramount importance to effective operation and safety, a maximum practical length for the entire barcode is 300 characters and implementers should seek to minimise the overall length of the payload. Scanning systems should be tested with barcodes containing 300 characters to confirm scanning time is within acceptable limits.

3.5.3 The additional data blocks after the encryption are structured as a concatenation of DATA_BLOCKs, without any separator bytes. Not to be confused with the former optional data section inside the encrypted portion of the barcode.

[DATA_BLOCK 1] [DATA_BLOCK 2] ... [DATA_BLOCK N]

3.5.4 Each DATA_BLOCK has a DATA_BLOCK_HEADER and one or more DATA_ELEMENTS. The number of DATA_ELEMENTS is only constrained such that the length of the DATA_BLOCK is less than the BLOCK_LENGTH defined in the DATA_BLOCK_HEADER.

[DATA_BLOCK 1] => [DATA_BLOCK_HEADER 1] [DATA_ELEMENT 1] ... [DATA_ELEMENT N]

[1] As of June 2019, RDG understand that existing validation systems will fail to process barcodes which contains any additional data blocks. If you plan to develop systems to make use of the Additional Data Blocks please seek advice from RDG as early as possible.

3.5.5 The DATA_BLOCK_HEADER determines how the content within the DATA_BLOCK is interpreted. The header format is as follows:

[DATA_BLOCK_HEADER 1] => [INTENDED_AUDIENCE_ID1] [DATA_BLOCK_LENGTH_1]

- INTENDED_AUDIENCE ID1 (2 bytes) : defines who is the intended audience for the DATA_BLOCKS
  - 0x0001 (hex) is the only DATA_BLOCK defined in this document, named “UnencryptedRsp6”
    - It is used as an unencrypted area for information which may change after encoding that should be visible to all TOCs, such as Activations and Walk-Up Reservations.
  - IDs in the range 0x0000 – 0x20ff are reserved for RDG internal usage.
  - IDs in the range 0x4100 – 0x5aff are reserved for TOCs (this should sensibly be co-ordinated by RDG)
  - All other IDs are available for anyone.
- SIZE (1 byte): determines the size of the envelope in Bytes that must enclose the contents of the DATA_BLOCK.
- Note that this does not include the 3 bytes taken by the DATA_BLOCK_HEADER itself.

3.5.6 The contents of the DATA_BLOCK can be organised in any way, however for any DATA_BLOCK intended to be inter-available, a format is defined comprising a list of one or more DATA_ELEMENTs concatenated without separators, each made up of three parts:

[DATA_ELEMENT 1] => [FIELD_ID 1] [FIELD_LENGTH 1] [FIELD_CONTENT 1]

- FIELD_ID (1 byte) : defines what FIELD is encoded in this DATA_ELEMENT. The scope of the ID is restricted to the INTENDED_AUDIENCE, such that each block co-exists in an independent FIELD_ID space.
  - For instance, for the “RESERVED FOR RSP” DATA_ELEMENT:
    - FIELD_ID 0x01 (hex) is defined as “Activation Date/Time”.
- FIELD_LENGTH (1 unsigned byte): determines the number of bytes taken up by the FIELD_CONTENT.
- For instance, for the “RESERVED FOR RSP” DATA_ELEMENT:
  - FIELD_ID 0x01 is “Activation Date/Time”
  - “Activation Date/Time” is encoded in binary according to EN1545 requires 4 bytes
  - Hence, the FIELD_LENGTH for this field would be 0x04 in hex.

3.5.7 Example of Additional data after the payload

Consider the following example in which there are two DATA_BLOCKs encoded:

1. The RSP-defined UnencryptedRsp6 block
   a. INTENDED_AUDIENCE_ID = 0x0001
   b. Content is a single field, Activation timestamp:
      i. FIELD_ID = 0x01
      ii. FIELD_LENGTH = 0x04
      iii. FIELD_CONTENT = 2008/10/27 14:27 = 0x10DD0363
         1. When encoded according to EN1545:
         2. 27/10/2008 = 4317 = 0x10DD
         3. 14:27 = 867 = 0x0363
   c. Hence DATA_BLOCK_LENGTH = 1+1+5 = 7
2. A TOC-specified block:
   a. INTENDED_AUDIENCE_ID = 0x4269
   b. Content is a pair of fields:
      i. First field:
         1. FIELD_ID = 0x00
         2. FIELD_LENGTH = 7 bytes = 0x07
         3. FIELD_CONTENT = 1,2,3,4,5,6,7 = 0x01020304050607
      ii. Second field:
         1. FIELD_ID = 0x01
         2. FIELD_LENGTH = 1 bytes = 0x01
         3. FIELD_CONTENT = 0x69
   c. Hence DATA_BLOCK_LENGTH = 1+1+7 +1+1+1 = 12 = 0x0C

3.5.8 The entire series of unencrypted blocks would require 25 bytes, and look like this (in hex):

UnencryptedRsp6 block	TOC’s block
000105010410DD0363	42690C00070102030405060701 0169

3.6 Base26 Encoding

3.6.1 Base26 encoding takes binary data (a byte array) and converts it into a stream of letters, drawn from a 26-character pool in capital letters.

e.g. byte arrays of 0xA5, 0x05, 0x4B = Base26 encoded value of “TDTTKA”
It’s similar to Base64 but with a smaller pool of characters

- Base64= ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/

- Base26= ABCDEFGHIJKLMNOPQRSTUVWXYZ

These 26 upper-case characters are assigned to the integers in alphabetic order: A=0, B=1, … Z=25. The use of exclusively upper-case letters ensures an efficient use of the Aztec code, which would otherwise be larger and of variable length.

3.6.2 The encoding scheme used differs from that used by standard base-26 libraries, and so must be custom built.

3.6.3 Decoding is the reverse of encoding, taking a string of capital letters and turning it back into a byte array.

3.6.4 The Base26 decoding algorithm may add an extra 0 byte at the end of the returned payload. The algorithm should check for this: typically, if the returned payload size is 129 (and not 128 which is the standard RSA encrypted payload size) and if it is equal to 0 (zero) remove it.

3.6.5 Appendix D contains pseudo-code examples describing the algorithm for Base26 encoding and decoding.

3.7 Process for Generating Barcode

3.7.1 The following describes the process to build and encrypt the data to form the barcode:

1. Encrypt Ticket Payload using the TIS Private Key.
2. Append any additional data blocks after this.
3. Encode the entire byte array (encrypted bytes + optional blocks) in Base26.
4. Append Data Header to the front of the Base26 encoded data packet.
5. Generate barcode with data content from last step.

3.8 Process for Ticket Scanning and Validation

3.8.1 The following describes the process to get to the original data elements and provides basic checks to perform:

3.8.1.1 Read the header to obtain the IssuingSystemID and retrieve the public key(s) for the issuer.

3.8.1.2 Decode the remainder of the payload from Base26 into a byte array.

3.8.1.3 The encrypted Ticket Payload will always be the first 128 bytes of this byte array. Decrypt using the freely available Issuer Public Key, corresponding with the IssuingSystemID. Issuer Public Keys are to be stored by the validation device.

3.8.1.4 Reject ticket if RSA decrypt fails or if the SHA256 hash inside the encrypted block does not match a freshly generated hash of the decrypted contents, or if the UTN is marked for rejection (such as after being refunded).

3.8.1.5 Check that the UTN specified within the encrypted data matches that specified in the header. This prevents anybody from altering the UTN in the header without detection.

3.8.1.6 Additional data may have been added after the encrypted portion of the payload. If there is a “RESERVED FOR RSP” DATA_BLOCK identified by the INTENDED_AUDIENCE_ID, then the unencrypted payload may be added to the encrypted payload and is used to inform gate or handheld scanner logic. Any other DATA_BLOCKs can be ignored by most systems, but should be stored and reported back to the issuing system in the upload of eTVD scanning records.

3.8.2 OPTIONAL Extra Anti-fraud steps to combat people copying tickets and using them against separate off-line scanners:

3.8.2.1 Lazy upload (i.e. when system can do so or at defined time of day/week) of scanned UTN’s and scanning time/location information back to the relevant Issuer to allow issuer to detect and block fraudulent (copying) users.

3.8.2.2 Occasional Live (networked) lookup check on issuing system database.

4. Barcode Type 06 – Large with Security

4.1 Applications

4.1.1 Type 06 is the recognised barcode to be used for travel products across National Rail services.

4.2 Data Encoding

4.2.1 The data to be encoded in a Type 06 barcode is to be stored in the following manner: a data header and an encrypted payload.

4.3 Data Header

4.3.1 The Data Header for a Type 06 barcode is comprised of:

Field	Notes	Example
BarcodeTypeIdentifier	Two-digit number specifying the type of barcode. See RSPS2000 for BarcodeTypeIdentifier's in use.	06
IssuingSystemUniqueETicketNumber	Nine-character field that must be unique for every ticket that has the same IssuingSystemID within a 5-year period. The value is calculated using the algorithm in Appendix C.	C8F23B8HG
Checksum	This field is to be set to '0', except where a sub-UTN is used (e.g. for sTicket fulfilment).	0
VersionNumber	The VersionNumber will always be '0', except where a sub-UTN is used (e.g. for sTicket fulfilment).	0
IssuingSystemID	Two-character code issued by RDG to identify the retailer of the barcode ticket. It defines which public key should be used to decrypt the data in the encrypted payload. The list of Issuing System ID's and their public keys are included in the RDG PKR. A separate IssuingSystemID will be assigned for each Lennon Machine Type, to ensure UTN uniqueness across TIS. A separate IssuingSystemID (and key) is to be used for test and production environments of TIS operation. An IssuingSystemID can be requested from RDG by contacting Tis.Accreditationcboservicedesk@raildeliverygroup.com with an email subject of "New IssuingSystemID Request".	MS

4.4 Encrypted Payload

4.4.1 See associated document:

RSPS3001 XX-YY-A[2] Type 06 Encoding Information.xlsx

[2] Where XX-YY refers to the related Subject Version and A refers to the Draft suffix character. Issued Versions do not have a Draft suffix character.

