iDynamo 6

Secure Card Reader Authenticator Programmer's Manual (COMMANDS)

February 2023

Document Number:
D998200324-30

REGISTERED TO ISO 9001:2015

INFORMATION IN THIS PUBLICATION IS SUBJECT TO CHANGE WITHOUT NOTICE. MAGTEK CANNOT BE HELD LIABLE FOR ANY USE OF THE CONTENTS OF THIS DOCUMENT. ANY CHANGES OR IMPROVEMENTS MADE TO THIS PRODUCT WILL BE INCLUDED IN THE NEXT PUBLICATION RELEASE. IF YOU HAVE QUESTIONS ABOUT SPECIFIC FEATURES AND FUNCTIONS OR WHEN THEY WILL BECOME AVAILABLE, PLEASE CONTACT YOUR MAGTEK REPRESENTATIVE.

Table 0-1 - Revisions

Rev Number	Date	Notes
10	Nov 7, 2019	Engineering prototype release
18	May 20, 2020	Production release from MagneSafe V5 Master Programmer's Manual rev 18
20	Jun 8, 2021	Update from master programmer's manual Rev 20: **1.5** add features Pairing Mode Control, Apple VAS, Conserve DUKPT Keys, No EMV MSR Flow; **8.2** add Result Code 0x55 (Embedded V5 Head Only); **Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)** add Apple VAS options and add CAUTION statements about device power and key consumption; **Extended Command 0x0310 - Modify EMV Configuration (MAC, Contact Only)** clarify parameter value vs. LoA value and the nature of the full list vs. LoA list, remove "in some LoAs this is referred to as" text to reflect LoAs being harmonized across product family, add configurations C8 through C13; Appendix **G.1.1** label values of DFDF53 that are EMV MSR Flow Only; Update description of 9F6D and 9F6E; Add DFDF1F; Add checksums DF49 and DF4A; Add DF00, DF01 & DF02 , Misc. clarifications and corrections.
25	Nov 16, 2021	Update from master programmer's manual Rev 25: Update RoHS statement; Add Software License Agreement; Separate OEM Features feature into EMV Settings Unlock feature and Infinite Transaction Timeout feature, add both features to iDynamo 6; Add Suppress Sounds feature and add to iDynamo 6, and in **Extended Command 0x0304 - Cancel Transaction (EMV Only)** add optional silence sounds parameter; **Extended Command 0x0310 - Modify EMV Configuration (MAC, Contact Only)** add Vendor Config ID C14, C15; **Property 0x00 - Firmware ID** clarify length; **Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)** add result code 0x039D; **Property 0x10 - Interface Type** add One-Time Automatic interface type; Misc. clarifications and corrections.

Rev Number	Date	Notes
30	Feb 14, 2023	Add an option 0x03 Enhanced prompt cardholder in **Property 0x73 - Application Selection Behavior (Application Selection Options Only);** Add **Property 0x75 - Apple VAS Support (Apple VAS Only)**; Add **Property 0x76 - Sound Notification Control**, Add Error! Reference source not found.**;** Add **H.3 APPLE VAS Settings (Contactless Only, Apple VAS Only);** Add a Selection Type 0x10 Enhanced Application Selection in **Notification 0x0302 - Cardholder Selection Request (EMV Only);** Add a mode for Apple VAS version in **Extended Command 0x030B - Read EMV Kernel Information;** Add 0x03 = Try Another Interface, 0x04 = Application Blocked, 0x5D = Application Blocked to **Transaction Result Messages (EMV Only);** Add methodology to toggle/identify the protocol **Property 0x10 - Interface Type;);** Edit the Dual Mode of Apple VAS support in the options of **Extended Command 0x0300 - Initiate EMV Transaction (EMV Only);** Add **H.3 APPLE VAS Settings (Contactless Only, Apple VAS Only);** Add **Command 0x73 – Contactless Read Delay,** Misc. clarifications and corrections

Table of Contents

1 Introduction

1.1 About This Document

This document describes how to communicate with Secure Card Reader Authenticator (SCRA) devices which implement MagneSafe V5.

1.2 About SDKs

MagTek provides convenient SDKs and corresponding documentation for many programming languages and operating systems. The API libraries included in the SDKs wrap the details of the connection in an interface that conceptually parallels the device's internal operation, freeing software developers to focus on the business logic, without having to deal with the complexities of platform APIs for connecting to the various available connection types, communicating using the various available protocols, and parsing the various available data formats. Information about using MagTek wrapper APIs is available in separate documentation, including ***D99875535 SECURE CARD READER AUTHENTICATOR API PROGRAMMER'S MANUAL***.

The SDKs and corresponding documentation include:

- Functions for sending the direct commands described in this manual
- Wrappers for commonly used commands that further simplify development
- Sample source code to demonstrate how to communicate with the device using the direct commands described in this manual

To download the SDKs and documentation, search www.magtek.com for “SDK” and select the SDK and documentation for the programming languages and platforms you need, or contact MagTek Support Services for assistance.

Software developers also have the option to revert to direct communication with the device using libraries available in the chosen development framework. For example, custom software written in Visual Basic or visual C++ may make API calls to the standard Windows USB HID driver. This document provides information and support for developing host software using that method.

MagTek has also developed software that demonstrates direct communication with the device, which software developers can use to test the device and to which provides a starting point for developing other software. For more information, see the MagTek web site, or contact your reseller or MagTek Support Services.

1.3 About Terminology

The general terms “device” and “host” are used in different, often incompatible ways in a multitude of specifications and contexts. For example, “host” may have different a meaning in the context of USB communication than in the context of networked financial transaction processing. In this document, “device” and “host” are used strictly as follows:

- **Device** refers to the Secure Card Reader Authenticator (SCRA) that receives and responds to the command set specified in this document. Devices include Dynamag, eDynamo, and so on.
- **Host** refers to the piece of general-purpose electronic equipment the device is connected or paired to, which can send data to and receive data from the device. Host types include PC and Mac computers/laptops, tablets, smartphones, teletype terminals, and even test harnesses. In many cases the host may have custom software installed on it that communicates with the device. When “host” must be used differently, it is qualified as something specific, such as “acquirer host” or “USB host.”

Similarly, the word “user” is used in different ways in different contexts. This document separates users into more descriptive categories:

- The **cardholder**
- The **operator** (such as a cashier, bank teller, customer service representative, or server), and
- The **developer** or the **administrator** (such as an integrator configuring the device for the first time).

Because some connection types, payment brands, and other vocabulary name spaces (notably Bluetooth LE, EMV, smart phones, and more recent versions of Windows) use very specific meanings for the term “Application,” this document favors the term **software** to refer to software on the host that provides a user interface for the operator.

The combination of device(s), host(s), software, firmware, configuration settings, physical mounting and environment, user experience, and documentation is referred to as the **solution**.

1.4 About Connections and Data Formats

MagneSafe V5 products transmit data using a set of common data formats across a variety of physical connection layers, which can include universal serial bus (USB) acting as a keyboard (“USB KB”), USB acting as a vendor-defined HID device (“USB HID”), RS-232, Apple iAP (Lightning or USB), bidirectional audio connectors, Bluetooth, Bluetooth LE, and so on. The set of available physical connection types and the data formats available on each connection type is device-dependent. **Table 1-1** shows the physical connection types available on each product, and the data formats supported on each connection type for that device. Details about connection types and formats can be found in section **2 Connection Types** and section **3 Data Formats**. Section headings in this document include tags that indicate which connection types and/or data formats they apply to.

Table 1-1 - Device Connection Types / Data Formats

Product / Connection	Audio	Bluetooth LE GATT	Bluetooth LE GATT KB	Bluetooth	iAP1 Lightning	iAP2 Lightning	iAP2 USB	RS-232 / UART	SPI	USB HID	USB KB
BulleT KB				Streaming (MSR data)						HID	
BulleT SPP				Streaming							
cDynamo					Streaming						
Dynamag, Dynamag Duo, USB Enc IntelliHead V5										HID	Streaming
DynaMAX		GATT	Streaming							HID	
DynaPAD										HID	Streaming
DynaWave								SLIP		HID	
eDynamo		GATT								HID	
iDynamo 5					Streaming						
iDynamo 5 (Gen II)						Streaming					
iDynamo 6						SLIP	SLIP			HID	
kDynamo						SLIP					

Product / Connection	Audio	Bluetooth LE GATT	Bluetooth LE GATT KB	Bluetooth	iAP1 Lightning	iAP2 Lightning	iAP2 USB	RS-232 / UART	SPI	USB HID	USB KB
mDynamo										HID	
P-series and I-65 w/V5										HID	Streaming
pDynamo		GATT								HID	
sDynamo						Streaming					
SPI Enc IntelliHead V5									Streaming		
tDynamo		GATT								HID	
UART Enc IntelliHead V5								Streaming			
uDynamo	TLV									HID	

1.5 About Device Features

The information in this document applies to multiple devices. When developing solutions that use a specific device or set of devices, integrators must be aware of each device's connection types, data formats, features, and configuration options, which affect the availability and behavior of some commands. **Table 1-2** provides a list of device features that may impact command availability and behavior. All section headings in this document include tags that indicate which features they apply to.

Table 1-2 - Device Features

Feature / Product	BulleT KB BulleT SPP	cDynamo	Dynamag, USB Enc IntelliHead V5	Dynamag Duo	DynaMAX	DynaPAD	DynaWave	eDynamo	iDynamo 5	iDynamo 5 (Gen II)	iDynamo 6	kDynamo	mDynamo	P-series, I-65 w/V5	pDynamo	sDynamo	SPI Encrypting IntelliHead V5	tDynamo	UART Enc IntelliHead V5	uDynamo
MSR Swipe	Y	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	N	N	Y	Y	Y	Y	Y	Y
MSR Insert	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N
MSR 3-Track	Y	Y	Y	Y	Y	N	N	Y	Y	Y	Y	Y	N	Y	Y	Y	Y	Y	Y	
MSR Disable	N	Y	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N
MSR Swap Tracks 1/3	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
MSR Embedded V5 Head	N	N	N	N	N	N	N	N	N	Y	Y	Y	N	N	N	Y	N	Y	N	N
MSR Configurabe MSR Variants		Y	Y	Y	Y		N	Y	Y	Y	Y	Y	Y		Y	Y		Y		
MSR Configurable MP Variants		N	N	N	Y		N	Y	N	N	Y	Y	Y		Y	N		Y		
MSR SureSwipe		N	Y	Y	Y	Y	N	Y	N	N	N	N	N	Y	N	N	N	N	N	N
MSR JIS Support		Y	Y^3	N	N	N	N	N	Y	N	N	N	N	N	N	Y	Y	N	Y	
MSR SHA-1		N	Y	Y	Y	Y	N	Y	N	N	N	N	N		Y	N		N		
MSR SHA-256		N	N	N	N	N	N	N	N	N	N	N	N	N	N	N		N		
MSR Configurable SHA		N	N	N	Y		N	Y	N	N	N	N	N			N		N		
MSR MagneSafe 2.0							N	Y		N	N	N			N			N		
Configurable Encryption Algorithm	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N		N	N	N
Set Mask Service Code	N	N	Y^2	N	N	N	Y	N	N	N	N	N	N	Y^2	N	N	Y^2	N	N	N
Never Mask Service Code			N^2				N	Y	Y	Y	Y	Y	Y	N^2		Y	N^2	Y		

Feature / Product	BulleT KB BulleT SPP	cDynamo	Dynamag, USB Enc IntelliHead V5	Dynamag Duo	DynaMAX	DynaPAD	DynaWave	eDynamo	iDynamo 5	iDynamo 5 (Gen II)	iDynamo 6	kDynamo	mDynamo	P-series, I-65 w/V5	pDynamo	sDynamo	SPI Encrypting IntelliHead V5	tDynamo	UART Enc IntelliHead V5	uDynamo
EMV Contact	N	N	N	N	N	N	N	Y	N	N	Y	Y	Y	N	N	N	N	Y	N	N
EMV Contact 4.3i Format	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N
EMV Contactless	N	N	N	N	N	N	Y	N	N	N	Y	Y	N	N	N	N	N	Y	N	N
EMV ODA	N	N	N	N	N	N	Y	Y	N	N	N	Y	Y	N	N	N	N	Y	N	N
EMV MSR Flow	N	N	N	N	N	N	N	N	N	N	Y	Y	N	N	N	N	N	Y	N	N
No EMV MSR Flow	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N	Y	Y	Y	Y	Y	N	Y	Y
EMV Contact Quick Chip	N	N	N	N	N	N	N	Y[4]	N	N	Y	Y	Y[4]	N	N		N	Y	N	N
EMV Contactless Quick Chip	N	N	N	N	N	N	Y	N	N	N	Y	Y	N	N	N	N	N	Y	N	N
Infinite Transaction Timeout	N	N	N	N	N	N	Y	N	N	N	Y	N	N	N	N	N	N	N	N	N
EMV Settings Unlock	N	N	N	N	N	N	Y	N	N	N	Y	N	N	N	N	N	N	N	N	N
Silence Sounds	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N
Comprehensive Checksums	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N
Conserve DUKPT Keys	N	N	N	N	N	N	N	Y[7]	N	N	N	N	Y[7]	N	N	N	N	N	N	N
QuickPass Support	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N
Apple VAS	N	N	N	N	N	N	Y	N	N	N	Y	N	N	N	N	N	N	N	N	N
Application Selection Options	N	N	N	N	N	N	Y	N	N	N	Y	Y	N	N	N	N	N	Y	N	N
External PIN Accessory Support	N	N	N	N	N	N	Y	N	N	N	Y	N	N	N	N	N	N	N	N	N
Keypad Entry	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N	N	N
Fixed Key	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N
MSR Secondary DUKPT Key	N	N	N	N	Y	N	Y	Y	N	N	N	N	Y	N	Y	N	N	N	N	Y
Power Mgt Scheme (PM#)	1	N	N	N	2	N	N	3	N	N	7	5	N	N	6	N	N	5	N	4
Battery-Backed RTC							N	Y		N	N	N	N					N		
Transaction Validation	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N

Feature / Product	BulleT KB BulleT SPP	cDynamo	Dynamag, USB Enc IntelliHead V5	Dynamag Duo	DynaMAX	DynaPAD	DynaWave	eDynamo	iDynamo 5	iDynamo 5 (Gen II)	iDynamo 6	kDynamo	mDynamo	P-series, I-65 w/V5	pDynamo	sDynamo	SPI Encrypting IntelliHead V5	tDynamo	UART Enc IntelliHead V5	uDynamo
Display	N	N	N	N	N	Y*	N	N	N	N	N	N	N	N	Y	N	N	N	N	N
Multi-Language	N	N	N	N	N	N	Y	N	N	N	Y	Y	N	N	N	N	N	Y	N	N
Tamper	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N	N	N	N	N
Extended Commands	N	N	N	N	N	N	Y	Y	N	N	Y	Y	Y	N	N	N	N	Y	N	N
Extended Notifications	N	N	N	N	N	N	Y	Y	N	N	Y	Y	Y	N	N	N	N	Y	N	N
Dual USB Ports	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	Y	N	N
Pairing Modes	N	N	N	N	N	N	N	Y[5]	N	N	N	N	N	N	Y	N	N	Y	N	N
Pairing Mode Control	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N
Custom Advertising	N	N	N	N	N	N	N	Y[6]	N	N	N	N	N	N	Y	N	N	Y	N	N
Configurable iAP FID	N	Y	N	N	N	N	N	N	N	N	Y	Y	N	N	N	N	N	N	N	N
Auxiliary Ports	N	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N
Configurable Baud Rate	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	Y	N
Configurable Pushbutton	N	N	N	N	N	N	N	N	N	N	N	Y	N	N	N	N	N	N	N	N
External LED Control	N	N	N	N	N	N	N	N	N	N	N	N	Y[1]	N	N	N	N	N	N	N
Encrypt Bulk Data (b)	120	120	24	24	24	N	N	24	120	N	N	N	24	N	N	N	120	N	12	24

Feature / Product	BulleT KB BulleT SPP	cDynamo	Dynamag, USB Enc IntelliHead V5	Dynamag Duo	DynaMAX	DynaPAD	DynaWave	eDynamo	iDynamo 5	iDynamo 5 (Gen II)	iDynamo 6	kDynamo	mDynamo	P-series, I-65 w/V5	pDynamo	sDynamo	SPI Encrypting IntelliHead V5	tDynamo	UART Enc IntelliHead V5	uDynamo

1) This feature is available in mDynamo firmware revision ***1000003358D00*** (released August 2017) and newer.
2) This feature was introduced in SPI Encrypting IntelliHead V5 in firmware version 21042876C01 released July 2017, P-series and I-65 w/V5 in firmware version 21165822E01 released March 2018, Dynamag and USB Encrypting IntelliHead V5 in firmware version 21042840K00 released January 2019.
3) This feature is available in Dynamag and USB Enc IntelliHead V5 firmware version 21042840K00 (released January 2019) and newer.
4) EMV Contact Quick Chip is available in mDynamo firmware revision ***1000003358F01*** (released December 2017), eDynamo firmware revision ***1000003354F00*** (released October 2018), and newer.
5) Pairing Modes feature is available in eDynamo firmware revision ***1000002650B01*** and newer, except Bluetooth LE Property 0x13 which was added in ***1000002650C01***.
6) Custom Advertising feature is available in eDynamo firmware revision ***1000002650C02*** and newer, except Bluetooth LE Property 0x08 Configuration Bits "Never Advertise" and "USB Power Not Exit Airplane Mode" which were added in ***1000002650C01***.
7) This feature is only supported by the eDynamo starting with firmware revision ***1000003354J00*** and mDynamo starting with firmware revision ***1000003358G01***.

2 Connection Types

Table 1-1 on page **17** includes a list of connection types available for each device. The following subsections provide details developers will need to communicate with the device using each connection type.

2.1 How to Use USB Connections (USB Only)

NOTICE

This section describes how to use devices that use USB HID and Keyboard (KB mode) connections. For information about using the USB connection on Apple devices, see section 2.6 How to Use Apple iAP Connections (iAP Only) instead.

These USB devices conform to the USB specification revision 1.1. They also conform to the Human Interface Device (HID) class specification version 1.1. This document assumes the reader is familiar with USB HID class specifications, which are available at ***www.usb.org***. MagTek strongly recommends becoming familiar with that standard before trying to communicate with the device directly via USB.

These devices are full-speed, high-powered USB devices that draw power from the USB bus they are connected to. They enter and wake up from Suspend mode when directed to do so by the USB host. They do not support remote wakeup.

When connecting via USB, MagneSafe V5 devices connect to the USB host either as a vendor-defined HID device ("HID") or as an HID Keyboard Emulation device ("KB"), depending on the device type and configuration. Details for using the device in each of these modes are provided in the sections that follow. In addition to connecting to the USB host as different USB device types depending on their mode, the device can transmit data in different formats (see section **3 Data Formats**). To decode data coming from HID devices, see section **3.1 How to Use HID Format (HID Only)**. To decode data coming from KB devices, see section **3.3 How to Use Streaming Format (Streaming Only)**.

The devices have an adjustable endpoint descriptor polling interval value that can be set to any value in the range of 1ms to 255ms. To change the setting, use **Property 0x02 - USB Polling Interval (HID Only | KB Only)**.

MagneSafe V5 devices identify themselves to the host with MagTek's vendor ID **0x0801** and a Product ID (PID) from this list:

- tDynamo reports **0x001C**.
- kDynamo reports **0x001D**.
- DynaWave reports **0x001E**.
- iDynamo 6 reports **0x001F**.
- MSR Swipe devices report PID **0x0011** when in HID mode.
- Audio devices report PID **0x0017** when in HID mode.
- Devices that implement a combination of EMV Contact / EMV Contactless / MSR swipe report PID **0x0019** when in HID mode.
- EMV-only devices (such as mDynamo and DynaWave) report PID **0x001A** when in HID mode.
- All devices report PID **0x0001** when in KB mode.
- Wireless USB device dongles report PID **0x0011** when in HID mode.

- Wireless USB device dongles report PID **0x0001** when in KB mode.
- Wireless USB devices report PID **0x0014** when plugged directly into the host with a USB cable.

2.1.1 About USB Reports, Usages, Usage Pages, and Usage IDs

All USB HID devices send and receive data using **Reports**. Each report can contain several sections, called **Usages**, each of which has its own unique four-byte (32-bit) identifier. The two most significant bytes of a usage are called the **usage page**, and the two least significant bytes are called the **usage ID**. Vendor-defined HID usages must have a usage page in the range **0xFF00 - 0xFFFF**, and it is common practice for related usage IDs share the same usage page. For these reasons, all usages for MagneSafe V5 devices use vendor-defined usage page **0xFF00**, **Magnetic Stripe Reader**.

HID reports used by the host can be divided into two types:

- **Feature Reports**, which the host uses to send commands to the device. Feature reports can be further subdivided into **Get Feature** and **Set Feature** types. MagneSafe V5 devices only use one feature report.
- **Input Reports** are used by the device to send unsolicited notifications to the host when the device's state changes, or to send asynchronous responses to the host when a command completes. The device commonly uses input reports when reporting unpredictable cardholder interactions, or when a command takes more time for the device to process than is reasonable for the host to wait on a blocking call for the device to acknowledge completion.

For information about using feature reports to send commands to the device and receive responses from the device, see section **2.1.2 How to Send Commands On the USB Connection**. For information about receiving unsolicited data from the device via Input Reports, see section **2.1.3 How to Receive Data On the USB Connection (HID Only)**.

2.1.2 How to Send Commands On the USB Connection

Because many MagneSafe V5 devices support connection types beyond USB, this documentation abstracts host-device communication by referring to **Commands**, which are most often a pairing of a **Request** from the host and a corresponding **Response** from the device This section explains how these terms apply when using the USB HID connection.

When the device is connected to the host via USB, regardless of whether it identifies and operates as a vendor-defined HID device or as a keyboard, the host sends a **Set Feature Report** to the device to send the requests for **Commands**, and sends a **Get Feature Report** to the device to retrieve a synchronous response when appropriate. All reports use Usage Page **0xFF00**, Usage ID **0x20**, and no Feature Report ID (Extended Commands Only) or, on devices that support Extended Commands, Feature report ID **0x01**.

The host should send both Feature Report types using the default Control pipe using a blocking call to the operating system's native USB libraries. The device NAKs the Status page of a **Set Feature Report** until it finishes the requested operation, and if it does not respond, the operating system will generally time out and report failure. This method ensures that as soon as the device has fulfilled the command request embedded in the **Set Feature Report**, the host software can immediately call a follow-up **Get Feature Report** to retrieve the command response, if one is required, and that the host software will not hang on a blocking call indefinitely.

The host should follow this general command sequence to send a request and receive a response:

1) Choose the command to invoke from section **8 Commands**. Every command has a corresponding **Command Number** listed in the header of its documentation section.
2) Construct a **Command Request Data** value using the Request table in the documentation for the command.
3) Determine the length of the **Command Request Data** value, referred to as the **Command Request Data Length**.
4) Examine the device's Report Descriptor to determine what payload length the device expects for a **Set Feature Report** and **Get Feature Report** (the operating system libraries may refer to this length as the "Report Length" or "Report Count").
5) Pad the **Command Request Data** value with 0x00 so the total length of the payload is consistent with the Set Feature Report's Report Length / Report Count.
6) Construct a Set Report Structure using **Table 8-1** in section **8.1 About Commands**.
7) Send a **Set Feature Report** containing the finalized padded Set Report Structure. The call to send the report may succeed, fail on a timeout, or fail for some other reason.
8) If the call succeeds, send a **Get Feature Report** to retrieve the device's response in the Get Report Structure shown in **Table 8-2** in section **8.1 About Commands**.
9) Parse the Get Report Structure, and truncate the **Command Response Data** field to the provided **Command Response Data Length**.
10) Examine the **Result Code**, which is a one-byte value the device sends to indicate success or the failure mode of the command. See section **8.2 About Result Codes** for more detail.
11) Parse the truncated **Command Response Data** field using the Response table in the documentation for the command.

In very rare cases, the host may simply send a **Get Feature Report** directly without a preceding **Set Feature Report**. The **Commands** documentation specifies these special cases if they exist.

(Extended Commands Only)

Commands that use two-byte Command Numbers are called **Extended Commands**. Generally these are commands that require a **Data Length** that is longer than the number of bytes available in a single report. The host must call these commands using **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**. Similarly, commands that send responses greater than the number of bytes available in a single report require the host to use **Command 0x4A - Get Extended Response (Extended Commands Only)** to retrieve Extended Responses. See the documentation for those two commands for details about how Extended Commands and Extended Responses work.

2.1.3 How to Receive Data On the USB Connection (HID Only)

When the device communicates with the host as a vendor-defined HID device, it sends unsolicited messages such as card data to the host via one or more **Input Reports**, which are asynchronous data packets (i.e., events) sent from the device to the host using the USB **Interrupt IN** pipe. Events occur when the device state changes or when an asynchronous command (such as a command that requires cardholder interaction) has reached a pre-defined event, such as completion. Per the USB HID standard, the host polls the device on a regular Polling Interval to see if it has input data available to send. If the device does not, it responds to the poll with a USB `NAK`.

Devices that do not support "Extended Notifications" (a specific way of sending asynchronous data to the host, see section **1.5 About Device Features**) implement a single input report for **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)**. Because these devices only implement one input report, the input report they send to the host does not include a report identifier, in accordance with the USB HID specification.

The host can locate a specific data element in the input report for **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)** by finding the corresponding Usage and interpreting its contents as binary data. For example, the host software can find **Track 1 Decode Status (HID Only | TLV Only | GATT Only | SLIP Only)** as follows:

1) Knowing from section **2.1.1** that the device uses usage page `0xFF00`, and knowing from the "Where to Find Value" column in the first table of section **6.2** that the desired data is found in the usage with Usage ID `0x0020`, call the platform's USB SDK to retrieve the data from usage `0xFF000020` in the input report.
2) Interpret the single binary data byte from that Usage according to the second table in section **6.2**.

(Extended Notifications Only)

The remainder of this section provides important information about compatibility between host software designed for other MagneSafe V5 devices and this device.

Devices that support **Notification Messages Sent from Device to Host (Extended Notifications Only)** implement a HID input report specifically for sending notification message packets. Because the USB HID specification requires any device with more than one report of the same type to use HID report identifiers, such devices include a report identifier with every report:

- **Magnetic Stripe Card Data Sent from Device to Host** uses **Input Report ID 1** for card data. The card data report descriptor is typically included even if the device does not send card data, to allow for future flexibility.
- **Notification Messages Sent from Device to Host (Extended Notifications Only)** use **Input Report ID 2** for asynchronous notifications [see section **7**].

Host software written for devices that support notification messages must specify these report identifiers when sending or retrieving reports to communicate with the device. Some pre-existing host software for Windows may expect to see report identifier zero, which the platform APIs may send when report IDs are not in use; this may need to change for compatibility with devices that use Extended Notifications.

The host can determine the size of notification message packet Input Reports by looking at the HID report descriptor. Notification message packet reports are generally 63 bytes long. If a notification message can't fit into one packet, the device sends multiple packets, each containing the notification message packet format in **Table 7-2** and partial notification message data.

The host can locate a specific data element in a notification input report by finding the corresponding Usage and interpreting its contents as binary data. For example, upon receiving an input report with **ID 2**, the host software can find the message payload as follows:

1) Knowing from section **2.1.1** that the device uses usage page `0xFF00`, and knowing from the "Where to Find Value" column in the first table of section **7.1** that the desired data is found in the usage with Usage ID `0x0020`, call the platform's USB SDK to retrieve the data from usage `0xFF000020` in the input report.
2) Interpret the blob of data from that Usage according to the second table in section **7.1**.

2.1.4 How to Use the USB Connection in Keyboard Emulation Mode (KB Only)

NOTICE

(SureSwipe Only)
When the device is set to Security Level 2, the factory default setting is for the device to transmit data in SureSwipe format, and this section does not apply. See *D99875206 TECHNICAL REFERENCE MANUAL, USB KB SURESWIPE & SWIPE READER* **instead.**

When the device is operating in USB keyboard emulation ("KB") mode (see **Property 0x10 - Interface Type**], it expects to receive commands and send command responses using HID format (see section **2.1.2 How to Send Commands On the USB Connection**), and sends **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)** using Streaming format [see section **3.3.1 Magnetic Stripe Card Data In Streaming Format)]** as follows:

A device in KB mode identifies itself to the USB host as a keyboard, and transmits streaming data to the host as ASCII as though it is being typed by a person on an actual keyboard. It does this by mapping each of the possible ASCII characters in the stream to keystrokes. By default, to send an ASCII character to the host, the device looks up the ASCII character in the key map [see **Property 0x16 - Active Keymap (KB Only, MSR Only)**] and retrieves a combination of a single **Key Usage ID** (defined in **Appendix D Keyboard Usage ID Definitions**), which is a unique value assigned to every keyboard key, and a **Key Modifier Byte** (defined in appendix **D.2 Modifier Byte Definitions**), and sends them to the host. The key modifier byte modifies the meaning of the key usage ID, by indicating whether any combination of the right or left **Ctrl**, **Shift**, **Alt** or GUI keys [as defined by ***Universal Serial Bus (USB) Device Class Definition for Human Interface Devices (HID)***] are pressed at the same time as the key usage ID.

The device transmits ASCII 0 to 31 and 127 as their equivalent control code combinations. For example, for a carriage return value 13 (`0x0D`), the device appears to the host as a keyboard where a person very quickly presses and holds the **Ctrl** key, then presses the **M** key, then releases both keys.

When the keymap contains a Key Usage ID and Key Modifier Byte of `0xFF` for the ASCII value the device wants to send, or if **Property 0x17 - ASCII to Keypress Conversion Type (KB Only, MSR Only)** is set to **Alt ASCII Code**, the device uses **Alt** ASCII code keystrokes instead of key map values, meaning it simulates holding down the **Alt** key on a keyboard and typing the three-digit decimal value of the ASCII character it wants to send. For example, to transmit the ASCII character '?' (063 decimal in the ASCII table), the device sends keypad '0' combined with the **Left Alt** key modifier, then keypad '6' combined with the **Left Alt** key modifier, then keypad '3' combined with the **Left Alt** key modifier.

NOTICE

Because the host perceives a KB mode device as a keyboard, pressing keys on another keyboard connected to the host while the device is transmitting may corrupt the data the host receives.

2.2 How to Use Bluetooth LE Connections (Bluetooth LE Only)

This section provides information about developing software for a Bluetooth LE-capable host that needs to communicate with the device using Bluetooth Low Energy (Bluetooth LE). It assumes **Bluetooth LE Property 0x11 - Bluetooth LE Connection Type (MSR Only, KB Only)** is set to GATT, meaning the device is configured to behave as a vendor-defined GATT device, as opposed to a Bluetooth LE keyboard ("KB mode"). In this arrangement, the device acts as a Bluetooth LE server/peripheral, and the host acts as a client/central.

2.2.1 About GATT Characteristics

When **Bluetooth LE Property 0x11 - Bluetooth LE Connection Type (MSR Only, KB Only)** is set to GATT, the device uses the set of GATT characteristics below.

Table 2-1 - <DeviceName> GATT Service Characteristic

Characteristic Name	<DeviceName> GATT Service
Properties	Read
Data Size	N/A
UUID (LSB Order)	For eDynamo/tDynamo: 03:01:B6:0C:41:E3:43:F8:8F:89:82:AD:F8:E6:08:05
Description/Usage	Used to identify the Service.

Table 2-2 - Command Data Characteristic

Characteristic Name	Command Data
Properties	Read/Write
Data Size	Variable (currently 60 bytes maximum but may increase).
UUID (LSB Order)	00:02:B6:0C:41:E3:43:F8:8F:89:82:AD:F8:E6:08:05
Description/Usage	Contains the command data in USB HID feature report format without the fixed report size and padding (see **Table 8-1** and **Table 8-2** in section **2.1.2 How to Send Commands On the USB Connection** for details). The data length field of the feature report is used to determine the length to be read or written. The length of the characteristic is 2 + the data length field value.

Table 2-3 - Card Data Characteristic

Characteristic Name	Card Data
Properties	Read, Notify
Data Size	Variable (512 max)
UUID (LSB Order)	01:02:B6:0C:41:E3:43:F8:8F:89:82:AD:F8:E6:08:05
Description/Usage	Contains the card data. If the host software configures the **Card Data** characteristic to enable notifications, the device sends card data to the host in multiple notification messages when a card is swiped. This is the fastest way for the device to transmit card data. The first byte of each notification message always contains the block identifier of the card data, starting with block 0 for the first message and incrementing in subsequent messages. The host software can use the block identifier field to detect whether blocks have been lost due to communication loss or out-of-range problems. The remaining bytes of each notification message contain card data. After the device has sent all card data, it sends one more notification message with block identifier **0xFF** to indicate all the card data has been sent. This last notification message also contains a second byte indicating the total number of blocks of card data it transmitted. If the host configures the **Card Data** characteristic to disable notifications (default configuration), the device does not include block identifier fields in the blocks; the read simply fails if a communication error occurs. The host software must read the card data from the **Card Data** characteristic in blocks using long reads, and must use the **Data Ready** and **Data Read Status** characteristics.

Table 2-4 - Data Ready Characteristic

Characteristic Name	Data Ready
Properties	Notify
Data Size	4
UUID (LSB Order)	02:02:B6:0C:41:E3:43:F8:8F:89:82:AD:F8:E6:08:05
Description/Usage	Contains the characteristic identifier (byte 0), characteristic block identifier (byte 1), and the block length (byte 2 and 3 LSB first) of the data that is ready to be read. The characteristic identifiers are defined as 0 = Command data, 1 = Card or notification data. The first block of card or notification data is block 0, the second block is block 1, and so on. The host software knows it has received all available data when the data block is less than 512 bytes long. If the last block of card data happens to be exactly 512 bytes long, the device sends an additional **Data ready** notification with a block length of zero.

Table 2-5 - Data Read Status Characteristic

Characteristic Name	Data Read Status
Properties	Write
Data Size	3
UUID (LSB Order)	03:02:B6:0C:41:E3:43:F8:8F:89:82:AD:F8:E6:08:05
Description/Usage	Contains the characteristic identifier (byte 0), characteristic block identifier (byte 1), and the read status (byte 2) of the data that was ready to be read. The host software should write a 0 to this characteristic after reading a block of card data, to notify the device it is ready to read the next block of card data, at which point the device posts the next block of data to the Data ready characteristic. The device does not accept any more card swipes until the host writes to this characteristic. If the host fails to write to this characteristic within 10 seconds of being notified a card data block is ready, the device terminates the transaction and discards all card data.

2.2.2 How to Connect to a Device Using Bluetooth LE

The general steps for a host to communicate with the device via Bluetooth LE are as follows:

1) Scan for nearby Bluetooth LE peripherals advertising the desired GATT service UUID.
2) If multiple devices of the desired type are available, examine each device's name property. A specific device's default name is a constant, and by default is equal to the product name plus a hyphen plus the serial number on the device label.
3) Establish a Bluetooth LE connection with the device.
4) Pair with the device using passkey 000000. In many cases this step is operator-driven.
5) Make sure, if the host is expecting to receive data from any Bluetooth LE characteristics, those characteristics are configured to enable notifications (see section **2.2.1 About GATT Characteristics**). The specific method to enable notifications for a characteristic is different in different Bluetooth LE development libraries. For example, iOS code would be similar to `[servicePeripheral setNotifyValue:YES forCharacteristic:characteristic]`.
6) Send commands to the device (see section **2.2.3 How to Send Commands On the Bluetooth LE Connection**) and process incoming messages from the device (see section **2.2.4 How to Receive Data On the Bluetooth LE Connection**).

2.2.3 How to Send Commands On the Bluetooth LE Connection

To send a command request and to receive the command response, the host should do the following:

1) Make sure it is connected to the device (see section **2.2.2 How to Connect to a Device Using Bluetooth LE)**.
2) Write the command request data to the **Command Data** characteristic (see section **2.2.1 About GATT Characteristics**).
3) Wait to receive a **Data Ready** notification with the characteristic identifier set to 0 (command data).
4) Read the command response data from the **Command Data** characteristic.
5) Interpret the data according to section **3.2 How to Use GATT Format (GATT Only)**.

For a full list of commands and details about how to use them, see section **8 Commands**.

2.2.4 How to Receive Data On the Bluetooth LE Connection

This section describes how the device sends unsolicited messages (messages that are not the direct response to a command) to the Bluetooth LE host. This includes **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)**, and **Notification Messages Sent from Device to Host (Extended Notifications Only)**.

Some of the details in this section may be abstracted by the libraries in the development framework used to write the host software. For general information about Bluetooth LE and the associated terms, see the Bluetooth specifications found at https://www.bluetooth.org/Technical/Specifications/adopted.htm.

In the normal operating mode for the device in GATT HID Vendor Defined mode, the device is always advertising when not connected. The Bluetooth LE host is responsible for optimizing the device's power consumption by only connecting when needed. If the Bluetooth LE host is not able to disconnect directly through its Bluetooth LE API, it can force the device to disconnect by using **Bluetooth LE Command 0x0B - Terminate Bluetooth LE Connection**.

To receive card data when the **Card Data** characteristic is configured to send notifications, the host software should do the following:

1) Make sure it is connected to the device (see section **2.2.2 How to Connect to a Device Using Bluetooth LE)**.
2) Wait to receive a **Card Data** notification.
3) If the block identifier is 0xFF (no more card data), all card data has been received. Otherwise, continue to wait to receive more **Card Data** notifications.
4) Verify the number of card data blocks received equals the **number of card data blocks sent** field contained in the last notification message. A mismatch indicates a transmission error occurred.
5) Interpret the data according to section **3.2 How to Use GATT Format (GATT Only)**.

To receive card data when the **Card Data** characteristic is not configured to send notifications, the host should do the following:

1) Make sure it is connected to the device (see section **2.2.2 How to Connect to a Device Using Bluetooth LE)**.
2) Wait to receive a **Data Ready** notification with the characteristic identifier set to 1 (card data).
3) If the **length** field of the **Data Ready** notification is greater than zero, read the block of card data from the **card data** characteristic.
4) Write the **data read status** characteristic with the characteristic identifier, block identifier, and read status of the card data block that is done being read.
5) If the length field of the data ready notification is less than 512, all data has been received. Otherwise, loop back to receive more **data ready** notifications with characteristic identifier set to 1.
6) Interpret the data according to section **3.2 How to Use GATT Format (GATT Only)**.

2.2.5 How to Use the Bluetooth LE Connection In Keyboard Emulation Mode (KB Only)

When a Bluetooth LE device is configured to behave like a keyboard ["Keyboard Mode" or "KB" for short, see **Bluetooth LE Property 0x11 - Bluetooth LE Connection Type (MSR Only, KB Only)**], it uses the same data format used by USB devices configured to use KB mode. For details, see section **2.1.4 How to Use the USB Connection in Keyboard Emulation Mode (KB Only)**.

2.3 How to Use UART and RS-232 Connections (RS-232 Only | UART Only)

When the device is communicating with the host via a serial connection [RS-232 UART or logic level UART, see **Property 0x10 - Interface Type**], it uses one of two formats. See section **1.4 About Connections and Data Formats** to determine which of these format rules the device uses:

- Devices identified in **Table 1-1** as using **Streaming** format use that format to receive commands, send command responses, and send **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)**. See section **3.3 How to Use Streaming Format (Streaming Only)**.
- Devices identified in **Table 1-1** as using **SLIP** format use that format to send and receive commands and send command responses. See section **3.4 How to Use SLIP Format (SLIP Only)**.

2.4 How to use SPI Connections (SPI Only)

2.5 How to Use Audio Connections (Audio Only)

This section provides general information about developing software for an iOS host or an Android host that needs to communicate with the device via the audio connector. In this arrangement, the host sends the device command requests and receives responses from the device using TLV data objects. See section **3.5 How to Use Tag-Length-Value (TLV) Format** for details.

Decoding incoming audio data into a set of TLV formatted values using operating system SDKs is very complex; the device transmits data to the host as a raw PCM audio stream with Manchester encoded values, and the host software must examine the PCM stream directly and decode the data. Because of the complexity involved in stream decoding, MagTek strongly recommends developers of custom host software use ***99510109 DYNAMAG, DYNAMAX, EDYNAMO, UDYNAMO, ADYNAMO, BULLET, MDYNAMO, DYNAWAVE, TDYNAMO, IDYNAMO 6, DYNAGLASS SDK FOR ANDROID***, or ***99510111 DYNAMAX, EDYNAMO, UDYNAMO, ADYNAMO, IDYNAMO, KDYNAMO, SDYNAMO, TDYNAMO SDK FOR IOS***, which handle decoding of incoming data and presents it as easy-to-use objects.

Many of the values sent in the HID report over USB when a card is swiped are also sent to the Audio connection (card swipe data is sent only on the active connection). The elements that can be sent in both are noted in the descriptions in section **6 Magnetic Stripe Card Data Sent from Device to Host**.

2.6 How to Use Apple iAP Connections (iAP Only)

This section provides information about developing an iOS app that interfaces with the device via the Lightning or USB connector using iPod Accessory Protocol (iAP). For sample code and other supporting materials, see ***99510111 DYNAMAX, EDYNAMO, UDYNAMO, ADYNAMO, IDYNAMO, KDYNAMO, SDYNAMO, TDYNAMO SDK FOR IOS***, available from MagTek.

To develop host software for an iOS host that connects to the device, you must know the following device properties, which are specified by the purchaser when ordering, and loaded by the manufacturer:

- ***BundleSeedIDString***, which is a 10-character string assigned by Apple, Inc. to the host software developer
- ***protocolString***, also known as the SDK Protocol, usually in the form of a reverse DNS string unique to the host software developer or the device purchaser.

The host software project must include the protocolString in its ***.plist*** file before compiling. Spelling, including punctuation and capitalization, must exactly match the protocolString of the device.

The host software should initiate a connection to the device using the iOS SDK's ***ExternalAccessory*** Framework (for sample code, see Apple's ***EADemo*** app). Upon establishing the connection, the host can begin exchanging data with the device. Devices may use different formats to send and receive different types of data on different connections, or may change their behavior based on configuration. To determine the data format to use, look up the device and connection type in **Table 1-1**. For details about using **Streaming** format, see section **3.3 How to Use Streaming Format (Streaming Only)**. For details about using **SLIP** format, see section **3.4 How to Use SLIP Format (SLIP Only)**.

On some devices, code upgrade commands are not available through this connection.

2.7 How to Use Bluetooth Connections (Bluetooth Only)

This section provides information about developing software for a Bluetooth-capable host that needs to communicate with the device using Bluetooth. For information about using Bluetooth Low Energy (Bluetooth LE) devices, see section **2.2 How to Use Bluetooth LE Connections** instead.

3 Data Formats

3.1 How to Use HID Format (HID Only)

When the device and host are communicating in vendor-defined HID mode, data comes from the device as described in section **2.1.3 How to Receive Data On the USB Connection (HID Only)**. The host software can retrieve the incoming data by examining the various usages in the report(s). For details about which usages to examine and how to interpret the data, see section **6 Magnetic Stripe Card Data Sent from Device to Host** for card data, and section **7 Notification Messages Sent from Device to Host (Extended Notifications Only)**.

3.2 How to Use GATT Format (GATT Only)

When operating as a vendor-defined GATT device, the device may send **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)** or **Notification Messages Sent from Device to Host (Extended Notifications Only)** in either normal or RLE format, depending on whether RLE would help compress the data or not. The host software should understand both formats. Regardless of whether the device has GATT notifications enabled or disabled (see section **2.2.1 About GATT Characteristics** and section **2.2.4 How to Receive Data On the Bluetooth LE Connection**), the first byte of the card data block contains the GATT card data format field, which indicates what type of data it is sending and whether the data is RLE compressed as follows:

- 0x00 = **Card Data Normal**, which indicates the card data payload contains uncompressed card data in USB HID vendor defined report format (see section **6 Magnetic Stripe Card Data Sent from Device to Host**).
- 0x01 = **Card Data RLE**, which indicates the card data payload contains run-length-encoded compressed card data in USB HID vendor-defined report format (see section **6 Magnetic Stripe Card Data Sent from Device to Host** and the information below about RLE decoding).
- 0x02 = **Notification Uncompressed**, which indicates the payload contains an uncompressed notification message [see section **7 Notification Messages Sent from Device to Host (Extended Notifications Only)**].
- 0x03 = **Notification RLE**, which indicates the payload contains a run-length-encoded compressed notification message [see section **7 Notification Messages Sent from Device to Host (Extended Notifications Only)** and the information below about RLE decoding].

The device implements RLE as follows:

1) Any byte that is repeated more than once consecutively is run length encoded. Bytes that are not repeated stay as-is.
2) Repeated bytes are run-length encoded by repeating the byte twice, followed by the number of times the byte was repeated in the original data.
3) The maximum length of an encoded run is 255, so runs larger than 255 bytes are encoded as multiple runs of 255 bytes each until the last run.

For example, the data `0x44 0x55 0x55 0x55 0x55 0x55 0x55 0x55 0x55 0x055 0x66 0x00 0x00` is encoded as `0x44 0x55 0x55 0x09 0x66 0x00 0x00 0x02`. A run of 260 `0x00` bytes would be encoded as `0x00 0x00 0xFF 0x00 0x00 0x05`.

The second and third byte of card data and notification data contain the uncompressed data payload field size in big endian order.

The fourth byte onward contains the data for the **Magnetic Stripe Card Data Sent from Device to Host** or **Notification Messages Sent from Device to Host (Extended Notifications Only)**.

The data size for command data and card data may increase with firmware updates, so the host software should be able to adapt to this. Adapting can be as simple as ignoring any extra data bytes that are not understood or expected.

If the **Card Data** characteristic (see section **2.2.1 About GATT Characteristics**) is not configured to use notifications, the maximum notification message packet data length is the maximum characteristic size allowed by the Bluetooth LE specification (512), times the maximum number of block identifiers (256) = 131072 bytes minus headers (3 + 8) = 131061 bytes, which is large enough to fit a maximum sized notification message with a complete data length of 65535 bytes without splitting it into multiple packets.

If the **Card Data** characteristic (see section **2.2.1 About GATT Characteristics**) is configured to use notifications, the maximum notification message partial data length supported by the protocol is the maximum notification payload size (19), times the maximum number of block identifiers (255) = 4845 bytes - headers (3 + 8) = 4834 bytes.

3.3 How to Use Streaming Format (Streaming Only)

This section describes how the device functions when it is using Streaming format on its current connection to the host. Some device connection types use streaming format for both commands/responses and for magnetic stripe data, while other device connection types use streaming format just for magnetic stripe data. The following sections describe each of these separately.

3.3.1 Magnetic Stripe Card Data In Streaming Format (Swipe Only | Keypad Entry Only)

In streaming format, the device sends **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)** as a series of potentially variable length fields in a fixed order, separated by delimiters. Many of the delimiters are configurable, which allows the device to output customized sequences of characters to the host. These options are most commonly used when the device communicates with the host as if the device were a keyboard [see section **2.1.4 How to Use the USB Connection in Keyboard Emulation Mode (KB Only)**], where developers may configure the delimiters to drive the host's user interface to advance from one user interface field to the next, or to submit a filled out form.

Streaming data is composed entirely of ASCII characters, but the host should interpret the characters differently depending on the nature of the data. The tables in section **6** that describe where to find these values in the data stream also describe how to decode them:

- **ASCII** fields like **Masked Track Data**, **Device Serial Number**, and **Format Code (Streaming Only)** simply contain the data as ASCII characters.
- **Binary** fields like **Device Encryption Status**, **Encrypted Track Data**, **MagnePrint Status**, **Encrypted MagnePrint Data**, **Encrypted Session ID**, **DUKPT Key Serial Number**, **Clear Text CRC (Streaming Only)**, and **Encrypted CRC (Streaming Only)** are hexadecimal encoded, where the contents consist only of the characters 0123456789ABCDEF, and every two bytes represents the hexadecimal value of the binary byte being sent. The host should decode every two characters as one byte.

The delimiters the device sends between fields are stored as **Properties** in the device's non-volatile memory, which the host can configure using **Command 0x01 - Set Property (MAC)**. **Table 3-1** shows the format the device uses to transmit magnetic stripe data in Streaming mode, where the delimiter properties are abbreviated as a "P" followed by the property number. When transmitting card data to the host, the device replaces each bracketed [P0x##] value with the actual value contained in the specified property, and replaces other bracketed values with card data. For information about a specific property's valid values and effects on device behavior, see its documentation in section **9 Properties**.

Table 3-1 - Card Data Format (Streaming Mode)

Card Data Format
[P0x1E] [P0x20] [P0x24] [**Track 1 Masked Data**] [P0x2B or P0x2D] [P0x21] [P0x20] [P0x1C or P0x25 or P0x28] [**Track 2 Masked Data**] [P0x1D or P0x2B or P0x2E] [P0x21] [P0x20] (Only if exists: [P0x26 or P0x27 or P0x29] [**Track 3 Masked Data**] [P0x2B or P0x2F] [P0x21]) [P0x1F] [P0x23] [**Device Encryption Status**] [P0x23] (Encrypted together: [P0x24] [**Track 1 Encrypted Data**] [P0x2B or P0x2D]) [P0x23] (Encrypted together: [P0x1C or P0x25 or P0x28] **Track 2 Encrypted Data**] [P0x1D or P0x2B or P0x2E]) [P0x23] (Encrypted together: [P0x26 or P0x27 or P0x29] [**Track 3 Encrypted Data**] [P0x2B or P0x2F]) [P0x23] [**MagnePrint Status**] [P0x23] [**Encrypted MagnePrint Data**] [P0x23] [**Device Serial Number**] [P0x23] [**Encrypted Session ID**] [P0x23] [**DUKPT Key Serial Number**] [P0x23] [**Remaining MSR Transactions**] (optional, off by default) [P0x23] [**Clear Text CRC (Streaming Only)**] [P0x23] [**Encrypted CRC (Streaming Only)**] [P0x23] [**Format Code (Streaming Only)**] [P0x22]

If the device detects an error on a track, it transmits ASCII character “E” in place of the track data to indicate an error.

The device sends the **Device Encryption Status** value to help the host parse and interpret the incoming stream of data:

- The device only encrypts data if Encryption Enabled (bit 2) and Initial DUKPT Key Injected (bit 1) are set. Otherwise, it instead sends data it would usually encrypt as clear text in ASCII HEX format, and does not include the **DUKPT Key Serial Number**.
- When the DUKPT Keys Exhausted (bit 0) is set, the device no longer reads cards, and the card data format in **Table 3-1** excludes **MagnePrint Status**, **Encrypted MagnePrint Data**, **Masked Track Data**, **Encrypted Track Data**, and all corresponding Pre-Track Strings, Start Sentinels, End Sentinels, and Post-Track Strings. All other delimiters and data elements remain the same.

3.3.2 Commands and Responses In Streaming Format

If section **2 Connection Types** says the connection the device and host should use streaming format to exchange **Commands** and their corresponding responses over the selected connection type, all command and response messages are composed of a series of hexadecimal values encoded as two readable ASCII characters (‘0’ through ‘F’ only) per byte.

Each command should include the data shown in **Table 3-2**, and each response includes the data shown in **Table 3-3**.

Table 3-2 - Command Format for Streaming

Byte	Meaning
0..n	Command Request Message from Table 8-1 in section **8.1 About Commands**
n+1	Carriage Return (ASCII 0x0D)

Table 3-3 - Response Format for Streaming

Byte	Meaning
0..n	Command Response Message from **Table 8-2** in section **8.1 About Commands**
n+1	Carriage Return (ASCII 0x0D)

For example:

- A command with a one byte parameter in the **Command Request Data** field in Table 8-1 would send ASCII ‘0’ (0x30), ASCII ‘1’ (0x31) representing a length of `0x01`; a command with 18 bytes of data would send ASCII ‘1’ (0x31), ASCII ‘2’ (0x32) representing a length of `0x12`.
- To send **Command 0x00 - Get Property** to get **Property 0x03 - Device Serial Number**, the host would send a stream consisting of ASCII ‘0’ (0x30), ASCII ‘0’ (0x30) for Command Number 0x00, ASCII ‘0’ (0x30), ASCII ‘1’ (0x31) for Data Length 0x01, ASCII ‘0’ (0x30), ASCII ‘3’ (0x33) for Data, and a Carriage Return (0x0D) to signal the end of the message.
- The device’s responses are encoded similarly.

3.4 How to Use SLIP Format (SLIP Only)

When the host and device exchange data using SLIP format, all messages are composed of a series of binary values between 0x00 and 0xFF.

The SLIP format is defined in Part D, Section 3 of ***Specification of the Bluetooth System, Host Controller Interface, Volume 4***, which is available at https://www.bluetooth.org/Technical/Specifications/adopted.htm. Note the reference to bluetooth.org is intentional, and the specification does indeed apply to other device connection types.

Host software should begin and end commands with SLIP's frame delimiter **C0**, and must take into account SLIP escape sequences that deal with occurrences of C0 inside the SLIP data frame:

- If outbound data contains the byte value **C0**, software should encode it into SLIP as **DB DC**; if inbound SLIP data contains the byte sequence **DB DC**, software should decode it to **C0**.
- If outbound data contains the byte value **DB**, software should encode it into SLIP as **DB DD**; if inbound SLIP data contains the byte sequence **DB DD**, software should decode it to **DB**.

The data size for command data and card data may increase with firmware updates, so the host software should be able to adapt to this. Adapting can be as simple as ignoring any extra data bytes that are not understood or expected.

3.4.1 Device-Initiated Messages In SLIP Format

When using SLIP format, the device may send **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)** or **Notification Messages Sent from Device to Host (Extended Notifications Only)** in either normal or Run-Length Encoded (RLE) compressed format, depending on whether RLE would help compress the data or not. The host software should understand both formats. The first byte of the incoming message is a message type field, which indicates what type of data the device is sending and whether the data is RLE compressed, as follows:

- 0x00 = **Card Data Normal**, which indicates the card data payload contains uncompressed card data in USB HID vendor defined report format (see section **6 Magnetic Stripe Card Data Sent from Device to Host**).
- 0x01 = **Card Data RLE**, which indicates the card data payload contains run-length-encoded compressed card data in USB HID vendor-defined report format (see section **6 Magnetic Stripe Card Data Sent from Device to Host** and the information below about RLE decoding).
- 0x02 = **Notification Uncompressed**, which indicates the Notification Data contains an uncompressed notification message [see section **7 Notification Messages Sent from Device to Host (Extended Notifications Only)**].
- 0x03 = **Notification RLE**, which indicates the Notification Data contains a run-length-encoded compressed notification message [see section **7 Notification Messages Sent from Device to Host (Extended Notifications Only)** and the information below about RLE decoding].

The device implements RLE encoding of Notification Message data as follows:

1) Any byte that is repeated more than once consecutively is run length encoded. Bytes that are not repeated stay as-is.
2) Repeated bytes are run-length encoded by repeating the byte twice, followed by the number of times the byte was repeated in the original data.
3) The maximum length of an encoded run is 255, so runs larger than 255 bytes are encoded as multiple runs of 255 bytes each until the last run.

For example, the data `0x44 0x55 0x55 0x55 0x55 0x55 0x55 0x55 0x55 0x055 0x66 0x00 0x00` is encoded as `0x44 0x55 0x55 0x09 0x66 0x00 0x00 0x02`. A run of 260 `0x00` bytes would be encoded as `0x00 0x00 0xFF 0x00 0x00 0x05`.

3.4.1.1 Magnetic Stripe Card Data In SLIP Format (MSR Only | Keypad Entry Only)

Uncompressed **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)** is wrapped in the following block:

Table 3-4 - SLIP Format Card Data Uncompressed Wrapper

Bit	7	6	5	4	3	2	1	0
Byte 0	SLIP frame delimiter = 0xC0							
Byte 1	Message Type = 0x00 Card Data Uncompressed							
Bytes 2..3	Length of Card Data field, in big endian order							
Bytes 4..n	Card Data as defined in section **6 Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)**							
Byte n+1	SLIP frame delimiter = 0xC0							

RLE encoded **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)** is wrapped in the following block:

Table 3-5 - SLIP Format Card Data RLE Wrapper

Bit	7	6	5	4	3	2	1	0
Byte 0	SLIP frame delimiter = 0xC0							
Byte 1	Message Type = 0x01 Card Data RLE							
Bytes 2..3	Length of uncompressed Card Data field, in big endian order							
Bytes 4..n	RLE encoded Card Data as defined in section **6 Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)**							
Byte n+1	SLIP frame delimiter = 0xC0							

3.4.1.2 Notification Messages In SLIP Format (Extended Notifications Only)

Uncompressed **Notification Messages Sent from Device to Host (Extended Notifications Only)** are wrapped in the following block:

Table 3-6 - SLIP Format Notification Uncompressed Wrapper

Bit	7	6	5	4	3	2	1	0
Byte 0	SLIP frame delimiter = 0xC0							
Byte 1	Message Type = 0x02 Notification Uncompressed							
Bytes 2..3	Length of Notification Message field, in big endian order							
Bytes 4..n	Notification Message as defined in section **7.1 About Notification Messages**							
Byte n+1	SLIP frame delimiter = 0xC0							

RLE encoded **Notification Messages Sent from Device to Host (Extended Notifications Only)** are wrapped in the following block:

Table 3-7 - SLIP Format Notification RLE Wrapper

Bit	7	6	5	4	3	2	1	0
Byte 0	SLIP frame delimiter = 0xC0							
Byte 1	Message Type = 0x03 = Notification RLE							
Bytes 2..3	Length of uncompressed Notification Message field, in big endian order							
Bytes 4..n	RLE encoded Notification Message as defined in section **7.1 About Notification Messages**							
Byte n+1	SLIP frame delimiter = 0xC0							

3.4.2 Commands and Responses In SLIP Format

When the device and host are using SLIP format for commands and responses, the host software should wrap all commands in the following block:

Table 3-8 - SLIP Format Command Request Wrapper

Bit	7	6	5	4	3	2	1	0
Byte 0	SLIP frame delimiter = 0xC0							
Byte 1	Message Type = 0x05 Command Request							
Bytes 2..3	Length of Command Request Message field, in big endian order							
Bytes 4..n	Command Request Message from Table 8-1 in section **8.1 About Commands**							
Byte n+1	SLIP frame delimiter = 0xC0							

The device wraps all command responses in the following block:

Table 3-9 - SLIP Format Command Response Wrapper

Bit	7	6	5	4	3	2	1	0
Byte 0	SLIP frame delimiter = 0xC0							
Byte 1	Message Type = 0x04 Command Response Normal, which indicates the payload contains an uncompressed command response message.							
Bytes 2..3	Length of Command Response Message field, in big endian order							
Bytes 4..n	Command Response Message from **Table 8-2** in section **8.1 About Commands**							
Byte n+1	SLIP frame delimiter = 0xC0							

3.5 How to Use Tag-Length-Value (TLV) Format (TLV Only)

Some devices use a tag-length-value (TLV) format to transmit data over specific connection types, either by default or by being configured to use TLV format. It is important to note that this refers to a data transmission format which is similar to, but completely separate from, the TLV encoding of EMV data in the EMV database.

1) For information about which devices use TLV format over which connection types, see section **1.4 About Connections and Data Formats**.
2) For a detailed example of sending a command in the TLV format and interpreting the response, see **Appendix B**, section **C.3 TLV Example**.
3) For details about the data objects in the TLV data stream, see **Appendix F TLV Tag Allocation Dictionary**.

There are three kinds of messages involved with TLV communication between the host and the device:

- Unsolicited messages from the device to the host, which contain:
    - Card Swipe Data
    - Discovery Data (if configured to deliver at power up *)
    - TLV Version (if configured to deliver at power up *)
    - Timeout (when the Authentication Mode times out)
    - No MSR Transactions Remaining [when a card is swiped but no more encryption cycles remain, see **Command 0x1C - Get Remaining MSR Transactions Counter (MSR Only)**].
    - DUKPT Keys Exhausted (when a card is swiped but the DUKPT encryption scheme has used all available keys).
- Command messages sent from the host to the device:
    - Supports all the commands listed in section **8 Commands**, using the TLV data object 8402 (MagneSafe V5 commands).
    - TLV Native commands – The device supports TLV data object 8409 (Retrieve Discovery Information).
- Responses to commands, sent from the device to the host.
- Responses to MagneSafe V5 commands reported in TLV data object 8403 (MagneSafe V5 Response).
- TLV Native responses. The device supports TLV data object C306 (Discovery).

* Not currently supported

3.5.1 TLV Command Format

The host software should send requests to the device using TLV data object `C102` (Host to Device Request), containing TLV data object `8402` (MagneSafe V5 commands), which contains a command formatted as shown in **Table 8-1** in section **8.1 About Commands**.

The response from the device to the host uses TLV data object `C104` (Device to Host), containing TLV data object `8403` (MagneSafe V5 Response), which contains a command formatted as shown in **Table 8-2** in section **8.1 About Commands**.

4 Security Levels

Devices can be configured to operate at different Security Levels, which affects **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)**, the host software's ability to modify **Properties**, and the host software's ability to execute certain **Commands**. The Security Level can be increased by sending commands to the device, but can never be decreased. The sections below provide details about how each security level affects device behavior.

4.1 About Message Authentication Codes (MAC)

Commands in this manual that are tagged "MAC" are **privileged commands**. If the device is set to a Security Level higher than **Security Level 2**, the host software must calculate and append a four-byte Message Authentication Code ("MAC") to the Data field of the message, extending the length of the field by 4 bytes, to prove the sender is authorized to execute that command. If the device is set to **Security Level 2**, the device ignores the MAC field and the Device Serial Number field and the host can set them to all zeroes. If a MAC is required but not present or incorrect, the device returns `0x07`. (Fixed Key Only) If the device is configured to use fixed key encryption instead of DUKPT using **Property 0x6B - Key Management Scheme (Fixed Key Only)**, then MACing is not ever required.

In most cases, the host must calculate the MAC using the current DUKPT Key (which can be retrieved using **Command 0x09 - Get Current TDES DUKPT KSN** to get a reference to the key). In some cases, documented in the commands that are affected by it, the host must compute the MAC using the UIK installed in the device. In cases where only MagTek knows the UIK, MagTek must be involved to populate the MAC field.

The host must calculate the MAC over the whole command per ***ISO 9797-1***, MAC Algorithm 3, Padding Method 1, using the **Message Authentication, request or both ways** variant as specified in ***ANS X9.24-1:2009, Annex A***. Data supplied to the MAC algorithm should be provided in raw binary form, not converted to ASCII-hexadecimal.

Upon successfully completing a MACed command that used the DUKPT key, the device advances the DUKPT Key.

The serial number value included with MACs is always 16 bytes long. The 16th byte always contains 0x00. If the serial number is less than 15 bytes, it is left-justified and padded with binary zeroes.

4.2 Security Level 2

Security Level 2 is the least secure mode. In this mode, keys are loaded but the device does not require the host software to use them for most operations: Keys are used/needed to load new keys and to move to Security Level 3 or 4, but all other properties and commands are freely usable. The host can use **Command 0x15 - Get / Set Security Level (MAC)** to determine the device's current security level.

(SureSwipe Only, Streaming Only, KB Only, MSR Only)
In Security Level 2, if the device is using Streaming format [see section **3.3 How to Use Streaming Format (Streaming Only)**], the device sends data in the MagneSafe V5 format described in this manual or in USB KB SureSwipe format, based on the setting in **Property 0x1A - Keyboard SureSwipe Flags (SureSwipe Only, Streaming Only, KB Only, MSR Only)**. For information about USB KB SureSwipe format, see ***D99875206 TECHNICAL REFERENCE MANUAL, USB KB SURESWIPE & SWIPE READER***.

(MSR Only, HID Only)
In Security Level 2, if the device is using HID format [see section **3.1 How to Use HID Format (HID Only)**], the device sends data in the MagneSafe V5 format described in this manual or in USB HID

SureSwipe format using the SureSwipe VID/PID, based on the setting in **Property 0x38 - HID SureSwipe Flag (SureSwipe Only, HID Only, MSR Only)**. For information about USB HID SureSwipe format, see ***D99875191 TECHNICAL REFERENCE MANUAL, USB HID SURESWIPE & SWIPE READER***.

4.3 Security Level 3

At Security Level 3, many commands require security; most notably **Command 0x01 - Set Property (MAC)**. See section **4.1 About Message Authentication Codes (MAC)** for details. The host can use **Command 0x15 - Get / Set Security Level (MAC)** to determine the device's current security level.

Security Level 3 also enables encryption of data and inclusion of encrypted data where it may have been left out at a lower security level. For a list of specific data the device encrypts at this security level and how the host can decrypt it, see section **5 Encryption, Decryption, and Key Management**.

4.4 Security Level 4 (MSR Only)

When the device is at Security Level 4, the device requires the host to successfully complete an Authentication Sequence before it will transmit data from a magnetic stripe card swipe (see section **8.3.9 Command 0x10 - Activate Authenticated Mode**). Correctly executing the Authentication Sequence also causes the green LED to blink, alerting the operator that the device is being controlled by a host with knowledge of the keys—that is, an Authentic Host. The host can use **Command 0x15 - Get / Set Security Level (MAC)** to determine the device's current security level.

4.5 Command Behaviors By Security Level

Table 4-1 shows the commands that are affected by the device's security level. Commands that are not affected by the security level are not listed. The key is as follows:

- **Y** means the command can run at the specified security level.
- **N** means the command is prohibited at the specified security level.
- **C** means the customer may specify **Y** or **S** for that command when ordering.
- **S** means the command is secured [may require MACing, see section **4.1 About Message Authentication Codes (MAC)**]. (Fixed Key Only) If the device is configured to use fixed keys instead of DUKPT key management using **Property 0x6B - Key Management Scheme (Fixed Key Only)**, commands marked with **S** do not require MACing.
- * indicates **Command 0x02 - Reset Device** has special behavior. If an Authentication sequence has failed, only a correctly MACed **Command 0x02 - Reset Device (MAC)** can be used to reset the device. This is to prevent a dictionary attack on the keys and to minimize a denial of service (DoS) attack.

Table 4-1 - Command Behaviors At Each Security Level

Command	Level 2	Level 3	Level 4 (MSR Only)
Any command that is not listed in this table works the same at all Security Levels.	Y	Y	Y
Command 0x01 - Set Property (MAC)	Y	S	S
Command 0x02 - Reset Device (MAC)	Y	*	*
Command 0x04 - Set Keymap Item (MAC, KB Only)	Y	S	S
Command 0x05 - Save Custom Keymap (MAC, KB Only)	Y	S	S

Command	Level 2	Level 3	Level 4 (MSR Only)
Command 0x10 - Activate Authenticated Mode	N	Y	Y
Command 0x11 - Activation Challenge Response	N	Y	Y
Command 0x12 - Deactivate Authenticated Mode	N	Y	Y
Command 0x15 - Get / Set Security Level (MAC)	S	S	S
Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)			
Command 0x32 - Abort Transaction Validation (Transaction Validation Only)			
Command 0x33 - Get Transaction Validation Result (Transaction Validation Only)			
Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)	N	Y	Y
Extended Command 0x0302 - Cardholder Selection Result	N	Y	Y
Extended Command 0x0303 - Online Processing Result / Acquirer Response (EMV Only)	N	Y	Y
Extended Command 0x0304 - Cancel Transaction (EMV Only)	N	Y	Y
Extended Command 0x0305 - Modify Terminal Configuration (MAC)	N	S	S
Extended Command 0x0307 - Modify Application Configuration (MAC)	N	S	S
Extended Command 0x0309 - Modify Acquirer Public Key CAPK (MAC, EMV ODA Only)	N	S	S
Extended Command 0x030C - Set Date and Time (MAC)	N	S	S
Extended Command 0x030E - Commit Configuration	N	Y	Y
Extended Command 0x0310 - Modify EMV Configuration (MAC, Contact Only)	N	S	S

5 Encryption, Decryption, and Key Management

5.1 About Encryption and Decryption

Some data exchanged between the device and the host is encrypted. This includes **Encrypted Track Data**, **Encrypted MagnePrint Data**, **Encrypted Session ID**, **Encrypted CRC (Streaming Only)**, and parts of the **ARQC Messages (EMV Only)** and **Transaction Result Messages (EMV Only)**. To decrypt this data, the host must first determine what key to use, then decrypt the data.

