Tech Note #011

Tags for Obtaining Encrypted Track Data

Rev. A

Revised 11/30/2016

International Technologies & Systems Corporation
10721 Walker Street, Cypress, CA 90630-4720; Tel: (714) 761-6368; Fax (714) 761-8880
www.idtechproducts.com

Executive Summary

ID TECH proprietary tags DFEF4B, DFEF4C, and DFEF4D provide a way for track data (and optionally, PAN data) to be supplied in conjunction with an EMV transaction, with or without sentinels, in a form similar to the form track data would take in a conventional MSR transaction. The data in these tags is considered sensitive and is thus encrypted.

This document describes the semantics of the aforementioned tags and how the tags can be used by payment-app integrators who need to supply encrypted credit card data to gateways, issuers, or other processing entities that may have a need to consume such data.

The information presented here augments, but does not supersede or replace, the more extensive information provided on this subject in document 80000502-001, *ID TECH Encrypted Data Output Formats*. Please continue to refer to that document for the most authoritative, up-to-the-minute information on encrypted data handling.

Motivation

Your payment processor may require you to provide encrypted track data that, when decrypted, looks something like this:

```
3b34373631373339303031303130313030313d313531323230313131343338373830383938393f
```

Which, after rendering as ASCII, would look like:

```
;4761739001010010=15122011143878089?
```

Using terminal configuration tag DFEF4B, you can cause such data to appear in tag DFEF4D. What's more, you can control:

- Which tracks (1, 2, or 3) are allowed to appear in DFEF4D
- Whether sentinels appear (or not) in tracks 1, 2, or 3, in tag DFEF4D
- Whether PAN data appears, as a separate element, in DFEF4D
- Whether to include *all* eligible track data, or just the first element found

If you do not specify tag DFEF4B in your terminal configuration settings, tag DFEF4D will not appear in the transaction output; instead, you will see the track-data tags appropriate to your EMV transaction (e.g., tag 56 for contactless track 1, tag 57 for contact track 2, etc.), and those tags will be encrypted in the manner previously described in ID TECH document 80000502-001, which is to say, the encrypted data will consist of the *entire* TLV (including tag and length) as a data block within a tag of the same name. (See the "Encrypted EMV Data" section of document 80000502-001 for details.) Your payment processor may not want the *entire* TLV as an encrypted *payload*. Many processors expect the 'V' of an encrypted TLV to contain *just* track data, which is the case handled by tags DFEF4B, DFEF4C, and DFEF4D.

Tag DFEF4B

Tag DFEF4B is a configuration tag. Use it to tell your ID TECH reader which tracks you want to receive in tag DFEF4D, whether or not to use sentinels, and whether or not to include the PAN as a separate string.

If you are using ID TECH's Universal SDK, you will provide this tag as one of the terminal configuration settings in `SharedController.emv_setTerminalData()`.

Tag DFEF4B has a length of 03. Only the first data byte is currently used (the other two bytes are reserved for future use). The byte semantics are as follows:

Byte 1:

8	7	6	5	4	3	2	1	NOTES
-	-	-	-	-	-	-	X	0 – Disable Track 3 Sentinels 1 – Enable Track 3 Sentinels
-	-	-	-	-	-	X	-	0 – Disable Track 2 Sentinels 1 – Enable Track 2 Sentinels
-	-	-	-	-	X	-	-	0 – Disable Track 1 Sentinels 1 – Enable Track 1 Sentinels
-	-	-	-	X	-	-	-	0 – Disable Track 3 1 – Enable Track 3
-	-	-	X	-	-	-	-	0 – Disable Track 2 1 – Enable Track 2
-	-	X	-	-	-	-	-	0 – Disable Track 1 1 – Enable Track 1
-	X	-	-	-	-	-	-	0 – Disable PAN 1 – Enable PAN
X	-	-	-	-	-	-	-	0 – All Data Elements Found 1 – Only First Element Found

Byte 2: RFU

Byte 3: RFU

You can use the top bit of the first byte of DFEF4B to control search behavior: If the bit is OFF, all data elements requested will be provided (if they exist). If the bit is ON, only the first element found will be retrieved and placed in DFEF4D.

If you request multiple data items, they will be concatenated. To know the original lengths of the items, you must retrieve and inspect Tag DFEF4C (see discussion below).