5. Barcode Type 08 - Large with Security

5.1 Applications

5.1.1 Type 08 is the barcode to be used for Railcards, and potentially other accompanying travel documents, such as photocards.

5.2 Data Encoding

5.2.1 The data to be encoded within Type 08 barcodes is to be stored in the following manner: a data header and an encrypted payload.

5.3 Data Header

5.3.1 The Data Header for a Type 08 barcode is comprised of:

Field	Notes	Example
BarcodeTypeIdentifier	Two-digit number specifying the type of barcode. See RSPS2000 for BarcodeTypeIdentifier's in use.	08
IssuingSystemUniqueETicketNumber	Nine-character field that must be unique for every ticket that has the same IssuingSystemID.	ABC123DEF
Checksum	Included for compatibility with the Type 06 barcode. This field is to be set to '0'.	0
VersionNumber	Included for compatibility with the Type 06 barcode. This field is to be set to '0'.	0
IssuingSystemID	Two-character code issued by RDG to identify the retailer of the barcode ticket. It defines which public key should be used to decrypt the data in the encrypted payload. The list of Issuing System ID's and their public keys are included in the RDG PKR. A separate IssuingSystemID will be assigned for each Lennon Machine Type, to ensure UTN uniqueness across TIS. A separate IssuingSystemID (and key) is to be used for test and production environments of TIS operation. An IssuingSystemID can be requested from RDG by contacting Tis.Accreditationcboservicedesk@raildeliverygroup.com with an email subject of "New IssuingSystemID Request".	RC

5.4 Encrypted Payload

5.4.1 See associated document for more information.

RSPS3001 XX-YY-A[3] Type 08 Encoding Information.xlsx

[3] Where XX-YY refers to the related Subject Version and A refers to the Draft suffix character. Issued Versions do not have a Draft suffix character.

6. Key Management

6.1 Requirements and Assumptions

6.1.1 The public keys are used for barcode ticket validation on remote equipment, both portable units and fixed gates, none of which can be considered to have reliable always-on network connectivity, so the system should run reliably even if some of these systems cannot go online every day, or if updates must be done manually or with human intervention.

6.1.2 The maximum expected lifetime of any ticket product or pass that is signed by keys in this system is 12 months (or possibly just beyond this period).

6.1.3 The ticket signing keys will not be the same as any ecommerce keys, and therefore can have a lifespan more than the 12 months stipulated by PCI-DSS.

6.1.4 The expectation is that the same set of keys will be held by all barcode validation equipment. This list is maintained and made available by RDG.

6.1.5 An issuer of barcode tickets must be accredited by RDG against the General Requirements of RSPS3007: 'TIS Accreditation Requirements – Barcode Ticketing' in order for their public key to be made available by RDG.

6.2 Key Sharing and Exchange Mechanism

6.2.1 RDG maintains a machine-readable XML page that contains a list of IssuingSystemID's and the corresponding public keys currently in use for inter-available barcode ticketing, as a one-stop place for devices or servers to retrieve keys from. The RDG Public Key Repository (PKR).

6.2.2 In addition, the RDG PKR contains the eTVD instances of operators and retailers, indicating where a Ticket Event should be directed to notify other participants - See RSPS5043: 'eTVD Messaging Specification' for more information.

6.2.3 The page is available from ASSIST under HTTPS connection, to prove the authenticity of the data on the page. Appendix G outlines the XML format for Key Sharing.

6.2.4 RDG PKR access requests are to be sent to TIS.Accreditation@raildeliverygroup.com with a subject of "Barcode Key Access Request".

6.2.5 It is expected that identifying changes to the feed will become an automated daily task. For an interim period, RDG will send an email notification to the nominated contact when changes occur.

6.2.6 An RDG Test PKR is also available and contains test Public keys only.

6.3 Key Lifespan

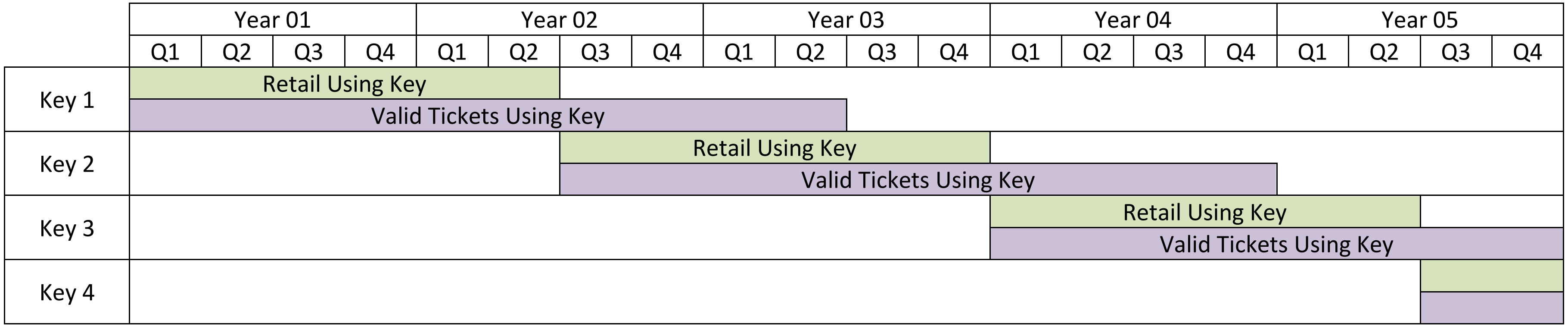
6.3.1 Each key has a 2.5-year lifespan, overlapping at 1.5 years.

6.3.2 A key should only start to be used to issue tickets following authorisation from RDG, as per the ‘Add a New Production Key’ process. A TIS Supplier must make allowance for this process when introducing a new barcode key.

6.3.3 It is acceptable for the same IssuingSystemID to be retained following key rollover. Decryption software must try all appropriate keys held for an IssuingSystemID, if decryption fails with the first key, then decryption is attempted with the next key, and so on. The expiry dates in the key are used to prevent the need to try an excessive number of keys.

6.3.4 To ensure that a ticket is accepted, the associated public key must be retained by validation devices until all valid tickets have been used or expired, which may be up to 12 months after retailing using that key has ceased.

6.3.5 The key rollover schedule is represented in the diagram below.



6.4 Add a New Production Key

6.4.1 The following process must be followed prior to retailing using a new barcode key in the production environment. This process is used for a new issuer of barcode tickets and for key rollover of an existing issuer.

6.4.2 Narrative

1. TIS Supplier provides RDG (TIS.Accreditation@raildeliverygroup.com) with the following information:

   - Email Subject “New Barcode Key Request – Production”,
   - Company Name,
   - TIS Name,
   - IssuingSystemID,
   - .txt file containing Public Key (as a PEM encoded X509 public certificate), and
   - Example barcode ticket payload (encoded using new key)
   - Public key start date
   - Public key end date
   - Indicate whether request is part of key rollover activity