5.2 How to Determine the Key

When the device and the host are using TDES DUKPT key management [see **Property 0x6B - Key Management Scheme (Fixed Key Only)**] and the device is encrypting data (see **Security Levels**), the host software must do the following to generate a key (the “derived key”) to use for decryption:

1) **Determine the value of the Initial Key loaded into the device**. The lookup methods the host software uses depend on the overall solution architecture, and are outside the scope of this document. However, most solutions do this in one of two ways, both of which use the Initial Key Serial Number that arrives with the encrypted data (see **Command 0x09 - Get Current TDES DUKPT KSN** for details about interpreting the KSN):
   a) Look up the value of the Base Derivation Key using the Initial KSN portion of the current KSN as an index value, then use TDES DUKPT algorithms to calculate the value of the Initial Key; or
   b) Look up the value of the Initial Key directly, using the Initial KSN portion of the current KSN as an index value.
2) **Derive the current key**. Apply TDES DUKPT algorithms to the Initial Key value and the encryption counter portion of the KSN that arrives with the encrypted data.
3) **Determine which variant of the current key the device used to encrypt**. The variants are defined in ***ANS X9.24-1:2009 Annex A***, which programmers of host software must be familiar with. Which variant the host should use depends on the type of data the host is decrypting or encrypting, and on device settings:
   a) **Encrypted CRC (Streaming Only)** data is always encrypted using the **Message Authentication, request or both ways** variant.
   b) **Encrypted MagnePrint Data** is encrypted according to the setting in **Property 0x56 - MagnePrint Data Encryption Variant (MSR Only, Configurable MP Variants Only)**, if the device supports it. Otherwise, it is encrypted according to the setting in **Property 0x54 - Card Data Encryption Variant (MSR Only, Configurable MSR Variants Only)**, if the device supports it. Otherwise, it is encrypted using the **PIN Encryption variant**.
   c) **Encrypted Track Data** and **Encrypted Session ID** is encrypted according to the setting in **Property 0x54 - Card Data Encryption Variant (MSR Only, Configurable MSR Variants Only)**, if the device supports it. Otherwise, it is encrypted using the **PIN Encryption variant**.
   d) EMV data is encrypted according to the setting in **Property 0x67 - EMV Data Encryption Variant (EMV Only)**.
4) Use the variant algorithm with the current key to calculate that variant.
5) Decrypt the data according to the steps in section **5.3 How to Decrypt Data**.

(Fixed Key Only)
As an alternative to TDES DUKPT key management, the device can also be configured to allow the host to manage keys by changing **Property 0x6B - Key Management Scheme (Fixed Key Only)** to use fixed keys. In this case, the host must load fixed keys using **Command 0x4E - Load Fixed Key (Fixed Key Only)** and keep track of which key is currently loaded. All operations that would ordinarily use DUKPT

then used fixed keys instead. The device can be set to require proof that the host knows the current key before it allows the host to load a new fixed key. The device ships with the following defaults:

- **Initial Fixed Key**: `0000000000000000` (16 zeroes)
- **Initial Fixed Key Key Serial Number (KSN)**: `0000000000` (10 zeroes)
- **Initial Fixed Key Key Check Value (KCV)**: `0x8CA64D`

5.3 How to Decrypt Data

For **Encrypted Track Data** and encrypted EMV data in **ARQC Messages (EMV Only)** and **Transaction Result Messages (EMV Only)**, the device begins by encrypting the first 8 bytes of clear text track data. The 8-byte result of this encryption is placed in an encrypted data buffer. The process continues using the DES CBC (Cipher Block Chaining) method with the encrypted 8 bytes XORed with the next 8 bytes of clear text. That result is placed in next 8 bytes of the encrypted data buffer, and the device continues until all clear text bytes have been encrypted. If the final block of clear text contains fewer than 8 bytes, the device pads the end of the block to make 8 bytes. After the final clear text block is XORed with the prior 8 bytes of encrypted data, the device encrypts it and places it in the encrypted data value. No Initial Vector is used in the process.

The host must decrypt the data in 8 byte blocks, ignoring any final unused bytes in the last block. When a value consists of more than one block, the host should use the CBC method to decrypt the data by following these steps:

1) Start decryption on the last block of 8 bytes (call it block N) using the key.
2) XOR the result of the decryption with the next-last block of 8 bytes (block N-1).
3) Repeat until reaching the first block.
4) Do not XOR the first block with anything.
5) Concatenate all blocks.
6) Determine the expected length of the decrypted data. In some cases this may be a standard field length, and in other cases the expected data length may accompany the encrypted data. When decrypting track data where no length is available, the host software can use the End Sentinel to find the actual end of the data (ignoring the padding at the end, which contains all zeroes).
7) Truncate the end of the decrypted data block to the expected data length, which discards the padding at the end.

6 Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)

The device sends card swipe data to the host even if it can not fully decode the data. How the host interprets incoming messages to find the data detailed in this section depends on the connection type (see section **2 Connection Types**) and the data format (see section **3 Data Formats**). Each subsection is tagged with the features, connection types, and data formats for which it is relevant. **Table 6-1** provides a convenient summary / index of all available values and their offsets.

Table 6-1 - List of Magnetic Stripe Data Sorted By GATT/SLIP Offset

Data	HID Usage	GATT/SLIP Offset
Track 1 Decode Status (HID Only | TLV Only | GATT Only | SLIP Only)	0x20	0
Track 2 Decode Status (HID Only | TLV Only | GATT Only | SLIP Only)	0x21	1
Track 3 Decode Status (HID Only | TLV Only | GATT Only | SLIP Only, 3-Track Only)	0x22	2
Track 1 Encrypted Data Length (HID Only | GATT Only | SLIP Only)	0x28	3
Track 2 Encrypted Data Length (HID Only | GATT Only | SLIP Only)	0x29	4
Track 3 Encrypted Data Length (HID Only | GATT Only | SLIP Only, 3-Track Only)	0x2A	5
Card Encode Type (HID Only | TLV Only | GATT Only | SLIP Only)	0x38	6
Track 1 Encrypted Data	0x30	7..118
Track 2 Encrypted Data	0x31	119..230
Track 3 Encrypted Data	0x32	231..342
Card Status (HID Only | GATT Only | SLIP Only)	0x39	343
MagnePrint Status	0x23	344..347
MagnePrint Data Length (HID Only | GATT Only | SLIP Only)	0x2B	348
Encrypted MagnePrint Data	0x33	349..476
Device Serial Number	0x40	477..492
Device Encryption Status	0x42	493..494
DUKPT Key Serial Number (KSN)	0x46	495..504
Track 1 Masked Data Length (HID Only | GATT Only | SLIP Only)	0x47	505
Track 2 Masked Data Length (HID Only | GATT Only | SLIP Only)	0x48	506

Data	HID Usage	GATT/SLIP Offset
Track 3 Masked Data Length (HID Only | GATT Only | SLIP Only, 3-Track Only)	0x49	507
Track 1 Masked Data	0x4A	508..619
Track 2 Masked Data	0x4B	620..731
Track 3 Masked Data (3-Track Only)	0x4C	732..843
Encrypted Session ID	0x50	844..851
Track 1 Absolute Data Length (HID Only | GATT Only | SLIP Only)	0x51	852
Track 2 Absolute Data Length (HID Only | GATT Only | SLIP Only)	0x52	853
Track 3 Absolute Data Length (HID Only | GATT Only | SLIP Only, 3-Track Only)	0x53	854
MagnePrint Absolute Data Length (HID Only | TLV Only | GATT Only | SLIP Only)	0x54	855
Remaining MSR Transactions	0x55	856..858
MagneSafe Version Number (HID Only | GATT Only | SLIP Only)	0x56	859..866
SHA-1 Hashed Track 2 Data (HID Only | TLV Only | GATT Only | SLIP Only, SHA-1 Only)	0x57	867…886
HID Report Version (HID Only | GATT Only | SLIP Only)	0x58	887
SHA-256 Hashed Track 2 Data (SHA-256 Only)	0x59	888..919
MagnePrint KSN (HID Only | TLV Only | GATT Only | SLIP Only)	0x5A	920..929
Battery Level (HID Only | GATT Only | SLIP Only)	0x5B	930
Clear Text CRC (Streaming Only)	N/A	N/A
Encrypted CRC (Streaming Only)	N/A	N/A
Format Code (Streaming Only)	N/A	N/A

6.1 About Track Data

After the host receives and decrypts **Encrypted Track Data** from the device, or receives clear text track data (based on device settings or state), or receives **Masked Track Data**, it may need to parse each track into individual values embedded in the tracks. The device can read multiple card formats, which vary even between different issuers and payment brands using the same underlying standards. Describing all possible formats is beyond the scope of this document, but this section describes how to parse data from tracks 1, 2, and 3 in a generic ISO/ABA compliant format as an example.

Table 6-2 shows an example of ISO/ABA track data the device sends to the host, using unmasked placeholder numbers to make it easier to see the relative positions of the values embedded in the track data. It is important to note that some cards do not include Track 3 data, and some devices do not read or transmit Track 3 data (see section **1.5 About Device Features**). (Keypad Entry Only) Manually entered data does not include Track 3.

Table 6-2 – Example Generic ISO/ABA Track Data Format

Generic ISO/ABA Track Data Format	
Track 1 Data	`%75555555555555555^CARDHOLDER NAME/^33338880004440000006?`
Track 2 Data	`;5555555555555555=33338880004440006?`
Track 3 Data	`;5555555555555555=333388800044400000006?`

The example track data in **Table 6-2** can be interpreted as follows:

- The %, ?, and ; are Sentinels / delimiters, and are taken directly from the data on the card, except when using Streaming format, where they may be overridden by **Properties** as described in section **3.3.1 Magnetic Stripe Card Data In Streaming Format (Swipe Only | Keypad Entry Only)**. (Keypad Entry Only) Manually entered data in Streaming format (and only in Streaming format) also constructs track data using those **Properties** as delimiters.
- The 7 at the beginning of Track 1 data is the card format code. For swiped credit / debit cards, this comes from the card and is generally B. (Keypad Entry Only) Manually entered data uses M.
- The string of 5s is the Account Number / License Number / PAN.
- The carets ^ are a standard ISO track 1 delimiter surrounding the Cardholder Name.
- CARDHOLDER NAME/ is the Cardholder Name. (Keypad Entry Only) Manually entered data uses string literal MANUAL ENTRY/.
- The string of 3s is the Expiration Date.
- The string of 8s is the Service Code. For swiped credit / debit cards, this comes from the card. (Keypad Entry Only) Manually entered data uses 000.
- The remaining characters (0s, 4s, and 6) are Discretionary Data. For swiped debit / credit cards this data is of varying length and content and comes from the card, and must be interpreted according to the standards established by issuers, payment brands, and so on. (Keypad Entry Only) Manually entered track data uses a MagTek standard for Discretionary Data as follows:
    - The string of 4s is the CVV2 a cardholder or operator entered on the keypad. This may be 3 or 4 characters long and is not padded, so the host software must find it by using the fixed-length padding and sentinels that surround it.
    - The strings of 0s are literals of fixed length: Track 1 has three zeroes after the Service Code, and five zeroes after the CVV2; Track 2 has three zeroes after the Service Code, and two zeroes after CVV2.

6.2 Track 1 Decode Status (HID Only | TLV Only | GATT Only | SLIP Only)

This one-byte value indicates the status of decoding Track 1. If bit 0 is OFF, no error occurred. If bit 0 is ON, the device found non-noise data that was not decodable, and the device reports the track data length is zero and does not provide valid track data to the host.

Format	Where to Find Value
HID	Usage 0x20
Streaming	N/A
TLV	Data Object 8262 Byte 1
GATT/SLIP	Offset 0

Bit Position	7	6	5	4	3	2	1	0
Value	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Error

6.3 Track 2 Decode Status (HID Only | TLV Only | GATT Only | SLIP Only)

This one-byte value indicates the status of decoding Track 2. If bit 0 is OFF, no error occurred. If bit 0 is ON, the device found non-noise data that was not decodable, and the device reports the track data length is zero and does not provide valid track data to the host.

Format	Where to Find Value
HID	Usage 0x21
Streaming	N/A
TLV	Data Object 8262 Byte 2
GATT/SLIP	Offset 1

Bit Position	7	6	5	4	3	2	1	0
Value	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Error

6.4 Track 3 Decode Status (HID Only | TLV Only | GATT Only | SLIP Only, 3-Track Only)

This one-byte value indicates the status of decoding Track 3. If bit 0 is OFF, no error occurred. If bit 0 is ON, the device found non-noise data that was not decodable, and the device reports the track data length is zero and does not provide valid track data to the host.

Format	Where to Find Value
HID	Usage 0x22
Streaming	N/A
TLV	Data Object 8262 Byte 3

Format	Where to Find Value
GATT/SLIP	Offset 2

Bit Position	7	6	5	4	3	2	1	0
Value	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Error

6.5 Card Encode Type (HID Only | TLV Only | GATT Only | SLIP Only)

This one-byte value indicates the type of encoding the device found on a swiped magnetic stripe card. **Table 6-3** defines the possible values. For details about how the device determines the card's encode type, see **Appendix E Identifying ISO/ABA and AAMVA Cards For Masking (MSR Only)**.

Format	Where to Find Value
HID	Usage 0x38
Streaming	N/A (see section **3.3.1**)
TLV	Data Object 8261
GATT/SLIP	Offset 6

Table 6-3 - Card Encode Types

Value	Encode Type	Description
0	ISO/ABA	ISO/ABA encode format. At least one track in ISO/ABA format, Track 3 not AAMVA format. See **Appendix E Identifying ISO/ABA and AAMVA Cards** for ISO/ABA description.
1	AAMVA	AAMVA encode format. Track 3 is AAMVA format, Tracks 1 and 2 are ISO/ABA if correctly decoded. See **Appendix E Identifying ISO/ABA and AAMVA Cards** for AAMVA description.
2	Reserved	Reserved.
3	Blank	The card is blank. All tracks decoded without error and without data.
4	Other	The card has a non-standard encode format. For example, ISO/ABA track 1 format on track 2.
5	Undetermined	The card encode type could not be determined because no tracks could be decoded. Combination of Error tracks and Blank Tracks, at least one Error track.
7	JIS Type 2 (JIS Support Only)	JIS Type 2 encode format. A Single track Japanese Industrial Standard (JIS) encoding type. The devuce only decodes this encode type if it is enabled using **Property 0x1B - Decode Enable (JIS Support Only, MSR** Only).

6.6 Device Encryption Status

This two-byte value contains the Device Encryption Status in big endian byte order. Byte 1 is the least significant byte; the LSB of byte 1 is status bit 0, and the LSB of byte 2 is status bit 15.

If the **Encryption Enabled** bit or **Initial DUKPT Key Injected** bit are not set, the device sends card data it would usually encrypt as clear text, and does not include a valid **DUKPT Key Serial Number**.

When the **DUKPT Keys Exhausted** bit is set, the device no longer reads cards, but continues to send **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)** to report status. The data it sends to the host in this case does not include valid **MagnePrint Status**, **Encrypted MagnePrint Data**, **Masked Track Data**, or **Encrypted Track Data**.

Format	Where to Find Value
HID	Usage 0x42
Streaming	See **Table 3-1** p. **44**. Hex-encoded binary.
TLV	Data Object 8001
GATT/SLIP	Offset 493..494

The Device Encryption Status is defined as follows:

Bit	Meaning
0	DUKPT keys exhausted (1 = Exhausted, 0 = Keys available)
1	Initial DUKPT key injected, always set to 1
2	Encryption Enabled, always set to 1
3	Authentication Required
4	Timed out waiting for cardholder to swipe card
8	No MSR Transactions Remaining [see **Command 0x1C - Get Remaining MSR Transactions Counter (MSR Only)**]
9	(Secondary DUKPT Key Only) Initial Secondary DUKPT key injected
10	(Transaction Validation Only) Transaction Validation Result Available
10	(Secondary DUKPT Key Only) DUKPT Key used to encrypt **Encrypted Track Data**, **Encrypted Session ID**, **Encrypted CRC (Streaming Only)**. 0 = Primary, 1 = Secondary
11	(Configurable MSR Variants Only) DUKPT Key Variant used to encrypt **Encrypted Track Data**. 0 = PIN Encryption. 1 = Data Encryption, request or both ways
12	(Secondary DUKPT Key Only) DUKPT Key used to encrypt **Encrypted MagnePrint Data**, 0 = Primary DUKPT Key. 1 = Secondary DUKPT Key.

Bit	Meaning
13	(Configurable MP Variants Only) DUKPT Key Variant used to encrypt **Encrypted MagnePrint Data**. 0 = PIN Encryption, 1 = Data Encryption, request or both ways
14	Unused (always set to 0)
15	Unused (always set to 0)

6.7 Encrypted Track Data

If decodable track data exists for a given track, the device returns it in the corresponding **Track x Encrypted Data** value, described in the subsections below.

When the device is transmitting data in HID, GATT, or SLIP format, the **Encrypted Data** values are always 112 bytes long, which is the maximum number of bytes that can be encoded on a card. However, the length of actual valid data in each value may be less than 112 bytes, and is stored in the corresponding **Encrypted Data Length** value. The host software should ignore data located beyond the data length reported by the device.

The device decodes the data from each track on the card and converts it to ASCII, and includes all data starting with the start sentinel and ending with the end sentinel. For additional information about configuration-specific or card-type-specific start and end sentinel behavior, see sections **9.23**, **9.24**, **9.31**, **9.32**, **9.33**, **9.34**, **9.35**, **9.36**, **9.37**, **9.39**, **9.40**, and **9.41**.

If the device is in a security level below **Security Level 3**, it sends the resulting track data in the **Track x Encrypted Data** values unencrypted. If the device is in **Security Level 3** or **Security Level 4**, it encrypts the data before sending. For information about how the device encrypts the data and how the host should decrypt it, see section **5 Encryption, Decryption, and Key Management**.

6.7.1 Track 1 Encrypted Data Length (HID Only | GATT Only | SLIP Only)

This one-byte value indicates the number of bytes in the **Track 1 Encrypted Data** value. The value is always a multiple of 8. If the value is 0, the device found no data on the track or encountered an error decoding the track.

After data is decrypted, there may be fewer bytes of decoded track data than indicated by this value. The number of bytes of decoded track data is indicated by the **Track 1 Absolute Data Length** value.

Format	Where to Find Value
HID	Usage 0x28
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 3

6.7.2 Track 2 Encrypted Data Length (HID Only | GATT Only | SLIP Only)

This one-byte value indicates the number of bytes in the **Track 2 Encrypted Data** value. The value is always a multiple of 8. If the value is 0, the device found no data on the track or encountered an error decoding the track.

After data is decrypted, there may be fewer bytes of decoded track data than indicated by this value. The number of bytes of decoded track data is indicated by the **Track 2 Absolute Data Length (HID Only | GATT Only | SLIP Only)** value.

Format	Where to Find Value
HID	Usage 0x29
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 4

6.7.3 Track 3 Encrypted Data Length (HID Only | GATT Only | SLIP Only, 3-Track Only)

This one-byte value indicates the number of bytes in the **Track 3 Encrypted Data** value. The value is always a multiple of 8. If the value is 0, the device found no data on the track or encountered an error decoding the track.

After data is decrypted, there may be fewer bytes of decoded track data than indicated by this value. The number of bytes of decoded track data is indicated by the **Track 3 Absolute Data Length** value.

Format	Where to Find Value
HID	Usage 0x2A
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 5

6.7.4 Track 1 Absolute Data Length (HID Only | GATT Only | SLIP Only)

This one-byte value indicates the number of usable bytes in the **Track 1 Encrypted Data** value after decryption. If the value is 0, the device found no data on the track or encountered an error decoding the track.

Format	Where to Find Value
HID	Usage 0x51
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 852

6.7.5 Track 2 Absolute Data Length (HID Only | GATT Only | SLIP Only)

This one-byte value indicates the number of usable bytes in the **Track 2 Encrypted Data** value after decryption. If the value is 0, the device found no data on the track or encountered an error decoding the track.

Format	Where to Find Value
HID	Usage 0x52
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 853

6.7.6 Track 3 Absolute Data Length (HID Only | GATT Only | SLIP Only, 3-Track Only)

This one-byte value indicates the number of usable bytes in the **Track 3 Encrypted Data** value after decryption. If the value is 0, the device found no data on the track or encountered an error decoding the track.

Format	Where to Find Value
HID	Usage 0x53
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 854

6.7.7 Track 1 Encrypted Data

For information about the contents of track data, see section **6.1 About Track Data**. For information about decryption, see section **5 Encryption, Decryption, and Key Management**.

Format	Where to Find Value
HID	Usage 0x30
Streaming	See **Table 3-1** p. **44**. Hex-encoded binary.
TLV	Data Object 8215
GATT/SLIP	Offset 7..118

6.7.8 Track 2 Encrypted Data

For information about the contents of track data, see section **6.1 About Track Data**. For information about decryption, see section **5 Encryption, Decryption, and Key Management**.

Format	Where to Find Value
HID	Usage 0x31
Streaming	See **Table 3-1** p. **44**. Hex-encoded binary.
TLV	Data Object 8216
GATT/SLIP	Offset 119..230

6.7.9 Track 3 Encrypted Data

On 2-track devices (see **Table 1-2 - Device Features**), this value is included in incoming data as a null value.

For information about the contents of track data, see section **6.1 About Track Data**. For information about decryption, see section **5 Encryption, Decryption, and Key Management**.

Format	Where to Find Value
HID	Usage 0x32
Streaming	See **Table 3-1** p. **44**. Hex-encoded binary.
TLV	Data Object 8217
GATT/SLIP	Offset 231...342

6.8 Card Status (HID Only | GATT Only | SLIP Only)

This is a one-byte value which indicates the card status. If bit 0 is ON, the card was swiped in the insertion direction. If bit 0 is OFF, the card was swiped in the withdrawal direction. All other bit positions are reserved.

If the device is configured to only output data when swiped in one direction, as it is by default, this value is always the same.

Format	Where to Find Value
HID	Usage 0x39
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 343

Bit Position	7	6	5	4	3	2	1	0
Value	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Card Inserted

6.9 MagnePrint Status

This four-byte value contains 32 bits of MagnePrint status information. The first byte, byte 1, is the least significant byte and its least significant bit is status bit 0. The final byte, byte 4, is the most significant byte and its most significant bit is status bit 31. For an example, see **Table 6-4** on page **69** which shows how to interpret MagnePrint Status bits for a value of `A1050000`.

If the device is set to a security level below **Security Level 3**, it uses the current value of **Property 0x15 - MagnePrint Flags** to determine the behavior of this value.

- Bit 0 = MagnePrint capable flag
- Bits 1 to 15 = Product revision and mode
- Bit 16 = Reserved
- Bit 17 = Reserved for noise measurement
- Bit 18 = Swipe too slow
- Bit 19 = Swipe too fast
- Bit 20 = Reserved
- Bit 21 = Actual card swipe direction (0 = Forward, 1 = Reverse)
- Bits 22-31 = Reserved

Format	Where to Find Value
HID	Usage 0x23
Streaming	See **Table 3-1** p. **44**. Hex-encoded binary.
TLV	Data Object 8263
GATT/SLIP	Offset 344..347

Table 6-4 - MagnePrint Status Example for Value A1050000

Byte #	Nibble #	Bit #	Hex Value	Binary Value	Bit Meaning
Byte 1 = A1	1	7	A	1	Product Revision/Mode
		6		0	Product Revision/Mode
		5		1	Product Revision/Mode
		4		0	Product Revision/Mode
	2	3	1	0	Product Revision/Mode
		2		0	Product Revision/Mode
		1		0	Product Revision/Mode
		0		1	MagnePrint capable
Byte 2 = 05	3	15	0	0	Product Revision/Mode
		14		0	Product Revision/Mode
		13		0	Product Revision/Mode
		12		0	Product Revision/Mode
	4	11	5	0	Product Revision/Mode
		10		1	Product Revision/Mode
		9		0	Product Revision/Mode
		8		1	Product Revision/Mode
Byte 3 = 00	5	23	0	0	Reserved
		22		0	Reserved
		21		0	Direction
		20		0	Reserved
	6	19	0	0	Too Fast
		18		0	Too Slow
		17		0	Reserved for noise measurement
		16		0	Reserved
Byte 4 = 00	7	31	0	0	Reserved
		30		0	Reserved
		29		0	Reserved
		28		0	Reserved
	8	27	0	0	Reserved
		26		0	Reserved
		25		0	Reserved
		24		0	Reserved

6.10 MagnePrint Data Length (HID Only | GATT Only | SLIP Only)

This one-byte value indicates the number of bytes in the **Encrypted MagnePrint Data** value, which is always a multiple of 8 bytes in length. This value is zero if there is no MagnePrint data. After the Encrypted MagnePrint data is decrypted, there may be fewer bytes of MagnePrint data than indicated by this value. The number of bytes of decrypted MagnePrint data is indicated by **MagnePrint Absolute Data Length**.

If the device is set to a security level below **Security Level 3**, it uses the current value of **Property 0x15 - MagnePrint Flags** to determine the behavior of this value.

Format	Where to Find Value
HID	Usage 0x2B
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 348

6.11 MagnePrint Absolute Data Length (HID Only | TLV Only | GATT Only | SLIP Only)

This one-byte value indicates the number of usable bytes in **Encrypted MagnePrint Data** value after decryption.

If the device is set to a security level below **Security Level 3**, it uses the current value of **Property 0x15 - MagnePrint Flags** to determine the behavior of this value.

Format	Where to Find Value
HID	Usage 0x54
Streaming	N/A
TLV	Data Object 8263
GATT/SLIP	Offset 855

6.12 Encrypted MagnePrint Data

This value contains Encrypted MagnePrint data, which when decrypted generally yields a 54-byte value. The least significant bit of the first byte of data in the decrypted value corresponds to the first bit of MagnePrint data.

If the device is set to a security level below **Security Level 3**, it uses the current value of **Property 0x15 - MagnePrint Flags** to determine the behavior of this value.

To derive a decrypted MagnePrint value to authenticate a card, the host should do the following:

1) If the device transmitted a **MagnePrint Data Length**, truncate the data to that length to strip out protocol padding and yield a decryptable data block.
2) Decrypt the data block.

3) If the device transmitted a **MagnePrint Absolute Data Length (HID Only | TLV Only | GATT Only | SLIP Only)**, truncate the data to that length to yield the MagnePrint data.

Format	Where to Find Value
HID	Usage 0x33
Streaming	See **Table 3-1** p. **44**. Hex-encoded binary.
TLV	Data Object 8218
GATT/SLIP	Offset 349..476

6.13 Device Serial Number

This 16-byte ASCII value contains the device serial number in a null-terminated string, so the maximum length of the device serial number, not including the null terminator, is 15 bytes. This device serial number can also be retrieved and set with **Property 0x03 - Device Serial Number**. This value is stored in non-volatile memory, so it persists when the device is power cycled.

Format	Where to Find Value
HID	Usage 0x40
Streaming	See **Table 3-1** p. **44**. ASCII.
TLV	Data Object 8102
GATT/SLIP	Offset 477..492

6.14 Masked Track Data

6.14.1 About Masking

If decodable track data exists for a given track, the device uses the **Track x Masked Track Data** value for that track to send a masked version of the data. The masked version includes one byte of data for each character decoded from the track, starting with the Start Sentinel and ending with the End Sentinel.

In masked track data, the device sends a specified mask character instead of the actual character read from the track. Which characters are masked depends on the **Card Encode Type (HID Only | TLV Only | GATT Only | SLIP Only)**: Only ISO/ABA (Financial Cards with ***ISO/IEC 7813*** Format code B) and AAMVA cards are selectively masked; all other card types are either sent entirely masked or entirely unmasked. More detail about masking is included in the sections below about each specific track.

There are separate masking settings for ISO/ABA format cards and AAMVA format cards (See **Property 0x07 - ISO Track Mask** and **Property 0x08 - AAMVA Track Mask** for more information). Each of these settings allows the host software to specify masking details for the Primary Account Number and Driver's License / ID Number (DL/ID#), the masking character to be used, and whether a correction should be applied to make the Mod 10 (Luhn algorithm) digit at the end of the PAN be correct.

Table 6-5 provides an example of data from tracks 1, 2, and 3 of a swiped ISO/ABA card after it has been decrypted or if the device has sent it in the clear. **Table 6-6** shows the same data as it might appear with a specific set of **Masked Track Data** rules applied.

Table 6-5 – Sample ISO/ABA Swiped Track Data, Clear Text / Decrypted

Sample ISO/ABA Swiped Track Data, Clear Text / Decrypted	
Track 1	%B6011000995500000^ TEST CARD ^15121015432112345678?
Track 2	;6011000995500000=15121015432112345678?
Track 3	;6011000995500000=15121015432112345678333333333333333333333?

Table 6-6 – Sample ISO/ABA Swiped Track Data, Masked

Sample ISO/ABA Swiped Track Data, Masked	
Track 1	%B6011000020000000^ TEST CARD ^15120000000000000000?
Track 2	;6011000020000000=15120000000000000000?
Track 3	;6011000020000000=00000000000000000000000000000000000000000?

Data Formats with fixed Data field lengths (such as USB HID format, GATT format, and SLIP format, which are fixed at 112 bytes) include a **Masked Track Data Length** value for each track, which the host should use to truncate and ignore undefined data past the end of the track data. Formats where the host can easily determine where masked track data begins and ends (such as formats with delimiters or with data length built in to the format itself) do not include explicit masked track data lengths.

6.14.2 Track 1 Masked Data Length (HID Only | GATT Only | SLIP Only)

This one-byte value indicates how many bytes of decoded card data are in the **Track 1 Masked Data** value. This value is zero if there is no data on the track or if there was an error decoding the track.

Format	Where to Find Value
HID	Usage 0x47
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 505

6.14.3 Track 2 Masked Data Length (HID Only | GATT Only | SLIP Only)

This one-byte value indicates how many bytes of decoded card data are in the **Track 2 Masked Data** value. This value is zero if there was no data on the track or if there was an error decoding the track.

Format	Where to Find Value
HID	Usage 0x48
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 506

6.14.4 Track 3 Masked Data Length (HID Only | GATT Only | SLIP Only, 3-Track Only)

This one-byte value indicates how many bytes of decoded card data are in the **Track 3 Masked Data** value. This value is zero if there was no data on the track or if there was an error decoding the track.

Format	Where to Find Value
HID	Usage 0x49
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 507

6.14.5 Track 1 Masked Data

This value contains the masked track data for track 1. All characters are transmitted. For information about the contents of track data, see section **6.1 About Track Data**. For general information about masking, see section **6.14.1 About Masking** and **Appendix E Identifying ISO/ABA and AAMVA Cards For Masking (MSR Only)**.

For an ISO/ABA card, the PAN is masked as follows:

- The number of initial characters and trailing characters specified by **Property 0x07 - ISO Track Mask** is sent unmasked. If Mod 10 correction is specified (see section **9.8 Property 0x07 - ISO Track Mask**), all but one of the intermediate characters of the PAN are set to zero; one of them is set such that the last digit of the PAN calculates an accurate Mod 10 check of the rest of the PAN as transmitted. If the Mod 10 correction is not specified, all of the intermediate characters of the PAN are set to the specified mask character.
- Cardholder Name and the Expiration Date are sent unmasked.

- On legacy devices, the Service Code is always masked.
- All Field Separators are sent unmasked.
- All other characters are set to the specified mask character.

For an AAMVA card, the specified mask character is substituted for all characters read from the card.

Format	Where to Find Value
HID	Usage 0x4A (112 bytes fixed, must be truncated)
Streaming	See **Table 3-1** p. **44**. ASCII.
TLV	Data Object 8221
GATT/SLIP	Offset 508..619

6.14.6 Track 2 Masked Data

This 112-byte value contains the masked track data for track 2. For general information about masking, see section **6.14.1 About Masking** and **Appendix E Identifying ISO/ABA and AAMVA Cards For Masking (MSR Only)**.

For an ISO/ABA card, the PAN is masked as follows:

- The number of initial characters and trailing characters specified by **Property 0x07 - ISO Track Mask** is sent unmasked. If Mod 10 correction is specified (see **Property 0x07 - ISO Track Mask**), all but one of the intermediate characters of the PAN are set to zero; one of them is set such that the last digit of the PAN calculates an accurate Mod 10 check of the rest of the PAN as transmitted. If the Mod 10 correction is not specified, all of the intermediate characters of the PAN are set to the specified mask character.
- The Expiration Date is sent unmasked.
- The Service Code is always unmasked on newer devices (Never Mask Service Code Only). On legacy devices, the Service Code is always masked.
- All Field Separators are sent unmasked.
- All other characters are set to the specified mask character.

For an AAMVA card, the DL/ID# is masked as follows:

- The specified number of initial characters are sent unmasked. The specified number of trailing characters are sent unmasked. If Mod 10 correction is specified (see **Property 0x08 - AAMVA Track Mask**), all but one of the intermediate characters of the DL/ID#PAN are set to zero; one of them is set such that last digit of the DL/ID# calculates an accurate Mod 10 check of the rest of the DL/ID# as transmitted. If the Mod 10 correction is not specified, all of the intermediate characters of the DL/ID# are set to the specified mask character.
- The Expiration Date and Birth Date are transmitted unmasked.
- All other characters are set to the specified mask character.

Format	Where to Find Value
HID	Usage 0x4B (112 bytes fixed, must be truncated)
Streaming	See **Table 3-1** p. **44**. ASCII.

Format	Where to Find Value
TLV	Data Object 8222
GATT/SLIP	Offset 620-731

6.14.7 Track 3 Masked Data (3-Track Only)

This 112-byte value contains the Masked Track Data for track 3. On 2-track devices (see **Table 1-2 - Device Features**), this value is not included in the incoming data. For general information about masking, see section **6.14.1 About Masking** and **Appendix E Identifying ISO/ABA and AAMVA Cards For Masking (MSR Only)**.

For an ISO/ABA card, the PAN is masked as follows:

- The number of initial characters and trailing characters specified by **Property 0x07 - ISO Track Mask** is sent unmasked. If Mod 10 correction is specified (see section **9.8 Property 0x07 - ISO Track Mask**), all but one of the intermediate characters of the PAN are set to zero; one of them is set such that last digit of the PAN calculates an accurate Mod 10 check of the rest of the PAN as transmitted. If the Mod 10 correction is not specified, all of the intermediate characters of the PAN are set to the specified mask character.
- All Field Separators are sent unmasked.
- All other characters are set to the specified mask character.

For an AAMVA card, the specified mask character is substituted for all characters read from the card.

Format	Where to Find Value
HID	Usage 0x4C (112 bytes fixed, must be truncated)
Streaming	See **Table 3-1** p. **44**. ASCII.
TLV	Data Object 8223
GATT/SLIP	Offset 732-843

6.15 Encrypted Session ID

This 8-byte value contains the encrypted version of the current Session ID. Its primary purpose is to prevent replays. After a card is read, this property is encrypted, along with the card data, and supplied as part of the transaction message. The clear text version is never transmitted. To avoid replay, the host software should set the Session ID property before a transaction, and verify that the Encrypted Session ID returned with card data decrypts to the original value it set.

Format	Where to Find Value
HID	Usage 0x50
Streaming	See **Table 3-1** p. **44**. Hex-encoded binary.
TLV	Data Object 8309
GATT/SLIP	Offset 844..851

6.16 DUKPT Key Serial Number (KSN)

This 80-bit value contains the TDES DUKPT **Key Serial Number** (KSN) associated with encrypted values included in the same message. For details about how to interpret this value, see section **8.3.7 Command 0x09 - Get Current TDES DUKPT KSN**. If no keys are loaded, all bytes have the value `0x00`.

Format	Where to Find Value
HID	Usage 0x46
Streaming	See **Table 3-1** p. **44**. Hex-encoded binary.
TLV	Data Object 8301
GATT/SLIP	Offset 495..504

6.17 Remaining MSR Transactions

This 3-byte value contains the number of MSR transactions remaining at the end of the current transaction. The value is also sometimes referred to as the transaction threshold. See **Command 0x1C - Get Remaining MSR Transactions Counter (MSR Only)** and **Property 0x30 - Send Remaining MSR Transactions Counter (Streaming Only, MSR Only)** for more information.

Format	Where to Find Value
HID	Usage 0x55
Streaming	See **Table 3-1** p. **44**. Hex-encoded binary.
TLV	Data Object 810A
GATT/SLIP	Offset 856..858

6.18 MagneSafe Version Number (HID Only | GATT Only | SLIP Only)

This eight-byte value contains the MagneSafe Version Number with at least one terminating `0x00` byte to make string manipulation convenient. See **Property 0x04 - MagneSafe Version Number** for more information.

Format	Where to Find Value
HID	Usage 0x56
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 859..866

6.19 SHA-1 Hashed Track 2 Data (HID Only | TLV Only | GATT Only | SLIP Only, SHA-1 Only)

If the device supports SHA-1 (see **Table 1-2 - Device Features**), this 20-byte value contains a SHA-1 hash of either the PAN from track 2 or all the track 2 data, depending on the device's configuration stored in **Property 0x57 - SHA Hash Configuration (HID Only | TLV Only, Configurable SHA Only, MSR Only)**. If the device does not support SHA-1, this value is absent or contains padding.

Format	Where to Find Value
HID	Usage 0x57
Streaming	N/A
TLV	Data Object 8308
GATT/SLIP	Offset 867..886

6.20 HID Report Version (HID Only | GATT Only | SLIP Only)

This one-byte value identifies which variation of sets of values the device sends the host for **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)**.

If the data does not contain this value, the host should implicitly assume it is equal to `0x01`. If the report does contain this value, it indicates the following:

HID Report Version	Changes
Empty	Original magnetic stripe card data contents
0x02	Added **HID Report Version (HID Only | GATT Only | SLIP Only)** Added **SHA-256 Hashed Track 2 Data**
0x03	Added **Battery Level (HID Only | GATT Only | SLIP Only)**

Format	Where to Find Value
HID	Usage 0x58
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 887

6.21 SHA-256 Hashed Track 2 Data (SHA-256 Only)

This 32-byte value contains a SHA-256 hash of either the PAN from track 2 or all the track 2 data (depending on the device's configuration). The data can be configured using **Property 0x57 - SHA Hash Configuration (HID Only | TLV Only, Configurable SHA Only, MSR Only)**.

If the device does not support SHA-256 hashed track 2 data (see section **1.5 About Device Features**), it may still transmit this value as all zeroes.

Format	Where to Find Value
HID	Usage 0x59
Streaming	N/A
TLV	Data Object 8308

Format	Where to Find Value
GATT/SLIP	Offset 888..919

6.22 MagnePrint KSN (HID Only | TLV Only | GATT Only | SLIP Only)

This 80-bit value contains the TDES DUKPT Key Serial Number associated with encrypted MagnePrint values included in the same message. The rightmost 21 bits are the current value of the encryption counter. The leftmost 59 bits are a combination of the Key Set ID (KSID) that identifies the Base Derivation Key injected into the device during manufacture, and the device's serial number; how those two values are combined into the 59 bits is defined by a convention the customer decides when architecting the solution, with support from MagTek. If no keys are loaded, all bytes have the value `0x00`.

Format	Where to Find Value
HID	Usage 0x5A
Streaming	N/A
TLV	Data Object 8305
GATT/SLIP	Offset 920..929

6.23 Clear Text CRC (Streaming Only)

This two-byte value contains a clear text version of a Cyclical Redundancy Check (CRC-16 CCITT, polynomial 0x1021), with the least significant byte sent first. It provides a CRC of all characters sent prior to this CRC. The CRC is converted to four characters of ASCII before being sent. The host software may calculate a CRC from the data received prior to this CRC and compare it to the CRC received. If they are the same, the host software can have high confidence that all the data was received correctly. **Property 0x19 - CRC Flags (Streaming Only, MSR Only)** controls whether this value is sent.

Format	Where to Find Value
HID	N/A
Streaming	See **Table 3-1** p. **44**. Hex-encoded binary.
TLV	N/A
GATT/SLIP	N/A

6.24 Encrypted CRC (Streaming Only)

This 8-byte value contains an encrypted version of a Cyclical Redundancy Check (CRC). It provides a CRC of all characters sent prior to this CRC. The CRC is converted to 16 characters of ASCII before being sent. After the receiver decrypts the message, the CRC is contained in the first 2 bytes of the message; the remaining bytes are unused. The host software may calculate a CRC from the data received prior to this CRC and compare it to the CRC received. If they are the same, the host software can have high confidence that all the data was received correctly. **Property 0x19 - CRC Flags (Streaming Only, MSR Only)** controls whether this value is sent.

Format	Where to Find Value
HID	N/A
Streaming	See **Table 3-1** p. **44**. Hex-encoded binary.
TLV	N/A
GATT/SLIP	N/A

6.25 Format Code (Streaming Only)

This four-character ASCII value contains the Format Code [stored in **Property 0x2C - Format Code (Streaming Only, MSR Only)**], which provides information for the host software to locate values within the incoming message:

Format	Where to Find Value
HID	N/A
Streaming	See **Table 3-1** p. **44**. ASCII.
TLV	N/A
GATT/SLIP	N/A

6.26 Battery Level (HID Only | GATT Only | SLIP Only)

This one-byte value contains the battery level of the device between 0% and 100%. `0x00` represents the lowest safe operating voltage; `0x64` means the battery is at full voltage. When the device is powered by USB, it always returns 100%. This field should be ignored for devices that do not contain a battery.

Format	Where to Find Value
HID	Usage 0x5B
Streaming	N/A
TLV	N/A
GATT/SLIP	Offset 930

7 Notification Messages Sent from Device to Host (Extended Notifications Only)

7.1 About Notification Messages

This section provides detail about unsolicited generic notification messages the device sends to the host, excluding magnetic stripe card data documented separately in section **6 Magnetic Stripe Card Data Sent from Device to Host**. Each subsection is tagged with the features, connection types, and data formats for which it is relevant.

Notification messages may be split into multiple packets, each containing a portion of the complete notification message. This allows notification messages to exceed the maximum packet sizes of the connection type and data format. After the host receives a complete notification message, it will have a notification identifier, a complete data length, and a complete notification message data field.

How the host interprets incoming packets to find the data detailed in this section depends on the connection type (see section **2 Connection Types**) and the data format (see section **3 Data Formats**). All packets arrive at the host in the format-dependent structure shown in **Table 7-1**. Each incoming packet can be interpreted using **Table 7-2**. The **notification message** can be interpreted by first assembling all packets pertaining to the notification message, then looking up the corresponding **Notification Identifier** in the sections that follow.

Table 7-1 - How Notification Message Packets Arrive

Format	Where to Find Value
HID	Report identifier 2, Usage identifier 0x20
Streaming	N/A
TLV	N/A
GATT/SLIP	Similar to card data. For GATT, see section **2.2.4 How to Receive Data On the Bluetooth LE Connection** and section **3.2 How to Use GATT Format (GATT Only)**. For SLIP, see section **3.4 How to Use SLIP Format (SLIP Only)**.

Table 7-2 - Structure of Packets That Form a Notification Message

Offset	Field Name	Description
0..1	Partial Data Length	The length of the **Data** field contained in the current message. This field is in big endian format. If this value is not equal to the **Complete Data Length**, the device is sending the notification using multiple packets.
2..3	Data Offset	The offset position in bytes within the entire assembled notification where the first byte of the current packet's **Data** field is located. This field is in big endian format. The first byte of the entire notification's Data is at offset zero.
4..5	Notification Identifier	The type of notification being sent. This field is in big endian format. The value corresponds to the notification identifier numbers in the headings of the subsections of section **7 Notification Messages Sent from Device to Host (Extended Notifications Only)**. In many cases, two-byte notification identifiers are assigned such that the high byte indicates a group of related commands, and the low byte specifies a command within that group.

Offset	Field Name	Description
6..7	Complete Data Length	The total length of data for the entire notification message, summing all **Partial Data Lengths** for multiple packets. This field is in big endian format. If this value is not equal to the **Partial Data Length** of the current packet, the device is sending the data using multiple packets.
8..n	Data	May contain part or all of the notification data. The size of this field is contained in the **Partial Data Length** field.

7.2 Notification Group 0x03 - EMV L2 (EMV Only)

Notification Group 0x03 is reserved for EMV L2 notifications that support **Command Group 0x03 - EMV L2 (EMV Only, Extended Commands Only)**. For more information about the general flow of EMV transactions, see section **8.4 Command Group 0x03 - EMV L2 (EMV Only, Extended Commands Only)**.

7.2.1 Notification 0x0300 - Transaction Status / Progress Information

The device sends the host this notification to report progress during an EMV transaction the host has initiated using **Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)**. The granularity of notifications is designed to give specific information about transaction steps that involve interaction with either the cardholder or the host. More information about when the device sends this notification to the host can be found in the documentation for that command.

Some devices also send this notification outside the context of an EMV transaction to more generally notify the host that a card has been removed.

The behavior of this notification is partly driven by the settings in **Property 0x6D - EMV Contact Notification Configuration (Contact Only)**.

Notification Data

Offset	Field Name	Value
0	Event	Indicates the event that triggered this notification: 0x00 = No events since start of transaction 0x01 = Card Inserted (Contact Only) 0x02 = Payment Method Communication Error / Data Error 0x03 = Transaction Progress Change 0x04 = Waiting for Cardholder Response 0x05 = Timed Out 0x06 = End of Transaction 0x07 = Host Cancelled Transaction 0x08 = Card Removed (Contact Only) (Contactless Only) 0x09 = Contactless Token Detected, Powering Up Card (EMV MSR Flow Only) 0x0A = MSR Swipe Detected
1	Current Operation Time remaining	Indicates the remaining time available, in seconds, for the indicated operation to complete. The host specifies this timeout when calling **Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)**.

2	Current Transaction Progress Indicator	This one-byte field indicates the current processing stage for the transaction: 0x00 = No Transaction In Progress 0x01 = Waiting for Cardholder to Present Payment 0x02 = Powering Up Card / Reading Magnetic Stripe 0x03 = Selecting the Application 0x04 = Waiting for Cardholder Language Selection (Contact Only) 0x05 = Waiting for Cardholder Application Selection 0x06 = Initiating Application 0x07 = Reading Application Data 0x08 = Offline Data Authentication 0x09 = Process Restrictions 0x0A = Cardholder Verification 0x0B = Terminal Risk Management 0x0C = Terminal Action Analysis 0x0D = Generating First Application Cryptogram 0x0E = Card Action Analysis 0x0F = Online Processing 0x10 = Waiting for Online Processing Response 0x11 = Transaction Complete 0x12 = Transaction Error 0x13 = Transaction Approved 0x14 = Transaction Declined 0x15 = Transaction Cancelled by MSR Swipe (MSR Only) 0x16 = EMV Error - Conditions Not Satisfied (Contact Only) 0x17 = EMV Error - Card Blocked (Contact Only) 0x18 = Contact Application Selection Failed (Contact Only) 0x19 = EMV Error - Card Not Accepted (Contact Only) 0x1A = Empty Candidate List 0x1B = Application Blocked 0x91 = Host Canceled EMV Transaction Before Card Was Presented (Contactless Only) 0x1C = Start Emulating Contactless LED. When using devices that do not have a four integrated LEDs, the host can use this event to begin showing an emulated LED sequence, implementing EMV contactless user interface guidelines using the host's display or external LEDs. 0x28 = Contactless Application Selection Failed 0x29 = Contactless Remove Card 0x2A = Collision Detected 0x2B = Refer to Mobile. The cardholder's mobile device is prompting for additional cardholder interaction. 0x2C = Contactless Transaction Complete 0x2D = Request Switch to ICC/MSR. The EMV Kernel has determined it can not continue with contactless payment method (Contact Only | MSR Only) 0x2E = Wrong Card Type (MSD or EMV) 0x2F = No Application Interchange Profile (Tag 82) Received (EMV MSR Flow Only)

Offset	Field Name	Value
		0x31 = Magnetic Stripe Decoding Error. The device found no decodable non-noise data on a swiped or inserted card. 0x3C = Magnetic Stripe Card Decoded During Technical Fallback. Device reverted to MSR after detecting a chip card, powering up the card, and not receiving an Answer to Reset from the card. 0x3D = Magnetic Stripe Card Decoded During MSR Fallback. Device reverted to MSR after detecting and powering up a chip card, receiving an Answer to Reset, and encountering fatal errors during communication. 0x3E = Magnetic Stripe Card Decoded During a Non-Fallback MSR Read.
3..4	Final Status	TBD

7.2.2 Notification 0x0301 - Display Message Request

The device sends this notification to request that the host display a message for the cardholder. The host should display the message.

Notification Data

Offset	Field Name	Value
0	Message	This is an array of bytes that should be displayed by the host on its display exactly as received. If the message is too long to fit on a single line it may be split to multiple lines if the host wishes. Messages are limited to 1024 bytes. If the message is zero length, this is a request for the host to clear the display.

7.2.3 Notification 0x0302 - Cardholder Selection Request (EMV Only)

This notification is used to inform the host that a cardholder selection is needed before the device can continue processing the current transaction. The host should prompt the cardholder to select an item from the menu, then send **Extended Command 0x0302 - Cardholder Selection Result** to inform the device that the transaction can proceed with the selected result.

Offset	Field Name	Value
0	Selection Type	This field specifies what kind of selection request this is: 0x00 = Application Selection 0x01 = Language Selection 0x10 = Enhanced Application Selection
1	Timeout	Specifies the maximum time, in seconds, allowed to complete the selection process. If this time is exceeded, the host should send **Extended Command 0x0302 - Cardholder Selection Result** with the Selection Status field set to `0x02` (Cardholder Selection Request aborted, timeout) after which the transaction is aborted and an appropriate Transaction Status is available. Value 0 (Cardholder Selection Request completed) is not allowed in this case.
2	Menu Items	This field is variable length and is a collection of null-terminated strings (maximum 17 strings). The maximum length of each string is 50 characters, not including a Line Feed (`0x0A`) character that may be in the string. The last string may not have the Line Feed character. The first string is a title and should not be considered for selection. It is expected that the host displays the menu items to the cardholder, then, after the cardholder makes a selection, call **Extended Command 0x0302 - Cardholder Selection Result** to return the number of the item the cardholder selected, which should be between 1 and the number of menu selection items being displayed. The first item, 0, is the title only. For Application Selection, host expects to receive a list of available applications. <ASCII AID Value1>|<NULL> <ASCII AID Value2>|<NULL> <ASCII AID Value3>|<NULL>..<NULL> For Enhanced Application Selection, host expects to receive a list of AID in ASCII, Priority Indicator, Issuer Country Code and Issuer Identification Number. <ASCII AID Value1> | <TLV>|<TLV>|..<NULL> <ASCII AID Value2> | <TLV>|<TLV>|..<NULL> <ASCII AID Value3> | <TLV>|<TLV>|..<NULL>

7.2.4 Notification 0x0303 - ARQC Message

The device uses this notification to send ARQC data for the host to process. After the host processes the ARQC data, it should send **Extended Command 0x0303 - Online Processing Result / Acquirer Response** to inform the device it can proceed with the transaction.

Table 7-3 - Notification Data, ARQC Message

Offset	Field Name	Value
0	Message Length	Two byte binary, most significant byte first. This gives the total length of the ARQC message that follows, excluding padding and CBC-MAC.
2	ARQC Message	The host is expected to use this data to process a request.

7.2.5 Notification 0x0304 - Transaction Result Message

The device sends this notification to provide the host with final information from the transaction. It usually includes data and an indication of whether a signature is required.

Table 7-4 - Notification Data, Transaction Result Message

Offset	Field Name	Value
0	Signature Required	This field indicates whether a cardholder signature is required to complete the transaction: 0x00 = No signature required 0x01 = Signature required If a signature is required, the host should acquire the signature from the cardholder as part of the transaction data.
1	Data Length	Two byte binary, most significant byte first. This gives the total length of the Data message that follows, excluding padding and CBC-MAC.
3	Data	It is expected that the host will save this data as a record of the transaction.

7.2.6 Notification 0x0305 - PIN Required (External PIN Accessory Support Only)

The device sends this notification when running **Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)** if its communication with card indicates the transaction requires the cardholder to enter a PIN. In response, the host should coordinate PIN entry from the cardholder using an external PIN entry accessory, and use that PIN data in its communication with the payment processor. The host should not share the PIN information with the device.

This notification does not apply to transactions where the cardholder has swiped a magnetic stripe card. In this case, the card verification method (CVM) requirements must be determined and enforced by the host software.

Notification Data

Offset	Field Name	Value
0	Reserved	Reserved for future use
1	Application Name	Contains the application name (often the payment brand name) reported by the card, terminated with a line feed (0x0A). The value is based on the contents of tag 9F12 (Application Preferred Name) if available, or tag 50 (Application Label) if 9F12 is not available.

7.3 Notification Group 0x04 - Auxiliary UART (Auxiliary Ports Only)

Notification Group 0x04 is reserved for Auxiliary UART port notifications that support **Command Group 0x04 - Auxiliary UART (Auxiliary Ports Only, Extended Commands Only)**.

7.3.1 Notification 0x0400 - Auxiliary UART Received Data

The device sends this notification to pass data to the host that it has received from an external UART device via the auxiliary UART port. For information about the auxiliary UART port, see section **8.4.21 Command Group 0x04 - Auxiliary UART (Auxiliary Ports Only, Extended Commands Only)**.

Notification Data

Offset	Field Name	Value
0	Port Identifier	The identifier of the port. Always set to zero.
1..n	Received Data	The data received. A message may be received over multiple notification messages depending on its length and time between each byte sent.

7.4 Notification Group 0x05 - Auxiliary SPI (Auxiliary Ports Only)

Notification Group 0x05 is reserved for Auxiliary SPI port notifications that support **Command Group 0x05 - Auxiliary SPI (Auxiliary Ports Only, Extended Commands Only)**.

7.4.1 Notification 0x0500 - Auxiliary SPI Data Change

The device sends this notification to inform the host that an external SPI device has changed the data on the auxiliary SPI port. For information about the auxiliary SPI port, see section **8.6 Command Group 0x05 - Auxiliary SPI (Auxiliary Ports Only, Extended Commands Only)**.

Notification Data

Offset	Field Name	Value
0	Port Identifier	The identifier of the port. Always set to zero.
1	Data	The data byte. The data byte indicates what data changed. See the Data Field table below for information about interpreting the value of this field.

Data Field (Bit 0 is the least significant bit)

Bit	Field Name	Value
0	DAV Low	This bit is set high when the DAV (Data available) input signal from the connected SPI device transitions from high to low. **Property 0x6A - Auxiliary SPI Configuration (Auxiliary Ports Only)**, bit DAV Notify Low, can be used by the host to enable/disable this notification.
1	DAV High	This bit is set high when the DAV (Data available) input signal from the connected SPI device transitions from low to high. **Property 0x6A - Auxiliary SPI Configuration (Auxiliary Ports Only)**, bit DAV Notify High, can be used by the host to enable/disable this notification.
2..7	Reserved	These bits should be ignored.

8 Commands

This section describes the commands available on the device. Each command's section heading indicates the **Connection Types**, **Data Formats**, and device features (see section **1.5 About Device Features**) that are relevant to it.

8.1 About Commands

Regardless of connection type and data format, all MagneSafe V5 devices use common structures to receive command request messages from the host and to send command response messages back. For information about connection-specific wrappers for these commands, see section **2 Connection Types**.

Table 8-1 - Command Request Message (Host Sends to Device to Initiate a Command)

Offset	Field Name
0	Command Number
1	Command Request Data Length
2..n Maximum / fixed length depends on device and connection type.	Command Request Data

Command Number is a one byte value that contains the requested command number. Section **8 Commands** lists all available commands.

Command Request Data Length is a one byte value that contains the length of the **Command Request Data** field.

Command Request Data contains command data as defined in the documentation for the selected command in section **8 Commands**.

Table 8-2 - Command Response Message (Host Sends to Device to Retrieve Data or Responses)

Offset	Field Name
0	Result Code
1	Command Response Data Length
2..n	Command Response Data

Result Code is a one-byte value the device sends to indicate success or the failure mode of the command. Section **8.2 About Result Codes** provides more detail.

Command Response Data Length is a one byte value that contains the length of the **Command Response Data** field.

Command Response Data contains response data as defined in the documentation for the selected command in section **8 Commands**.

8.2 About Result Codes

There are two types of **Result Code** values the device can return in its response: **Generic** result codes (listed in **Table 8-3**), which have the same meaning for all commands, and **command-specific** result codes, which can have different meanings for different commands, and are listed with every command in this section. Generic result codes always have the most significant bit set to zero, and command-specific result codes always have the most significant bit set to one.

Table 8-3 - Generic Result Codes

Value (Hex)	Result Code	Description
0x00	Success	The command completed successfully.
0x01	Failure	The command failed.
0x02	Bad Parameter	The command failed due to a bad parameter or command syntax error.
0x03	Redundant	The command is redundant.
0x04	Bad Cryptography	A bad cryptography operation occurred.
0x05	Delayed	The request is refused because the device is delaying requests as a defense against brute-force hacking.
0x06	No Keys	No keys are loaded.
0x07	Invalid Operation	Depends on the context of the command.
0x08	Response not available	The response is not available.
0x09	Not enough power	The battery is too low to operate reliably.
0x0A	Extended response first packet (Extended Commands Only)	The device is returning the first (and possibly only) packet of an Extended Response.
0x0B	Extended command pending (Extended Commands Only)	An extended command is pending and the device is waiting for more data.
0x0C	Extended command notification (Extended Commands Only)	Deprecated
0x0D	Not implemented	The command is not implemented.
0x0E	Unarmed tamper, device not ready (Tamper Only)	The tamper device is not ready to be armed.
0x0F	Unarmed tamper, bad signature (Tamper Only)	The tamper is not armed because of a bad signature.
0x55	Failure communicating with Embedded V5 IntelliHead (Embedded V5 Head Only)	The device's main microcontroller is unable to communicate with the device's embedded MagneSafe V5 IntelliHead. This indicates a hardware failure. Return the device to the manufacturer for service.

8.3 General Commands

8.3.1 Command 0x00 - Get Property

This command gets a property from the device. For details about properties, see section **9 Properties**.

Most properties have a firmware default value that may be changed during manufacturing or the order fulfillment process to support different customer needs.

Table 8-4 - Request Data for Command 0x00 - Get Property

Data Offset	Value
0	Property ID

Table 8-5 - Response Data for Command 0x00 - Get Property

Data Offset	Value
0..n	Property Value

Property ID is a one-byte value that identifies the property. A full list of properties can be found in section **9 Properties**.

Property Value consists of the multiple-byte value of the property. The number of bytes in this value depends on the type of property and the length of the property. **Table 8-6** describes the available property types.

Table 8-6 - Property Types

Property Type	Description
Byte	This is a one-byte value. The range of valid values depends on the property.
String	This is a null-terminated ASCII string. Its length can be zero to a maximum length that depends on the property. The length of the string does not include the terminating NULL character.

The result codes for the **Get Property** command can be any of the generic result codes listed in **Table 8-3** on page **93**.

8.3.2 Command 0x01 - Set Property (MAC)

This command sets a property in the device. For security purposes, this command is privileged. When the Security Level is set to higher than 2 (see section **4 Security Levels**), this command must be MACed to be accepted [see section **4.1 About Message Authentication Codes (MAC)**]. The command is logically paired with **Command 0x00 - Get Property**. For details about properties, see section **9 Properties**.

Some properties require the device to be reset using **Command 0x02 - Reset Device (MAC)** or power cycled to take effect. In those cases, the documentation for the property indicates what is required.

Table 8-7 - Request Data for Command 0x01 - Set Property (MAC)

Data Offset	Value
0	Property ID
1..n	Property Value

Response Data: None

The result codes for the **Set Property** command can be any of the generic result codes listed in **Table 8-3** on page **93**. If the **Set Property** command gets a result code of `0x07`, it means the required MAC was absent or incorrect.

Property ID is a one-byte value that identifies the property. A full list of properties can be found in section **9 Properties**.

Property Value consists of multiple bytes containing the value of the property. The number of bytes in this value depends on the property. **Table 8-4** describes the available property types.

Table 8-8 - Response Data for Command 0x01 - Set Property (MAC)

Property Type	Description
Byte	This is a one-byte value. The range of valid values depends on the property.
String	This is a multiple-byte ASCII string. Its length can be zero to a maximum length that depends on the property. The data length listed in the tables for each property does not include the terminating NULL character.

8.3.3 Command 0x02 - Reset Device (MAC)

This command is used to reset the device, and can be used to make property changes take effect without power cycling the device.

(USB Only)
When resetting a device that is using the USB connection, the device automatically does a USB Detach followed by an Attach. After the host sends this command to the device, it should close the USB port, wait a few seconds for the operating system to handle the device detach followed by the attach, then re-open the USB port before trying to communicate further with the device.

(PM3 | PM6 Only, Bluetooth LE Only)
When resetting a device that is using a Bluetooth LE connection, the device disconnects from Bluetooth LE. After the reset is complete, the device will be in airplane mode, and will not advertise over Bluetooth LE until either USB power is connected or the cardholder / operator presses and releases the button.

(MSR Only) If the device is in the midst of an Authentication Sequence initiated by **Command 0x10 - Activate Authenticated Mode (MSR Only)**, the device does not honor the Reset Device command until after the Authentication Sequence has successfully completed, or a cardholder swipes a card, or the device is power cycled. If the Authentication Sequence fails, the device initiates anti-hack mode and will require that the host MAC the Reset Device command (see section **4 Security Levels**). This prevents a dictionary attack on the keys and reduces the potential impact of denial of service attacks.

In rare instances, devices may optionally be configured at the manufacturer to require a MAC for every Reset Device command call, not just when anti-hack behavior is active.

Request Data Field: None

Response Data Field: None

Result codes:
0x00 = Success
0x07 = Incorrect MAC, or authentication sequence is pending

Example Request (Hex)

Cmd Num	Data Len	Data
02	00	

Example Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.4 Command 0x03 - Get Keymap Item (KB Only)

This command is used to get a key map item from the active key map determined by **Property 0x16 - Active Keymap (KB Only, MSR Only)**. Developers of host software can use this command to see which keystrokes and key modifiers the device uses to transmit a given ASCII character, and if necessary can use **Command 0x04 - Set Keymap Item (MAC, KB Only)** to modify that behavior. For a full description of how the key map works, see section **2.1.4 How to Use the USB Connection in Keyboard Emulation Mode (KB Only)** or section **2.2.5 How to Use the Bluetooth LE Connection In Keyboard Emulation Mode**. Supporting information specifically about keymaps is in **Appendix D Keyboard Usage ID Definitions (KB Only)**.

Table 8-9 - Request Data for Command 0x03 - Get Keymap Item (KB Only)

Offset	Field Name	Description
0	ASCII value	Value of the ASCII character to be retrieved from the key map. This can be any value between 0 and 127 (0x7F). For example, to retrieve the key map item for ASCII character ‘?’ (card data end sentinel), use the ASCII value of ‘?’ which is 63 (0x3F).

Table 8-10 - Response Data for Command 0x03 - Get Keymap Item (KB Only)

Offset	Field Name	Description
0	Key Usage ID	The value of the USB key usage ID that is mapped to the given ASCII value. For example, for the United States keyboard map, usage ID 56 (0x38) (keyboard / and ?) is mapped to ASCII character ‘?’.
1	Key Modifier Byte	The value of the USB key modifier byte that is mapped to the given ASCII value. For example, for the United States keyboard map, modifier byte 0x02 (left shift key) is mapped to ASCII character ‘?’.

Result codes:
0x00 = Success

Example Request (Hex)

Cmd Num	Data Len	Data
03	01	3F

Example Response (Hex)

Result Code	Data Len	Data
00	02	38 02

8.3.5 Command 0x04 - Set Keymap Item (MAC, KB Only)

This command is used to set a key map item in the active key map determined by **Property 0x16 - Active Keymap (KB Only, MSR** Only). . The command is logically paired with **Command 0x03 - Get Keymap Item (KB Only)**. For a full description of how the key map works, see section **2.1.4 How to Use the USB Connection in Keyboard Emulation Mode (KB Only)**. Supporting information specifically about keymaps is in **Appendix D Keyboard Usage ID Definitions (KB Only)**.

After host software modifies a key map item, the changes take effect immediately. However, the changes will be lost if the device is reset or power cycled. To make the changes permanent, the host software must issue **Command 0x05 - Save Custom Keymap (MAC, KB Only)**. To use the new custom key map after a reset or power cycle, the host must set **Property 0x16 - Active Keymap (KB Only, MSR Only)** to **Custom**.

Table 8-11 - Request Data for Command 0x04 - Set Keymap Item (MAC, KB Only)

Offset	Field Name	Description
0	ASCII value	Value of the ASCII character to be set in the key map. This can be any value between 0 and 127 (0x7F). For example, to set the key map item for ASCII character '?' (card data end sentinel) use the ASCII value of '?' which is 63 (0x3F).
1	Key Usage ID	The value of the USB key usage ID that is to be mapped to the given ASCII value. For example, for the United States keyboard map, usage ID 56 (0x38) (keyboard / and ?) is mapped to ASCII character '?'. To change this to the ASCII character '>' use usage ID 55 (0x37) (keyboard . and >).
2	Key Modifier Byte	The value of the USB key modifier byte that is to be mapped to the given ASCII value. For example, for the United States keyboard map, modifier byte 0x02 (left shift key) is mapped to ASCII character '?'. To change this to the ASCII character '>' use modifier byte 0x02 (left shift key).

Response Data: None

Result codes:
0x00 = Success
0x07 = Incorrect MAC - Command not authorized

The following example maps the card ASCII data end sentinel character '?' to the '>' keyboard key.

Example Set Keymap Item Request (Hex)

Cmd Num	Data Len	Data
04	03	3F 37 02

Example Set Keymap Item Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.6 Command 0x05 - Save Custom Keymap (MAC, KB Only)

This command must be issued to save the active key map, determined by **Property 0x16 - Active Keymap (KB Only, MSR Only)**, as the custom key map in non-volatile memory. See section **8.3.5 Command 0x04 - Set Keymap Item (MAC, KB Only)** for details.

Request Data: None

Response Data: None

Result codes:
0x00 = Success
0x07 = Incorrect MAC - Command not authorized

Example Save Custom Keymap Request (Hex)

Cmd Num	Data Len	Data
05	00	

Example Save Custom Keymap Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.7 Command 0x09 - Get Current TDES DUKPT KSN

The host uses this command to get the current Triple Data Encryption Standard (TDES) DUKPT Key Serial Number (KSN) on demand.

This 80-bit value contains the TDES DUKPT **Key Serial Number** (KSN) associated with encrypted values included in the same message. The rightmost 21 bits are the current value of the encryption counter. The leftmost 59 bits are the device's **Initial KSN**, which is a combination of the **Key Set ID** that identifies the Base Derivation Key (BDK) injected into the device during manufacture, and the device's serial number (DSN); how those two values are combined into the 59 bit Initial KSN is defined by a convention the customer defines when architecting the solution, with support from MagTek. For example, one common scheme is to concatenate a 7 hex digit (28 bit) Key Set ID, a 7 hex digit (28 bit) Device Serial Number, and 3 padding zero bits. In these cases, the key can be referenced by an 8-digit MagTek part number ("key ID") consisting of the 7 hex digit Key Set ID plus a trailing "0."

Request Data: None

Table 8-12 - Response Data for Command 0x09 - Get Current TDES DUKPT KSN

Offset	Field Name	Description
0	Current Key Serial Number	80-bit TDES DUKPT KSN

Result codes:
0x00 = Success
0x02 = Bad Parameter - The Data field in the request is not the correct length. The request command contains no data, so the Data Length must be 0.

Example Request (Hex)

Cmd Num	Data Len	Data
09	00	None

Example Response (Hex)

Result Code	Data Len	Data
00	0A	FFFF 9876 5432 10E0 0001

8.3.8 Command 0x0A - Set Session ID (MSR Only)

This command is used to set the current Session ID, which the device transmits to the host in the **Encrypted Session ID**. The new Session ID stays in effect until one of the following occurs:

- The host sends the device another Set Session ID command.
- The device is powered off.
- The device is put into Suspend mode.

The Session ID is used by the host to uniquely identify the present transaction. Its primary purpose is to prevent replays. After the device reads a card, it encrypts the Session ID along with the card data, and supplies it as part of the **Magnetic Stripe Card Data Sent from Device to Host**. The device never transmits a clear text version of this data.

