MAGTEK®
SECURITY FROM THE INSIDE

SDK - MagneFlex Powder, Middleware

PIN PEDs

Programmer’s Manual (MagneFlex Powder V2 API)

August 2024

Manual Part Number:
D998200582-102

REGISTERED TO ISO 9001:2015

Table 0.1 – Revisions

Rev Number	Date	Notes
10	June 28, 2023	Initial Release. Based from DynaPro and extended to support DynaFlex.
100	January 22, 2024	Added support for AdditionalTags to DynaFlex RequestSmartCardV1 at section 3.12. Added support for DynaFlex II Go at sections 1.5, 2.3, and A.4.
101	June 24, 2024	Added RawUserSelectData to RequestSmartCardV1 response at section 3.12. Added RequestUserSelectionV1 at section 3.13.
102	August 7, 2024	Updated ReleaseDeviceV1 to cancel PIN and SmartCard request (Section 3.4). Added support for SuppressThankYouMessage to RequestSmartCardV1 (Section 3.12).

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 ON THE FRONT PAGE OF 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 PC 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 HE 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.

Table of Contents

1 Introduction

This document provides instructions for software developers to create software solutions that include IPAD, DynaPro, DynaPro Mini, and DynaProx, DynaFlex, DynaFlex II PED, and DynaFlex II Go. This document is part of a library of documents which includes the following from MagTek:

- ***D99875585 DYNAPRO PROGRAMMER'S MANUAL (COMMANDS)***
- ***D99875629 DYNAPRO MINI PROGRAMMER'S MANUAL (COMMANDS)***
- ***D99875430 IPAD PROGRAMMER'S MANUAL (COMMANDS)***
- ***D998200383 DYNAFLEX PRODUCTS PROGRAMMER'S MANUAL (COMMANDS)***

1.1 About the MagneFlex Powder V2 API

The MagneFlex Powder V2 API provides a convenient HTTP command application programming interface to a device connected to a host. An HTTP client makes JSON calls to the host that are mapped to the device's low-level command set, as found in the Programmer's Manual (Commands). The MagneFlex Powder V2 can be launched as either a Windows Service, or a through a standalone executable.

The SDK also includes a sample SOAPUI project and Java jQuery html file that demonstrates JSON calls to the MagneFlex Powder V2. In addition, source code for the standalone executable is provided, if the developer wishes to integrate the MagneFlex Powder V2 directly into their own code.

The MagneFlex Powder V2 API is single-threaded. If the service is busy processing a command to the device, other calls will be rejected.

1.2 Nomenclature

In this document, the nomenclature below are used as follows:

- **Device** refers to the device (e.g., DynaPro, DynaFlex II PED) that receives and responds to commands.
- **Host** refers to the piece of general-purpose electronic computing equipment the device is connected or paired to. Host sends data to and receive data from the device via the MagneFlex Powder V2 API.
- **V1** at the end of a resource name identifies an extension of the API to be used but not limited for a DynaFlex device.

  For example:

  RequestSmartCard is for DynaPro devices.

  RequestSmartCardV1 is an extension for DynaFlex devices.
- **User** in this document generally refers to the cardholder.

1.3 SDK Contents

Executables:

File	Description
MTUSDK.WEBAPI.Host.exe	MagTek WEBAPI executable
MTUSDK.WEBAPI.Host.exe.config	MagTek WEBAPI executable configuration file
MTUSDK.WEBAPI.HostService.exe	MagTek WEBAPI Windows service
MTUSDK.WEBAPI.HostService.exe.config	MagTek WEBAPI Windows service configuration files

Sample SOAPUI project:

File	Description
MagneFlex Powder Dual Sample-soapui-project.xml	Sample SOAPUI project file

Sample Java web page:

File	Description
Sample.html	Sample Java jQuery html file
tlvdecoder.js	Java TLV decoder

1.4 System Requirements

Tested operating systems:
Windows 7
Windows 8
Windows 8.1
Windows 10

Microsoft .Net Framework 4.5 installed. (The API installation process will install this if it does not already exist on the host.)

Tested development environments:
Windows 10 with Microsoft Visual Studio 2017

1.5 Interfaces for Operating Systems

The following table matches the device interface to operating system.

Device	Interface	Operating System
DynaPro	USB	Windows 7, Windows 8, 8.1 & Windows 10
	ETHERNET	Windows 7, Windows 8, 8.1 & Windows 10
DynaPro Mini	USB	Windows 7, Windows 8, 8.1 & Windows 10
	BLE	Windows 8, 8.1 & Windows 10
DynaPro Go	USB	Windows 7, Windows 8, 8.1 & Windows 10
	BLE	Windows 8, 8.1 & Windows 10
	802.11 Wireless	Windows 8, 8.1 & Windows 10
IPAD	USB	Windows 7, Windows 8, 8.1 & Windows 10
DynaProx	USB	Windows 7, Windows 8, 8.1 & Windows 10
DynaFlex	USB	Windows 7, Windows 8, 8.1 & Windows 10
DynaFlex II PED	USB	Windows 7, Windows 8, 8.1 & Windows 10
DynaFlex II Go	USB	Windows 7, Windows 8, 8.1 & Windows 10
	BLUETOOTH LE	Windows 8, 8.1 & Windows 10

2 How to Set Up the MagneFlex Powder V2 API

2.1 How to Connect MagneFlex Powder V2 Service to a Host App

To use the MagneFlex Powder (MUSDK.WEBAPI.HostService.exe)

1) In all Request, set the header ContentType to `"application/json"`
2) Build the JSON object for the MagneFlex Powder V2 resource to be accessed.
3) Send HTTP request methods GET and POST to the base address http://localhost:9001/api/mtppscrahost/ with the resource endpoint concatenated.

2.2 How to Connect MagneFlex Powder V2 Service to Java jQuery

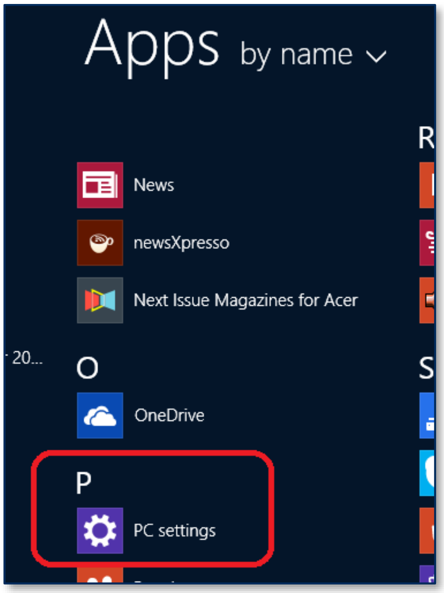
To use the MagneFlex Powder V2 (MTUSDK.WEBAPI.HostService.exe)

1) In all Request, set the header ContentType to `"application/json"`
2) Build the JSON object for the MagneFlex Powder V2 resource to be accessed.
3) Send HTTP request methods GET and POST to the base address http://localhost:9001/api/mtppscrahost/ with the resource endpoint concatenated.
4) Access the Sample.html file in a browser by navigating to http://localhost:9001/sample/Sample.html
5) The Sample.html includes jQuery Ajax sample code in the <script> portion of the html body.
6) The Sample.html page demonstrates the functions of the API.

2.3 How to Connect DynaPro Mini or DynaFlex II Go to a Windows Host via BLE

To connect the device to a host with Windows 8.1 or higher and Bluetooth 4.0 hardware that supports BLE, follow these steps:

1) If you are using an external Bluetooth adapter, install any required drivers and connect it to the host.
2) On the host, install and configure the software you intend to use:
   a) Make sure the host software is configured to look for the device on the proper connection.
   b) Make sure the host software knows with which device(s) it should interface.
   c) Make sure the host software is configured to properly interpret incoming data from the device. This depends on whether the device is configured to transmit data in GATT format or streaming format emulating a keyboard.
3) Make sure the DynaPro Mini or DynaFlex II Go has an adequate charge.
4) Unpair from any other host it is already paired with before continuing.
5) Enter app mode, scroll down to **Apps by name**, and launch the Windows **PC Settings** app.



6) In the left side navigator, select **PC and devices** > **Bluetooth**.
7) Make sure Bluetooth is turned on and close the **PC and devices** app.
8) If using Dyna Pro Mini, place it into pairing mode by sequence F1-1-2-3-F3.
9) If using DynaFlex II Go, place it into pairing mode by pressing and holding its power button until 4 beeps and release the button. The 4th LED will blink green.
10) Launch the Windows **Manage Bluetooth Devices** app by following these steps:
    a) Enter desktop mode by swiping in from the left side of the touchscreen.
    b) Touch the Bluetooth icon in the system tray and select **Add a Bluetooth Device** (see **Figure 2-1**).

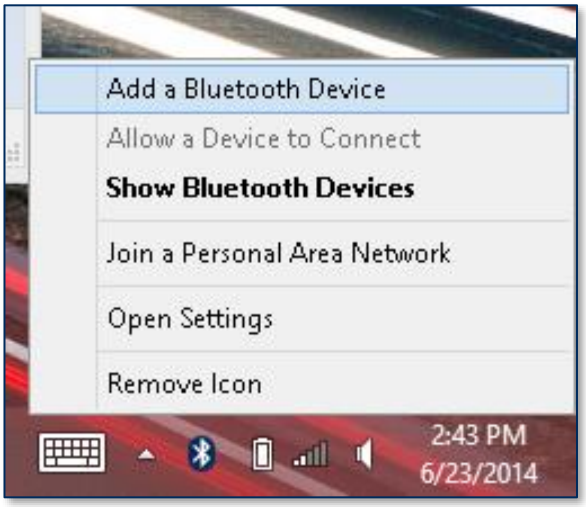
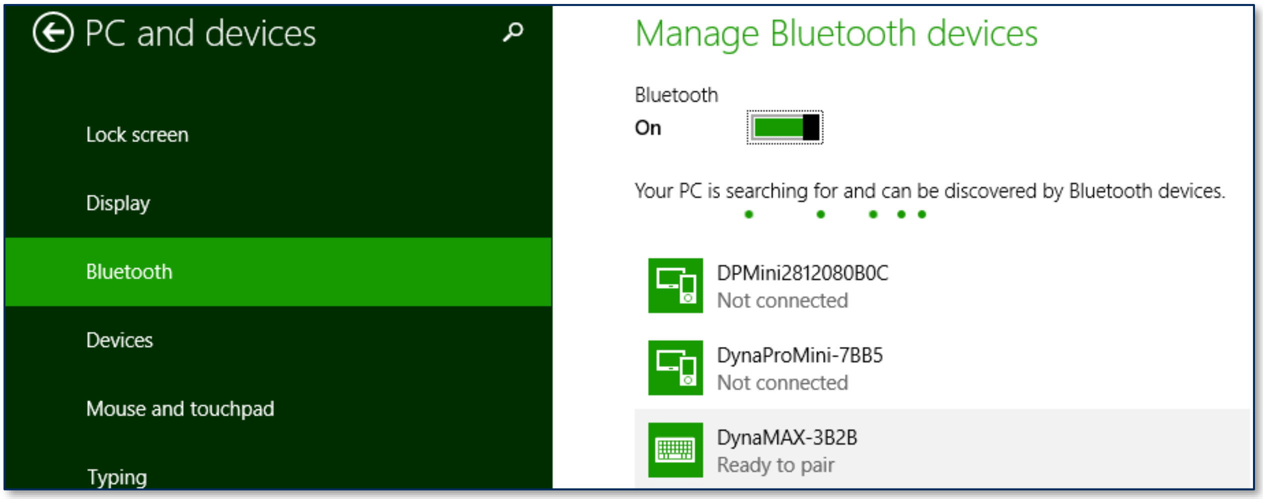


Figure 2-1 - Launch Manage Bluetooth Devices App from Desktop Mode



Manage Bluetooth devices

Bluetooth
On

Your PC is searching for and can be discovered by Bluetooth devices.