2. RDG tests that the example barcode payload can be decrypted using the Public Key provided

3. Outcome of test is that Public Key can be decrypted successfully or not.

   3.1 If the barcode cannot be successfully decrypted, RDG informs TIS Supplier of rejection of new Public Key

   3.2 If the barcode can be successfully decrypted, RDG adds the Public Key to the RDG PKR.

      3.2.1 RDG instructs Validation Equipment suppliers add the new Public Key to their devices.

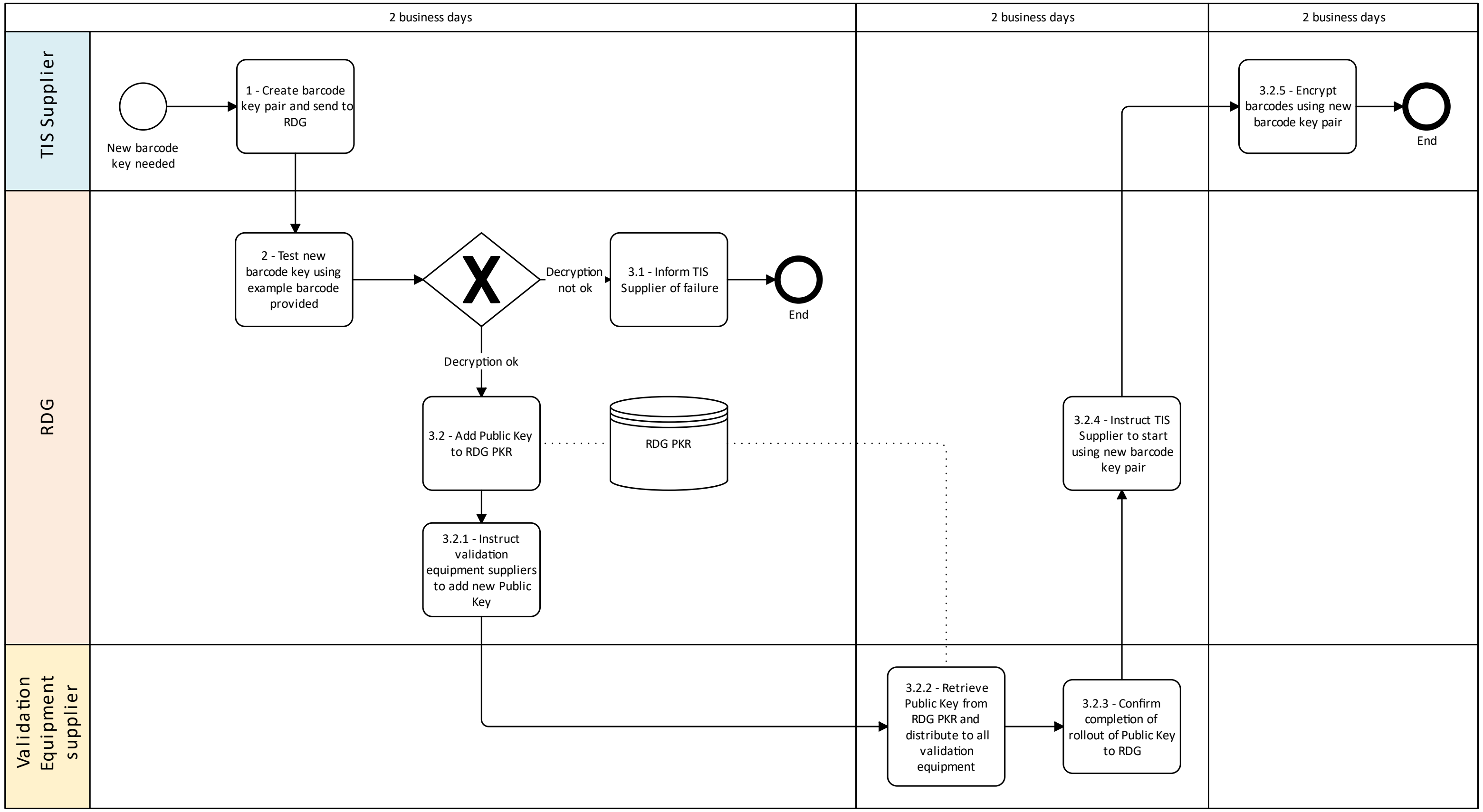
      3.2.2 Validation Equipment supplier retrieves new Public Key from RDG PKR and applies to devices.

      3.2.3 Validation Equipment supplier confirms download of new Public Key to all devices.

      3.2.4 RDG instructs TIS Supplier to issue barcode tickets using new Public Key.

      3.2.5 Barcode tickets are issued using new Public Key.

6.4.3 Diagram[4]


2 business days
2 business days
2 business days
TIS Supplier
RDG
Validation Equipment supplier
New barcode key needed
1 - Create barcode key pair and send to RDG
2 - Test new barcode key using example barcode provided
Decryption not ok
3.1 - Inform TIS Supplier of failure
End
Decryption ok
3.2 - Add Public Key to RDG PKR
RDG PKR
3.2.1 - Instruct validation equipment suppliers to add new Public Key
3.2.2 - Retrieve Public Key from RDG PKR and distribute to all validation equipment
3.2.3 - Confirm completion of rollout of Public Key to RDG
3.2.4 - Instruct TIS Supplier to start using new barcode key pair
3.2.5 - Encrypt barcodes using new barcode key pair
End

[4] The time expectations in the diagram are aspirational and may change.

6.5 Emergency Key Replacement

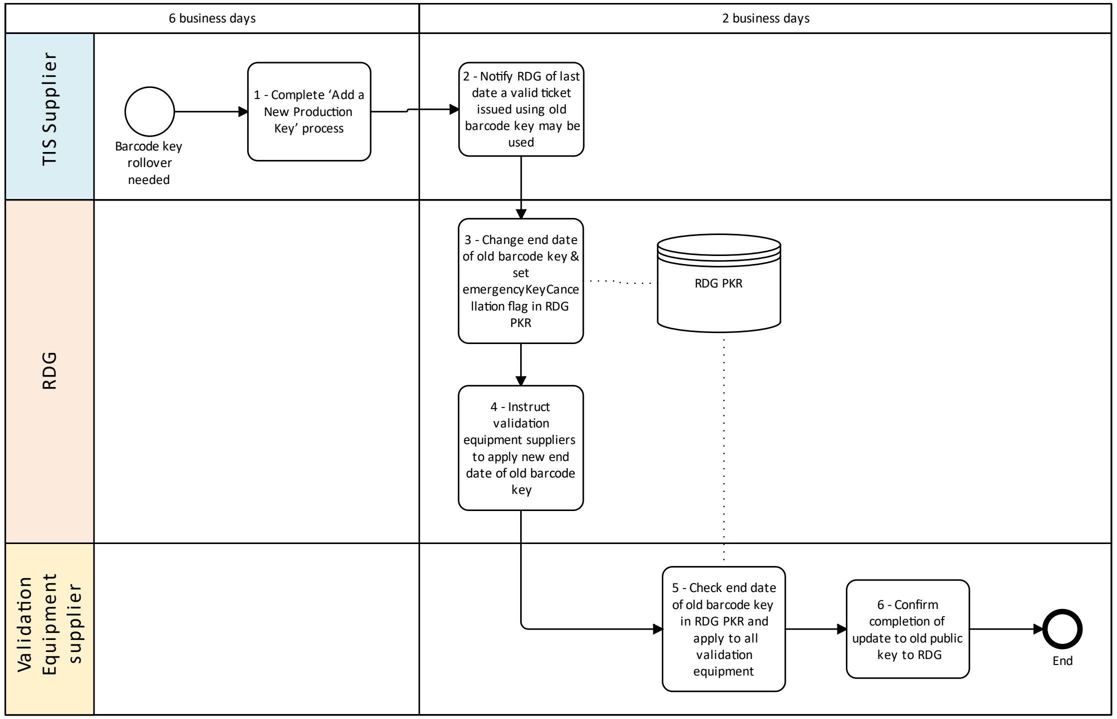
6.5.1 If a participant suspects that a private key currently in use has been compromised, and therefore an unauthorised party holds a private key with which to generate apparently valid tickets, then the following process is to be followed.

6.5.2 RDG reserve the right to suspend TIS retailing of barcode tickets using a compromised key during this process.

6.5.3 Narrative

1. The TIS Supplier initiates the ‘Add a New Production Key’ process.
2. When the ‘Add a New Production Key’ process is complete, the TIS Supplier informs RDG of the last date that a genuine ticket using the compromised key is valid for travel.
3. RDG updates the end date of the compromised key in the RDG PKR and sets the emergencyKeyCancellation flag.
4. RDG instructs Validation Equipment suppliers to apply the new end date for the compromised key to all devices.
5. Validation Equipment supplier obtains the new end date for the compromised key from the RDG PKR and updates all devices.
6. Validation Equipment supplier confirms update to RDG.

6.5.4 Diagram[5]


6 business days
2 business days
TIS Supplier
RDG
Validation Equipment supplier
Barcode key rollover needed
1 - Complete 'Add a New Production Key' process
2 - Notify RDG of last date a valid ticket issued using old barcode key may be used
3 - Change end date of old barcode key & set emergencyKeyCancellation flag in RDG PKR
RDG PKR
4 - Instruct validation equipment suppliers to apply new end date of old barcode key
5 - Check end date of old barcode key in RDG PKR and apply to all validation equipment
6 - Confirm completion of update to old public key to RDG
End

[5] The time expectations in the diagram are aspirational and may change.

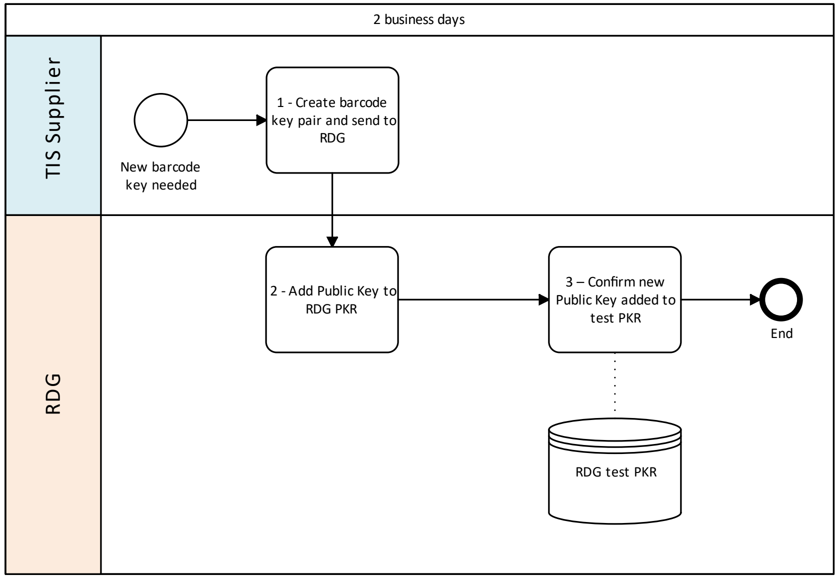
6.6 Add a New Test Key

6.6.1 The process for adding a test public key to the test instance of the RDG PKR is defined below.

6.6.2 Narrative

1. TIS Supplier sends the information below to TIS.Accreditation@raildeliverygroup.com:
    - Email Subject “New Barcode Key Request – Test”
    - Company Name,
    - TIS Name,
    - IssuingSystemID,
    - txt file containing Public Key (as a PEM encoded X509 public certificate), and
2. RDG add new test Public Key to test instance of RDG PKR.
3. RDG notifies TIS Supplier of addition of test Public Key.