Table 8-13 - Request Data for Command 0x0A - Set Session ID (MSR Only)

Offset	Field Name	Description
0	New Session ID	This eight byte value may be any value the host software wishes.

Response Data: None

Result codes:
0x00 = Success
0x02 = Bad Parameter - The Data field in the request is not the correct length. The Session ID is an 8-byte value, so the Data Length must be 8.

Example Set Session ID Request (Hex)

Cmd Num	Data Len	Data
0A	08	54 45 53 54 54 45 53 54

Example Set Session ID Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.9 Command 0x10 - Activate Authenticated Mode (MSR Only)

This command is used by the host software to activate Authenticated Mode, and is the only way to enter that mode. When the device is set to Security Level 4 (see section **4.4 Security Level 4**), it does not gather and transmit card data after a swipe until Authenticated Mode has been established with the host, indicating both devices have established a direct two-way trust relationship. The general sequence of events for entering Authenticated Mode is as follows:

1) The cardholder or operator performs an action as a lead-in to swiping a card, such as signing in to a web page that interacts with the device.
2) The host software is aware of the cardholder action, and in response it sends the Activate Authenticated Mode command to the device. As part of this command, the host software specifies a PreAuthentication Time Limit parameter in units of seconds. The device uses this time limit in subsequent steps. The device interprets any value less than 120 seconds to mean 120 seconds.
3) The device responds to the host with the current Key Serial Number (KSN) and two challenges (Challenge 1 and Challenge 2), which it encrypts using a custom variant of the current DUKPT Key (Key XOR F0F0 F0F0 F0F0 F0F0 F0F0 F0F0 F0F0 F0F0). Challenge 1 contains 6 bytes of random numbers followed by the last two bytes of the KSN. Challenge 2 contains 8 bytes of random numbers.
4) The device waits up to the PreAuthentication Time Limit. If the device times out waiting for the host to respond, the Authentication attempt fails and the device may activate anti-hacking behavior. See below for details.
5) The host software decrypts Challenge 1 and Challenge 2 and compares the last two bytes of the KSN with the last two bytes of the clear text KSN to authenticate the device.
6) The host software completes the Activate Authentication sequence using **Command 0x11 - Activation Challenge Response**, including the length of time the device should keep Authenticated Mode active without a swipe.
7) The device determines whether the Activation Challenge Reply is valid. If it is valid, the device activates Authenticated Mode and allows transmission of swiped card data to the host. The device may optionally indicate to the operator that the host and the device are mutually authenticated. See below for information about device behavior when the Activation Challenge Reply is not valid.
8) Authenticated mode stays active until the timeout previously specified by the host in **Command 0x11 - Activation Challenge Response**, or until the device sends valid swipe data to the host, at which point the device deactivates Authenticated Mode.

The first two Activate Authenticated Mode commands may proceed without any delay (one error is allowed with no anti-hacking consequences). If a second Activate Authenticated Mode in a row fails, the device activates anti-hacking behavior by enforcing an increasing delay between incoming Activate Authenticated Mode commands. The first delay is 10 seconds, increasing by 10 seconds up to a maximum delay of 10 minutes. The operator may deactivate anti-hacking mode at any time by swiping any encoded magnetic stripe card. When the device is in this anti-hacking mode, it requires the host to take additional steps to call **Command 0x02 - Reset Device**

To support use of Authenticated Mode, the host software can use **Command 0x14 - Get Device State (MSR Only)** at any time to determine the current state of the device.

Table 8-14 - Request Data for Command 0x10 - Activate Authenticated Mode (MSR Only)

Offset	Field Name	Description
0	PreAuthentication Time Limit (msb)	Most significant byte of the PreAuthentication Time Limit in seconds (120 seconds or greater)

Offset	Field Name	Description
1	PreAuthentication Time Limit (lsb)	Least significant byte of the PreAuthentication Time Limit in seconds (120 seconds or greater)

Table 8-15 – Response Data for Command 0x10 - Activate Authenticated Mode (MSR Only)

Offset	Field Name	Description
0	Current Key Serial Number	This eighty-bit value includes the Initial Key Serial Number in the leftmost 59 bits and the value of the encryption counter in the rightmost 21 bits.
10	Challenge 1	The host should use this eight-byte challenge later in **Command 0x11 - Activation Challenge Response**, and to authenticate the device.
18	Challenge 2	The host should use this eight-byte challenge later in **Command 0x12 - Deactivate Authenticated Mode**.

Result codes:
0x00 = Success
0x03 = Redundant - the device is already in this mode
0x05 = Delayed - the request is refused due to anti-hacking mode
0x07 = Sequence Error - the current Security Level is too low (see section **4 Security Levels**)
0x80 = No MSR Transactions Remaining [see **Command 0x1C - Get Remaining MSR Transactions Counter (MSR Only)**]

Example Request (Hex)

Cmd Num	Data Len	Data
10	00	

Example Response (Hex)

Result Code	Data Len	Data
00	1A	FFFF 0123 4567 8000 0003 9845 A48B 7ED3 C294 7987 5FD4 03FA 8543

8.3.10 Command 0x11 - Activation Challenge Response (MSR Only)

This command is used as the second part of an Activate Authentication sequence following **Command 0x10 - Activate Authenticated Mode**. In this command, the host software sends the first 6 bytes of Challenge 1 (received in response to **Command 0x10 - Activate Authenticated Mode**) plus two bytes of timeout information, and (optionally) an eight byte Session ID encrypted with the a custom variant of the current DUKPT Key (Key XOR 3C3C 3C3C 3C3C 3C3C 3C3C 3C3C 3C3C 3C3C).

The time information contains the maximum number of seconds the device should remain in Authenticated Mode. Regardless of the value of this timer, a card swipe in the Authenticated Mode ends the Authenticated Mode. The maximum time allowed is 3600 seconds (one hour). For example, for a full hour, use `0x0E10`; for 3 minutes, use `0x012C`. A value of `0x00` forces the device to stay in Authenticated Mode until a card swipe or power down occurs (no timeout).

If the host includes Session ID information and the command is successful, it changes the Session ID in the device in the same way as calling **Command 0x0A - Set Session ID**.

If the device decrypts the Challenge Response correctly, Activate Authenticated Mode has succeeded. If the device can not decrypt the Challenge Response correctly, Activate Authenticated Mode fails and the TDES DUKPT Key Serial Number advances.

Table 8-16 - Request Data for Command 0x11 - Activation Challenge Response (MSR Only)

Offset	Field Name	Description
0	Response to Challenge 1	First 6 bytes of Challenge 1 plus a two-byte timeout (MSB first), encrypted by the specified variant of the current DUKPT Key.
8	Session ID	Optional eight byte Session ID encrypted by the specified variant of the current DUKPT Key.

Response Data: None

Result codes:
0x00 = Success
0x02 = Bad Parameters - the Data field in the request is not a correct length
0x04 = Bad Data - the encrypted reply data could not be verified
0x07 = Sequence - not expecting this command

Example Request (Hex)

Cmd Num	Data Len	Data
11	08	8579827521573495

Example Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.11 Command 0x12 - Deactivate Authenticated Mode (MSR Only)

This command is used to exit Authenticated Mode initiated by **Command 0x10 - Activate Authenticated Mode**. It can be used to exit the mode with or without incrementing the DUKPT transaction counter (lower 21 bits of the Key Serial Number). The host software must send the first 7 bytes of Challenge 2 (from the response to **Command 0x10 - Activate Authenticated Mode**) and the Increment flag (0x00 indicates no increment, 0x01 indicates increment the KSN) encrypted with a custom variant of the current DUKPT Key (Key XOR 3C3C 3C3C 3C3C 3C3C 3C3C 3C3C 3C3C 3C3C).

If the device decrypts Challenge 2 successfully, it exits Authenticated Mode, and depending on the Increment flag, may increment the KSN.

If the device cannot decrypt Challenge 2 successfully, it stays in Authenticated Mode until either the time specified in **Command 0x10 - Activate Authenticated Mode** elapses or the cardholder or operator swipes a card. This behavior is intended to discourage denial of service attacks. Exiting Authenticated Mode by timeout or card swipe always increments the KSN; exiting Authenticated Mode using **Command 0x12 - Deactivate Authenticated Mode** may increment the KSN.

Table 8-17 - Request Data for Command 0x12 - Deactivate Authenticated Mode (MSR Only)

Offset	Field Name	Description
0	Response to Challenge 2	Seven bytes of Challenge 2 plus one byte of Increment flag, encrypted by the specified variant of the current DUKPT Key

Response Data: None

Result codes:
0x00 = Success
0x02 = Bad Parameters - the Data field in the request is not the correct length
0x03 = Bad Data - the encrypted reply data could not be verified
0x07 = Sequence - not expecting this command

Example Request (Hex)

Cmd Num	Data Len	Data
12	08	8579827521573495

Example Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.12 Command 0x14 - Get Device State (MSR Only)

When the device is set to **Security Level 4 (MSR Only)**, it requires mutual authentication with the host [see **Command 0x10 - Activate Authenticated Mode (MSR Only)**]. The host can use this command to determine the state of Authenticated Mode at a given point in time. For convenience, this manual refers to states with the notation ***State:Antecedent*** (e.g., **WaitActAuth:BadSwipe**), showing the current state and the state that led to it. Lists of possible states and their definitions are provided in the device response tables below.

In most cases, the host software can also track the state of Authenticated Mode by inference. As the host software interacts with the device, most state transitions are marked by the messages exchanged with the device. The exception is the transition from **WaitActRply:x** to **WaitActAuth:TOAuth**, which happens if the device times out waiting for the host to send **Command 0x11 - Activation Challenge Response (MSR Only)**, which the device does not report to the host. To cover this case, the host must be aware that a timeout could occur, in which case the device responds to **Command 0x11 - Activation Challenge Response (MSR Only)** with Result Code 0x07 (Sequence Error).

Example 1 – Power Up followed by Authentication and good swipe:

1) Device powers on. Host software should send this command to discover the current state of the device is WaitActAuth:PU.
2) Host sends a valid **Command 0x10 - Activate Authenticated Mode (MSR Only)**. Device responds with result code 0x00, inferring the transition to the WaitActRply:PU state.
3) Host sends a valid **Command 0x11 - Activation Challenge Response (MSR Only)**. Device responds with result code 0x00, inferring the transition to the WaitSwipe:PU state.
4) Cardholder swipes a card correctly. Device sends card data to the host, inferring the transition to the WaitActAuth:GoodSwipe state.

Example 2 – Device times out waiting for swipe:

1) Device waiting after a good swipe. Host software may send this command to discover the current state of the device is WaitActAuth:GoodSwipe.
2) Host sends valid **Command 0x10 - Activate Authenticated Mode (MSR Only)**. Device responds with result code 0x00, inferring the transition to the WaitActRply:GoodSwipe state.
3) Host sends a valid **Command 0x11 - Activation Challenge Response (MSR Only)**. Device responds with result code 0x00, inferring the transition to the WaitSwipe:GoodSwipe state.
4) Authenticated mode times out before a swipe occurs. Device sends mostly empty card data to the host to report the timeout in Device Encryption Status. The host infers the transition to the WaitActAuth:TOSwipe state.

Example 3 – Host sends invalid Command 0x11 - Activation Challenge Response (MSR Only):

1) Device waiting after a good swipe. Host software may send this command to discover the current state of the device is WaitActAuth:GoodSwipe.
2) Host sends valid **Command 0x10 - Activate Authenticated Mode (MSR Only)**. Device responds with result code 0x00, inferring the transition to the WaitActRply:GoodSwipe state.
3) Host sends invalid **Command 0x11 - Activation Challenge Response (MSR Only)**. Device responds with result code 0x02 or 0x04, inferring the transition to the WaitActAuth:FailAuth state.

Example 4 – Host waits too long sending Command 0x11 - Activation Challenge Response (MSR Only):

1) Device waiting after a good swipe. Host software may send this command to discover the current state of the device is WaitActAuth:GoodSwipe.

2) Host sends valid **Command 0x10 - Activate Authenticated Mode (MSR Only)**. Device responds with result code 0x00, inferring the transition to the WaitActRply:GoodSwipe state.
3) Device times out waiting for host to send **Command 0x11 - Activation Challenge Response (MSR Only)** (State => WaitActAuth:TOAuth). Host doesn't know because the device does not send any message.
4) Host eventually sends **Command 0x11 - Activation Challenge Response (MSR Only)** (State remains WaitActAuth:TOAuth). Device responds with result code 0x07, inferring the previous transition to WaitActAuth:TOAuth state.

Request Data: None

Table 8-18 - First Byte, Response Data for Command 0x14 - Get Device State (MSR Only)

Current Device State		
Value	Name	Meaning
0x00	WaitActAuth	Waiting for Activate Authenticated Mode. The device requires the host to authenticate using **Command 0x10 - Activate Authenticated Mode** before it accepts swipes.
0x01	WaitActRply	Waiting for Activation Challenge Reply. The host has started to authenticate, and the device is waiting for **Command 0x11 - Activation Challenge Response**.
0x02	WaitSwipe	Waiting for swipe. The device is waiting for the cardholder or operator to swipe a card.
0x03	WaitDelay	Waiting for Anti-Hacking Timer. Two or more previous attempts to Authenticate have failed; the device is waiting for the Anti-Hacking timer to expire before it accepts **Command 0x10 - Activate Authenticated Mode**.

Table 8-19 - Second Byte, Response Data for Command 0x14 - Get Device State (MSR Only)

Current State Antecedent		
Value	Name	Meaning
0x00	PU	Just Powered Up. The device has had no swipes and has not been Authenticated since it was powered up.
0x01	GoodAuth	Authentication Activation Successful. The host has sent the device a valid **Command 0x11 - Activation Challenge Response**.
0x02	GoodSwipe	Good Swipe. The cardholder swiped a valid card correctly.
0x03	BadSwipe	Bad Swipe. The cardholder swiped a card incorrectly or the card is not valid.
0x04	FailAuth	Authentication Activation Failed. The most recent **Command 0x11 - Activation Challenge Response** failed.
0x05	FailDeact	Authentication Deactivation Failed. A recent **Command 0x12 - Deactivate Authenticated Mode** failed.
0x06	TOAuth	Authentication Activation Timed Out. The host failed to send **Command 0x11 - Activation Challenge Response** in the time period specified by **Command 0x10 - Activate Authenticated Mode**.

Current State Antecedent		
0x07	TOSwipe	Swipe Timed Out. The cardholder failed to swipe a card in the time period specified in **Command 0x11 - Activation Challenge Response**.

Result codes:
0x00 = Success

Example Request (Hex)

Cmd Num	Data Len	Data
14	00	

Example Response (Hex)

Result Code	Data Len	Data
00	02	00 00

8.3.13 Command 0x15 - Get / Set Security Level (MAC)

This command is used to set or get the device's current Security Level (see section **4 Security Levels**). The host can use this to raise the Security Level, but can not lower it.

When using this command to set the device's security level, the host should include the specified data in the request, and the device will not return an explicit response. When using this command to get the device's current security level, the host should include no data, and the device will return a response.

(Fixed Key Only)
If the device is configured to use fixed key encryption using **Property 0x6B - Key Management Scheme (Fixed Key Only)**, then MACing is not required. In this case, the MAC field can be omitted.

Table 8-20 - Request Data for Command 0x15 - Get / Set Security Level (MAC)

Offset	Field Name	Description
0	Security Level	Optional: if present must be either `0x03` or `0x04`. If absent, this is a query for the current Security Level.
1	MAC	Four byte MAC to secure the command [see section **4.1 About Message Authentication Codes (MAC)**]. If the host does not include a value for Security Level, it should not include the MAC value.

Table 8-21 - Response Data for Command 0x15 - Get / Set Security Level (MAC)

Offset	Field Name	Description
0	Security Level	Only present if there was no Data in the request. This value gives the current Security Level.

Result codes:
0x00 = Success
0x02 = Bad Parameters. The Data field in the request is not a correct length OR the specified Security Level is invalid; OR the current Security Level is 4.
0x07 = Incorrect MAC; command not authorized

Example Set Security Level Request (Hex)

Cmd Num	Data Len	Data
15	05	03 xx xx xx xx, where xx xx xx xx is a valid MAC

Example Set Security Level Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Data
15	00	

Example Get Security Level Response (Hex)

Result Code	Data Len	Data
00	01	03

8.3.14 Command 0x1C - Get Remaining MSR Transactions Counter (MSR Only)

This command is used to get the maximum number of remaining card swipe transactions or correctly completed Authentication sequences (**Command 0x10 - Activate Authenticated Mode** followed by a correct **Command 0x11 - Activation Challenge Response**) that the device can process. The value it returns is also sometimes referred to as the transaction threshold.

The value has three possible states:

- **Disabled** - value 0xFFFFFF - In this state there is no limit to the number of transactions that can be performed.
- **Expired** - value 0x000000 - This state indicates MSR transactions and Authentication commands are prohibited.
- **Active** - value 1 to 1,000,000 (0x000001 to 0x0F4240) - In this state, each transaction or successful Authentication sequence causes the value to be decremented and allows transactions to be processed. If an Authentication sequence decrements this value to 0, the device permits one final encrypted card swipe.

Request Data: None

Table 8-22 - Response Data for Command 0x1C - Get Remaining MSR Transactions Counter (MSR Only)

Offset	Field Name	Description
0	Device Serial Number	16 bytes of device serial number. If the serial number is shorter than 15 bytes, this value is left-justified and padded with binary zeroes. At least one byte (usually the last one) must contain binary zero.
16	Remaining MSR Transactions	This three byte value contains the current number of remaining MSR transactions.

Result codes:
0x00 = Success

0x02 = Invalid length

Example Request (Hex)

Cmd Num	Data Len	Data
1C	00	

Example Response (Hex)

Result Code	Data Len	Data
00	13	544553542053455455502030303031000007F1 (2033 MSR Transactions Remaining)

8.3.15 Command 0x28 - Power Down (PM1 Only)

This command is used to power down the magnetic stripe circuit. If the device is running on battery only (no USB cable attached), the entire device is powered down. The behavior of the device is the same as if a person had pressed and held down the pushbutton for three seconds to turn it off.

Request Data: None

Response Data: None

Result codes:
0x00 = Success

Example Power Down Request (Hex)

Cmd Num	Data Len	Data
28	00	

Example Power Down Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.16 Command 0x29 - Get Battery Status (PM1 Only | PM5 Only)

This command is used to get the status of the battery.

Request Data: None

Table 8-23 - Response Data for Command 0x29 - Get Battery Status (PM1 Only | PM5 Only)

Offset	Field Name	Description
0	Battery Status	0x00 = Battery is critically low and should be recharged before further use. 0x01 = Battery charge is sufficient for normal use. 0x10 = Battery is charging, but is critically low and should continue to charge before use (PM5 Only) 0x11 = Battery is charging, and battery charge is sufficient for normal use (PM5 Only)

Result codes:
0x00 = Success

Example Request (Hex)

Cmd Num	Data Len	Data
29	00	

Example Response (Hex)

Result Code	Data Len	Data
00	01	00 (Battery is critically low and should be recharged)

8.3.17 Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)

This command is used to initiate a Transaction Validation sequence on the device's display. The device securely displays transaction information and allows the cardholder to securely validate the transaction. The cardholder approves or rejects the transaction by pressing the **Confirm/Approve** (green) button or the **Reject** (red) button.

The Authentication Sequence generally consists of the following steps:

1) The host determines the current DUKPT counter either by using an internal counter from the previous operation's **DUKPT Key Serial Number (KSN)** or by calling **Command 0x09 - Get Current TDES DUKPT KSN**.
2) The host software sends a secured display message with a time period, message, and challenge, using **Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)**.
3) The device locks against accepting new swipes or new Authentication Sequences. Any attempt to run **Command 0x10 - Activate Authenticated Mode** causes the device to return result code `0x07` (Sequence Error).
4) The device decrypts and displays the message to the cardholder.
5) The cardholder either approves or rejects by pressing the green or red button, or the host intervenes. In any of the following cases, the device ends the validation sequence and proceeds to the next step:
   a) If the cardholder presses the **Approve** (green) button, the device shows **APPROVED** for two seconds, then blanks the display.
   b) If the cardholder presses the **Reject** (red) button, the device shows **REJECTED** for two seconds, then blanks the display.
   c) If the cardholder does not press either button within the Validation Time Limit parameter of **Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)**, the device shows **TIME-OUT** for two seconds, then blanks the display.
   d) If the host sends **Command 0x32 - Abort Transaction Validation**, the device displays **Aborted** for two seconds, then blanks the display.
6) The device XORs the response string with the challenge provided by the host earlier, encrypts the result, and sends a fake swipe to the host, with the Transaction Validation Result Available bit in **Device Encryption Status** set to 1.
7) The device unlocks to accept new swipes or new Authentication Sequences.
8) The host calls **Command 0x33 - Get Transaction Validation Result** and uses the decrypted result to determine how to handle the transaction.

The data to be displayed is encrypted so it can only be observed on the secure display.

If the MAC is not valid, the command terminates with an error and the Transaction Validation sequence is not started.

If the MAC is valid, the Display Text is decrypted and displayed. The Display Text value is always 128 bytes long. The message is split into four 32 character segments. Each segment of the message is progressively displayed until all segments have been shown, then the sequence repeats until the cardholder either confirms/rejects the details, the process times-out, or the host aborts the sequence. The device skips any null segments in the display progression.

The display is formatted to show 2 lines of 16 characters each, or a single vertically centered single line of 16 characters. Each display segment or page (32 characters) is split into two lines and displayed. If the string is shorter than 17 characters, the single line is centered vertically on the display. To show a single line on the top or bottom row, the host software can fill the blank line with 16 spaces (`0x20`).

The **Validation Time Limit** indicates the maximum number of seconds the device should remain in the Transaction Validation sequence. The maximum time allowed is 3600 seconds (one hour), set with value `0x0E10`. For a list of properties that govern the behavior of the display during the Authentication Sequence, see the subsections of section **9 Properties** that include the tag **(Transaction Validation)**.

For concrete examples of how this sequence progresses in various use cases, see section **C.1.11 Example: Authentication**.

Table 8-24 - Request Data for Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)

Offset	Field Name	Description
0	Validation Time Limit in seconds (MSB)	Most significant byte of the Validation Time Limit
1	Validation Time Limit in seconds (LSB)	Least significant byte of the Validation Time Limit.
2..129	Encrypted Display Text	This 128 byte value contains a TDES CBC cryptogram of data to display. The data is encrypted using the **Data Encryption, request or both ways** variant of the current DUKPT Key.
130..137	Challenge	Eight bytes with an encrypted random challenge. When the cardholder approves the transaction, the challenge is used to compute the value of the approval.
138..141	MAC	Validates the source of this command. This MAC is computed using the **Message Authentication, request or both ways** variant of the current DUKPT Key on bytes 0..137 of this message (padding the left most bytes of the last block with `0x00`).

Response Data: None

Result codes:
0x00 = Success
0x02 = Bad Parameters - the Data field in the request is not a correct length or the time specified is invalid.
0x03 = Redundant - the device is already in this mode
0x04 = Crypto Error - The MAC is incorrect
0x07 = Sequence Error - the current Security Level is too low (see section **4 Security Levels**)
0x80 = No MSR Transactions Remaining [see **Command 0x1C - Get Remaining MSR Transactions Counter (MSR Only)**

Example Request (Hex)

Cmd Num	Data Len	Data
31	8E	00 1E 20 56 65 72 69 66 79 20 44 65 74 61 69 6C 63 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 23 31 32 33 34 35 36 37 38 39 30 31 32 33 34 35 24 20 20 20 20 20 20 20 20 33 35 30 30 2E 39 39 20 20 20 20 20 41 63 63 65 70 74 20 20 20 20 20 20 20 50 72 65 73 73 20 47 72 65 65 6E 20 20 20 20 20 20 20 20 52 65 6A 65 63 74 20 20 20 20 20 20 20 20 50 72 65 73 73 20 52 65 64 20 20 20 20 01234567

Example Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.18 Command 0x32 - Abort Transaction Validation (Transaction Validation Only)

This command is used to abort a transaction validation display sequence the host initiated with **Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)**. On success, the device shows **Aborted** on the display for two seconds, then blank the display.

Table 8-25 - Request Data for Command 0x32 - Abort Transaction Validation (Transaction Validation Only)

Offset	Field Name	Description
0..7	Abort Response	This is formulated by XORing the cleartext value of the Challenge used in **Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)** with the ASCII letters **ABORTED** and encrypting the result.

Response Data: None

Result codes:
0x00 = Success.
0x03 = Redundant - No Display Transaction Validation sequence is running.
0x04 = Crypto Error - The Abort Response was not correct.

Example Request (Hex)

Cmd Num	Data Len	Data
32	08	FA03D476984784C2

Example Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.19 Command 0x33 - Get Transaction Validation Result (Transaction Validation Only)

This command is used to get the result of the most recent Transaction Validation sequence [see **Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)**]. It returns the 8 byte Transaction Validation Result as follows:

- If no Transaction Validation sequence has been started since the device was powered on, the Result Code is `0x03`, indicating an error.
- If the Transaction Validation sequence is still in progress, the Result Code is `0x08`, indicating the device is busy.
- If the Transaction Validation sequence terminated with the cardholder pressing the **APPROVE** button, the device XORs the decrypted Challenge from **Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)** with `0x415050524F564544` (ASCII string **APPROVED**), encrypts the result using the **Data Encryption, request or both ways** variant of the current DUKPT Key, and returns it in the Transaction Validation Result.
- If the Transaction Validation sequence terminated with the cardholder pressing the **REJECT** button, the device XORs the decrypted Challenge from **Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)** with `0x52454A4543544544` (ASCII string **REJECTED**), encrypts the result using the **Data Encryption, request or both ways** variant of the current DUKPT Key, and returns it in the Transaction Validation Result.
- If the Transaction Validation sequence terminated by timing out (no cardholder interaction), the device XORs the decrypted Challenge from **Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)** with 0x54494D452D4F5554 (ASCII string **TIME-OUT**), encrypts the result using the **Data Encryption, request or both ways** variant of the current DUKPT Key, and returns it in the Transaction Validation Result.
- If the Transaction Validation sequence terminated by the host sending **Command 0x32 - Abort Transaction Validation**, the device XORs the decrypted Challenge from **Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)** with `0x41424F5254454420` (ASCII string **ABORTED**), encrypts the result using the **Data Encryption, request or both ways** variant of the current DUKPT Key, and returns it in the Transaction Validation Result.

Request Data: None

Table 8-26 - Response Data for Command 0x33 - Get Transaction Validation Result (Transaction Validation Only)

Offset	Field Name	Description
0..7	Transaction Validation Result	As described above.

Result codes:
0x00 = Success.
0x03 = Redundant - There was no Display Transaction Validation sequence running.
0x08 = Busy, Transaction Validation sequence is still in progress.

Example Request

Cmd Num	Data Len	Data
33	00	

Example Response (Hex)

Result Code	Data Len	Data
00	08	FA03D476984784C2

8.3.20 Command 0x45 - Get Battery Percentage (PM3 Only | PM4 Only | PM5 Only | PM6 Only | PM7 Only)

This command is used to get the percentage of useful battery charge remaining, in a range between `0x00` (0%) and `0x64` (100%).

Request Data: None

Table 8-27 - Response Data for Command 0x45 - Get Battery Percentage (PM3 Only | PM4 Only | PM5 Only | PM6 Only | PM7 Only)

Offset	Field Name	Description
0	Battery Percentage	

Result codes:
0x00 = Success

Example Request (Hex)

Cmd Num	Data Len	Data
45	00	

Example Response (Hex)

Result Code	Data Len	Data
00	01	62 (Battery at 98%, almost full charge)

8.3.21 Command 0x46 - Send Command to Bluetooth LE Controller (Bluetooth LE Only)

This command sends commands to the device's Bluetooth LE controller, which has its own command set used to control Bluetooth LE-specific aspects of the device. The valid command identifiers and data are defined in the following subsections.

Table 8-28 - Request Data for Command 0x46 - Send Command to Bluetooth LE Controller (Bluetooth LE Only)

Offset	Field Name	Description
0	Data type	Always set to 1 (control data).
1	Message type	Always set to 0 (request).
2	Bluetooth LE command identifier	The identifier of the Bluetooth LE command.
3..n	Bluetooth LE command request data	The data associated with the Bluetooth LE command request.

Table 8-29 - Response Data for Command 0x46 - Send Command to Bluetooth LE Controller (Bluetooth LE Only)

Offset	Field Name	Description
0	Data type	Always 1 (control data).
1	Message type	Always 1 (response).
2	Bluetooth LE result code	A code that indicates the result of the Bluetooth LE command. Valid values for this code are 0 for success, 1 for failure and 2 for bad parameter.
3..n	Bluetooth LE command response data	The data associated with the Bluetooth LE command response.

Result codes:
0x00 = Success
0x01 = Fail (timed out waiting for a response)

Example Request (Hex)

Cmd Num	Data Len	Data
46	06	01 00 02 01 02 03 (Send Echo command)

Example Response (Hex)

Result Code	Data Len	Data
00	06	01 01 00 01 02 03

8.3.21.1 Bluetooth LE Command 0x00 - Get Property

This command gets Bluetooth LE controller properties. The properties are listed in **Appendix A Bluetooth LE Controller Properties**.

Table 8-30 - Request Data

Byte offset	Field name	Description
0	Property identifier	The identifier of the property.

Table 8-31 - Response Data

Byte offset	Field name	Description
0..n	Property value	The value of the property.

8.3.21.2 Bluetooth LE Command 0x01 - Set Property

This command sets Bluetooth LE controller properties. The properties are listed in **Appendix A Bluetooth LE Controller Properties**.

Table 8-32 - Request Data

Byte offset	Field name	Description
0	Property identifier	The identifier of the property.
1..n	Property value	The value of the property.

Response Data: None

8.3.21.3 Bluetooth LE Command 0x02 - Echo

This is a testing command that echoes the data received in the request by transmitting it back to the host as a response.

Table 8-33 - Request Data

Byte offset	Field name	Description
0 - N	Echo data	Data to echo

Table 8-34 - Response Data

Byte offset	Field name	Description
0 - N	Echo data	Data echoed

Example Request (Hex)

Command Identifier	Request Data Length	Request Data
46	06	01 00 02 01 02 03 (echo 01 02 03)

Example Response (Hex)

Result Code	Response Data Length	Response Data
00	06	01 01 00 01 02 03

8.3.21.4 Bluetooth LE Command 0x06 - Erase All Non-Volatile Memory

This command erases the Bluetooth LE module's non-volatile memory, which returns it to its un-configured factory default state. This includes erasing all bonds (see **Bluetooth LE Command 0x07 - Erase All Bonds**). The command requires the host software to include a pair of Secure Code values in the request to make sure the host software does not accidentally invoke this command.

After calling this command, either the host must send **Command 0x02 - Reset Device** or a cardholder / operator must power it off for at least 30 seconds, then power it on, before the changes will take effect. Because this property affects Bluetooth LE communication, it is best to send it using the USB connection.

Table 8-35 - Request Data

Byte offset	Field name	Description
0	Secure code 1	Set to 0x55
1	Secure code 2	Set to 0xAA

Response Data: None

Example Request (Hex)

Command identifier	Request data length	Request data
46	05	01 00 06 55 AA

Example Response (Hex)

Result code	Response data length	Response data
00	03	01 01 00

8.3.21.5 Bluetooth LE Command 0x07 - Erase All Bonds

This command clears all pairing information about known Bluetooth LE hosts from the device. After issuing this command, unpair the device from all paired Bluetooth LE hosts prior to trying to re-pair the device. If any previously paired Bluetooth LE hosts are still in range of the device after issuing this command, they may try to re-connect to the device, which would cause the device to stop advertising and render it unable to re-pair. After clearing the device from all Bluetooth LE hosts, re-pair with the desired Bluetooth LE host(s).

The command requires the host software to include a pair of Secure Code values in the request to make sure the host software does not accidentally invoke this command.

After calling this command, either the host must send **Command 0x02 - Reset Device** or a cardholder / operator must power it off for at least 30 seconds, then power it on, before the changes will take effect. Because this property affects Bluetooth LE communication, it is best to send it using the USB connection.

Table 8-36 - Request Data

Byte offset	Field name	Description
0	Secure code 1	Set to 0x55
1	Secure code 2	Set to 0xAA

Response Data: None

Example Request (Hex)

Command identifier	Request data length	Request data
46	05	01 00 07 55 AA

Example Response (Hex)

Result code	Response data length	Response data
00	03	01 01 00

8.3.21.6 Bluetooth LE Command 0x0B - Terminate Bluetooth LE Connection

This command signals the device to wait 1 second then terminate the specified Bluetooth LE connection. The delay allows time for the host software to receive a response from the device if the command is issued over Bluetooth LE. To conserve battery power, the Bluetooth LE host should terminate the Bluetooth LE connection when it does not need to communicate to the device. Instead of using this command, the Bluetooth LE host may also directly terminate the Bluetooth LE connection if it is capable.

Request Data: None

Response Data: None

Example Request (Hex)

Command identifier	Request data length	Request data
46	03	01 00 0B

Example Response (Hex)

Result code	Response data length	Response data
00	03	01 01 00

8.3.21.7 Bluetooth LE Command 0x0D - Get Bond Count (Pairing Modes Only)

This command can be used to retrieve the number of hosts currently bonded to the device. The device can bond with up to the maximum number of bonds specified in **Bluetooth LE Property 0x16 - Maximum Bond Count (Pairing Modes Only)**.

Request Data: None

Response Data: One byte that contains the number of hosts currently bonded to the device.

Example Request (Hex)

Command identifier	Request data length	Request data
46	03	01 00 0D

Example Response (Hex)

Result code	Response data length	Response data
00	04	01 01 00 03 (03 = 3 bonds)

8.3.22 Command 0x48 - Notification Output Connection Override (Bluetooth LE Only | iAP Only, USB Only)

The host uses this command to immediately and temporarily override the current setting in **Property 0x5F - Notification Output Connection (Bluetooth LE Only | iAP Only, USB Only**) until the device is power cycled or reset, changing the connection the device uses to send **Magnetic Stripe Card Data Sent from Device to Host** [see section **2 Connection Types**] and **Notification Messages Sent from Device to Host (Extended Notifications Only)** to the host.

If the host does not specify a connection type in the request, the response's Connection value returns the current connection type, otherwise the response contains no additional data.

Table 8-37 - Request Data for Command 0x48 - Notification Output Connection Override (Bluetooth LE Only | iAP Only, USB Only)

Offset	Field Name	Description
0	Connection	0x00 = USB 0x01 = Bluetooth LE (Bluetooth LE Only) 0x02 = iAP (iAP Only)

Table 8-38 - Response Data for Command 0x48 - Notification Output Connection Override (Bluetooth LE Only | iAP Only, USB Only)

Offset	Field Name	Description
0	Connection	0x00 = USB 0x01 = Bluetooth LE (Bluetooth LE Only) 0x02 = iAP (iAP Only)

Result codes:
0x00 = Success

Example Request (Hex)

Cmd Num	Data Len	Data
48	01	01

Example Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.23 Command 0x49 - Send Extended Command Packet (Extended Commands Only)

The host uses this command to send **extended commands** to the device as one or more data packets. This **extended commands protocol** doubles the command number namespace to two bytes, doubles the result code namespace to two bytes, and supports commands and responses which require larger data payloads than those available for standard commands (shown in **Table 8-1** and **Table 8-2** in section **8.1 About Commands**).

If the required command data is 52 bytes or shorter, the host can send the entire command using a single extended command packet. If the command data is longer than 52 bytes, the host should split the data into multiple packets of 52 or fewer bytes, and send multiple extended command packets. Assuming 52-byte packets, the first packet the host sends should specify Extended Data Offset = 0, the next packet should specify Extended Data Offset = 52, and so on, until the host has sent all the command data. The device's response to each packet contains either an extended command result code or a standard result code for the command that was sent:

- **Result Code 0x0B - Extended Protocol Request Pending** indicates the device is buffering the incoming data and expects the host to send subsequent packets.
- **Result Code 0x0A - Extended Command Response** indicates the device has received the complete data set and has executed the command. If the device has 52 bytes or fewer to return to the host, that concludes the round trip of the command. If the response data is greater than 52 bytes, the host must retrieve additional data by continuing to call **Command 0x4A - Get Extended Response** until it has retrieved all response data.
- **Standard Result Code.** When using this command to invoke a standard command (as opposed to an extended command), see the Result Codes in the documentation for the command the host is invoking.

To simplify the development of custom host software, developers who are working exclusively with devices that support extended commands may choose to send all commands, including the single-byte commands described in this manual, using the extended commands protocol.

Table 8-39 - Request Data for Command 0x49 - Send Extended Command Packet (Extended Commands Only)

Offset	Field Name	Description
0..1	Extended Data Offset	This field is in big endian format. It indicates the byte offset position of this packet's Extended Data field, relative to the complete extended data field being sent as multiple packets. The Extended Data Offset of Packet 0 is 0.
2..3	Extended Command Number	This field is in big endian format and contains the number of the command to execute. For one-byte command numbers, the high byte should be set to zero.
4..5	Complete Extended Data Length	This field is in big endian format and gives the total length of the Extended Data field the host is sending as multiple packets.
6..n	Extended Data	This field contains either part or all of the extended data request the host is sending to the device. The size of this Extended Data field can be determined by subtracting the Extended Data field's offset within the request (6) from the request's total data length (N). In most cases the request's complete data payload can have a maximum value of 58 (for example see section **2.1.2 How to Send Commands On the USB Connection**), so this field can have a maximum length of 58 - 6 = 52 bytes.

Table 8-40 - Response Data for Command 0x49 - Send Extended Command Packet (Extended Commands Only)

Offset	Field Name	Description
0..1	Extended Data Offset	This field is in big endian format. It indicates the byte offset position of this packet's Extended Data field, relative to the complete extended data field being sent as multiple packets. The first byte is offset zero.
1..2	Extended Result Code	This field is in big endian format. For one-byte result codes, the high byte is set to zero.
4..5	Complete Extended Data Length	This field is in big endian format and gives the total length of the extended data field the host is sending as multiple packets.
6..n	Extended Data	This field contains either part or all of the complete Extended Data response the device is sending to the host. The size of this Extended Data field can be determined by subtracting the Extended Data field's offset within the response (6) from the response's total data length (N). In most cases the response's complete data payload can have a maximum length of 58 (for example see section **2.1.2 How to Send Commands On the USB Connection**), so this field can have a maximum length of 58 - 6 = 52 bytes.

Result Codes:
See command description.

Example Request (Hex)

Cmd Num	Data Len	Data
49	06	00 00 03 0D 00 00 [**Extended Command 0x030D - Read Date and Time**]

Example Response (Hex)

Result Code	Data Len	Data
0A	0D	00 00 00 00 00 07 06 14 11 00 00 00 01 (6/20/2009 5:00pm)

8.3.24 Command 0x4A - Get Extended Response (Extended Commands Only)

The host uses this command to retrieve additional response data longer than the current connection type's maximum packet size. After calling a command, if the device returns generic result code **0x0A Extended response first packet** (see **Table 8-3 - Generic Result Codes** on page **93**), the host software should begin buffering the complete Extended Response starting with the initial response, then call this command repeatedly until it has retrieved the complete Extended Response.

The response data from the device follows the same Extended Data Offset rule as **Command 0x49 - Send Extended Command Packet** from the host: The first packet the device sends to the host specifies Extended Data Offset = 0, and subsequent packets, if any, specify Extended Data Offset = 52 (or other packet length depending on connection type), then 104, 156, and so on, until the device has sent all the response data. The host should continue sending this command to the device and buffering the returned Extended Data until the Extended Data Offset plus the length of the Extended Data equals the Complete Extended Data Length.

Request Data: None

Table 8-41 - Response Data for Command 0x4A - Get Extended Response (Extended Commands Only)

Offset	Field Name	Description
0..1	Extended Data Offset	This field is in big endian format. It indicates the byte offset position of this packet's Extended Data field, relative to the complete extended data field being sent as multiple packets. The first byte is offset zero.
2..3	Extended Result Code	This field is in big endian format. For one byte result codes, the high byte is set to zero.
4..5	Complete Extended Data Length	This field is in big endian format and gives the total length of the extended data field the device is returning to the host in multiple packets. If the complete extended data fits in a single packet, this field is equal to the Data Length field minus 6.
6..n	Extended Data	This field contains either part or all of the extended data the device is sending to the host. The size of this Extended Data field can be determined by subtracting the Extended Data field's offset within the response (6) from the response's total data length (N). In most cases the response's complete data payload can have a maximum value of 58 (for example see section **2.1.2 How to Send Commands On the USB Connection**), so this field can have a maximum length of 58 - 6 = 52 bytes.

Result Codes: Same as defined in **Command 0x49 - Send Extended Command Packet**.

Example Request (Hex)

Cmd Num	Data Len	Data
4A	00	

Example Response (Hex)

Result Code	Data Len	Data
0A	09	00 34 00 00 00 37 35 36 37 (Last 3 bytes of extended data out of 55 bytes)

8.3.25 Command 0x4C - Get Tamper Status (Tamper Only)

This command retrieves two bytes representing the device's tamper history and current tamper status, including which tamper circuits are active / armed to detect tampers, which ones have detected a tamper in the past, and which ones are currently registering a tamper. It also reports whether the device signature has been erased in response to tampering or removal of the tamper detection battery. When a device has detected a tamper, all incoming commands, including this one, fail.

Request Data: None

Table 8-42 - Response Data for Command 0x4C - Get Tamper Status (Tamper Only)

Offset	Field Name	Description
Byte 0	Tamper History	Bit 0 Tamper Armed Status: 1 = Tamper is active / armed 0 = Tamper is not active / armed Bit 1 Tamper Circuit 1 History: 1 = Circuit 1 was tampered 0 = Circuit 1 was not tampered Bit 2 Tamper Circuit 2 History: 1 = Circuit 2 was tampered 0 = Circuit 2 was not tampered Bit 3 Device Signature Tamper History: 1 = Device Signature was tampered 0 = Device Signature was not tampered Bit 4 Device Signature Erased History: 1 = Device Signature was erased 0 = Device Signature was not erased.
Byte 1	Tamper Status	Bit 0 Tamper Circuit 1 Status 1 = Circuit 1 is open 0 =-Circuit 1 is closed Bit 1 Tamper Circuit 2 Status 1 = Circuit 2 is open 0 = Circuit 2 is closed

Result codes:
0x00 = Success

Example Request (Hex)

Cmd Num	Data Len	Data
4C	00	

In the following example, tamper is armed, circuit 1 has registered a tamper, and the device signature was erased.

Example Response (Hex)

Result Code	Data Len	Data
00	02	13 02

8.3.26 Command 0x4D - Configure General Status LED (PM3 Only)

Caution: Leaving the General Status LED continuously turned on when the device is running on battery power drastically reduces the battery life.

This command temporarily sets whether the General Status LED is Off or Solid Green when the device is battery powered and ready to read a card. This setting reverts to its default value when the device is power cycled or reset.

If the host doesn't specify an LED state value in the request, the device responds with the current LED state, otherwise the response contains no additional data.

Table 8-43 - Request Data for Command 0x4D - Configure General Status LED (PM3 Only)

Offset	Field Name	Description
0	LED State	0 = Off 1 = Green

Table 8-44 - Response Data for Command 0x4D - Configure General Status LED (PM3 Only)

Offset	Field Name	Description
0	LED State	0 = Off 1 = Green

Result codes: 0x00 = Success

Example Request (Hex)

Cmd Num	Data Len	Data
4D	01	01 (Green)

Example Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.27 Command 0x4E - Load Fixed Key (Fixed Key Only)

Caution: Make sure the host knows the current fixed key before calling this command with the Authentication Required field set to True, or there will be no way to load a new fixed key onto the device.

The host software uses this command to load a 16-byte fixed key into the device when **Property 0x6B - Key Management Scheme (Fixed Key Only)** is set to **Fixed Key**.

To load a fixed key, the host software should follow these steps:

1) Generate or obtain the key to be injected. This can be as simple as generating a string of 16 random hexadecimal digits, or as involved as requesting a key management team generate a key in a specific format using a Hardware Security Module (HSM). Specific methods of generating keys are outside the scope of this documentation.
2) Decide whether to load a single encrypted fixed key or send it as two components in the clear. If the key is a simple string of 16 hexadecumal digits but the host is loading it as two components in the clear, one component can be the actual key and the other can be `0000000000000000` (16 zeroes).
3) Decide on a 10-byte KSN for the key. The device includes this KSN alongside any encrypted data it sends [for example in data object `DFDF56` in **ARQC Messages (EMV Only)** and **Transaction Result Messages (EMV Only)**], so the host can identify the key it should use to decrypt. Choice of KSN is completely the purview of the host software. For example, implementers may choose to use a fixed KSN (effectively no KSN) for closed systems, or may start at `0000000000` (10 zeroes) and increment on each call to this command, or any other algorithm that fits the chosen implementation.
4) Know the current fixed key value, if the device is set to require authentication. When the device ships, the intial fixed key is set to `0000000000000000` (16 zeroes).
5) Know if the device is configured to require authentication before loading a fixed key. When the device ships, it is configured to not require authentication so the host can load the first non-default fixed key, and when the host loads that key, it can use the **Authentication Required** flag in this command to require authentication for future fixed key load operations.
6) If loading a single encrypted key:
   a) Encrypt the key using the current fixed key and Electronic Codebook (ECB) encryption
   b) If the device is configured to require authentication, successfully complete the authentication sequence of **Command 0x4F - Fixed Key Authentication Challenge (Fixed Key Only)** followed by **Command 0x50 - Fixed Key Authentication Response (Fixed Key Only)**.
   c) Send this command with Key Data Type set to Encrypted Fixed Key, the chosen KSN, the encrypted Key Data, and the desired Authentication Required setting for subsequent fixed key load operations.
7) If loading two components in the clear:
   a) If the device is configured to require authentication, successfully complete the authentication sequence of **Command 0x4F - Fixed Key Authentication Challenge (Fixed Key Only)** followed by **Command 0x50 - Fixed Key Authentication Response (Fixed Key Only)**.
   b) Send this command with Key Data Type set to Clear Component 1, KSN null, the first key component in Key Data, and null in Authentication Required.
   c) Within 2 minutes, send this command again with Key Data Type set to Clear Component 2, the chosen KSN, the second key component in Key Data, and the desired Authentication Required setting for subsequent fixed key load operations.

The host can also send this command with a request data length of zero; the device does not load any key, and returns information about the current fixed key in its response.

Changes to fixed key settings are effective immediately and persist after a power cycle or reset.

Table 8-45 - Request Data for Command 0x4E - Load Fixed Key (Fixed Key Only)

Offset	Field Name	Description
0	Key Data Type	0 = Encrypted Fixed Key 1 = Clear Component 1. The Key Data field contains component 1 of the new fixed key in the clear. 2 = Clear Component 2. The Key Data field contains component 2 of the new fixed key in the clear. If sent successfully, the device XORs the components to form the new fixed key.
1..10	Key Serial Number	10 byte Key Serial Number (KSN) The device ignores this field when the key data type field is set to 1 (Clear Component 1), because the host also sends this field for Clear Component 2.
11..26	Key Data	16 byte key data, formatted as defined in the Key Data Type field.
27	Authentication Required	0x00 = Authentication not required the next fixed key load 0x01 = Authentication required for the next fixed key load The device ignores this field when the key data type field is set to 1 (Clear Component 1), because the host also sends this field for Clear Component 2.

Table 8-46 - Response Data for Command 0x4E - Load Fixed Key (Fixed Key Only)

Offset	Field Name	Description
0..9	Key Serial Number	See this field's description in the request data.
10..12	Key Check Value (KCV)	The KCV can be used to help identify/validate what key is loaded. It contains the first 3 bytes of the result of encrypting an 8 byte field of zeroes with the current fixed key.
13	Authentication Required	See this field's description in the request data.

Result codes:
0x00 = Success
0x01 = Failure
0x02 = Bad Parameter

Example Request (Hex)

Cmd Num	Data Len	Data
4E	1C	00 0102030405060708090A 8CA64DE9C1B123A78CA64DE9C1B123A7 00 (New key of all zeroes is encrypted under the current key of all zeroes)

Example Response (Hex)

Result Code	Data Len	Data
00	0E	01 02 03 04 05 06 07 08 09 0A 8C A6 4D 00

8.3.28 Command 0x4F - Fixed Key Authentication Challenge (Fixed Key Only)

The host uses this command, along with **Command 0x50 - Fixed Key Authentication Response (Fixed Key Only)**, to authenticate before calling **Command 0x4E - Load Fixed Key (Fixed Key Only)** if that command has previously required authentication to load subsequent fixed keys. Performing this sequence successfully proves to the device that the host has knowledge of the current fixed key value.

Request Data: None

Table 8-47 - Response Data for Command 0x4F - Fixed Key Authentication Challenge (Fixed Key Only)

Offset	Field Name	Description
0 - 7	Authentication Challenge	An 8-byte random number encrypted using the current fixed key.

Result codes:
0x00 = Success

Example Request (Hex)

Cmd Num	Data Len	Data
4F	00	

Example Response (Hex)

Result Code	Data Len	Data
00	08	5B0BF27FCDB6C280 (Random number 371B5A89B509B5FD is encrypted under the current key of all zeroes

8.3.29 Command 0x50 - Fixed Key Authentication Response (Fixed Key Only)

The host uses this command, along with **Command 0x4F - Fixed Key Authentication Challenge (Fixed Key Only)**, to authenticate before calling **Command 0x4E - Load Fixed Key (Fixed Key Only)** if that command has previously configured the device to require authentication for loading subsequent fixed keys. Performing this sequence successfully proves to the device that the host has knowledge of the current fixed key value.

The host should first call **Command 0x4F - Fixed Key Authentication Challenge (Fixed Key Only)**. The device encrypts a random string using the current fixed key, and sends the resulting encrypted 8-byte challenge in its response to the host. The host should then decrypt the challenge using the current fixed key, and send the first four bytes back to the device using this command.

If the Authentication Response field matches the first 4 bytes of the random string the device used, authentication succeeds, otherwise it fails. The result code indicates if it succeeded or failed.

If successful, the device remains in authenticated mode until it is power cycled/reset or until it receives another **Command 0x4F - Fixed Key Authentication Challenge (Fixed Key Only)**.

Table 8-48 - Request Data for Command 0x50 - Fixed Key Authentication Response (Fixed Key Only)

Offset	Field Name	Description
0 - 3	Authentication Response	First four bytes of the unencrypted Authentication Challenge

Response Data: None

Result codes:
0x00 = Success
0x01 = Failure
0x02 = Bad Parameter

Example Request (Hex)

Cmd Num	Data Len	Data
50	04	371B5A89 (Challenge 5B0BF27FCDB6C280 is decrypted under the current key of all zeroes)

Example Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.30 Command 0x51 - External LED Control (External LED Control Only)

The host uses this command to directly control the External LED Connector. After a power cycle or reset, the device defaults to driving the external LED and the on-board General Status LED identically. This command only affects the external LED and does not affect the on-board LED.

If the host includes data in the command request message, the device drives the external LED as the command specifies, and does not include data in the response. If the host does not include data in the request message, the device returns the current LED control settings in the response.

See **Table 1-2** External LED feature for information about which devices support this feature.

Table 8-49 - Request Data for Command 0x51 - External LED Control (External LED Control Only)

Offset	Field Name	Description
0	State	One byte specifying what the external LED should do: 0 = Firmware application controlled (default) 1 = Off 2 = Green 3 = Red 4 = Amber
1	Blink Period	One byte specifying the blink period in 10ms units. If the blink period is set to 0x00, the LED does not blink, otherwise the LED continuously turns on for the blink period then off for the blink period. For example, if the blink period is set to 5, the LED turns on for 50ms then off for 50ms then repeats. If the **State** field is set to **Firmware application controlled** or **Off**, the host should set Blink Period to 0x00 and the device ignores it.

Table 8-50 - Response Data for Command 0x51 - External LED Control (External LED Control Only)

Offset	Field Name	Description
0	State	See request data field description.
1	Blink Period	See request data field description.

Result codes: 0x00 = Success

Example Request (Hex)

Cmd Num	Data Len	Data
51	02	03 00

Example Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.31 Command 0x57 - Enter Sleep Mode (PM6 Only)

The host uses this command to force the device to enter low-power sleep mode. The device will wake up automatically when the host makes a connection. To save power, the host may opt to call this command after completing transactions or other operations. It should not send this command while a transaction is pending.

Request Data: None

Response Data: None

Result Codes:
0x00 = SUCCESS
0x01 = FAILURE
0x02 = BAD_PARAMETER

Example Request (Hex)

Cmd Num	Data Len (1 byte)	Data
57	00	

Example Response (Hex)

Cmd Num	Data Len (1 byte)	Data
00	00	

8.3.32 Command 0x58 - Set Head Subsystem Power State (PM5 Only | PM7 Only)

The host uses this command to directly control power to the magnetic stripe reader head inside the device, to manage device power consumption and battery life. The tradeoff to using direct control is that the host must power up the head before a cardholder swipes a magnetic stripe card outside the scope of an EMV transaction. In the case of EMV transactions, the device controls head power automatically.

After the device is power cycled or reset, it sets the head power state to the default value set in **Property 0x70 - Head Subsystem Power State Default**, at which point the host can directly manage the head power state by calling this command. If the host includes data in the command request message, the device sets the state as the command specifies, and does not include data in the response. If the host does not include data in the request message, the device returns the current state in the response.

When the state is set to **Always On**, the head will always be powered.

When the state is set to **Off When Idle**, the head will be off when the device is idle. In this state, the device will not be able to read magnetic stripe cards outside the scope of EMV transactions without first turning the head on. After reading a card, the host may opt to turn the head back off to conserve power. If the device needs to use the head for any operation other than reading a magnetic stripe card outside an EMV transaction, it will automatically turn the head on while using it, then off again. For example:

- If the host sends **Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)** and the transaction requires an MSR swipe, the device will automatically power up the head, then power it down when the EMV transaction terminates.
- When the host sends **Command 0x09 - Get Current TDES DUKPT KSN**, the device will automatically power up the head, then power it down after retrieving the required data.
- The same behavior applies to all commands and operations where the device needs to use the head.

Table 8-51 - Request Data for Command 0x58 - Set Head Subsystem Power State (PM5 Only | PM7 Only)

Offset	Field Name	Description
0	State	One byte specifying what the state should be: 0x00 = Off When Idle 0x01 = Always On

Table 8-52 - Response Data for Command 0x58 - Set Head Subsystem Power State (PM5 Only | PM7 Only)

Offset	Field Name	Description
0	State	See request data field description.

Result codes: 0x00 = Success

Example Request (Hex)

Cmd Num	Data Len	Data
58	01	01

Example Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.33 Command 0x59 - Power Saving Timeouts Override (PM5 Only)

The host uses this command to immediately and temporarily override the current setting in **Property 0x71 - Power Saving Timeout (PM5 Only)**. After the device is power cycled or reset, it reverts to using the timeouts specified by that property. See the documentation for the overridden property for details.

If the host does not specify a set of timeout values in the request, the response returns the current timeout settings, otherwise the response contains no additional data.

Table 8-53 - Request Data for Command 0x59 - Power Saving Timeouts Override (PM5 Only)

Offset	Field Name	Description
0	Timeout Values	Byte 1 = Sleep Mode timeout Byte 2 = Power Off Mode timeout

Table 8-54 - Response Data for Command 0x59 - Power Saving Timeouts Override (PM5 Only)

Offset	Field Name	Description
0	Timeout Values	Byte 1 = Sleep Mode timeout Byte 2 = Power Off Mode timeout

Result codes:
0x00 = Success

Example Get Request (Hex)

Cmd Num	Data Len	Data
59	00	

Example Get Response (Hex)

Result Code	Data Len	Data
00	02	0F20

Example Set Request (Hex)

Cmd Num	Data Len	Data
59	02	0F20

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.34 Command 0x71 - Override Pushbutton Functions (Configurable Pushbutton Only)

The host uses this command to immediately and temporarily override the function of pushbutton. After the device is power cycled or reset, it reverts its default functionality. The default value is 0x01.

Bit Position	7	6	5	4	3	2	1	0
Value	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	S

Set Reserved bits to 0.

Setting S to 0 causes the device not power off when an operator pushes the pushbutton. After disabling the pushbutton, the host must set S back to 1 to re-enable the pushbutton to allow operators to power off the device.

Result Code:
0x00 = Success
0x01 = Failure

Example Get Request (Hex)

Cmd Num	Data Len	Data
71	00	

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	01 (Enable)

Example Set Request (Hex)

Cmd Num	Data Len	Data
71	01	00 (Disable)

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

8.3.35 Command 0x73 – Contactless Read Delay

The host uses this command to immediately change the current setting of Contactless Read Delay. The read delay can be configured to enable or disable. This command also provides an option to the host to keep the change permanent or temporary. If the change is temporary, after the device is power cycled or reset, it reverts to using the last value.

Offset	Field Name	Description
0	Read Delay	One byte specifying what the state of delay should be: 0x00 = Delay is disabled 0x01 = Delay is enabled (Default)
1	Preservation Value	One byte specifying what the state should be: 0x00 = Temporary 0x01 = Permanent

Result codes:
0x00 = Success
0x01 = Failure

Table 8-55 - Example Set Request (Hex)

Cmd Num	Data Len	Data
73	02	00 00

Table 8-56 - Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Table 8-57 - Example Get Request (Hex)

Cmd Num	Data Len	Data
73	00	

Table 8-58 - Example Get Response (Hex)

Result Code	Data Len	Data
00	02	00 00

8.4 Command Group 0x03 - EMV L2 (EMV Only, Extended Commands Only)

When calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, a value of `0x03` in the most significant byte of the Extended Command Number is reserved for EMV L2 commands, which are documented in this section.

8.4.1 About MACs

Many commands in this command group require a MAC field, which the host must populate using the UIK loaded into the device. For details, see section **4.1 About Message Authentication Codes (MAC)**.

8.4.2 About EMV L2 Transaction Flows (EMV Only)

The general flow of an EMV L2 transaction is as follows (bear in mind the device does not have a display, so in these steps the host drives the user interface for both the terminal operator / cashier and for the cardholder / customer):

1) The terminal operator / cashier performs steps external to the transaction, generally resulting in a total balance owed, and directs the host software to initiate a transaction. If the device supports Quick Chip and the system is designed to use that feature, the host may skip this step and instead start the transaction with a default amount as a placeholder, which is generally a pre-determined non-zero value that is consistent with the system's payment processing environment. Further differences pertaining to Quick Chip transactions are included in the steps below.
2) The host software sends the device **Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)**.
3) From this point until the host sends the device transaction results to the transaction processor, the host may cancel the EMV transaction by sending **Extended Command 0x0304 - Cancel Transaction (EMV Only)** and the device sends report **Notification 0x0300 - Transaction Status / Progress Information** to report **End of Transaction / Host Canceled EMV Transaction Before Card Was Presented**.
4) If the cardholder has not already presented payment, the device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Waiting for Cardholder Response / Waiting for Cardholder to Present Payment**, followed by **Notification 0x0301 - Display Message Request** to prompt the cardholder to PRESENT CARD. The device waits until the cardholder presents payment, pending a timeout. (Contactless Only) If the cardholder presents more than one chip card or contactless payment device at the same time, the device sends **Notification 0x0301 - Display Message Request** to prompt the cardholder to PLEASE PRESENT ONE CARD ONLY.
5) Upon chip card insertion or contactless tap, the device sends **Notification 0x0300 - Transaction Status / Progress Information** to report **Card Inserted** (or **Contactless Token Detected**) / **Powering Up Card**.
6) (Contactless Only) If the device and the contactless payment device (such as a smartphone) mutually support Mobile CVM, and the contactless payment device's conditions for requiring Mobile CVM occur, the device sends the host **Notification 0x0301 - Display Message Request** with the message SEE PHONE FOR INSTRUCTIONS. Upon withdrawing the contactless payment device from the NFC field, the device sends the host **Notification 0x0301 - Display Message Request** with the message PRESENT CARD AGAIN. Simultaneously, the contactless payment device prompts the cardholder to enter a PIN. The cardholder should enter a PIN on the contactless payment device, then re-tap.
7) (MSR Only) At this point, if the MSR is enabled and **Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)** indicated the MSR should be armed, the cardholder may swipe a magnetic stripe card:
   a) (EMV MSR Flow Only) Upon detecting a swipe, the device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report a successful non-fallback swipe with **MSR**

Swipe Detected / Reading Magnetic Stripe. If decoding fails, the device sends the host **Notification 0x0301 - Display Message Request** with message SWIPE AGAIN. If decoding fails three times, the device sends **Notification 0x0300 - Transaction Status / Progress Information** to report payment method error and transaction error, then again to report end of transaction and no transaction in progress, then sends **Notification 0x0301 - Display Message Request** to the host with the message TRANSACTION TERMINATED and terminates the transaction.

b) (EMV MSR Flow Only) If the device finds the Service Code on the card begins with '2' or '6', indicating the card is a chip card, the device clears all swipe data from memory and sends **Notification 0x0301 - Display Message Request** to the host with the message USE CHIP READER. The host should display the message to the operator or cardholder. The device waits for the cardholder to insert the card. Otherwise, if the Service Code indicates the card is an MSR-only card, the device treats the transaction as Online Only, jumps to the sending of **Notification 0x0303 - ARQC Message** below, and continues to the end of the transaction.

c) (No EMV MSR Flow Only) On devices that do not support EMV MSR Flow, the device cancels the EMV transaction the host initiated, and reverts to the behavior described in section **6 Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)**.

8) (Contact Only) If the cardholder has inserted a chip card, the device attempts to communicate with the card. If it is unable to do so:

a) (Contact Only, No EMV MSR Flow Only) On devices that do not support EMV MSR Flow, the device immediately terminates the transaction with no retries and sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Payment Method Communication Error / Transaction Error**, followed by **Notification 0x0301 - Display Message Request** to the host with the message TRANSACTION TERMINATED.

b) (Contact Only, EMV MSR Flow Only) On devices that support EMV MSR Flow, the device sends **Notification 0x0301 - Display Message Request** to the host with the message INSERT AGAIN. If subsequent insertions succeed, the device continues, otherwise it will repeat this process for a total of three insertions. Upon the third failure:

i) If Terminal setting DFDF67 is set to **Fallback Disabled**, the device terminates the transaction and sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Payment Method Communication Error/ Transaction Error**, followed by **Notification 0x0301 - Display Message Request** to the host with the message TRANSACTION TERMINATED.

ii) If Terminal setting DFDF67 is set to **Fallback Enabled**:

(1) The device sends the host **Notification 0x0301 - Display Message Request** with the message CHIP ERROR: SWIPE CARD. The host should display the message.

(2) Upon successfully decoding a magnetic stripe swipe, the device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report event **MSR Swipe Detected/Reading Magnetic Stripe**, followed by **Notification 0x0301 - Display Message Request** with message PLEASE WAIT, followed by **Notification 0x0300 - Transaction Status / Progress Information** to report event **Transaction Progress Change / Online Processing**. If the swipe fails, the device sends the host **Notification 0x0301 - Display Message Request** with message SWIPE AGAIN and waits for the cardholder to swipe. If three swipes fail, the device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change / Magnetic Stripe Decoding Error**, then **Notification 0x0300 - Transaction Status / Progress Information** to report **Data Error / Transaction Error**, then sends **Notification 0x0304 - Transaction Result Message** and terminates the transaction.

(3) Because the MSR transaction must always be performed online, the device passes the MSR track data to the host for forwarding to the transaction processor using **Notification 0x0303 - ARQC Message** containing MSR Data Container F4. Inside the message, the host should examine tag DFDF53 to determine the cause of the MSR Fallback (**Technical Fallback** or **MSR Fallback**). The device then sends **Notification 0x0300 - Transaction Status / Progress Information** to report status **Transaction Progress Change / Waiting Online Processing Response**.

(4) The host processes the ARQC Message data and uses it to coordinate with the transaction processor to receive an ARPC Response, which it processes and sends to the device using **Extended Command 0x0303 - Online Processing Result / Acquirer Response (EMV Only)**.

(5) The device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change / Magnetic stripe card decoded during Technical Fallback**.

(6) The device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **End of Transaction** and either **Transaction Approved** or **Transaction Declined**.

(7) The device sends **Notification 0x0301 - Display Message Request** with message **APPROVED** or **DECLINED** to notify the cardholder of the transaction result.

(8) The device ends the transaction by sending the host **Notification 0x0304 - Transaction Result Message**, which contains transaction details the host should save for later verification. Inside the message, the host should examine tag DFDF53 to determine the cause of the fallback to an MSR swipe (**Technical Fallback** or **MSR Fallback**).

9) The device negotiates with the card to determine which payment application to use as follows:

a) (Application Selection Options Only) If the card holds more than one mutually supported payment application, the device proceeds based on the setting of **Property 0x73 - Application Selection Behavior (Application Selection Options Only)**. If the property is set to **Prompt Cardholder**, the device follows the same steps as below. If the property is set to a different value, the device deviates from the steps below as documented in that property's description.

b) The device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change / Selecting the Application**, followed by **Notification 0x0301 - Display Message Request** to prompt the cardholder to **PLEASE WAIT**.

c) If the card holds only one mutually supported payment application, the device proceeds to use that application. If the card holds more than one mutually supported application:

i) The device sends the host **Notification 0x0302 - Cardholder Selection Request** to prompt the cardholder to **Select Application** with a list of available applications, followed by **Notification 0x0300 - Transaction Status / Progress Information** to report **Waiting for Cardholder Response / Waiting for Cardholder Application Selection**.

ii) After the cardholder selects an application, the host passes the selection to the device by sending **Extended Command 0x0302 - Cardholder Selection Result**.

d) The device sends **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change / Initiating Application**.

10) (Contact Only) If the cardholder has inserted a chip card, and the card's selected application reports to the device that the cardholder should select a language, the device sends the host **Notification 0x0302 - Cardholder Selection Request** to prompt the cardholder to **Select Language** with a list of available languages, followed by **Notification 0x0300 - Transaction Status / Progress Information** to report event **Waiting for Cardholder Response / Waiting for Cardholder Language Selection**.

After the cardholder selects a language, the host passes the selection to the device by sending **Extended Command 0x0302 - Cardholder Selection Result**.

11) The device initiates communication with the card and sends the host **Notification 0x0300 - Transaction Status / Progress Information** reporting **Transaction Progress Change/ Reading Application Data**. If an error or other type of failure occurs during this step, the device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Data Error / Transaction Error**, followed by **Notification 0x0301 - Display Message Request** to the host with the message TRANSACTION TERMINATED, followed by **Notification 0x0304 - Transaction Result Message**.
12) (Contactless Only) The device sends the host **Notification 0x0301 - Display Message Request** with message CARD READ OK, REMOVE CARD to notify the cardholder the card can be removed, , followed by **Notification 0x0300 - Transaction Status / Progress Information** to report event **Contactless Powered Up / Contactless Remove Card.**
13) Depending on the capabilities of the card and the device, the device authenticates the card data using SDA, DDA, or CDA. The device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change/ Offline Data Authentication**.
14) The steps from here through **Card Action Analysis** below are collectively referred to as the **Risk Management** process.
15) The device checks to make sure the selected application is valid for the transaction, and is compatible with the device (such as application version number, application usage control, and application effective / expiration date), and sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change/ Process Restrictions**.
16) The device uses the cardholder verification related data in the card or contactless payment device to determine which cardholder verification method (CVMs) to use. The device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change/ Cardholder Verification**.
17) (External PIN Accessory Support Only) If the device determines the transaction requires the cardholder to enter a PIN, it sends the host **Notification 0x0305 - PIN Required**. In response, the host should coordinate PIN entry with an external PIN entry device, and use that data in its communication with the payment processor. The host should not share the PIN information back to the device.
18) The device performs terminal risk management procedures, which involves floor limit checking, velocity checking, and periodically forcing online authorization to protect against fraud, and sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change/ Terminal Risk Management**.
19) The device analyzes the results of the previous steps and sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change / Terminal Action Analysis**.
20) The device rolls up the results of the previous Risk Management process:
    a) If the Risk Management process encounters an error or determines the transaction or payment method fails to meet required criteria, the device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Data Error / Transaction Error**, followed by **Notification 0x0301 - Display Message Request** to the host with the message TRANSACTION TERMINATED, followed by **Notification 0x0304 - Transaction Result Message**, and terminates the transaction
    b) If the Risk Management process determines the transaction is too risky to approve, the device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Data Error / Transaction Error**, followed by **Notification 0x0304 - Transaction Result Message**,

followed by **Notification 0x0301 - Display Message Request** with message DECLINED to notify the cardholder, and terminates the transaction.