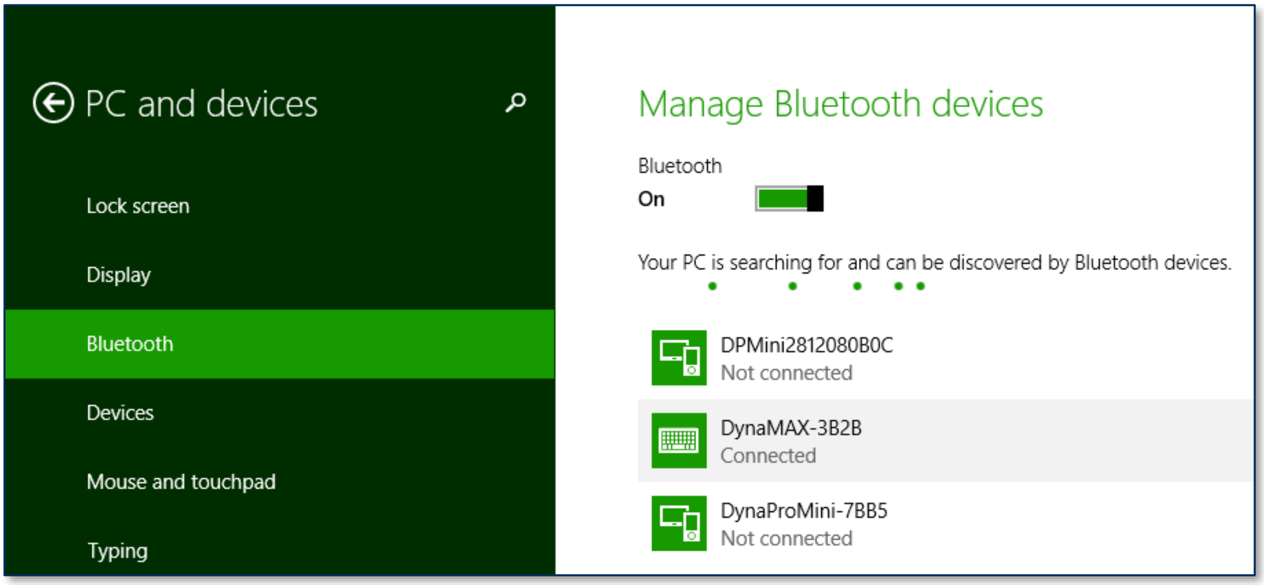
DPMini2812080B0C
Not connected

DynaProMini-7BB5
Not connected

DynaMAX-3B2B
Ready to pair

Figure 2-2 – Windows 8 Manage Bluetooth Devices App

11) Locate the serial number on the label on the bottom of the device. Note the final four digits.
12) Read through the list of pairable devices and locate the device called **DynaProMini-nnnn** or **DF II Go-nnnn** where nnnn is the last four digits of the device's serial number (if the device does not show in the list, power it off then power it back on). Below the device name you should see the text **Ready to pair**.
13) Select the device and press the **Pair** button.
14) Enter default passcode **000000** (or the device's actual password if it has been configured differently), then press the **Next** button. Windows will return you to the **Manage Bluetooth devices** page. After a short period of time, you will see the text **Connected** below the device you are pairing with. After a few seconds, the device will disconnect, which is normal power-saving behavior.



Manage Bluetooth devices

Bluetooth
On

Your PC is searching for and can be discovered by Bluetooth devices.

DPMini2812080B0C
Not connected

DynaMAX-3B2B
Connected

DynaProMini-7BB5
Not connected

15) The device consumes very little power when not transmitting card data, so it is not necessary to power off the device to conserve power. If the device appears as **Not connected** in the Windows list of Bluetooth devices, swiping a card should cause the device to reconnect briefly, transmit the card data, then disconnect.

16) Remember to change the default password. See the DynaPro Mini or DynaFlex Programmer's Reference documents for details.

To unpair from the device:

1) Locate the device in the **Manage Bluetooth devices** window.

Press the **Remove device** button.

3 MagneFlex Powder V2 Resources – DynaFlex devices

MagneFlex Powder V2 API can be hosted as a Windows service (MagTek Powder (PinPAD, DynaFlex) Host service or as an executable (MUSDK.WEBAPI.Host.exe).

3.1 CheckHealth

Returns the operational status of the MagneFlex Powder.
Using Method GET:

```
api/mtppscrahost/CheckHealth
```

Return value: A string array containing API name and status.

```
[
   "MagTek PPSCRA WEB API",
   "OK"
]
```

3.2 DisplayMessageV1

Displays a message on the device.
Using Method POST:

```
api/mtppscrahost/DisplayMessageV1
{
      "DeviceID": "",
      "WaitTime": ,
      "MessageID":
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds to complete the operation. (1 - 255)

MessageID (Integer)	Value representing the message. For DynaFlex family: 0x01 = “AMOUNT” 0x02 = “AMOUNT OK?” 0x03 = “APPROVED” 0x04 = “CALL YOUR BANK” 0x05 = “CANCEL OR ENTER” 0x06 = “CARD ERROR” 0x07 = “DECLINED” 0x08 = “ENTER AMOUNT” 0x09 = reserved, do not use. 0x0A = reserved, do not use. 0x0B = “INSERT CARD” 0x0C = “NOT ACCEPTED” 0x0D = reserved, do not use. 0x0E = “PLEASE WAIT” 0x0F = “PROCESSING ERROR” 0x10 = “REMOVE CARD” 0x11 = “USE CHIP READER” 0x12 = “USE MAGSTRIPE” 0x13 = “TRY AGAIN” 0x14 = “WELCOME” 0x15 = “PRESENT CARD” 0x16 = “PROCESSING” 0x17 = “CARD READ OK - REMOVE CARD” 0x18 = “INSERT OR SWIPE CARD” 0x19 = “PRESENT ONE CARD ONLY” 0x1A = “APPROVED PLEASE SIGN” 0x1B = “AUTHORIZING PLEASE WAIT” 0x1C = “INSERT, SWIPE OR TRY ANOTHER CARD” 0x1D = “PLEASE INSERT CARD” 0x1E = Null prompt (empty screen) 0x1F = reserved, do not use. 0x20 = “SEE PHONE” 0x21 = “PRESENT CARD AGAIN” 0x22 = “INSERT/SWIPE/TRY OTHER CARD” 0x23 = “TAP or SWIPE CARD” 0x24 = “TAP or INSERT CARD” 0x25 = “TAP, INSERT or SWIPE CARD” 0x26 = “TAP CARD” 0x27 = “TIMEOUT” 0x28 = “TRANSACTION TERMINATED” 0x29 = “USE CHIP READER or MAGSTRIPE” 0x2A = “SCAN BARCODE” 0x2B = “BARCODE READ SUCCESSFULLY” 0x2C = “CANCELED” 0x2D = “SWIPE CARD or SCAN BARCODE” 0x2E = “INSERT CARD or SCAN BARCODE” 0x2F = “INSERT, SWIPE or SCAN BARCODE” 0x30 = “TAP CARD or SCAN BARCODE”

	`0x31 = “TAP, SWIPE or SCAN BARCODE”`
	`0x32 = “TAP, INSERT or SCAN BARCODE”`
	`0x33 = “TAP, INSERT, SWIPE or SCAN BARCODE”`
	`0x34 = “TRY ANOTHER INTERFACE”`
	`0x35 = “NFC TAG DETECTED”`
	`0x36 = “ERROR REMOVE CARD”`
	`0x37 = “MIFARE CLASSIC 1K DETECTED”`
	`0x38 = “MIFARE CLASSIC 4K DETECTED”`
	`0x39 = “MIFARE DESFIRE DETECTED”`
	
	For DynaPro family:
	`0x00 = “Blank”`
	`0x01 = “Approved”`
	`0x02 = “Declined”`
	`0x03 = “Cancelled”`
	`0x04 = “Thank You”`
	`0x05 = “PIN Invalid”`
	`0x06 = “Processing”`
	`0x07 = “Please Wait”`
	`0x08 = “Hands Off”`
	`0x09 = “PIN PAD not available”`
	`0x0A = “Call Your Bank”`
	`0x0B = “CARD ERROR”`
	`0x0C = “Not Accepted”`
	`0x0D = “Processing Error”`
	`0x0E = “Use CHIP READER”`
	`0x0F = “Refer to your payment device”`
	`0x80..0x83` = Bitmap in slots 1..4
	`0xFF` = Custom Bitmap Message

Return value:

```
{
   "DeviceID": "",
   "ResultStatus": ,
   "ResultMessage": ""
}
```

Parameter (Type)	Description
DeviceID (String)	Device ID returned from RequestDeviceList.
ResultStatus (Boolean)	Status of the operation. `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.

Example Request/Response:

```
{
      "DeviceID": "",
      "WaitTime": 5,
      "MessageID": 1
}



{
   "DeviceID": "[USB DynaFlex] B62CA53",
   "ResultStatus": true,
   "ResultMessage": "feature_display_message,status_timed_out"
}
```

3.3 EndPINSessionV1

Ends the PIN session on the device. RequestPINWithPANFromCardV1 or RequestPINWithSuppliedPANV1 must be called beforehand to start the PIN session. The host app makes the determination of which status to display on the device.

Using Method POST:

```
api/mtppscrahost/EndPINSessionV1
{
      "DeviceID": "",
      "Option": ""
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.