6.6.3 Diagram[6]


2 business days
TIS Supplier
RDG
New barcode key needed
1 - Create barcode key pair and send to RDG
2 - Add Public Key to RDG PKR
3 – Confirm new Public Key added to test PKR
End
RDG test PKR

[6] The time expectations in the diagram are aspirational and may change.

Appendix A – Not Used

Appendix B – Not Used

Appendix C – IssuingSystemUniqueETicketNumber Calculator

C.1 The following information is needed to derive the IssuingSystemUniqueETicketNumber:

Property	Definition
Date of Issue	The date the barcode ticket is issued. Number incremented each day and reset every 5 years, with 1 = 01/01/16 = 01/01/21 = 01/01/26 For example, 14/08/17 becomes *592*.
Lennon Machine Number	4-digit number. Must match the 'Machine Number' used for the associated SDCI+ ticket issue record.
Lennon Transaction Number (Primary)	5-digit number. Must match the 'Transaction Number (Primary)' used for the associated SDCI+ ticket issue record

C.2 Concatenate Date of Issue (11 bits), Machine Number (14 bits) and Primary Transaction Number (17 bits), which results in 42 bits.

Then,

Convert 42 bits to nine characters using Base-30.

Then,

Transform result using the following character map

0 →	1 →	2 →	3 →	4 →	5 →	6 →	7 →	8 →	9 →
B	C	D	E	F	G	H	J	K	L
10 →	11 →	12 →	13 →	14 →	15 →	16 →	17 →	18 →	19 →
M	N	P	Q	R	S	T	V	W	X
20 →	21 →	22 →	23 →	24 →	25 →	26 →	27 →	28 →	29 →
Y	Z	2	3	4	5	6	7	8	9

C.3 The algorithm assumes that rail products will always have a validity period of less than 5 years, and that a specific Machine Number does not rollover the Primary Transaction Number on a single day.

C.4 For processing, and reverse processing, of the IssuingSystemUniqueETicketNumber algorithm, see associated document:

RSPS3001 XX-YY-A[7] IssuingSystemUniqueETicketNumber Calculator.xlsx

C.5 For example, a barcode ticket issued on 14th August 2022 by Machine Number 4722 as Transaction Number 13785 would produce an IssuingSystemUniqueETicketNumber of C8F23B8HG.

Date of Issue = 14/08/17, 11 bits (592) = 01001010000

[7] Where XX-YY refers to the related Subject Version and A refers to the Draft suffix character. Issued Versions do not have a Draft suffix character.

Machine Number = 4722, 14 bits (4722) = 01001001110010
Primary Transaction Number = 13785, 17 bits (13785) = 00011010111011001

Leading to the following 42 bits
010010100000100100111001000011010111011001

Converted to Base-30 results in 1S4MN0S65

Using the defined character mapping table leads to

1 → C
S → 8
4 → F
M → 2
N → 3
0 → B
S → 8
6 → H
5 → G

And therefore C8F23B8HG

Appendix D – Example code for Base26 encoding and decoding

D.1 Pseudo code

The following subsections provide a description of the algorithm required to implement the conversion from a hex representation of a raw byte-string to Base-26, and the reverse operation from a Base-26 string back to Hex. Note that the details of this conversion is unlikely to be available as a standard library function, since it does not follow the standard convention for byte order. There is no formal syntax for this pseudo-code, but conventions are used consistently, and generally commented on first use, e.g. left-arrow is used for assignment, ampersand for string concatenation and forward slash for integer division (with floor). Arrays are indexed from 1.

D.2 Base-26 Encoding

```
function hex_to_base26
(
   hex_string string
)
return string

/* This function is passed a variable hex_string, being a string of hex characters representing the
bytes to be converted. It returns a string containing the corresponding base-26 representation. */

   /* variable declarations */
   declare in_bytes[] as byte array /* to store the input as a string of bytes */
   declare out_string as string /* to hold the base-26 string to be returned */

   /* Variables required during the calculation */
   declare in_length as integer
   declare out_length as integer
   declare accumulator as integer
   declare full_value as integer
   declare full_value_m26 as integer
   declare b26_value as integer

begin
   /* convert the input string to a byte array an assign to a variable */
   in_bytes ← convert_hex_to_raw(hex_string)

   /* establish the length of the input, and estimate the length of the output */
   in_length ← length(in_bytes)
   out_length ← (in_length * 851) + 500 - 1
   out_length ← (out_length – (out_length mod 500)) / 500 /* integer division with floor */

   /* each iteration of the outer loop establishes one base-26 symbol */
   for x in 1 to out_length loop

      accumulator ← 0

      /* the inner loop processes each input byte from least to most significant */
      for i in in_length down to 1 loop

         full_value ← (accumulator * 256) + (integer) in_bytes[i]
         /* establish the base-26 quotient and modulus of full_value */
         full_value_m26 ← full_value mod 26
         b26_value ← (full_value - full_value_m26) / 26
         /* update the current byte */
         in_bytes[i] ← b26_value
         accumulator ← full_value_m26;

      end loop

      /* Concatenate the Base-26 symbol, being an uppercase character between A and Z */
      out_string ← out_string & map_to_base_26_symbol(accumulator)

   end loop
   return out_string
```

```
end function
```

D.3 Base-26 Decoding

```
function base26_to_hex
(
   base26_string
)
return byte[]
/* This is passed a string containing only uppercase characters (A to Z) representing a value
expressed in base 26. It converts to binary data and returns this as a byte string. */

   /* declare byte array to hold the integer values represented by the base-26 string */
   declare in_values as byte[]

   /* declare byte array to hold the output */
   declare out_bytes as byte[]

   /* Variables required during the calculation */
   declare accumulator as integer
   declare full_value as integer
   declare full_value_m26 as integer
   declare b26_value as integer

begin

   /* populate a byte array with the integer values represented by the input base-26 string */
   for i in 1 to length(base26_string) loop
      /* Decode the value of the base-26 symbol, A → 0, B → 1 etc. Cast to BYTE */
      in_values[i] ← base_26_symbol_to_integer(base26_string[i])
   end loop

   /* estimate the result array. Note - this might be oversized by 1 byte */
   out_size ← ((length(base26_string) * 500) + 851 - 1) / 851

   /* Loop through the initially empty output byte string from Least to Most significant, noting
   that it is little-endian, so left to right */
   for p = 1 to out_size loop

      accumulator ← 0

      /* Loop through the input base-26 symbols from Least to Most significant, noting that it is
      little-endian, so left to right */
      for i = length(in_values) down to 1 loop

         full_value ← accumulator * 26 + in_values[i]

         in_values[i] ← full_value / 256 /* Integer division with floor */

         accumulator ← full_value & 256 /* where & is the bitwise AND operator*/

      end loop

      out_bytes[p] ← accumulator

   end loop

   /* there may be an extra zero character at the end of this array. Check if the 129th byte is zero,
   and if so, truncate to 128 bytes. */

   if length(out_bytes) = 129 and out_bytes[129] = 0

      truncate out_bytes to 128

   end if

   return out_bytes

end function
```

Appendix E – 6Bit Character Table

E.1 Simply add 32 to a 6Bit character to get the 7Bit ASCII character:

6Bit	7Bit ASCII	char
0	32	space
1	33	!
2	34	"
3	35	#
4	36	$
5	37	%
6	38	&
7	39	'
8	40	(
9	41	)
10	42	*
11	43	+
12	44	,
13	45	-
14	46	.
15	47	/
16	48	0
17	49	1
18	50	2
19	51	3
20	52	4
21	53	5
22	54	6
23	55	7
24	56	8
25	57	9
26	58	:
27	59	;
28	60	<
29	61	=
30	62	>
31	63	?
32	64	@
33	65	A
34	66	B
35	67	C
36	68	D
37	69	E
38	70	F
39	71	G
40	72	H
41	73	I
42	74	J
43	75	K
44	76	L
45	77	M
46	78	N
47	79	O
48	80	P
49	81	Q
50	82	R
51	83	S
52	84	T
53	85	U
54	86	V
55	87	W
56	88	X
57	89	Y

Appendix F – Example Security Key and Ticket

The example ticket is encrypted using the following fixed 1024bit RSA key.

Never use these keys in production. A TIS Supplier must generate its own key pair. This key is for testing only.