21) The device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change / Generating First Application Cryptogram**. The device sends the host **Notification 0x0300 - Transaction Status / Progress Information** reporting **Transaction Progress Change/ Card Action Analysis**.

22) If the device is NOT configured as Online-Only Terminal Type [see **Appendix H EMV Terminal and Application Settings (EMV Only)**] and the Risk Management processes determined the transaction is OK to perform offline, the device reports the transaction result to the host as follows:

    a) The device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change / Transaction Complete**.

    b) The device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **End of Transaction** and either **Transaction Approved** or **Transaction Declined**, then sends the host **Notification 0x0301 - Display Message Request** with message APPROVED or DECLINED to notify the cardholder.

    c) The device ends the transaction by sending the host **Notification 0x0304 - Transaction Result Message**, which contains transaction details the host should save for later verification. The transaction result message indicates whether the host must prompt the cardholder to provide a signature.

23) If the device is configured as an Online Only Terminal Type [see **Appendix H EMV Terminal and Application Settings (EMV Only)**] or the Risk Management processes determined the transaction must be performed online, the device reports the transaction result to the host as follows:

    a) The device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change/ Online Processing**, followed by **Notification 0x0303 - ARQC Message**.

    b) The next event depends on whether the device supports the Contact Quick Chip feature or Contactless Quick Chip feature (see **Table 1-2**) and whether the host specified Quick Chip as an Option when it started the transaction:

    c) If Quick Chip operation is supported and in effect:

        i) The device immediately constructs its own internal ARPC Response, with tag 8A set to ‘Z3’ to coordinate the transaction with the card or other payment method, and sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change / Transaction Complete**, followed by **Notification 0x0300 - Transaction Status / Progress Information** to report **End of Transaction / Transaction Declined**.

        ii) The device sends the host **Notification 0x0301 - Display Message Request** with message REMOVE CARD to notify the cardholder the card can be removed.

        iii) The host should then process the ARQC Message data, including replacing the default amount with the final transaction amount, and should coordinate with the transaction processor to retrieve a final transaction result. Because in this case the device is not involved in determining the final transaction result, it does not send a notification to the host to show APPROVED or DECLINED. Instead, the host should display an appropriate message (such as QUICK CHIP APPROVED / QUICK CHIP DECLINED) to the cardholder based on the final transaction result.

        iv) The device ends the transaction by sending the host **Notification 0x0304 - Transaction Result Message**, which contains transaction details the host should save for later verification. The transaction result message indicates whether the host must prompt the cardholder to provide a signature.

    d) If Quick Chip operation is NOT supported or is not in effect:

i) The device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change/ Waiting for Online Processing Response**.

ii) The host processes the ARQC Message data and uses it to coordinate with the transaction processor to receive an ARPC Response, which it processes and sends to the device using **Extended Command 0x0303 - Online Processing Result / Acquirer Response (EMV Only)**. (Contact Only) Alternatively, the host may implement host-driven Quick Chip by instead constructing its own preliminary ARPC Response with tag 8A set to 'Z3' and sending it to the device immediately, without waiting for a transaction processor response. The device responds by sending **Notification 0x0301 - Display Message Request** to the host with message DECLINED and ending the transaction. The host should suppress this message and take over the remainder of the transaction, including notifying the cardholder to remove the card, determining the final transaction amount, coordinating with the transaction processor to retrieve a final transaction result, and interacting with the cardholder.

iii) The device communicates with the chip card to determine whether to approve or decline the transaction, then sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **Transaction Progress Change / Transaction Complete**.

iv) The device sends the host **Notification 0x0300 - Transaction Status / Progress Information** to report **End of Transaction** and either **Transaction Approved** or **Transaction Declined**, then and sends the host **Notification 0x0301 - Display Message Request** with message APPROVED or DECLINED to notify the cardholder of the transaction result.

v) The device ends the transaction by sending the host **Notification 0x0304 - Transaction Result Message**, which contains transaction details the host should save for later verification. The transaction result message indicates whether the host must prompt the cardholder to provide a signature.

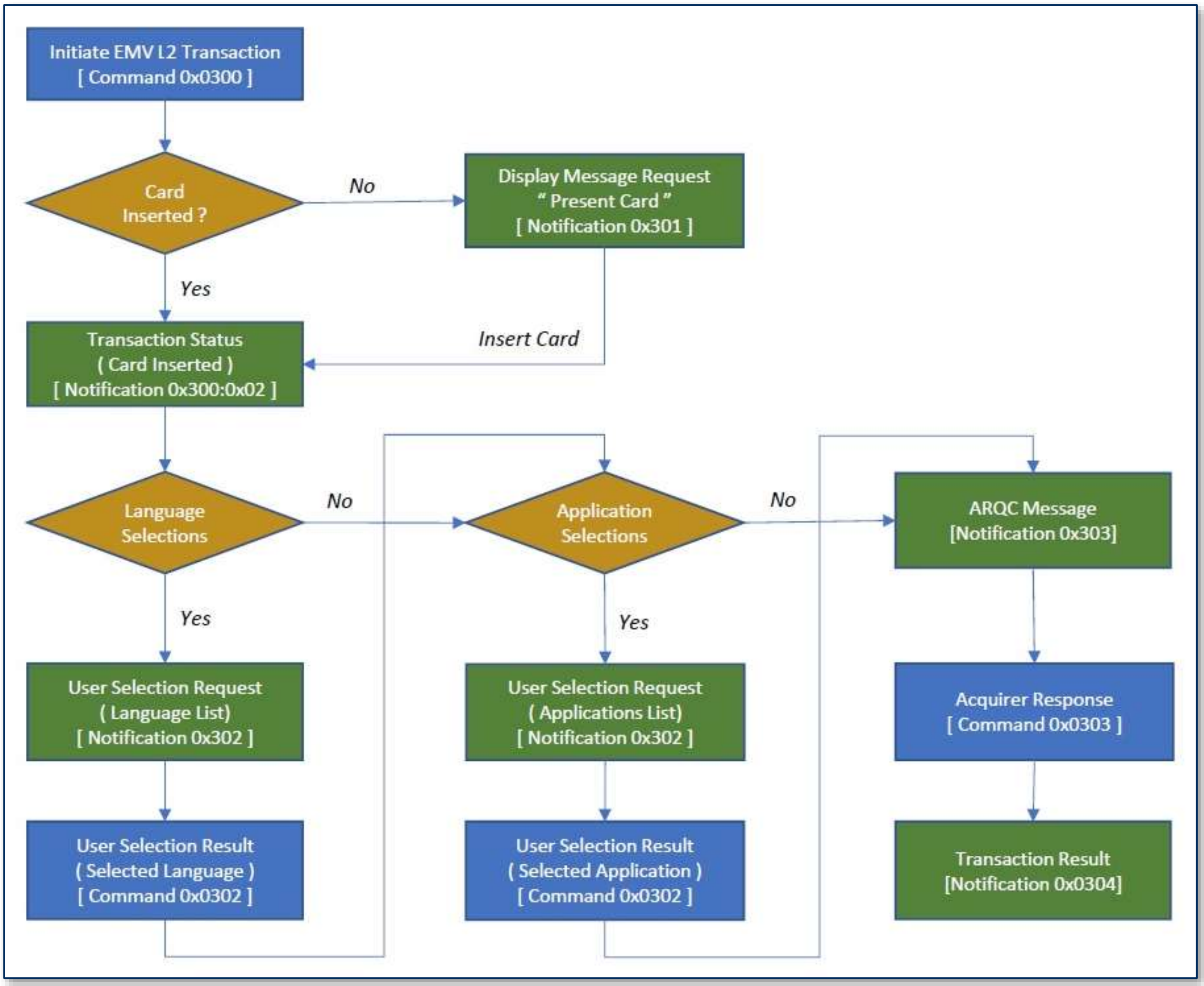

Figure 8-1 - Simplified EMV Transaction Flow

8.4.3 Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

⚠ CAUTION

After the host receives Notification 0x0304 - Transaction Result Message **at the end of a transaction, it is very important to keep the device fully powered for at least 3 seconds. Disconnecting the device's power while it is advancing DUKPT keys after a transaction can corrupt the device's non-volatile memory and render the device unusable.**

⚠ CAUTION

By default, when a transaction terminates without the cardholder presenting payment due to a timeout or Extended Command 0x0304 - Cancel Transaction (EMV Only)**, the device still sends** Notification 0x0304 - Transaction Result Message **and advances its DUKPT keys. In solution designs where this causes excessive consumption of DUKPT keys (such as solutions that continusly loop transactions to impement an "always armed" mode), the host should set** Property 0x74 - EMV Transaction Result Format (EMV Only, Conserve DUKPT Keys Only) **to** Conserve DUKPT Keys**.**

The host uses this command to start an EMV transaction sequence, which flows as described in section **8.4.2 About EMV L2 Transaction Flows (EMV Only)**. The command provides all the data the device needs to start the transaction, and the device returns a response to the host to indicate whether the transaction will proceed. If the input fields for the command are not formatted correctly and within defined limits, the response message returns an error code indicating why the command could not proceed. If the device is set to a lower security level than **Security Level 3**, the device refuses this command, unless the device is an mDynamo, which accepts this command at **Security Level 2**.

If the command proceeds, the response indicates the transaction is proceeding. During transaction processing, the device may generate several notification messages. Some of these notifications may require the host to process data and initiate new commands. Whenever this happens, there is an associated timeout that causes the device to abandon the transaction with an error code if it occurs.

The device's system date and time must be set prior to sending this command:

- Devices that have a battery-backed real time clock (see **Table 1-2 - Device Features**) would typically have the date and time set at the factory.
- Devices without a battery-backed real time clock require the host to set the date and time using **Extended Command 0x030C - Set Date and Time (MAC)** every time the device is power cycled or reset.

After the host sends this command, the device is busy performing the EMV transaction. Until the transaction is complete or terminated, the host should only send commands to the device that directly pertain to the EMV transaction:

- **Extended Command 0x0302 - Cardholder Selection Result**
- **Extended Command 0x0303 - Online Processing Result / Acquirer Response**

- **Extended Command 0x0304 - Cancel Transaction (EMV Only)**
- **Extended Command 0x0305 - Modify Terminal Configuration (MAC)**

Table 8-59 - Request Data for Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)

Offset	Field Name	Value
0	Transaction Flow Time	Specifies the maximum time, in seconds, for cardholder interaction events to complete while processing a transaction. Values from 0x01 to 0xFF are allowed (1 to 255 seconds). The timer starts at the beginning of each event. If the cardholder action does not occur within the specified time, the transaction proceeds as follows: • **Cardholder present payment** timeout: The transaction terminates. • (Contact Only) **Cardholder language selection** timeout: The transaction continues with the default language. • (Contact Only) **Cardholder application selection** timeout: The transaction terminates. (Infinite Transaction Timeout Only) A value of 0x00 directs the device to initiate an EMV transaction and wait indefinitely until a cardholder presents payment, or the host issues **Extended Command 0x0304 - Cancel Transaction (EMV Only)** or the device is power cycled or reset. This allows the host to drive a loop in unattended solutions that idle until the next cardholder initiates a transaction by swiping, inserting, or tapping.
1	Card Type to Read	Card Type to Read (OR the following values together): 0x01 = Magnetic stripe card (MSR Only) 0x02 = Contact chip card (Contact Only) 0x04 = Contactless chip card / payment device (Contactless Only) (MSR Only) Magnetic stripe card and Contact chip card can be enabled at the same time. For details about how the MSR and Contact functions interact, see the introduction to section **8.4 Command Group 0x03 - EMV L2 (EMV Only, Extended Commands Only)**.
2	Options	0x00 = Normal 0x01 = Reserved for Bypass PIN 0x02 = Reserved for Force Online (Apple VAS Only) 0x40 = **VAS App AND Payment Mode (Dual Mode)**. The device reads both Apple VAS data and EMV payment data from a tapped smartphone, or reads EMV payment data from a tapped card. The host must enable **Contactless chip card / payment device** in **Card Type to Read** parameter. Other parameters can use any available options. When the device sends **ARQC Message Format Security Level 3** to the host to conclude the transaction, it includes EMV payment data in container FC and includes VAS data, if available, in container FE.

Offset	Field Name	Value
		0x41 = **VAS App Only Mode (VAS Mode)**. The device reads only Apple VAS data from a tapped smartphone, and does not read data from a tapped card. The host must enable **Contactless chip card / payment device** in **Card Type to Read** parameter. Other parameters can use any available options. If the tapped smartphone does not support VAS, the device does not detect or read from the smartphone. When the device sends **ARQC Message Format Security Level 3** to conclude the transaction, it includes VAS data in container FE and does not include EMV payment data in container FC. 0x42 = **VAS App OR Payment Mode (Single Mode)**. The device reads only Apple VAS data from a tapped smartphone, or reads EMV payment data from a tapped card. The host must enable **Contactless chip card / payment device** in **Card Type to Read** parameter. Other parameters can use any available options. When the device sends **ARQC Message Format Security Level 3** to conclude the transaction, it only includes either EMV payment data in container FC for cards, or includes VAS data in container FE for smartphones. 0x44 = **Payment Only Mode (Payment Mode)**. The device operates the same as when this parameter is set to **Normal**. It reads only EMV payment data from a tapped smartphone or a tapped card. The host must enable **Contactless chip card / payment device** in **Card Type to Read** parameter. Other parameters can use any available options. When the device sends **ARQC Message Format Security Level 3** to conclude the transaction, it includes EMV payment data in container FC and does not include VAS data in container FE. (Quick Chip Only | Contactless Quick Chip Only) To use Quick Chip mode, set the most significant bit to ‘1’. For example: 0x80 = Normal, Use Quick Chip
3..8	Amount Authorized	Amount Authorized (EMV Tag 9F02, format n12, 6 bytes). For Transaction Type **Refund** (0x20), this must contain the refund amount.
9	Transaction Type	0x00 = Purchase (covers transaction types Payment, Goods, and Services) 0x02 or 0x09 = Cash back (0x09 only supported when using contactless) 0x20 = Refund. If the specified Card Type to Read does not formally support refunds, the host can still use **Refund** to retrieve card data it needs to process a refund transaction, but internally and in its responses to the host, the device forces Transaction Type to **Purchase** and replaces Amount Authorized with **0.00**.
10..15	Cash Back	Cash back amount (if non-zero, EMV Tag 9F03, format n12, 6 bytes). For Transaction Type **Refund** (0x20) this must be 0.00.
16..17	Transaction Currency Code	Transaction Currency Code (EMV Tag 5F2A, format n4, 2 bytes) Valid values are the numerical codes from ***ISO 4217 Codes for the representation of currencies***, for example: 0x0000 = Use Selected Application’s Currency Code Terminal Setting 0x0840 = US Dollar 0x0978 = Euro
18	Reporting Option	This single byte field indicates the level of Transaction Status notifications the host wants the device to send during the transaction:

Offset	Field Name	Value
		0x00 = Termination status only (normal termination, payment method communication or data error, timeout, host cancel) 0x01 = Major status changes (terminations plus card insertions and waiting for cardholder) (Contact Only) 0x02 = All status changes (documents the entire transaction flow)

Response Data: None. The response to this command only contains a result code.

Result codes:
0x0000 = Success, the transaction process has been started
0x0381 = Failure, DUKPT scheme is not loaded
0x0382 = Failure, DUKPT scheme is loaded but all of its keys have been used
0x0383 = Failure, DUKPT scheme is not loaded (Security Level not 3 or 4)
0x0384 = Invalid Total Transaction Time field
0x0385 = Invalid Card Type field
0x0386 = Invalid Options field
0x0387 = Invalid Amount Authorized field
0x0388 = Invalid Transaction Type field
0x0389 = Invalid Cash Back field
0x038A = Invalid Transaction Currency Code field
0x038E = Invalid Reporting Option
0x038F = Transaction Already In Progress
0x0391 = Invalid Device Serial Number
0x0396 = Invalid System Date and Time
0x039D = Failure, Failure communicating with Embedded V5 IntelliHead. The device's main microcontroller is unable to communicate with the device's embedded MagneSafe V5 IntelliHead. This may indicate the head is busy or may indicate a hardware failure. If the problem persists, return the device to the manufacturer for service. (Embedded V5 IntelliHead Only)

Example Request (Hex)

Header	
Command Number	49
Data Length	19
Data	
Extended Data Offset	0000
Extended Command Number	0300
Complete Extended Data Length	0013
Extended Data	3C020000000000150000000000000000084002

Example Response (Hex)

Header	
Result Code	0A
Data Length	06

Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	0000
Extended Data	Not Applicable

8.4.4 Extended Command 0x0302 - Cardholder Selection Result

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

The host uses this command to respond to **Notification 0x0302 - Cardholder Selection Request**. After the device sends **Notification 0x0302 - Cardholder Selection Request** to the host, it expects the host to display the specified menu items to the cardholder, then, after the cardholder makes a selection, call **Extended Command 0x0302 - Cardholder Selection Result** to return the number of the item the cardholder selected. The number should be between 1 and the number of menu selection items being displayed. The first item, 0, is the title only.

Table 8-60 - Request Data for Extended Command 0x0302 - Cardholder Selection Result

Offset	Field Name	Value
0	Selection Status	Indicates the status of Cardholder Selection: 0x00 = Cardholder Selection Request completed, see Selection Result 0x01 = Cardholder Selection Request cancelled by cardholder, Transaction Aborted 0x02 = Cardholder Selection Request timed out, Transaction Aborted 0x03 = Cardholder Selection Request timed out, Use Device Defaults (FEATURE TAG) The behavior of the device to each of the responses is dictated by EMV rules.
1	Selection Result	Indicates the menu item selected by the cardholder. This is a single byte binary value.

Response Data: None. The response to this command only contains a result code.

Result codes:
0x0000 = Success, the Selection Result was received
0x038B = Invalid Selection Status
0x038C = Invalid Selection Result
0x038D = Failure, no transaction currently in progress

Example Request (Hex)

Header	
Command Number	49
Data Length	08
Data	
Extended Data Offset	0000
Extended Command Number	0302
Complete Extended Data Length	0002
Extended Data	0001

Example Response (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	0000
Extended Data	Not Applicable

8.4.5 Extended Command 0x0303 - Online Processing Result / Acquirer Response (EMV Only)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

The host uses this command to inform the device of the result of on-line processing. It usually contains an ARPC and optionally Issuer Script 1 / Issuer Script 2 data.

Table 8-61 - Request Data for Extended Command 0x0303 - Online Processing Result / Acquirer Response (EMV Only)

Offset	Field Name	Value
0	Message Length	Two byte binary, most significant byte first. This gives the total length of the ARPC message that follows, excluding padding and CBC-MAC.
2..n	Acquirer Response Message	This is the response from the acquirer. See Appendix **G.2 ARPC Response from Online Processing** for details.

Response Data: None. The response to this command only contains a result code.

Result codes:
0x0000 = Success, the Selection Result was received
0x038D = Failure, no transaction currently in progress
0x038F = Failure, transaction already in progress

Example Request (Hex)

Header	
Command Number	49
Data Length	39
Data	
Extended Data Offset	0000
Extended Command Number	0303
Complete Extended Data Length	003C
Extended Data	003AF92EDFDF540A00000000000000000000DFDF550182DFDF250F4233354532434443038303131364141FA0670048A02303000

Example Response (Hex)

Header	
Result Code	0B
Data Length	00
Data	
Extended Data Offset	Not Applicable

Extended Result Code	Not Applicable
Complete Extended Data Length	Not Applicable
Extended Data	Not Applicable

Example Request Following Up For Packet 0 (Hex)

Header	
Result Code	49
Data Length	0F
Data	
Extended Data Offset	0033
Extended Result Code	0303
Complete Extended Data Length	003C
Extended Data	000000000000000000

Example Response Following Up For Packet 0 (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	0000
Extended Data	Not Applicable

8.4.6 Extended Command 0x0304 - Cancel Transaction (EMV Only)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

The host uses this command to cancel a transaction while the device is waiting for the cardholder to present payment.

Table 8-62 - Request Data for Extended Command 0x0304 - Cancel Transaction (EMV Only)

Offset	Field Name	Value
0	Suppress Sound	(Suppress Sounds Only) Optional one byte parameter to suppress any sounds the device would ordinarily make when the host calls this command: • Not included = Sounds not suppressed • 0x00 = Sounds not suppressed • 0x01 = Sounds suppressed

Response Data: None

Result codes:
0x0000 = Success, the transaction was cancelled
0x038D = Failure, no transaction currently in progress
0x038F = Failure, transaction in progress, cardholder already presented payment

Example Request (Hex)

Header	
Command Number	49
Data Length	06
Data	
Extended Data Offset	0000
Extended Command Number	0304
Complete Extended Data Length	0000
Extended Data	Not Applicable

Example Response (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	0000

Extended Data	Not Applicable

8.4.7 Extended Command 0x0305 - Modify Terminal Configuration (MAC)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

The host uses this command is used to directly modify tags in the device's EMV Terminal configuration. See **Extended Command 0x0306 - Read Terminal Configuration** and the Terminal Configuration subsections in **Appendix H EMV Terminal and Application Settings (EMV Only)**.

Some of the device's EMV Terminal configuration tags can only be set to EMV certified combinations. To change those settings, the host should use **Extended Command 0x0310 - Modify EMV Configuration (MAC, Contact Only)**.

(Fixed Key Only)
If the device is configured to use fixed key encryption using **Property 0x6B - Key Management Scheme (Fixed Key Only)** or the device's security level is less than 3, then MACing is not required. In this case, the Device Serial Number and MAC fields can be set all zeroes.

Configuration changes will be lost after a power cycle or reset unless the host sends **Extended Command 0x030E - Commit Configuration** after making all configuration changes.

Table 8-63 - Request Data for Extended Command 0x0305 - Modify Terminal Configuration (MAC)

Offset	Field Name	Value
0	Type of MAC	MAC algorithm designator 0x00 = ISO 9797 MAC Algorithm 3, Padding Method 1.
1	Slot Number	EMV Terminal Slot Number. Must be 0x01.
2	Operation	0x01 = Write Operation 0xFF = Set to Factory Defaults (sets all items, Terminal, Applications, and Application Public Keys to factory default values)
3	Database Selector	(Contact Only) 0x00 = EMV Contact L2 (Contactless Only) 0x01 = MCL 0x02 = payWave 0x03 = Expresspay 0x04 = D-PAS 0x05 = UnionPay QuickPass (QuickPass Support Only)
4..19	Device Serial Number (DSN)	16 Bytes DSN

20..n	Objects To Write	Note: Not needed if Operation is 0xFF Set to Factory Defaults.
		FA<len> /* container for generic data */
		  <tag><len><value>
		  …
		  <tag><len><value>

Offset	Field Name	Value
n..n+3	MAC	MAC computed on **Device Serial Number (DSN)** and **Objects to Write** fields. See section **8.4.1 About MACs**. (EMV Settings Unlock Only) If **Property 0x72 - EMV Configuration Security (EMV Settings Unlock Only)** is set to **OEM Behavior**, pad this field with zeroes.

Response Data: None. The response to this command only contains a result code.

Result codes:
0x0000 = Success
0x0390 = Device Has No Keys
0x0391 = Invalid Device Serial Number
0x0392 = Invalid Type of MAC field
0x0393 = Invalid Slot Number field
0x0394 = Invalid Operation field
0x0395 = Invalid Database Selector field
0x0396 = Invalid Objects to Write field
0x0397 = Invalid MAC
0x0399 = Object Write Protected

Example Request (Hex)

Header	
Command Number	49
Data Length	2A
Data	
Extended Data Offset	0000
Extended Command Number	0305
Complete Extended Data Length	0024
Extended Data	00010100nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnFA0A9F160731 323334353637xxxxxxxx Where nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn is the 16-byte Device Serial Number (DSN) and xxxxxxxx is the 4-byte MAC

Example Response (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	0000

Extended Result Code	0000
Complete Extended Data Length	0000
Extended Data	Not Applicable

8.4.8 Extended Command 0x0306 - Read Terminal Configuration

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

This command is used to read EMV Terminal configuration data. See **Extended Command 0x0305 - Modify Terminal Configuration (MAC)** and the Terminal Configuration subsections in **Appendix H EMV Terminal and Application Settings (EMV Only)**.

Table 8-64 - Request Data for Extended Command 0x0306 - Read Terminal Configuration

Offset	Field Name	Value
0	Slot Number	EMV Terminal Slot Number. Must be 0x01.
1	Operation	0x00 = Read Operation 0x0F = Read All Tags of selected slot
2	Database Selector	(Contact Only) 0x00 = EMV Contact L2 (Contactless Only) 0x01 = MCL 0x02 = payWave 0x03 = Expresspay 0x04 = D-PAS 0x05 = UnionPay QuickPass (QuickPass Support Only)

3..	Tags to Read	Note: Not needed if Operation is 0x0F Read All Tags of selected slot.
		
		FA<len> /* container for generic data */
		  <tag>
		  …
		  <tag>
		
		Tag DFDF47 cannot be read individually. This tag can only be retrieved using the ‘Read All Tags’ option.

Table 8-65 - Response Data for Extended Command 0x0306 - Read Terminal Configuration

Offset	Field Name	Value
0..1	Message Length	Two byte binary, most significant byte first. This gives the total length of the EMV Terminal Configuration message that follows.
2..	Tags Read	FA<len> /* container for generic data */ <tag><len><value> … <tag><len><value> When reading all tags for the selected slot, the last two tags are: DFDF26, the Configuration Label DFDF47, the Database Checksum

Result codes:
0x0000 = Success
0x0393 = Invalid Slot Number field
0x0394 = Invalid Operation field
0x0395 = Invalid Database Selector field
0x0396 = Invalid Tags to Read field

Example Request (Hex)

Header	
Command Number	49
Data Length	0D
Data	
Extended Data Offset	0000
Extended Command Number	0306
Complete Extended Data Length	0007
Extended Data	010000FA029F1A

Example Response (Hex)

Header	
Result Code	0A
Data Length	11
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	000B
Extended Data	0009FA8200059F1A020840

8.4.9 Extended Command 0x0307 - Modify Application Configuration (MAC)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

This command is used to modify EMV Application configurations. See **Extended Command 0x0308 - Read Application Configuration** and the Application Settings subsections in **Appendix H EMV Terminal and Application Settings (EMV Only)**.

(Fixed Key Only)
If the device is configured to use fixed key encryption using **Property 0x6B - Key Management Scheme (Fixed Key Only)** or the device's security level is less than 3, then MACing is not required. In this case, the Device Serial Number and MAC fields can be set all zeroes.

Configuration changes will be lost after a power cycle or reset unless the host sends **Extended Command 0x030E - Commit Configuration** after making all configuration changes.

Table 8-66 - Request Data for Extended Command 0x0307 - Modify Application Configuration (MAC)

Offset	Field Name	Value
0	Type of MAC	MAC algorithm designator 0x00 = ISO 9797 MAC Algorithm 3, Padding Method 1.
1	Slot Number	EMV Application Slot Number See **Appendix H EMV Terminal and Application Settings (EMV Only)** to determine how many application slots the device has for the selected database.
2	Operation	0x01 = Write Operation
3	Database Selector	(Contact Only) 0x00 = EMV Contact L2 (Contactless Only) 0x01 = MCL 0x02 = payWave 0x03 = Expresspay 0x04 = D-PAS 0x05 = UnionPay QuickPass (QuickPass Support Only) 0x06 = Apple VAS (Apple VAS Only)
4..19	Device Serial Number (DSN)	16 Bytes DSN

20..n	Objects to Write	FA<len> /* container for generic data */
		  <tag><len><value>
		  …
		  <tag><len><value>

Offset	Field Name	Value
n..n+3	MAC	MAC computed on **Device Serial Number (DSN)** and **Objects to Write** fields. See section **8.4.1 About MACs**. (EMV Settings Unlock Only) If **Property 0x72 - EMV Configuration Security (EMV Settings Unlock Only)** is set to **OEM Behavior**, pad this field with zeroes.

Response Data: None. The response to this command only contains a result code.

Result codes:
0x0000 = Success
0x0390 = Device Has No Keys
0x0391 = Invalid Device Serial Number
0x0392 = Invalid Type of MAC field
0x0393 = Invalid Slot Number field
0x0394 = Invalid Operation field
0x0395 = Invalid Database Selector field
0x0396 = Invalid Objects to Write field
0x0397 = Invalid MAC

Example Request (Hex)

Header	
Command Number	49
Data Length	27
Data	
Extended Data Offset	0000
Extended Command Number	0307
Complete Extended Data Length	0021
Extended Data	00010100nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnFA079F1B0400002710xxxxxxxx Where nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn is the 16-byte Device Serial Number (DSN) and xxxxxxxx is the 4-byte MAC

Example Response (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	0000

Extended Data	Not Applicable

8.4.10 Extended Command 0x0308 - Read Application Configuration

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

This command is used to read back EMV Application configurations. See **Extended Command 0x0307 - Modify Application Configuration (MAC)** and Appendix **H.2.2 EMV Contact Application Settings**.

Table 8-67 - Request Data for Extended Command 0x0308 - Read Application Configuration

Offset	Field Name	Value
0	Slot Number	EMV Application Slot Number See **Appendix H EMV Terminal and Application Settings (EMV Only)** to determine how many application slots the device has for the selected database.
1	Operation	0x00 = Read Operation 0x0F = Read All Tags of selected slot
2	Database Selector	(Contact Only) 0x00 = EMV Contact L2 (Contactless Only) 0x01 = MCL 0x02 = payWave 0x03 = Expresspay 0x04 = D-PAS 0x05 = UnionPay QuickPass (QuickPass Support Only) 0x06 = Apple VAS (Apple VAS Only)
3..	Tags to Read	Note: Not needed if Operation is 0x0F Read All Tags of selected slot. FA<len> /* container for generic data */ <tag> … <tag>

Table 8-68 - Response Data for Extended Command 0x0308 - Read Application Configuration

Offset	Field Name	Value
0	Message Length	Two byte binary, most significant byte first. This gives the total length of the EMV Application Configuration message that follows.
2..	Tags Read	FA<len> /* container for generic data */ <tag><len><value> … <tag><len><value>

Result codes:
0x0000 = Success
0x0393 = Invalid Slot Number field
0x0394 = Invalid Operation field
0x0395 = Invalid Database Selector field
0x0396 = Invalid Tags to Read field

Example Request (Hex)

Header	
Command Number	49
Data Length	0D
Data	
Extended Data Offset	0000
Extended Command Number	0308
Complete Extended Data Length	0007
Extended Data	010000FA029F06

Example Response (Hex)

Header	
Result Code	0A
Data Length	15
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	000F
Extended Data	000DFA8200099F0606A00000002501

8.4.11 Extended Command 0x0309 - Modify Acquirer Public Key CAPK (MAC, EMV ODA Only)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

This command is used to modify CA Public Keys, which are specified by each of the payment brands and which the device can use to perform offline data authentication (ODA) to authenticate data from a chip card or contactless card or payment device on its own, in cases where network access to a payment processor is not available. See **Extended Command 0x030A - Read Acquirer Public Key CAPK (EMV ODA Only)** for details about storage of keys.

(Fixed Key Only)
If the device is configured to use fixed key encryption using **Property 0x6B - Key Management Scheme (Fixed Key Only)** or the device's security level is less than 3, then MACing is not required. In this case, the Device Serial Number and MAC fields can be set all zeroes.

Configuration changes will be lost after a power cycle or reset unless the host sends **Extended Command 0x030E - Commit Configuration** after making all configuration changes.

Table 8-69 - Request Data for Extended Command 0x0309 - Modify Acquirer Public Key CAPK (MAC, EMV ODA Only)

Offset	Field Name	Value
0	Type of MAC	MAC algorithm designator 0x00 = MSV5 MSCI CBC-MAC
1	Slot Number	CA Public Key Slot Number = Any value from 0x01 to 0x33 inclusive 0xFF = Next Available (slot with RID TLV length set to zero) If the Operation field is set to Erase All, this field is not used and can be set to any value.
2	Operation	0x00 = Erase All (Erases all tags in all CAPK slots). This sets the TLV length of every TLV data object in each slot to 1 and the value to 0. A slot is considered erased and available for use by the Next Available Slot Number (0xFF) if its RID TLV length is set to 1 and its value is set to 0. 0x01 = Writes a CA Public Key. To erase a single slot, write all of the slot's tags' TLV lengths to 1 and values to 0.
3	Database Selector	(Contact Only) 0x00 = EMV Contact L2 (Contactless Only) 0x01 = MCL 0x02 = payWave 0x03 = Expresspay 0x04 = D-PAS 0x05 = UnionPay QuickPass (QuickPass Support Only)
4..19	Device Serial Number (DSN)	16 Bytes DSN

Offset	Field Name	Value
20..n	Objects to Write	Note: Not needed if Operation is 0x00 Erase All. FA<len> /* container for generic data */ < DFDF79><len><value> /* RID */ < DFDF7A><len><value>/* Index */ < DFDF7B><len><value>/* Modulus */ < DFDF7C><len><value>/* Key Exponent */ < DFDF7D><len><value> /* Checksum */
n..n+3	MAC	MAC computed on **Device Serial Number (DSN)** and **Objects to Write** fields. See section **8.4.1 About MACs**. (EMV Settings Unlock Only) If **Property 0x72 - EMV Configuration Security (EMV Settings Unlock Only)** is set to **OEM Behavior**, pad this field with zeroes.

Table 8-70 - Response Data for Extended Command 0x0309 - Modify Acquirer Public Key CAPK (MAC, EMV ODA Only)

Offset	Field Name	Value
0	Slot Number	When the Next Available slot number (0xFF) is used, this field returns the next available slot used if successful, otherwise this field is omitted.

Result codes:
0x0000 = Success
0x0390 = Device Has No Keys
0x0391 = Invalid Device Serial Number
0x0392 = Invalid Type of MAC field
0x0393 = Invalid Slot Number field
0x0394 = Invalid Operation field
0x0395 = Invalid Database Selector field
0x0396 = Invalid Objects to Write field
0x0397 = Invalid MAC
0x0398 = No Slots Available
0x039B = Invalid CAPK Checksum

Example Request (Hex)

Header	
Command Number	49
Data Length	39
Data	
Extended Data Offset	0000
Extended Command Number	0309
Complete Extended Data Length	0033

Extended Data	00010100nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnFA19DFDF790100DFDF7A0100DFDF7B0100DFDF7C0100DFDF7D0100xxxxxxxx Where nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn is the 16-byte Device Serial Number and xxxxxxxx is the 4-byte MAC

Example Response (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	0000
Extended Data	Not Applicable

8.4.12 Extended Command 0x030A - Read Acquirer Public Key CAPK (EMV ODA Only)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

This command is used to read back CA Public Keys. For details about the purpose of these keys, see **Extended Command 0x0309 - Modify Acquirer Public Key CAPK (MAC, EMV ODA Only)**.

Each CAPK database contains up to 51 key slots, each formatted as shown in **Table 8-71**.

Table 8-71 - Certificate Authority Public Key Slots 1 to 51 Data

Tag	Value (hex)	Length (bytes)	Max Length	Description
DFDF79	00	0x01	0x05	CA Public Key RID
DFDF7A	00	0x01	0x01	CA Public Key Index
DFDF7B	00	0x01	0xF8	CA Public key Modulus
DFDF7C	00	0x01	0x03	CA Public Key Exponent
DFDF7D	00	0x01	0x14	CA Public Key Checksum

Table 8-72 - Request Data for Extended Command 0x030A - Read Acquirer Public Key CAPK (EMV ODA Only)

Offset	Field Name	Value
0	Slot Number	CA Public Key Slot Number = Any value from 0x01 to 0x33 inclusive
1	Operation	0x00 = Read Operation 0x0F = Read All Tags of selected slot
2	Database Selector	(Contact Only) 0x00 = EMV Contact L2 (Contactless Only) 0x01 = MCL 0x02 = payWave 0x03 = Expresspay 0x04 = D-PAS 0x05 = UnionPay QuickPass (QuickPass Support Only)
3..	Tags to Read	Note: Not needed if Operation is 0x0F Read All Tags of selected slot. FA<len> /* container for generic data */ <tag> … <tag>

Table 8-73 - Request Data for Extended Command 0x030A - Read Acquirer Public Key CAPK (EMV ODA Only)

Offset	Field Name	Value
0..1	Message Length	Two byte binary, most significant byte first. This gives the total length of the message that follows.
2..	Tags Read	FA<len> /* container for generic data */ <tag><len><value> … <tag><len><value>

Result codes:
0x0000 = Success
0x0393 = Invalid Slot Number field
0x0394 = Invalid Operation field
0x0395 = Invalid Database Selector field
0x0396 = Invalid Tags to Read field

Example Request (Hex)

Header	
Command Number	49
Data Length	09
Data	
Extended Data Offset	0000
Extended Command Number	030A
Complete Extended Data Length	0003
Extended Data	010F00

Example Response (Hex)

Header	
Result Code	0A
Data Length	25
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	001F
Extended Data	001DFA820019DFDF790100DFDF7A0100DFDF7B0100DFDF7C0100DFDF7D0100

8.4.13 Extended Command 0x030B - Read EMV Kernel Information

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

This command is used to read kernel information.

Table 8-74 - Request Data for Extended Command 0x030B - Read EMV Kernel Information

Offset	Field Name	Value
0	Mode	(Contact Only) 0x01 = Version of EMV Contact L2 Kernel 0x11 = Checksum of EMV Contact L2 Kernel 0x12 = Checksum of EMV Contact L2 Configuration (Contactless Only) 0x02 = Version of payWave Kernel 0x21 = Checksum of payWave Kernel 0x22 = Checksum of payWave Kernel Configuration 0x03 = Version of MCL Kernel 0x31 = Checksum of MCL Kernel 0x32 = Checksum of MCL Kernel Configuration 0x04 = Version of ExpressPay Kernel 0x41 = Checksum of ExpressPay Kernel 0x42 = Checksum of ExpressPay Kernel Configuration 0x05 = Version of D-PAS Kernel 0x51 = Checksum of D-PAS Kernel 0x52 = Checksum of D-PAS Kernel Configuration (QuickPass Support Only) 0x06 = Version of UnionPay QuickPass Kernel 0x61 = Checksum of UnionPay QuickPass Kernel 0x62 = Checksum of UnionPay QuickPass Kernel Configuration 0x07 = Version of Apple VAS

Table 8-75 - Response Data for Extended Command 0x030B - Read EMV Kernel Information

Offset	Field Name	Value
0	Response Data	Requested kernel version or checksum. The kernel version is a human-readable string describing the kernel and its version, and the checksums are 40-character hexadecimal strings.

0x0000 = Success
0x0386 = Invalid Mode

Example Request (Hex)

Header	
Command Number	49
Data Length	07
Data	
Extended Data Offset	0000
Extended Command Number	030B
Complete Extended Data Length	0001
Extended Data	01

Example Response (Hex)

Header	
Result Code	0A
Data Length	1E
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	0018
Extended Data	6544796E616D6F204C32204B65726E656C20526576204135 (eDynamo L2 Kernel Rev A5)

8.4.14 Extended Command 0x030C - Set Date and Time (MAC)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

This command is used to set the device's date and time. See **Extended Command 0x030D - Read Date and Time**.

Devices with a battery-backed real time clock (see **Table 1-2 - Device Features**) have the date and time set by the manufacturer, so this command may not need to be used after that. Devices that do not have a battery-backed real time clock must use this command frequently because (a) the clock must be set before the device can process EMV transactions, and (b) the host software must use this command every time the device is power cycled or reset.

Table 8-76 - Request Data for Extended Command 0x030C - Set Date and Time (MAC)

Offset	Field Name	Value
0	Type of MAC	MAC algorithm designator 0x00 = ISO 9797 MAC Algorithm 3, Padding Method 1.
1..16	Device Serial Number	16 Bytes Device Serial Number. The host can set this field to all zeroes, except when using these devices: • eDynamo with firmware part number ***1000003354*** revisions earlier than **G02** • eDynamo with firmware part number ***1000002649***
17	Month	Value from 0x01..0x0C
18	Day	Value from 0x01..0x1F (less depending on month)
19	Hour	Value from 0x00..0x17
20	Minute	Value from 0x00..0x3B
21	Second	Value from 0x00..0x3B
22	Unused	Value from 0x00..0x06
23	Year	Value from 0x00 (2008)..0x44 (2076)
24..27	MAC	MAC computed over all preceding fields except **Type of MAC**. The host can set this field to all zeroes, except when using these devices: • eDynamo with firmware part number ***1000003354*** revisions earlier than G02 • eDynamo with firmware part number ***1000002649***

Response Data: None. The response to this command only contains a result code.

Result codes:
0x0000 = Success
0x0390 = Device Has No Keys
0x0391 = Invalid Device Serial Number
0x0392 = Invalid Type of MAC field
0x0396 = Invalid Date / Time data

0x0397 = Invalid MAC

Example Request (Hex)

Header	
Command Number	49
Data Length	22
Data	
Extended Data Offset	0000
Extended Command Number	030C
Complete Extended Data Length	001C
Extended Data	0000000000000000000000000000000000021C0F380B0009xxxxxxxx Where xxxxxxxx is the 4-byte MAC

Example Response (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	0000
Extended Data	Not Applicable

8.4.15 Extended Command 0x030D - Read Date and Time

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

The host uses this command to get the date / time from the device's internal clock. See **Extended Command 0x030C - Set Date and Time (MAC)**.

Request Data: None

Table 8-77 - Response Data for Extended Command 0x030D - Read Date and Time

Offset	Field Name	Value
0	Month	Value from 0x01..0x0C
1	Day	Value from 0x01..0x1F (less depending on month)
2	Hour	Value from 0x00..0x17
3	Minute	Value from 0x00..0x3B
4	Second	Value from 0x00..0x3B
5	Unused	0x00
6	Year	Value from 0x00 (2008)..0xFF (2263)

Result codes:
0x0000 = Success
0x0396 = Invalid Date / Time data (Date / Time has not been set yet)

Example Request (Hex)

Header	
Command Number	49
Data Length	06
Data	
Extended Data Offset	0000
Extended Command Number	030D
Complete Extended Data Length	0000
Extended Data	Not Applicable

Example Response (Hex)

Header	
Result Code	0A
Data Length	0D
Data	
Extended Data Offset	0000

Extended Result Code	0000
Complete Extended Data Length	0007
Extended Data	0204130D340009

8.4.16 Extended Command 0x030E - Commit Configuration

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

This command is used to commit configuration changes to non-volatile memory so they remain in place after a power cycle or reset. If this command is not sent after changing the configuration, the changes are lost on power cycle or reset.

Because non-volatile memory has limited erase/write cycles, the host should send this command after all configuration changes have been made. It should not be sent after each configuration change out of many.

Table 8-78 - Request Data for Extended Command 0x030E - Commit Configuration

Offset	Field Name	Value
0	Database Selector	(Contact Only) 0x00 = EMV Contact L2 (Contactless Only) 0x01 = MCL 0x02 = payWave 0x03 = Expresspay 0x04 = D-PAS 0x05 = UnionPay QuickPass (QuickPass Support Only) 0x06 = Apple VAS (Apple VAS Only)

Response Data: None

Result codes:
0x0000 = Success
0x0001 = Failure
0x0395 = Invalid Database Selector field

Example Request (Hex)

Header	
Command Number	49
Data Length	07
Data	
Extended Data Offset	0000
Extended Command Number	030E
Complete Extended Data Length	0001
Extended Data	00

Example Response (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	0000
Extended Data	Not Applicable

8.4.17 Extended Command 0x0310 - Modify EMV Configuration (MAC, Contact Only)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

The host uses this command to select a set of predetermined allowable values for the EMV configuration tags marked as only settable as part of Terminal Configuration in **Appendix H.2.1 EMV Contact Terminal Settings and Defaults (Contact Only,** not 4.3i Format). These values can not be set directly, they must be set to one of a specified set of values, selected from the list of **Vendor Config IDs** in the device's ***Letter of Approval for Contact Level 2*** posted in the list of ***Approved / Evaluated*** products on the EMVCo web site. Detailed descriptions of the tags set by this command can be found in ***EMV Integrated Circuit Card Specifications for Payment Systems v4.3***.

Separate from these values, the host may set unrestricted tags directly using **Extended Command 0x0305 - Modify Terminal Configuration (MAC)** and **Extended Command 0x0307 - Modify Application Configuration (MAC)**.

(Fixed Key Only)
If the device is configured to use fixed key encryption using **Property 0x6B - Key Management Scheme (Fixed Key Only)** or the device's security level is less than 3, then MACing is not required. In this case, the Device Serial Number and MAC fields can be set all zeroes.

Configuration changes will be lost after a power cycle or reset unless the host sends **Extended Command 0x030E - Commit Configuration** after making all configuration changes.

Table 8-79 - Request Data for Extended Command 0x0310 - Modify EMV Configuration (MAC, Contact Only)

Offset	Field Name	Value
0	Type of MAC	MAC algorithm designator 0x00 = ISO 9797 MAC Algorithm 3, Padding Method 1. (4 byte MAC)
1	Database Selector	0x00 = EMV Contact L2
2..17	Device Serial Number	16 Bytes DSN
18	Configuration ID	One byte field that specifies one of the following configurations. Each device implements a subset of this standard list; the supported subset is specified in the device's EMVCo Letter of Approval (LoA) as **Vendor Config ID**s: 0x00 = Vendor Config ID **C1** • Attended, Online Only • SDA, DDA and CDA enabled • No MSR Fallback • Signature, No CVM Required • Goods, Services, Cashback, Payment • Print Attendant, Display Attendant/Cardholder, Code Table 1 • Tag 9F35 set to 21 • Tag 9F33 set to 20 28 C8 • Tag 9F40 set to 72 00 00 B0 01

Offset	Field Name	Value
		0x01 = Vendor Config ID **C2** • Attended, Online Only • SDA, DDA and CDA disabled • No MSR Fallback, Signature • No CVM Required • Goods, Services, Cashback, Payment • Print Attendant, Display Attendant/Cardholder, Code Table 1 • Tag 9F35 = 21 • Tag 9F33 = 20 28 00 • Tag 9F40 = 72 00 00 B0 01 0x02 = Vendor Config ID **C3** • Attended, Offline/Online • SDA, DDA and CDA enabled • No MSR Fallback • Signature, No CVM Required • Goods, Services, Cashback, Payment • Print Attendant, Display Attendant/Cardholder, Code Table 1 • Tag 9F35 = 22 • Tag 9F33 = 20 28 C8 • Tag 9F40 = 72 00 00 B0 01 0x03 = Vendor Config ID **C4** • Attended, Online Only • SDA, DDA and CDA enabled • With MSR Fallback • Signature, No CVM Required • Cash, Goods, Services, Cashback, Inquiry, Transfer, Payment, Administrative, Cash Deposit • Numeric, Alphabetic, Special, Command and Function keys • Print Attendant, Display Attendant, Code Table 1 • Tag 9F35 = 21 • Tag 9F33 = 60 28 C8 • Tag 9F40 = FF 80 F0 A0 01 0x04 = Vendor Config ID **C5** • Unattended, Online Only • SDA, DDA and CDA enabled • With MSR Fallback • No CVM Required • Cash, Goods, Services, Cashback, Inquiry, Transfer, Payment, Administrative, Cash Deposit. • Numeric, Alphabetic, Special, Command and Function keys • Print Cardholder, Display Cardholder, Code Table 1 • Tag 9F35 = 24 • Tag 9F33 = 60 08 C8

Offset	Field Name	Value
		• Tag 9F40 = FF 80 F0 50 01 0x05 = Vendor Config ID **C6** • Attended, Online Only • SDA, DDA and CDA disabled • With MSR Fallback • Signature, No CVM Required • Cash, Goods, Services, Cashback, Inquiry, Transfer, Payment, Administrative, Cash Deposit. • Numeric, Alphabetic, Special, Command and Function keys • Print Attendant, Display Attendant, Code Table 1 • Tag 9F35 = 21 • Tag 9F33 = 60 28 00 • Tag 9F40 = FF 80 F0 A0 01 0x06 = Vendor Config ID **C7** • Unattended, Online Only • SDA, DDA and CDA disabled • With MSR Fallback • No CVM Required • Cash, Goods, Services, Cashback, Inquiry, Transfer, Payment, Administrative, Cash Deposit. • Numeric, Alphabetic, Special, Command and Function keys • Print Cardholder, Display Cardholder, Code Table 1 • Tag 9F35 = 24 • Tag 9F33 = 60 08 00 • Tag 9F40 = FF 80 F0 50 01 0x07 = Vendor Config ID **C8** • Attended, Online Only • SDA, DDA and CDA enabled • Manual Key Entry, With MSR Fallback • PIN, Signature, No CVM Required • Cash, Goods, Services, Cashback, Inquiry, Transfer, Payment, Administrative, Cash Deposit. • Numeric, Alphabetic, Special, Command and Function keys • Print Attendant, Display Attendant, Code Table 1 • Tag 9F35 = 21 • Tag 9F33 = E0 F8 C8 • Tag 9F40 = FF 80 F0 A0 01 0x08 = Vendor Config ID **C9** • Unattended, Online Only • SDA, DDA and CDA enabled • Manual Key Entry, With MSR Fallback • PIN, No CVM Required

Offset	Field Name	Value
		• Cash, Goods, Services, Cashback, Inquiry, Transfer, Payment, Administrative, Cash Deposit. • Numeric, Alphabetic, Special, Command and Function keys • Print Cardholder, Display Cardholder, Code Table 1 • Tag 9F35 = 24 • Tag 9F33 = E0 D8 C8 • Tag 9F40 = FF 80 F0 50 01 0x09 = Vendor Config ID **C10** • Attended, Online Only • SDA, DDA and CDA disabled • Manual Key Entry, With MSR Fallback • PIN, Signature, No CVM Required • Cash, Goods, Services, Cashback, Inquiry, Transfer, Payment, Administrative, Cash Deposit. • Numeric, Alphabetic, Special, Command and Function keys • Print Attendant, Display Attendant, Code Table 1 • Tag 9F35 = 21 • Tag 9F33 = E0 F8 00 • Tag 9F40 = FF 80 F0 A0 01 0x0A = Vendor Config ID **C11** • Unattended, Online Only • SDA, DDA and CDA disabled • Manual Key Entry, With MSR Fallback • PIN, No CVM Required • Cash, Goods, Services, Cashback, Inquiry, Transfer, Payment, Administrative, Cash Deposit. • Numeric, Alphabetic, Special, Command and Function keys • Print Cardholder, Display Cardholder, Code Table 1 • Tag 9F35 = 24 • Tag 9F33 = E0 D8 00 • Tag 9F40 = FF 80 F0 50 01 0x0B = Vendor Config ID **C12** • Attended, Online Only • SDA, DDA and CDA disabled • Manual Key Entry, With MSR Fallback • Signature, No CVM Required • Cash, Goods, Services, Cashback, Inquiry, Transfer, Payment, Administrative, Cash Deposit. • Numeric, Alphabetic, Special, Command and Function keys • Print Attendant, Display Attendant, Code Table 1 • Tag 9F35 = 21 • Tag 9F33 = E0 28 00 • Tag 9F40 = FF 80 F0 A0 01

Offset	Field Name	Value
		0x0C = Vendor Config ID **C13** • Unattended, Online Only • SDA, DDA and CDA disabled • Manual Key Entry, With MSR Fallback • No CVM Required • Cash, Goods, Services, Cashback, Inquiry, Transfer, Payment, Administrative, Cash Deposit. • Numeric, Alphabetic, Special, Command and Function keys • Print Cardholder, Display Cardholder, Code Table 1 • Tag 9F35 = 24 • Tag 9F33 = E0 08 00 • Tag 9F40 = FF 80 F0 50 01 0x0D = Vendor Config ID **C14** • Unattended, Online Only • SDA, DDA and CDA disabled • No MSR Fallback • No CVM Required • Cash, Goods, Services, Payment. • Print Cardholder, Display Cardholder, Code Table 1 • Tag 9F35 = 24 • Tag 9F33 = 20 08 00 • Tag 9F40 = E2 00 00 50 01 0x0E = Vendor Config ID **C15** • Unattended, Offline/Online • SDA, DDA and CDA enabled • No MSR Fallback • No CVM Required • Cash, Goods, Services, Payment. • Print Cardholder, Display Cardholder, Code Table 1 • Tag 9F35 = 25 • Tag 9F33 = 20 08 C8 • Tag 9F40 = E2 00 00 50 01
19..22	MAC	MAC computed on **Device Serial Number (DSN)** and **Configuration Identifier** fields. See section **8.4.1 About MACs**.

Response Data: None. The response to this command only contains a result code.

Result codes:
0x0000 = Success
0x0390 = Device Has No Keys
0x0391 = Invalid Device Serial Number
0x0392 = Invalid Type of MAC field
0x0393 = Invalid Slot Number field
0x0395 = Invalid Database Selector field
0x0397 = Invalid MAC

0x039C = Invalid Configuration Identifier

Example Request (Hex)

Header	
Command Number	49
Data Length	1D
Data	
Extended Data Offset	0000
Extended Command Number	0310
Complete Extended Data Length	0017
Extended Data	0000nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn01xxxxxxxx Where nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn is the 16-byte Device Serial Number (DSN) and xxxxxxxx is the 4-byte MAC

Example Response (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	0000
Extended Data	Not Applicable

8.4.18 Extended Command 0x0311 - Read EMV Configuration (Contact Only)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

The host uses this command to read which contact EMV configuration the device us using. For details, see **Extended Command 0x0310 - Modify EMV Configuration (MAC, Contact Only)**.

Table 8-80 - Request Data for Extended Command 0x0311 - Read EMV Configuration (Contact Only)

Offset	Field Name	Value
0	Database Selector	0x00 = EMV Contact L2

Table 8-81 - Response Data for Extended Command 0x0311 - Read EMV Configuration (Contact Only)

Offset	Field Name	Value
0	Configuration Identifier	One byte field containing the Configuration ID that was set using **Extended Command 0x0310 - Modify EMV Configuration (MAC, Contact Only)**.

Result codes:
0x0000 = Success
0x0395 = Invalid Database Selector field

Example Request (Hex)

Header	
Command Number	49
Data Length	07
Data	
Extended Data Offset	0000
Extended Command Number	0311
Complete Extended Data Length	0001
Extended Data	00

Example Response (Hex)

Header	
Result Code	0A
Data Length	07
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	0001
Extended Data	01

8.4.19 Extended Command 0x0312 - Modify Dynamic Reader Limits Configuration (MAC, Contactless Only)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

This command is used to modify EMV Dynamic Reader Limit configurations. See **Extended Command 0x0313 - Read Dynamic Reader Limits Configuration (Contactless Only)** and the application settings for the relevant payment brand(s) in **Appendix H EMV Terminal and Application Settings (EMV Only)**.

(Fixed Key Only)
If the device is configured to use fixed key encryption using **Property 0x6B - Key Management Scheme (Fixed Key Only)** or the device's security level is less than **Security Level 3**, then MACing is not required. In this case, the Device Serial Number and MAC fields can be set all zeroes.

Configuration changes will be lost after a power cycle or reset unless the host sends **Extended Command 0x030E - Commit Configuration** after making all configuration changes.

Table 8-82 - Request Data for Extended Command 0x0312 - Modify Dynamic Reader Limits Configuration (MAC, Contactless Only)

Offset	Field Name	Value
0	Type of MAC	MAC algorithm designator 0x00 = ISO 9797 MAC Algorithm 3, Padding Method 1.
1	Slot Number	EMV Dynamic Reader Limit Slot Number See **Appendix H EMV Terminal and Application Settings (EMV Only)** to determine how many Dynamic Reader Limit slots the device has for the selected database.
2	Operation	0x01 = Write Operation
3	Database Selector	0x01 = MCL (Reserved because MasterCard does not use DRL) 0x02 = payWave 0x03 = Expresspay 0x04 = D-PAS (Reserved because D-PAS does not use DRL) 0x05 = UnionPay QuickPass (QuickPass Support Only, Reserved because UnionPay does not use DRL)
4..19	Device Serial Number (DSN)	16 Bytes DSN
20..n	Objects to Write	FA<len> /* container for generic data */ <tag><len><value> … <tag><len><value>

Offset	Field Name	Value
n..n+3	MAC	MAC computed on **Device Serial Number (DSN)** and **Objects to Write** fields. See section **8.4.1 About MACs**. (EMV Settings Unlock Only) If **Property 0x72 - EMV Configuration Security (EMV Settings Unlock Only)** is set to **OEM Behavior**, pad this field with zeroes.

Response Data: None. The response to this command only contains a result code.

Result codes:
0x0000 = Success
0x0390 = Device Has No Keys
0x0391 = Invalid Device Serial Number
0x0392 = Invalid Type of MAC field
0x0393 = Invalid Slot Number field
0x0394 = Invalid Operation field
0x0395 = Invalid Database Selector field
0x0396 = Invalid Objects to Write field
0x0397 = Invalid MAC

Example Request (Hex)

Header	
Command Number	49
Data Length	24
Data	
Extended Data Offset	0000
Extended Command Number	0312
Complete Extended Data Length	001E
Extended Data	00010102nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnFA049F5A0102 xxxxxxxx Where nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn is the 16-byte Device Serial Number (DSN) and xxxxxxxx is the 4-byte MAC

Example Response (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	0000

Extended Data	Not Applicable

8.4.20 Extended Command 0x0313 - Read Dynamic Reader Limits Configuration (Contactless Only)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

This command is used to read back EMV Dynamic Reader Limit configurations. See **Extended Command 0x0312 - Modify Dynamic Reader Limits Configuration (MAC, Contactless Only)** and the application settings for the relevant payment brand(s) in **Appendix H EMV Terminal and Application Settings (EMV Only)**.

Table 8-83 - Request Data for Extended Command 0x0313 - Read Dynamic Reader Limits Configuration (Contactless Only)

Offset	Field Name	Value
0	Slot Number	EMV Dynamic Reader Limit Slot Number See **Appendix H EMV Terminal and Application Settings (EMV Only)** to determine how many Dynamic Reader Limit slots the device has for the selected database.
1	Operation	0x00 = Read Operation 0x0F = Read All Tags of selected slot
2	Database Selector	0x01 = MCL (Reserved) 0x02 = payWave 0x03 = Expresspay 0x04 = D-PAS (Reserved) 0x05 = UnionPay QuickPass (QuickPass Support Only, Reserved because UnionPay does not use DRL)

3..	Tags to Read	Note: Not needed if Operation is 0x0F Read All Tags of selected slot.
		
		FA<len> /* container for generic data */
		  <tag>
		  …
		  <tag>

Table 8-84 - Response Data for Extended Command 0x0313 - Read Dynamic Reader Limits Configuration (Contactless Only)

Offset	Field Name	Value
0	Message Length	Two byte binary, most significant byte first. This gives the total length of the EMV Application Configuration message that follows.

2..	Tags Read	FA<len> /* container for generic data */
		  <tag><len><value>
		  …
		  <tag><len><value>

Result codes:
0x0000 = Success
0x0393 = Invalid Slot Number field

0x0394 = Invalid Operation field
0x0395 = Invalid Database Selector field
0x0396 = Invalid Tags to Read field

Example Request (Hex)

Header	
Command Number	49
Data Length	0D
Data	
Extended Data Offset	0000
Extended Command Number	0313
Complete Extended Data Length	0007
Extended Data	010002FA029F5A

Example Response (Hex)

Header	
Result Code	0A
Data Length	10
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	000A
Extended Data	0008FA8200049F5A0101

8.4.21 Extended Command 0x0314 - Get EMV Transaction Status (Infinite Transaction Timeout Only)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

The host uses this command to determine whether the device is in the midst of processing an EMV transaction the host initiated using **Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)**.

Request Data: None

Response Data: None

Result codes:
0x038F = EMV Transaction In Progress
0x038D = No EMV Transaction In Progress

Example Request (Hex)

Header	
Command Number	49
Data Length	06
Data	
Extended Data Offset	0000
Extended Command Number	0314
Complete Extended Data Length	0000
Extended Data	Not Applicable

Example Response (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	0000
Extended Result Code	038F
Complete Extended Data Length	0000
Extended Data	Not Applicable

8.4.22 Extended Command 0x0315 - Get Overall Contactless Checksum (Comprehensive Checksums Only)

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

The host can use this command to retrieve a CRC32 checksum the device calculates across all of its contactless databases, which the host can compare to the checksum from a "gold" / baseline / known-good device to ensure that any given device contains the desired configuration. The device recalculates this value when the host uses **Extended Command 0x0305 - Modify Terminal Configuration (MAC)**, **Extended Command 0x0307 - Modify Application Configuration (MAC)**, or **Extended Command 0x0309 - Modify Acquirer Public Key CAPK (MAC, EMV ODA Only)** to load settings into any contactless database.

Result codes:
0x0000 = Success

Example Request (Hex)

Header	
Command Number	49
Data Length	06
Data	
Extended Data Offset	0000
Extended Command Number	0315
Complete Extended Data Length	0000
Extended Data	N/A

Example Response (Hex)

Header	
Result Code	0A
Data Length	0A
Data	
Extended Data Offset	0000
Extended Result Code	0000
Complete Extended Data Length	0004
Extended Data	9F5A0101

8.5 Command Group 0x04 - Auxiliary UART (Auxiliary Ports Only, Extended Commands Only)

The host uses the commands in this section to control its auxiliary UART port, which enables customers to implement solutions where an external UART device, such as magnetic stripe reader or contactless reader, can communicate with the host by piggybacking on the device's connection to the host.

If the device is not connected to the host using UART, the device automatically converts between UART and the device/host connection type. This conversion may introduce data propagation delays and buffering limitations. MagTek advises thoroughly testing external devices with the auxiliary port to make sure they are completely compatible.

To configure the behavior of the auxiliary UART port, use **Property 0x69 - Auxiliary UART Configuration (Auxiliary Ports Only)**.

8.5.1 Extended Command 0x0400 - Auxiliary UART Transmit Data

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

The host uses this command to transmit data to the device's auxiliary UART. When the external UART device sends data to the device, the device passes the data to the host using **Notification 0x0400 - Auxiliary UART Received Data**.

Table 8-85 - Request Data for Extended Command 0x0400 - Auxiliary UART Transmit Data

Offset	Field Name	Value
0	Port Identifier	The identifier of the port to transmit on. Always set this field to zero.
1..n	Transmit Data	The data to transmit. If the length of the data to be transmitted exceeds 1023 bytes, the host must transmit it using multiple commands.

Response Data: None

Result codes:
0x0000 = Success
0x0481 = Port not opened

Example Request (Hex)

Header	
Command Number	49
Data Length	0A
Data	
Extended Data Offset	00 00
Extended Command Number	04 00
Complete Extended Data Length	00 04
Extended Data	00 31 32 33

Example Response (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	00 00
Extended Result Code	00 00
Complete Extended Data Length	00 00
Extended Data	Not Applicable

8.5.2 Extended Command 0x0401 - Auxiliary UART Control

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

The host uses this command to control the device's auxiliary UART.

Table 8-86 - Request Data for Extended Command 0x0401 - Auxiliary UART Control

Offset	Field Name	Value
0	Port Identifier	The identifier of the port to transmit on. Always set this field to zero.
1	Control Data	The control data. If the host omits this field, the device returns the existing control data field to the host in the response data. See the Control Data Field table below for a description of each bit in this field.

Control Data Field: (Bit 0 is the least significant bit)

Bit	Field Name	Value
0	Open	0 = Port is closed. Ports power output is off, port signal lines will float, and data can not be transmitted or received on the port. 1 = Port is open.
1	DTR Level	0 = DTR output signal is set low when the port is open. 1 = DTR output signal is set high when the port is open.
2	RTS Level	0 = RTS output signal is set low when the port is open. 1 = RTS output signal is set high when the port is open.
3	DSR Level	0 = DSR input signal is read low when the port is open. 1 = DSR input signal is read high when the port is open. This bit always reads zero when the port is closed. This bit is read-only.
4	CTS Level	0 = CTS input signal is read low when the port is open. 1 = CTS input signal is read high when the port is open. This bit always reads zero when the port is closed. This bit is read only.
5..7	Reserved	These bits should always be written as zeroes.

Table 8-87 - Response Data for Extended Command 0x0401 - Auxiliary UART Control

Offset	Field Name	Value
0	Control Data	The control data. If the host transmitted Control Data in the request data, this field is not returned in the response data, otherwise the existing control data is returned in this field. See the Control Data Field table above for a description of each bit in this field.

Result codes:
0x0000 = Success

Example Request (Hex)

Header	
Command Number	49
Data Length	08
Data	
Extended Data Offset	00 00
Extended Command Number	04 01
Complete Extended Data Length	00 02
Extended Data	00 01

Example Response (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	00 00
Extended Result Code	00 00
Complete Extended Data Length	00 00
Extended Data	Not Applicable

8.6 Command Group 0x05 - Auxiliary SPI (Auxiliary Ports Only, Extended Commands Only)

The host uses the commands in this section to control its auxiliary SPI port, which enables customers to implement solutions where an external SPI device, such as magnetic stripe reader or contactless reader, can communicate with the host by piggybacking on the device's connection to the host.

If the device is not connected to the host using SPI, the device automatically converts between SPI and the device/host connection type. This conversion may introduce data propagation delays and buffering limitations. MagTek advises thoroughly testing external devices with the auxiliary port to make sure they are completely compatible.

To configure the behavior of the auxiliary SPI port, use **Property 0x6A - Auxiliary SPI Configuration (Auxiliary Ports Only)**.

8.6.1 Extended Command 0x0500 - Auxiliary SPI Transmit and Receive Data

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

The host uses this command to transmit and receive data to communicate with an external SPI device.

The host calls this command when it needs to send data to the auxiliary SPI device, and should also call this command to read data from the auxiliary SPI port if it receives **Notification 0x0500 - Auxiliary SPI Data Change**.

Table 8-88 - Request Data for Extended Command 0x0500 - Auxiliary SPI Transmit and Receive Data

Offset	Field Name	Value
0	Port Identifier	The identifier of the port to transmit on. Always set this field to zero.
1..n	Transmit Data	The data to transmit. If the transmit data length exceeds 1023 bytes it must be sent using multiple commands.

Table 8-89 - Response Data for Extended Command 0x0500 - Auxiliary SPI Transmit and Receive Data

Offset	Field Name	Value
0..n	Receive Data	The data received. Since, for SPI, a byte is always received when a byte is transmitted, the number of bytes received is equal to the number of bytes transmitted.

Result codes:
0x0000 = Success
0x0581 = Port not opened
0x0582 = Data length too large

Example Request (Hex)

Header	
Command Number	49
Data Length	0A

Data	
Extended Data Offset	00 00
Extended Command Number	05 00
Complete Extended Data Length	00 04
Extended Data	00 FF FF FF

Example Response (Hex)

Header	
Result Code	0A
Data Length	09
Data	
Extended Data Offset	00 00
Extended Result Code	00 00
Complete Extended Data Length	00 03
Extended Data	FF FF FF

8.6.2 Extended Command 0x0501 - Auxiliary SPI Control

Like all extended commands, the host initiates this command by calling **Command 0x49 - Send Extended Command Packet (Extended Commands Only)**, and receives a response as documented there.

The host uses this command to control the device's auxiliary SPI.

Table 8-90 - Request Data for Extended Command 0x0501 - Auxiliary SPI Control

Offset	Field Name	Value
0	Port Identifier	The identifier of the port to transmit on. Always set this field to zero.
1	Control Data	The control data. If this field is omitted, the existing control data is returned in the response data. See the control data field table for a description of each bit in this field.