To use tag DFEF4B, add it (as a TLV) to your terminal configuration settings. Send the settings to your device as you normally would. For example, in ID TECH's Augusta, you would use firmware command 72 46 02 03 to Set Terminal Settings. In the Universal SDK, you would provide the TLV in `SharedController.emv_setTerminalData()`.

> NOTE: If tag DFEF4B does not exist in Terminal Settings, tags DFEF4C and DFEF4D *will not be generated*. Instead, encrypted track data will be provided in tags 56, 57, 9F6B, or one of the other tags documented in the Encrypted EMV Data section of document 80000502-001, *ID TECH Encrypted Data Output*.

A reasonable default value of byte 1 of this tag is 0x12 (Track 2 enabled, with Sentinels).

Data Search Order

When **"Only First Element Found" (bit 8 = 1)** is set in DFEF4B, Tag DFEF4D will be populated with a *single* data element according to the following search order

Track 2, Tag 57 (converted to alpha numeric format)
Track 2, Tag 9F6B
Track 2, Tag 5F22
Track 1, Tag 56
Track 1, Tag 5F21
PAN, Tag 5A (converted to alpha numeric format)
Track 3, Tag 58
Track 3, Tag 5F23

Regardless of the original format, the data will be placed in the DFEF4D tag in alpha numeric format, such that after decryption (and with padding removed) the data will look similar to:

3b34373631373339303031303130313031303d313531323230313131343338373830383938393f

Which means that after rendering it as ASCII, it would look like:

;4761739001010010=15122011143878089?

When **"All Data Elements Found" (BIT 8),** is specified in DFEF4B, Tag DFEF4D will be populated with a single instance of each requested data element, according to the following order:

Track 1 requested (bit 6 = 1). Includes first instance of:

Tag 56 = Track 1 Equivalent
Tag 5F21 = Track 1, identical to the data coded

Track 2 requested (bit 5 = 1). Includes first instance of:

Tag 57 = Track 2 Equivalent (converted to alpha numeric format)
Tag 9F6B = Track 2 Data
Tag 5F22 = Track 2, identical to the data coded

Track 3 requested (bit 4 = 1). Includes first instance of:

Tag 58 = Track 3 Equivalent
Tag 5F23 = Track 3, identical to the data coded

PAN requested (bit 7 = 1). Includes:

Tag 5A = PAN (converted to alpha numeric format)

Sentinels

For any found data element of Track1, Track2 or Track3, sentinels will be included or not included according to the preferences set in bits 1, 2 and 3.

Compressed Numeric Elements

For any data element captured as compressed numeric, the following rules shall apply:

Padding (0xf) shall not be included
Center separators: 0xd shall be converted to 0x3d ("=")
Data shall be encoded as ASCII representation of binary data
example 0x123f = 0x313233 = "123" (ignore padding)
example 0x1234 = 0x31323334 = "1234"
example 0x123d456f = 0x3132333d343536 = "123=456"

Tag DFEF4C

This tag's 6-byte value provides the native lengths of tracks 1, 2, and 3, and the PAN (if applicable). Two bytes are reserved for future use.

<Track 1 Length><Track 2 Length><Track 3 length><PAN length><RFU><RFU>

A length of 0 indicates *track disabled* in DFEF4B, or *data not available*.

This tag also serves as an indicator of which data element was found first, when "Only First Element Found" is enabled in DFEF4B.

Native lengths are provided to facilitate accurate removal of padding after decryption. Recall that in TDES-encrypted data, raw data gets null-padded up to a length that is a multiple of 8. In AES-encrypted data, the raw data is null-padded up to a length that is a multiple of 16.

Tag DFEF4D

This variable-length tag contains track and/or PAN data, encrypted. The exact contents will vary depending on values supplied previously in DFEF4B (see above).

When TDES or AES encryption have been used in conjunction with traditional DUKPT, you can decrypt the data using standard DUKPT methodology (ANSI X9.24), taking advantage of the 10-byte KSN found in tag DFEE12.

When TransArmor PKI data are present, decrypt with the KeyID value in DFEE12 and Terminal ID found in 9F1C.

(Note: Each track encoded with TransArmor PKI will be encrypted to 344 bytes.).