F.1 Security Keys

Modulus =
1654036815525377345412565448379453843559486914301545776815067791670464434948100162116393533906833356026635527060528398017212524170789714748325265707476885395169247950736258763381375352981206739423990722608337580381525951053112600022686509519686593664094717627079864357538240609711489337721490127577701737223698

= HEX [00eb8aeb d600a893 2858268c 78c76235 8ae0867a 5b550e76 b9edc450 064f5b33 6656ca25 f5867813 8f4a4e8c 9d6e8079 7e78cc4b 5aef80e7 a23e0c05 3f8c95e9 15436ce9 1b7a8866 9a780d1f a417a795 46582e52 0f65fc58 2bf59ac5 824ba608 a231fdc7 1ceabbd9 70dbcf9c e55d3c20 55a0c46b dd7e7d5a 52e61f87 7e4408af 01]

Public Exponent =

65537

= HEX [010001]

Private Exponent =
23262061627321059848127951748956201956280865601289878122136319690902346303487555417896454219783150041194286666946748011847731564279977418611340119361298885082044657909018181086012602037113822561490684256783145576331417553050536990136491069182415201509485835072189773434923487177372245370361799421231555208773

= HEX [21205394 b0590501 3a8c895a ff2797c2 255ba45f adf1afce ec5a9caa 96848c11 0b89b896 f44774f0 c5119103 1f246071 e209515b c3ad4c66 6bf582d3 72312f2b 7250fe61 f6abed7f e219c08d c3985ae1 3f6b6db2 0e3c040b df7a817d 14a5a6f1 20d94047 08512132 aca00baa 29805440 4ad5dec2 1bd544bb 8938c74b 2904e645]

F.2 Example Ticket

Ticket Data:

No	Field Name	Encoded Value
1	RSP - Mandatory Manual Check	0
2	RSP - Multiple Supplements Applies	0
3	RSP - On Paper or Screen	1
4	Static / Dynamic barcode Indicator	1
5	RSP - NonRevenue/Unload Coupon	0
	RSP - RFU	0
	RSP - HasDiscountCardNumber	0
7	IssuingSystemUniqueEticketNumber	000000115
8	Checksum/RFU	3
9	VersionNumber	0
10	Class	1
11	LTOT	BAA
12	FTOT	SOR

No	Field Name	Encoded Value
13	OriginNLC	0418
14	DestinationNLC	5520
15	SellingNLC	5148
16	AdultChildFlag	0
17	CouponType	3
18	DiscountStatusCode	
19	RouteCode	00000
20	StartDate	4317
21	TimeValidFrom	867
22	DepartTimeFlag	1
23	PassengerID	0
24	ParentTicketReference	
25	CustomerGender	
26	SupplementCode	
27	ViaLondonFlag	1
28	OutOfStationInterchangeNLC	0
29	Bidirectional	0
30	CarnetCount	0
31	LimitedDuration	0
32	NoIPEflag	1
33	optionalData flag	1
34	PrintRetailerFreeUse	0
35	NumberOfJourneyLegsUsed	0
36	*PurchaseDate*	4894
37	*PurchaseTime*	490
38	*PurchasePrice*	11800
39	*DiscountedFlag*	0
40	*ValidityCode*	
41	*PurchaseReferenceCode*	REF12345
42	*NumberOfDaysValid*	30
43	*NumberOfAdditionalAdults*	0
44	*NumberOfAdditionalChildren*	0
	RetailerFreeUse	
202	IPESeal / SHA256	E0F28D6324DC69BB

Note: The above example is intended to demonstrate the encoding principles prescribed in this document, and the field content may not be compliant with the current specification.

Encoding the above example results in 108 bytes containing the ticket attributes, and a further 8 bytes which is the IPESeal / SHA256.

The 108 bytes of ticket data resulting from the value shown in the example above are as follows (n hex):
284104104104515530C50C39DF920A22C2AAA482A8A8C3000000086EB63400000000000000000000000
0001000000001813003D402E18000652CC8A49A8A8780000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000

These 108 bytes are then applied to the SHA256 algorithm to obtain the 32-byte value (in hex):
E0F28D6324DC69BBDD4943F089A883E6CC9D13D77D4287C7F540A66E19EA15A5

Appending the most significant 8 bytes of this hash to the 108 bytes of ticket data yields the 116 byte Binary Ticket which is the cleartext to be RSA-encrypted

Binary Ticket (Hex):

284104104104515530C50C39DF920A22C2AAA482A8A8C3000000086EB63400000000000000000000000
0001000000001813003D402E18000652CC8A49A8A8780000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000E0F28D6324DC69BB

This 116 byte binary string is now encrypted, and an example encrypted string follows. Note that the size of the encrypted string is 128 bytes, due to the size of the key. Furthermore, the RSA encryption process includes an inherent random element when encrypting, so a different value will be obtained each time the same value is encrypted. For this reason, you should not expect the same value as shown here.

RSA Encrypted Ticket (Hex):
4A2A7852EB17D80493BFB526187F5504564244581F600D649AF81B28627B9DF08F7378BB4B0C9C75A
FE0AF8342A86154CDC7AE41BD51EFE291FD9B816B92506A1622228B073D08B92F3CA83808380362A8
F6C4D60875AA268247504825A094CDE117FCD4F3188090D881EF82DFF3102F0D745E269DBF9895FFE
4239FFEB51AB4

Base 26 Encoded Encrypted Ticket:
KMUZPLCLWJWJENIQYWEMJMRSQIRNGIKCUIQNDFCNNUSIVIDSGVOOMZLVTVNNLJIHXPQLUMKG
SACYWRORIRQGULTWDHVEJBZUHVYVVDIJXIBZJZKHCAEAEYFEFRBXTCXXFHUXIQNHTBJGFWGU
CVMLDUSJXJZGRVJRMEGBUSYCAZSABIKFLVIHXNVOHVFPWDVKXBFJRXGVCPFGNEIYAPGFGWPP
HL

Final Barcode Payload (Including Header):
0600000011530MSKMUZPLCLWJWJENIQYWEMJMRSQIRNGIKCUIQNDFCNNUSIVIDSGVOOMZLVTVN
NLJIHXPQLUMKGSACYWRORIRQGULTWDHVEJBZUHVYVVDIJXIBZJZKHCAEAEYFEFRBXTCXXFHUX
IQNHTBJGFWGUCVMLDUSJXJZGRVJRMEGBUSYCAZSABIKFLVIHXNVOHVFPWDVKXBFJRXGVCPFG
NEIYAPGFGWPPHL

Appendix G – Key Sharing and Exchange Mechanism

G.1 Ticket Issuers

G.1.1 The XML file consists of a list of Ticket Issuers and their eTVD instance elements.

G.1.2 The following information will be provided in the feed

lastUpdated: this records the last time this config file was updated, to help eTVD servers to determine if they need to update. All dates and times in this file will be provided in the ISO 8601 format.

name: The name of the participant.

shortName: A short name of the participant, excluding spaces.

ticketValidationDatabaseID: ID of the nominated eTVD instance used by a Ticket Issuer.

ticketValidationDatabaseURL: A URL for the Ticket Issuer's Ticket Validation Database. This is provided by each Ticket Issuer's to nominate which service they wish to use as their eTVD. Any individual ticket queries, or Ticket Event records for tickets bearing that ticket's IssuingSystemID should be sent to this nominated URL. The service at this URL must comply with the eTVD messaging specification, RSPS5043: 'eTVD Messaging Specification'. These ticket database services should only accept action requests from other servers that are listed in this (or the Ticket Validator) XML page, having validated the other server's identity via mutual HTTPS authentication to prevent abuse of the system by 3rd parties. The SSL certificate used by the eTVD server at the URL specified must carry a valid 3rd party certified SSL certificate from a major certification authority (verisign, thawte, geotrust) for mutual SSL authentication to be trusted.

shortScanningRecordSubscriptions: Not populated for a Ticket Issuer entry. Previous purpose of field was to allow an eTVD instance to subscribe to receive Ticket Event records that occurred at other participants' validation equipment via that participants' eTVD. This approach is now considered too simplistic for wider availability of barcode ticketing and associated flows.

admin: A phone number and email address for the technical administrators for this participant. These should be notified if there are any serious issues with the ticketing system, such as the unexpected replacement of a participant's keys due to key theft. These admin contact details are not for handing out to the public.

customerService: A phone number and email address for customer issues with this participant, these can be handed out to members of the public.

publicKey: Zero or more elements containing public keys and associated metadata.

issuerID: The IssuingSystemID for a ticket decryption public key, which has a period of validity between start and end dates and times.

start: A date in ISO 8601 format indicating the start of validity of the public key.

end: A date in ISO 8601 format indicating the end of validity of the public key.

keyX509PEM: A public key formatted as a PEM encoded X509 certificate. If participants have their keys in a different format, or wish to make use of the keys in a different format, there is an excellent and free key format translation tool available here to convert their X509 certificates to PEM: http://www.openssl.org.

exampleTicket: One or more elements containing encoded payloads that may be decrypted using the accompanying public key. These are included so that RSP and other participants can ensure that they have correctly loaded the key, and that they can correctly decode and verify the example tickets.

emergencyKeyCancellation: Flag present where a key has been confirmed to be compromised.

G.1.3 It is not essential for the public keys to be signed by a certificate authority under this scheme if RDG are accepted to be a trusted source, but if participants use signed and certified keys protected by seals from Verisign, Thawte, or other recognised bodies, this ensures that the key material that they submit to RDG has arrived correctly, and it allows RDG to validate the key's identity prior to uploading to the XML page.

G.1.4 The example tickets are a convenient way of any other participants having examples from each Ticket Issuer's to test against their own scanning.

G.1.5 The XML format can be used by systems implementers to test their systems prior to going live to ensure that they can pick up the Public Key details and other exchange requirements from other contributors to the scheme. Once live, the eTVD should also access the machine-readable XML page once a day to pick up any new contributors to the scheme or changes in subscription details.