Option (String)	Status to display on the device.
	`Success = "PIN Entry Successful"`
	`Failed = "PIN Entry Failed"`
	`Cancel = "PIN Entry Canceled"`

Return value:

```
{
   "DeviceID": "",
   "ResultStatus": true,
   "ResultMessage": "",
   "State":
}
```

Parameter (Type)	Description
DeviceID (String)	Device ID returned from RequestDeviceList.

ResultStatus (Boolean)	Status of the operation. `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.
State (Integer)	State of the PIN session. 0 = Unknown 1 = End 2 = Started

Example Request/Response:

```
{
      "DeviceID": "",
      "Option": "Success"
}



{
   "DeviceID": "",
   "ResultStatus": true,
   "ResultMessage": "Session Ended",
   "State": 1
}
```

3.4 ReleaseDeviceV1

Cancels pending requests and closes the connection to the device. This operation is not applicable on a device which was already closed.

The following are requests which can be canceled while waiting for user interaction.

```
RequestPINWithPANFromCardV1
RequestPINWithSuppliedPANV1
RequestSmartCardV1
```

Using Method POST:

```
api/mtppscrahost/ReleaseDeviceV1
{
      "DeviceID": ""
}
```

Parameter (type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.

Return value:

```
{
   "DeviceID": "",
   "ResultStatus": ,
   "ResultMessage": ""
}
```

Parameter (Type)	Description
DeviceID (String)	Device ID returned from RequestDeviceList.
ResultStatus (Boolean)	Status of the operation. `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.

Example Request/Response:

```
{
   "DeviceID": ""
}



{
   "DeviceID": "",
   "ResultStatus": true,
   "ResultMessage": "All devices are released!"
}
```

3.5 RequestDeviceListV1

Returns a string array containing name/value pairs of devices detected on the host based on device type.
Using Method POST:

```
api/mtppscrahost/RequestDeviceListV1
{
   "DeviceType": ""
}
```

Parameter (Type)	Description
DeviceType (String)	Type of device to detect. `MMS` = DynaFlex family `PPSCRA` = DynaPro family Empty string for both MMS and PPSCRA devices

Return value:

```
{"Devices": [{
      "DeviceID": "",
       "DeviceType": "",
       "ConnectionType": "",
       "Address": ""
    },
{
      "DeviceID": "",
       "DeviceType": "",
       "ConnectionType": "",
       "Address": ""
}]
}
```

Parameter (Type)	Description
DeviceID (String)	Device ID. This is used in API resources where DeviceID is required.

DeviceType
(String)
	Type of device detected.
	MMS = DynaFlex family
	PPSCRA = DynaPro family

ConnectionType (String)	Device communication interface to the Host. USB WEBSOCKET SERIAL BLE BLUETOOTH_LE
Address (String)	Address of the device.

Example Request/Response:

```
{
     "DeviceType": "MMS"
}


{"Devices": [{
   "DeviceID": "[USB DynaFlex] B62CA53",
   "DeviceType": "MMS",
   "ConnectionType": "USB",
   "Address":
"\\\\?\\hid#vid_0801&pid_2020#6&1faf27ac&0&0000#{4d1e55b2-f16f-11cf-
88cb-001111000030}"
}]}
```

3.6 RequestManualCardV1

Prompts the user to manually enter card data.
Using Method POST:

```
api/mtppscrahost/RequestManualCardV1
{
   "DeviceID": "",
   "WaitTime": ,
   "Amount":
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds for each prompt on the device screen to be completed. (1 - 255)
Amount (Decimal)	Transaction amount

Return value:

```
{
   "DeviceID": "",
   "ResultStatus": true,
   "ResultMessage": "",
   "DataBlob": "",
   "CardManualOutput":    {
      "CardOperationStatus": ,
      "CardStatus": ,
      "CardType": ,
      "DataType": ,
      "EncryptedMagnePrint": "",
      "EncryptedTrack1": "",
      "EncryptedTrack2": "",
      "EncryptedTrack3": "",
      "EncryptedMagnePrintLength": ,
      "EncryptedMagnePrintStatus": ,
      "EncryptedTrack1Length": ,
      "EncryptedTrack1Status": ,
      "EncryptedTrack2Length": ,
      "EncryptedTrack2Status": ,
      "EncryptedTrack3Length": ,
      "EncryptedTrack3Status": ,
      "MagStripeStatus": ,
      "PANDataLength": ,
      "Track1Length": ,
      "Track1Status": ,
      "Track2Length": ,
      "Track2Status": ,
      "Track3Length": ,
      "Track3Status": ,
      "StatusCode": ,
      "CardData": "",
      "CBCMAC": "",
      "KSN": "",
      "MagnePrintStatus": "",
      "PANData": "",
      "MagTekSerialNumber": "",
      "Track1": "",
      "Track2": "",
      "Track3": ""
   },
   "StatusCode": ,
   "AdditionalOutputData": null
}
```

Parameter (Type)	Description
DeviceID (String)	Device ID from the request.
ResultStatus (Boolean)	Status of the operation. `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.
DataBlob (Hexadecimal string)	Manually entered data encoded in TLV.

CardManualOutput	Container for the manual card data. See RequestCardSwipe operation for details. "CardManualOutput": { "CardOperationStatus": , "CardStatus": , "CardType": , "DataType": , "EncryptedMagnePrint": "", "EncryptedTrack1": "", "EncryptedTrack2": "", "EncryptedTrack3": "", "EncryptedMagnePrintLength": , "EncryptedMagnePrintStatus": , "EncryptedTrack1Length": , "EncryptedTrack1Status": , "EncryptedTrack2Length": , "EncryptedTrack2Status": , "EncryptedTrack3Length": , "EncryptedTrack3Status": , "MagStripeStatus": , "PANDataLength": , "Track1Length": , "Track1Status": , "Track2Length": , "Track2Status": , "Track3Length": , "Track3Status": , "StatusCode": , "CardData": "", "CBCMAC": "", "KSN": "", "MagnePrintStatus": "", "PANData": "", "MagTekSerialNumber": "", "Track1": "", "Track2": "", "Track3": "" }, "StatusCode": , "AdditionalOutputData": null
StatusCode (Integer)	Status code
AdditionalOutputData (name/value pair)	Addition output data

Example Request/Response:

```
{
   "DeviceID": "",
   "WaitTime": 10,
   "Amount": 1.00
}


{
   "DeviceID": "[USB DynaFlex] B62CA53",
   "ResultStatus": true,
   "ResultMessage": "Data received",
   "DataBlob": "0165F9820161...",
   "CardManualOutput":    {
      "CardOperationStatus": 0,
      "CardStatus": 0,
      "CardType": 0,
      "DataType": 0,
      "EncryptedMagnePrint": "",
      "EncryptedTrack1": "",
      "EncryptedTrack2": "",
      "EncryptedTrack3": "",
      "EncryptedMagnePrintLength": 0,
      "EncryptedMagnePrintStatus": 0,
      "EncryptedTrack1Length": 0,
      "EncryptedTrack1Status": 0,
      "EncryptedTrack2Length": 0,
      "EncryptedTrack2Status": 0,
      "EncryptedTrack3Length": 0,
      "EncryptedTrack3Status": 0,
      "MagStripeStatus": 0,
      "PANDataLength": 0,
      "Track1Length": 0,
      "Track1Status": 0,
      "Track2Length": 0,
      "Track2Status": 0,
      "Track3Length": 0,
      "Track3Status": 0,
      "StatusCode": 0,
      "CardData": "",
      "CBCMAC": "",
      "KSN": "",
      "MagnePrintStatus": "",
      "PANData": "",
      "MagTekSerialNumber": "",
      "Track1": "",
      "Track2": "",
      "Track3": ""
   },
   "StatusCode": 0,
   "AdditionalOutputData": null
}
```

3.7 RequestPINWithPANFromCardV1

Returns the encrypted PIN data entered on the device and the PAN is retrieved from the card.

This function prompts the user to present their card and enter a PIN. A card must be presented so that the device can retrieve the PAN, which is used for PIN Format blocks requiring a PAN. To complete the PIN session call EndPINSessionV1 or ReleaseDeviceV1.

Using Method POST:

```
api/mtppscrahost/RequestPINWithPANFromCardV1
{
    "DeviceID": "",
    "WaitTime": ,
    "PINMode": ,
    "MaxPINLength": ,
    "MinPINLength": ,
    "Format": ,
    "UserSupplyPANMethods": ["","",""]
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds to complete the operation. (1 - 255)
PINMode (Integer)	Message to display as a user prompt: For DynaFlex devices: 0x01 = Present Card / Enter PIN 0x04 = Present Card / Enter PIN / Enter PIN Again
MaxPINLength (Integer)	Maximum PIN length. Less than or equal to 12.
MinPINLength (Integer)	Minimum PIN length. Greater than or equal to 4.
Format (Integer)	PIN format: 0 = ISO0 Format 1 = Reserved / Invalid 3 = ISO3 Format 4 = ISO4 Format (AES PIN key required)
UserSupplyPANMethods (Array of string)	List of payment methods. `MSR` = For magnetic stripe cards. `Contact` = For EMV chip cards. `Contactless` = For NFC contactless cards.

Return value:

```
{
   "PAN": {
      "KSN": "",
      "EncryptionType": ,
      "Data": ""
```

```
         },
   "DeviceID": "",
   "ResultStatus": ,
   "ResultMessage": "",
   "PINOutput": null,
   "State": ,
   "PINBlock": "",
   "KSN": "",
   "Format": ,
   "EncryptionType": ,
   "OperationStatus":
}
```

Parameter (Type)	Description
PAN (name/value pairs)	Contains the Primary Account Number key value pairs. KSN = Key serial number the PAN. EncryptionType = Encryption type for the PAN. Data = Encrypted PAN.
DeviceID (String)	Device ID returned from RequestDeviceList.
ResultStatus (Boolean)	Status of the operation. `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.
PINOutput	Container for the PIN data. Reserved for future use.
State (Integer)	PIN session state. `0` = Unknown `1` = Started `2` = End
PINBlock (Hexadecimal string)	PIN block data
KSN (Hexadecimal string)	Key serial number for the PINBlock
Format (Integer)	PIN block format
EncryptionType (Integer)	Encryption type. 128 = Data 129 = PIN
OperationStatus (Integer)	Status of the operation

Example Request/Response:

```
{
    "DeviceID": "",
    "WaitTime": 10,
    "PINMode": 1,
    "MaxPINLength": 8,
    "MinPINLength": 4,
    "Format": 0,
    "UserSupplyPANMethods": ["MSR","Contact","Contactless"]
}


{
   "PAN":    {
      "KSN": "FFFF9876543210200007",
      "EncryptionType": 128,
      "Data": "22CF3A59C33E4FC3EEEA44CEAB54BE74"
   },
   "DeviceID": "[USB DynaFlex] B62CA53",
   "ResultStatus": true,
   "ResultMessage": "PAN and PIN received",
   "PINOutput": null,
   "State": 0,
   "PINBlock": "5D6BF1257BAC537B",
   "KSN": "FFFF9876543211E00004",
   "Format": 0,
   "EncryptionType": 129,
   "OperationStatus": 0
}
```

3.8 RequestPINWithSuppliedPANV1

Returns the encrypted PIN data entered on the device. No card is needed. PAN data is supplied by the host app. To complete the PIN session call EndPINSessionV1 or ReleaseDeviceV1.