Control Data Field (Bit 0 is the least significant bit)

Bit	Field Name	Value
0	Open	If this bit is set to one, the port is open. When this bit is set to zero, the port is closed. When the port is closed, the port's power output is turned off, the port's signal lines will float, and data can not be transmitted or received on the port.
1	CS Level	If this bit is set to one, the CS (chip select) output signal is set high when the port is open. If this bit is set to zero, the CS output signal is set low when the port is open.
2	DAV Level	If this bit is set to one, the DAV (Data Available) input signal is read high when the port is open. If this bit is set to zero, the DAV input signal is read low when the port is open. This bit always reads zero when the port is closed. This bit is read only. Writing to this bit does nothing.
3-7	Reserved	These bits should always be written as zeroes.

Table 8-91 - Response Data for Extended Command 0x0501 - Auxiliary SPI Control

Offset	Field Name	Value
0	Control Data	The control data. If the control data field is included with the request data then this field is not returned in the response data, otherwise the existing control data is returned in this field. See the control data field table for a description of each bit in this field.

Result codes:
0x0000 = Success

Example Request (Hex)

Header	
Command Number	49
Data Length	08

Data	
Extended Data Offset	00 00
Extended Command Number	05 01
Complete Extended Data Length	00 02
Extended Data	00 01

Example Response (Hex)

Header	
Result Code	0A
Data Length	06
Data	
Extended Data Offset	00 00
Extended Result Code	00 00
Complete Extended Data Length	00 00
Extended Data	Not Applicable

9 Properties

9.1 About Properties

MagneSafe V5 devices have a number of programmable configuration properties stored in non-volatile memory. Most of the programmable properties pertain to data formats other than vendor-defined HID, but some of the properties deal with the device regardless of format (for information about changing formats and making format-specific properties visible, see **Property 0x10 - Interface Type**). These properties can be configured at the factory or by an administrator using software tools supplied by MagTek. Changing these configuration properties requires low-level communication with the device. Details for communicating with the device to read or change programmable properties are provided in section **8.3.1 Command 0x00 - Get Property** and section **8.3.2 Command 0x01 - Set Property (MAC)**.

9.2 Property 0x00 - Firmware ID

Property ID: `0x00`
Property Type: String
Length: 11 bytes or 13 bytes
Get Property: Yes
Set Property: No
Default Value: Part number of installed firmware

This is an 11-byte or 13-byte read-only property that identifies the firmware part number and revision installed on the device. The first 8 or 10 bytes represent the part number, the next byte represents the firmware major revision number, and the final two bytes represent an internal build number. For example, this property might be "21042812D01".

(Embedded V5 Head Only)
For products that have a MagneSafe V5 IntelliHead embedded, two firmware IDs are available: One in the device itself, and one in the embedded IntelliHead. This property returns the values for the embedded IntelliHead. To get the values for the device, use **Property 0x3A - Firmware ID 2 (Embedded V5 Head Only)**.

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	00

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	0B	32 31 30 34 32 38 31 32 44 30 31

9.3 Property 0x01 - USB Serial Number (HID Only | KB Only)

Property ID: `0x01`
Property Type: String
Length: 0 - 15 bytes
Get Property: Yes
Set Property: Yes
Default Value: Null string / ASCII device serial number set when the device is configured.

The value contains the USB serial number, from 0 to 15 bytes long. The device sends the value of this property (if any) to the host during USB device enumeration. This is useful for distinguishing between devices when more than one of the same kind of device is attached to the host.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	04	01	31 32 33

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	01

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	03	31 32 33

9.4 Property 0x02 - USB Polling Interval (HID Only | KB Only)

Property ID: 0x02
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x01

This one-byte value contains the device's polling interval in milliseconds for the **Interrupt In** Endpoint, can be between 1 - 255. The device sends the value of this property (if any) to the host during USB device enumeration, and the host can use it to determine how often to poll the device for USB Input Reports [see section **2.1.3 How to Receive Data On the USB Connection (HID Only)**]. For example, if the polling interval is set to 10, the host polls the device for Input Reports every 10ms. This property can be used to speed up or slow down the time it takes to send Input Reports to the host. The trade-off is that speeding up the polling interval increases the USB bus bandwidth used by the device.

If the USB host hardware is configured to use a small keyboard buffer, the device may drop characters and host software developers may use this setting to reduce the device's transmission speed to compensate. However, a better solution is to increase the host hardware's keyboard buffer size. For example, on a USB host with a buffer size of 100 bytes, increasing the buffer size to 1000 may allow much shorter polling intervals resulting in faster transmission speeds without reducing reliability. For details about adjusting keyboard buffer size, see the documentation about "Keyboard Buffer Size" for the specific host hardware.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	02	02	0A

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	02

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	01

9.5 Property 0x03 - Device Serial Number

Property ID: `0x03`
Property Type: String
Length: 0 - 15 bytes
Get Property: Yes
Set Property: Yes (Once)
Default Value: Null string / ASCII device serial number set when the device is configured.

The property contains the device serial number, and is 0 to 15 bytes long. The device sends the value of this property (if any) to the host in the Device Serial Number field of **Magnetic Stripe Card Data Sent from Device to Host**, and in **ARQC Messages (EMV Only)**, **ARPC Response from Online Processing (EMV Only)**, and **Transaction Result Messages (EMV Only)**. This property may be Set only once; attempts to Set the property again fail with RC = 0x07 (Sequence Error). Note this value does not necessarily have the same value as **Property 0x01 - USB Serial Number (HID Only | KB Only)**, which is used mostly for differentiating identical devices after USB enumeration.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	04	03	31 32 33

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	03

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	03	31 32 33

9.6 Property 0x04 - MagneSafe Version Number

Property ID: 0x04
Property Type: String
Length: 0 - 7 bytes
Get Property: Yes
Set Property: No
Default Value: “V05”

This is a maximum 7-byte read-only property that identifies the MagneSafe Feature Level supported on this device. Attempts to set this property fail with RC = 0x01.

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	04

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	03	56 30 35

9.7 Property 0x05 - Track ID Enable (MSR Only)

Property ID: `0x05`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x95

This property is defined as follows:

Bit Position	7	6	5	4	3	2	1	0
	id	0	T_3	T_3	T_2	T_2	T_1	T_1

id = 0: Decodes standard ISO/ABA cards only
id = 1: Decodes AAMVA and 7-bit cards also

If the id flag is set to 0, only tracks that conform to the ISO card data format allowed for that track are decoded. If the track cannot be decoded by the ISO method, the device reports a decode error.

For each pair of track bits, valid values are as follows:
$T_\#$ = 00: Track Disabled
$T_\#$ = 01: Track Enabled
$T_\#$ = 10: Track Enabled and Required (Error if blank)

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	02	05	95

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	05

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	95

9.8 Property 0x07 - ISO Track Mask

Property ID: `0x07`
Property Type: String
Length: 6 bytes
Get Property: Yes
Set Property: Yes
Default Value: “04040Y”

This property specifies how the device should mask data on ISO/ABA type cards: Each byte in the sequence has the following meaning:

Offset	Description
0..1	These bytes are an ASCII representation of a decimal value that specifies how many of the leading characters of the PAN the device sends unmasked. The range is from “00” to “99”.
2..3	These bytes are an ASCII representation of a decimal value that specifies how many of the trailing characters of the PAN the device sends unmasked. The range is from “00” to “99”.
4	**Masking Character.** This byte specifies which character the device uses for masking. If this byte contains the uppercase letter ‘V’, the following rules apply: 1) The device masks the PAN using character ‘0’ 2) The device leaves all data after the PAN unmasked, leaving Discretionary Data (“DD”) and other non-PAN data available for the host to read.
5	This byte specifies whether the device applies Mod 10 Correction to the PAN. “Y” means Yes, “N” means No. This option is only effective if the Masking Character specified by this command is “0”.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.9 Property 0x08 - AAMVA Track Mask (MSR Only)

Property ID: `0x08`
Property Type: String
Length: 6 bytes
Get Property: Yes
Set Property: Yes
Default Value: `"04040Y"`

This property specifies the factors for masking data on AAMVA type cards. Each byte in the property has the following meaning:

Offset	Description
0..1	These bytes are an ASCII representation of a decimal value that specifies how many of the leading characters of the Driver's License/ID Number (DL/ID#) the device sends unmasked. The range is from "00" to "99".
2..3	These bytes are an ASCII representation of a decimal value that specifies how many of the trailing characters of the DL/ID# sends unmasked. The range is from "00" to "99".
4	**Masking Character.** This byte specifies which character the device uses for masking. If this byte contains the uppercase letter 'V', the following rules apply: • The device masks the PAN according to the rules of this property (**Property 0x34 - Send AAMVA Card Data** is ignored) • The device uses '0' for masking the PAN • The device sends all data after the PAN without masking
5	This byte specifies whether the device applies Mod 10 Correction to the DL/ID#. "Y" means Yes, "N" means No. This option is only effective if the masking character specified in this command is "0".

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.10 Property 0x0A - USB HID Max Packet Size (HID Only)

Property ID: 0x0A
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes (Read-Only on some devices)
Default Value: 0x08 for all devices except eDynamo, iDynamo 6, kDynamo, mDynamo, DynaWave, and tDynamo, which use 0x40

The value is a byte that contains the device's maximum packet size for the USB **Interrupt In** endpoint when using the HID data format [see section **2.1.3 How to Receive Data On the USB Connection (HID Only)**]. The device sends the value of this property to the host during USB device enumeration. The value can be set in the range of 1 - 64 and has units of bytes. For example, if the maximum packet size is set to 8, the device sends HID reports in multiple packets of 8 bytes each, possibly fewer bytes for the last packet of the report. This property can be used to speed up or slow down the time it takes to send data to the host. Larger packet sizes speed up communications and smaller packet sizes slow down communications. The trade-off is that speeding up the data transfer rate increases the USB bus bandwidth used by the device.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	02	0A	08

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	0A

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	08

9.11 Property 0x0A - RS-232 / UART Communication Settings (RS-232 Only | UART Only, Configurable Baud Rate Only)

Property ID: `0x0A`

Property Type: Byte

Length: 1 byte

Get Property: Yes

Set Property: Yes

Default Value: 0x02

The host uses this property to specify the speed, stop bits, and parity the device will use to transmit and receive on its serial connection (RS-232 UART or logic level UART). The speed, stop bits, and parity are independently selectable. The new parameters will be in effect after the command is accepted and the device is reset or power cycled. A host that wishes to communicate with the device must use the correct speed and parity. There is no method of bringing a device to a “default” speed and parity, thus the host software that changes this property must be aware of the settings, or must discover the settings by sending repeated commands over the range of possible settings (48 possible configurations). The number of data bits in each character is dependent on the parity setting (see **Table 9-1)**.

Note that Mark parity (see **Table 9-1**) can be interpreted two ways:

- 7 bit characters, parity always equals 1, one stop bit, or
- 7 bit characters, no parity, 2 stop bits.

The Stop Bits specification applies only to data transmitted from the device; the device never requires more than one stop bit from the host, but functions normally if the host includes more than one stop bit.

Table 9-1 defines how to interpret the values this property may have.

Table 9-1 - RS-232 / UART Communications Property Key

Bit								Description
7	6	5	4	3	2	1	0	
					0	0	0	Baud Rate 2400
					0	0	1	Baud Rate 4800
					0	1	0	Baud Rate 9600
					0	1	1	Baud Rate 14400
					1	0	0	Baud Rate 19200
					1	0	1	Baud Rate 38400
					1	1	0	Baud Rate 9600 [sic.]
					1	1	1	Baud Rate 9600 [sic.]
			0	0				No Parity (8 bit characters)
			0	1				Even Parity (7 bit characters)
			1	0				Odd Parity (7 bit characters)
			1	1				Mark (Parity = 1 all the time, 7 bit characters)
		0						1 Stop Bit

Bit								Description
		1						2 Stop Bits
0	0							Not used. Set to zeroes.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.12 Property 0x0B - Activity Timeout Power-Down Period (PM1 Only)

Property ID: `0x0B`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 120 (0x78) seconds

The host uses this property to specify, in seconds, the minimum amount of time a wireless device operates in the absence of activity, and is used to conserve battery life. When the specified time passes without activity, the device powers down. Activity is defined as any of the following:

- (MSR Only) Swiping and processing a card.
- Receiving and processing commands from the host.
- Pressing the pushbutton.

Setting the timeout period to `0x00` directs the device to never power down because of inactivity. Setting **Property 0x0E - Stay Powered After Swipe (PM1 Only)** to `0x01` directs the device to not power down after a good swipe. Together, the host can use these two properties to direct the device to stay powered on until an operator powers it off manually.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.13 Property 0x0D - Bluetooth Disconnect Message (Bluetooth Only)

Property ID: 0x0D
Property Type: String
Length: 7 bytes
Get Property: Yes
Set Property: Yes
Default Value: Null string

This property is used as part of a Bluetooth Disconnect Message. The message is intended to give the host software a warning that the device is disconnecting. The full disconnect message consists of the specified string followed by the character '-' (hyphen), followed by a single-character Reason Code, followed by a Carriage Return (0x0D) character. The possible Reason Codes are:

- 'T' = Timeout
- 'U' = Pushbutton (User-initiated)
- 'B' = Battery Low
- 'S' = Card Swipe
- 'R' = Reset command
- 'I' = Interface changed

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.14 Property 0x0E - Stay Powered After Swipe (PM1 Only)

Property ID: `0x0E`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00 (Don't stay powered)

This property controls whether the device stays powered after a good swipe. If the property value is `0x00` (the default), the device powers down after a good swipe. If the property value is `0x01`, the device stays powered after a good swipe. In this case, the device may be powered down by pressing and holding the pushbutton, or it will power off after the activity timeout set by **Property 0x0B - Activity Timeout Power-Down Period (PM1 Only)**.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.15 Property 0x10 - Interface Type

Property ID: `0x10`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes (No for devices that switch connections automatically)
Default Value: Depends on device type:

- BulleT KB / BulleT SPP- 0x03
- cDynamo - 0x02
- Dynamag / USB Encrypting IntelliHead with V5 - 0x00
- DynaWave - N/A
- eDynamo - 0x00
- iDynamo, iDynamo 5, iDynamo 5 Gen II, iDynamo 6 with Lightning connector - 0x02
- iDynamo 6 with USB-C connector - 0x00
- kDynamo - 0x02
- mDynamo - 0x00
- MSR Insert Reader - 0x00
- pDynamo - 0x03
- RS-232/UART MagneSafe V5 MSR - 0x02
- sDynamo - 0x02
- tDynamo - 0x00

This property represents the device's current connection type (see section **2 Connection Types**) and data format (see section **3 Data Formats**):

- Valid values for this property are
  - `0x00` = USB HID (HID Only)
  - `0x01` = USB Keyboard Emulation (KB) (USB KB Only)
  - `0x02` = iAP / RS-232 / UART (RS-232 Only | UART Only | iAP Only)
  - `0x03` = Bluetooth (Bluetooth Only)
  - `0xFF` = One-Time Automatic (HID Only | iAP Only). When the property is set to this value and the device connects to a host, the device attempts to determine which interface type the host is using. After it successfully detects the interface type, it automatically sets this property to the value that corresponds to that interface type.
- On devices that have only one possible value for this property, the property is read-only.
- On devices that support multiple values for this property and do not handle connection switching automatically, the host can use this property to change the device's behavior. MagTek strongly recommends the host set this property before setting other properties, and immediately power cycle or reset the device (see **Command 0x02 - Reset Device)**, because it changes which other properties are available.
- (HID Only | iAP Only) The iDynamo 6 supports two different modes (iAP or HID) of communication. The device can only support one communication mode at a time: HID mode, for Windows and Android devices, or iAP mode for iOS devices. If the device is set to the wrong mode, the host (phone, tablet, or PC) will not be able to communicate with the reader.

If it is suspected that the iDynamo 6 is set to the wrong communication mode, the below described method would be used to toggle the communication mode from/to iAP/HID modes.

1 – With the reader disconnected from any USB power source (including an active host connection), the user will insert (and leave inserted) any card into the dip card slot of iDynamo 6. Blocking the inserted card sensor of the reader prior to power up is necessary to trigger the software to allow toggling the communication mode.

2 – The user then connects iDynamo 6 to any USB-C power source (Windows host, Android host, Apple host, or power adapter plugged into wall power).

3 – When power is established, the iDynamo 6 immediately powers on with a blocked sensor in the dip card slot (this is the 0 second mark). As described in step 1, if the card is NOT blocking the sensor in the dip card slot when the device first receives power, the regular power up cycle commences, and the LED will turn solid GREEN and you will not be able to toggle the communication mode.

4 – The LED flash cycle begins and lasts for 6 seconds total, starting with the beginning flash of the LED.

a. If iDynamo 6 is set to iAP, the Flash Cycle will illuminate the LED with a BLUE light
a. If iDynamo 6 is set to HID, the Flash Cycle will illuminate the LED with a GREEN light

5 – During this 6 second flash cycle, the user should rapidly withdraw and insert the card into the dip card slot 3 or more times. Doing this correctly will cause the iDynamo 6 to toggle the existing communication mode.

At the end of the flash cycle (after 6 seconds have elapsed), the LED will turn solid GREEN. iDynamo 6 should now ready be ready to connect to the host application.

If the connection still cannot be established, please repeat the process from Step 1 and try again.

- (Bluetooth LE Only | iAP Only, USB Only) This property only governs behavior on devices with a single primary host connection (see section **1.4 About Connections and Data Formats**). Devices with more than one connection should use **Property 0x5F - Notification Output Connection (Bluetooth LE Only | iAP Only, USB Only)** to set which connection the device will use when it sends **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)** and **Notification Messages Sent from Device to Host (Extended Notifications Only)**.
- (Bluetooth LE Only, KB Only) This property does not govern the data format for Bluetooth LE devices that support multiple data formats (see section **1.4 About Connections and Data Formats**). For those devices, the host can change the output format using **Bluetooth LE Property 0x11 - Bluetooth LE Connection Type (MSR Only, KB Only)**.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	02	10	00

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	10

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	00

9.16 Property 0x14 - Track Data Send Flags (KB Only | Streaming Only, MSR Only)

Property ID: `0x14`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x63 for all models except BulleT KB which defaults to 0x6B

The host uses this property to alter the formatting of **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)** as follows:

Bit Position	7	6	5	4	3	2	1	0
	ICL	SS	ES	LRC	MKR	LC	Er	

Er = 00: The device does not send card data when a decode error occurs. Not currently implemented.
Er = 01: The device does not send track data when a decode error occurs.
Er = 11: Send the single character 'E' as the track data for each track with a decode error.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.16.1 KB Mode Flags (KB Only)

The Caps Lock key on a host's keyboard does not merely affect the keyboard; it sets the Caps Lock state for all keyboard devices connected to the host. Devices in KB mode would therefore be affected. The device provides an Ignore Caps Lock (ICL) setting to compensate for this:

- ICL = 0: Enabling **Caps Lock** using another keyboard connected to the host does not affect the case of the data coming from the device.
- ICL = 1: Enabling **Caps Lock** using another keyboard connected to the host inverts the case of the data coming from the device.

Minimizing key reports (MKR) means the device sends the minimum number of key reports to represent each character. When **Property 0x17 - ASCII to Keypress Conversion Type (KB Only, MSR Only)** is set to `ACTIVE KEYMAP`, the minimum number consists of one key-down report per character, except in the case of transmitting more than one of the same character in a row. In this case, the device sends a key-down followed by a key-up. When **Property 0x17 - ASCII to Keypress Conversion Type (KB Only, MSR Only)** is set to `ALT ASCII code`, the minimum number consists of four key reports per character (Alt and first digit down, second digit down, third digit down, Alt and third digit up). This mode is up to two times faster, but it may not work with all host software.

When not minimizing key reports, the maximum number of key reports is sent to represent each character. When **Property 0x17 - ASCII to Keypress Conversion Type (KB Only, MSR Only)** is set to `ACTIVE KEYMAP`, the maximum number consists of two key reports per character (one for the key down and one for the key up). When **Property 0x17 - ASCII to Keypress Conversion Type (KB Only, MSR Only)** is set to `ALT ASCII code`, the maximum number consists of eight key reports per character (Alt down, first digit down, first digit up, second digit down, second digit up, third digit down, third digit up, Alt up). This mode is slower but it works with all host software.

The MKR flag is currently only supported for the BulleT KB. It should be set to zero for all other connection types. Support for this flag may eventually be added to the USB connection type with KB data format.

- MKR = 0: Don’t minimize key reports.
- MKR = 1: Minimize key reports.

The state of the Caps Lock key on the host keyboard has no effect on the case of transmitted card data unless the **ICL** bit in this property is set to 1, in which case if Caps Lock is enabled, the card data is transmitted opposite to what is specified by the following Lower case (LC) bit:

- LC = 0: Send card data as uppercase.
- LC = 1: Send card data as lowercase.

9.16.2 Streaming Flags (Streaming Only)

SS = 0: Don’t send Start Sentinel for each track.
SS = 1: Send Start Sentinel for each track.

ES = 0: Don’t send End Sentinel for each track.
ES = 1: Send End Sentinel for each track.

The LRC is the unmodified LRC from the track data. If the host software needs to verify the LRC, it would need to restore the original Start Sentinels, then convert the track data from ASCII to card data format, and apply the LRC calculation algorithm appropriate for the card format (e.g., ISO or AAMVA). The LRC property only applies to track data sent in Streaming mode and completely in the clear (Security Level 2).

- LRC = 0: Don’t send LRC for each track.
- LRC = 1: Send LRC for each track.

9.17 Property 0x15 - MagnePrint Flags (MSR Only)

NOTICE

(Streaming Only)
Using this command adds or suppresses fields in the data stream between the device and host, which changes the position of all subsequent data elements in the stream. This may render the device incompatible with host software that expects to parse a fixed format, rather than using Property 0x2C - Format Code (Streaming, MSR Only) **to determine the position of data in the stream.**

Property ID: `0x15`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00 in all devices except DynaPAD and MSR insert readers
Default Value: 0x01 for DynaPAD and MSR insert readers

The host uses this property to direct the device to either include or exclude MagnePrint data in **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)** when the device is in **Security Level 2**. At higher security levels, the device always sends encrypted MagnePrint data.

Bit Position	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	0	S

S = 0: Device does not include MagnePrint Data
S = 1: Device includes MagnePrint Data

Setting S to 1 directs the device to send **MagnePrint Status**, **MagnePrint Data Length (HID Only | GATT Only | SLIP Only)**, **MagnePrint Absolute Data Length (HID Only | TLV Only | GATT Only | SLIP Only)**, and **Encrypted MagnePrint Data** with each swipe when it is in **Security Level 2**.

Setting S to 0 directs the device to zero-fill these values. (Streaming Only) When the device is using Streaming format, it omits these values altogether, along with the programmable field separators that precede each value.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.18 Property 0x16 - Active Keymap (KB Only, MSR Only)

Property ID: `0x16`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00 (United States)

This property is a byte that specifies which key map the device should use. The value can be set to 0x00 for the United States key map, or to 0x01 for a custom key map. The active key map is used by the device to convert ASCII data into keystrokes. The United States key map should be used with any host configured to use United States keyboards. The custom key map can be used to set up the device to work with hosts configured to use keyboards for other locales. The default custom key map is the same as the United States key map, and can be modified to another country's key map as follows:

1) Set **Property 0x16 - Active Keymap (KB Only, MSR** Only) to select an active key map to be modified.
2) Reset the device to make this change take effect.
3) Use **Command 0x03 - Get Keymap Item (KB Only)** and **Command 0x04 - Set Keymap Item (MAC, KB Only)** to modify the active key map.
4) Use **Command 0x05 - Save Custom Keymap (MAC, KB Only)** to save the active key map as the custom key map.
5) Set **Property 0x16 - Active Keymap (KB Only, MSR Only)** to use the custom key map.
6) Reset the device to make these changes take effect.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	02	16	00

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	16

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	00

9.19 Property 0x17 - ASCII to Keypress Conversion Type (KB Only, MSR Only)

Property ID: `0x17`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00 (keymap)

This property is a byte that specifies how the device converts ASCII data to keystrokes. The value can be set to `0x00` for **keymap** [the active keymap is set with **Property 0x16 - Active Keymap (KB Only, MSR Only)**] or to `0x01` for **ALT ASCII code** (international keyboard emulation).

When the value is set to `0x00` (keymap), data is transmitted to the host according to the active keymap, which defaults to the United States keyboard keymap. For example, to transmit the ASCII character '?' (063 decimal), the device looks up the character in a keymap. For a United States keyboard keymap, the '/' (forward slash) key combined with the left shift key modifier are stored in the keymap to represent the key press combination that is used to represent the ASCII character '?' (063 decimal).

When the value is set to `0x01` (ALT ASCII code), instead of using the key map, the device transmits an international keyboard keypress combination, consisting of the decimal values of the ASCII character combined with the ALT key modifier. For example, to transmit the ASCII character '?' (063 decimal), the device sends keypad '0' combined with left ALT key modifier, keypad '6' combined with the left ALT key modifier, then keypad '3' combined with the left ALT key modifier.

Generally, if the device only needs to emulate a United States keyboard, set this property to `0x00`. If the device needs emulate all countries' keyboards, set it to `0x01`. The tradeoff is that ALT ASCII code mode is slightly slower than keymap mode, because the device transmits more keypresses. Some host software is not compatible with ALT ASCII code mode.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	02	17	00

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	17

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	00

9.20 Property 0x19 - CRC Flags (Streaming Only, MSR Only)

Property ID: `0x19`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x01

This property specifies the behavior of the values **Clear Text CRC (Streaming Only)** and **Encrypted CRC (Streaming Only)** within **Magnetic Stripe Card Data Sent from Device to Host (MSR Only | Keypad Entry Only)**.

Bit Position	7	6	5	4	3	2	1	0
	0	0	0	0	0	0	E	S

E = 0: Device does NOT send Encrypted CRC.
E = 1: Device sends the Encrypted CRC.

S = 0: Device does NOT send the Clear Text CRC.
S = 1: Device sends the Clear Text CRC.

With the default setting `0x01`, the device sends only the Clear Text CRC. If both E and S are set to 0, the device still sends the Programmable Field Separator that precedes each of these fields.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.21 Property 0x1A - Keyboard SureSwipe Flags (SureSwipe Only, Streaming Only, KB Only, MSR Only)

Property ID: `0x1A`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x01

This property enables/disables SureSwipe emulation when in **Security Level 2** with **Property 0x10 - Interface Type** set to Keyboard. A value of `0x01` enables SureSwipe emulation, a value of `0x00` disables it. The default is `0x01`, meaning the device sends keyboard data in SureSwipe format (see MagTek document ***D99875206 TECHNICAL REFERENCE MANUAL, USB KB SURESWIPE & SWIPE READER***. This allows customers to receive a device without security enabled (**Security Level 2**) and use it exactly like a SureSwipe device. Later, when the customer is ready, they can switch the device to a higher Security Level and take advantage of the robust security features offered by the device. Developers might disable SureSwipe emulation to allow the device to transmit keyboard data in the full MagneSafe V5 format without encryption so they can write host software that works with this format without initially having to deal with cryptography.

This property is only effective when using the USB or Bluetooth LE connection with Streaming format [see section **3.3 How to Use Streaming Format (Streaming Only)**]. If you wish to send SureSwipe data using HID format, see **Property 0x38 - HID SureSwipe Flag (SureSwipe Only, HID Only, MSR Only)**.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.22 Property 0x1B - Decode Enable (JIS Support Only, MSR Only)

Property ID: 0x1B
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00

This property is defined as follows:

Bit Position	7	6	5	4	3	2	1	0
Decode Type	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	JIS Type 2

When a Decode Type bit is set to 1 (True), the device decodes the corresponding type of data, otherwise the device will not decode that type of data. The reserved bits should always be set to zero.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	02	1B	01 (enable JIS Type 2 decode type)

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	1B

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	01

9.23 Property 0x1C - Start Sentinel JIS Type 2 (Streaming Only, JIS Support Only, MSR Only)

Property ID: `0x1C`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x7F 'DEL'

This property is the single character the device sends as the start sentinel for cards encoded in JIS Type 2 format (see **Property 0x1B - Decode Enable** for details about enabling JIS decoding). If the value is in the range 0 - 127, the device sends the equivalent ASCII character.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.24 Property 0x1D - End Sentinel JIS Type 2 (Streaming Only, JIS Support Only, MSR Only)

Property ID: `0x1D`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x7F 'DEL'

This property is the single character the device sends as the end sentinel for cards encoded in JIS type 2 format (see **Property 0x1B - Decode Enable** for details about enabling JIS decoding). If the value is in the range 0 - 127, the device sends the equivalent ASCII character.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.25 Property 0x1E - Pre Card String (Streaming Only, MSR Only)

Property ID: `0x1E`
Property Type: String
Length: 0 - 7 bytes
Get Property: Yes
Set Property: Yes
Default Value: Null string

The device sends the value of this property to the host before all other card data. For example, if the host software requires a set of keystrokes to begin the process of receiving card data, this property could be set to transmit that keystroke sequence.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	04	1E	31 32 33

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	1E

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	03	31 32 33

9.26 Property 0x1F - Post Card String (Streaming Only, MSR Only)

Property ID: `0x1F`
Property Type: String
Length: 0 - 7 bytes
Get Property: Yes
Set Property: Yes
Default Value: Null string.

The device sends the host the value of this property after all other card data.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	04	1F	31 32 33

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	1F

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	03	31 32 33

9.27 Property 0x20 - Pre Track String (Streaming Only, MSR Only)

Property ID: `0x20`
Property Type: String
Length: 0-7 bytes
Get Property: Yes
Set Property: Yes
Default Value: Null string

The device sends the host the value of this property before the data for each track. If the value is 0, the device does not send a pre-track string.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	04	20	31 32 33

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	20

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	03	31 32 33

9.28 Property 0x21 - Post Track String (Streaming Only, MSR Only)

Property ID: `0x21`
Property Type: String
Length: 0-7 bytes
Get Property: Yes
Set Property: Yes
Default Value: Null string.

The device sends the host the value of this property after the data for each track. If the value is 0, the device does not send a pre-track string.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	04	21	31 32 33

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	21

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	03	31 32 33

9.29 Property 0x22 - Termination String (Streaming Only, MSR Only)

Property ID: `0x22`
Property Type: String
Length: 0-7 bytes
Get Property: Yes
Set Property: Yes
Default Value: 0x0D (carriage return)

The device sends the host the value of this property after the all the data for a transaction. If the value is 0, the device does not send a termination string.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	02	22	0D

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	22

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	0D

9.30 Property 0x23 - Field Separator (Streaming Only, MSR Only)

Property ID: 0x23
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x7C ('|')

This property stores the character the device uses for P35 (see section **3.3 How to Use Streaming Format**). If the value is in the range 1 - 127, the device sends the equivalent ASCII character. If the value is 0, the device does not send a delimiter, which is inadvisable.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	02	23	7C

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	23

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	7C

9.31 Property 0x24 - Start Sentinel Track 1 ISO ABA (Streaming Only, MSR Only | Keypad Entry Only)

Property ID: `0x24`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x25 ('%')

This property stores the character the device uses to replace the Track 1 Start Sentinel for cards where it recognizes Track 1 encoded in ISO/ABA format. The default value is the standard ISO/ABA Track 1 Start Sentinel, meaning no replacement. If the value is in the range 1 - 127, the device sends the equivalent ASCII character. If the value is 0, the device does not send a character.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.32 Property 0x25 - Start Sentinel Track 2 ISO ABA (Streaming Only, MSR Only)

Property ID: `0x25`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x3B (';')

This property stores the character the device uses to replace the Track 2 Start Sentinel for cards where it recognizes Track 2 encoded in ISO/ABA format. The default value is the standard ISO/ABA Track 2 Start Sentinel, meaning no replacement. If the value is in the range 1 - 127, the device sends the equivalent ASCII character. If the value is 0, the device does not send a character.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.33 Property 0x26 - Start Sentinel Track 3 ISO ABA (Streaming Only, MSR Only)

Property ID: `0x26`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x2B ('+')

This property stores the character the device uses as a Track 3 Start Sentinel for cards where it recognizes Track 3 encoded in ISO/ABA format. If the value is in the range 1 - 127, the device sends the equivalent ASCII character. If the value is 0, the device does not send a character.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.34 Property 0x27 - Start Sentinel Track 3 AAMVA (Streaming Only, MSR Only)

Property ID: `0x27`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x23 (‘#’)

This property stores the character the device uses as a Track 3 Start Sentinel for cards where it recognizes Track 3 encoded in AAMVA format. If the value is in the range 1 - 127, the device sends the equivalent ASCII character. If the value is 0, the device does not send a character.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.35 Property 0x28 - Start Sentinel Track 2 7bits (Streaming Only, MSR Only)

Property ID: `0x28`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x40 (‘@’)

This property stores the character the device uses to replace the Track 2 Start Sentinel for cards where it recognizes Track 2 encoded in the 7-bit ISO format used for Track 1. If the value is in the range 1 - 127, the device sends the equivalent ASCII character. If the value is 0, the device does not send a character.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.36 Property 0x29 - Start Sentinel Track 3 7bits (Streaming Only, MSR Only, 3-Track Only)

Property ID: `0x29`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x26 (‘&’)

This property stores the character the device uses to replace the Track 3 Start Sentinel for cards where it recognizes Track 3 encoded in the 7-bit ISO format used for Track 1. If the value is in the range 1 - 127, the device sends the equivalent ASCII character. If the value is 0, the device does not send a character.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.37 Property 0x2B - End Sentinel (Streaming Only, MSR Only)

Property ID: `0x2B`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x3F ('?')

This property stores the character the device sends as the End Sentinel for all tracks in any card data format [unless it is overridden, track by track, by **Property 0x2D - End Sentinel Track 1 (Streaming Only, MSR Only)**, **Property 0x2E - End Sentinel Track 2 (Streaming Only, MSR Only)**, or **Property 0x2F - End Sentinel Track 3 (Streaming Only, MSR Only, 3-Track Only)**]. In tracks that have a standard end sentinel embedded, it replaces them. If the value is in the range 1 - 127, the device sends the equivalent ASCII character. If the value is 0, the device does not send a character.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.38 Property 0x2C - Format Code (Streaming Only, MSR Only)

Property ID: `0x2C`
Property Type: String
Length: 4 bytes
Get Property: Yes
Set Property: Yes
Default Value: "0000"

This property specifies the Format Code the device includes when it sends card swipe data to the host [see section **6.25 Format Code (Streaming Only)**], and is designed to allow programmers of host software to populate the final three characters as "notes" the host software can use to determine how to parse or interpret the accompanying data. When setting this property, the host must send four characters: The device ignores the first character, which is reserved for MagTek/device use, and changes the final three characters of the property to the final three characters the host specified.

The value can be interpreted as follows:

- By default, the Format Code is "0000".
- If the manufacturer configures **Property 0x30 - Send Remaining MSR Transactions Counter (Streaming Only, MSR Only)** the device changes the Format Code from "0000" to "0001."
- If the host directly sets the value of the Format Code property, the new value overrides the factory set value. The host can change the final three characters, but making such a change automatically causes the first character to change to "1".
- If an administrator or host software changes a setting that automatically updates Format Code, the first character of the Format Code changes to a "1". Such settings include:
    - **Property 0x15 - MagnePrint Flags (MSR Only)**
    - **Property 0x16 - Active Keymap (KB Only, MSR Only)**

- **Property 0x17 - ASCII to Keypress Conversion Type (KB Only, MSR Only)**
- **Property 0x14 - Track Data Send Flags (KB Only | Streaming Only, MSR Only)**
- **Property 0x19 - CRC Flags (Streaming Only, MSR Only)**
- **Property 0x1E - Pre Card String (Streaming Only, MSR Only)**
- **Property 0x1F - Post Card String (Streaming Only, MSR Only)**
- **Property 0x20 - Pre Track String (Streaming Only, MSR Only)**
- **Property 0x21 - Post Track String (Streaming Only, MSR Only)**
- **Property 0x22 - Termination String (Streaming Only, MSR Only)**
- **Property 0x23 - Field Separator (Streaming Only, MSR Only)**
- **Property 0x24 - Start Sentinel Track 1 ISO ABA (Streaming Only, MSR Only | Keypad Entry Only)**
- **Property 0x25 - Start Sentinel Track 2 ISO ABA (Streaming Only, MSR Only)**
- **Property 0x26 - Start Sentinel Track 3 ISO ABA (Streaming Only, MSR Only)**
- **Property 0x27 - Start Sentinel Track 3 AAMVA (Streaming Only, MSR Only)**
- **Property 0x28 - Start Sentinel Track 2 7bits (Streaming Only, MSR Only)**
- **Property 0x29 - Start Sentinel Track 3 7bits (Streaming Only, MSR Only, 3-Track Only)**
- **Property 0x2B - End Sentinel (Streaming Only, MSR Only)**
- The custom keymap changed by a call to **Command 0x05 - Save Custom Keymap (MAC, KB Only)**

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.39 Property 0x2D - End Sentinel Track 1 (Streaming Only, MSR Only)

Property ID: `0x2D`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0xFF

This property specifies the character the device sends as the end sentinel for Track 1 with any card format. In tracks that have standard end sentinels embedded, it replaces them. If the value is in the range 1 - 127, the device sends the equivalent ASCII character. If the value is `0xFF`, the device uses the value of **Property 0x2B - End Sentinel** instead. If the value is 0, the device does not send a character.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.40 Property 0x2E - End Sentinel Track 2 (Streaming Only, MSR Only)

Property ID: `0x2E`
Property Type: Byte
Length: 1 byte
Get Property: Yes

Set Property: Yes
Default Value: 0xFF

This property specifies the character the device sends as the end sentinel for Track 2 with any card format. In tracks that have standard end sentinels embedded, it replaces them. If the value is in the range 1 - 127, the device sends the equivalent ASCII character. If the value is `0xFF`, the device uses the value of **Property 0x2B - End Sentinel** instead. If the value is 0, the device does not send a character.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.41 Property 0x2F - End Sentinel Track 3 (Streaming Only, MSR Only, 3-Track Only)

Property ID: `0x2F`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0xFF

This property specifies the character the device sends as the end sentinel for Track 3 with any card format. In tracks that have standard end sentinels embedded, it replaces them. If the value is in the range 1 - 127, the device sends the equivalent ASCII character. If the value is `0xFF`, the device uses the value of **Property 0x2B - End Sentinel** instead. If the value is 0, the device does not send a character.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.42 Property 0x30 - Send Remaining MSR Transactions Counter (Streaming Only, MSR Only)

Property ID: `0x30`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00 (Don't send Remaining MSR Transactions counter)

This property specifies whether device sends the Remaining MSR Transactions counter (sometimes known as the transaction threshold, see **Command 0x1C - Get Remaining MSR Transactions Counter (MSR Only)** for details) as part of a Streaming message. If the property is set to `0x00`, the device sends neither the remaining MSR transactions counter nor the field separator. If the property is set to `0x01`, the device sends the remaining MSR transactions counter immediately following the **DUKPT Key Serial Number (KSN)** in a swipe message.

If this property is set to 0x01 and **Property 0x2C - Format Code (Streaming Only, MSR Only)** is currently "0001", the device changes **Property 0x2C - Format Code (Streaming Only, MSR Only)** to "0002".

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.43 Property 0x31 - Mask Other Cards (MSR Only)

Property ID: `0x31`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00 (Don't Mask Other cards)

This property designates whether cards which do not decode as either ISO/ABA (Financial) or AAMVA (Driver License) format should be sent with their data masked or unmasked. The default value (`0x00`) is to send the data unmasked. If this property is set to `0x01`, the device sends the track(s) to the host using a "0" for each byte of track data the device reads from the card.

If a card is encoded according to ISO/ABA rules (Track 1 in 7 bit format, Tracks 2 and Track 3 in 5 bit format), and Track 1 does not begin with the character 'B', the device always sends the **Track 1 Masked Data** value unmasked, regardless of the value of this property. See **Appendix E** for details.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

9.44 Property 0x34 - Send AAMVA Card Data Unmasked (MSR Only)

Property ID: `0x34`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00

This property controls how the device sends AAMVA card data when the security level is higher than **Security Level 2**:

- 0 = Send masked AAMVA card data.
- 1 = Send clear AAMVA card data.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	34	01

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	34

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	01

9.45 Property 0x38 - HID SureSwipe Flag (SureSwipe Only, HID Only, MSR Only)

Property ID: `0x38`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00

This property enables/disables SureSwipe emulation when in **Security Level 2** with **Property 0x10 - Interface Type** set to HID. This allows customers to receive a device without security enabled (**Security Level 2**) and use it in a similar manner to a SureSwipe device (for example, for convenience during software development). Later, when the customer is ready, they can switch the device to a higher Security Level and take advantage of the robust security features offered by the device.

When this property is set to `0x00`, the device functions as described in this document.

When this property is set to `0x01`, the device returns card swipes and enumerates with the same VID/PID as described in ***D99875191 TECHNICAL REFERENCE MANUAL, USB HID SURESWIPE & SWIPE READER***. It does not emulate the property settings defined there.

This property is only effective in USB HID mode. If you wish to send SureSwipe data in Streaming mode, see **Property 0x1A - Keyboard SureSwipe Flags (SureSwipe Only, Streaming Only, KB Only, MSR Only)**.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	38	01

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	38

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	01

9.46 Property 0x3A - Firmware ID 2 (Embedded V5 Head Only)

Property ID: `0x3A`
Property Type: String
Length: Fixed at 11 bytes
Get Property: Yes
Set Property: No
Default Value: Part number, major revision number, and build number of installed firmware

This 11-byte or 13 byte read-only property returns the part number, major revision number, and build number of secondary firmware installed on the device. The first 8 or 10 bytes represent the firmware part number, the next byte represents the firmware major revision number, and the final two bytes represent a firmware internal build number. For example, this property might be “21042812D01” where 21042812 is the part number, D is the major revision number, and 01 is the internal build number. To get the device’s primary firmware ID, use **Property 0x00 - Firmware ID**.

(Embedded V5 Head Only)
For devices with embedded MagneSafe V5 IntelliHeads, two firmware IDs are available: One for the device, and one for the embedded IntelliHead. This property returns values for the device.

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	3A

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	0B	32 31 30 34 32 38 31 32 44 30 31

9.47 Property 0x40 - Display Rotation Period (Display Only)

Property ID: `0x40`
Property Type: Word
Length: 2 bytes
Get Property: Yes
Set Property: Yes
Default Value: 0x96 (150) - 1.5 seconds

This property specifies the amount of time, in units of 10 milliseconds, that the device will show each segment of the display text during a Transaction Validation sequence (see **Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)**.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	03	40	02 58

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	40

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	00 96

9.48 Property 0x41 - Status Message Period (Display Only)

Property ID: `0x41`
Property Type: Word
Length: 2 bytes
Get Property: Yes
Set Property: Yes
Default Value: 0x12C (300) - 3 seconds

This property specifies the amount of time, in units of 10 milliseconds, the device shows each status message on the display before blanking the display [see **Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)**].

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	03	41	02 58

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	41

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	01 2C

9.49 Property 0x52 - Host Poll Timeout (HID Only | KB Only)

Property ID: `0x52`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x02 (2 seconds)

The host can use this property to adjust the device's host poll timeout. The property can be set to 0 to disable the timeout, or it can be set to a value in the range of 1 to 60 seconds.

If the host fails to retrieve a USB HID input report from the device within the timeout period, the device discards the report. The intent of this timeout is to avoid having the device lock up while trying to send a report to a host that is failing to retrieve it due to error conditions or because the host is not ready to receive.

Not all devices have such a timeout, and not all readers implement this property to adjust it.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	52	02

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	52

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	02

9.50 Property 0x53 - Inter-Key Delay (Bluetooth Only, KB Only)

Property ID: 0x53
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x0C (12ms)

This property controls how long the device pauses between each key report sent to the Bluetooth module. This delay can be adjusted between 0 and 250 milliseconds. If the delay is too small, characters will be dropped from the card data. Larger delays lengthen the time it takes the device to send card data to the host.

MagTek tests indicate the following delays work reliably with the specified devices:

- 12ms (0x0C): Windows XP and Windows 7 PCs using an IOGear GB421 USB Bluetooth Adapter; iPhone, iPad 2, and MacBook Pro with OS X 10.6.8.
- 35ms (0x23): Samsung Galaxy Nexus with Android 4.0.2.
- 50ms (0x32): Motorola tablet with Android 3.2.1.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	53	0C

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	53

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	0C

9.51 Property 0x54 - Card Data Encryption Variant (MSR Only, Configurable MSR Variants Only)

Property ID: 0x54
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00 (PIN Variant)

This property specifies which variant of the current DUKPT Key the device uses to encrypt magnetic stripe **Track 1 Encrypted Data**, **Track 2 Encrypted Data**, **Track 3 Encrypted Data**, and **Encrypted Session ID**:

- 0x00 = Use **PIN Encryption** variant
- 0x01 = Use **Data Encryption, request or both ways** variant

The host software should use this value to determine how to create the correct Derived Key to decrypt **Encrypted Track Data** (see section **5 Encryption, Decryption, and Key Management**). The algorithms for creating the Derived Key fitting each of the possible variants are fully specified in ***ANS X9.24-1:2009***.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	54	01

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	54

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	01

9.52 Property 0x56 - MagnePrint Data Encryption Variant (MSR Only, Configurable MP Variants Only)

Property ID: `0x56`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00 (PIN Variant)

This property specifies which variant of the current DUKPT Key the device uses to encrypt magnetic stripe **Encrypted MagnePrint Data**:

- 0x00 = Use **PIN Encryption** variant
- 0x01 = Use **Data Encryption, request or both ways** variant

The host software should use this value to determine how to create the correct Derived Key to decrypt **Encrypted MagnePrint Data** (see section **5 Encryption, Decryption, and Key Management**). The algorithms for creating the Derived Key fitting each of the possible variants are fully specified in ***ANS X9.24-1:2009***.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	56	01

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	56

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	01

9.53 Property 0x57 - SHA Hash Configuration (HID Only | TLV Only, Configurable SHA Only, MSR Only)

Property ID: 0x57
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00

This property specifies whether and how the device returns a SHA-x Hash code with swipe data. See section **6.19 SHA-1 Hashed Track 2 Data (HID Only | TLV Only | GATT Only | SLIP Only, SHA-1 Only)** and section **6.21 SHA-256 Hashed Track 2 Data (SHA-256 Only)**.

The possible options are:

Value	Meaning
0x00	Device sends a SHA-1 Hash code of all Track 2 data
0x01	Device sends a SHA-1 Hash code of the Track 2 PAN
0x02	Device sends a Salted SHA-1 Hash code of all Track 2 data
0x03	Device sends a Salted SHA-1 Hash code of the Track 2 PAN
0x04	Device sends a SHA-256 Hash code of all Track 2 data (SHA-256 Only)
0x05	Device sends a SHA-256 Hash code of the Track 2 PAN (SHA-256 Only)
0x06	Device sends a Salted SHA-256 Hash code of all Track 2 data (SHA-256 Only)
0x07	Device sends a Salted SHA-256 Hash code of the Track 2 PAN (SHA-256 Only)
0xFF	Device does not send any Hash code

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	57	07

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	57

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	01

9.54 Property 0x5F - Notification Output Connection (Bluetooth LE Only | iAP Only, USB Only)

Property ID: 0x5F
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x01 (Bluetooth LE Only)
Default Value: 0x02 (iAP Only)

This property specifies which connection the device uses to send **Magnetic Stripe Card Data Sent from Device to Host** [see section **2 Connection Types**] and **Notification Messages Sent from Device to Host (Extended Notifications Only)** to the host. To immediately and temporarily override the card swipe output connection, see **Command 0x48 - Notification Output Connection Override (Bluetooth LE Only | iAP Only, USB Only**).

- 0x00 = USB connection
- 0x01 = Bluetooth LE connection (Bluetooth LE Only)
- 0x02 = iAP connection (iAP Only)

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	5F	01

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	5F

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	01

9.55 Property 0x64 - LCD Brightness (Display Only)

Property ID: `0x64`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x02

This property specifies the brightness of the LCD display, and can be set and get by the host, in addition to manual control. It can be assigned the following values:

- 0x00 = Low Brightness
- 0x01 = Medium Brightness
- 0x02 = High Brightness

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	64	01

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	64

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	01

9.56 Property 0x65 - Manual CVV Prompt (Keypad Entry Only)

Property ID: `0x65`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00

This property specifies whether the device prompts the cardholder for manual CVV entry:

- 0x00 = CVV Prompt off
- 0x01 = CVV Prompt on

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	65	01

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	65

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	01

9.57 Property 0x66 - Manual MOD10 Prompt (Keypad Entry Only)

Property ID: 0x66
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x01

This property specifies whether the device prompts the cardholder to manually enter MOD10:

- 0x00 = MOD10 Prompt off
- 0x01 = MOD10 Prompt on

If the manual MOD10 prompt is on, the device performs MOD10 calculations on manual account entry and the device returns an error if there is a MOD10 error. At that point, the cardholder or operator can cancel the entry and start over, or ignore the error and continue with the entry.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	66	01

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	66

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	01

9.58 Property 0x67 - EMV Data Encryption Variant (EMV Only)

Property ID: 0x67
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x01 (Data Variant)

This property specifies which variant of the current DUKPT Key the device uses to encrypt EMV Data.

- 0x00 = Use the **PIN Encryption** variant.
- 0x01 = Use the **Data Encryption, request or both ways** variant.

The device uses this value to determine how to create the correct Derived Key to encrypt data involved in EMV transactions (see section **5 Encryption, Decryption, and Key Management**). The algorithms for creating the Derived Key fitting each of the possible variants are fully specified in ***ANS X9.24-1:2009***.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	67	01

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	67

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	01

9.59 Property 0x69 - Auxiliary UART Configuration (Auxiliary Ports Only)

Property ID: `0x69`
Property Type: Word
Length: 2 bytes
Get Property: Yes
Set Property: Yes
Default Value: 0x0001 (9600 baud, port open)

This property controls the behavior of the auxiliary UART. For more information about the auxiliary UART, see Command Group 0x04 - Auxiliary UART (Auxiliary Ports Only, Extended Commands Only)**.**

The property consists of a two byte word that contains bit fields that control various aspects of the auxiliary UART. This two byte property should be transmitted most significant byte first.

Auxiliary UART configuration: (Bit 0 is the least significant bit)

Bit	Field Name	Value
0	Open Initial State	1 = UART port opens automatically after a power cycle or reset.
1	Auto Close USB Suspend	1 = If the UART port is open when a USB suspend event occurs, the device automatically closes the UART port, and re-opens it upon USB resume. This bit can be used to help meet USB suspend current requirements for devices that draw power from the UART port.
2	(Contact Only) Auto Close Chip Card Power On	1 = If the UART port is open when a chip card power on event occurs, the device automatically closes the UART port, and re-opens it upon chip card power off. This bit can be used to avoid exceeding the amount of current the device can supply to both the chip card and the UART port at the same time for devices that draw power from the UART port.
3	DTR Initial Level	This bit specifies the initial level of the DTR signal after a power cycle or reset occurs. 0 = DTR output signal is set low when the port is open. 1 = DTR output signal is set high when the port is open.
4	RTS Initial State	This bit specifies the initial level of the RTS signal after a power cycle or reset occurs. 0 = RTS output signal is set low when the port is open. 1 = RTS output signal is set high when the port is open.
5..7	Reserved	These bits should always be written as zeroes.
8..10	Baud Rate	This bit field specifies the baud rate of the UART port. 0 = 9600 baud 1 = 19200 baud 2 = 38400 baud 3 = 57600 baud 4 = 115200 baud.
11	Reserved	This bit should always be written as zero.

Bit	Field Name	Value
12..13	Data Parity Stop	This bit field specifies the number of data bits, parity, and the number of stop bits for the UART port. 0 = 8N1 (8 data bits, no parity, 1 stop bit) 1 = 7E1 2 = 7O1 3 = 7M1
14..15	Reserved	These bits should always be written as zeroes.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	03	69	00 01 (default)

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	69

Example Get Response (Hex)

Result Code	Data Len	Data
00	02	00 01 (default)

9.60 Property 0x6A - Auxiliary SPI Configuration (Auxiliary Ports Only)

Property ID: `0x6A`
Property Type: Word
Length: 2 bytes
Get Property: Yes
Set Property: Yes
Default Value: 0x0311 (75000 baud, DAV notify high, port open)

This property controls the behavior of the auxiliary SPI. For more information about the auxiliary SPI, see section **8.6 Command Group 0x05 - Auxiliary SPI (Auxiliary Ports Only, Extended Commands Only)**.

The property consists of a two byte word that contains bit fields that control various aspects of the auxiliary SPI. This two byte property should be transmitted most significant byte first.

Auxiliary SPI configuration (bit 0 is the least significant bit)

Bit	Field Name	Value
0	Open Initial State	1 = The SPI port opens automatically after a power cycle or reset.
1	Auto Close USB Suspend	1 = If the SPI port is open when a USB suspend event occurs, the device automatically closes the SPI port, and re-opens it upon USB resume This bit can be used to help meet USB suspend current requirements for devices that draw power from the SPI port.
2	CS Initial Level	This bit specifies the initial level of the CS (Chip Select) signal after a power cycle or reset occurs. 0 = CS output signal is set low when the port is open. 1 = CS output signal is set high when the port is open.
3	DAV Notify Low	1 = If the SPI port is open and the DAV (Data Available) input goes low, the device sends a **Notification 0x0500 - Auxiliary SPI Data Change** to the host.
4	DAV Notify High	1 = If the SPI port is open and the DAV (Data Available) input goes high, the device sends a **Notification 0x0500 - Auxiliary SPI Data Change** to the host.
5	Polarity	This bit specifies the idle state of the SPI clock signal. 0 = Clock signal low when idle. 1 = Clock signal high when idle (unsupported – always write as 0)
6	Phase	This bit specifies whether the SPI data signal is valid ***before*** or ***on*** the first SPI clock edge. 0 = SPI data signal available before the first SPI clock edge. 1 = SPI data signal available on the first SPI clock edge (unsupported – always write as 0)
7	Reserved	This bit should always be written as zero.

Bit	Field Name	Value
8..11	Baud Rate	This field specifies the baud rate of the SPI port. 0 = 9375 baud 1 = 18750 baud 2 = 37500 baud 3 = 75000 baud 4 = 150000 baud 5 = 300000 baud 6 = 600000 baud 7 = 1200000 baud 8 = 2400000 baud 9 = 4800000 baud.
12..15	Reserved	These bits should always be written as zeroes.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	03	6A	03 11 (default)

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	6A

Example Get Response (Hex)

Result Code	Data Len	Data
00	02	03 11 (default)

9.61 Property 0x6B - Key Management Scheme (Fixed Key Only)

Property ID: `0x6B`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00 (TDES DUKPT)

This property controls the key management scheme the device uses to generate keys to encrypt all data it sends to the host. When this property is set to Fixed Key, the host must use **Command 0x4E - Load Fixed Key (Fixed Key Only)** to load a 16-byte fixed key into the device. The device then uses the fixed key to encrypt all encrypted data it sends to the host. All references in this manual to the current DUKPT key should then be read to mean the single fixed key instead. For details about encryption, see section **5 Encryption, Decryption, and Key Management**.

The possible values are:

- 0x00 = **TDES DUKPT**. The device uses the current DUKPT Key to encrypt data.
- 0x01 = **Fixed Key**. The device uses the current fixed key as-is (no variant) to encrypt data.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	6B	01

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	6B

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	01

9.62 Property 0x6C - Swap Tracks 1 and 3 (Swap Tracks 1/3 Only)

Property ID: 0x6C
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00

This property can be used to swap the track 1 and track 3 magnetic stripe read head inputs. This property can be useful in cases where a solution design incorporates a head that is rotated 180 degrees from its intended use, such that track 1 is in the track 3 location and track 3 is in the track 1 location.

When this property is set to 0x00, the device does not swap tracks 1 and 3. When this property is set to 0x01, the device swaps tracks 1 and 3.

This property is not supported by most products. The property was introduced in the Dynamag / USB Encrypting IntelliHead with V5 product at firmware version ***21042840J01*** released in June 2017.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	02	6C	00

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	6C

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	00

9.63 Property 0x6D - EMV Contact Notification Configuration (Contact Only)

Property ID: `0x6D`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x02 (Card Removed)

The host uses this property to enable or disable specific EMV notification messages the device sends with **Notification 0x0300 - Transaction Status / Progress Information**. Setting a bit to 1 enables the specified notification message, and setting a bit to 0 disables it.

The following table defines each bit in the property and describes which notification message it controls.

EMV Notification Enable: (Bit 0 is the least significant bit)

Bit	Field Name	Description
0	Card Inserted	When an EMV transaction is not in progress, this setting controls whether the device sends **Notification 0x0300 - Transaction Status / Progress Information** with its event field set to **0x01 = Card Inserted** when a cardholder inserts a card into the EMV card slot. When an EMV transaction is in progress, insertion notifications are controlled using the Reporting Option field in **Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)**.
1	Card Removed	Controls whether the device sends **Notification 0x0300 - Transaction Status / Progress Information** message with its event field set to **0x08 = Card Removed** when the cardholder removes a card from the EMV card slot.
2-7	Reserved	Always set to zeroes.

For eDynamo, this property is available in firmware revisions ***1000003354E00*** (released June 2017) and later.

For mDynamo, this property is available in firmware revisions ***1000003358C00*** (released in June 2017) and later.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	6D	02

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	6D

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	02

9.64 Property 0x6F - Active USB Port (Dual USB Ports Only)

Property ID: `0x6F`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00

This property controls which USB connector the device listens to and transmits to for bidirectional data communication with the host, on devices with more than one USB connector where only one can be active at a time. It is intended to be set during manufacturing, with the value depending on whether the device has been purchased for handheld use or for permanently mounted use (such as at a point of sale). If the host attempts to set this property to a value that corresponds to a USB connector the device does not physically implement, the device returns an error and leaves the property unchanged.

- 0x00 = The device only uses the USB receptacle and no other USB connection.
- 0x01 = The device only uses the dock / stand / cradle USB connector and no other USB connection. In this mode, when the device is placed on the dock and the dock is connected to a USB host with an active data connection (not a charger), the device automatically turns off all wireless functions until an operator removes it from the dock.
- 0x02 = The device only uses the handheld host USB plug and no other USB connection. The handheld host USB plug is any type of USB plug that is designed to connect the device directly to and flush with the power / communication port on the bottom of a handheld host (such as Android or iOS tablets or smartphones), making the device a direct mechanical extension of the host.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	02	6F	00

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	6F

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	00

9.65 Property 0x70 - Head Subsystem Power State Default (PM5 Only | PM7 Only)

Property ID: 0x70
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00 (Off When Idle)

This property controls the head subsystem power default state after the device has been powered up or reset. See **Command 0x58 - Set Head Subsystem Power State (PM5 Only | PM7 Only)** for a description of what this state controls and its valid values.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Property Value
01	02	70	00

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	70

Example Get Response (Hex)

Result Code	Data Len	Property Value
00	01	00

9.66 Property 0x71 - Power Saving Timeout (PM5 Only)

Property ID: 0x71
Property Type: Byte
Length: 2 bytes
Get Property: Yes
Set Property: Yes
Default Value: 0x0A1E (10 minutes/30 minutes)

The host uses this property to adjust the device's power saving timeouts. The first byte governs the amount of time, in minutes, the device stays awake when there is no host activity or cardholder / operator activity before it automatically enters Sleep Mode. The second byte governs the amount of time, in minutes, the device will remain in Sleep Mode with no host activity or cardholder / operator activity before it powers off completely.

The host can set the Sleep Mode timeout to 0x00 (timeout disabled) or between 0x01 and 0xFF (one minute to 255 minutes).

The host can set the Power Off Mode timeout to 0x00 (power down instantly when Sleep Mode timer triggers), 0xFF (timeout disabled, remain in Sleep Mode indefinitely), or 0x01 to 0xFE (one minute to 254 minutes).

To immediately and temporarily override the card swipe output connection, see **Command 0x59 - Power Saving Timeouts Override (PM5 Only)**.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	03	71	0F20

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	71

Example Get Response (Hex)

Result Code	Data Len	Data
00	02	0F20

9.67 Property 0x72 - EMV Configuration Security (EMV Settings Unlock Only)

Property ID: 0x72
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: No (Manufacturer Only)
Default Value: 0x00

The host uses this property to change the security behavior of a subset of the device's EMV configuration commands. The non-default setting allows OEMs, acquirers, and field technicians to update EMV configuration settings that change frequently, without requiring access to the device UIK key, or network connectivity to request a signed command from a remote service, or pre-generated signed commands.

Valid property values are:

- **0x00 = Standard Behavior (UIK MAC required)**
- **0x01 = OEM Behavior (no MAC required)**

When the property is set to **OEM Behavior**, the host should transmit padding in place of the MAC when it invokes any of the affected commands, which are:

- **Extended Command 0x0305 - Modify Terminal Configuration (MAC)**
- **Extended Command 0x0307 - Modify Application Configuration (MAC)**
- **Extended Command 0x0309 - Modify Acquirer Public Key CAPK (MAC, EMV ODA Only)**
- **Extended Command 0x0312 - Modify Dynamic Reader Limits Configuration (MAC, Contactless Only)**

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	01	72	01

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	72

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	00

9.68 Property 0x73 - Application Selection Behavior (Application Selection Options Only)

Property ID: `0x73`
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00

The host uses this property to set the device's behavior for payment brand selection. See **About EMV L2 Transaction Flows (EMV Only)**.

Valid property values are:

- **0x00 = Card preference**. The device automatically chooses the application that is mutually supported by the card and the terminal, based on the priority order specified by the card. This is the default and is standard EMV transaction flow behavior.
- **0x01 = Prompt cardholder**. The device sends the host **Notification 0x0302 - Cardholder Selection Request** with Application Selection option 0x00 to request that the cardholder select from a list of available applications. After the cardholder selects an application, the host passes the selection to the device using **Extended Command 0x0302 - Cardholder Selection Result**. Note that if the device is configured to use this option, the cardholder must hold the card or contactless payment device in contact with the device for until cardholder application selection is complete, otherwise the device will report the transaction failed. For this reason, MagTek does not recommend using this setting.
- **0x02 = Default to US Common Debit**. The device automatically uses the US Common Debit application if it is available.

  This property will only affect device behavior when all of these conditions occur:

  1) The cardholder presents a card or payment device loaded with one of the following payment brand applications for US Common Debit:
  a) DNA US Common Debit AID: A0 00 00 06 20 06 20
  b) Discover US Common Debit AID: A0 00 00 01 52 40 10
  c) MasterCard US Common Debit AID: A0 00 00 00 04 22 03
  d) Visa US Common Debit AID: A0 00 00 00 98 08 40
  e) UnionPay US Common Debit AID: A0 00 00 03 33 01 01 08
  2) The device's configuration has that same AID loaded into one of the Application Settings slots described in **Appendix H EMV Terminal and Application Settings (EMV Only)**.

- **0x03 = Enhanced prompt cardholder**. The device sends the host **Notification 0x0302 - Cardholder Selection Request** with Enhanced Application Selection option 0x10 to request that the cardholder selects from a list of available applications. Each application may include additional tags such as - priority indicator, Issuer Country Code and Issuer Identification Number. After the cardholder selects an application, the host passes the selection to the device using **Extended Command 0x0302 - Cardholder Selection Result**. Note that if the device is configured to use this option, the cardholder must hold the card or contactless payment device in contact with the device for until cardholder application selection is complete, otherwise the device will report the transaction failed. For this reason, MagTek does not recommend using this setting.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	73	01

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	73

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	01

9.69 Property 0x74 - EMV Transaction Result Format (EMV Only, Conserve DUKPT Keys Only)

Property ID: 0x74
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00 (Legacy format)

The host uses this property to control the format of **Notification 0x0304 - Transaction Result Message**.

- 0x00 = **Use Legacy Format**. The device sends the message in the standard format described in **Appendix G.3.2 Transaction Result Message Format Security Level 3**.

- 0x01 = **Conserve DUKPT Keys**. The device omits the TLV data object F8 for encryption from **Transaction Result Message Format Security Level 3** if the transaction terminates without the cardholder presenting payment, which can occur if the host cancels the transaction or if the transaction times out. This prevents the device from advancing any DUKPT keys in this situation.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	74	00

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	74

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	00

9.70 Property 0x75 - Apple VAS Support (Apple VAS Only)

Property ID: 0x75
Property Type: Byte
Length: 1 byte
Get Property: Yes
Set Property: Yes
Default Value: 0x00 (Disable Apple VAS)

The host can use this property to enable / disable the device's support for Apple Value Added Service Protocol (Apple VAS).

- 0x00 = **Disable Apple VAS**. The device processes transactions without Apple VAS data.
- 0x01 = **Enable Apple VAS**. The device supports Apple VAS and places all Apple VAS tags in TLV data object FE when it sends **ARQC Message Format Security Level 3** to the host.
- 0x02 = **Enable Apple VAS**. The device supports multiple Apple VAS Data, and all Apple VAS tags in TLV data object FE when it sends **ARQC Message Format Security Level 3** to the host.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

(Apple VAS Only)
If Apple VAS is enabled by **Property 0x75 - Apple VAS Support (Apple VAS Only)** and the host invoked **Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)** with options that enable Apple VAS support for the current transaction, the ARQC message also includes Apple VAS data in TLV data object FE after TLV data object F8.

```
FE<len>/* container for Apple VAS data * /
  FF01<len>
   9F27<len><val> /*Mobile Token */
   9F2A<len><val> /*VAS Data */
  FF02<len>
   9F27<len><val> /*Mobile Token */
   9F2A<len><val> /*VAS Data */
 …
  FF05<len>
   9F27<len><val> /*Mobile Token */
   9F2A<len><val> /*VAS Data */
  FF06<len>
   9F27<len><val> /*Mobile Token */
   9F2A<len><val> /*VAS Data */
```

If the Apple device returns multiple Apple VAS data back to the reader, the reader should send Apple VAS tags to the host according to its slot number.

For example, the merchant in slot 2 and 5 are used for Apple VAS transaction. The reader should return the Apple VAS Data shown as below.

```
FE<len>/* container for Apple VAS data * /
   FF02<len>
     9F27<len><val> /*Mobile Token */
     9F2A<len><val> /*VAS Data */
   …
   FF05<len>
     9F27<len><val> /*Mobile Token */
     9F2A<len><val> /*VAS Data */
```

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	02	75	00

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	75

Example Get Response (Hex)

Result Code	Data Len	Data
00	01	00

9.71 Property 0x76 - Sound Notification Control

Property ID: 0x76
Property Type: Byte
Length: 2 bytes
Get Property: Yes
Set Property: Yes
Default Value: 00 00 (Sound Notification Enabled)

This property controls sound notifications.
Valid property values are:

- **0x00 0x00 = Beeper is Enabled.** The sound notification is on under normal events. The normal events are cancelled, successful & unsuccessful transaction.
- **0xFF 0xFF = Beeper is Disabled**. The sound notification is completely turned off under any events.
- **0x01 0x00 = Beep Suppression on Cancel Transaction.** The sound notification is turned off only when the transaction is cancelled.
- **0x02 0x00 = Beep Suppression on Successful Transaction.** The sound notification is turned off only when the transaction is successfully proceeded.
- **0x04 0x00 = Beep Suppression on Unsuccessful Transaction.** The sound notification is turned off only when transaction is not successfully proceeded.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Property ID	Data
01	03	76	FF FF

Example Set Response (Hex)

Result Code	Data Len	Data
00	00	

Example Get Request (Hex)

Cmd Num	Data Len	Property ID
00	01	76

Example Get Response (Hex)

Result Code	Data Len	Data
00	02	FF FF

Appendix A Bluetooth LE Controller Properties (Bluetooth LE Only)

The properties in the following subsections can be get and/or set using **Bluetooth LE Command 0x00 - Get Property** and **Bluetooth LE Command 0x01 - Set Property**.

A.1 Bluetooth LE Property 0x00 - Bluetooth LE Firmware ID

Bluetooth LE Property ID: 0x00
Get Property: Yes
Set Property: No
Default value: None

This is an 11 byte read-only property that identifies the firmware part number and revision for the firmware that resides in the device's Bluetooth LE controller. The first 8 bytes represent the firmware part number and the last 3 bytes represent the revision. For example, this property might be "21043029B04."