G.1.6 XML format for Key Sharing

```
<?xml version="1.0" encoding="UTF-8"?>
<rsp6InteroperationConfiguration lastUpdated="2012-07-01T00:00:00+01:00">
 <ticketIssuer>
     <name>Steam Revival Railways</name>
     <shortName>SteamRailways</shortName>
<ticketValidationDatabaseID>3P<\ticketValidationDatabaseID>
<ticketValidationDatabaseURL>https://eticket.3rdParty.co.uk:2051/ticketDB/</ticketValidationDatabase
URL>
     <shortScanningRecordSubscriptions>
     <stationNLCSubscriptions>1129,4426,4433,5148,5693,5696,7910</stationNLCSubscriptions>
     </shortScanningRecordSubscriptions>

     <admin>
         <email>rsp6Admin@steamrailways.co.uk</email>
         <phone>+4420712981298</phone>
     </admin>
     <customerService>
         <email>eTicketSupport@steamrailways.co.uk</email>
         <phone>+44845400400</phone>
     </customerService>

     <publicKey>
```

```
         <issuerID>A1</issuerID>
         <start>2011-01-01T00:00:00+00:00</start>
         <end>2013-12-31T23:59:59+00:00</end>
         <keyX509PEM>-----BEGIN CERTIFICATE-----
MIICWDCCAcGgAwIBAgIJAN0p/v4UTcIgMA0GCSqGSIb3DQEBBQUAMEUxCzAJBgNV
BAYTAkFVMRMwEQYDVQQIDApTb21lLVN0YXRlMSEwHwYDVQQKDBhJbnRlcm5ldCBX
aWRnaXRzIFB0eSBMdGQwHhcNMTIwMzA4MTkzMjM3WhcNMTQxMjAzMTkzMjM3WjBF
MQswCQYDVQQGEwJBVTETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW50
ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQDejKJQh+wf9hA9o723qPlgr5Or/RsfXr6+iOd4ha1b/I1HWbee/gFztQ/ZasKw
USSuXMLbuhvIBPqA2e6xzFR/OeVSTXBMrKz/4jXod0Ti8KdmC9uGlLPYTKsY1xol
k7v1vDn3/2dUdHeAO4uOvdOQrrY/dCoIGvlHwOhaadvj6wIDAQABo1AwTjAdBgNV
HQ4EFgQUB5WI9c3wSv/gQ6JER9ELwmzjoeowHwYDVR0jBBgwFoAUB5WI9c3wSv/g
Q6JER9ELwmzjoeowDAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQUFAAOBgQAUrOvc
ByhuWD6yq5GXQnYW8GWw20aWeEHu967e2vrVRUKiACNjKpLMdmCZdIKUcTUMn2kE
SIfJqgLfyYb4lo8RVHgdj9H4sxEqZ1F8geaZ2j3ND0ROMnCOeTZQ6NqIUGqOu7lP
HbG/Fz3YPk6n3Y27jhdWfUMvkcUo0SVwbta+Gw==
-----END CERTIFICATE-----</keyX509PEM>

<exampleTicket>0600000010120A1UTEWZQWFPSTSQBWQGWMOQQINZXUDEOGXTRDLZXWUGXRYQSZEDQAGQZKFGDUTBXFMTRUIOX
NXMVEFZGCYQJCADQMMVXNJWITIHYHMLRQYLXUSKBTCVGAIUHAAECYZEPPDFOJAWZVNLWOJQVFRRCEJIGIGTWYTEGGZTLPYMRCHHK
OSGJBBQVWJNGGYSKYHYAVSHGPWVDGARFIGUFEYRAVXJBVDYK</exampleTicket>

<exampleTicket>0600000010200A1DDFRECLPOLICZFUYJUWSRSVFRXBWIQHRJORRBONTGWLAVRNYKUVAVJMDYSQCJYVLMEHTGX
LXOQFKYDDWBNUXIHXYJKCUKIECWXTXOGGUCVRYZCYAFZIDEWFBSBMIMRSISDOQBLAUDPJRQBURZJEEWQQWZDVEVLWATPMKFDEXMO
QSUPCKKZCWRTJCDLVJCCWJVFTDNXBMODOCZUFEUUCYBPETDA</exampleTicket>
     </publicKey>

     <publicKey>
         <issuerID>A2</issuerID>
         <start>2012-07-01T00:00:00+01:00</start>
         <end>2015-06-30T23:59:59+01:00</end>
         <keyX509PEM>-----BEGIN CERTIFICATE-----
MIICWDCCAcGgAwIBAgIJAJpaixQVCFLWMA0GCSqGSIb3DQEBBQUAMEUxCzAJBgNV
BAYTAkFVMRMwEQYDVQQIDApTb21lLVN0YXRlMSEwHwYDVQQKDBhJbnRlcm5ldCBX
aWRnaXRzIFB0eSBMdGQwHhcNMTIwMzA5MDc0NjE4WhcNMTQxMjA0MDc0NjE4WjBF
MQswCQYDVQQGEwJBVTETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW50
ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQDocdcLPWj1Xm5a7Q+e9RAIFqJKGHay5tcLGZohgeXlGgOPi/aSOywVip7EKpg2
mkWdZ4jkeWNJ3GxcbhwnpgL/vn0h0OxgbM0DjtI9CZnFrzP8yxObcPvycFrjXTA+
0spMQoKAqx7Ilp4iXmUl/+gwNAQCITnejfeE8vPA/22jXwIDAQABo1AwTjAdBgNV
HQ4EFgQUejIoVS90lJar2Od4QHRHk/4nfqswHwYDVR0jBBgwFoAUejIoVS90lJar
2Od4QHRHk/4nfqswDAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQUFAAOBgQBvfAOY
S3kkbd7Z7+ORv5kmwWpqOaNI1AjgH2i68Bv6IcGu9CHhDICC8O03FEMogOy/+DLL
Fi5k8TncNwHpUR7VM1KYcfBY6UZ8Du/fyScNbM1ebydK0En5kTjYYRnGjHkEzcMd
GYlDHphEEgU+zvHc00yalJ6ONPnSwuywo1XaeA==
-----END CERTIFICATE-----</keyX509PEM>

<exampleTicket>0600000010370A2IBHLGDDACNUDMTVZVNHEVCIIIOFITPXOFDXCRAYTEKHTZROGIYTFUSTPONWECCSDKLWYUA
GRSQFXDHUQEEUIBIYSLNZYULXYXQIDRLGCWNYTSTOLLANGTMBMDOJRONDVIDMXXRDTANSCOBCCAMFCRRZTZRMZJYZBOTJZVXSTNF
IUUATOUVKMMOFWXFPBJMHCRMJDXWRMTSYDFTYIYGIBLUUSTL</exampleTicket>
```

```
<exampleTicket>0600000010440A2VDFYMAVDBWTOPBHMXQOHXSNHZYFIRMYVBKIYRVGJYQIAGCCBUVYLVMIMGXMCTGTADDSTOG
VFJXQUSIQEBVPYEPAOJLQRPEXUJZKNPQLEGPPIEFHADVXRMYAHHIGGEWMEIWUXWBBIXTJKHWPURQOYHZQYEUBYOBYPSCCPBBGWOC
QCDRJLNYCWETKJVCJVRGOKBZLBRMAAGULEQPJQPIQFARLJEO</exampleTicket>
     </publicKey>

     <publicKey> <!-- example of retired key, now out of date -->
         <issuerID>A1</issuerID>
         <start>2008-05-06T00:00:00+00:00</start>
         <end>2010-12-31T23:59:59+00:00</end>
         <keyX509PEM>-----BEGIN CERTIFICATE-----
MIICWDCCAcGgAwIBAgIJAK+uCQDKw5ZFMA0GCSqGSIb3DQEBBQUAMEUxCzAJBgNV
BAYTAkFVMRMwEQYDVQQIDApTb21lLVN0YXRlMSEwHwYDVQQKDBhJbnRlcm5ldCBX
aWRnaXRzIFB0eSBMdGQwHhcNMTIwMzA5MDc0OTMzWhcNMTQxMjA0MDc0OTMzWjBF
MQswCQYDVQQGEwJBVTETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW50
ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQCyixumMDYmfh9Uv8Rbb8wmqMwpLLvqQ0JQXV+HyxgbJBsYVH6zahhGEha+2BZu
B1I84ZArieVFvBtL6HuDjgoHJIJhwROklmlzm5gkRQt/hDcEWH42MjMzGvEO/oyh
SzW+YAv5VAO8FkTAR/q5YtUSaD7qrI5lJ3N1wuI2yeBviwIDAQABo1AwTjAdBgNV
HQ4EFgQU5llijlwQlBnsq3jAiClP4dNHeoAwHwYDVR0jBBgwFoAU5llijlwQlBns
q3jAiClP4dNHeoAwDAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQUFAAOBgQCP347O
f3xgy1/ICpNBkd7xDcdxBkTsHoP9EXV9CXho6P8C/We1blulFEx+JKrh+I7fMIUe
YZ3oRQ+FzL44dYgOr6PtwuYMoG70D2aRQWf59iOu/ldvsdIy2ezWC+vqks5ZUu95
xjwcXpbpNyqr0WJGGSoprnOxbcVxOdhgzpXRQg==
-----END CERTIFICATE-----</keyX509PEM>

<exampleTicket>0600000009060A1ACIJYDYDVNKYEXUGJZONAJXZKCACMFUZIKGUXFLUXZAETIUSTUNOOYKJDAWQKLTWBJXAHF
CHGFPPVQAIQOJTEZQPWRFPNHKQRHRRSFKLORQWUHLNBUZWVTRVZPYPWNKYFORLTVVZQGJYMJBBCVKZVBRSZIWCNVCSQNHYDAXHRD
WCCMOVQDPQWFYITSRWBULMNWMUWEPPBULWRTRXZSSJRHLZVM</exampleTicket>

<exampleTicket>0600000009130A1CRPIWOJXBEQEWDKTSHIAKFLYLBGETIAUVJTBAVRGGAGSNXXBORCYFPZNCEFUUKDTGDXSPV
YDBWLYTSVGEZUHKYMVIUHQPGNWMBQUUWFGYMCPBURAGTQLPWUMTRPVFDKBIZCNJMHUKHYTYKVZGTISORTEZEPAVYMCVRYYBGEODQ
HLIOKRMAYKJNWNEVQGTNDAAUBNNFNMDZPHTJMAMNLGGGQALA</exampleTicket>
     </publicKey>