This function prompts the user to enter a PIN.
Using Method POST:

```
api/mtppscrahost/RequestPINWithSuppliedPANV1
{
    "DeviceID": "",
    "WaitTime": ,
    "PINMode": ,
    "MaxPINLength": ,
    "MinPINLength": ,
    "Format": ,
    "PAN": "",
    "Options" :
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds to complete the operation. (1 - 255)
PINMode (Integer)	Message to display as a user prompt: 0x00 = Enter PIN 0x02 = PIN Incorrect, Try Again 0x03 = Enter PIN / Enter PIN Again 0x05 = Enter PIN Again
MaxPINLength (Integer)	Maximum PIN length. Less than or equal to 12.
MinPINLength (Integer)	Minimum PIN length. Greater than or equal to 4.
Format (Integer)	PIN format: 0 = ISO0 Format 1 = ISO1 Format 3 = ISO3 Format 4 = ISO4 Format (AES PIN key required)
PAN (Hexadecimal String)	12 digit PAN used for PIN block formats requiring a PAN.
Option (Integer)	Reserved for future use.

Return value:

```
{
   "DeviceID": "",
   "ResultStatus": ,
   "ResultMessage": "",
   "PINOutput": null,
   "State": ,
   "PINBlock": "",
   "KSN": "",
   "Format": ,
   "EncryptionType": ,
   "OperationStatus":
}
```

Parameter (Type)	Description
DeviceID (String)	Device ID returned from RequestDeviceList.

ResultStatus (Boolean)	Status of the operation. `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.
PINOutput	Container for the PIN data. Reserved for future use.
State (Integer)	PIN session state. `0` = Unknown `1` = Started `2` = End
PINBlock (Hexadecimal string)	PIN block data
KSN (Hexadecimal string)	Key serial number
Format (Integer)	PIN block format. 0 = ISO0 Format, No verify PIN 1 = ISO3 Format, No verify PIN 2 = ISO0 Format, Verify PIN 3 = ISO3 Format, Verify PIN
EncryptionType (Integer)	Encryption type. 128 = Data 129 = PIN
OperationStatus (Integer)	Status of the operation

Example Request/Response:

```
{
    "DeviceID": "",
    "WaitTime": 10,
    "PINMode": 0,
    "MaxPINLength": 8,
    "MinPINLength": 4,
    "Format": 0,
    "PAN": "923456789012",
    "Options": null
}



{
   "DeviceID": "[USB DynaFlex] B62CA53",
   "ResultStatus": true,
   "ResultMessage": "PIN received",
   "PINOutput": null,
   "State": 1,
   "PINBlock": "FC7817FD5686501B",
   "KSN": "FFFF9876543211E00001",
   "Format": 0,
   "EncryptionType": 129,
   "OperationStatus": 0
}
```

3.9 RequestSendAcquirerResponseV1

Sends the ARPC to the device. Applicable only after a RequestSmartCardV1 with QwickChipMode set to false.

Using Method POST:

```
api/mtppscrahost/RequestSendAcquirerResponseV1
{
      "DeviceID": "",
      "WaitTime": ,
      "ApprovalStatus": ,
      "IssuerAuthenticationData": "",
      "IssuerScriptTemplate1": "",
      "IssuerScriptTemplate2": ""
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds the device will wait for the action to be completed. (1 - 255)
ApprovalStatus (Integer)	Status from acquirer/issuer. This field represents the data portion of the EMV Tag 8A. Example: 0 = Approve 1 = Decline
IssuerAuthenticationData (String)	Issuer response to the transaction request in hexadecimal format. This field is for the data portion of the EVM Tag 91. Use 00 if not provided.
IssuerScriptTemplate1 (String)	Issuer Script to send to ICC in hexadecimal format. This field is for the data portion of the EVM Tag 71. Use 00 if not provided.
IssuerScriptTemplate2 (String)	Issuer Script to send to ICC in hexadecimal format. This field is for the data portion of the EVM Tag 72. Use 00 if not provided.

Return value:

```
{ "DeviceID": "",
  "TransactionOutput": {
      "TransactionType": ,
      "TransactionStatus": ,
      "OperationStatus": ,
      "DataType": ,
      "ApplicationIdentifier": ,
      "CardBrand": ,
      "ARQCData": ,
      "BatchData": ,
      "RawARQCData": ,
      "RawBatchData": ,
      "KSN": ,
      "DeviceSerialNumber": ,
      "EncryptionType": ,
      "NumberOfPaddedBytes": ,
      "NumberOfPaddedBytesForBatch": ,
      "EMVSREDData": ,
      "EMVSREDDataForBatch": ,
      "MerchantData": ,
      "FallbackIndicator": ,
      "MaskedICCTrack2": ,
      "ServiceCode": ,
      "CardHolderName": ,
      "CardType": ,
      "ApprovalStatus":
}}
```

Parameter (Type)	Description
DeviceID	URI of the device. See **DeviceID URI** for details.
TransactionType (Integer)	Transaction type 0 = ARQC 1 = Batch

TransactionStatus (Integer)	Transaction status 0x00 = Accept 0x01 = Decline 0x02 = Error 0x10 = Cancelled by Host 0x11 = Confirm Amount No 0x12 = Confirm Amount Timeout 0x13 = Confirm Amount Cancel 0x14 = MSR Select Credit 0x15 = MSR Select Debit 0x16 = MSR Select Credit/Debit timeout 0x17 = MSR Select Credit/Debit cancel 0x18 = Signature Capture Cancelled by Host 0x19 = Signature Capture Timeout 0x1A = Signature Capture Cancelled by Cardholder 0x1B = PIN entry Cancelled by Host 0x1C = PIN entry timeout 0x1D = PIN entry Cancelled by Cardholder 0x1E = Manual Selection Cancelled by Host 0x1F = Manual Selection timeout 0x20 = Manual Selection Cancelled by Cardholder 0x21 = Waiting For Card Cancelled by Host 0x22 = Waiting For Card timeout 0x23 = Waiting For Card Cancelled by Cardholder 0x24 = Waiting For Card ICC Seated 0x25 = Waiting For Card MSR Swiped 0xFF = Unknown
OperationStatus (Integer)	Operation status 0 = OK / Done 1 = Cardholder Cancel 2 = Timeout 3 = Host Cancel
DataType (Integer)	Data type. This is the report ID from the raw HID report descriptor.
ApplicationIdentifier (Hexadecimal string)	EMV Application Identifier
CardBrand (String)	Card brand
ARQCData (Hexadecimal string)	Authorization Request Cryptogram for the transaction. This should be coordinated with the transaction processor to request approval for the transaction.
BatchData (Hexadecimal string)	Batch data for the transaction. This contains the final result of the transaction.
RawARQCData (Base64 string)	Raw Authorization Request Cryptogram for the transaction

RawBatchData (Base64 string)	Raw Batch data for the transaction
KSN (Hexadecimal string)	Key serial number
DeviceSerialNumber (Hexadecimal string)	MagTek device serial number
EncryptionType (Hexadecimal)	Encryption type 80 = DUKPT Key Data variant 81 = DUKPT Key PIN variant
NumberOfPaddedBytes (Integer)	Number of padded bytes to the end of the decrypted EMV SRED data to make a multiple of 8 bytes.
NumberOfPaddedBytesForBatch (Integer)	Number of padded bytes to the end of the decrypted EMV SRED Batch data to make a multiple of 8 bytes.
EMVSREDData (Hexadecimal string)	EMV SRED data. This is data portion of the TLV tag DFDF59 from ARQCData.
EMVSREDDataForBatch (Hexadecimal string)	EMV SRED Batch data
FallbackIndicator (Hexadecimal)	Fallback indicator 00 = No Fallback 01 = Technical Fallback 81 = MSR Fallback
MaskedICCTrack2 (String)	Masked magnetic stripe data for track 2
ServiceCode (Integer)	Service code
CardHolderName (Hexadecimal string)	Card holder name
CardType (Integer)	Card type. 0 = Other 1 = Financial 2 = AAMVA 3 = Manual 4 = Unknown 5 = ICC 6 = Contactless ICC - EMV 7 = Financial MSR + ICC 8 = Contactless ICC - MSD
ApprovalStatus (Integer)	Approval status

Example Request/Response:

```
{
      "DeviceID": "",
      "WaitTime": 5,
      "ApprovalStatus": 0,
      "IssuerAuthenticationData": "00",
      "IssuerScriptTemplate1": "00",
      "IssuerScriptTemplate2": "00"
}



{
   "DeviceID": "[USB DynaFlex] B62CA53",
   "TransactionOutput":    {
      "TransactionType": 1,
      "TransactionStatus": 0,
      "OperationStatus": 0,
      "DataType": 2,
      "ApplicationIdentifier": "A0000000041010",
      "CardBrand": "MasterCard",
      "ARQCData": "01E9F98201E5...",
      "BatchData": "01E7F98201E3...",
      "RawARQCData": "Aen5ggH...",
      "RawBatchData": "Aef5ggHj...",
      "KSN": "FFFF9876543210200004",
      "DeviceSerialNumber": "42363243413533",
      "EncryptionType": "80",
      "NumberOfPaddedBytes": 6,
      "NumberOfPaddedBytesForBatch": 7,
      "EMVSREDData": "16AF008CDE78...",
      "EMVSREDDataForBatch": "322A3399A9CF...",
      "MerchantData": "9C01005F2503...",
      "FallbackIndicator": "00",
      "MaskedICCTrack2": "3B3534343336...",
      "ServiceCode": null,
      "CardHolderName": null,
      "CardType": "06",
      "ApprovalStatus": 0
   }
}
```

3.10 RequestSendCommandV1

Sends a command to the device and returns the raw response from the device.
Using Method POST:

```
api/mtppscra/RequestSendCommandV1
{
   "DeviceID": "",
   "Data": "",
   "WaitTime":
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
Data (Hexadecimal string)	Command to send to the device.
WaitTime (Integer)	Time in seconds to complete the operation. (1 - 255)

Return value:

```
{
   "DeviceID": "",
   "Data": "",
   "ResultStatus": ,
   "ResultMessage": ""
}
```

Parameter (Type)	Description
DeviceID (String)	Device ID returned from RequestDeviceList.
Data (Hexadecimal string)	Response of the command sent.
ResultStatus (Boolean)	Status of the operation. `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.

Example Request/Response:

```
{
   "DeviceID": "",
   "Data": "AA0081040101DF018405DF01810101",
   "WaitTime": 5
}



{
   "DeviceID": "",
   "Data": "AA0081048201DF01820400000008405DF01810101",
   "ResultStatus": true,
   "ResultMessage": "Completed."
}
```

3.11 RequestSignatureV1

Prompts the user to sign on the device's screen.
Using Method POST:

```
api/mtppscrahost/RequestSignatureV1
{
      "DeviceID": "",
      "WaitTime":
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds the device will wait for the action to be completed. (1 - 255)

Return value:

```
{
   "DeviceID": "",
   "ResultStatus": ,
   "ResultMessage": "",
   "SignatureOutput":
      {
         "SignatureOperationStatus": ,
         "SignatureData": ""
      }
   "AdditionalOutputData":
}
```

Parameter (Type)	Description
DeviceID (String)	Device ID returned from RequestDeviceList.
ResultStatus (Boolean)	Status of the operation. `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.