Example Get Request (Hex)

Cmd Num	Data Len	Data
46	04	01 00 00 00

Example Get Response (Hex)

Result Code	Data Len	Data
00	0E	01 01 00 32 31 30 34 33 30 32 39 42 30 34 (value "21043029B04")

A.2 Bluetooth LE Property 0x01 - Bluetooth LE Device Address

Bluetooth LE Property ID: 0x01
Get Property: Yes
Set Property: No
Default value: None

This is a 6 byte read-only property that contains the Bluetooth LE device address. The first byte contains the least significant byte of the address. This address varies with each device.

Example Get Request (Hex)

Cmd Num	Data Len	Data
46	04	01 00 00 01

Example Get Response (Hex)

Result Code	Data Len	Data
00	09	01 01 00 EC 11 A0 E5 C5 78 (value 0x78C5E5A011EC)

A.3 Bluetooth LE Property 0x02 - Bluetooth LE Device Name

Bluetooth LE Property ID: `0x02`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes

Default Value: String **<ProductName>-XXYY**, where XX is the second-to-least significant byte of the Bluetooth LE device address converted to ASCII hex, and YY is the least significant byte. For example, if the second to least significant byte of an eDynamo's Bluetooth LE device address is **0x11** and the least significant byte is **0xEC**, the Bluetooth LE device name would be **eDynamo-11EC**. To reset the device to this default, set this property using a zero-length string. Shipped (factory default) values may differ. For example, some devices may be shipped with the last five characters of the **Device Name** property set to the last five characters of the device's serial number.

This property contains the Bluetooth LE device name, which the Bluetooth LE host typically uses to present the operator with a choice of devices to interact with. If more than one device of the same name is available, MagTek recommends including a unique identifier in the device name and labeling the device accordingly so to visually distinguish one device from another.

This property can be 0 to 20 ASCII characters long. It should not contain any null characters (0x00). If set to a length of 0, the value reverts to the original default value.

Changes made to this property persist even if the device is powered off or reset. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	07	01 00 01 02 31 32 33 (value "123")

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

A.4 Bluetooth LE Property 0x03 - Configuration Revision

Bluetooth LE Property ID: `0x03`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 0

This property is a one-byte value between 0 and 255 the host can use to track the device's configuration status. For example, the host may read the default value of 0 and determine the module needs to be configured, then configure the device and set the value to 1 to indicate configuration is complete. On subsequent powerups, the host could then verify the property equals 1 before proceeding with normal operation, or perform further configuration steps and advance the property to 2.

Changes made to this property persist even if the device is powered off or reset. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Example Get Request (Hex)

Cmd Num	Data Len	Data
46	04	01 00 00 03

Example Get Response (Hex)

Result Code	Data Len	Data
00	04	01 01 00 01 (value 1)

A.5 Bluetooth LE Property 0x05 - Battery Level (PM2 Only)

Bluetooth LE Property ID: `0x05`
Get Property: Yes
Set Property: No
Default value: None

This property is a one-byte value representing the battery level between 0% and 100%. `0x00` represents the lowest safe operating voltage; `0x64` means the battery is at full voltage. When the device is powered by USB, the device always returns the maximum possible battery level, `0x64` (100%).

Example Get Request (Hex)

Cmd Num	Data Len	Data
46	04	01 00 00 05

Example Get Response (Hex)

Result Code	Data Len	Data
00	04	01 01 00 64 (value 100%)

A.6 Bluetooth LE Property 0x07 - Passkey

Bluetooth LE Property ID: `0x07`
Get Property: No
Set Property: Yes
Non-Volatile: Yes
Default value: 0 (representing passkey 000000)

This property is a four-byte integer that represents the six-decimal-digit Bluetooth LE passkey (for example, 123456). To maximize the security of the Bluetooth LE connection, the passkey should be changed to something other than its default value by an administrator. The minimum value of the property is decimal 000000, and the maximum value of the property is decimal 999999. The first byte is the least significant byte (LSB).

Changes made to this property persist even if the device is powered off or reset. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	08	01 00 01 07 3F 42 0F 00 (value 999999 decimal)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

A.7 Bluetooth LE Property 0x08 - Configuration Bits

Bluetooth LE Property ID: `0x08`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default Value: Depends on data format. See **Table 9-2**.

Table 9-2 - Configuration Bits Default Values Per Data Format

Bluetooth LE Interface Type	Default value	USB Power Not Exit Airplane Mode	Never Advertise	Normally Connectable	Use Whitelist
HID	0x02	False	False	True	False
KB	0x00	False	False	False	False
GATT	0x02	False	False	True	False

This property is a one byte value that contains configuration bits that control various Bluetooth LE features. Bits 7-2 are reserved for future use and should always be set to 0.

Bit Position	7	6	5	4	3	2	1	0
Decode Type	R	R	R	R	USB Power Not Exit Airplane Mode	Never Advertise	Normally Connectable	Use Whitelist

Bit 0 is the **Use Whitelist** bit. Not all devices allow the host to set this bit. When this bit is set, the device behaves according to Bluetooth LE standard whitelist rules, which prevents unpaired hosts that are not on the device's whitelist from connecting to the device when it is advertising. This makes the device compliant with the HID over GATT profile defined by the Bluetooth LE standard. Setting this bit is appropriate only for solutions where the Bluetooth LE host has a fixed Bluetooth LE address; Bluetooth LE hosts that use random Bluetooth LE addresses – such as iPhones and other Apple devices – will fail to reconnect, because random Bluetooth LE addresses are incompatible with whitelisting.

Bit 1 is the **Normally Connectable** bit. Not all devices allow the host to set this bit. When this bit is set, the device always advertises if it is not connected to a Bluetooth LE host, even when it has no card data to send. Because the device's advertising controls whether a Bluetooth LE host can connect, this flag effectively allows the host to connect at will. This setting should be considered carefully, because granting the Bluetooth LE host full control over the connection state can drain the device's battery, but it can be useful in specific cases:

- If the host needs to send commands over Bluetooth LE at any time, or
- If the battery drain is worth eliminating any delays generally introduced by re-connecting every time the device has card data to send.

When the Normally Connectable bit is set to 1, it usually also desirable to only have the host initiate Bluetooth LE disconnects, instead of the device. To prevent the device from disconnecting from the Bluetooth LE host automatically, set **Bluetooth LE Property 0x0B - General Connection Timeout** to 0 (Disabled).

(Custom Advertising Only) Bit 2 is the **Never Advertise** bit. When this bit is set to 1, Bluetooth LE never advertises. This effectively disables Bluetooth LE functionality for solutions that only require use of other physical connection types, such as USB. On devices that do not support it, this bit is reserved and should always be written with zero.

(Custom Advertising Only) Bit 3 is the **USB Power Not Exit Airplane Mode** bit. By default, applying USB power to the device triggers it to exit airplane mode and start advertising. Setting this bit to 1 disables this behavior. Pressing and releasing the button can still be used to exit airplane mode, regardless of how this bit is set. On devices that do not support it, this bit is Reserved and should always be written with zero.

Bits 4 to 7 are reserved. These bits should always be written with zeroes.

When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Changes made to this property persist even if the device is powered off or reset. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	05	01 00 01 08 01 (use white list bit is set)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

A.8 Bluetooth LE Property 0x0B - General Connection Timeout

Bluetooth LE Property ID: 0x0B
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default Value: Depends on data format. See **Table 9-3**.

Table 9-3 - General Connection Timeout Property Default Values Per Data Format

Bluetooth LE Interface Type	Default value
HID	0 (disabled)
KB	20000 (milliseconds)
GATT	0 (disabled)

This property is a four byte integer in least significant byte order that sets how long the device stays connected to the Bluetooth LE host when there has been no communication. The device disconnects from the Bluetooth LE host after this timeout expires. This can be used for power saving purposes. In addition, devices that use the KB connection type may prevent some hosts from displaying their virtual touch keyboards when the device is connected, so disconnecting when not in use may be desirable.

Setting this property to zero stops the device from timeout-disconnecting from the Bluetooth LE host, which causes the battery to drain more quickly.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	08	01 00 01 0B 20 4E 00 00 (20000 (0x4E20) milliseconds)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

A.9 Bluetooth LE Property 0x0C - Desired Minimum Connection Interval

Bluetooth LE Property ID: `0x0C`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 10 (12.5 milliseconds)

This property is a two byte integer in least significant byte order, in 1.25 millisecond increments, that contains the **Interval Min** value the device sends to the Bluetooth LE host in a CONNECTION PARAMETER UPDATE REQUEST (see the core Bluetooth specification for details). Only values between 6 and 3200 are valid.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	06	01 00 01 0C 0A 00 (12.5 milliseconds)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

A.10 Bluetooth LE Property 0x0D - Desired Maximum Connection Interval

Bluetooth LE Property ID: `0x0D`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 10 (12.5 milliseconds)

This property is a two byte integer in least significant byte order, in 1.25 millisecond increments, that contains the **Interval Max** value the device sends to the Bluetooth LE host in a CONNECTION PARAMETER UPDATE REQUEST (see the core Bluetooth specification for details). Only values between 6 and 3200 are valid.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	06	01 00 01 0D 0A 00 (12.5 milliseconds)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

A.11 Bluetooth LE Property 0x0E - Desired Slave Latency

Property identifier: `0x0E`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 4

This property is a two byte integer in least significant byte order that contains the **Slave Latency** value the device sends to the Bluetooth LE host in a CONNECTION PARAMETER UPDATE REQUEST (see the core Bluetooth specification for details). Only values between 0 and 499 are valid.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	06	01 00 01 0E 04 00 (value 4)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

A.12 Bluetooth LE Property 0x0F - Desired Supervision Timeout

Bluetooth LE Property ID: `0x0F`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 500 (5000 milliseconds)

This property is a two byte integer in least significant byte order, in 10 millisecond increments, that contains the **Timeout Multiplier** value the device sends to the Bluetooth LE host in a CONNECTION PARAMETER UPDATE REQUEST (see the core Bluetooth specification for details). Only values between 10 and 3200 are valid.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	06	01 00 01 0F F4 01 (5000 milliseconds)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

A.13 Bluetooth LE Property 0x11 - Bluetooth LE Connection Type (MSR Only, KB Only)

Bluetooth LE Property ID: `0x11`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 0x02 = GATT

This property is a one byte value that contains Bluetooth LE interface type. Changing this property automatically erases all bonds. Valid values are:

- 0x01 = Keyboard emulation (KB). With this option card swipe data are sent to the host as if it was typed on a Bluetooth LE keyboard. The card data has the same format as USB keyboard emulation. See section **3.3 How to Use Streaming Format**. This value is only valid on devices that support USB KB or GATT KB keyboard emulation.
- 0x02 = Vendor-defined GATT (GATT). See section **3.2 How to Use GATT Format (GATT Only)**.

On devices that only support one Bluetooth LE connection type, this property is read-only.

When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect. Because this property affects Bluetooth LE communications, it is best to change it using the USB connection.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	05	01 00 01 11 02 (GATT)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

Example Get Request (Hex)

Cmd Num	Data Len	Data
46	04	01 00 00 11

Example Get Response (Hex)

Result Code	Data Len	Data
00	04	01 01 00 02 (GATT)

A.14 Bluetooth LE Property 0x12 - Connection Parameter Update Request Control

Bluetooth LE Property ID: `0x12`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 0x01 (send connection parameter update bit is set)

This property is a one byte value whose bits control various connection parameter update features. Bits 7-1 are reserved for future use and should always be set to 0.

Bit 0 = **Send Connection Parameter Update** bit. When this bit is set to 1, the device sends a connection parameter update request once after each Bluetooth LE connection.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	05	01 00 01 12 01 (send connection parameter update bit is set)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

A.15 Bluetooth LE Property 0x13 - Bluetooth Status LED Functionality Control (Pairing Modes Only)

Bluetooth LE Property ID: `0x13`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 0x00 (Off During Bluetooth LE Connection)

This property is a one byte value that controls the Bluetooth Status LED functionality. On devices that do not have a dedicated Bluetooth Status LED, the device always returns 0x00.

When this byte is set to 0x00, the Bluetooth Status LED is OFF when the device is connected to a host via Bluetooth LE, which saves battery power.

When this byte is set to 0x01, the Bluetooth Status LED is ON when the host has established an *encrypted* Bluetooth LE connection with the device, indicating the device is accepting commands and transactions. This provides additional visual cues for cardholders and operators, but uses more battery power.

When this byte is set to 0x02, the Bluetooth Status LED is ON when the host has established *any* Bluetooth LE connection with the device. If the connection is not yet encrypted, the device does not process commands or transactions. This option can be useful for diagnosing whether a host is connected to the device. When a host is connected to the device, the device does not advertise and is not able to connect to any other host until the connection is broken.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	05	01 00 01 13 00 (off during Bluetooth LE connection)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

A.16 Bluetooth LE Property 0x15 - Pairable Timeout (Pairing Modes Only)

Bluetooth LE Property ID: `0x15`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 0x03 (3 minutes)

This property is a one byte value that controls how many minutes the device waits to be paired before exiting pairing mode. This property can be set to a value between 0 and 5 minutes. When set to 0, the device is always pairable if it is not in Airplane Mode.

When set to 0, the Bluetooth Status LED blinks on briefly every 2 seconds for one minute after the device exits airplane mode to indicate the device is pairable. After one minute, the LED turns off to conserve power, but the device remains pairable.

When set to a value between 1 and 5 minutes and when the device is not pairable, the device rejects pairing requests from any host that tries to pair with it. To make the device pairable, press the button for two seconds until the Bluetooth Status LED changes from solid on to blinking, then immediately release the button. Do not hold the button for more than one second past the three flashes, or the device resets and is not pairable. While the device is pairable, the LED blinks on briefly every 2 seconds for the number of minutes the device is pairable for.

On devices that do not support this property, the firmware behaves as if the property is set to 0. On devices that do support it, the default is now set to 3, and the firmware behavior changes accordingly when using the new default.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	05	01 00 01 15 03 (03 = 3 minutes)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

A.17 Bluetooth LE Property 0x16 - Maximum Bond Count (Pairing Modes Only)

Bluetooth LE Property ID: `0x16`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 0x09 (9 bonds)

This property is a one byte value that controls how many hosts the device remains bonded with at one time. The property can be set to a value between 1 and 9 bonds. See **Bluetooth LE Property 0x17 - Maximum Bond Mode (Pairing Modes Only)** for a description of how the device behaves when the maximum number of bonds is reached.

Changing this property automatically causes the device to erase all bonds. The operator should unpair and then re-pair with any host that was previously paired with the device before trying to connect to the device with that host.

When this property is changed, the device must be reset manually or with a command (see **Command 0x02 - Reset Device**), before using the device further with a Bluetooth LE connection. Because this property affects Bluetooth LE communication, it is best to change it using the USB connection.

On devices that do not support this property, the firmware behaves as if this property is set to 9.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	05	01 00 01 16 09 (09 = 9 bonds)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

A.18 Bluetooth LE Property 0x17 - Maximum Bond Mode (Pairing Modes Only)

Bluetooth LE Property ID: `0x17`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 0x00 (FIFO mode)

This property is a one byte value that controls how the device behaves when the maximum number of bonds [controlled by **Bluetooth LE Property 0x16 - Maximum Bond Count (Pairing Modes Only)**] is reached. This property can be set to value 0 (FIFO) or 1 (Not Pairable).

When the property is set to 0 (FIFO) and an operator attempts to pair with a new host when the device is bonded with the maximum number of Bluetooth LE hosts, the device deletes the oldest bond and continues to pair/bond with the new host.

When the property is set to 1 (Not Pairable) and the device is bonded with the maximum number of Bluetooth LE hosts, the device leaves pairing mode and can not be placed into pairing mode until all bonds have been erased using **Bluetooth LE Command 0x07 - Erase All Bonds**.

When this property is changed, the device must be reset manually or with a command (see **Command 0x02 - Reset Device**), before using the device further with a Bluetooth LE connection. Because this property affects Bluetooth LE communication, it is best to change it using the USB connection.

On devices that do not support this property, the firmware behaves as if this property is set to 0.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	05	01 00 01 17 00 (00 = FIFO)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

A.19 Bluetooth LE Property 0x18 - Minimum Background Advertising Interval (Custom Advertising Only)

Bluetooth LE Property ID: `0x18`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 32 (20 milliseconds)

This property is a two byte integer in least significant byte order, in 625 microsecond increments, that contains the **minimum background advertising interval**. This property, combined with **Bluetooth LE Property 0x19 - Maximum Background Advertising Interval (Custom Advertising Only)**, determines the Bluetooth LE advertising interval the device uses when the initial or pairing advertising interval is not in effect. Smaller advertising intervals cause the device to consume more power when advertising, which may be a concern when running on battery power.

Only values between 32 (20ms) and 16384 (10.24s) are valid. Using values outside this range causes unpredictable behavior. **Bluetooth LE Property 0x19 - Maximum Background Advertising Interval (Custom Advertising Only)** may also need to be adjusted when changing this property. If the maximum background advertising interval is less than the minimum, the device may behave unpredictably.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Because this property affects Bluetooth LE behavior, MagTek recommends only changing it using the USB interface. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	06	01 00 01 18 40 06 (1000ms / .625ms) = 1600 (0x0640)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

Example Get Request (Hex)

Cmd Num	Data Len	Data
46	04	01 00 00 18

Example Get Response (Hex)

Result Code	Data Len	Data
00	05	01 01 00 40 06 (1000ms / .625ms) = 1600 (0x0640)

A.20 Bluetooth LE Property 0x19 - Maximum Background Advertising Interval (Custom Advertising Only)

Bluetooth LE Property ID: `0x19`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 32 (20 milliseconds)

This property is a two byte integer in least significant byte order, in 625 microsecond increments, that contains the **maximum background advertising interval**. This property, combined with **Bluetooth LE Property 0x18 - Minimum Background Advertising Interval (Custom Advertising Only)**, determines the Bluetooth LE advertising interval the device uses when no other advertising interval is in effect. Smaller advertising intervals cause the device to consume more power when advertising, which may be a concern when running on battery power.

Only values between 32 (20ms) and 16384 (10.24s) are valid. Using values outside this range causes unpredictable behavior.

Bluetooth LE Property 0x18 - Minimum Background Advertising Interval (Custom Advertising Only) may also need to be adjusted when changing this property. Using a maximum background advertising interval less than the minimum causes unpredictable behavior.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Because this property affects Bluetooth LE behavior, it is recommended to only change it using the USB interface. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	06	01 00 01 19 40 06 (1000ms / .625ms) = 1600 (0x0640)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

Example Get Request (Hex)

Cmd Num	Data Len	Data
46	04	01 00 00 19

Example Get Response (Hex)

Result Code	Data Len	Data
00	05	01 01 00 40 06 (1000ms / .625ms) = 1600 (0x0640)

A.21 Bluetooth LE Property 0x1C - Minimum Initial Advertising Interval (Custom Advertising Only)

Bluetooth LE Property ID: `0x1C`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 32 (20 milliseconds)

This property is a two byte integer in least significant byte order, in 625 microsecond increments, that contains the **minimum initial advertising interval**. This property, combined with **Bluetooth LE Property 0x19 - Maximum Background Advertising Interval (Custom Advertising Only)**, determines the Bluetooth LE advertising interval the device uses for one minute after the device exits airplane mode. MagTek recommends setting both properties to the same value. Smaller advertising intervals cause the device to consume more power when advertising, which may be a concern when running on battery power.

Only values between 32 (20ms) and 16384 (10.24s) are valid. Using values outside this range causes unpredictable behavior. **Bluetooth LE Property 0x1D - Maximum Initial Advertising Interval (Custom Advertising Only)** may also need to be adjusted when changing this property. If the maximum initial advertising interval is less than the minimum, the device may behave unpredictably.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Because this property affects Bluetooth LE behavior, it is recommended to only change it using the USB interface. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	06	01 00 01 1C 40 06 (1000ms / .625ms) = 1600 (0x0640)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

Example Get Request (Hex)

Cmd Num	Data Len	Data
46	04	01 00 00 1C

Example Get Response (Hex)

Result Code	Data Len	Data
00	05	01 01 00 40 06 (1000ms / .625ms) = 1600 (0x0640)

A.22 Bluetooth LE Property 0x1D - Maximum Initial Advertising Interval (Custom Advertising Only)

Bluetooth LE Property ID: `0x1D`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 32 (20 milliseconds)

This property is a two byte integer in least significant byte order, in 625 microsecond increments, that contains the **maximum initial advertising interval**. This property, combined with **Bluetooth LE Property 0x1C - Minimum Initial Advertising Interval (Custom Advertising Only)**, determines the Bluetooth LE advertising interval the device uses for one minute after the device exits airplane mode. MagTek recommends setting both properties to the same value. Smaller advertising intervals cause the device to consume more power when advertising, which may be a concern when running on battery power.

Only values between 32 (20ms) and 16384 (10.24s) are valid. Using values outside this range causes unpredictable behavior. **Bluetooth LE Property 0x1C - Minimum Initial Advertising Interval (Custom Advertising Only)** may also need to be adjusted when changing this property. If the maximum initial advertising interval is less than the minimum, the device may behave unpredictably.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Because this property affects Bluetooth LE behavior, it is recommended to only change it using the USB interface. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	06	01 00 01 1D 40 06 (1000ms / .625ms) = 1600 (0x0640)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

Example Get Request (Hex)

Cmd Num	Data Len	Data
46	04	01 00 00 1D

Example Get Response (Hex)

Result Code	Data Len	Data
00	05	01 01 00 40 06 (1000ms / .625ms) = 1600 (0x0640)

A.23 Bluetooth LE Property 0x1E - Minimum Pairable Advertising Interval (Custom Advertising Only)

Bluetooth LE Property ID: `0x1E`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 32 (20 milliseconds)

This property is a two byte integer in least significant byte order, in 625 microsecond increments, that contains the **minimum pairable advertising interval**. This property, combined with **Bluetooth LE Property 0x1F - Maximum Pairable Advertising Interval (Custom Advertising Only)**, determines the advertising interval the device uses when it is pairable Smaller advertising intervals cause the device to consume more power when advertising, which may be a concern if the device is running on battery power.

Only values between 32 (20ms) and 16384 (10.24s) are valid. Using values outside this range causes unpredictable behavior. **Bluetooth LE Property 0x1F - Maximum Pairable Advertising Interval (Custom Advertising Only)** may also need to be adjusted when changing this property. If the maximum pairable advertising interval is less than the minimum, the device may behave unpredictably.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Because this property affects Bluetooth LE behavior, it is recommended to only change it using the USB interface. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	06	01 00 01 1E 40 06 (1000ms / .625ms) = 1600 (0x0640)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

Example Get Request (Hex)

Cmd Num	Data Len	Data
46	04	01 00 00 1E

Example Get Response (Hex)

Result Code	Data Len	Data
00	05	01 01 00 40 06 (1000ms / .625ms) = 1600 (0x0640)

A.24 Bluetooth LE Property 0x1F - Maximum Pairable Advertising Interval (Custom Advertising Only)

Bluetooth LE Property ID: `0x1F`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 32 (20 milliseconds)

This property is a two byte integer in least significant byte order, in 625 microsecond increments, that contains the **maximum pairable advertising interval**. This property, combined with **Bluetooth LE Property 0x1E - Minimum Pairable Advertising Interval (Custom Advertising Only)**, determines the advertising interval the device uses when it is pairable. Smaller advertising intervals cause the device to consume more power when advertising, which may be a concern when running on battery power.

Only values between 32 (20ms) and 16384 (10.24s) are valid. Using values outside this range causes unpredictable behavior. **Bluetooth LE Property 0x1E - Minimum Pairable Advertising Interval (Custom Advertising Only)** may also need to be adjusted when changing this property. If the maximum pairable advertising interval is less than the minimum, the device may behave unpredictably.

This property is stored in non-volatile memory, so it persists when the device is power cycled. This property would typically only be changed once during device configuration if needed. Modifying this property too many times wears out flash memory.

Because this property affects Bluetooth LE behavior, it is recommended to only change it using the USB interface. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	06	01 00 01 1F 40 06 (1000ms / .625ms) = 1600 (0x0640)

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

Example Get Request (Hex)

Cmd Num	Data Len	Data
46	04	01 00 00 1F

Example Get Response (Hex)

Result Code	Data Len	Data
00	05	01 01 00 40 06 (1000ms / .625ms) = 1600 (0x0640)

A.25 Bluetooth LE Property 0x20 - Automatic / Manual Pairing (Pairing Mode Control Only)

Bluetooth LE Property ID: `0x20`
Get Property: Yes
Set Property: Yes
Non-Volatile: Yes
Default value: 0x00 (Manual Pairing)

The host uses this property to control whether the device should always be in pairing mode without requiring manual steps, or behave normally (pairing on demand).

- 0x00 = **Manual Pairing**. An operator must activate pairing mode using the pushbutton. This is the default value and the most secure value for this property.
- 0x01 = **Automatic Pairing**. The device is in pairing mode all the time.

This property is stored in non-volatile memory, so it persists when the device is power cycled. When this property is changed, the device must be reset (see **Command 0x02 - Reset Device**) or powered off for at least 30 seconds, then powered on, before the changes will take effect.

Example Set Request (Hex)

Cmd Num	Data Len	Data
46	05	01 00 01 20 01

Example Set Response (Hex)

Result Code	Data Len	Data
00	03	01 01 00

Example Get Request (Hex)

Cmd Num	Data Len	Data
46	04	01 00 00 20

Example Get Response (Hex)

Result Code	Data Len	Data
00	04	01 01 00 01

Appendix B Warranty, Standards, and Certifications

LIMITED WARRANTY

MagTek warrants that the products sold pursuant to this Agreement will perform in accordance with MagTek's published specifications. This warranty shall be provided only for a period of one year from the date of the shipment of the product from MagTek (the "Warranty Period"). This warranty shall apply only to the "Buyer" (the original purchaser, unless that entity resells the product as authorized by MagTek, in which event this warranty shall apply only to the first repurchaser).

During the Warranty Period, should this product fail to conform to MagTek's specifications, MagTek will, at its option, repair or replace this product at no additional charge except as set forth below. Repair parts and replacement products will be furnished on an exchange basis and will be either reconditioned or new. All replaced parts and products become the property of MagTek. This limited warranty does not include service to repair damage to the product resulting from accident, disaster, unreasonable use, misuse, abuse, negligence, or modification of the product not authorized by MagTek. MagTek reserves the right to examine the alleged defective goods to determine whether the warranty is applicable.

Without limiting the generality of the foregoing, MagTek specifically disclaims any liability or warranty for goods resold in other than MagTek's original packages, and for goods modified, altered, or treated without authorization by MagTek.

Service may be obtained by delivering the product during the warranty period to MagTek (1710 Apollo Court, Seal Beach, CA 90740). If this product is delivered by mail or by an equivalent shipping carrier, the customer agrees to insure the product or assume the risk of loss or damage in transit, to prepay shipping charges to the warranty service location, and to use the original shipping container or equivalent. MagTek will return the product, prepaid, via a three (3) day shipping service. A Return Material Authorization ("RMA") number must accompany all returns. Buyers may obtain an RMA number by contacting MagTek Support Services at (562) 546-6800.

EACH BUYER UNDERSTANDS THAT THIS MAGTEK PRODUCT IS OFFERED AS-IS. MAGTEK MAKES NO OTHER WARRANTY, EXPRESS OR IMPLIED, AND MAGTEK DISCLAIMS ANY WARRANTY OF ANY OTHER KIND, INCLUDING ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

IF THIS PRODUCT DOES NOT CONFORM TO MAGTEK'S SPECIFICATIONS, THE SOLE REMEDY SHALL BE REPAIR OR REPLACEMENT AS PROVIDED ABOVE. MAGTEK'S LIABILITY, IF ANY, SHALL IN NO EVENT EXCEED THE TOTAL AMOUNT PAID TO MAGTEK UNDER THIS AGREEMENT. IN NO EVENT WILL MAGTEK BE LIABLE TO THE BUYER FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS, OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF, OR INABILITY TO USE, SUCH PRODUCT, EVEN IF MAGTEK HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR FOR ANY CLAIM BY ANY OTHER PARTY.

LIMITATION ON LIABILITY

EXCEPT AS PROVIDED IN THE SECTIONS RELATING TO MAGTEK'S LIMITED WARRANTY, MAGTEK'S LIABILITY UNDER THIS AGREEMENT IS LIMITED TO THE CONTRACT PRICE OF THIS PRODUCT.

MAGTEK MAKES NO OTHER WARRANTIES WITH RESPECT TO THE PRODUCT, EXPRESSED OR IMPLIED, EXCEPT AS MAY BE STATED IN THIS AGREEMENT, AND MAGTEK DISCLAIMS ANY IMPLIED WARRANTY, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

MAGTEK SHALL NOT BE LIABLE FOR CONTINGENT, INCIDENTAL, OR CONSEQUENTIAL DAMAGES TO PERSONS OR PROPERTY. MAGTEK FURTHER LIMITS ITS LIABILITY OF ANY KIND WITH RESPECT TO THE PRODUCT, INCLUDING NEGLIGENCE ON ITS PART, TO THE CONTRACT PRICE FOR THE GOODS.

MAGTEK'S SOLE LIABILITY AND BUYER'S EXCLUSIVE REMEDIES ARE STATED IN THIS SECTION AND IN THE SECTION RELATING TO MAGTEK'S LIMITED WARRANTY.

FCC INFORMATION

This device complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) This device may not cause harmful interference, and (2) This device must accept any interference received, including interference that may cause undesired operation.

Note: This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

Caution: Changes or modifications not expressly approved by MagTek could void the user’s authority to operate this equipment.

CANADIAN DECLARATION OF CONFORMITY

This digital apparatus does not exceed the Class B limits for radio noise from digital apparatus set out in the Radio Interference Regulations of the Canadian Department of Communications.

Le présent appareil numérique n’émet pas de bruits radioélectriques dépassant les limites applicables aux appareils numériques de la classe B prescrites dans le Réglement sur le brouillage radioélectrique édicté par le ministère des Communications du Canada.

This Class B digital apparatus complies with Canadian ICES-003.

Cet appareil numérique de la classe B est conformé à la norme NMB-003 du Canada.

INDUSTRY CANADA (IC) RSS

This device complies with Industry Canada licence-exempt RSS standard(s). Operation is subject to the following two conditions: (1) This device may not cause interference, and (2) This device must accept any interference, including interference that may cause undesired operation of the device.

Le présent appareil est conforme aux CNR d'Industrie Canada applicables aux appareils radio exempts de licence. L'exploitation est autorisée aux deux conditions suivantes: (1) L'appareil ne doit pas produire de brouillage, et (2) L'utilisateur de l'appareil doit accepter tout brouillage radioélectrique subi, même si le brouillage est susceptible d'en compromettre le fonctionnement.

CUR/UR

This product is recognized per Underwriter Laboratories and Canadian Underwriter Laboratories 1950.

CE STANDARDS

Testing for compliance with CE requirements was performed by an independent laboratory. The unit under test was found compliant with standards established for Class B devices.

EU STATEMENT

Hereby, MagTek Inc. declares that the radio equipment types **Wideband Transmission System** (802.11 wireless and Bluetooth Low Energy), and **Non-Specific Short Range Device** (contactless) are in compliance with ***Directive 2014/53/EU***. The full text of the EU declarations of conformity is available at the following internet addresses:

- https://www.magtek.com/Content/DocumentationFiles/D998200238.pdf.
- https://www.magtek.com/Content/DocumentationFiles/D998200296.pdf

AUSTRALIA / NEW ZEALAND STATEMENT

Testing for compliance with AS/NZS standards was performed by a registered and accredited laboratory. The unit under test was found compliant with standards established under AS/NZS CISPR 32 (2013), AS/NZS 4268 Table 1, Row 59 DTS 2400-2483MHz SRD (802.11), and AS/NZS 4268 (2017) Table 1, Row 43 13.553-13.567MHz (contactless reader).

UL/CSA

This product is recognized per ***UL 60950-1, 2nd Edition, 2011-12-19*** (Information Technology Equipment - Safety - Part 1: General Requirements), ***CSA C22.2 No. 60950-1-07, 2nd Edition, 2011-12*** (Information Technology Equipment - Safety - Part 1: General Requirements).

ROHS STATEMENT

When ordered as RoHS compliant, this product meets the Electrical and Electronic Equipment (EEE) Reduction of Hazardous Substances (RoHS) Directive (EU) 2015/863 amending Annex II to Directive 2011/65/EU. The marking is clearly recognizable, either as written words like “Pb-free,” “lead-free,” or as another clear symbol (Pb).

PCI STATEMENT

PCI Security Standards Council, LLC (“PCI SSC”) has approved this PIN Transaction Security Device to be in compliance with PCI SSC’s PIN Security Requirements.

When granted, PCI SSC approval is provided by PCI SSC to ensure certain security and operational characteristics important to the achievement of PCI SSC’s goals, but PCI SSC approval does not under any circumstances include any endorsement or warranty regarding the functionality, quality or performance of any particular product or service. PCI SSC does not warrant any products or services provided by third parties. PCI SSC approval does not under any circumstances include or imply any product warranties from PCI SSC, including, without limitation, any implied warranties of merchantability, fitness for purpose, or non-infringement, all of which are expressly disclaimed by PCI SSC. All rights and remedies regarding products and services which have received PCI SSC approval shall be provided by the party providing such products or services, and not by PCI SSC.

SAFETY

This product has been evaluated by multiple safety certification agencies, including Underwriters Laboratories (UL) and the United States Federal Communications Commission (FCC Class A and Class B), and is designed to protect both the user and the device. This document is written specifically to work in conjunction with these safety and integrity features to protect the user and the device. It is very important to follow all steps in the product documentation carefully, in the order in which they are described, and at the recommended times. Failure to do so could result in personal injury, and / or cause damage to the device, and / or void the product warranty.

SAFETY REQUIREMENTS

⚠CAUTION

Never do any of the following:

- DO NOT use a ground adapter plug to connect equipment to a power socket-outlet that lacks a ground connection terminal.
- DO NOT attempt any maintenance function that is not specifically described in this manual or in other ExpressCard 3000 instructional documents published by MagTek.
- DO NOT remove any of the covers or guards that are fastened with screws. There are no user-serviceable areas within these covers.
- DO NOT override or “cheat” electrical or mechanical interlock devices.
- DO NOT use EC3000 supplies or cleaning materials for other than their intended purposes.
- DO NOT operate the equipment if you or anyone else have noticed unusual noises or odors.

Consider the following before operating the ExpressCard 3000:

- Connect the EC3000 to a properly grounded AC power socket-outlet. If in doubt, have the socket-outlet checked by a qualified electrician. Improper connection of the device’s grounding conductor creates a risk of electric shock.
- Place the EC3000 on a solid surface that can safely support the device’s weight plus the weight of a person leaning against it (such as a service technician).
- Be careful when moving or relocating the device. Use proper lifting techniques.
- Use materials and supplies specifically designed for MagTek devices. Using unsuitable materials may result in poor performance, and in some cases may be hazardous.

SOFTWARE LICENSE AGREEMENT

IMPORTANT: YOU SHOULD CAREFULLY READ ALL THE TERMS, CONDITIONS AND RESTRICTIONS OF THIS LICENSE AGREEMENT BEFORE INSTALLING THE SOFTWARE PACKAGE. YOUR INSTALLATION OF THE SOFTWARE PACKAGE PRESUMES YOUR ACCEPTANCE OF THE TERMS, CONDITIONS, AND RESTRICTIONS CONTAINED IN THIS AGREEMENT. IF YOU DO NOT AGREE WITH THESE TERMS, CONDITIONS, AND RESTRICTIONS, PROMPTLY RETURN THE SOFTWARE PACKAGE AND ASSOCIATED DOCUMENTATION TO THE ADDRESS IN THIS DOCUMENT, ATTENTION: CUSTOMER SUPPORT.

TERMS, CONDITIONS, AND RESTRICTIONS

MagTek, Incorporated (the "Licensor") owns and has the right to distribute the described software and documentation, collectively referred to as the "Software."

LICENSE: Licensor grants you (the "Licensee") the right to use the Software in conjunction with MagTek products. LICENSEE MAY NOT COPY, MODIFY, OR TRANSFER THE SOFTWARE IN WHOLE OR IN PART EXCEPT AS EXPRESSLY PROVIDED IN THIS AGREEMENT. Licensee may not decompile, disassemble, or in any other manner attempt to reverse engineer the Software. Licensee shall not tamper with, bypass, or alter any security features of the software or attempt to do so.

TRANSFER: Licensee may not transfer the Software or license to the Software to another party without the prior written authorization of the Licensor. If Licensee transfers the Software without authorization, all rights granted under this Agreement are automatically terminated.

COPYRIGHT: The Software is copyrighted. Licensee may not copy the Software except for archival purposes or to load for execution purposes. All other copies of the Software are in violation of this Agreement.

TERM: This Agreement is in effect as long as Licensee continues the use of the Software. The Licensor also reserves the right to terminate this Agreement if Licensee fails to comply with any of the terms, conditions, or restrictions contained herein. Should Licensor terminate this Agreement due to Licensee's failure to comply, Licensee agrees to return the Software to Licensor. Receipt of returned Software by the Licensor shall mark the termination.

LIMITED WARRANTY: Licensor warrants to the Licensee that the disk(s) or other media on which the Software is recorded are free from defects in material or workmanship under normal use.

THE SOFTWARE IS PROVIDED AS IS. LICENSOR MAKES NO OTHER WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

Because of the diversity of conditions and hardware under which the Software may be used, Licensor does not warrant that the Software will meet Licensee specifications or that the operation of the Software will be uninterrupted or free of errors.

IN NO EVENT WILL LICENSOR BE LIABLE FOR ANY DAMAGES, INCLUDING ANY LOST PROFITS, LOST SAVINGS, OR OTHER INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE, OR INABILITY TO USE THE SOFTWARE. Licensee's sole remedy in the event of a defect in material or workmanship is expressly limited to replacement of the Software disk(s) if applicable.

GOVERNING LAW: If any provision of this Agreement is found to be unlawful, void, or unenforceable, that provision shall be removed from consideration under this Agreement and will not affect the enforceability of any of the remaining provisions. This Agreement shall be governed by the laws of the State of California and shall inure to the benefit of MagTek, Incorporated, its successors or assigns.

ACKNOWLEDGMENT: LICENSEE ACKNOWLEDGES THAT LICENSEE HAS READ THIS AGREEMENT, UNDERSTANDS ALL OF ITS TERMS, CONDITIONS, AND RESTRICTIONS, AND AGREES TO BE BOUND BY THEM. LICENSEE ALSO AGREES THAT THIS AGREEMENT SUPERSEDES ANY AND ALL VERBAL AND WRITTEN COMMUNICATIONS BETWEEN LICENSOR AND LICENSEE OR THEIR ASSIGNS RELATING TO THE SUBJECT MATTER OF THIS AGREEMENT.

QUESTIONS REGARDING THIS AGREEMENT SHOULD BE ADDRESSED IN WRITING TO MAGTEK, INCORPORATED, ATTENTION: CUSTOMER SUPPORT, AT THE ADDRESS LISTED IN THIS DOCUMENT, OR E-MAILED TO SUPPORT@MAGTEK.COM.

DEMO SOFTWARE / SAMPLE CODE: Unless otherwise stated, all demo software and sample code are to be used by Licensee for demonstration purposes only and MAY NOT BE incorporated into any production or live environment. The PIN Pad sample implementation is for software PIN Pad test purposes only and is not PCI compliant. To meet PCI compliance in production or live environments, a third-party PCI compliant component (hardware or software-based) must be used.

Appendix C Examples

This section includes direct command examples and information about using demonstration software. In addition to the examples here, source code with detailed comments is included with the demo software and can be used as a guide for custom software development.

The book ***USB Complete*** by Jan Axelson is also a good guide for software developers, especially the chapter "Human Interface Devices: Host Applications."

C.1 Command Examples

This section provides examples of command sequences and cryptographic operations. Each example shows a sequence as it actually runs, so developers of custom software can check their code against the examples step-by-step to make sure the software is parsing and computing values correctly.

C.1.1 Example: HID Device Card Swipe In Security Level 2 (HID Only, MSR Only)

This example shows the data received in a HID report for a device at **Security Level 2** [see section **2.1 How to Use USB Connections (USB Only)**].

The raw HID report is:

```
Byte   Content
   0   00 00 00 3C 25 1F 00 25 42 35 34 35 32 33 30 30 35 35 31 32
  20   32 37 31 38 39 5E 48 4F 47 41 4E 2F 50 41 55 4C 20 20 20 20
  40   20 20 5E 30 38 30 34 33 32 31 30 30 30 30 30 30 30 37 32 35
  60   30 30 30 30 30 30 3F 00 00 00 00 00 00 00 00 00 00 00 00 00
  80   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 100   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 3B
 120   35 34 35 32 33 30 30 35 35 31 32 32 37 31 38 39 3D 30 38 30
 140   34 33 32 31 30 30 30 30 30 30 30 37 32 35 30 3F 00 00 00 00
 160   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 180   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 200   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 220   00 00 00 00 00 00 00 00 00 00 00 3B 35 31 36 33 34 39 39 30
 240   38 30 30 32 30 34 34 35 3D 30 30 30 30 30 30 30 30 30 30 30
 260   30 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 280   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 300   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 320   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 340   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 360   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 380   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 400   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 420   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 440   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 460   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 480   00 00 00 00 00 00 00 00 00 00 00 00 00 00 02 00 00 00 00 00
 500   00 00 00 00 00 3C 25 1F 25 42 35 34 35 32 33 30 30 35 35 31
 520   32 32 37 31 38 39 5E 48 4F 47 41 4E 2F 50 41 55 4C 20 20 20
 540   20 20 20 5E 30 38 30 34 33 32 31 30 30 30 30 30 30 30 37 32
 560   35 30 30 30 30 30 30 3F 00 00 00 00 00 00 00 00 00 00 00 00
 580   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 600   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 620   3B 35 34 35 32 33 30 30 35 35 31 32 32 37 31 38 39 3D 30 38
 640   30 34 33 32 31 30 30 30 30 30 30 30 37 32 35 30 3F 00 00 00
 660   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 680   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 700   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 720   00 00 00 00 00 00 00 00 00 00 00 00 3B 35 31 36 33 34 39 39
 740   30 38 30 30 32 30 34 34 35 3D 30 30 30 30 30 30 30 30 30 30
 760   30 30 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 780   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 800   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 820   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
 840   00 00 00 00 00 00 00 00 00 00 00 00 3C 25 1F 36
```

The HID report can be broken down using the information in section **6 Magnetic Stripe Card Data Sent from Device to Host**, which is summarized as the **Offset** and **Usage Name** columns of **Table 9-4**. This provides a structure for organizing the raw data in the **Data** column:

Table 9-4 - Interpreting HID Data

Offset	Usage Name	Data
0	Track 1 decode status	00
1	Track 2 decode status	00
2	Track 3 decode status	00
3	Track 1 encrypted data length	3C (60 bytes, see Track 1 encrypted data below)
4	Track 2 encrypted data length	25 (37 bytes, see Track 2 encrypted data below)
5	Track 3 encrypted data length	1F (31 bytes, see Track 3 encrypted data below)
6	Card encode type (ISO/ABA)	00
7..118	Track 1 encrypted data	60 bytes, not encrypted, device is in security level 2: 25 42 35 34 35 32 33 30 30 35 35 31 32 32 37 31 38 39 5E 48 4F 47 41 4E 2F 50 41 55 4C 20 20 20 20 20 20 5E 30 38 30 34 33 32 31 30 30 30 30 30 30 30 37 32 35 30 30 30 30 30 30 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
119..230	Track 2 encrypted data	37 bytes, not encrypted, device is in security level 2: 3B 35 34 35 32 33 30 30 35 35 31 32 32 37 31 38 39 3D 30 38 30 34 33 32 31 30 30 30 30 30 30 30 37 32 35 30 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
231..342	Track 3 encrypted data	31 bytes, not encrypted, device is in security level 2: 3B 35 31 36 33 34 39 39 30 38 30 30 32 30 34 34 35 3D 30 30 30 30 30 30 30 30 30 30 30 30 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
343	Card status	00 (not used, always zero)
344..347	MagnePrint status	00 00 00 00 (not available in Security Level 2)
348	MagnePrint data length	00 (Security Level 2, no MagnePrint data)

Offset	Usage Name	Data
349..476	MagnePrint data	MagnePrint not available in Security Level 2: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
477..492	Device serial number	Not set, not filled: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
493..494	Device Encryption Status	Security Level 2, keys loaded: 00 02
495..504	DUKPT Key Serial Number (KSN) / counter	Security Level 2, not available: 00 00 00 00 00 00 00 00 00 00
505	Track 1 Masked data length	3C
506	Track 2 Masked data length	25
507	Track 3 Masked data length	1F
508..619	Track 1 Masked data	Same as encrypted data: 25 42 35 34 35 32 33 30 30 35 35 31 32 32 37 31 38 39 5E 48 4F 47 41 4E 2F 50 41 55 4C 20 20 20 20 20 20 5E 30 38 30 34 33 32 31 30 30 30 30 30 30 30 37 32 35 30 30 30 30 30 30 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
620..731	Track 2 Masked data	Same as encrypted data: 3B 35 34 35 32 33 30 30 35 35 31 32 32 37 31 38 39 3D 30 38 30 34 33 32 31 30 30 30 30 30 30 30 37 32 35 30 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
732 to 843	Track 3 Masked data	Same as encrypted data: 3B 35 31 36 33 34 39 39 30 38 30 30 32 30 34 34 35 3D 30 30 30 30 30 30 30 30 30 30 30 30 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
844..851	Encrypted Session ID	Host software didn't set, so all zeroes: 00 00 00 00 00 00 00 00
852	Track 1 Absolute data length	3C (same as above)

Offset	Usage Name	Data
853	Track 2 Absolute data length	25 (same as above)
854	Track 3 Absolute data length	1F (same as above)
855	MagnePrint Absolute data length	36 (same as above)

C.1.2 Example: Keyboard Card Swipe In Security Level 2, SureSwipe Mode (SureSwipe Only, KB Only, MSR Only)

This example shows how to interpret card data received on a device set to **Security Level 2** transmitting in streaming format (see section **3.3 How to Use Streaming Format**). All properties are set to the defaults, making this a SureSwipe format.

The incoming streaming data is:

```
Byte   Content
  0    %B5452300551227189^HOGAN/PAUL      ^08043210000000
 50    725000000?;5452300551227189=080432100000007250?+51
100    63499080020445=000000000000?
```

The information in section **2.1.4 How to Use the USB Connection in Keyboard Emulation Mode (KB Only)** and ***D99875206 TECHNICAL REFERENCE MANUAL, USB KB SURESWIPE & SWIPE READER*** provides a basic template showing the expected order of fields in the data:

```
[Tk1 SS] [Tk1 Data] [ES] [Tk2 SS] [Tk2 Data] [ES] [Tk3 SS] [Tk3 Data]
[ES] [CR]
```

Each of the Pxx elements has the default value in this configuration, so this can be re-interpreted as:

```
%[Tk1 Data]?;[Tk2 Data]?+[Tk3 Data]?<ENTER>
```

More easily read as:

```
%[Tk1 Data]?
;[Tk2 Data]?
+[Tk3 Data]?
<ENTER>
```

Using the above as a template and filling in with the received raw swipe data yields the following three tracks of data:

```
%B5452300551227189^HOGAN/PAUL      ^08043210000000725000000?
;5452300551227189=080432100000007250?
+5163499080020445=000000000000?
```

C.1.3 Example: Streaming Card Swipe In Security Level 2, Not SureSwipe (Streaming Only, MSR Only)

This example shows how to interpret card data received on a device set to **Security Level 2** transmitting in streaming format (see section **3.3 How to Use Streaming Format**), and on devices connected as keyboards, with **Property 0x1A - Keyboard SureSwipe Flags (SureSwipe Only, Streaming Only, KB Only, MSR Only)** set to `0x00` (False).

The incoming streaming data is:

```
Byte   Content
  0    %B5452000000007189^HOGAN/PAUL      ^08040000000000
 50    000000000?;5452000000007189=080400000000000000?+51
100    63000050000445=000000000000?|0200|%B54523005512271
150    89^HOGAN/PAUL      ^08043210000000725000000?|;5452
200    300551227189=080432100000007250?|+5163499080020445
250    =000000000000?||||0000000000000000||6F36||1000
```

The information in section **3.3 How to Use Streaming Format (Streaming Only)** provides a basic template showing the expected order of fields in the data:

```
[P0x1E]
[P0x20] [Tk1 SS] [Tk1 Masked Data] [ES] [P0x21]
[P0x20] [Tk2 SS] [Tk2 Masked Data] [ES] [P0x21]
[P0x20] [Tk3 SS] [Tk3 Masked Data] [ES] [P0x21]
[P0x1F]
[P0x23] [Device Encryption Status]
[P0x23] [Tk1 Encrypted Data (including TK1 SS and ES)]
[P0x23] [Tk2 Encrypted Data (including TK2 SS and ES)]
[P0x23] [Tk3 Encrypted Data (including TK3 SS and ES)]
[P0x23] [MagnePrint Status]
[P0x23] [Encrypted MagnePrint data]
[P0x23] [Device serial number]
[P0x23] [Encrypted Session ID]
[P0x23] [DUKPT Key Serial Number (KSN) / counter]
[P0x23] [Clear Text CRC]
[P0x23] [Encrypted CRC]
[P0x23] [Format Code]
[P0x22]
```

Each of the Pxx elements has the default value in this configuration, so this can be re-interpreted as:

```
%[Tk1 Masked Data]?
;[Tk2 Masked Data]?
+[Tk3 Masked Data]?
|[Device Encryption Status]
|[Tk1 Encrypted Data (including TK1 SS and ES)]
|[Tk2 Encrypted Data (including TK2 SS and ES)]
|[Tk3 Encrypted Data (including TK3 SS and ES)]
|[MagnePrint Status]
|[Encrypted MagnePrint data]
|[Device serial number]
```

```
|[Encrypted Session ID]
|[DUKPT Key Serial Number (KSN) / counter]
|[Clear Text CRC]
|[Encrypted CRC]
|[Format Code]
<ENTER>
```

Using the above as a template and filling in with the received raw swipe data yields the following:

```
%B5452000000007189^HOGAN/PAUL      ^0804000000000000000000000?
;5452000000007189=080400000000000000?
+5163000050000445=000000000000?
|0200
|%B5452300551227189^HOGAN/PAUL      ^08043210000000725000000?
|;5452300551227189=080432100000007250?
|+5163499080020445=000000000000?
|
|
|
|0000000000000000
|
|6F36
|
|1000
```

The Device Serial Number value is empty because the DSN has not been set.

The MagnePrint Status, the MagnePrint Data, the DUKPT Key Serial Number (KSN) / counter and Encrypted CRC values are empty because this device is at Security Level 2 (encryption not enabled).

When the device is set to Security Level 2, the following values are represented as ASCII characters:

- Masked Track data
- Encrypted Track data
- Format Code

All other values are represented as hexadecimal data (two ASCII characters together specify the value of a single byte).

C.1.4 Example: Swipe Decryption, HID Device In Security Level 3 or 4 (HID Only, MSR Only)

This example shows the data received in a HID report [see section **2.1 How to Use USB Connections (USB Only)**] for a device set to **Security Level 3**, KSN Count = 8. It includes steps showing how to decrypt the received data.

The raw incoming HID report is:

```
Byte   Content
  0    00 00 00 40 28 20 00 C2 5C 1D 11 97 D3 1C AA 87 28 5D 59 A8
 20    92 04 74 26 D9 18 2E C1 13 53 C0 51 AD D6 D0 F0 72 A6 CB 34
 40    36 56 0B 30 71 FC 1F D1 1D 9F 7E 74 88 67 42 D9 BE E0 CF D1
 60    EA 10 64 C2 13 BB 55 27 8B 2F 12 00 00 00 00 00 00 00 00 00
 80    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
100    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 72
120    4C 5D B7 D6 F9 01 C7 F0 FE AE 79 08 80 10 93 B3 DB FE 51 CC
140    F6 D4 83 E7 89 D7 D2 C0 07 D5 39 49 9B AA DC C8 D1 6C A2 00
160    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
180    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
200    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
220    00 00 00 00 00 00 00 00 00 00 00 76 BB 01 3C 0D FD 81 95 F1
240    6F 2F BC 50 A3 51 71 AA 37 01 31 F8 74 42 31 3E E3 64 57 B8
260    7C 87 F9 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
280    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
300    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
320    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
340    00 00 00 00 A1 05 00 00 38 47 03 57 6B C5 C2 CB 20 BC 04 C6
360    8B 5C E1 97 2A E8 9E 08 7B 1C 4D 47 D5 D0 E3 17 06 10 69 03
380    E6 0B 82 03 07 92 69 0A 57 1D B0 2D 0A 88 85 5A 35 AB B5 54
400    97 98 00 6B 42 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
420    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
440    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
460    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
480    00 00 00 00 00 00 00 00 00 00 00 00 00 00 06 FF FF 98 76 54
500    32 10 E0 00 08 3C 25 1F 25 42 35 34 35 32 30 30 30 30 30 30
520    30 30 37 31 38 39 5E 48 4F 47 41 4E 2F 50 41 55 4C 20 20 20
540    20 20 20 5E 30 38 30 34 30 30 30 30 30 30 30 30 30 30 30 30
560    30 30 30 30 30 30 30 3F 00 00 00 00 00 00 00 00 00 00 00 00
580    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
600    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
620    3B 35 34 35 32 30 30 30 30 30 30 30 30 37 31 38 39 3D 30 38
640    30 34 30 30 30 30 30 30 30 30 30 30 30 30 30 30 3F 00 00 00
660    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
680    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
700    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
720    00 00 00 00 00 00 00 00 00 00 00 00 3B 35 31 36 33 30 30 30
740    30 35 30 30 30 30 34 34 35 3D 30 30 30 30 30 30 30 30 30 30
760    30 30 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
780    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
800    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
820    00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
840    00 00 00 00 21 68 5F 15 8B 5C 6B E0 3C 25 1F 36
```

The HID report can be broken down using the information in section **6 Magnetic Stripe Card Data Sent from Device to Host**, which is summarized as the **Offset** and **Usage Name** columns of **Table 9-5**. This provides a structure for organizing the raw data in the **Data** column:

Table 9-5 - Interpreting HID Data

Offset	Usage Name	Data
0	Track 1 decode status	00
1	Track 2 decode status	00
2	Track 3 decode status	00
3	Track 1 encrypted data length	40 (64 bytes, always in multiples of 8)
4	Track 2 encrypted data length	28 (40 bytes, always in multiples of 8)
5	Track 3 encrypted data length	20 (32 bytes, always in multiples of 8)
6	Card encode type (ISO/ABA)	00
7 to 118	Track 1 encrypted data	C2 5C 1D 11 97 D3 1C AA 87 28 5D 59 A8 92 04 74 26 D9 18 2E C1 13 53 C0 51 AD D6 D0 F0 72 A6 CB 34 36 56 0B 30 71 FC 1F D1 1D 9F 7E 74 88 67 42 D9 BE E0 CF D1 EA 10 64 C2 13 BB 55 27 8B 2F 12 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
119 to 230	Track 2 encrypted data	72 4C 5D B7 D6 F9 01 C7 F0 FE AE 79 08 80 10 93 B3 DB FE 51 CC F6 D4 83 E7 89 D7 D2 C0 07 D5 39 49 9B AA DC C8 D1 6C A2 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
231 to 342	Track 3 encrypted data	76 BB 01 3C 0D FD 81 95 F1 6F 2F BC 50 A3 51 71 AA 37 01 31 F8 74 42 31 3E E3 64 57 B8 7C 87 F9 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
343	Card status	00 (not used, always zero)
344 to 347	MagnePrint status	A1 05 00 00
348	MagnePrint data length	38
349 to 476	MagnePrint data	47 03 57 6B C5 C2 CB 20 BC 04 C6 8B 5C E1 97 2A E8 9E 08 7B 1C 4D 47 D5 D0 E3 17 06 10 69 03 E6 0B 82 03 07 92 69 0A 57 1D B0 2D 0A 88 85 5A 35 AB B5 54 97 98 00 6B 42 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Offset	Usage Name	Data
477 to 492	Device serial number	(Not set, not filled) 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
493 to 494	Device Encryption Status	(Keys loaded, encrypting) 00 06
495 to 504	DUKPT Key Serial Number (KSN) / counter	FF FF 98 76 54 32 10 E0 00 08
505	Track 1 Masked data length	3C
506	Track 2 Masked data length	25
507	Track 3 Masked data length	1F
508 to 619	Track 1 Masked data	25 42 35 34 35 32 30 30 30 30 30 30 30 30 37 31 38 39 5E 48 4F 47 41 4E 2F 50 41 55 4C 20 20 20 20 20 20 5E 30 38 30 34 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
620 to 731	Track 2 Masked data	3B 35 34 35 32 30 30 30 30 30 30 30 30 37 31 38 39 3D 30 38 30 34 30 30 30 30 30 30 30 30 30 30 30 30 30 30 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
732 to 843	Track 3 Masked data	3B 35 31 36 33 30 30 30 30 35 30 30 30 30 34 34 35 3D 30 30 30 30 30 30 30 30 30 30 30 30 3F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
844 to 851	Encrypted Session ID	(Host software didn't set, so decrypts to all zeroes) 21 68 5F 15 8B 5C 6B E0
852	Track 1 Absolute data length	3C
853	Track 2 Absolute data length	25
854	Track 3 Absolute data length	1F
855	MagnePrint Absolute data length	36

To decrypt this data, the host software would first examine the KSN field `FFFF9876543210E00008`, and break it down into base key `FFFF9876543210E` and the key counter is `0x00008` (see section **6.16 DUKPT Key Serial Number** for details). The host would use this information to calculate encryption key `27F66D5244FF621E AA6F6120EDEB427F`, which is also provided in the ANSI standard documentation's examples for convenience.

There are five encrypted values: Track 1 encrypted data, track 2 encrypted data, track 3 encrypted data, encrypted MagnePrint data, and encrypted session ID. The remainder of this section details the procedure for decrypting these data values.

The track 1 encrypted data is:

```
C2 5C 1D 11 97 D3 1C AA
87 28 5D 59 A8 92 04 74
26 D9 18 2E C1 13 53 C0
51 AD D6 D0 F0 72 A6 CB
34 36 56 0B 30 71 FC 1F
D1 1D 9F 7E 74 88 67 42
D9 BE E0 CF D1 EA 10 64
C2 13 BB 55 27 8B 2F 12
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
```

Because the **Track 1 Encrypted Data Length (HID Only | GATT Only | SLIP Only)** value in the incoming data says Track 1 Encrypted data is 64 bytes long, the host software can truncate the trailing blocks:

```
Block #    Content
1          C25C1D1197D31CAA
2          87285D59A8920474
3          26D9182EC11353C0
4          51ADD6D0F072A6CB
5          3436560B3071FC1F
6          D11D9F7E74886742
7          D9BEE0CFD1EA1064
8          C213BB55278B2F12
```

Section **5 Encryption, Decryption, and Key Management** tells us to decrypt the last block first: `C213BB55278B2F12` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `E98ED0F0D1EA1064`, XOR `D9BEE0CFD1EA1064` gets `3030303F00000000`, which is the decrypted last block.

Continuing in reverse block order, `D9BEE0CFD1EA1064` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `E12DA84C41B85772`, XOR `D11D9F7E74886742` gets `3030373235303030`, which is decrypted block 7.

Continuing in reverse block order, `D11D9F7E74886742` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `0704673B0041CC2F`, XOR `3436560B3071FC1F` gets `3332313030303030`, which is decrypted block 6.

Continuing in reverse block order, 3436560B3071FC1F TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 718DF68EC04A96FF, XOR 51ADD6D0F072A6CB gets 2020205E30383034, which is decrypted block 5.

Continuing in reverse block order, 51ADD6D0F072A6CB TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 0989597B8D3373E0, XOR 26D9182EC11353C0 gets 2F5041554C202020, which is decrypted block 4.

Continuing in reverse block order, 26D9182EC11353C0 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets BF110311E7D5453A, XOR 87285D59A8920474 gets 38395E484F47414E, which is decrypted block 3.

Continuing in reverse block order, 87285D59A8920474 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets F2692820A5E12B9B, XOR C25C1D1197D31CAA gets 3035353132323731, which is decrypted block 2.

Continuing in reverse block order, C25C1D1197D31CAA TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 2542353435323330, which is decrypted block 1.

Ordering the decrypted blocks first to last yields the following. The ASCII translation on the right shows the host ignoring the final four bytes from the HEX block because the **Track 1 Absolute Data Length (HID Only | GATT Only | SLIP Only)** value in the data indicates Track 1 only contains 60 characters:

```
HEX               ASCII
2542353435323330  %B545230
3035353132323731  05512271
38395E484F47414E  89^HOGAN
2F5041554C202020  /PAUL
2020205E30383034     ^0804
3332313030303030  32100000
3030373235303030  00725000
3030303F00000000  000?
```

The resulting ASCII string for track 1 is:

```
%B5452300551227189^HOGAN/PAUL      ^08043210000000725000000?
```

The track 2 encrypted data is:

```
72 4C 5D B7 D6 F9 01 C7
F0 FE AE 79 08 80 10 93
B3 DB FE 51 CC F6 D4 83
E7 89 D7 D2 C0 07 D5 39
49 9B AA DC C8 D1 6C A2
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
```

```
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
```

Because the **Track 2 Encrypted Data Length (HID Only | GATT Only | SLIP Only)** value in the incoming data says Track 2 encrypted data is 40 bytes long, the host software can truncate the trailing blocks:

```
Block #    Data
1          724C5DB7D6F901C7
2          F0FEAE7908801093
3          B3DBFE51CCF6D483
4          E789D7D2C007D539
5          499BAADCC8D16CA2
```

Section **5 Encryption, Decryption, and Key Management** tells us to decrypt the last block first: 499BAADCC8D16CA2 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets D0BBE2E2FF07D539, XOR E789D7D2C007D539 gets 373235303F000000, which is the decrypted final block.

Continuing in reverse block order, E789D7D2C007D539 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 82EBCE61FCC6E4B3, XOR B3DBFE51CCF6D483 gets 3130303030303030, which is decrypted block 4.

Continuing in reverse block order, B3DBFE51CCF6D483 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets C9C39E4138B423A1, XOR F0FEAE7908801093 gets 393D303830343332, which is decrypted block 3.

Continuing in reverse block order, F0FEAE7908801093 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 47796C85E4CE30FF, XOR 724C5DB7D6F901C7 gets 3535313232373138, which is decrypted block 2.

Continuing in reverse block order, 724C5DB7D6F901C7 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 3B35343532333030, which is decrypted block 1.

Ordering the decrypted blocks first to last gives:

```
HEX                    ASCII
3B35343532333030       ;5452300
3535313232373138       55122718
393D303830343332       9=080432
3130303030303030       10000000
373235303F000000       7250?
```

The host software can ignore the last three bytes because the **Track 2 Absolute Data Length (HID Only | GATT Only | SLIP Only)** value in the incoming data specifies that data is 37 characters long.

The resulting ASCII string for track 2 is:

```
;5452300551227189=080432100000007250?
```

The track 3 encrypted data is:

```
76 BB 01 3C 0D FD 81 95
F1 6F 2F BC 50 A3 51 71
AA 37 01 31 F8 74 42 31
3E E3 64 57 B8 7C 87 F9
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
```

Following the same procedures described above for track 1 and track 2 yields this ASCII string for track 3:

```
;5163499080020445=000000000000?
```

The MagnePrint encrypted data is:

```
47 03 57 6B C5 C2 CB 20
BC 04 C6 8B 5C E1 97 2A
E8 9E 08 7B 1C 4D 47 D5
D0 E3 17 06 10 69 03 E6
0B 82 03 07 92 69 0A 57
1D B0 2D 0A 88 85 5A 35
AB B5 54 97 98 00 6B 42
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00
```

Because the **MagnePrint Data Length (HID Only | GATT Only | SLIP Only)** value in the incoming data says MagnePrint encrypted data is 56 bytes long, the host software can truncate the trailing blocks:

```
Block #    Data
1          4703576BC5C2CB20
2          BC04C68B5CE1972A
3          E89E087B1C4D47D5
4          D0E31706106903E6
5          0B82030792690A57
```

```
6          1DB02D0A88855A35
7          ABB5549798006B42
```

Section **5 Encryption, Decryption, and Key Management** tells us to decrypt the last block first: ABB5549798006B42 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets D3B7EDDFD3045A35, XOR 1DB02D0A88855A35 gets CE07C0D55B810000, which is the decrypted final block.

Continuing in reverse block order, 1DB02D0A88855A35 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets B52307C37D314482, XOR 0B82030792690A57 gets BEA104C4EF584ED5, which is decrypted block 6.

Continuing in reverse block order, 0B82030792690A57 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets AF4EABEE4973E402, XOR D0E31706106903E6 gets 7FADBCE8591AE7E4, which is decrypted block 5.

Continuing in reverse block order, D0E31706106903E6 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 269870C3659D905E, XOR E89E087B1C4D47D5 gets CE0678B879D0D78B, which is decrypted block 4.

Continuing in reverse block order, E89E087B1C4D47D5 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 7B8F912DAF1B3149, XOR BC04C68B5CE1972A gets C78B57A6F3FAA663, which is decrypted block 3.

Continuing in reverse block order, BC04C68B5CE1972A TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 078FD0419993F7B0, XOR 4703576BC5C2CB20 gets 408C872A5C513C90, which is decrypted block 2.

Continuing in reverse block order, 4703576BC5C2CB20 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 01000184EA10B939, which is decrypted block 1.

Ordering the decrypted blocks first to last yields:

```
HEX
01000184EA10B939
408C872A5C513C90
C78B57A6F3FAA663
CE0678B879D0D78B
7FADBCE8591AE7E4
BEA104C4EF584ED5
CE07C0D55B810000
```

The host software can ignore the last three bytes because the **MagnePrint Absolute Data Length (HID Only | TLV Only | GATT Only | SLIP Only)** value in the incoming data specifies that data is 54 characters long.

The resulting decrypted MagnePrint data is:

```
01000184EA10B939408C872A5C513C90C78B57A6F3FAA663CE0678B879D0D78B7FADBC
E8591AE7E4BEA104C4EF584ED5CE07C0D55B81
```

The Encrypted Session ID data is:

```
21 68 5F 15 8B 5C 6B E0
```

This is a simple eight byte block, so the host software can simply decrypt it with the appropriate key. `21685F158B5C6BE0` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `0000000000000000`. It contains all zeroes because the host software did not specify a session ID.

C.1.5 Example: Swipe Decryption, Streaming Mode, Device In Security Level 3 or 4 (Streaming Only, MSR Only)

This example shows the data received in streaming format for a device using streaming format that is set to **Security Level 3** or **Security Level 4**, with KSN Count = 8 (see **Command 0x09 - Get Current TDES DUKPT KSN**). It includes steps that show how to decrypt the incoming data.