     <publicKey> <!-- example of retired key, was replaced in an emergency -->
         <issuerID>A1</issuerID>
         <start>2008-01-01T00:00:00+00:00</start>
         <end>2008-05-05T23:59:59+00:00</end>
         <keyX509PEM>-----BEGIN CERTIFICATE-----
MIICWDCCAcGgAwIBAgIJAJZTRfXphYJ4MA0GCSqGSIb3DQEBBQUAMEUxCzAJBgNV
BAYTAkFVMRMwEQYDVQQIDApTb21lLVN0YXRlMSEwHwYDVQQKDBhJbnRlcm5ldCBX
aWRnaXRzIFB0eSBMdGQwHhcNMTIwMzA5MDc0OTQ1WhcNMTQxMjA0MDc0OTQ1WjBF
MQswCQYDVQQGEwJBVTETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW50
ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQDushAx5bZa/EHUucjUZu8KWq8TwdJ5ATMF3U6jPKBXymvtJHxIgkYI9OEemWNg
ccbSVbgVSJj4QOO8c373k4sKPUxA44xZ9PhHG3V/+OulPp7wECp8NRcpEafuLX+S
u8hc7EXBMMQ61kT+c27T/L+fRkIkHXynhRnJDgTyIXKzKQIDAQABo1AwTjAdBgNV
HQ4EFgQUQ6fztuRm/W7Bo31YUtqqej5l9L8wHwYDVR0jBBgwFoAUQ6fztuRm/W7B
o31YUtqqej5l9L8wDAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQUFAAOBgQDitLSY
SCxsoijwJMvjtsJRklrULZTw8Y9aZe9aLu3vTcMKUNh63JdU3aSs/GiBVkB047uS
```

```
f26QgWOISvnllZtnoSVeBAmdgvtB1iWw/rENC78l86ihLxBVJry+tOb/kn4v21kF
M1nHVSoBHcK2Kfp37SyQwEcqjVjbQ6p1Cq2p/g==
-----END CERTIFICATE-----</keyX509PEM>

<exampleTicket>0600000008040A1QXLXCOIZWWXYHJMZMEBCTNGYTGTMYKONXJBDBUFKNEXVEXJFDVZAUPACGEIVPGOXECRTPT
XAWBQINFJIBAYNNWSVDLVKZYPYSPFERWFRZJOHZVAAQPFKVSUAZVGYWUBGFDGUYLHTIATAXTMHIRTTPAPVUJNUECPEGZNMXKWSXQ
PXGFCNQXJCYTLCDKYDEFSEKURKOTSWCZNVCEUZMODSUJLNWP</exampleTicket>

<exampleTicket>0600000008110A1NTMMVERGTSHBZNFGYTZCMSNGWWMWLNHMUCEXATBAMJCQBGLWADHPUUNIBIARCFHJRXJUTL
NLAXBOOFZMSCFRSSDSEDOCIFXBNCBZNPLVRSNEABSTMEDFMWAKGMHPBUDAPHDNTXJQWLCQVONRFPQOVHBGXIGTYUXQSORIBQWBDD
RQHBOIIUSKPHOAOEWRAUKIMAFXTDCKAXQGCULXPOQOGGHQGN</exampleTicket>
         <emergencyKeyCancellation>true</emergencyKeyCancellation>
     </publicKey>

 </ticketIssuer>

 <ticketIssuer>
     <name>Electric Express Trains Limited</name>
     <shortName>ElectricExpress</shortName>
<ticketValidationDatabaseID>IN<\ticketValidationDatabaseID>

<ticketValidationDatabaseURL>https://etvd.integrator.com/electricExpress/</ticketValidationDatabaseU
RL>

     <shortScanningRecordSubscriptions>
 <subscribeAll>true</subscribeAll>
     </shortScanningRecordSubscriptions>

     <admin>
         <email>rsp6Admin@electric.co.uk</email>
         <phone>+4412039871928</phone>
     </admin>
     <customerService>
         <email>mobileHelp@electric.co.uk</email>
         <phone>+448001251234</phone>
     </customerService>

     <publicKey>
         <issuerID>A3</issuerID>
         <start>2011-01-01T00:00:00+00:00</start>
         <end>2013-12-31T23:59:59+00:00</end>
         <keyX509PEM>-----BEGIN CERTIFICATE-----
MIICWDCCAcGgAwIBAgIJANbnPFeTwCc2MA0GCSqGSIb3DQEBBQUAMEUxCzAJBgNV
BAYTAkFVMRMwEQYDVQQIDApTb21lLVN0YXRlMSEwHwYDVQQKDBhJbnRlcm5ldCBX
aWRnaXRzIFB0eSBMdGQwHhcNMTIwMzA5MDc1MDAzWhcNMTQxMjA0MDc1MDAzWjBF
MQswCQYDVQQGEwJBVTETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW50
ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQDXG6GqHnDwl6pkb78hlVXBXsnY+FDPcO0h39Sr23LETkw5KGtIItdOWs0g0JAK
F3EL8/FcxkRvUhe+XZ5eBU02PcHRu0oWuVC9IOLGYamuxrbi0pO65LWY6I24ygsO
4USAT9L3nzgUz2ERux11AlhPSsA8A4lipGysFxzZbOtR3QIDAQABo1AwTjAdBgNV
HQ4EFgQUk9RC8zEbYHDeJqMezmgRfBt8qKowHwYDVR0jBBgwFoAUk9RC8zEbYHDe
```

```
JqMezmgRfBt8qKowDAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQUFAAOBgQB5oV6T

tTyM/jjpOKpASh/vNOvyNXjgKIlOU1a/SvIeohBhfa0KGX0HFCW3WRHMozBFDP4u

4QQl/A6/a1mtDpRvzX6pHwkyQ2m898DxKoxOMg7wj09lT/aQ0wCq0vgIPuikvY2P

zyHCJDSCKAlx/coZbuCFgOeEnDFcjBVk1tCsVg==

-----END CERTIFICATE-----</keyX509PEM>

<exampleTicket>0600000008040A3EFKDOJTDHPQSTIUXTCSECLONRTKAIANLSTFIDFJZLVPSAGAYLFBWATPJGHFMAOAONZXAKK
WEGMMGQCJZIRHICEDGLUIPNPVGWGJJIRIXSLBNQOLDCDTOCUOMYJMYXBUNWORDMPVGYQIUROSZCLPUWWPMRNJWQMMUYMQROEQTDW
HZVMFOAKITYATCIIJUOICKURZDXNTYRPLMADTKYRPFQDVAUN</exampleTicket>

<exampleTicket>0600000008110A3VHXDJGEEPEJZYMKDNYLZRGRVMSDBOODXTXYHNZDTBTBHBLPUQDXBIGWZGHDNAJHONRKUPX
PPBARGEGOBZUUSWFZYIXOAVGRCWZDTOLWIQEYMOCMDCTMWZSATHWWHGTNSLFTOGVMKDOFOIETTRACTYTVCPATRRRFAWDONJXMHEP
KVOVHAJXAVAGFQWTVOXPFELLYZYYAJYSLNUAJYHZEYSSXGNO</exampleTicket>
     </publicKey>


     <publicKey>
         <issuerID>A4</issuerID>
         <start>2012-07-01T00:00:00+01:00</start>
         <end>2015-06-30T23:59:59+01:00</end>
         <keyX509PEM>-----BEGIN CERTIFICATE-----
MIICWDCCAcGgAwIBAgIJAOHNVpZpgmCnMA0GCSqGSIb3DQEBBQUAMEUxCzAJBgNV

BAYTAkFVMRMwEQYDVQQIDApTb21lLVN0YXRlMSEwHwYDVQQKDBhJbnRlcm5ldCBX

aWRnaXRzIFB0eSBMdGQwHhcNMTIwMzA5MDc1MDIwWhcNMTQxMjA0MDc1MDIwWjBF

MQswCQYDVQQGEwJBVTETMBEGA1UECAwKU29tZS1TdGF0ZTEhMB8GA1UECgwYSW50

ZXJuZXQgV2lkZ2l0cyBQdHkgTHRkMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB

gQDIAYBbkrbbYOG4VST19ZYYZaCgAe0wGAWghYIOOJ1Dk23lduAabfe7XQSFjC0I

1SPd5xdnOfVx8yj2cYr4holkwsSGA27nkM3mPcsvZO4V9Slp3txBb+EoR6eQ+orZ

Hk93/5c7u6An6Wa7fEoEeUeuiNwc7F0mJa9BmNHt2j1UUQIDAQABo1AwTjAdBgNV

HQ4EFgQU/wDi75o8QMhbGxovmXFX3JjP4/wwHwYDVR0jBBgwFoAU/wDi75o8QMhb

GxovmXFX3JjP4/wwDAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQUFAAOBgQBTI4Hz

rz9z5BQwgfRp46crQ2wwDxgAJdyld8Xi/pLpZP5NO9uH/OpEwUQId5XqOS+8+U+N

iT95BqJkHRnmgP87HOJekCiXzt+1HIRveDfEi4cajRvF+rZsS/FYPnWfUTdO4a5S

CrtoKFyVTm8TokBi/RANNXSnx2b83oSHhVUqTg==

-----END CERTIFICATE-----</keyX509PEM>

<exampleTicket>0600000080030A4PROGQQJRLFSKEWMKPWRLJGVUIARYVCQLXOIAQBBCJFJPNYGIOXYKDUFULDGWBBKCOBLAUY
ZUTVQVXBQWWNEKJZTFGLMYCTDGCNMXETXJEZVZKEFCOSHHTKNYZKMYKITBOEYTWHSUIZPEWXTNVDOMPQYYRBJRRJIFFTRYHAIFXQ
WGEVQTBCLSHHLFTTXODCJLSKYLDGJBIHHDOLRLHSCEGYVUPM</exampleTicket>

<exampleTicket>0600000080100A4MOFLQHCPHWGKOWNZXKUQVDZHCCARHRVVFIVOYNETEFUHJQAGSZRWYHELPQJLMYXPJJOZYM
FVDPIXUGMBEMHPPVJVSUSMYRQGDAYWLLKTTVFVKTCVVSUYLJWJRGYSWDMFBDFYONENBOYRRJQMUJNAWYABPYJBQXITEDFXYEIFLU
XWRLHAIXLACEBMLULFBSZFYVIRKKSODATBNLPCEDFGWFDHOB</exampleTicket>
     </publicKey>
 </ticketIssuer>


</rsp6InteroperationConfiguration>
```