SignatureOutput (name/value pairs)	Container for the Signature data.
	{
	"SignatureOperationStatus": ,
	"SignatureData": ""
	}

SignatureOperationStatus (Integer)	Signature operation status. 0 = success. Otherwise, an error

SignatureData (Base64 string)	Signature data encoded as Base64.
AdditionaOutputData (name/value strings)	Additional output data.

Example Request/Response:

```
{
      "DeviceID": "",
      "WaitTime": 10
}

{
   "DeviceID": "[USB DynaFlex] B62CA53",
   "ResultStatus": true,
   "ResultMessage": "signature captured",
   "SignatureOutput":    {
      "SignatureOperationStatus": 0,
      "SignatureData":
"DWoOXA9SEEcRPBIuEyIUIBYfGCAaIR0jICYjKSYtKTEsNC83MTozPDU+Nj83QDhAOUA7Q
D0+PzxBOUM1RTFGLEgoSiRLIEwdTRlOF08UUBJREVIQUw9UD1YQWBFaFF0XXxxiIWQmZyt
pMWs2bTpvPnBCcUVyR///iRqPGpIalxqcGqIaqBqvGrUavBrCGcgZzBnQGf//oSGiJqMqp
C6lM6Y3pzuoQv//"
   },
   "AdditionalOutputData": null
}
```

3.12 RequestSmartCardV1

Begins an EMV transaction. To cancel the transaction session before a payment method is presented, call ReleaseDeviceV1.

Using Method POST:

```
api/mtppscrahost/RequestSmartCardV1
{
   "DeviceID": "",
   "WaitTime": ,
   "PaymentMethods": ["","","",""],
   "QwickChipMode": ,
   "TransactionType": ,
   "Amount": ,
   "Cashback": ,
   "AdditionalTags": "",
   "SuppressThankYouMessage": ,
   "AdditionalRequestData": null
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds to complete the operation. (1 - 255)
PaymentMethods (Array of string)	List of payment methods. `MSR` = For magnetic stripe cards. `Contact` = For EMV chip cards. `Contactless` = For NFC contactless cards.

QwickChipMode
(Boolean)
	In QwickChip mode, the device does not prompt for an amount. Device sends an ARQC request to the host. Device automatically populates the ARPC response data with EMV Tag 8A set to “Z3”. Card holder is prompted to remove the card. Transaction result is later determined by the processor and not by the card.
	false = Do not enable QwickChip mode.
	true = Enable QwickChip mode. Default.

TransactionType (Integer)	Type of transaction to be used: 0x00 = Purchase 0x01 = Cash Advance 0x02 or 0x09 = Cashback 0x04 = Goods (Purchase) 0x08 = Services (Purchase) 0x10 = International Goods (Purchase) 0x12 = Cash Manual 0x20 = Refund (Chip Card Contactless Only) 0x40 = International Cash Advance or Cash Back 0x50 = Payment (Chip Card Contact Only) 0x80 = Domestic Cash Advance or Cash Back
Amount (Decimal)	The amount to be used and authorized, EMV Tag 9F02. Format in decimal.
CashBack (Decimal)	Amount of cash back to be used, EMV Tag 9F02. Format in decimal.

AdditionalTags (String)	Additional transaction TLV data objects to include in the start transaction request and ARQC. Example: Single tag - "5F2A020840" Multiple tags - "5F2A0208405F360102" No additional tags - "" Supported tags: • 9F7C Merchant Custom Data • 5F2A Transaction Currency Code • 5F36 Transaction Currency Exponent • 9F53 Transaction Category Code • 9F15 Merchant Category Code • 9F16 Merchant ID
SuppressThankYouMessage (Boolean)	By default, devices with a display signals the end of a transaction by briefly showing “THANK YOU,” then “WELCOME” idle screen. false = Do not suppress thank you message (default). true = Suppress thank you message.
AdditionalRequestData (name/value pairs)	Additional name/value pairs of data to be forwarded in the transaction.

Return value:

```
{
   "DeviceID": "",
   "TransactionOutput":    {
      "TransactionType": ,
      "TransactionStatus": ,
      "OperationStatus": ,
      "DataType": ,
      "RawUserSelectData": ,
      "ApplicationIdentifier": ,
      "CardBrand": "",
      "ARQCData": "",
      "BatchData": ,
      "RawARQCData": "",
      "RawBatchData": ,
      "KSN": "",
      "DeviceSerialNumber": "",
      "EncryptionType": "",
      "NumberOfPaddedBytes": ,
      "NumberOfPaddedBytesForBatch": ,
      "EMVSREDData": "",
      "EMVSREDDataForBatch": ,
      "MerchantData": ,
      "FallbackIndicator": "",
      "MaskedICCTrack2": "",
      "ServiceCode": ,
      "CardHolderName": ,
      "CardType": "",
      "ApprovalStatus":
   },
   "ResultStatus": ,
   "ResultMessage": ""
}
```

Parameter (Type)	Description
DeviceID	URI of the device. See **DeviceID URI** for details.
TransactionType (Integer)	Transaction type 0 = ARQC 1 = Batch
TransactionStatus (Integer)	Transaction status 0x00 = Accept 0x01 = Decline 0x02 = Error 0x10 = Cancelled by Host 0x11 = Confirm Amount No 0x12 = Confirm Amount Timeout 0x13 = Confirm Amount Cancel 0x14 = MSR Select Credit 0x15 = MSR Select Debit 0x16 = MSR Select Credit/Debit timeout 0x17 = MSR Select Credit/Debit cancel 0x18 = Signature Capture Cancelled by Host 0x19 = Signature Capture Timeout 0x1A = Signature Capture Cancelled by Cardholder 0x1B = PIN entry Cancelled by Host 0x1C = PIN entry timeout 0x1D = PIN entry Cancelled by Cardholder 0x1E = Manual Selection Cancelled by Host 0x1F = Manual Selection timeout 0x20 = Manual Selection Cancelled by Cardholder 0x21 = Waiting For Card Cancelled by Host 0x22 = Waiting For Card timeout 0x23 = Waiting For Card Cancelled by Cardholder 0x24 = Waiting For Card ICC Seated 0x25 = Waiting For Card MSR Swiped 0xFF = Unknown
OperationStatus (Integer)	Operation status 0 = OK / Done 1 = Cardholder Cancel 2 = Timeout 3 = Host Cancel
DataType (Integer)	Data type. This is the report ID from the raw HID report descriptor.

<table>
<tr><td>RawUserSelectData
(Base64 String)</td><td>A set of one or more ASCII strings the host should show to the user. When present in the response, the host should display the list of strings to the user, and then send the selection to the device using operation RequestUserSelectionV1.

The raw data is structured as below.
<table>
<tr><th>Offset</th><th>Description</th></tr>
<tr><td>0</td><td>Append or Clear Display.
• 0x00 = Append, display the messages on the next line.
• 0x01 = Clear, clear the display first and display the requested message on the first line.</td></tr>
<tr><td>1</td><td>Time in seconds the device will wait for response. Device selects the first of the list if response times out.</td></tr>
<tr><td>nn</td><td>Message strings in ASCII Hex. Each string is terminated by any one of the following characters.
• Null (0x00)
• LF (0x0A)</td></tr>
</table>

Example conversion and parsing:
RawUserSelectData Base64
<code>“AApQbGVhc2UgU2VsZWN0AERlYml0IE1hc3RlcmNhcmQA</code>
<code>RGViaXQA”</code>

In ASCII Hex
<code>“000A506C6561736520536 56C656374004465626974 20</code>

Parsed
<code>00 0A</code>
<code>506C656173652053656C656374 00</code>
<code>4465626974204D617374657263617264 00</code>
<code>4465626974 00</code>

Displayed
<code>Please Select</code>
<code>Debit Mastercard</code>
<code>Debit</code></td></tr>
<tr><td>ApplicationIdentifier
(Hexadecimal string)</td><td>EMV Application Identifier</td></tr>
<tr><td>CardBrand
(String)</td><td>Card brand</td></tr>
<tr><td>ARQCData
(Hexadecimal string)</td><td>Authorization Request Cryptogram for the transaction. This should be coordinated with the transaction processor to request approval for the transaction.</td></tr>
</table>

BatchData (Hexadecimal string)	Batch data for the transaction. This contains the final result of the transaction.
RawARQCData (Base64 string)	Raw Authorization Request Cryptogram for the transaction
RawBatchData (Base64 string)	Raw Batch data for the transaction
KSN (Hexadecimal string)	Key serial number
DeviceSerialNumber (Hexadecimal string)	MagTek device serial number
EncryptionType (Hexadecimal)	Encryption type 80 = DUKPT Key Data variant 81 = DUKPT Key PIN variant
NumberOfPaddedBytes (Integer)	Number of padded bytes to the end of the decrypted EMV SRED data to make a multiple of 8 bytes.
NumberOfPaddedBytesForBatch (Integer)	Number of padded bytes to the end of the decrypted EMV SRED Batch data to make a multiple of 8 bytes.
EMVSREDData (Hexadecimal string)	EMV SRED data. This is data portion of the TLV tag DFDF59 from ARQCData.
EMVSREDDataForBatch (Hexadecimal string)	EMV SRED Batch data
FallbackIndicator (Hexadecimal)	Fallback indicator 00 = No Fallback 01 = Technical Fallback 81 = MSR Fallback
MaskedICCTrack2 (String)	Masked magnetic stripe data for track 2
ServiceCode (Integer)	Service code
CardHolderName (Hexadecimal string)	Card holder name
CardType (Integer)	Card type. 0 = Other 1 = Financial 2 = AAMVA 3 = Manual 4 = Unknown 5 = ICC 6 = Contactless ICC - EMV 7 = Financial MSR + ICC 8 = Contactless ICC - MSD
ApprovalStatus (Integer)	Approval status

ResultStatus (Boolean)	Status of the operation. `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.

Example Request/Response:

```
{
   "DeviceID": "",
   "WaitTime": 10,
   "PaymentMethods": ["MSR","Contact","Contactless"],
   "QwickChipMode": true,
   "TransactionType": 0,
   "Amount": 1.00,
   "CashBack": 0.00,
   "AdditionalTags": "5F2A020840"
   "AdditionalRequestData": null
}

{
   "DeviceID": "[USB DynaFlex] B62CA53",
   "TransactionOutput":    {
      "TransactionType": 1,
      "TransactionStatus": 0,
      "OperationStatus": 0,
      "DataType": 2,
      "ApplicationIdentifier": "A0000000041010",
      "CardBrand": "MasterCard",
      "ARQCData": "01E9F98201E5...",
      "BatchData": "01E7F98201E3...",
      "RawARQCData": "Aen5ggHl...",
      "RawBatchData": "Aef5ggHj3...",
      "KSN": "FFFF9876543210200003",
      "DeviceSerialNumber": "42363243413533",
      "EncryptionType": "80",
      "NumberOfPaddedBytes": 6,
      "NumberOfPaddedBytesForBatch": 7,
      "EMVSREDData": "1D370E75C2D7...",
      "EMVSREDDataForBatch": "8B37E0E13E4A...",
      "MerchantData": "9C01005F2503...",
      "FallbackIndicator": "00",
      "MaskedICCTrack2": "3B3534343336...",
      "ServiceCode": null,
      "CardHolderName": null,
      "CardType": "06",
      "ApprovalStatus": 0
   },
   "ResultStatus": true,
   "ResultMessage": "Batch data received"
}
```

3.13 RequestUserSelectionV1

Sends the result of user selection to the device that have no display. The host should use its user interface to request a selection from the cardholder based on the selectable items returned in key RequestSmartCardV1.RawUserSelectData.

Using Method POST:

```
api/mtppscrahost/RequestUserSelectionV1
{
    "DeviceID": "",
    "Status": ,
    "Selection":
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
Status (Integer)	The status of User Selection: 0 = User Selection Request completed, see Selection Result 1 = User Selection Request aborted, cancelled by user 2 = User Selection Request aborted, timeout
Selection (Integer)	The menu item selected by the user. This is a single byte zero based binary value.

Return value:

```
{
   "DeviceID": "",
   "TransactionOutput":    {
      "TransactionType": ,
      "TransactionStatus": ,
      "OperationStatus": ,
      "DataType": ,
      "RawUserSelectData": ,
      "ApplicationIdentifier": "",
      "CardBrand": "",
      "ARQCData": "",
      "BatchData": "",
      "RawARQCData": "",
      "RawBatchData": ",
      "KSN": "",
      "DeviceSerialNumber": "",
      "EncryptionType": "",
      "NumberOfPaddedBytes": ,
      "NumberOfPaddedBytesForBatch": ,
      "EMVSREDData": "",
      "EMVSREDDataForBatch": "",
      "MerchantData": "",
      "FallbackIndicator": "",
      "MaskedICCTrack2": "",
      "CardType": "",
      "ApprovalStatus":
   },
   "ResultStatus":,
   "ResultMessage": ""
}
```

Parameter (Type)	Description
DeviceID	URI of the device. See **DeviceID URI** for details.
TransactionType (Integer)	Transaction type 0 = ARQC 1 = Batch
TransactionStatus (Integer)	Transaction status 0x00 = Accept 0x01 = Decline 0x02 = Error 0x10 = Cancelled by Host 0x11 = Confirm Amount No 0x12 = Confirm Amount Timeout 0x13 = Confirm Amount Cancel 0x14 = MSR Select Credit 0x15 = MSR Select Debit 0x16 = MSR Select Credit/Debit timeout 0x17 = MSR Select Credit/Debit cancel 0x18 = Signature Capture Cancelled by Host 0x19 = Signature Capture Timeout 0x1A = Signature Capture Cancelled by Cardholder 0x1B = PIN entry Cancelled by Host 0x1C = PIN entry timeout 0x1D = PIN entry Cancelled by Cardholder 0x1E = Manual Selection Cancelled by Host 0x1F = Manual Selection timeout 0x20 = Manual Selection Cancelled by Cardholder 0x21 = Waiting For Card Cancelled by Host 0x22 = Waiting For Card timeout 0x23 = Waiting For Card Cancelled by Cardholder 0x24 = Waiting For Card ICC Seated 0x25 = Waiting For Card MSR Swiped 0xFF = Unknown
OperationStatus (Integer)	Operation status 0 = OK / Done 1 = Cardholder Cancel 2 = Timeout 3 = Host Cancel

<table>
<tr><td>DataType
(Integer)</td><td>Data type. This is the report ID from the raw HID report descriptor.</td></tr>
<tr><td>RawUserSelectData
(Base64 String)</td><td>A set of one or more ASCII strings the host should show to the user. When present in the response, the host should display the list of strings to the user, and then send the selection to the device using operation RequestUserSelectionV1.

Each string is terminated by any one of the following characters.
• Null (0x00)
• LF (0x0A)

The raw data is composed as below.
<table>
<tr><th>Offset</th><th>Description</th></tr>
<tr><td>0</td><td>Append or Clear Display.
• 0x00 = Append, display the messages on the next line.
• 0x01 = Clear, clear the display first and display the requested message on the first line.</td></tr>
<tr><td>1</td><td>Time in seconds the device will wait for response.</td></tr>
<tr><td>nn</td><td>Message in ASCII Hex.</td></tr>
</table>

Example conversion and parsing:
RawUserSelectData Base64
“AApQbGVhc2UgU2VsZWN0AERlYml0IE1hc3RlcmNhcmQARGViaXQA”

In ASCII Hex
“000A506C6561736520053656C6563740044656269742042D617374657263617264004465626974 00”

Parsed
00 0A
506C656173652053656C656374 00
4465626974204D617374657263617264 00
4465626974 00

Displayed
Please Select
Debit Mastercard
Debit</td></tr>
<tr><td>ApplicationIdentifier
(Hexadecimal string)</td><td>EMV Application Identifier</td></tr>
<tr><td>CardBrand
(String)</td><td>Card brand</td></tr>
</table>

ARQCData (Hexadecimal string)	Authorization Request Cryptogram for the transaction. This should be coordinated with the transaction processor to request approval for the transaction.
BatchData (Hexadecimal string)	Batch data for the transaction. This contains the final result of the transaction.
RawARQCData (Base64 string)	Raw Authorization Request Cryptogram for the transaction
RawBatchData (Base64 string)	Raw Batch data for the transaction
KSN (Hexadecimal string)	Key serial number
DeviceSerialNumber (Hexadecimal string)	MagTek device serial number
EncryptionType (Hexadecimal)	Encryption type 80 = DUKPT Key Data variant 81 = DUKPT Key PIN variant
NumberOfPaddedBytes (Integer)	Number of padded bytes to the end of the decrypted EMV SRED data to make a multiple of 8 bytes.
NumberOfPaddedBytesForBatch (Integer)	Number of padded bytes to the end of the decrypted EMV SRED Batch data to make a multiple of 8 bytes.
EMVSREDData (Hexadecimal string)	EMV SRED data. This is data portion of the TLV tag DFDF59 from ARQCData.
EMVSREDDataForBatch (Hexadecimal string)	EMV SRED Batch data
FallbackIndicator (Hexadecimal)	Fallback indicator 00 = No Fallback 01 = Technical Fallback 81 = MSR Fallback
MaskedICCTrack2 (String)	Masked magnetic stripe data for track 2
ServiceCode (Integer)	Service code
CardHolderName (Hexadecimal string)	Card holder name

CardType (Integer)	Card type. 0 = Other 1 = Financial 2 = AAMVA 3 = Manual 4 = Unknown 5 = ICC 6 = Contactless ICC - EMV 7 = Financial MSR + ICC 8 = Contactless ICC - MSD
ApprovalStatus (Integer)	Approval status
ResultStatus (Boolean)	Status of the operation. `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.

Example Request/Response:

```
{
      "DeviceID": "",
      "Status" : 0,
      "Selection" : 0
}

{
   "DeviceID": "[USB DynaFlex] B62CA53",
   "TransactionOutput":    {
      "TransactionType": 1,
      "TransactionStatus": 0,
      "OperationStatus": 0,
      "DataType": 2,
      "ApplicationIdentifier": "A0000000041010",
      "CardBrand": "MasterCard",
      "ARQCData": "01E9F98201E5...",
      "BatchData": "01E7F98201E3...",
      "RawARQCData": "Aen5ggHl...",
      "RawBatchData": "Aef5ggHj3...",
      "KSN": "FFFF9876543210200003",
      "DeviceSerialNumber": "42363243413533",
      "EncryptionType": "80",
      "NumberOfPaddedBytes": 6,
      "NumberOfPaddedBytesForBatch": 7,
      "EMVSREDData": "1D370E75C2D7...",
      "EMVSREDDataForBatch": "8B37E0E13E4A...",
      "MerchantData": "9C01005F2503...",
      "FallbackIndicator": "00",
      "MaskedICCTrack2": "3B3534343336...",
      "ServiceCode": null,
      "CardHolderName": null,
      "CardType": "06",
      "ApprovalStatus": 0
   },
   "ResultStatus": true,
   "ResultMessage": "Batch data received"
}
```

3.14 ScanBarCodeV1

Starts the barcode reader.
Using Method POST:

```
api/mtppscrahost/ScanBarCodeV1
{
    "DeviceID": "",
    "WaitTime": ,
    "Encryption":
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds to complete the operation. 0 = Wait until a barcode is read. 1 to 255 seconds
Encryption (Integer)	Whether to return the data encrypted. 0 = response data will not be encrypted 1 = response data will be encrypted

Return value:

```
{
   "DeviceID": "",
   "ResultStatus": ,
   "ResultMessage": "",
   "Data": "",
   "Encrypted": ,
   "EncryptionType": ,
   "KSN": ""
}
```

Parameter (Type)	Description
DeviceID (String)	Device ID from the request.

ResultStatus (Boolean)	Status of the operation. `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.
Data (Hexadecimal string)	Data payload
Encrypted (Boolean)	`false` = data is not encrypted `true` = data is encrypted
EncryptionType (Integer)	Encryption type in decimal format. 0 = not encrypted 128 = Data variant 129 = PIN variant The possible values are an ORed bitmask using the following elements: 1x00 xxxx = TDES DUKPT key 1x01 xxxx = AES128 DUKPT key 1x10 xxxx = AES256 DUKPT key xxxx 0000 = Data Encrypt/Decrypt Variant xxxx 0001 = PIN Variant xxxx 0010 = MAC Variant xxxx 0011 = Data, Encrypt Variant xxxx 0100 = MAC Verify Variant xxxx 0111 = AES PIN Encrypt xxxx 1000 = AES MAC Generate xxxx 1001 = AES MAC Verify xxxx 1010 = AES MAC Generate/Verify xxxx 1011 = AES Data Encrypt xxxx 1100 = AES Data Decrypt xxxx 1101 = AES Data Encrypt/Decrypt
KSN (Hexadecimal string)	Key Serial Number used for the transaction.

Example Request/Response:

```
{
    "DeviceID": "",
    "WaitTime": 10,
    "Encryption": 0
}

{
   "DeviceID": "[USB DynaFlex] B62CA53",
   "ResultStatus": true,
   "ResultMessage": "barcode data received",
   "Data":
"5C303030303236687474470733A2F2F7777772E6D616774656B2E636F6D",
   "Encrypted": false,
   "EncryptionType": 0,
   "KSN": ""
}
```

3.15 ShowBarCodeV1

Shows a barcode on the device’s display.
Using Method POST:

```
api/mtppscrahost/ShowBarCodeV1
{
    "DeviceID": "",
    "WaitTime" : ,
    "Message": "",
    "Prompt": "",
    "BlockColor": "",
    "BackgroundColor": ""
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds to complete the operation. 0 = Indefinite 1 to 255 seconds
Message (Hexadecimal string)	Data to encode into a barcode.
Prompt (String)	Text to display below the QR code. In Landscape orientation, the limit is approximately 30 characters. In Portrait orientation, the limit is approximately 22 characters.
BlockColor (Hexadecimal string)	Block color. Use RRGGBB format. “000000” = Black (default)
BackgroundColor (Hexadecimal string)	Background color. Use RRGGBB format. “FFFFFF” = White (default)

Return value:

```
{
   "DeviceID": "",
   "ResultStatus": ,
   "ResultMessage": ""
}
```

Parameter (Type)	Description
DeviceID (String)	Device ID from the request.

ResultStatus (Boolean)	Status of the operation. `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.

Example Request/Response:

```
{
   "DeviceID": "",
   "WaitTime": 5,
   "MessageID": 1
}



{
   "DeviceID": "[USB DynaFlex] B62CA53",
   "ResultStatus": true,
   "ResultMessage": "feature_display_message,status_timed_out"
}
```

4 MagneFlex Powder V2 Resources – DynaPro devices

MagneFlex Powder V2 can be hosted as a Windows service (MagTek Powder (PinPAD, DynaFlex) Host service or as an executable (MUSDK.WEBAPI.Host.exe).

4.1 CheckHealth

Returns the operational status of the MagneFlex Powder.
Using Method GET:

```
api/mtppscrahost/CheckHealth
```

Return value: A string array containing API name and status.

```
[
   "MagTek PPSCRA WEB API",
   "OK"
]
```

4.2 ReleaseDevice

Closes the connection to the device.
Using Method POST:

```
api/mtppscrahost/ReleaseDevice
```

Return value:
None

4.3 ReleaseDeviceEx

Closes the connection to the device and displays an idle bitmap message. This operation is not applicable on a device which was already closed.
Using Method POST:

```
api/mtppscrahost/ReleaseDeviceEx
{
      "DeviceID": "",
      "EndSessionDisplayMessage":
}
```

Parameter (type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
EndSessionDisplayMessage (Integer)	Display to show on the device. 0 = “Welcome” (default) 1 = Bitmap Slot 1 2 = Bitmap Slot 2 3 = Bitmap Slot 3 4 = Bitmap Slot 4

Return value:

None

Example Request:

```
{
      "DeviceID": "",
      "EndSessionDisplayMessage": 0
}
```

4.4 RequestCardSwipe

Prompts for a magnetic card swipe.
Using Method POST:

```
api/mtppscrahost/RequestCardSwipe
{
      "DeviceID": "",
      "WaitTime": ,
      "DisplayMessage": ,
      "Tones": ,
      "FieldSeparator": "",
      "CloseDevice": ,
      "EndSession": ,
      "EndSessionDisplayMessage":
}
```

Parameter (type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds the device will wait for the action to be completed. (1 - 255)
DisplayMessage (Integer)	Message to prompt the user with: 0 = Swipe Card / Idle alternating 1 = Swipe Card 2 = Please Swipe Card 3 = Please Swipe Card Again 4 = Chip Error, Use Mag Stripe
Tones (Integer)	Tones to use: 0 = No sound 1 = One beep 2 = Two beeps
FieldSeparator (String)	Delimiter to separate the output data for CardData.
CloseDevice (Boolean)	Close the connection to the device after the request is processed. `false` = Do not close the device. (default) `true` = Close the device.

EndSession (Boolean)	Send an EndSession command after the request is processed. `false` = Do not end the session. (default) `true` = End the session.
EndSessionDisplayMessage (Integer)	Display to show on the device when EndSession is set to true. 0 = “Welcome” (default) 1 = Bitmap Slot 1 2 = Bitmap Slot 2 3 = Bitmap Slot 3 4 = Bitmap Slot 4

Return value:

```
{“CardSwipeOutput”: {}, “AdditionalOutputData”:{}}
{
   "CardSwipeOutput":    {
      "CardOperationStatus": ,
      "CardStatus": ,
      "CardType": ,
      "DataType": ,
      "EncryptedMagnePrint": "",
      "EncryptedTrack1": "",
      "EncryptedTrack2": "",
      "EncryptedTrack3": "",
      "EncryptedMagnePrintLength": ,
      "EncryptedMagnePrintStatus": ,
      "EncryptedTrack1Length": ,
      "EncryptedTrack1Status": ,
      "EncryptedTrack2Length": ,
      "EncryptedTrack2Status": ,
      "EncryptedTrack3Length": ,
      "EncryptedTrack3Status": ,
      "MagStripeStatus": ,
      "PANDataLength": ,
      "Track1Length": ,
      "Track1Status": ,
      "Track2Length": ,
      "Track2Status": ,
      "Track3Length": ,
      "Track3Status": ,
      "StatusCode": ,
      "CardData": "",
      "CBCMAC": "",
      "KSN": "",
      "MagnePrintStatus": "",
      "PANData": "",
      "MagTekSerialNumber": "",
      "Track1": "",
      "Track2": "",
      "Track3": ""
   },
   "AdditionalOutputData": null
}
```

Parameter (type)	Description
CardOperationStatus (Integer)	Card operation status
CardStatus (Integer)	Bit masked card status. 0 = OK Otherwise, for each track, the possible values are listed below: When bit value is 0 = No error When bit value is 1 = Error detected for that track. Bit 7 - 0 Bit 6 - 0 Bit 5 - 0 Bit 4 - ICC Bit 3 - Track 3 Bit 2 - Track 2 Bit 1 - Track 1 Bit 0 - 0
CardType (Integer)	Card type: 0 = Other 1 = Financial 2 = AAMVA 3 = Manual 4 = Unknown 5 = ICC 6 = Contactless ICC - EMV 7 = Financial MSR + ICC 8 = Contactless ICC - MSD
DataType (Integer)	Data type. This is the report ID from the raw HID report descriptor.
EncryptedMagnePrint (Hexadecimal string)	Encrypted MagnePrint data.
EncryptedTrack1 (Hexadecimal string)	Encrypted track 1 data.
EncryptedTrack2 (Hexadecimal string)	Encrypted track 2 data.
EncryptedTrack3 (Hexadecimal string)	Encrypted track 3 data.
EncryptedMagnePrintLength (Integer)	Encrypted MagnePrint data length.
EncryptedMagnePrintStatus (Integer)	Encrypted MagnePrint status.

EncryptedTrack1Length (Integer)	Encrypted track 1 length.
EncryptedTrack1Status (Integer)	Encrypted track 1 status
EncryptedTrack2Length (Integer)	Encrypted track 2 length
EncryptedTrack2Status (Integer)	Encrypted track 2 status
EncryptedTrack3Length (Integer)	Encrypted track 3 length
EncryptedTrack3Status (Integer)	Encrypted track 3 status
MagStripeStatus (Integer)	Magnetic stripe status
PANDataLength (Integer)	PAN data length
Track1Length (Integer)	Track 1 data length
Track1Status (Integer)	Track 1 status
Track2Length (Integer)	Track 2 data length
Track2Status (Integer)	Track 2 status
Track3Length (Integer)	Track 3 data length
Track3Status (Integer)	Track 3 status
StatusCode (Integer)	Status code
CardData (String)	Card data delimited by a field separator, which was supplied from the request parameter FieldSeparator. This contains most of the fields of the response but in one string blob.
CBCMAC (Hexadecimal string)	CBC Mac
KSN (Hexadecimal string)	Key Serial Number used for the transaction.

MagnePrintStatus (Hexadecimal string)	Bit masked MagnePrint status: Bit 0 = MagnePrint capable flag Bits 1 to 15 = Product revision & 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
PANData (Hexadecimal string)	Primary Account Number data
MagTekSerialNumber (Hexadecimal string)	MagTek device serial number
Track1 (String)	Masked Track 1 data
Track2 (String)	Masked Track 2 data
Track3 (String)	Masked Track 3 data
AdditionalOutputData	Additional output data

Example Request/Response:

```
{
      "DeviceID": "",
      "WaitTime": 30,
      "DisplayMessage": 1,
      "Tones": 2,
      "FieldSeparator": "|",
      "CloseDevice": true,
      "EndSession": true,
      "EndSessionDisplayMessage": 0
}


{
   "CardSwipeOutput":    {
      "CardOperationStatus": 0,
      "CardStatus": 0,
      "CardType": 7,
      "DataType": 34,
      "EncryptedMagnePrint": "18083F638D47...",
      "EncryptedTrack1": "10BC99CBD8BD...",
      "EncryptedTrack2": "550D25B03440...",
      "EncryptedTrack3": "550D25B03440...",
      "EncryptedMagnePrintLength": 56,
      "EncryptedMagnePrintStatus": 0,
      "EncryptedTrack1Length": 72,
      "EncryptedTrack1Status": 0,
      "EncryptedTrack2Length": 40,
      "EncryptedTrack2Status": 0,
      "EncryptedTrack3Length": 24,
      "EncryptedTrack3Status": 0,
      "MagStripeStatus": 0,
      "PANDataLength": 24,
      "Track1Length": 72,
      "Track1Status": 0,
      "Track2Length": 39,
      "Track2Status": 0,
      "Track3Length": 18,
      "Track3Status": 0,
      "StatusCode": 0,
      "CardData":
"CardType=7|OperationStatus=0|CardStatus=0|DataType=34|Track1Status=0|
Track1Length=72|Track1=%B5443000040...|Track2Status=0|Track2Length=39|
Track2=;54430000400...|Track3Status=0|Track3Length=18|Track3=;54436848
000...|EncTrack1Status=0|EncTrack1Length=72|EncTrack1=10BC99CBD8BD...|
EncTrack2Status=0|EncTrack2Length=40|EncTrack2=550D25B03440807...|EncT
rack3Status=0|EncTrack3Length=24|EncTrack3=550D25B03440...|EncMPStatus
=0|EncMPLength=56|EncMP=18083F638D47...|MPSTS=00001000|MSStatus=0|KSN=
9010010B9999990000EF|SerialNumber=992B808819160710|PAN=550D25B03440807
D370B53E340DCAC68B0C54EFB292D66A5|CBCMAC=0D7D0055",
      "CBCMAC": "0D7D0055",
      "KSN": "9010010B9999990000EF",
      "MagnePrintStatus": "00001000",
      "PANData": "550D25B03440807D370B53E340DCAC68B0C54EFB292D66A5",
      "MagTekSerialNumber": "992B808819160710",
      "Track1": "%B5443000040...",
      "Track2": ";54430000400...",
      "Track3": ";54436848000..."
   },
   "AdditionalOutputData": null
}
```

4.5 RequestDeviceStatus

Retrieves the device status.