The incoming streaming data is:

```
Byte   Content
  0    %B5452000000007189^HOGAN/PAUL      ^08040000000000
 50    000000000?;5452000000007189=080400000000000000?+51
100    63000050000445=000000000000?|0600|C25C1D1197D31CAA
150    87285D59A892047426D9182EC11353C051ADD6D0F072A6CB34
200    36560B3071FC1FD11D9F7E74886742D9BEE0CFD1EA1064C213
250    BB55278B2F12|724C5DB7D6F901C7F0FEAE7908801093B3DBF
300    E51CCF6D483E789D7D2C007D539499BAADCC8D16CA2|E31234
350    A91059A0FBFE627954EE21868AEE3979540B67FCC40F61CECA
400    54152D1E|A1050000|8628E664C59BBAA232BA90BFB3E6B41D
450    6F4B691E633C311CBE6EE7466B81196EC07B12648DCAC4FD7F
500    D0E212B479C60BAD8C74F82F327667||21685F158B5C6BE0|F
550    FFF9876543210E00008|B78F||0000
```

The information in section **2.1.4 How to Use the USB Connection in Keyboard Emulation Mode (KB Only)** provides a basic template showing the expected order of fields in the data:

```
[P0x1E]
[P0x20] [Tk1 SS] [Tk1 Masked Data] [ES] [P0x21]
[P0x20] [Tk2 SS] [Tk2 Masked Data] [ES] [P0x21]
[P0x20] [Tk3 SS] [Tk3 Masked Data] [ES] [P0x21]
[P0x1F]
[P0x23] [Device Encryption Status]
[P0x23] [Tk1 SS] [Tk1 Encrypted Data] [ES]
[P0x23] [Tk1 SS] [Tk2 Encrypted Data] [ES]
[P0x23] [Tk1 SS] [Tk3 Encrypted Data] [ES]
[P0x23] [MagnePrint Status]
[P0x23] [Encrypted MagnePrint data]
[P0x23] [Device serial number]
[P0x23] [Encrypted Session ID]
[P0x23] [DUKPT serial number/counter]
[P0x23] [Clear Text CRC]
[P0x23] [Encrypted CRC]
[P0x23] [Format Code]
[P0x22]
```

The device has the default configuration for each of the Pxx elements, so the host software can interpret the format above as:

```
%[Tk1 Masked Data]?
;[Tk2 Masked Data]?
+[Tk3 Masked Data]?
|[Device Encryption Status]
```

```
|[Tk1 Encrypted Data (including TK1 SS and ES)]
|[Tk2 Encrypted Data (including TK2 SS and ES)]
|[Tk3 Encrypted Data (including TK3 SS and ES)]
|[MagnePrint Status]
|[Encrypted MagnePrint data]
|[Device serial number]
|[Encrypted Session ID]
|[DUKPT Key Serial Number (KSN) / counter]
|[Clear Text CRC]
|[Encrypted CRC]
|[Format Code]
<ENTER>
```

Using the above as a template and filling in with the received raw swipe data yields the following data:

```
%B5452000000007189^HOGAN/PAUL      ^08040000000000000000000?
;5452000000007189=080400000000000000?
+5163000050000445=000000000000?
|0600
|C25C1D1197D31CAA87285D59A892047426D9182EC11353C051ADD6D0F072A6CB34365
60B3071FC1FD11D9F7E74886742D9BEE0CFD1EA1064C213BB55278B2F12
|724C5DB7D6F901C7F0FEAE7908801093B3DBFE51CCF6D483E789D7D2C007D539499BA
ADCC8D16CA2
|E31234A91059A0FBFE627954EE21868AEE3979540B67FCC40F61CECA54152D1E
|A1050000
|8628E664C59BBAA232BA90BFB3E6B41D6F4B691E633C311CBE6EE7466B81196EC07B1
2648DCAC4FD7FD0E212B479C60BAD8C74F82F327667
|
|21685F158B5C6BE0
|FFFF9876543210E00008
|B78F
|
|0000
```

The Device Serial Number value is empty because the DSN has not been set.

The Encrypted CRC value is empty because the default configuration is to send it empty.

At Security Level 3, these values are represented as ASCII characters:

- Masked Track data
- Format Code

All other values are represented as hexadecimal data (two ASCII characters together specify the value of a single byte).

To decrypt this data, the host software would first examine the KSN field `FFFF9876543210E00008`, and break it down into base key `FFFF9876543210E` and the key counter is `0x00008` (see section **6.16 DUKPT Key Serial Number** for details). The host would use this information to calculate encryption key `27F66D5244FF621E AA6F6120EDEB427F`, which is also provided in the ANSI standard documentation's examples for convenience.

There are five encrypted values: Track 1 encrypted data, track 2 encrypted data, track 3 encrypted data, encrypted MagnePrint data, and encrypted session ID. The remainder of this section details the procedure for decrypting these data values.

The track 1 encrypted data is:

Block #	Data
1	C25C1D1197D31CAA
2	87285D59A8920474
3	26D9182EC11353C0
4	51ADD6D0F072A6CB
5	3436560B3071FC1F
6	D11D9F7E74886742
7	D9BEE0CFD1EA1064
8	C213BB55278B2F12

Section **5 Encryption, Decryption, and Key Management** tells us to decrypt the last block first: `C213BB55278B2F12` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `E98ED0F0D1EA1064`, XOR `D9BEE0CFD1EA1064` gets `3030303F00000000`, which is the decrypted last block.

Continuing in reverse block order, `D9BEE0CFD1EA1064` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `E12DA84C41B85772`, XOR `D11D9F7E74886742` gets `3030373235303030`, which is decrypted block 7.

Continuing in reverse block order, `D11D9F7E74886742` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `0704673B0041CC2F`, XOR `3436560B3071FC1F` gets `3332313030303030`, which is decrypted block 6.

Continuing in reverse block order, `3436560B3071FC1F` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `718DF68EC04A96FF`, XOR `51ADD6D0F072A6CB` gets `2020205E30383034`, which is decrypted block 5.

Continuing in reverse block order, `51ADD6D0F072A6CB` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `0989597B8D3373E0`, XOR `26D9182EC11353C0` gets `2F5041554C202020`, which is decrypted block 4.

Continuing in reverse block order, `26D9182EC11353C0` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `BF110311E7D5453A`, XOR `87285D59A8920474` gets `38395E484F47414E`, which is decrypted block 3.

Continuing in reverse block order, `87285D59A8920474` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `F2692820A5E12B9B`, XOR `C25C1D1197D31CAA` gets `3035353132323731`, which is decrypted block 2.

Continuing in reverse block order, `C25C1D1197D31CAA` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `2542353435323330`, which is decrypted block 1.

The host software can ignore the last four bytes because they are all hex 0x00, and are located after the End Sentinel. Ordering the decrypted blocks first to last while ignoring the null padding at the end yields:

```
HEX               ASCII
2542353435323330  %B545230
3035353132323731  05512271
38395E484F47414E  89^HOGAN
2F5041554C202020  /PAUL
2020205E30383034     ^0804
3332313030303030  32100000
3030373235303030  00725000
3030303F00000000  000?
```

The resulting ASCII string is:

```
%B5452300551227189^HOGAN/PAUL      ^08043210000000725000000?
```

The track 2 encrypted data is:

```
Block #    Data
1          724C5DB7D6F901C7
2          F0FEAE7908801093
3          B3DBFE51CCF6D483
4          E789D7D2C007D539
5          499BAADCC8D16CA2
```

Section **5 Encryption, Decryption, and Key Management** tells us to decrypt the last block first: 499BAADCC8D16CA2 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets D0BBE2E2FF07D539, XOR E789D7D2C007D539 gets 373235303F000000, which is the decrypted last block.

Continuing in reverse block order, E789D7D2C007D539 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 82EBCE61FCC6E4B3, XOR B3DBFE51CCF6D483 gets 3130303030303030, which is decrypted block 4.

Continuing in reverse block order, B3DBFE51CCF6D483 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets C9C39E4138B423A1, XOR F0FEAE7908801093 gets 393D303830343332, which is decrypted block 3.

Continuing in reverse block order, F0FEAE7908801093 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 47796C85E4CE30FF, XOR 724C5DB7D6F901C7 gets 3535313232373138, which is decrypted block 2.

Continuing in reverse block order, 724C5DB7D6F901C7 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 3B35343532333030, which is decrypted block 1.

The host software can ignore the last four bytes because they are all hex 0x00, and are located after the End Sentinel. Ordering the decrypted blocks first to last while ignoring the null padding at the end yields:

```
HEX                   ASCII
3B35343532333030      ;5452300
3535313232373138      55122718
393D303830343332      9=080432
```

```
3130303030303030      10000000
373235303F000000      7250?
```

The resulting ASCII string for track 2 is:

```
;5452300551227189=080432100000007250?
```

The track 3 encrypted data is:

```
Block #    Data
1          E31234A91059A0FB
2          FE627954EE21868A
3          EE3979540B67FCC4
4          0F61CECA54152D1E
```

Section **5 Encryption, Decryption, and Key Management** tells us to decrypt the last block first: `0F61CECA54152D1E` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `DE0949643B57C3C4`, XOR `EE3979540B67FCC4` gets `3030303030303F00`, which is the decrypted last block.

Continuing in reverse block order, `EE3979540B67FCC4` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `CB5F4964DE11B6BA`, XOR `FE627954EE21868A` gets `353D303030303030`, which is decrypted block 3.

Continuing in reverse block order, `FE627954EE21868A` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `D32A0499226994CF`, XOR `E31234A91059A0FB` gets `3038303032303434`, which is decrypted block 2.

Continuing in reverse block order, `E31234A91059A0FB` TDES Decrypt with `27F66D5244FF621E AA6F6120EDEB427F` gets `2B35313633343939`, which is decrypted block 1.

Ordering the decrypted blocks first to last gives:

```
HEX                   ASCII
2B35313633343939      +5163499
3038303032303434      08002044
353D303030303030      3=000000
3030303030303F00      000000?
```

The host software can ignore the last byte because it is hex 0x00 and is located after the End Sentinel. The resulting ASCII string for track 3 is:

```
+5163499080020443=000000000000?
```

The MagnePrint data is:

```
Block #    Data
1          8628E664C59BBAA2
2          32BA90BFB3E6B41D
```

3	6F4B691E633C311C
4	BE6EE7466B81196E
5	C07B12648DCAC4FD
6	7FD0E212B479C60B
7	AD8C74F82F327667

Section **5 Encryption, Decryption, and Key Management** tells us to decrypt the last block first: AD8C74F82F327667 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 09162DCA11E5C60B, XOR 7FD0E212B479C60B gets 76C6CFD8A59C0000, which is the decrypted last block.

Continuing in reverse block order, 7FD0E212B479C60B TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets AE81BFA4A2C80006, XOR C07B12648DCAC4FD gets 6EFAADC02F02C4FB, which is decrypted block 6.

Continuing in reverse block order, C07B12648DCAC4FD TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets AAC8D06ACCF27E6D, XOR BE6EE7466B81196E gets 14A6372CA7736703, which is decrypted block 5.

Continuing in reverse block order, BE6EE7466B81196E TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 01D78CB7D1DAEA95, XOR 6F4B691E633C311C gets 6E9CE5A9B2E6DB89, which is decrypted block 4.

Continuing in reverse block order, 6F4B691E633C311C TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 0D2620B051231748, XOR 32BA90BFB3E6B41D gets 3F9CB00FE2C5A355, which is decrypted block 3.

Continuing in reverse block order, 32BA90BFB3E6B41D TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 41499B60A6AAD427, XOR 8628E664C59BBAA2 gets C7617D0463316E85, which is decrypted block 2.

Continuing in reverse block order, 8628E664C59BBAA2 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets 010002D4B69CD2C0, which is decrypted block 1.

Ordering the decrypted blocks first to last gives:

```
HEX
010002D4B69CD2C0
C7617D0463316E85
3F9CB00FE2C5A355
6E9CE5A9B2E6DB89
14A6372CA7736703
6EFAADC02F02C4FB
76C6CFD8A59C0000
```

The host software can ignore the last two bytes because by definition MagnePrint data is 54 bytes long:

```
010002D4B69CD2C0C7617D0463316E853F9CB00FE2C5A3556E9CE5A9B2E6DB8914A637
2C A77367036EFAADC02F02C4FB76C6CFD8A59C0000
```

The encrypted session ID data is:

```
21685F158B5C6BE0
```

As this is a simple eight byte block, we only need decrypt it with the appropriate key:
21685F158B5C6BE0 TDES Decrypt with 27F66D5244FF621E AA6F6120EDEB427F gets
0000000000000000. All zeroes is the expected value because in this example the host software did not specify a session ID.

C.1.6 Example: Configuring a Device Before Encryption Is Enabled (HID Only)

This example configures the device to use the USB-HID data format (see section **2.1.3 How to Receive Data On the USB Connection**).

```
; This script demonstrates configuration commands for HID mode.
; It assumes the device is at Security Level 2 and that the Device
; Serial Number has never been set.
00 01 10     ; Get current interface
Request      : CMND=00, LEN=01, DATA=10
Response     : RC=  00, LEN=01, DATA=01

01 02 10 00  ; Set Interface to HID
Request      : CMND=01, LEN=02, DATA=10 00
Response     : RC=  00, LEN=00, DATA=

02 00        ; Reset so changes take effect
Request      : CMND=02, LEN=00, DATA=
Response     : RC=  00, LEN=00, DATA=

Delay        : (waited 5 seconds)

00 01 10     ; Get current interface (should return 0)
Request      : CMND=00, LEN=01, DATA=10
Response     : RC=  00, LEN=01, DATA=00
```

C.1.7 Example: Configuring a Keyboard Emulation Device After Encryption Is Enabled (KB Only)

```
; This script demonstrates configuration commands for KB mode.
; It assumes the device is at Security Level 3 or 4 and that the KSN
counter
; is at 0x10.
09 00        ; Get current KSN (should be FFFF9876543210E00010)
Request      : CMND=09, LEN=00, DATA=
Response     : RC=  00, LEN=0A, DATA=FF FF 98 76 54 32 10 E0 00 10

; For this KSN counter the MAC Key is: 59598DCBD9BD6BC0
94165CE45358A057
00 01 02     ; Get current Polling Interval
Request      : CMND=00, LEN=01, DATA=02
Response     : RC=  00, LEN=01, DATA=01

; Form MAC for Set Property command
;  Message to be sent is: 01 06 02 01 nnnnnnnn (nnnnnnnn is the MAC)
;  Message to be MACd is: 0106020100000000
;  This is the simplest MAC, simply TDES encrypt the message to be
MACd with
;   the MAC Key:
;           0106020100000000 MACd with 59598DCBD9BD6BC0
94165CE45358A057
;     gets  8720CE23310961B5
;     MAC is first four bytes: 8720CE23
01 06 02 01  8720CE23   ; Set Polling Interval to 1 ms
Request      : CMND=01, LEN=06, DATA=02 01 87 20 CE 23
Response     : RC=  00, LEN=00, DATA=

00 01 1E     ; Get current Pre Card String
Request      : CMND=00, LEN=01, DATA=1E
Response     : RC=  00, LEN=00, DATA=

; Form MAC for Set Property command
;  Message to be sent is: 01 05 1E nnnnnnnn (nnnnnnnn is the MAC)
;  Message to be MACd is: 01051E0000000000
;  This is the simplest MAC, simply TDES encrypt the message to be
MACd with
;   the MAC Key:
;           01051E0000000000 MACd with 59598DCBD9BD6BC0
94165CE45358A057
;     gets  5157FCBC179B0B95
;     MAC is first four bytes: 5157FCBC
01 05 1E 5157FCBC       ; Set to ""
Request      : CMND=01, LEN=05, DATA=1E 51 57 FC BC
Response     : RC=  00, LEN=00, DATA=

00 01 1F     ; Get current Post Card String
Request      : CMND=00, LEN=01, DATA=1F
Response     : RC=  00, LEN=00, DATA=

; Form MAC for Set Property command
;  Message to be sent is: 01 05 1F nnnnnnnn (nnnnnnnn is the MAC)
;  Message to be MACd is: 01051F0000000000
;  This is the simplest MAC, simply TDES encrypt the message to be
MACd with
;   the MAC Key:
;           01051F0000000000 MACd with 2B5F01F4F0CCFAEA
639D523231BFE4A2
;     gets  4885838CCC672376
;     MAC is first four bytes: 4885838C
01 05 1F 4885838C       ; Set to ""
Request      : CMND=01, LEN=05, DATA=1F 48 85 83 8C Response     : RC=
00, LEN=00, DATA=
Response     : RC=  00, LEN=00, DATA=

00 01 20     ; Get current Pre Track String
Request      : CMND=00, LEN=01, DATA=20
Response     : RC=  00, LEN=00, DATA=

; Form MAC for Set Property command
;  Message to be sent is: 01 05 20 nnnnnnnn (nnnnnnnn is the MAC)
;  Message to be MACd is: 0105200000000000
;  This is the simplest MAC, simply TDES encrypt the message to be
MACd with
;   the MAC Key:
;           0105200000000000 MACd with  9CF640F279C251E6
15F725EEEAC234AF
;     gets  442A09E6588BBF04
;     MAC is first four bytes: 442A09E6
01 05 20 442A09E6   ; Set to ""
Request      : CMND=01, LEN=05, DATA=20 44 2A 09 E6
Response     : RC=  00, LEN=00, DATA=

00 01 21     ; Get current Post Track String
Request      : CMND=00, LEN=01, DATA=21
Response     : RC=  00, LEN=00, DATA=

; Form MAC for Set Property command
;  Message to be sent is: 01 05 21 nnnnnnnn (nnnnnnnn is the MAC)
;  Message to be MACd is: 0105210000000000
;  This is the simplest MAC, simply TDES encrypt the message to be
MACd with
;   the MAC Key:
;           0105210000000000 MACd with C3DF489FDF11ACB4
F03DE97C27DCB32F
;     gets  1FA9A44C703099E1
;     MAC is first four bytes: 1FA9A44C
01 05 21 1FA9A44C     ; Set to ""
Request      : CMND=01, LEN=05, DATA=21 1F A9 A4 4C
Response     : RC=  00, LEN=00, DATA=

00 01 22     ; Get current Termination String
Request      : CMND=00, LEN=01, DATA=22
Response     : RC=  00, LEN=01, DATA=0D
```

```
; Form MAC for Set Property command
;  Message to be sent is: 01 06 22 0D nnnnnnnn (nnnnnnnn is the MAC)
;  Message to be MACd is: 0106220D00000000
;  This is the simplest MAC, simply TDES encrypt the message to be
MACd with
;   the MAC Key:
;           0106220D00000000 MACd with 6584885077214CF1
4737FA93F92334D2
;     gets  381AD461F2BDC522
;     MAC is first four bytes: 381AD461
01 06 22 0D 381AD461   ; Set to "<ENTER>"
Request      : CMND=01, LEN=06, DATA=22 0D 38 1A D4 61
Response     : RC=  00, LEN=00, DATA=

00 01 2C     ; Get current Format Code
Request      : CMND=00, LEN=01, DATA=2C
Response     : RC=  00, LEN=05, DATA=31 FF FF FF FF

; Form MAC for Set Property command
;  Message to be sent is: 01 09 2C 31303030 nnnnnnnn (nnnnnnnn is the
MAC)
;  Message to be MACd is: 01092C3130303000
;  This is the simplest MAC, simply TDES encrypt the message to be
MACd with
;   the MAC Key:
;           01092C3130303000 MACd with E161D1956A6109D2
F37AFD7F9CC3969A
;     gets  D153861529E88020
;     MAC is first four bytes: D1538615
01 09 2C 31303030 D1538615 ; Set to "1000"
Request      : CMND=01, LEN=09, DATA=2C 31 30 30 30 D1538615
Response     : RC=  00, LEN=00, DATA=

02 00        ; Reset so changes take effect
Request      : CMND=02, LEN=00, DATA=
Response     : RC=  00, LEN=00, DATA=

Delay        : (waited 5 seconds)
00 01 02     ; Get current Polling Interval (should return 01)
Request      : CMND=00, LEN=01, DATA=02
Response     : RC=  00, LEN=01, DATA=01

00 01 1E     ; Get current Pre Card String (should return "")
Request      : CMND=00, LEN=01, DATA=1E
Response     : RC=  00, LEN=00, DATA=

00 01 1F     ; Get current Post Card String (should return "")
Request      : CMND=00, LEN=01, DATA=1F
Response     : RC=  00, LEN=00, DATA=

00 01 20     ; Get current Pre Track String (should return "")
Request      : CMND=00, LEN=01, DATA=20
```

```
Response     : RC=  00, LEN=00, DATA=

00 01 21     ; Get current Post Track String (should return "")
Request      : CMND=00, LEN=01, DATA=21
Response     : RC=  00, LEN=00, DATA=

00 01 22     ; Get current Termination String (should return
"<ENTER>")
Request      : CMND=00, LEN=01, DATA=22
Response     : RC=  00, LEN=01, DATA=0D

00 01 2C     ; Get current Format Code
Request      : CMND=00, LEN=01, DATA=2C
Response     : RC=  00, LEN=04, DATA=31 30 30 30
```

C.1.8 Example: Changing from Security Level 2 to Security Level 3

```
; This script demonstrates changing from Security Level 2 to Security
Level 3.
; It assumes the device is at Security Level 2 with the ANS X9.24
Example
; key loaded and the KSN counter set to 1.
09 00         ; Get current KSN (should be FFFF9876543210E00001)
Request       : CMND=09, LEN=00, DATA=
Response      : RC=  00, LEN=0A, DATA=FF FF 98 76 54 32 10 E0 00 01

; For KSN 1, MAC Key: 042666B4918430A3 68DE9628D03984C9
;
; The command to change Security Level looks like: 15 05 03 nnnnnnnn
;  where nnnnnnnn is the MAC.
;
; The data to be MACd is: 15 05 03
; Data to be MACd must be in blocks of eight bytes, so we left justify
and
; zero fill the block to get: 15 05 03 00 00 00 00 00 (This is the
block to MAC)
; For convenience show it as the compacted form: 1505030000000000
;
; The MAC algorithm run with this data uses the following
cryptographic
; operations:
;
;  Single DES Encrypt the data to be MACd with the left half of the
MAC Key:
;     1505030000000000 1DES Enc with 042666B4918430A3 =
BFBA7AE4C1597E3D
;
;  Single DES Decrypt the result with the right half of the MAC Key:
;     BFBA7AE4C1597E3D 1DES Dec with 68DE9628D03984C9 =
DA91AB9A8AD9AB4C
;
;  Single DES Encrypt the result with the left half of the MAC Key:
;     DA91AB9A8AD9AB4C 1DES Enc with 042666B4918430A3 =
E7E2FA3882BB386C
;
; The leftmost four bytes of the final result are the MAC = E7E2FA38
;
; Send the MACd Set Security Level command
15 05 03 E7E2FA38
Request       : CMND=15, LEN=05, DATA=03 E7 E2 FA 38
Response      : RC=  00, LEN=00, DATA=

02 00         ; Reset so changes take effect
Request       : CMND=02, LEN=00, DATA=
Response      : RC=  00, LEN=00, DATA=

Delay         : (waited 5 seconds)
09 00         ; Get current KSN (should be FFFF9876543210E00002)
Request       : CMND=09, LEN=00, DATA=
Response     : RC=  00, LEN=0A, DATA=FF FF 98 76 54 32 10 E0 00 02

15 00        ; Get current Security Level (Should be 03)
Request      : CMND=15, LEN=00, DATA=
Response     : RC=  00, LEN=01, DATA=03
```

C.1.9 Example: Changing from Security Level 2 to Security Level 4 (MSR Only)

```
; This script demonstrates changing from Security Level 2 to Security
Level 4.
; It assumes the device is at Security Level 2 with the ANS X9.24
Example
; key loaded and the KSN counter set to 1.
09 00         ; Get current KSN (should be FFFF9876543210E00001)
Request       : CMND=09, LEN=00, DATA=
Response      : RC=  00, LEN=0A, DATA=FF FF 98 76 54 32 10 E0 00 01

; For KSN 1, MAC Key: 042666B4918430A3 68DE9628D03984C9
;
; The command to change Security Level looks like: 15 05 04 nnnnnnnn
;  where nnnnnnnn is the MAC.
;
; The data to be MACd is: 15 05 04
; Data to be MACd must be in blocks of eight bytes, so we left justify
and
; zero fill the block to get: 15 05 04 00 00 00 00 00 (This is the
block to MAC)
; For convenience show it as the compacted form: 1505040000000000
;
; The MAC algorithm run with this data uses the following
cryptographic
; operations:
;
;  Single DES Encrypt the data to be MACd with the left half of the
MAC Key:
;     1505040000000000 1DES Enc with 042666B4918430A3 =
644E76C88FFA0044
;
;  Single DES Decrypt the result with the right half of the MAC Key:
;     644E76C88FFA0044 1DES Dec with 68DE9628D03984C9 =
DEAC363779906C06
;
;  Single DES Encrypt the result with the left half of the MAC Key:
;     DEAC363779906C06 1DES Enc with 042666B4918430A3 =
2F38A60E3F6AD6AD
;
; The leftmost four bytes of the final result are the MAC = 2F38A60E
;
; Send the MACd Set Security Level command
15 05 04 2F38A60E
Request       : CMND=15, LEN=05, DATA=04 2F 38 A6 0E
Response      : RC=  00, LEN=00, DATA=

02 00         ; Reset so changes take effect
Request       : CMND=02, LEN=00, DATA=
Response      : RC=  00, LEN=00, DATA=

Delay         :  (waited 5 seconds)
09 00         ; Get current KSN (should be FFFF9876543210E00002)
Request       : CMND=09, LEN=00, DATA=
Response     : RC=  00, LEN=0A, DATA=FF FF 98 76 54 32 10 E0 00 02

15 00        ; Get current Security Level (Should be 04)
Request      : CMND=15, LEN=00, DATA=
Response     : RC=  00, LEN=01, DATA=04
```

C.1.10 Example: Changing from Security Level 3 to Security Level 4 (MSR Only)

```
; This script demonstrates changing from Security Level 3 to Security
Level 4.
; It assumes the device is at Security Level 3 with the ANS X9.24
Example
; key loaded and the KSN counter set to 2.
09 00         ; Get current KSN (should be FFFF9876543210E00002)
Request       : CMND=09, LEN=00, DATA=
Response      : RC=  00, LEN=0A, DATA=FF FF 98 76 54 32 10 E0 00 02

; For KSN 2, MAC Key: C46551CEF9FDDBB0 AA9AD834130DC4C7
;
; The command to change Security Level looks like: 15 05 04 nnnnnnnn
;  where nnnnnnnn is the MAC.
;
; The data to be MACd is: 15 05 04
; Data to be MACd must be in blocks of eight bytes, so we left justify
and
; zero fill the block to get: 15 05 04 00 00 00 00 00 (This is the
block to MAC)
; For convenience show it as the compacted form: 1505040000000000
;
; The MAC algorithm run with this data uses the following
cryptographic
; operations:
;
;  Single DES Encrypt the data to be MACd with the left half of the
MAC Key:
;     1505040000000000 1DES Enc with C46551CEF9FDDBB0 =
735323A914B9482E
;
;  Single DES Decrypt the result with the right half of the MAC Key:
;     735323A914B9482E 1DES Dec with AA9AD834130DC4C7 =
390E2E2AC8CB4EE6
;
;  Single DES Encrypt the result with the left half of the MAC Key:
;     390E2E2AC8CB4EE6 1DES Enc with C46551CEF9FDDBB0 =
D9B7F3D8064C4B26
;
; The leftmost four bytes of the final result are the MAC = D9B7F3D8
;
; Send the MACd Set Security Level command
15 05 04 D9B7F3D8
Request       : CMND=15, LEN=05, DATA=04 D9 B7 F3 D8
Response      : RC=  00, LEN=00, DATA=

02 00         ; Reset so changes take effect
Request       : CMND=02, LEN=00, DATA=
Response      : RC=  00, LEN=00, DATA=

Delay         : (waited 5 seconds)
09 00         ; Get current KSN (should be FFFF9876543210E00003)
Request       : CMND=09, LEN=00, DATA=
Response      : RC=  00, LEN=0A, DATA=FF FF 98 76 54 32 10 E0 00 03

15 00         ; Get current Security Level (Should be 04)
Request       : CMND=15, LEN=00, DATA=
Response      : RC=  00, LEN=01, DATA=04
```

C.1.11 Example: Authentication (MSR Only)

In this example, the device is already in **Security Level 3** or **Security Level 4**. The script puts the device into Authenticated Mode, leaves it in that mode for a time, then deactivates it.

```
; This example demonstrates the Authentication Sequence.
; It is not scripted, some of the data is deliberately randomized.
This
; makes it impossible for a simple script to produce the correct
results.
; As an example it shows all the steps in authentication and
deactivation.

; It assumes the device is at Security Level 4, with the DUKPT KSN
;  counter set to 2.

09 00        ; Get current KSN (should be FFFF9876543210E00002)

; Send the Activate Authenticated Mode command (4 minutes)
10 02 00F0
Request      : CMND=10, LEN=02, DATA=00 F0
Response     : RC=  00, LEN=1A, DATA=FF FF 98 76 54 32 10 E0 00 03 AA
AA AA AA AA AA AA AA DD DD DD DD DD DD DD DD
                                     |------- Current KSN -------| |--
-- Challenge 1 ----| |---- Challenge 2 ----|
Response     : RC=  00, LEN=1A, DATA=FF FF 98 76 54 32 10 E0 00 03 BE
5C 98 35 17 7E 45 2A A7 2D 2D B2 36 BF 29 D2
;  Challenge 1 Encrypted: BE5C9835177E452A
;  Challenge 2 Encrypted: A72D2DB236BF29D2

; Note that the KSN now ends with a counter of 3!
; Decrypt Challenge 1 using variant of Current Encryption Key
;  (Current Encryption Key XOR with F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0F0)
;
;  Current Key   0DF3D9422ACA561A 47676D07AD6BAD05
;          XOR   F0F0F0F0F0F0F0F0 F0F0F0F0F0F0F0F0
;            =   FD0329B2DA3AA6EA B7979DF75D9B5DF5
;
;     BE5C9835177E452A TDES Decrypt with FD0329B2DA3AA6EA
B7979DF75D9B5DF5 = 7549AB6EB4840003
;
;  Note that the final two bytes of the result = 0003, matching the
KSN as
;   transmitted in the clear.  This provides Authentication to the
host that
;   the device is what it claims to be (proves key knowledge).
;
; Decrypt Challenge 2 using Current Encryption Key variant as above
;     A72D2DB236BF29D2 TDES Decrypt with FD0329B2DA3AA6EA
B7979DF75D9B5DF5 = 34DB9230698281B4
;
;
; Build an Activation Challenge Reply command (cmd, len, cryptogram)
;  11 08 XXXXXXXXXXXXXXXX
```

```
;
;  The clear text input for the cryptogram is composed of the first
six bytes
;  of the decrypted Challenge 1 followed by two bytes specifying how
long to
;  stay in the Authenticated Mode.
;
;     CCCCCCCCCCCC TTTT
;
;     Time examples:
;        For 30 seconds use 001E
;        For 99 seconds use 0063
;        For 480 seconds use 01E0
;        For 1200 seconds use 04B0
;
;  These values are concatenated to form an eight byte block, we will
use 480
;  seconds:
;
;     CCCCCCCCCCCC01E0
;
;  The block is encrypted using a variant of the Current Encryption
Key
;  (Current Encryption Key XOR with 3C3C3C3C3C3C3C3C3C3C3C3C3C3C3C3C)
;
;  Current Key   0DF3D9422ACA561A 47676D07AD6BAD05
;          XOR   3C3C3C3C3C3C3C3C 3C3C3C3C3C3C3C3C
;            =   31CFE57E16F66A26 7B5B513B91579139
;
;     7549AB6EB48401E0 TDES Enc with 31CFE57E16F66A26 7B5B513B91579139
= A30DDE3BFD629ACD
;
; Send the Activation Challenge Reply Command
11 08 A30DDE3BFD629ACD

; Build a Deactivate Authenticated Mode command (cmd, len, cryptogram)
;  12 08 XXXXXXXXXXXXXXXX
;
;  The clear text input for the cryptogram is composed of the first
seven bytes
;  of the decrypted Challenge 2 followed by one byte specifying
whether to
;  increment the DUKPT KSN or not (00 = no increment, 01 = increment).
;
;     DDDDDDDDDDDDDD II
;
;  These values are concatenated to form an eight byte block, we will
specify
;  No Increment:
;
;     DDDDDDDDDDDDDD00
;
```

```
;  The block is encrypted using a variant of the Current Encryption
Key
;  (Current Encryption Key XOR with 3C3C3C3C3C3C3C3C3C3C3C3C3C3C3C3C)
;
;  Current Key   0DF3D9422ACA561A 47676D07AD6BAD05
;          XOR   3C3C3C3C3C3C3C3C 3C3C3C3C3C3C3C3C
;            =   31CFE57E16F66A26 7B5B513B91579139
;
;     34DB923069828100 TDES Enc with 31CFE57E16F66A26 7B5B513B91579139
= CA CB BD 5F 58 D5 C9 50
;
; Send the Deactivate Authenticated Mode command
12 08 CACBBD5F58D5C950
```

C.2 Transaction Validation Sequence Example (Transaction Validation Only)

A Transaction Validation sequence (see **8.3.9 Command 0x10 - Activate Authenticated Mode**) where the cardholder confirms (Green button) would follow these steps:

1) The host determines the current TDES DUKPT Key Serial Number either by using an internal counter from the previous transaction or by calling **Command 0x09 - Get Current TDES DUKPT KSN**.
2) The host software sends a secured display message with a time period, message, and challenge, using **Command 0x31 - Display Transaction Validation Information (MAC, Transaction Validation Only)** with the following message and challenge `0x0102040810204080`:

```
" Verify Details"
"1234567890123456" & "$        3500.99"
"     Accept     " & "  Press Green   "
"     Reject     " & "   Press Red    "
```

3) Device decrypts and displays, alternating (about 1.5 seconds each) among:
   - `" Verify Details "`
   - `"1234567890123456" & "$        3500.99"`
   - `"     Accept     " & "  Press Green   "`
   - `"     Reject     " & "   Press Red    "`
4) Cardholder presses the **Approve** (Green) button.
5) Device XORs 0x415050524F564544 ("Approved") with challenge 0x0102040810204080 to get 0x4052545A5A7605C4, encrypts this, and sends a fake swipe with the Transaction Validation Information Available bit set to 1 in the **Device Encryption Status**.
6) Device displays `"    Approved    "` for two seconds and ends the sequence.
7) Host sends **Command 0x33 - Get Transaction Validation Result** and decrypts the result to find the "APPROVED" status.

C.3 TLV Examples (TLV Only)

C.3.1 Example: Sending a Command Using TLV (TLV Only)

This section provides an example of a command sent from the host to the device using the TLV format described in section **3.5 How to Use Tag-Length-Value (TLV) Format**.

The native Retrieve Discovery Information command is sent as follows:

```
0xC1,0x02,0x06,0xC2,0x05,0x03,0x84,0x09,0x00
```

Where:
`0xC1,0x02,0x06` is TLV data object `C102` (Host to Device Request) and `0x06` is the length of the rest of the TLV request.

`0xC2,0x05,0x03` is TLV data object `C205` (Device Standard commands) and `0x03` is the length of the rest of the TLV request.

`0x84,0x09,0x00` is TLV data object `8409` (Retrieve Discovery Information) and `0x00` is the length of the associated data (this command requires no data, thus a length of 0).

The response to this command might be received as:

```
C10481CAC20B81C6C30681C2C307658102008103OB32313034333031333435A3481040D
```

```
C10481CAC20B81C6C30681C2C3076581020081030B32313034333031333435A3481040D
```

This would be interpreted as:

```
C104 81CA (Device to Host Response, length OK)
     C20B 81C6 (Device Standard Responses, length OK)
          C306 81C2 (Discovery, length OK)
               C307 65  (Device Information, length OK)
                    8102 00  (Device SN - MagTek)
                    8103 0B 32313034333031333435A34  (Firmware Part Number /
Rev)
                    8104 0D 4368617365207544796E616D6F  (Device Model
Name)
                    8109 02 0003  (TLV Version)
               C308 3C  (Device Capabilities, length OK)
                    8120 01 01  (Capability - MSR)
                    8121 01 07  (Capability - Tracks)
                    8122 02 0001  (Capability - Magnetic Stripe Encryption)
                    8500 01 02  (Battery Powered, Rechargeable)
                    8501 01 01  (Supports Firmware Upgrade)
                    8502 01 03  (Track Decode)
                    8503 01 03  (Supported TDES DUKPT Key Variants)
                    8504 01 01  (Supports Hash Code)
                    8505 01 01  (Capability - MagnePrint)
```

```
            8506 01 01 (Capability - MagnePrint Encryption)
            8507 01 01 (Capability - MagneSafe 2.0 Encryption)
            8508 01 01 (Supports Card Swipe Counter)
            8509 01 01 (Supports Session ID)
            C309 04 (MagneSafe V5 Specific Capabilities, len OK)
                8700 01 01 (MagneSafe V5 Masked Track Support)
        C303 0B (Device Status, length OK)
            8140 01 00 (Battery Level)
            8141 04 00000016 (Card Swipe Count)
        C304 49 (Device Configuration, length OK)
            8600 01 01 (Firmware Upgrade Enabled)
            8601 01 01 (MSR Enabled)
            8602 01 07 (Tracks Enabled)
            860C 01 03 (MSR Decode Options)
            8603 01 00 (Magnetic Stripe Encryption Enabled)
            8604 01 01 (MagnePrint Enabled)
            8605 01 00 (MagnePrint Encryption Enabled)
            8606 01 00 (MagneSafe 2.0 Encryption Enabled)
            8607 01 00 (MSR Data Encryption Variant)
            8608 01 00 (MagnePrint Data Encryption Variant)
            8609 01 01 (Hash Code Enabled)
            C30A 1A (MagneSafe V5 Specific Configuration, length OK)
                8800 06 303430343059 (MagneSafe V5 ISO Mask Property)
                8801 06 303430343059 (MagneSafe V5 AAMVA Mask)
                8802 01 00 (MagneSafe V5 Mask Other Cards Property)
                8803 01 00 (MagneSafe V5 Send Clear AAMVA Card Data property)
```

C.3.2 Example: Receiving TLV Data from the Device (TLV Only)

This section provides an example of card data being transmitted from the device to the host (as defined in section **6 Magnetic Stripe Card Data Sent from Device to Host**) in the TLV format defined in section **3.5 How to Use Tag-Length-Value (TLV) Format**.

TLV swipe data is transmitted from the device to the host looking something like this:

```
C1068201FCC3022E810902000381030B32313034333031333435A3381040D4368617365
207544796E616D6F8140015081410400000001C2010F800102020682610100826203000
0000C2028982213C254235343532303030303030303030373138395E484F47414E2F5041
554C202020202020205E30383034303030303030303030303030303030303030303F8222
253B35343532303030303030303030373138393D30383034303030303030303030303030
30303F82231F3B35313633303030303530303030343434353D3030303030303030303030
303FC20382012881020F42313233343536313230373032414183010AFFFF9876543210
E00131830A40EABD119A27D8DF7EB94F42C5CF35D48F2BDE6D847109093C56DB917F4A
3402FD0E16B9255460DAA6FE3E92A8D78C3EAD028F2E2716E5E15601365F06B9F90DA3
830B28FA2C634B4B1481E67C4DBCC4F4093FD55A203D3D282A662080813B7950D8F82F
2CD3DC6F90945E29830C2099A5DFB8CE4D8AB159A386C182835A421F964045A27B0188
D617558BB804E323830E0461401000830D40BB5B4D8210EE5C25A05816F3CD519A9A03
B7802C2F0BC63D859AC65E9A57E45CFF105477FF36382B168996C86336DBCBB669E0C6
1E134C9701365F06B9F90DA3830820BD7D8EB92C55D4E6D1096DBA34D0E155DAE3E8D7
EADE275E11366B9DA3830B28830908C63B1467CCC493FD
```

This would be interpreted as:

```
C106 8201FC  (MagneSafe V5 Card Swipe Data, length OK)
        C302 2E (Supplemental Info, length OK)
                8109 02 0003 (TLV Version, length OK)
                8103 0B 32313034333031333435A33 (Device FW Part #, length OK)
                8104 0D 4368617365207544796E616D6F (Device Model Name, length OK)
                8140 01 50 (Battery % Level, 80%, length OK)
                8141 04 00000001 (Card Swipe Count, length OK)
        C201 0F (Swipe Status, length OK)
                8001 02 0206 (Operation Status, length OK)
                8261 01 00 (Card Status, length OK)
                8262 03 000000 (Track Status, length OK)
        C202 89 (Magnetic Stripe Local Merchant Data, length OK)
                8221 3C (Track 1 Data, Masked, length OK)
                        25423534353230303030303030303731
                        38395E484F47414E2F5041554C202020
                        2020205E303830343030303030303030
                        3030303030303030303030303F
                8222 25  (Track 2 Data, Masked, length OK)
                        3B353435323030303030303030373138
                        393D3038303430303030303030303030
                        303030303F
                8223 1F  (Track 3 Data, Masked, length OK)
                        3B353136333030303035303030303434
                        353D3030303030303030303030303F
        C203 820128 (Magnetic Stripe Secure Data, length OK)
                8102 0F 423132333435363132303730324141  (Device SN - MagTek, length
OK)
```

```
8301 0A FFFF9876543210E00131 (Key Identifier, Mag Stripe Data, length OK)
830A 40 (Encrypted Track 1 Data, raw, length OK)
        EABD119A27D8DF7EB94F42C5CF35D48F
        2BDE6D847109093C56DB917F4A3402FD
        0E16B9255460DAA6FE3E92A8D78C3EAD
        028F2E2716E5E15601365F06B9F90DA3
830B 28 (Encrypted Track 2 Data, raw, length OK)
        FA2C634B4B1481E67C4DBCC4F4093FD5
        5A203D3D282A662080813B7950D8F82F
        2CD3DC6F90945E29
830C 20 (Encrypted Track 3 Data, raw, length OK)
        99A5DFB8CE4D8AB159A386C182835A42
        1F964045A27B0188D617558BB804E323
830E 04 61401000 (MagnePrint Status, length OK)
830D 40 (Encrypted MagnePrint Data, raw, length OK)
        BB5B4D8210EE5C25A05816F3CD519A9A
        03B7802C2F0BC63D859AC65E9A57E45C
        FF105477FF36382B168996C86336DBCB
        B669E0C61E134C9701365F06B9F90DA3
8308 20 (SHA-x Hash Code, length OK)
        BD7D8EB92C55D4E6D1096DBA34D0E155
        DAE3E8D7EADE275E11366B9DA3830B28
8309 08 C63B1467CCC493FD (Encrypted Session ID, length OK)
```

C.4 About the SDKs and Additional Examples

MagTek provides SDKs and corresponding documentation for many programming languages and operating systems that enable software developers to quickly develop custom host software that communicates with this device, without having to deal with the complexities of platform APIs for direct communication across the various available connection types, connecting using the various available communication protocols, and parsing the various available data formats.

The SDKs and corresponding documentation include:

- Functions for sending the direct commands described in this manual
- Wrappers for commonly used commands and properties that further simplify development
- Detailed compilable examples of processing incoming payment data and using the direct commands and properties described in this manual

To download the SDKs and documentation, search www.magtek.com for “SDK” and select the SDK and documentation for the programming languages and platforms you need, or contact MagTek Support Services for assistance.

Appendix D Keyboard Usage ID Definitions (KB Only)

When the device is in keyboard mode, for each character it needs to send to the host, it looks up the character in an internal lookup table to find the keystroke and key modifier (if any) to send to the host to produce that character. The tables in the following sections show the default content of those internal lookup tables. In cases where the host operating system is set up to interpret incoming keystrokes as characters that are different from these tables, the host software can read the device's tables using **Command 0x03 - Get Keymap Item (KB Only)** and modify them to match the host's expectations using **Command 0x04 - Set Keymap Item (MAC, KB Only)**. For more information about using the device in keyboard mode, see section **2.1.4 How to Use the USB Connection in Keyboard Emulation Mode (KB Only)** or section **2.2.5 How to Use the Bluetooth LE Connection In Keyboard Emulation Mode**.

The information in the following subsections is from ***Section 10, Keyboard/Keypad Page (0x07)*** of ***Universal Serial Bus HID Usage Tables, Version 1.12***, found on *www.usb.org*.

D.1 Keyboard/Keypad Page (0x07) (KB Only)

This section is the Usage Page for key codes to be used in implementing a USB keyboard. A Boot Keyboard (84-, 101- or 104-key) should at a minimum support all associated usage codes indicated in the "Boot" column below.

The usage type of all key codes is Selectors (Sel), except for the modifier keys Keyboard Left Control (0x224) to Keyboard Right GUI (0x231) which are Dynamic Flags (DV).

Note

A general note on Usages and languages: Due to the variation of keyboards from language to language, it is not feasible to specify exact key mappings for every language. Where this list is not specific for a key function in a language, the closest equivalent key position should be used, so that a keyboard may be modified for a different language by simply printing different keycaps. One example is the Y key on a North American keyboard. In Germany this is typically Z. Rather than changing the keyboard firmware to put the Z Usage into that place in the descriptor list, the vendor should use the Y Usage on both the North American and German keyboards. This continues to be the existing practice in the industry, in order to minimize the number of changes to the electronics to accommodate other languages.

Table 9-6 - Keyboard/Keypad

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC-AT	Mac	UNIX	Boot
0	00	Reserved (no event indicated) 9	N/A	√	√	√	4/101/104
1	01	Keyboard ErrorRollOver9	N/A	√	√	√	4/101/104
2	02	Keyboard POSTFail9	N/A	√	√	√	4/101/104
3	03	Keyboard ErrorUndefined9	N/A	√	√	√	4/101/104
4	04	Keyboard a and A4	31	√	√	√	4/101/104
5	05	Keyboard b and B	50	√	√	√	4/101/104
6	06	Keyboard c and C4	48	√	√	√	4/101/104
7	07	Keyboard d and D	33	√	√	√	4/101/104

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC-AT	Mac	UNIX	Boot
8	08	Keyboard e and E	19	√	√	√	4/101/104
9	09	Keyboard f and F	34	√	√	√	4/101/104
10	0A	Keyboard g and G	35	√	√	√	4/101/104
11	0B	Keyboard h and H	36	√	√	√	4/101/104
12	0C	Keyboard i and I	24	√	√	√	4/101/104
13	0D	Keyboard j and J	37	√	√	√	4/101/104
14	0E	Keyboard k and K	38	√	√	√	4/101/104
15	0F	Keyboard l and L	39	√	√	√	4/101/104
16	10	Keyboard m and M	52	√	√	√	4/101/104
17	11	Keyboard n and N	51	√	√	√	4/101/104
18	12	Keyboard o and O4	25	√	√	√	4/101/104
19	13	Keyboard p and P4	26	√	√	√	4/101/104
20	14	Keyboard q and Q4	27	√	√	√	4/101/104
21	15	Keyboard r and R	20	√	√	√	4/101/104
22	16	Keyboard s and S4	32	√	√	√	4/101/104
23	17	Keyboard t and T	21	√	√	√	4/101/104
24	18	Keyboard u and U	23	√	√	√	4/101/104
25	19	Keyboard v and V	49	√	√	√	4/101/104
26	1A	Keyboard w and W4	18	√	√	√	4/101/104
27	1B	Keyboard x and X4	47	√	√	√	4/101/104
28	1C	Keyboard y and Y4	22	√	√	√	4/101/104
29	1D	Keyboard z and Z4	46	√	√	√	4/101/104
30	1E	Keyboard 1 and !4	2	√	√	√	4/101/104
31	1F	Keyboard 2 and !4	3	√	√	√	4/101/104
32	20	Keyboard 3 and #4	4	√	√	√	4/101/104
33	21	Keyboard 4 and $4	5	√	√	√	4/101/104
34	22	Keyboard 5 and %4	6	√	√	√	4/101/104
35	23	Keyboard 6 and ^4	7	√	√	√	4/101/104
36	24	Keyboard 7 and &4	8	√	√	√	4/101/104

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC-AT	Mac	UNIX	Boot
37	25	Keyboard 8 and *4	9	√	√	√	4/101/104
38	26	Keyboard 9 and (4	10	√	√	√	4/101/104
39	27	Keyboard 0 and)4	11	√	√	√	4/101/104
40	28	Keyboard Return (ENTER)5	43	√	√	√	4/101/104
41	29	Keyboard ESCAPE	110	√	√	√	4/101/104
42	2A	Keyboard DELETE (Backspace)	15	√	√	√	4/101/104
43	2B	Keyboard Tab	16	√	√	√	4/101/104
44	2C	Keyboard Spacebar	61	√	√	√	4/101/104
45	2D	Keyboard - and (underscore)4	12	√	√	√	4/101/104
46	2E	Keyboard = and +4	13	√	√	√	4/101/104
47	2F	Keyboard [and {4	27	√	√	√	4/101/104
48	30	Keyboard] and }4	28	√	√	√	4/101/104
49	31	Keyboard \ and |	29	√	√	√	4/101/104
50	32	Keyboard Non-US # and ~2	42	√	√	√	4/101/104
51	33	Keyboard ; and :4	40	√	√	√	4/101/104
52	34	Keyboard ‘ and “4	41	√	√	√	4/101/104
53	35	Keyboard Grave Accent and Tilde4	1	√	√	√	4/101/104
54	36	Keyboard, and <4	53	√	√	√	4/101/104
55	37	Keyboard. and >4	54	√	√	√	4/101/104
56	38	Keyboard / and ?	55	√	√	√	4/101/104
57	39	Keyboard Caps Lock11	30	√	√	√	4/101/104
58	3A	Keyboard F1	112	√	√	√	4/101/104
59	3B	Keyboard F2	113	√	√	√	4/101/104
60	3C	Keyboard F3	114	√	√	√	4/101/104
61	3D	Keyboard F4	115	√	√	√	4/101/104
62	3E	Keyboard F5	116	√	√	√	4/101/104
63	3F	Keyboard F6	117	√	√	√	4/101/104
64	40	Keyboard F7	118	√	√	√	4/101/104
65	41	Keyboard F8	119	√	√	√	4/101/104

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC-AT	Mac	UNIX	Boot
66	42	Keyboard F9	120	√	√	√	4/101/104
67	43	Keyboard F10	121	√	√	√	4/101/104
68	44	Keyboard F11	122	√	√	√	101/104
69	45	Keyboard F12	123	√	√	√	101/104
70	46	Keyboard PrintScreen1	124	√	√	√	101/104
71	47	Keyboard Scroll Lock11	125	√	√	√	4/101/104
72	48	Keyboard Pause1	126	√	√	√	101/104
73	49	Keyboard Insert1	75	√	√	√	101/104
74	4A	Keyboard Home1	80	√	√	√	101/104
75	4B	Keyboard PageUp1	85	√	√	√	101/104
76	4C	Keyboard Delete Forward1;14	76	√	√	√	101/104
77	4D	Keyboard End1	81	√	√	√	101/104
78	4E	Keyboard PageDown1	86	√	√	√	101/104
79	4F	Keyboard RightArrow1	89	√	√	√	101/104
80	50	Keyboard LeftArrow1	79	√	√	√	101/104
81	51	Keyboard DownArrow1	84	√	√	√	101/104
82	52	Keyboard UpArrow1	83	√	√	√	101/104
83	53	Keypad Num Lock and Clear11	90	√	√	√	101/104
84	54	Keypad /1	95	√	√	√	101/104
85	55	Keypad *	100	√	√	√	4/101/104
86	56	Keypad -	105	√	√	√	4/101/104
87	57	Keypad +	106	√	√	√	4/101/104
88	58	Keypad ENTER5	108	√	√	√	101/104
89	59	Keypad 1 and End	93	√	√	√	4/101/104
90	5A	Keypad 2 and Down Arrow	98	√	√	√	4/101/104
91	5B	Keypad 3 and PageDn	103	√	√	√	4/101/104
92	5C	Keypad 4 and Left Arrow	92	√	√	√	4/101/104
93	5D	Keypad 4 and Left Arrow	97	√	√	√	4/101/104
94	5E	Keypad 4 and Left Arrow	102	√	√	√	4/101/104

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC-AT	Mac	UNIX	Boot
95	5F	Keypad 7 and Home	91	√	√	√	4/101/104
96	60	Keypad 8 and Up Arrow	96	√	√	√	4/101/104
97	61	Keypad 9 and PageUp	101	√	√	√	4/101/104
98	62	Keypad 0 and Insert	99	√	√	√	4/101/104
99	63	Keypad . and Delete	104	√	√	√	4/101/104
100	64	Keyboard Non-US \ and |3;6	45	√	√	√	4/101/104
101	65	Keyboard Application10	129	√		√	104
102	66	Keyboard Power9 =			√	√	
103	67	Keypad =			√		
104	68	Keyboard F13	62		√		
105	69	Keyboard F14	63		√		
106	6A	Keyboard F15	64		√		
107	6B	Keyboard F16	65				
107	6C	Keyboard F17					
109	6D	Keyboard F18					
110	6E	Keyboard F19					
111	6F	Keyboard F20					
112	70	Keyboard F21					
113	71	Keyboard F22					
114	72	Keyboard F23					
115	73	Keyboard F24					
116	74	Keyboard Execute				√	
117	75	Keyboard Help				√	
118	76	Keyboard Menu				√	
119	77	Keyboard Select				√	
120	78	Keyboard Stop				√	
121	79	Keyboard Again				√	
122	7A	Keyboard Undo				√	
123	7B	Keyboard Cut				√	

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC-AT	Mac	UNIX	Boot
124	7C	Keyboard Copy				√	
125	7D	Keyboard Paste				√	
126	7E	Keyboard Find				√	
127	7F	Keyboard Mute				√	
128	80	Keyboard Volume Up				√	
129	81	Keyboard Volume Down				√	
130	82	Keyboard Locking Caps Lock12				√	
131	83	Keyboard Locking Num Lock12				√	
132	84	Keyboard Locking Scroll Lock12				√	
133	85	Keypad Comma27	107				
134	86	Keypad Equal Sign29					
135	87	Keyboard International115-28	56				
136	88	Keyboard International216					
137	89	Keyboard International317					
138	8A	Keyboard International418					
139	8B	Keyboard International519					
140	8C	Keyboard International620					
141	8D	Keyboard International721					
142	8E	Keyboard International822					
143	8F	Keyboard International922					
144	90	Keyboard Lang125					
145	91	Keyboard Lang226					
146	92	Keyboard Lang330					
147	93	Keyboard Lang431					
148	94	Keyboard Lang532					
149	95	Keyboard Lang68					
150	96	Keyboard Lang78					
151	97	Keyboard Lang88					
152	98	Keyboard Lang98					

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC-AT	Mac	UNIX	Boot
153	99	Keyboard Alternate Erase7					
154	9A	Keyboard Sys/Req Attention1					
155	9B	Keyboard Cancel					
156	9C	Keyboard Clear					
157	9D	Keyboard Prior					
158	9E	Keyboard Return					
159	9F	Keyboard Separator					
160	A0	Keyboard Out					
161	A1	Keyboard Oper					
162	A2	Keyboard Clear/Again					
163	A3	Keyboard Cr/Sel/Props					
164	A4	Keyboard Ex Sel					
165-175	A5-CF	Reserved					
176	B0	Keypad 00					
177	B1	Keypad 000					
178	B2	Thousands Separator33					
179	B3	Decimal Separator33					
180	B4	Currency Unit34					
181	B5	Currency Sub-unit34					
182	B6	Keypad (					
183	B7	Keypad)					
184	B8	Keypad {					
185	B9	Keypad}					
186	BA	Keypad Tab					
187	BB	Keypad Backspace					
188	BC	Keypad A					
189	BD	Keypad B					
190	BE	Keypad C					
191	BF	Keypad D					

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC-AT	Mac	UNIX	Boot
192	C0	Keypad E					
193	C1	Keypad F					
194	C2	Keypad XOR					
195	C3	Keypad ^					
196	C4	Keypad %					
197	C5	Keypad <					
198	C6	Keypad >					
199	C7	Keypad &					
200	C8	Keypad &&					
201	C9	Keypad |					
202	CA	Keypad ||					
203	CB	Keypad :					
204	CC	Keypad #					
205	CD	Keypad Space					
206	CE	Keypad @					
207	CF	Keypad !					
208	D0	Keypad Memory Store					
209	D1	Keypad Memory Recall					
210	D2	Keypad Memory Clear					
211	D3	Keypad Memory Add					
212	D4	Keypad Memory Subtract					
213	D5	Keypad Memory Multiple					
214	D6	Keypad Memory Divide					
215	D7	Keypad +/-					
216	D8	Keypad Clear					
217	D9	Keypad Clear Entry					
218	DA	Keypad Binary					
219	DB	Keypad Octal					
220	DC	Keypad Decimal					

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC-AT	Mac	UNIX	Boot
221	DD	Keypad Hexadecimal					
222-223	DE-DF	Reserved					
224	E0	Keyboard LeftControl	58	√	√	√	
225	E1	Keyboard LeftShift	44	√	√	√	
226	E2	Keyboard LeftAlt	60	√	√	√	
227	E3	Keyboard Left GUI10;23	127	√	√	√	
228	E4	Keyboard RightControl	64	√	√	√	
229	E5	Keyboard RightShift	57	√	√	√	
230	E6	Keyboard RightAlt	62	√	√	√	
231	E7	Keyboard Right GUI10;24	128	√	√	√	
232..65535	E8-FFFF	Reserved					

Footnotes

1. Usage of keys is not modified by the state of the Control, Alt, Shift or Num Lock keys. That is, a key does not send extra codes to compensate for the state of any Control, Alt, Shift or Num Lock keys.
2. Typical language mappings: US: \| Belg: µ`£ FrCa: <}> Dan:'* Dutch: <> Fren:*µ Ger: #' Ital: ù§ LatAm: }`] Nor:,* Span: }Ç Swed: ,* Swiss: $£ UK: #~.
3. Typical language mappings: Belg:<\> FrCa:«°» Dan:<\> Dutch:]||[Fren:<> Ger:<|> Ital:<> LatAm:<> Nor:<> Span:<> Swed:<|> Swiss:<\> UK:\| Brazil: \|.
4. Typically remapped for other languages in the host system.
5. Keyboard Enter and Keypad Enter generate different Usage codes.
6. Typically near the Left-Shift key in AT-102 implementations.
7. Example, Erase-Eaze™ key.
8. Reserved for language-specific functions, such as Front End Processors and Input Method Editors.
9. Reserved for typical keyboard status or keyboard errors. Sent as a member of the keyboard array. Not a physical key.
10. Windows key for Windows 95, and "Compose."
11. Implemented as a non-locking key; sent as member of an array.
12. Implemented as a locking key; sent as a toggle button. Available for legacy support; however, most systems should use the non-locking version of this key.
13. Backs up the cursor one position, deleting a character as it goes.
14. Deletes one character without changing position.

15-20. See additional foot notes in Universal Serial Bus HID Usage Tables, Copyright © 1996-2005, USB Implementers Forum.

21. Toggle Double-Byte/Single-Byte mode.
22. Undefined, available for other Front End Language Processors.
23. Windowing environment key, examples are Microsoft Left Win key, Mac Left Apple key, Sun Left Meta key
24. Windowing environment key, examples are Microsoft® RIGHT WIN key, Macintosh® RIGHT APPLE key, Sun® RIGHT META key.

Usage ID (Dec)	Usage ID (Hex)	Usage Name	Ref: Typical AT-101 Position	PC-AT	Mac	UNIX	Boot

25. Hangul/English toggle key. This usage is used as an input method editor control key on a Korean language keyboard.
26. Hanja conversion key. This usage is used as an input method editor control key on a Korean language keyboard.
27. Keypad Comma is the appropriate usage for the Brazilian keypad period (.) key. This represents the closest possible match, and system software should do the correct mapping based on the current locale setting.
28. Keyboard International1 should be identified via footnote as the appropriate usage for the Brazilian forward-slash (/) and question-mark (?) key. This usage should also be renamed to either "Keyboard Non-US / and ?" or to "Keyboard International1" now that it's become clear that it does not only apply to Kanji keyboards anymore.
29. Used on AS/400 keyboards.
30. Defines the Katakana key for Japanese USB word-processing keyboards.
31. Defines the Hiragana key for Japanese USB word-processing keyboards.
32. Usage 0x94 (Keyboard LANG5) "Defines the Zenkaku/Hankaku key for Japanese USB word-processing keyboards.
33. The symbol displayed depends on the current locale settings of the operating system. For example, the US thousands separator would be a comma, and the decimal separator would be a period.
34. The symbol displayed depends on the current locale settings of the operating system. For example the US currency unit would be $ and the sub-unit would be ¢.

D.2 Modifier Byte Definitions (KB Only)

This appendix is from ***Section 8.3 Report Format for Array Items*** of ***Device Class Definition for Human Interface Devices (HID) Version 1.11***, found on www.usb.org. The modifier byte is defined in **Table 9-7**.

Table 9-7 - Modifier Byte

Bit	Key
0	LEFT CTRL
1	LEFT SHIFT
2	LEFT ALT
3	LEFT GUI
4	RIGHT CTRL
5	RIGHT SHIFT
6	RIGHT ALT
7	RIGHT GUI

Appendix E Identifying ISO/ABA and AAMVA Cards For Masking (MSR Only)

E.1 ISO/ABA Financial Card

The device uses the rules below to determine if a card is an ISO/ABA card (per ***ISO 7811-2,2001***), which affects incoming **Card Encode Type (HID Only | TLV Only | GATT Only | SLIP Only)** or choice of sentinels in **Magnetic Stripe Card Data In Streaming Format (Swipe Only | Keypad Entry Only)**, as well as the masking used for **Masked Track Data**. ISO defines a particular and different bit-level character encoding format of the data on each of the three tracks of the card. The format of the card depends on decisions made by the entity that issued the card. For example, some organizations may choose to use the ISO Track 1 encoding format for Track 2 data, or other permutations that do not conform to the standard. The device only considers ISO Financial masking for cards it classifies as ISO, which it determines according to the following rules:

1) If the low level decoding algorithm determines the bit level character encoding for every track conforms to the ISO format defined for that track, the card is classified as ISO. Otherwise the device attempts to classify the card as an **AAMVA Driver's License**, and if the card fails that test, the device classifies the card as **Other**. A properly encoded ISO card has the following properties:
    a) At least one track must be decodable.
    b) Track 1 must be 7 bits per character.
    c) If Track 2 or Track 3 exist, they must be 5 bits per character.
2) If the device determines the card is ISO encoded, it then determines the masking behavior for each track independently. One track may qualify for masking and another may not, according to the following rules.
3) For Track 1, the device's intent is to send the card's Format Code unmasked, the PAN partially masked, the Name and Expiration Date unmasked, and the rest of the track masked, with exceptions:
    a) The Service Code is always unmasked on newer devices (Never Mask Service Code Only). On legacy devices, the Service Code is always masked.
    b) If the card's Format Code, PAN, Name, or Expiration Date are not correctly structured, the device transmits the rest of the track unmasked starting with the point of discrepancy. The device defines "correct structure" for Track 1 as follows:
        i) The card's Format Code, PAN, Name, or Expiration Date do not contain the '?' character (End Sentinel).
        ii) The Format Code is the first character on the track and is the character 'B'.
        iii) The PAN has a maximum of 19 digits and ends with character '^' (Field Separator).
        iv) The Cardholder Name has a maximum of 26 characters and is ended by the character '^' (Field Separator).
        v) The Expiration Date has 4 characters.
        vi) The Service Code has 3 characters.
4) For Track 2, the device's intent is to send the PAN partially masked, the Expiration Date unmasked, and most of the rest of the track masked, with exceptions:
    a) The Service Code is always unmasked on newer devices (Never Mask Service Code Only). On legacy devices, the Service Code is always masked.
    b) If the PAN or Expiration Date are not correctly structured, the device sends the rest of the track unmasked starting at the point of discrepancy. The device defines "correct structure" as follows:
        i) The PAN or Expiration Date does not contain the '?' character (End Sentinel).
        ii) The PAN has a maximum of 19 digits and ends with the character '=' (Field Separator).

    iii) The Expiration Date has 4 characters.
    iv) The Service Code has 3 characters.

5) For Track 3, the device's intent is to send the PAN partially masked and the rest of the track masked, with exceptions:
    a) If the PAN is not correctly structured, the device sends the rest of the track unmasked, starting at the point of discrepancy. The device defines "correct structure" as follows:
        i) The PAN does not contain the '?' character (End Sentinel).
        ii) The PAN has a maximum of 19 digits and ends with character '=' (Field Separator).

E.2 AAMVA Driver's License

The device uses the following rules to determine if a card is an AAMVA card:

1) If the device reads three tracks of data and Track 1 is formatted per ISO Track 1 rules, Track 2 is formatted per ISO Track 2 rules, and Track 3 is formatted per ***ISO Track 1*** [sic.] rules, the card is considered to be an AAMVA card. Some MagTek devices do not support reading of Track 3, so this rule does not apply on such devices.
2) If a low level decoding algorithm finds data for the available tracks to be in the ISO format particular to each track, and Track 2 contains a correctly structured PAN field whose first 6 digits are "604425" or contain values in the range "636000" to "636062" inclusive, the card is considered to be an AAMVA card.

AAMVA card masking, when enabled, works as follows:

1) The device sends track 1 and track 3 entirely masked; all character positions are filled with zeroes.
2) Track 2 is treated as follows:
    a) The device's intent is to send the Driver License ID (DLID) partially masked, the Expiration Date unmasked, the Birth Date unmasked, and the rest of the track masked.
    b) If the DLID, Expiration Date, or Birth Date are not correctly structured, the rest of the track, starting at the point of discrepancy, is sent unmasked. The device defines "correctly structured" as follows:
        i) If the DLID, Expiration Date, or Birth Date contain the '?' character (End Sentinel), the field is not correctly structured.
        ii) A correctly structured DLID has a maximum of 19 digits and is terminated by the character '=' (Field Separator).
        iii) A correctly structured Expiration Date has 4 characters.
        iv) A correctly structured Birth Date has 8 characters.

Appendix F TLV Tag Allocation Dictionary (TLV Only)

This section provides the complete dictionary of tags that could be used with TLV format in MagneSafe V5 devices (see section **3.5 How to Use Tag-Length-Value (TLV) Format**).

Not all tags are used by, or available from, all devices and all device configurations.

F.1 Tag Allocation Scheme (TLV Only)

Tags are chosen according to the following criteria:

- Tags starting with ‘C1’ (e.g. ‘C101’) are the highest level containers, they are used to define the overall message type.
- Tags starting with ‘C2’ (e.g. ‘C201’) specify sub-containers, although some sub-containers may have other tag names.
- Tags starting with ‘C3’ (e.g. ‘C306’) specify sub-containers.
- Tags starting with ‘80’ (e.g. ‘8001’) specify operational information.
- Tags starting with ‘81’ (e.g. ‘8101’) specify device specific information.
- Tags starting with ‘82’ (e.g. ‘8201’) specify card specific information.
- Tags starting with ‘83’ (e.g. ‘8301’) specify security elements.
- Tags starting with ‘84’ (e.g. ‘8401’) specify system level command / response elements.
- Tags starting with ‘85’ (e.g. ‘8501’) specify generic capabilities (older capabilities objects may have ‘81xx’ Tags).
- Tags starting with ‘86’ (e.g. ‘8601’) specify generic configuration (older configuration objects may have ‘81xx’ Tags).
- Tags starting with ‘87’ (e.g. ‘8701’) specify device specific capabilities (older capabilities objects may have ‘81xx’ Tags).
- Tags starting with ‘88’ (e.g. ‘88xx’) specify device specific configuration (older configuration objects may have ‘81xx’ Tags).

F.2 Structure of Tag Field (TLV Only)

The Tag field indicates the type of message. The Tag length (the number of octets in the Tag field) may be 1 or more octets. Each individual Tag octet may take on a value 0x00 - 0x7F.

The most significant bit (MSB) shall be either:

- 0 to indicate the current octet is the last octet of the Tag field, or
- 1 to indicate another Tag octet follows the current one

The next most significant bit (bit 6), shall be either:

- 1 to indicate the current object is Constructed: That is, it contains further TLV fields), or
- 0 to indicate the current object is Primitive: That is, it contains non-TLV data.

Information about Tags used as containers for other Tags (Constructed Tags) is given after the main table

MagTek TLV Tag Allocation Dictionary					
Tag	Name	Format	Length	Source	Description
High Level Containers					
C101	MSR Swipe Data	Constructed	Variable	Device	Used by Device to send data generated from a swipe of a magnetic stripe card. This Tag may be generated by asynchronously by some devices, and it may be requested in a DOL in some devices (some devices may support both modes).
C102	Host to Device Request	Constructed	Variable	Host	Used to contain messages from the Host to our Device.
~~C103~~	~~Device to Host Response~~ DEPRECATED	~~Constructed~~	~~Variable~~	~~Device~~	~~Used to contain responses to Host to Device Request tags.~~ This Tag was Deprecated in the transition from TLV Version 1 to 2 because the length calculation associated with became ambiguous. With this deprecated Tag, the length field is always one byte long with values from 0 - 255 possible. The new Tag is C104.
C104	Device to Host Response	Constructed	Variable	Device	Used to contain responses to Host to Device Request tags.
C105	Device to Host Event	Constructed	Variable	Device	Used to contain event notification tags from a Device to a Host.