Appendix H – Revision History

Version, Date and Authors	Comments
01-00 01-Apr-2008 M Gulam	Issued after approval at RSSG.
01-00-A 22-Sep-2008 M Gulam	
01-00-B 23-Oct-2008 M Gulam	Issued after initial review by suppliers.
01-00-C 21-Nov-2008 M Gulam	Issued after second review by suppliers.
01-00-D 25-Nov-2008 M Gulam	Issued after late but pertinent second review comments.
01-01 03-Dec-2008 J Law	Approved for release by RSSG 02-Dec-2008.
01-02 17-Dec-2009 S Standley	Cosmetic correction to section 2.5.2.
01-03 11-Aug-2010 S Standley	Cosmetic amendments for clarification. See bulletin 109.
01-03-A 30-Jul-2012 D Monney	Updated to include the additional requirements for flexible, inter-operable walk-up tickets with a long validity, such as Carnets and Season tickets on mobile phone. Changes to allow interoperable walk-up tickets on mobile provided by Masabi. Updated the Appendix.
01-03-B 10-Sep-2012 D Monney	Format changes. Type 06 specification removed to external excel document. New section 6- Key management added.
01-03-C 27-Sep-2012 D Monney	Corrected typos, formats and previous version comment. Updated tables.

Version, Date and Authors	Comments
01-03-D 05-Apr-2013 D Monney	1) More detailed description of IssuingSystem ID. 2) Emphasis on Appendix B title explaining attached Type 06 excel tables only includes the encrypted payload. 3) Sentences updated in section 1.2.6; 2.1.1; 2.1.2; 2.7.4; 4.2.2; 4.2.4; 6.1.3. 4) Removed multiple passengers & Carnet ticket types as requested by Barcode working group. They will be incorporate in a separate best practice guide document. 5) Clause 1.3.1 removed. 6) Updated this and associated documents to address JIRA STD-229, STD-297, STD-315, STD-409, STD-412, STD-459, STD-477, STD-478. 7) Multi-passenger and Carnet tickets removed from the Example ticket excel document. The further implementation of these ticket types will be on a best practice guide and would not be accredited. 8) Activation and de-activation is now at implementer's discretion and will not be covered by accreditation. The descriptions have been removed. 9) Two new fields have been added to the Type 06 barcode data structure doc. These are 'Discount Card Number' and 'RSP – HasDiscountCardNumber'.
01-03-E 04-Sep-2014 Tim Pickman	1) External review comments addressed. 2) Type 04 and 11 barcodes removed as they are no longer accredited by RSP and barcode data is managed by the issuing TOC. 3) JIRAs STD-508, STD-532 and STD-539 addressed. 4) Changes to the data fields stored in the barcode – see Appendix E for the proposed changes.
01-03-F 29-Oct-2014 Tim Pickman	Example barcode header details added. Additional information explaining the VersionNumber added. Sent to RSF for approval.
01-04 17-Nov-2014 Tim Pickman	Issued following approval from RSF
01-04-A 01-Aug-2016 James Wright	1) Optional Data now mandatory. 2) PurchaseReferenceCode field populated with full Photocard Number where required. 3) Unique eTicket Number now derived from algorithm held in RSPS3001. 4) RFU field redefined to identify compliance with version of RSPS3001. The following JIRAs have been addressed: STD-767, STD-768, STD-769, STD-770, STD-828, STD-830 and STD-842.
01-04-B 16-Aug-2016 James Wright	Comments from Internal Review addressed.
01-04-C 14-Sep-2016 James Wright	Comments from External Review addressed.
01-04-D 04-Oct-2016 James Wright	Comments from Final External Review addressed. Sent to RSF for approval to issue.
01-05 19-Oct-2016 James Wright	Issued following approval from RSF.
01-05-A 30-Jan-2018 James Wright	Inclusion of Type 08 barcode encoding. Inclusion of IssuingSystemUniqueETicketNumber calculator. Inclusion of Key Management processes, formalising existing informal practices, but including time expectations. Revised composition of RDG PKR. Reordered encoding information. Inclusion of expected encoding for various types of ticket.

Version, Date and Authors	Comments
01-05-B 05-Mar-2018 Lami Jinadu	Addressed internal review comments.
01-05-C 08-May-2018 Lami Jinadu	Addressed First External Review comments.
01-05-D 10-Dec-2018 Lami Jinadu	Reverted to First External Review stage to allow a fresh review following a long pause in progressing to next stage. Clarified in 'Appendix A Type 06 Encoding Information': • that populating the reservation details is mandatory for any ticket type where reservations are mandatory. • that the fields OriginNLC and DestinationNLC refer to the fare origin and destination, rather than the journey origin and destination. • the flows where the CROSS_LONDON_IND is 2, 3 or 4. • that the PassengerID field should always be populated with 00000 because barcodes do not require passenger ID. Corrected error in 'Appendix A Type 06 Encoding Information' which stated that RetailServiceIDLeg1Numbers should be encoded with the final 4 letters of RetailServiceID instead of the first 4 letters. Clarified in Section 2.6 that test and production UTN must have different IssuingSystemID.
01-05-E 12-Feb-2019 Lami Jinadu	Addressed comments from First External Review.
01-05-F 17-Jun-2019 Neil Barkham	Addressed comments from Final External Review. Clarification of Symbology, data capacity, presentation and size. SHA-256 correction and clarification. Various Base26 clarifications. Appendix A: Confirmation of 'RSP - RSPS3001 version number', field 6, moving to a value of '2' for Issued version 01-06. Appendix D: Base26 Java code replaced with Pseudo code. Appendix F: Corrections and clarifications.
01-05-G 05-Jul-2019 Neil Barkham	More background information provided for section 3.4 – Encrypted Payload.
02-00 22-Jul-2019 Neil Barkham	Issued.
02-01 04-Aug-2020 Neil Barkham Matthew Pickman	Replace Photocard Number with Photocard ID. Version history now recorded as an appendix. Updates to Fields 23, 24, and 25 of 'Appendix A Type 06 Encoding Information.xlsx' for GDPR reasons. Incorporation of Corrigendum from Version 02-00 (STD-1225): • No information has been provided to match with Version 02-00 of this Subject in accompanying spreadsheets • Base-26 Encoding 'in_bytes[x] = b26_value' changed to 'in_bytes[i] = b26_value' Base-26 Decoding 'accumulator <- full_value' changed to 'accumulator <- full_value & 256'

Version, Date and Authors	Comments
02-01-A 24-Nov-2021 Tim Handel Neil Barkham	STD-1482: Implement Excess functionality. STD-1342: Barcode Key accreditation requirements must be passed before distributing keys. STD-1434: Change of contact address for Key Management. STD-1200: Updates to Appendix A Fields *‘RSP - Mandatory Manual Check’*, *PassengerID*, *CustomerName* and *CustomerGender.*
02-01-B 20-Jan-2022 Neil Barkham Aaron White	Updates following External Review. References to spreadsheets updated.
02-01-C 04-Feb-2022 Matthew Pickman	Additional clarifications to appendix spreadsheet references in this document STD-1283 StartDate field of type 06 barcodes needs clarifying to accommodate Advance purchase Returns (in Spreadsheet Appendix A) STD-1408 Additional Validation Step
02-02 21-Feb-2022 Matthew Pickman	Subject Issued

End.