MagTek TLV Tag Allocation Dictionary

C106	MagneSafe V5 Card Swipe Data	Constructed	Variable	Device	Used by Device to send data generated from a swipe of a magnetic stripe card in a format compatible with MagneSafe V5 products. This Tag may be generated by asynchronously by some devices, and it may be requested in a DOL in some devices (some devices may support both modes).
Sub-Containers					
C201	Swipe Status	Constructed	Variable	Device	Used to give information about the status of a swipe. May contain Operation Status, Card Status, and Track Status tags, and possibly others.
C202	Magnetic Stripe Local Merchant Data	Constructed	Variable	Device	Used to give information about a card for use by a merchant in local processing. May contain Cardholder Name, IIN, Last 4 Digits of PAN, Expiration Date, Service Code, PAN Length, and other tags.
C203	Magnetic Stripe Secure Data	Constructed	Variable	Device	Used to give secure information (usually forwarded for processing by a secure service). May contain Device SN, Key Identifier, Key Type, Key Variant, Encrypted Track Data, MagnePrint Status, Encrypted MagnePrint Data,
~~C204~~	~~Unrecognized~~ DEPRECATED	~~Simple - binary~~	~~0~~	~~Device~~	~~Used by Device to inform a Host that the Tag included in the Host to Device Request message is Unrecognized (and therefore not processed)~~
C205	Device Standard commands	Constructed	Variable	Host	Used by the Host to send Device Standard commands to the Device
~~C206~~	~~Device Standard responses~~ DEPRECATED	~~Constructed~~	~~Variable~~	~~Device~~	~~Used by Device to respond to a Device Standard command.~~ This Tag was Deprecated in the transition from TLV Version 1 to 2 because the length calculation associated with became ambiguous. With this deprecated Tag, the length field is always one byte long with values from 0 - 255 possible. The new Tag is C20B.

MagTek TLV Tag Allocation Dictionary

~~C207~~	~~MagneSafe V5 commands~~ DEPRECATED	~~Simple - binary~~	~~Variable~~	~~Host~~	~~Used to send a MagneSafe V5 command from a host to a device~~
~~C208~~	~~MagneSafe V5 response~~ DEPRECATED	~~Simple - binary~~	~~Variable~~	~~Device~~	~~Used by Device to respond to a MagneSafe V5 command tag~~
C20A	Data Object List response Data	Constructed	Variable	Device	Used by Device to respond to a Data Object List request tag. This tag contains all of the requested tags that are available.
C20B	Device Standard responses	Constructed	Variable	Device	Used by Device to respond to a Device Standard command.
Common Sub-Containers					
~~C301~~	~~Discovery~~ DEPRECATED	~~Constructed~~	~~Variable~~	~~Device~~	~~Container for discovery objects, including Device Information, Device Status, Device Configuration, Device Security~~ This Tag was Deprecated in the transition from TLV Version 1 to 2 because the length calculation associated with became ambiguous. With this deprecated Tag, the length field is always one byte long with values from 0 - 255 possible. The new Tag is C306, later C30B.
C302	Supplemental Information	Constructed	Variable	Device	Container for Supplemental Information objects, including TLV Version, Device Main Firmware Part #, Device Model Name, Battery Level, and Card Swipe Count
C303	Device Status	Constructed	Variable	Device	Container for Device Status objects, for now including only the Battery Level object.
C304	Device Configuration	Constructed	Variable	Device	Container for Device Configuration objects, including Send Card Name, Send Card IIN, Send Card Last 4 Digits of PAN, Send Card Expiration Data, Send Card Service Code, Send Card PAN Length, Send TLV Version on Power UP, and Send Discovery on Power Up
C305	Device Security	Constructed	Variable	Device	Container for Signed Security objects, contents TBD.

MagTek TLV Tag Allocation Dictionary

~~C306~~	~~Discovery~~ DEPRECATED by request, see C30B to replace it.	~~Constructed~~	~~Variable~~	~~Device~~	~~Container for discovery objects, including Device Information, Device Status, Device Configuration, Device Security~~
~~C306~~	~~Device Capabilities~~ DEPRECATED	~~Constructed~~	~~Variable~~	~~Device~~	~~Contains objects that identify the basic capabilities of the device. Some of the capabilities may not be configured to active.~~ Deprecated in the transition from TLV Level 2 to 3, actually was a duplicate use of the Tag (Discovery).
C307	Device Information	Constructed	Variable	Device	Container for Device Information objects, including Device SN - Internal, Device SN - MagTek, Device Part Number, Device Main Firmware Part, TLV Version, Device Model Name, Capabilities
C308	Device Capabilities	Constructed	Variable	Device	Contains objects that identify the basic capabilities of the device. Some of the capabilities may not be configured to active.
C309	MagneSafe V5 Specific Capabilities	Constructed	Variable	Device	Contains objects that identify MagneSafe V5 specific capabilities supported by the device.
C30A	MagneSafe V5 Specific Configuration	Constructed	Variable	Device	Contains objects that identify MagneSafe V5 specific configuration information for the device.
C30B	Discovery	Constructed	Variable	Device	Container for discovery objects, including Device Information, Device Status, Device Configuration, Device Security
Operational Information Tags					
8001	Operation Status	Simple - binary	2	Device	**Device Encryption Status**
8002	Timeout	Simple - binary	2	Device	Two bytes, allows to pass a reason for the timeout (usage is optional, length is zero if no further information). This Tag is sent (by itself) whenever a timeout event needs to be reported. Initially this means MagneSafe V5 Authentication Timeout.

MagTek TLV Tag Allocation Dictionary					
8003	No MSR Transactions Remaining	Simple - binary	0	Device	No content required, simply documents that a swipe occurred, but there are no MSR transactions remaining. See **Command 0x1C - Get Remaining MSR Transactions Counter (MSR Only)**. This is a MagneSafe V5 particular Tag
8004	TDES DUKPT Key Counter Exhausted	Simple - binary	0	Device	No content required, simply documents that a swipe occurred, but the TDES DUKPT Key Counter is exhausted. This Tag is sent (by itself) whenever a card swipe occurs and the device has used all 1,048,000 keys available.
Device Information Tags					
8101	Device SN - Internal	Simple - binary	8	Device	The Device's Serial Number as created by the chip manufacturer. This never changes over the life of the device.
8102	Device SN - MagTek	Simple - binary	TBD	Device	The Device's Serial Number as created by MagTek. This never changes over the life of the device.
8103	Device Main Firmware Part Number and Revision	Simple - ASCII	11	Device	The firmware identifier of the main firmware in the device.
8104	Device Model Name	Simple - ASCII	Variable	Device	Identifies the Model Name of the device. A single model may have several firmware part numbers and revisions associated with it.
~~8105~~	~~Device Capabilities~~ DEPRECATED	~~Simple - ASCII~~	~~Variable~~	~~Device~~	~~Identifies the basic capabilities of the device. Some of the capabilities may not be configured to active.~~
8106	Device Status	Simple - binary	Variable	Device	The use of this object may change depending on the type of device, you need to see the device's specific documentation for details.
~~8107~~	~~TLV Version~~ DEPRECATED starting at TLV Level 2	~~Simple - binary~~	~~2~~	~~Device~~	~~Gives a version for the TLV scheme the device uses~~

MagTek TLV Tag Allocation Dictionary					
8108	Device Part Number	Simple - ASCII	11	Device	Identifies the part number for this device.
8109	TLV Version	Simple - binary	2	Device	Gives a version for the TLV scheme the device uses, lsb first.
810A	Remaining MSR Transactions	Simple - binary	4	Device	Count of encryption cycles remaining
810C	Unencrypted Session ID	Simple - binary	8	Device	Current Session ID, unencrypted.
8120	Capability - MSR	Simple - binary	1	Device	0 = No MSR, 1 = MSR
8121	Capability - Tracks	Simple - binary	1	Device	Bit 0 = 1 / Track 1 supported, Bit 1 = 1 / Track 2 supported, Bit 2 = 1 / Track 3 supported, all other bits 0.
8122	Capability - Magnetic Stripe Encryption	Simple - binary	2	Device	0 = No Encryption, 1 = PIN Encryption Variant, other values TBD
8140	Battery Level	Simple - binary	1	Device	Gives the useful percentage of battery available. This one byte unsigned integer gives a number between 0 and 100 indicating the percentage of battery available.
8141	Card Swipe Count	Simple - binary	4	Device	Gives count of card swipes sent to host over the lifetime of the device. Most significant byte is first, least significant byte is last.
8160	Send TLV Version on Power Up	Simple - binary	1	Device	0 = No TLV Version on Power Up, 1 = Send TLV Version on Power Up (TLV Version is Tag 8107)
8161	Send Discovery on Power Up	Simple - binary	1	Device	0 = No Discovery Data on Power Up, 1 = Send Discovery Data on Power Up (Discovery is Tag C301)
8162	Send Card Name	Simple - binary	1	Device	0 = Don’t send Card Name for each card swipe, 1 = Send Card Name for each swipe (Card Name is Tag 8241
8163	Send Card IIN	Simple - binary	1	Device	0 = Don’t send Card IIN for each card swipe, 1 = Send Card IIN for each swipe (Card IIN is Tag 8242)

MagTek TLV Tag Allocation Dictionary					
8164	Send Card Last 4 Digits of PAN	Simple - binary	1	Device	0 = Don't send Card Last 4 Digits of PAN for each card swipe, 1 = send for each swipe (Last 4 Digits of PAN is Tag 8243)
8165	Send Card Expiration Date	Simple - binary	1	Device	0 = Don't send Card Expiration Date for each card swipe, 1 = send for each swipe (Card Expiration Date is Tag 8244)
8166	Send Card Service Code	Simple - binary	1	Device	0 = Don't send Card Service Code for each card swipe, 1 = send for each swipe (Card Service Code is Tag 8245)
8167	Send Card PAN Length	Simple - binary	1	Device	0 = Don't send Card PAN Length for each card swipe, 1 = send for each swipe (Card PAN Length is Tag 8246)
Card Specific Information Tags					
8201	Standard Track 1 Data	Simple - ASCII	Variable	Card	The entire Track 1 data as read from the card.
8202	Standard Track 2 Data	Simple - ASCII	Variable	Card	The entire Track 2 data as read from the card.
8203	Standard Track 3 Data	Simple - ASCII	Variable	Card	The entire Track 3 data as read from the card.
8204	MagnePrint Data (Short)	Simple - binary	54	Device, calculated from Card	The MagnePrint Data (Short) as received from the ASIC that read the card.
8205	MagnePrint Data (Long)	Simple - binary		Device, calculated from Card	The MagnePrint Data (Long) as received from the ASIC that read the card.
8211	Encrypted Track 1 Data	Simple - binary	Variable	Card, calculated by Device	A container holding the encrypted version of Standard Track 1 Data, encrypted under the key specified by Key Identifier.
8212	Encrypted Track 2 Data	Simple - binary	Variable	Card, calculated by Device	A container holding the encrypted version of Standard Track 2 Data, encrypted under the key specified by Key Identifier.
8213	Encrypted Track 3 Data	Simple - binary	Variable	Card, calculated by Device	A container holding the encrypted version of Standard Track 3 Data, encrypted under the key specified by Key Identifier.

MagTek TLV Tag Allocation Dictionary					
8214	Encrypted MagnePrint Data	Simple - binary	Variable (depends on whether contained MP is Short or Long)	Device	A container holding the encrypted version of the MagnePrint Data (Short or Long), encrypted under the key specified by the Key Identifier - MagnePrint Data, or if the Key Identifier - MagnePrint Data tag is not included in the returned data, the Key Identifier - Magnetic Stripe Data.
8221	Track 1 Data, Masked	Simple - ASCII	Variable	Card, masked by Device	Track 1 Data, as read from the card, and as masked by the rules of the Device.
8222	Track 2 Data, Masked	Simple - ASCII	Variable	Card, masked by Device	Track 2 Data, as read from the card, and as masked by the rules of the Device.
8223	Track 3 Data, Masked	Simple - ASCII	Variable	Card, masked by Device	Track 3 Data, as read from the card, and as masked by the rules of the Device.
8241	Cardholder Name	Simple - ASCII	Variable	Card	Cardholder name as read from Track 1 of the card.
8242	Card IIN	Simple - ASCII	Variable	Card	The IIN as read from the card.
8243	Last 4 Digits of PAN	Simple - ASCII	4	Card	The last 4 digits of the PAN as read from the card.
8244	Expiration Data	Simple - ASCII	4	Card	The Expiration Data as read from the card.
8245	Service Code	Simple - ASCII	3	Card	The Service Code as read from the card
8246	PAN Length	Simple - binary	1	Card, calculated by Device	The Length of the PAN field read from the card.
8261	Card Status	Simple - binary	1	Device	**Card Status (HID Only | GATT Only | SLIP Only)**
8262	Track Status	Simple - binary	3	Device	**Track 1 Decode Status (HID Only | TLV Only | GATT Only | SLIP Only)** **Track 2 Decode Status (HID Only | TLV Only | GATT Only | SLIP Only)** **Track 3 Decode Status (HID Only | TLV Only | GATT Only | SLIP Only, 3-Track Only)**
Security Information Tags					

MagTek TLV Tag Allocation Dictionary					
8301	Key Identifier - Magnetic Stripe Data	Simple - binary	10	Device	The Key Identifier of the Key used to encrypt the value field of each of the Secure Data Tags which contains encrypted Magnetic Stripe data.
8302	Key Type, Magnetic Stripe Data	Simple - binary	1	Device	The type of the key specified by Key Identifier - Magnetic Stripe Data. 0 = TDES DUKPT Key Other values TBD. If this object does not appear, assume value of 0.
8303	Key Variant - Magnetic Stripe Data	Simple - binary	1	Device	The variant of the key specified by Key Identifier - Magnetic Stripe Data (usually used for TDES DUKPT keys). 0 = PIN Encryption variant 1 = Data Encryption, request or both ways variant If this object does not appear, assume value of 0.
8304	CBC MAC	Simple - binary	TBD	Device	Provides a Message Authentication Code (MAC) computed using the CBC method
8305	Key Identifier - MagnePrint Data	Simple - binary	10	Device	The Key Identifier of the Key used to encrypt the value field of each of the Secure Data Tags which contains MagnePrint encrypted data. Only sent when the key used to encrypt the MagnePrint encrypted data is different from the key identified in Tag 8301 (Key Identifier - Magnetic Stripe Data)
8306	Key Type, MagnePrint Data	Simple - binary	1	Device	The type of the key specified by Key Identifier - MagnePrint Data.
8307	Key Variant - MagnePrint Data	Simple - binary	1	Device	The variant of the key specified by Key Identifier - MagnePrint Data (usually used for TDES DUKPT keys).

MagTek TLV Tag Allocation Dictionary					
8308	SHA-x Hash Code	Simple - binary	32	Card, calculated by Device	A Hash Code of: An optional secret Salt field, concatenated with, data as read from Track 2. Configuration dictates: Whether the Salt field is used Whether the data hashed is the PAN only, or all of the Track 2 data. Whether SHA-1 or SHA-256 is used to perform the hash.
8309	Encrypted Session ID	Simple - binary	8	Device	The current 8 byte Session ID is encrypted using the specified variant of the current TDES DUKPT Key.
830A	Encrypted Track 1 Data, raw	Simple - binary	Variable	Card, calculated by Device	A simple object holding the encrypted version of Track 1 Data, encrypted under the key specified by Key Identifier (decrypted data does not contain TLV data object)
830B	Encrypted Track 2 Data, raw	Simple - binary	Variable	Card, calculated by Device	A simple object holding the encrypted version of Track 2 Data, encrypted under the key specified by Key Identifier (decrypted data does not contain TLV data object)
830C	Encrypted Track 3 Data, raw	Simple - binary	Variable	Card, calculated by Device	A simple object holding the encrypted version of Track 3 Data, encrypted under the key specified by Key Identifier (decrypted data does not contain TLV data object)
830D	Encrypted MagnePrint Data, raw	Simple - binary	Variable (depends on whether contained MP is Short or Long)	Device	A simple object holding the encrypted version of the MagnePrint Data (Short or Long), encrypted under the key specified by the Key Identifier - MagnePrint Data, or if the Key Identifier - MagnePrint Data tag is not included in the returned data, the Key Identifier - Magnetic Stripe Data (decrypted data does not contain TLV data object)
830E	MagnePrint Status	Simple - binary	4	Device, calculated from Card	The MagnePrint Status as received from the ASIC that read the card.

MagTek TLV Tag Allocation Dictionary					
System Level Commands / Responses					
8401	Unrecognized	Simple - binary	0	Device	Used by Device to inform a Host that the Tag included in the Host to Device Request message is Unrecognized (and therefore not processed)
8402	MagneSafe V5 commands	Simple - binary	Variable	Host	Used to send a MagneSafe V5 command from a host to a device
8403	MagneSafe V5 response	Simple - binary	Variable	Device	Used by Device to respond to a MagneSafe V5 command tag
8404	Data Object List (DOL) request	Simple - binary	Variable	Host	Used to send a Data Object List from a host to a device requesting the specified Data Objects be returned
8405	Retrieve Configuration Information DOL Request	Simple - binary	0	Host	Returns a Data Object List of Configuration Information Tags available for this device. The DOL may be used to choose items to retrieve using the DOL Request.
8406	Retrieve Configuration Information DOL Response	Simple - binary	Variable	Device	Used by Device to respond to a Retrieve Configuration Information DOL Request. This Tag contains a list of the available Configuration Information Tags available from the Device.
8407	Retrieve Security Information DOL Request	Simple - binary	0	Host	Returns a Data Object List of Security Information Tags available for this device. The DOL may be used to choose items to retrieve using the DOL Request.
8408	Retrieve Security Information DOL Response	Simple - binary	Variable	Device	Used by Device to respond to a Retrieve Security Information DOL Request. This Tag contains a list of the available Security Information Tags available from the Device.
8409	Retrieve Discovery Information	Simple - binary	0	Host	Used to retrieve the C30B Discovery Tag contents from the device.

MagTek TLV Tag Allocation Dictionary					
840A	Protocol Pass-through, commands action only.	Simple - binary	Variable	Host	Used to send a command requiring no data from a host to a device. The data field contains information designating what command should be executed.
840B	Protocol Pass-through, Moves data from Host to Device only	Simple - binary	Variable	Host	Used to send a command and accompanying data from a host to a device. The data field contains information designating the command and information to support its execution.
840C	Protocol Pass-through, Solicits Device to Host data only	Simple - binary	Variable	Host	Used to send a command from a host to a device and solicit a response which returns more than just a status of the operation.
840D	Protocol Pass-through, Moves data from Host to Device and solicits Device to Host data in response.	Simple - binary	Variable	Host	Used to send a command and accompanying data from a host to a device, and to solicit a response which returns more than just a status of the operation. The data field contains information designating the command and information to support its execution.
840E	Protocol Pass-through Response	Simple - binary	Variable	Device	Used by Device to respond to a command received with Tags 840A, 840B, 840C, or 840D.
840F	Protocol Pass-through Event	Simple - binary	Variable	Device	Used by Device to send a Protocol Pass-through Event Notification to a host.
'85xx' Generic Capabilities					
8500	Battery Powered	Simple - binary	1	Device	1 = Yes, 0 = No (usually not sent if 0)
8501	Supports Firmware Upgrade	Simple - binary	1	Device	1 = Yes, 0 = No (only sent if 1)
8502	Track Decode	Simple - binary	1	Device	Bit 0 = 1 / ISO, Bit 0 = 0 / No ISO Bit 1 = 1 / AAMVA, Bit 1 = 0 / No AAMVA Bit 2 = 1 / JIS Type 2, Bit 2 = 0 / No JIS Type 2

MagTek TLV Tag Allocation Dictionary					
8503	Supported TDES DUKPT Key Variants	Simple - binary	1	Device	Bit 0, PIN Encryption variant: 1 = Supported 0 = Not Supported Bit 1, Data Encryption, request or both ways variant: 1 = Supported 0 = Not Supported. Bit 2, Data Encryption, response variant: 1 = Supported 0 = Not Supported.
8504	Supports Hash Code	Simple - binary	1	Device	1 = Yes, 0 = No (only sent if 1)
8505	Capability - MagnePrint	Simple - binary	1	Device	0 = No MagnePrint, 1 = Short MagnePrint, 2 = Long MagnePrint
8506	Capability - MagnePrint Encryption	Simple - binary	2	Device	0 = No Encryption, 1 = Same as Magnetic Stripe (8122), other values TBD. If absent, default value is 1.
8507	Capability - MagneSafe 2.0 Encryption	Simple - binary	1	Device	0 = Not supported, 1 = supported.
8508	Supports Card Swipe Counter	Simple - binary	1	Device	1 = Yes, 0 = No (only sent if 1)
8509	Supports Session ID	Simple - binary	1	Device	1 = Yes, 0 = No (only sent if 1)
'86xx' Generic Configuration					
8600	Firmware Upgrade Enabled	Simple - binary	1	Device	1 = Yes, 0 = No (usually not sent if 0)
8601	MSR Enabled	Simple - binary	1	Device	1 = MSR Enabled, 0 = MSR Disabled
8602	Tracks Enabled	Simple - binary	1	Device	Bit 0 = 1 / Track 1 Enabled, 0 / Track 1 Disabled Bit 0 = 1 / Track 2 Enabled, 0 / Track 2 Disabled Bit 0 = 1 / Track 3 Enabled, 0 / Track 3 Disabled
8603	Magnetic Stripe Encryption Enabled	Simple - binary	1	Device	1 = Magnetic Stripe Encryption Enabled, 0 = Magnetic Stripe Encryption Disabled

MagTek TLV Tag Allocation Dictionary					
8604	MagnePrint Enabled	Simple - binary	1	Device	1 = MagnePrint Enabled, 0 = MagnePrint Disabled
8605	MagnePrint Encryption Enabled	Simple - binary	1	Device	1 = MagnePrint Encryption Enabled, 0 = MagnePrint Encryption Disabled
8606	MagneSafe 2.0 Encryption Enabled	Simple - binary	1	Device	1 = Yes, 0 = No (usually not sent if 0)
8607	MSR Data Encryption Variant	Simple - binary	1	Device	0 = PIN Variant used for encryption of Magnetic Stripe data 1 = Data Encryption, both ways 2 = Data Encryption, response (not currently supported)
8608	MagnePrint Data Encryption Variant	Simple - binary	1	Device	0 = PIN Variant used for encryption of Magnetic Stripe data 1 = Data Encryption, both ways 2 = Data Encryption, response (not currently supported)
8609	Hash Code Enabled	Simple - binary	1	Device	0x00 = Device emits a SHA-1 Hash code of all Track 2 data 0x01 = Device emits a SHA-1 Hash code of the Track 2 PAN 0x02 = Device emits a Salted SHA-1 Hash code of all Track 2 data 0x03 = Device emits a Salted SHA-1 Hash code of the Track 2 PAN 0x04 = Device emits a SHA-256 Hash code of all Track 2 data 0x05 = Device emits a SHA-256 Hash code of the Track 2 PAN 0x06 = Device emits a Salted SHA-256 Hash code of all Track 2 data 0x07 = Device emits a Salted SHA-256 Hash code of the Track 2 PAN 0xFF = Device does not emit any Hash code
860B	Send Encrypted Session ID with Card Swipe	Simple - binary	1	Device	1 = Yes, 0 = No (usually not sent if 0)

MagTek TLV Tag Allocation Dictionary

860C	MSR Decode Options	Simple - binary	1	Device	Bit 0, ISO/ABA, 1 = Enabled, 0 = Disabled Bit 1, AAMVA, 1 = Enabled, 0 = Disabled. Bit 2, JIS-2, 1 = Enabled, 0 = Disabled.
860D	TDES DUKPT key used for MagnePrint Encryption	Simple - binary	1	Device	0 = Primary TDES DUKPT Key 1 = Secondary TDES DUKPT Key
'87xx' Device Specific Capabilities (Tag usage may change depending on Tag 810B Device Type)					
8700	MagneSafe V5 Masked Track Support	Simple - binary	1	Device	1 = Yes, 0 = No (usually not sent if 0)
'88xx' Device Specific Configuration (Tag usage may change depending on Tag 810B Device Type)					
8800	MagneSafe V5 ISO Mask Property	Simple - ASCII	6	Device	Reports the MagneSafe V5 ISO Track Mask property
8801	MagneSafe V5 AAMVA Mask Property	Simple - ASCII	6	Device	Reports the MagneSafe V5 AAMVA Track Mask property
8802	MagneSafe V5 Mask Other Cards Property	Simple - binary	1	Device	Reports the MagneSafe V5 Mask Other Cards property
8803	MagneSafe V5 Send Clear AAMVA Card Data property	Simple - binary	1	Device	Reports the MagneSafe V5 Send Clear AAMVA Card Data property

F.3 Containers Used (TLV Only)

This section serves as a guideline as to which containers are used and what they may contain. Other containers may be added in the future and may contain tags that overlap with tags contained in already defined containers.

Protocol Level Containers

Tag	Name	Description	Items contained	
C101	MSR Swipe Data	All data generated from a card swipe	Includes the following tags	
			C302	Supplemental Information
			C201	Swipe Status
			C202	Magnetic Stripe Local Merchant Data
			C203	Magnetic Stripe Secure Data

Protocol Level Containers

C102	Host to Device Request	Used to contain messages from the Host to our Device.	Could include one of the following	
The list of possible objects is not all inclusive, but this shall contain only one Tag.			8402	MagneSafe V5 Commands
			8404	Data Object List request
C104	Device to Host Response	Used to contain responses to Host to Device Request tags	Could include one of the following:	
The list of possible objects is not all inclusive, but this shall contain only one Tag.			8401	Unrecognized
			8403	MagneSafe V5 Response
			C20A	Data Object List response Data
C106	MagneSafe V5 Card Swipe Data	MS V5 compatible Swipe Data	Could include one of the following:	
			C302	Supplemental Information
			C201	Swipe Status
			C202	Magnetic Stripe Local Merchant Data
			C203	Magnetic Stripe Secure Data

Object List Response Data container

Tag	Name	Description	Items contained
C20A	Data Object List response Data	All of the requested tags that are available.	See description

MSR Swipe Data Containers

Tag	Name	Description	Items contained	
C302	Supplemental Information	All Device Information objects	Includes the following tags	
			8109	TLV Version
			8103	Device Main Firmware Part #
			8104	Device Model Name
			8140	Battery Level
			8141	Card Swipe Count
C201	Swipe Status	Used to give information about the status of a swipe.	May include the following tags:	
The list of possible objects is not all-inclusive.			8001	Operation Status
			8261	Card Status
			8262	Track Status

MSR Swipe Data Containers				
C202	Magnetic Stripe Local Merchant Data	Used to give information about a card for use by a merchant in local processing.	May include the following tags, and others.	
The list of possible objects is not all inclusive. * These items do not appear is MagneSafe V5 Card Swipe Data containers.			8241	Cardholder Name *
			8242	Card IIN *
			8243	Last 4 Digits of PAN *
			8244	Expiration Date *
			8245	Service Code *
			8246	Card PAN Length *
			8221	Track 1 Data, Masked
			8222	Track 2 Data, Masked
			8223	Track 3 Data, Masked
C203	Magnetic Stripe Secure Data	Used to give secure information (usually forwarded for processing by a secure service).	May include the following tags, and others.	
The list of possible objects is not all inclusive. * These objects do not appear in MagneSafe V5 Swipe Data containers. ** These objects do not appear in non MagneSafe V5 Swipe Data containers. † This object only appears for devices that support an Internal Serial Number. †† These objects only appear for devices that support advanced features. See the object definition.			8101	Device SN - Internal †
			8102	Device SN - MagTek
			8301	Key Identifier - Magnetic Stripe Data
			8302	Key Type, Magnetic Stripe Data ††
			8303	Key Variant - Magnetic Stripe Data ††
			8211	Encrypted Track 1 Data *
			8212	Encrypted Track 2 Data *
			8213	Encrypted Track 3 Data *
			830A	Encrypted Track 1 Data, raw **
			830B	Encrypted Track 2 Data, raw **
			830C	Encrypted Track 3 Data, raw **
			830E	MagnePrint Status
			8305	Key Identifier - MagnePrint Data ††
			8306	Key Type, MagnePrint Data ††
			8307	Key Variant - MagnePrint Data ††
			8214	Encrypted MagnePrint Data *
			830D	Encrypted MagnePrint Data, raw **
			8308	SHA-x Hash Code ††
			8309	Encrypted Session ID ††

MSR Swipe Data Containers

			8201	Standard Track 1 Data
			8202	Standard Track 2 Data
			8203	Standard Track 3 Data
			8204	MagnePrint Data (Short)
			810C	Unencrypted Session ID

Discovery Container

Tag	Name	Description	Items contained	
C30B	Discovery	All discovery data objects	Includes the following tags	
			C307	Device Information
			C303	Device Status
			C304	Device Configuration
			C305	Device Security (TBD)

Device Information Container

Tag	Name	Description	Items contained	
C307	Device Information	All Device Information objects	Includes the following tags	
			8101	Device SN - Internal
			8102	Device SN - MagTek
			8108	Device Part Number
			8103	Device Main Firmware Part #
			8109	TLV Version
			8104	Device Model Name
			C308	Device Capabilities

Supplemental Information Container

Tag	Name	Description	Items contained	
C302	Supplemental Information	All Device Information objects	Includes the following tags	
			8109	TLV Version
			8103	Device Main Firmware Part #
			8104	Device Model Name
			8140	Battery Level
			8141	Card Swipe Count

Device Status Container				
Tag	Name	Description	Items contained	
C303	Device Status	All Device Status objects	Includes the following tags	
			8140	Battery Level
			8141	Card Swipe Count

Device Configuration Container				
Tag	Name	Description	Items contained	
C304	Device Configuration	All Device Configuration objects	Includes the following tags	
These objects only appear if the device supports the Capability, the objects shown here are only an example.			8160	Send TLV Version on Power Up
			8161	Send Discovery on Power Up
			8162	Send Card Name
			8163	Send Card IIN
			8164	Send Card Last 4 Digits of PAN
			8165	Send Card Expiration Date
			8166	Send Card Service Code
			8167	Send Card PAN Length
			8600	Firmware Upgrade Enabled
			8601	MSR Enabled
			8602	Tracks Enabled
			860C	MSR Decode Options
			8603	Magnetic Stripe Encryption Enabled
			8604	MagnePrint Enabled
			8605	MagnePrint Encryption Enabled
			8606	MagneSafe 2.0 Encryption Enabled
			8607	MSR Data Encryption Variant
			8608	MagnePrint Data Encryption Variant
			8609	Hash Code Enabled
			C30A	MagneSafe V5 Specific Configuration

MagneSafe V5 Specific Configuration Container			
Tag	Name	Description	Items contained
C30A	MagneSafe V5 Specific Configuration	MagneSafe V5 Specific Configuration objects	Includes the following tags

MagneSafe V5 Specific Configuration Container		
	8800	MagneSafe V5 ISO Mask Property
	8801	MagneSafe V5 AAMVA Mask Property
	8802	MagneSafe V5 Mask Other Cards Property
	8803	MagneSafe V5 Send Clear AAMVA Card Data property

Device Security Container				
Tag	Name	Description	Items contained	
C305	Device Security	All Device Security objects	Include the following tags	
				TBD

Capabilities Container				
Tag	Name	Description	Items contained	
C306 / C308	Capabilities	All Capabilities objects	Includes the following tags	
Previously, Capabilities were passed in container C306, but that container was used for other purposes in other MagTek devices. C306 was deprecated in TLV Version 3 in favor of Tag C308. This container contains all of the capability information for the device, what is shown here is only an example.			8120	Capability - MSR
			8121	Capability - Tracks
			8122	Capability - Encryption
			8500	Battery Powered
			8501	Supports Firmware Upgrade
			8502	Track Decode
			8503	Supported TDES DUKPT Key Variants
			8504	Supports Hash Code
			8505	Capability - MagnePrint
			8506	Capability - MagnePrint Encryption
			8507	Capability - MagneSafe 2.0 Encryption
			8508	Supports Card Swipe Counter
			8509	Supports Session ID
			C309	MagneSafe V5 Specific Capabilities

MagneSafe V5 Specific Capabilities Container				
Tag	Name	Description	Items contained	
C309	MagneSafe V5 Specific Capabilities	All MagneSafe V5 Capability information	Includes the following tags	
			8700	MagneSafe V5 Masked Track Support

Appendix G EMV Message Formats (EMV Only)

G.1 ARQC Messages (EMV Only)

This section gives the format of the ARQC Message delivered in **Notification 0x0303 - ARQC Message**. The contents of the ARQC Message is slightly different depending on whether the device is set to **Security Level 2** (not encrypting) or **Security Level 3** (encrypting). Support for EMV transactions at **Security Level 2** is only available on mDynamo.

G.1.1 ARQC Message Format Security Level 2

When the device is set to **Security Level 2** (not encrypting), the ARQC Message TLV data object contains the following:

```
F9<len> /* container for MAC structure and generic data */
           DFDF54(MAC KSN)<len><val>
           DFDF55(MAC Encryption Type)<len><val>
           DFDF25(IFD Serial Number)<len><val>
           FA<len>/* container for generic data */
                70<len> /*container for ARQC */
                      DFDF53<len><value> /*fallback indicator */
                      5F20<len><value> /*cardholder name */
                      5F30<len><value> /*service code */
                      DFDF4D<len><value> /* Masked T2 PICC/ICC Data */
                      DFDF52<len><value> /* card type */
                      <tags defined by DFDF02 >

(Buffer if any to be a multiple of 8 bytes)

CBC-MAC (4 bytes reserved, not calculated)
```

If the device is configured to prefer MSD data, it includes that data in additional TLV data objects in TLV data object 70. See section **G.4 Contactless Magnetic Stripe Data (MSD) Tags (Contactless Only)**.

The device populates TLV data object DFDF53 with one of the following fallback indicators:

- 0x00 = No fallback or missing tag
- 0x01 = Technical Fallback used (EMV MSR Flow Only)
- 0x81 = MSR Fallback used (EMV MSR Flow Only)

The device populates TLV data object DFDF52 with one of the following card types:

- 0x00 = Other
- 0x01 = Financial
- 0x02 = AAMVA
- 0x03 = Manual
- 0x04 = Unknown
- 0x05 = ICC
- 0x06 = Contactless ICC - EMV
- 0x07 = Financial MSR and ICC
- 0x08 = Contactless ICC - MSD

The device constructs the contents of tag DFDF4D, using EMV transaction data to emulate track 2 data as though it came from an ISO/ABA magnetic stripe card. Much of the data is masked; the device sends a specified mask character instead of the actual character from the transaction. The device provides masking settings in **Property 0x07 - ISO Track Mask**, which allows the host software to specify masking details for the Primary Account Number, the masking character to be used, and whether a correction should be applied to make the Mod 10 (Luhn algorithm) digit at the end of the PAN be correct.

Table 9-8 provides an example of track 2 data as it would appear if the device sent it in the clear. **Table 9-9** shows the same data as it might appear with a specific set of masking rules applied.

Table 9-8 – Sample ISO/ABA Swiped Track Data, Clear Text / Decrypted

Sample ISO/ABA Swiped Track Data, Clear Text / Decrypted	
Track 2	;6011000995500000=15121015432112345678?

Table 9-9 – Sample ISO/ABA Swiped Track Data, Masked

Sample ISO/ABA Swiped Track Data, Masked	
Track 2	;6011000020000000=15120000000000000000?

Table 9-10 shows an example of track 2 data using unmasked placeholders to make it easier to see the relative positions of the values embedded in the track data, and can be interpreted as follows:

- ? and ; are Sentinels / delimiters.
- The string of 5s is the Account Number / PAN.
- The string of 3s is the Expiration Date.
- The string of 8s is the Service Code.
- The remaining characters (0s, 4s, and 6) are Discretionary Data, which is of varying length and content and comes from the card, and must be interpreted according to the standards established by issuers, payment brands, and so on.

Table 9-10 – Example Generic ISO/ABA Track Data Format

Generic ISO/ABA Track Data Format	
Track 2 Data	;5555555555555555=33338880004444006?

The device masks the data as follows:

- The number of initial characters and trailing characters specified by **Property 0x07 - ISO Track Mask** is sent unmasked. If Mod 10 correction is specified (see **Property 0x07 - ISO Track Mask**), all but one of the intermediate characters of the PAN are set to zero; one of them is set such that last digit of the PAN calculates an accurate Mod 10 check of the rest of the PAN as transmitted. If the Mod 10 correction is not specified, all of the intermediate characters of the PAN are set to the specified mask character.
- The Expiration Date is transmitted unmasked.
- The Service Code is always unmasked on newer devices (Never Mask Service Code Only). On legacy devices, the Service Code is always masked.
- All Field Separators are sent unmasked.
- All other characters are set to the specified mask character.

G.1.2 ARQC Message Format Security Level 3

When the device is set to **Security Level 3** (encrypting), the ARQC Message TLV data object contains the following:

```
F9<len> /* container for MAC structure and generic data */
           DFDF54(MAC KSN)<len><val>
           DFDF55(MAC Encryption Type)<len><val>
           DFDF25(IFD Serial Number)<len><val>
           FA<len>/* container for generic data */
                70<len> /*container for ARQC */
                      DFDF53<len><value> /*fallback indicator */
                      5F20<len><value> /*cardholder name */
                      5F30<len><value> /*service code */
                      DFDF4D<len><value> /* Masked T2 PICC/ICC Data */
                      DFDF52<len><value> /* card type */
                      F8<len> /*container tag for encryption */
                           DFDF59(Encrypted Data
Primitive)<len><Encrypted Data val (Decrypt data to read tags)>
                           DFDF56(Encrypted Transaction Data
KSN)<len><val>
                           DFDF57(Encrypted Transaction Data
Encryption Type)<val>
                           DFDF58(# of bytes of padding in
DFDF59)<len><val>

(Buffer if any to be a multiple of 8 bytes)
CBC-MAC (4 bytes reserved, not calculated)
```

The values inside tags DFDF52, DFDF53, and DFDF4D are fully described in section **G.1.1**.

The device encrypts the Value inside data container DFDF59 using the **Data Encryption, request or both ways** variant [or other variant depending on **Property 0x67 - EMV Data Encryption Variant (EMV Only)**] of the current DUKPT Key used in the relevant transaction. As a requirement for using the DUKPT TDES encryption algorithm, the device pads it so the length of its value is a multiple of 8 bytes. The device uses tag DFDF58 to report how many bytes of tag DFDF59 are padding. After the host decrypts it, DFDF59 contains a list of TLV data objects defined by terminal setting DFDF02 or DFDF08 is card type is contactless-MSD. For example:

```
                      FC<len>/* container for encrypted generic data *
                           <tags defined by DFDF02 or DFDF08>
                           F4<len>/* container tag for encrypted MSR
data */
                                 DFDF36 <EncT1status><len><val>
                                 DFDF37 <EncT1data><len><val>
                                 DFDF38 <EncT2status><len><val>
                                 DFDF39 <EncT2data><len><val>
                                 DFDF3A <EncT3status><len><val>
                                 DFDF3B <EncT3data><len><val>
                                 DFDF3C <Encrypted Magneprint
Data><len><val>
                                 DFDF43 <Magneprint Status
Data><len><val>
```

```
                                 DFDF50(MSR KSN Data)<len><val> /*sent
in the clear*/
                                 DFDF51(MSR EncryptionType)<len><val>

                      <Padding to force DFDF59 plus padding to be a
multiple of 8 bytes>
```

If the device is configured to prefer MSD data, it includes that data in additional TLV data objects in TLV data object FC. See section **G.4 Contactless Magnetic Stripe Data (MSD) Tags (Contactless Only)**.

(Apple VAS Only)
If Apple VAS is enabled by **Property 0x75 - Apple VAS Support (Apple VAS Only)** and the host invoked **Extended Command 0x0300 - Initiate EMV Transaction (EMV Only)** with options that enable Apple VAS support for the current transaction, the ARQC message also includes Apple VAS data in TLV data object FE after TLV data object F8.

```
                      FE<len>/* container for Apple VAS data *
                           9F27<len><val> /*Mobile Token */
                           9F2A<len><val> /*VAS Data */
```

G.2 ARPC Response from Online Processing (EMV Only)

This section specifies the format of the data for **Extended Command 0x0303 - Online Processing Result / Acquirer Response (EMV Only)**. The host sends this request to the device in response to **Notification 0x0303 - ARQC Message**.

An ARPC Response is a TLV data object with the following contents:

```
F9<len> /* container for MAC structure and generic data */
     DFDF54 (MAC KSN)<len><val>
     DFDF55 (Mac Encryption Type><len><val>
     DFDF25 (IFD Serial Number)<len><val>
FA<len> /* Container for generic data */
     70 04 8A 02 30 30
     (ARPC padding, if any, to be a multiple of 8 bytes)
CBC-MAC (4 bytes, reserved, must be sent to the device, however, the
device does not check for the properly calculated CBC-MAC)
```

G.3 Transaction Result Messages (EMV Only)

This section specifies the format for data the device sends using **Notification 0x0304 - Transaction Result Message**.

TLV data object `DFDF1A` contains one of the following Transaction Status values:

- 0x00 = Approved
- 0x01 = Declined
- 0x02 = Error
- 0x03 = Try Another Interface
- 0x04 = Application Blocked
- 0x10 = Cancelled by Host
- 0x1E = Manual Selection Cancelled by Host
- 0x1F = Manual Selection Timeout
- 0x21 = Waiting for Card Cancelled by Host
- 0x22 = Waiting for Card Timeout
- 0x23 = Cancelled by Card Swipe (MSR Only)
- 0xFF = Unknown

Transaction Status values 0x03 and 0x04 are supported by ODM models only.

(Contactless Only) If data object DFDF1A reports **0x02 Error**, data object DFDF1B may contain additional information about the error using one of the following values:

- 0x50 = Select Application Error
- 0x51 = Application Initiation Error
- 0x52 = Read Record Error
- 0x53 = Offline Data Authentication Error
- 0x54 = Process Restriction Error
- 0x55 = Cardholder Verification Error
- 0x56 = Risk Management Error
- 0x57 = 1st Terminal Action Analysis Error
- 0x58 = 1st Generate AC Error
- 0x59 = 1st Card Action Analysis Error
- 0x5a = Online Processing Error
- 0x5b = Online Response Process Error
- 0x5c = 2nd Card Action Analysis Error
- 0x5D = MSD Card Reading Error

The format of Transaction Result messages depends on whether the device is set to **Security Level 2** (not encrypting) or **Security Level 3** (encrypting). Support for EMV transactions at **Security Level 2** is only available on mDynamo.

G.3.1 Transaction Result Message Format Security Level 2

When the device is set to **Security Level 2** (not encrypting), the Transaction Result TLV data object contains the following:

```
F9<len> /* container for MAC structure and generic data */
      DFDF54(MAC KSN)<len><val>
      DFDF55(MAC Encryption Type)<len><val>
      DFDF25(IFD Serial Number)<len><val>
      FA<len>/* container for generic data */
            F0<len> /* Transaction Results */
                  F1<len> /* container for Status Data */
                        /* Status Data tags */
                        DFDF1A - Transaction Status
                        DFDF1B - Additional Transaction Information

                  F2<len>/* container for Transaction Data */
                        /* Data tags (defined by DFDF17) */

                  F3<len>/* container for Reversal Data, if any */
                        /* Reversal Data tags (defined by DFDF05)*/

                  F7<len>/* container for Merchant Data */
                        /* Merchant Data tags */
                        5F25<len> /* Application Effective Date */
                        5F24<len> /* Application Expiration Date */
                        89<len> /* Authorization Code */
                        5F2A<len> /* Transaction Currency Code */
                        9F02<len> /* Amount, Authorized */
                        9F03<len> /* Amount, Other */
                        9F06<len> /* Application Identifier */
                        9F12<len> /* Application Preferred Name */
                        9F1C<len> /* Terminal Identification */
                        9F39<len> /* POS Entry Mode */
                        9C<len> /* Transaction Type */
                        9F34<len> /* Cardholder Verification Results */
                        5F57<len> /* Account Type */
                        5F20<len> /* Cardholder Name */
                        DFDF4D<len> /* Masked T2 PICC/ICC Data */

(Buffer if any to be a multiple of 8 bytes)
CBC-MAC (4 bytes reserved, not calculated)
```

The value inside tag DFDF4D is fully described in section **G.1.1**.

If the device is configured to prefer MSD data, it includes that data in additional TLV data objects in TLV data object F2. See section **G.4 Contactless Magnetic Stripe Data (MSD) Tags (Contactless Only)**.

G.3.2 Transaction Result Message Format Security Level 3

When the device is set to **Security Level 3** (encrypting), the Transaction Result TLV data object contains the following:

```
F9<len> /* container for MAC structure and generic data */
      DFDF54(MAC KSN)<len><val>
      DFDF55(MAC Encryption Type)<len><val>
      DFDF25(IFD Serial Number)<len><val>
      FA<len>/* container for generic data */
           F0<len> /* Transaction Results */

                F1<len> /* container for Status Data */
                      … /* Status Data tags */
                      DFDF1A - Transaction Status
                      DFDF1B - Additional Transaction Information

                F8<len> /* container tag for encryption */
                      DFDF59(Encrypted Data Primative)<len><Encrypted
Data val (Decrypt data to read tags)>
                      DFDF56(Encrypted Transaction Data KSN)<len><val>
                      DFDF57(Encrypted Transaction Data Encryption
Type)<val>
                      DFDF58(# of bytes of padding in DFDF59)<len><val>

                F7<len>/* container for Merchant Data */
                      /* Merchant Data tags */
                      5F25<len> /* Application Effective Date */
                      5F24<len> /* Application Expiration Date */
                      89<len> /* Authorization Code */
                      5F2A<len> /* Transaction Currency Code */
                      9F02<len> /* Amount, Authorized */
                      9F03<len> /* Amount, Other */
                      9F06<len> /* Application Identifier */
                      9F12<len> /* Application Preferred Name */
                      9F1C<len> /* Terminal Identification */
                      9F39<len> /* POS Entry Mode */
                      9C<len> /* Transaction Type */
                      9F34<len> /* Cardholder Verification Results */
                      5F57<len> /* Account Type */
                      5F20<len> /* Cardholder Name */
                      DFDF4D<len> /* Masked T2 PICC/ICC Data */

(Buffer if any to be a multiple of 8 bytes)
CBC-MAC (4 bytes reserved, not calculated)
```

The value inside tag DFDF4D is fully described in section **G.1.1**.

The device encrypts the Value inside data container DFDF59 using the **Data Encryption, request or both ways** variant [or other variant depending on **Property 0x67 - EMV Data Encryption Variant (EMV Only)**] of the current DUKPT Key used in the relevant transaction. As a requirement for using the DUKPT TDES encryption algorithm, the device pads it so the length of its value is a multiple of 8 bytes. The device uses tag DFDF58 to report how many bytes of tag DFDF59 are padding. After the host decrypts it, DFDF59 contains a list of TLV data objects defined by terminal setting DFDF17 and DFDF05. For example:

```
           FC<len>/* container for encrypted generic data */
                F2<len>/* container for Transaction Data */
                     … /* Data tags (defined by DFDF17) */
                F3<len>/* container for Reversal Data, if any */
                     … /* Reversal Data tags (defined by DFDF05)*/
```

If the device is configured to prefer MSD data, it includes that data in additional TLV data objects in TLV data object F2. See section **G.4 Contactless Magnetic Stripe Data (MSD) Tags (Contactless Only)**.

G.4 Contactless Magnetic Stripe Data (MSD) Tags (Contactless Only)

Some solutions may require the device to send ISO Track 1 and Track 2 equivalent data (“MSD mode” data) during contactless transactions, for compatibility with legacy hosts and back-end systems that are designed to use magnetic stripe data. Most payment methods (contactless cards and payment devices) support both MSD and EMV data modes, and the device’s kernel configuration(s) can be set to prefer MSD and to direct the payment method to provide it during a transaction.

If the device is configured to prefer MSD data, it includes that data in additional TLV data objects, listed in **Table 9-11** and **Table 9-12**, when it sends the host **ARQC Messages (EMV Only)** data container **FA**, or **Transaction Result Messages (EMV Only)** data container **F2**.

The device’s MSD preference is set differently for each payment brand. See the following tags to set or to determine the setting for each brand:

- Use tag 9F66 in Terminal Configuration.
- **EMV Contact Settings (Contact Only**) use tag DF811B in Application Configuration.
- **QuickPass** does not support MSD (QuickPass Support Only)

Table 9-11 - MCL MSD Data Tags

Tag	Description
56	MSD Track 1
9F6B	MSD Track 2 Although the payment method’s MCL application may send Track 2 equivalent data to the device using data container **57**, the device re-wraps it and sends it to the host using this data container instead.

Table 9-12 - payWave, Expresspay and D-PAS MSD Data Tags

Tag	Description
DFDF4B	MSD Track 1 Expresspay payment methods will not include the Longitudinal Redundancy Code (LRC).
DFDF4C	MSD Track 2 Expresspay payment methods will not include the Longitudinal Redundancy Code (LRC).

Appendix H EMV Terminal and Application Settings (EMV Only)

H.1 EMV Common Settings

This section lists settings that are common across all EMV databases on the device.

H.1.1 EMV Common Terminal Settings and Defaults

This section lists the default EMV Terminal Settings shared across all terminal databases on the device. For information about reading and changing these settings, see section **8.4.8 Extended Command 0x0306 - Read Terminal Configuration** and section **8.4.7 Extended Command 0x0305 - Modify Terminal Configuration**.

Table 9-13 - EMV Common Terminal Settings

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
5F2A	08 40	0x02	MagTek	Transaction Currency Code. Valid values are the numerical codes from ***ISO 4217 Codes for the representation of currencies***, for example: • 0x0000 = Use Selected Application's Currency Code Terminal Setting • 0x0840 = US Dollar • 0x0978 = Euro
5F36	02	0x01	MagTek	Transaction Currency Exponent
9F1A	08 40	0x02	MagTek	Terminal Country Code. The device's terminal country codes are numeric and derived from ***ISO 3166-1***, for example: • 0840 = United States • 0250 = France • 0380 = Italy • 0724 = Spain • 0276 = Germany
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal Identification
9F4E	30 30 30 30 30 30 30	0x28	MagTek	Merchant Name and Location
DFDF14	00 00 75 30	0x04	MagTek	Socket Timeout for Online Processing (ms)
DFDF15	00 00 00 01	0x04	MagTek	Socket Retries (number of connection retries in Online Processing)

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
DFDF19	65 6E	0x02	MagTek	(Multi-Language Only) Default Terminal Language. The device's terminal language codes are ASCII strings based on alpha-2, derived from ***ISO 3166-1***, for example: • 656E = English (en) • 6672 = French (fr) • 6974 = Italian (it) • 6465 = German (de) • 6573 = Spanish (es)
DFDF2D	65 6E 64 65	0x0A	Read Only	(Multi-Language Only) Supported Terminal Languages. The device's terminal language codes are ASCII strings based on alpha-2, derived from ***ISO 3166-1***, for example: • 656E = English (en) • 6672 = French (fr) • 6974 = Italian (it) • 6465 = German (de) • 6573 = Spanish (es)

H.1.2 EMV Common Application Settings and Defaults

There are no default EMV Application Settings shared across all application databases on the device.

H.2 EMV Contact Settings (Contact Only)

H.2.1 EMV Contact Terminal Settings and Defaults (Contact Only, not 4.3i Format)

This section lists the default EMV Contact Terminal default settings. For information about reading and changing these settings, see section **8.4.8 Extended Command 0x0306 - Read Terminal Configuration** and section **8.4.7 Extended Command 0x0305 - Modify Terminal Configuration**.

Table 9-14 - EMV Contact Terminal Settings

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Terminal Settings** from section **H.1.1**				
9F15	30 30	0x02	MagTek	Merchant Category Code
9F16	30 30 30 30 30 30 30	0x0F	MagTek	Merchant Identifier
9F33	20 28 C8	0x03	MagTek	Terminal Capabilities (Set by Terminal Configuration, see section **8.4.17**)
9F35	21	0x01	MagTek	Terminal Type (Set by Terminal Configuration, see section **8.4.17**)
9F3C	09 98	0x02	MagTek	Transaction Reference Code
9F3D	02	0x01	MagTek	Transaction Currency Exponent
9F40	72 00 00 B0 01	0x05	MagTek	Additional Terminal Capabilities (Set by Terminal Configuration, see section **8.4.17**)
DFDF01	A0 00 00 00 04 F8 00 10 00	0x09	MagTek	Certificate Revocation List
DFDF02	9A DF DF 28 9F 02 5A 89 9F 10 9F 15 9F 16 9F 4E 82 8E 5F 24 5F 25 9F 06 9F 07 9F 0D 9F 0E 9F 0F 9F 26 9F 27 9F 36 9C 9F 33 9F 34 9F 37 9F 39 9F 40 95 9B 9F 5B DF DF 00 9F 1E 9F 1A 5F 2A 9F 01 9F 21 8A DF 81 20 DF 81 21 DF 81 22 5F 20 50 5F 34 84 9F 03 9F 09 9F 1E 9F 35 9F 41 9F 53 F4	0x81	MagTek	Online message for EMV transaction

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
DFDF05	9A 82 9F 36 9F 1E 9F 10 9F 5B 9F 33 9F 35 95 9F 01 5F 24 5A 5F 34 8A 9F 15 9F 16 9F 39 9F 1A 9F 1C 57 9F 02 5F 2A 9F 21 9C	0x80	MagTek	Reversal message for EMV transaction
DFDF06	8A 91	0x02	MagTek	Tags participating in online response
DFDF16	00 00 00 80	0x04	MagTek	Maximum length of issuer script (Read Only)
DFDF17	9A DF DF 28 9F 02 9F 03 5A 89 9F 10 9F 15 9F 16 9F 4E 82 8E 5F 24 5F 25 9F 06 9F 07 9F 0D 9F 0E 9F 0F 9F 26 9F 27 9C 9F 33 9F 34 9F 35 9F 36 9F 37 9F 39 9F 40 9F 41 9F 53 95 9B 9F 5B DF DF 00 9F 1E 9F 1A 5F 2A 9F 01 8A DF 81 20 DF 81 21 DF 81 22 5F 20 5F 34 9F 09 84	0x80	MagTek	EMV Transaction Result Message Tags

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
DFDF20	43 28	0x02	Read Only	Terminal Features, read only Byte 1: Bit 8 TAC/IAC-Default process when unable to go online Bit 7 Manual Language Selection Enabled Bit 6 Referrals are supported Bit 5 CDA Failure detected prior to TAA is enabled Bit 4/Bit 3 0b00 = CDA Mode 1 is enabled, 0b01 = CDA Mode 2 is enabled, 0b10 = CDA Mode 3 is enabled, 0b11 = CDA Mode 4 is enabled Bit 2 Cardholder Confirmation is enabled Bit 1 EMV Language Selection is enabled Byte 2: Bit 8 RFU Bit 7 'Forced Acceptance' is enabled Bit 6 'Application Preferred Order' is enabled Bit 5 'Transaction log' is enabled Bit 4 'Revocation of Issuer Public Key' is enabled Bit 3 'Account Type selection' is enabled Bit 2 'Subsequent Bypass PIN Entry' is enabled Bit 1 'Bypass PIN Entry' is enabled
DFDF26	4D 41 47 54 45 4B 20 44 45 46 41 55 4C 54	0x10	MagTek	EMV Database Label
DFDF5B	0C	0x01	MagTek	Terminal Capabilities for Purchase transaction
DFDF5C	02	0x01	MagTek	Terminal Capabilities for Cashback transaction
DFDF67 (EMV MSR Flow Only)	01	0x01	MagTek	MSR Fallback Supported
DFDF6E	0C	0x01	MagTek	Terminal Capabilities for Payment transaction
DFDF75	0C	0x01	MagTek	Terminal Capabilities for Inquiry (Not Supported)

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
DFDF76	0C	0x01	MagTek	Terminal Capabilities for Transfer (Not Supported)

H.2.2 EMV Contact Application Settings and Defaults (Contact Only, not 4.3i Format)

This section lists the default EMV Contact Application Settings. For information about reading and changing these settings, see section **8.4.10 Extended Command 0x0308 - Read Application Configuration** and section **8.4.9 Extended Command 0x0307 - Modify Application Configuration**.

Table 9-15 - EMV Contact Application Slot 1 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 25 01	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 25 01	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 01	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-16 - EMV Contact Application Slot 2 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 06 20 06 20	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 06 20 06 20	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 01	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	00 00 00	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	FC 50 AC A0 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC 50 BC F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-17 - EMV Contact Application Slot 3 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 01 52 30 10	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 01 52 30 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 01	0x02	MagTek	Application Version Number
9F1B	00 00 27 10	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	DC 00 00 20 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC E0 9C F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-18 - EMV Contact Application Slot 4 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 98 08 40	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 98 08 40	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 01	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	DC 00 00 20 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC E0 9C F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-19 - EMV Contact Application Slot 5 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 02 77 10 10	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 02 77 10 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 01	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	FC 50 F8 A8 F0	0x05	MagTek	Terminal Action Code - Default
DF8121	10 10 58 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC F8 E4 B8 70	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-20 - EMV Contact Application Slot 6 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 65 10 10	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 65 10 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	02 00	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	FC 60 24 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC 60 AC F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-21 - EMV Contact Application Slot 7 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 04 10 10	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06 5F 2A 02 9A 03 9C 01 95 05 9F 37 04	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 04 10 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 02	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	FC 50 B8 A0 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC 50 B8 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-22 - EMV Contact Application Slot 8 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 04 30 60	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06 5F 2A 02 9A 03 9C 01 95 05 9F 37 04	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 04 30 60	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 02	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	FC 50 BC A0 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC 50 BC F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-23 - EMV Contact Application Slot 9 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 04 22 03	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06 5F 2A 02 9A 03 9C 01 95 05 9F 37 04	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 04 22 03	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 02	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	FC 50 BC A0 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC 50 BC F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-24 - EMV Contact Application Slot 10 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 03 33 01 01 01	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 03 33 01 01 01	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 20	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	D8 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-25 - EMV Contact Application Slot 11 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 03 33 01 01 02	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 03 33 01 01 02	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 20	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	D8 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-26 - EMV Contact Application Slot 12 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 03 33 01 01 03	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 03 33 01 01 03	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 20	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	D8 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-27 - EMV Contact Application Slot 13 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 03 10 10	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 03 10 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 8C	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	DC 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-28 - EMV Contact Application Slot 14 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 03 20 10	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 03 20 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 8C	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	DC 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-29 - EMV Contact Application Slot 15 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 03 30 10	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 03 30 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 8C	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	DC 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-30 - EMV Contact Application Slot 16 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	00	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	00	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 8C	0x02	MagTek	Application Version Number
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F49	9F 37 04	0x0A	MagTek	Default DDOL
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	DC 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

H.2.3 EMV Contact Terminal Settings and Defaults (Contact 4.3i Format Only)

This section lists the default EMV Contact Terminal default settings. For information about reading and changing these settings, see section **8.4.8 Extended Command 0x0306 - Read Terminal Configuration** and section **8.4.7 Extended Command 0x0305 - Modify Terminal Configuration**.

Table 9-31 - EMV Contact Terminal Settings

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Terminal Settings** from section **H.1.1**				
9F1D	31 31 32 32 33 33 34 34	0x02	MagTek	Terminal Risk Management Data
9F33	20 28 C8	0x03	MagTek	Terminal Capabilities (Set by Terminal Configuration, see section **8.4.17**)
9F35	21	0x01	MagTek	Terminal Type (Set by Terminal Configuration, see section **8.4.17**)
9F3A	00 00 00 00	0x04	MagTek	Amount,Reference Currency
9F3C	09 98	0x02	MagTek	Transaction Reference Code
9F3D	02	0x01	MagTek	Transaction Currency Exponent
9F40	72 00 00 B0 01	0x05	MagTek	Additional Terminal Capabilities (Set by Terminal Configuration, see section **8.4.17**)
DF812D	00 00 00	0x03	MagTek	Message Hold Time
DFDF01	A0 00 00 00 04 F8 00 10 00	0x09	MagTek	Certificate Revocation List
DFDF02	9A DF DF 28 9F 02 5A 89 9F 10 9F 15 9F 16 9F 4E 82 8E 5F 24 5F 25 9F 06 9F 07 9F 0D 9F 0E 9F 0F 9F 26 9F 27 9F 36 9C 9F 33 9F 34 9F 37 9F 39 9F 40 95 9B 9F 5B DF DF 00 9F 1E 9F 1A 5F 2A 9F 01 9F 21 8A DF 81 20 DF 81 21 DF 81 22 5F 20 50 5F 34 84 9F 03 9F 09 9F 1E 9F 35 9F 41 9F 53 F4	0x81	MagTek	Online message for EMV transaction

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
DFDF05	9A 82 9F 36 9F 1E 9F 10 9F 5B 9F 33 9F 35 95 9F 01 5F 24 5A 5F 34 8A 9F 15 9F 16 9F 39 9F 1A 9F 1C 57 9F 02 5F 2A 9F 21 9C	0x80	MagTek	Reversal message for EMV transaction
DFDF06	8A 91	0x02	MagTek	Tags participating in online response
DFDF16	00 00 00 80	0x04	MagTek	Maximum length of issuer script (Read Only)
DFDF17	9A DF DF 28 9F 02 9F 03 5A 89 9F 10 9F 15 9F 16 9F 4E 82 8E 5F 24 5F 25 9F 06 9F 07 9F 0D 9F 0E 9F 0F 9F 26 9F 27 9C 9F 33 9F 34 9F 35 9F 36 9F 37 9F 39 9F 40 9F 41 9F 53 95 9B 9F 5B DF DF 00 9F 1E 9F 1A 5F 2A 9F 01 8A DF 81 20 DF 81 21 DF 81 22 5F 20 5F 34 9F 09 84	0x80	MagTek	EMV Transaction Result Message Tags

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
DFDF20	63 28 E0	0x03	Read Only	Terminal Features Byte 1: Bit 8 TAC/IAC-Default process when unable to go online Bit 7 Manual Language Selection Enabled Bit 6 Referrals are supported Bit 5 CDA Failure detected prior to TAA is enabled Bit 4 / Bit 3 0x00 = CDA Mode 1 is enabled, 0x01 = CDA Mode 2 is enabled, 0x10 = CDA Mode 3 is enabled, 0x11 = CDA Mode 4 is enabled Bit 2 Cardholder confirmation is enabled Bit 1 EMV Language Selection is enabled Byte 2: Bit 8 RFU Bit 7 'Forced Acceptance' is enabled Bit 6 'Application Preferred Order' is enabled Bit 5 'Transaction log' is enabled Bit 4 'Revocation of Issuer Public Key' is enabled Bit 3 'Account Type selection' is enabled Bit 2 'Subsequent Bypass PIN Entry' is enabled Bit 1 'Bypass PIN Entry' is enabled Byte 3: Bit 8 Floor Limit Checking Enabled Bit 7 Random Transaction Selection Enabled Bit 6 Velocity Checking Enabled Bit 1..5 Reserved
DFDF21	10	0x01	Compile Only	Number of Applications
DFDF26	4D 41 47 54 45 4B 20 44 45 46 41 55 4C 54	0x10	MagTek	EMV Database Label
DFDF47	Calculated	0x04	Calculated	Terminal/Applications/CAPK crc
DFDF4E	00000110	0x04	MagTek	Transaction Reference Currency Conversion

H.2.4 EMV Contact Application Settings and Defaults (Contact 4.3i Format Only)

This section lists the default EMV Contact Application Settings. For information about reading and changing these settings, see section **8.4.10 Extended Command 0x0308 - Read Application Configuration** and section **8.4.9 Extended Command 0x0307 - Modify Application Configuration**.

Table 9-32 - EMV Contact Application Slot 1 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 25 01	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 25 01	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 01	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-33 - EMV Contact Application Slot 2 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 06 20 06 20	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 06 20 06 20	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 01	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	00 00 00	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	FC 50 AC A0 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC 50 BC F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-34 - EMV Contact Application Slot 3 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 01 52 30 10	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 01 52 30 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 01	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 27 10	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	DC 00 00 20 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC E0 9C F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-35 - EMV Contact Application Slot 4 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 98 08 40	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 98 08 40	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 01	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	DC 00 00 20 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC E0 9C F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-36 - EMV Contact Application Slot 5 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 02 77 10 10	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 02 77 10 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 01	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	FC 50 F8 A8 F0	0x05	MagTek	Terminal Action Code - Default
DF8121	10 10 58 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC F8 E4 B8 70	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-37 - EMV Contact Application Slot 6 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 65 10 10	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 65 10 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	02 00	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	FC 60 24 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC 60 AC F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-38 - EMV Contact Application Slot 7 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 04 10 10	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06 5F 2A 02 9A 03 9C 01 95 05 9F 37 04	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 04 10 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 02	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	FC 50 B8 A0 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC 50 B8 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-39 - EMV Contact Application Slot 8 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 04 30 60	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06 5F 2A 02 9A 03 9C 01 95 05 9F 37 04	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 04 30 60	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 02	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	FC 50 BC A0 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC 50 BC F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-40 - EMV Contact Application Slot 9 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 04 22 03	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06 5F 2A 02 9A 03 9C 01 95 05 9F 37 04	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 04 22 03	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 02	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	FC 50 BC A0 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 00 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	FC 50 BC F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-41 - EMV Contact Application Slot 10 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 03 33 01 01 01	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 03 33 01 01 01	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 20	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	D8 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-42 - EMV Contact Application Slot 11 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 03 33 01 01 02	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 03 33 01 01 02	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 20	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	D8 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-43 - EMV Contact Application Slot 12 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 03 33 01 01 03	0x10	MagTek	Dedicated File (DF) Name
97	00 00 00	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 03 33 01 01 03	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 20	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	D8 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-44 - EMV Contact Application Slot 13 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 03 10 10	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 03 10 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 8C	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	DC 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-45 - EMV Contact Application Slot 14 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 03 20 10	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 03 20 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 8C	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	DC 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-46 - EMV Contact Application Slot 15 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	A0 00 00 00 03 30 10	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	A0 00 00 00 03 30 10	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 8C	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	DC 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

Table 9-47 - EMV Contact Application Slot 16 Data

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
All **EMV Common Application Settings and Defaults** from section **H.1.2**				
84	00	0x10	MagTek	Dedicated File (DF) Name
97	9F 02 06	0x0F	MagTek	Default TDOL
9F01	00 00 00 00 00 01	0x06	MagTek	Acquirer Identifier
9F06	00	0x10	MagTek	Application Identifier (AID) - terminal
9F09	00 8C	0x02	MagTek	Application Version Number
9F15	30 30	0x02	MagTek	Marchant Category Code
9F16	4D 61 67 54 65 6B	0x0F	MagTek	Merchant Identifier
9F1B	00 00 00 00	0x04	MagTek	Terminal Floor Limit
9F1C	31 31 32 32 33 33 34 34	0x08	MagTek	Terminal ID
9F49	9F 37 04	0x0A	MagTek	Default DDOL
9F4E	4D 61 67 54 65 6B	0x40	MagTek	Merchant Name and Location
DFDF23	01	0x01	MagTek	Application Selection Indicator (ASI)
DF8120	DC 40 00 A8 00	0x05	MagTek	Terminal Action Code - Default
DF8121	00 10 00 00 00	0x05	MagTek	Terminal Action Code - Denial
DF8122	D8 40 04 F8 00	0x05	MagTek	Terminal Action Code - Online
DFDF10	00 00 00 00 00 00	0x06	MagTek	Threshold Value for Biased Random Selection
DFDF11	63	0x01	MagTek	Target Percentage to be used for Random Selection (0 - 63 hex)
DFDF12	63	0x01	MagTek	Maximum Target Percentage to be used for Biased Random Selection (0 - 63 hex)
DFDF67	01	0x01	MagTek	MSR Fallback Supported (Not Supported)
DFDF68	00	0x01	MagTek	PIN Bypass Supported (Not Supported)

H.3 APPLE VAS Settings (Contactless Only, Apple VAS Only)

This section lists factory defaults for the configurable tags supported by Apple VAS

H.3.1 APPLE VAS Application Settings and Factory Defaults (Apple VAS Only)

There are 6 Application Slots.

Table 9-48 - APPLE VAS Application Slot 1-6 Defaults

Tag	Default Value (Hex)	Max. Length	Configurable	Tag Description
9F25	Empty	0x20	MagTek	Merchant ID
9F29	Empty	0x40	MagTek	Merchant URL