Using Method POST:

```
api/mtppscrahost/RequestDeviceStatus
{
      "DeviceID": "",
      "WaitTime": ,
      "CreateNewConnection":
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds the device will wait for the action to be completed. (1 - 255)
CreateNewConnection (Boolean)	Create a new connection. `false` = Do not create new connection `true` = Create new connection

Return value:

```
{
   "CardPresent": ,
   "DeviceState": ""
}
```

Parameter (Type)	Description
CardPresent (String)	Card present indicator. `false` = no card present `true` = card present
DeviceState (Integer)	Device state. 0x00 = Idle 0x01 = Session 0x02 = Wait For Card 0x03 = Wait For PIN 0x04 = Wait For Selection 0x05 = Displaying Message 0x06 = Test (Reserved for future use) 0x07 = Manual Card Entry 0x08 = Wait for Signature Capture (SC-S Only | SC-F Only) 0x09 = Wait Cardholder Entry 0x0A = Chip Card 0x0B = ICC Kernel Test 0x0C = EMV Transaction 0x0D = Show PAN

Example Request/Response:

```
{
     "DeviceID": "",
     "WaitTime": 0,
     "CreateNewConnection": false
}



{
   "CardPresent": false,
   "DeviceState": "00"
}
```

4.6 RequestEMVTags

Retrieves EMV tags from the device.
Using Method POST:

```
api/mtppscrahost/RequestEMVTags
{
      "DeviceID": "",
      "WaitTime": ,
      "TagType": ,
      "TagOperation": ,
      "DataBase": ,
      "Data": "",
      "RequestType" : ,
      "CloseDevice": ,
      "EndSession" : ,
      "EndSessionDisplayMessage" : ,
      "AdditionalRequestData": null
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds the device will wait for the action to be completed. (1 - 255)
TagType (Integer)	EMV tag to set or get: 00 = Reader tags 80 = Application tags Lower 7 bits indicate which application slot for the operation.
TagOperation (Integer)	Type of operation to be performed: 0x00 = Read single tag operation 0x03 = Read all tags operation 0x04 = Write operation 0xFF = Set to factory defaults
DataBase (Integer)	Database Selector: 00 = Contact L2 EMV Tags 01 = PayPass-MasterCard 02 = PayWave-VISA 03 = ExpressPay-AMEX 04 = Discover

Data
(String)
	TLV data block to send to the device.
	Data block must be formed as an F9 CBC-MAC container message.
	Reference the device manual for details.
	`AAAA /* 2-byte MSB message length excluding padding and CBC-MAC */`
	`F9<len> /* container for MAC structure and generic data */`
	`DFDF55(MAC Encryption Type)<len><val>`
	`DFDF25(IFD Serial Number)<len><val>`
	`FA<len>/* container for generic data */`
	`<tag><len><val>`
	`...`
	`<tag><len><val>`
	`<Buffer if any to make blocks as multiple of 8 bytes>`
	`<CBC-MAC (4 bytes, use MAC variant of AMK)>`

RequestType (enum)	“SET” for setting an EMV tag. “GET” for getting an EMV tag.
CloseDevice (Boolean)	Close the connection to the device after the request is processed. `false` = Do not close the device. (default) `true` = Close the device.
EndSession (Boolean)	Sends an EndSession command after the request is processed. `false` = Do not end the session. (default) `true` = End the session.
EndSessionDisplayMessage (Integer)	Display to show on the device when EndSession is set to true. 0 = “Welcome” (default) 1 = Bitmap Slot 1 2 = Bitmap Slot 2 3 = Bitmap Slot 3 4 = Bitmap Slot 4

Return value:

```
{
   "EMVTagOutput":    {
      "EMVTagOperationStatus": ,
      "EMVTagData": ""
   },
   "AdditionalOutputData": null
}
```

Parameter (Type)	Description
EMVTagOperationStatus (Integer)	EMV tag operation status. 0 = OK / Done 128 = Device Error, tamper has been detected, device certificate missing. 129 = Device not Idle 130 = Data Error or the command contains bad parameters. 134 = Device busy
EMVTagData (Hexadecimal string)	EMV tag data

Example Request/Response:

```
{
      "DeviceID": "",
      "WaitTime": 5,
      "TagType": 0,
      "TagOperation": 03,
      "DataBase": 0,
      "Data": "",
      "RequestType": "GET",
     "CloseDevice": false,
      "EndSession": false,
      "EndSessionDisplayMessage": 0,
     "AdditionalRequestData": null,
}


{
   "EMVTagOutput":    {
      "EMVTagOperationStatus": 0,
      "EMVTagData":
"0231F982022DDFDF550102DFDF2508992B808819160710FA820218DFDF220E31504159
2E5359532E4444463031DFDF21010A5F2A0208405F360102DFDF2401029F1A020840DFD
F2508992B808819160710 9F1E0831393136303731309F1C0831313232333334349F3303
E0F8C89F3501229F4005F200B0B001DFDF00018DDFDF011BA00000000396FFFFFFA0000
0000496FFFFFFA00000000596FFFFFFDFDF026B4F575A828A8E959A9B9C5F245F255F2A
5F349F029F039F069F079F089F099F0D9F0E9F0F9F109F1A9F269F279F339F349F359F3
69F379F40DFDF53F5F4DFDF30DFDF31DFDF32DFDF33DFDF34DFDF35F49F219F039F1E9F
399F419F53849F6E5F2099DFDF4150DFDF259F12DFDF03015ADFDF0407917 1729F01898
ADFDF0529829F36DFDF259F109F5B9F339F35959F015F245A5F348A9F159F169F399F1A
9F1C579F025F2A9A9F21DFDF06028A91DFDF070100DFDF130101DFDF140400000BB8DFD
F150400000001DFDF160400000080DFDF172E828E5F245F259F069F079F0D9F0E9F0F9F
109F269F279F36959B9C9F339F349F379F40DFDF70DFDF71DFDF729F5BDFDF1902656ED
FDF200153DFDF2D0A656E6672697464656573DFDF670101DFDF6801019F3C0209989F3D
0102DFDF4E04000001109F530152DFDF26100000000000000000000000000000000DFD
F2714716DC2745731D4185A14BDD5A043F2EBD67A6AF0DFDF281000000000000000000
00000000000000DFDF2914E57AF16E76F6EAB9DB5D3F13692BB3C1E8C75B72000000000
04588BAD6"
   },
   "AdditionalOutputData": null
}
```

4.7 RequestEndEMVSession

Ends an EMV session for the device.
Using Method POST:

```
api/mtppscrahost/RequestEndEMVSession
{
     "DeviceID": "",
     "EndSessionDisplayMessage" :
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
EndSessionDisplayMessage (Integer)	Display to show on the device. 0 = “Welcome” (default) 1 = Bitmap Slot 1 2 = Bitmap Slot 2 3 = Bitmap Slot 3 4 = Bitmap Slot 4

Return value:

```
{
   "DeviceID": "",
   "ResultStatus": ,
   "ResultMessage": "",
   "CardPresent": ,
   "DeviceState": null
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
ResultStatus (Boolean)	Status of the operation. `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.
CardPresent (Boolean)	Card present indicator. `false` = no card present `true` = card present

DeviceState (Integer)	Device state

4.8 RequestManualSwipe

Prompts the user to manually enter card data.
Using Method POST:

```
api/mtppscrahost/RequestManualSwipe
{
      "DeviceID": "",
      "WaitTime": ,
      "Options": ,
      "Tones": ,
      "CloseDevice": ,
      "EndSession": ,
      "EndSessionDisplayMessage":
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds the device will wait for the action to be completed. (1 - 255)
Options (Integer)	This is an ORed combination of flags that changes the device's data entry request behavior as follows: Bits 0 and 1 0 = Acct,Date,CVC 1 = Acct,Date 2 = Acct,CVC 3 = Acct Bit 2 1=Use QwickCodes entry Bit 3 1=Use PAN in PIN block creation Bit 4 0=Use PAN min 9, max 19 1=Use PAN min 14, max 21 Bits 5-7 are reserved and should be set to 0.
Tones (Integer)	Tones to use: 0 = No sound 1 = One beep 2 = Two beeps

Parameter (Type)	Description
CloseDevice (Boolean)	Close the connection to the device after the request is processed. `false` = Do not close the device. (default) `true` = Close the device.
EndSession (Boolean)	Sends an EndSession command after the request is processed. `false` = Do not end the session. (default) `true` = End the session.
EndSessionDisplayMessage (Integer)	Display to show on the device when EndSession is set to true. 0 = “Welcome” (default) 1 = Bitmap Slot 1 2 = Bitmap Slot 2 3 = Bitmap Slot 3 4 = Bitmap Slot 4

Return value:

```
{
   "CardManualOutput":    {
      "CardOperationStatus": ,
      "CardStatus": ,
      "CardType": ,
      "DataType": ,
      "EncryptedMagnePrint": "",
      "EncryptedTrack1": "",
      "EncryptedTrack2": "",
      "EncryptedTrack3": "",
      "EncryptedMagnePrintLength": ,
      "EncryptedMagnePrintStatus": ,
      "EncryptedTrack1Length": ,
      "EncryptedTrack1Status": ,
      "EncryptedTrack2Length": ,
      "EncryptedTrack2Status": ,
      "EncryptedTrack3Length": ,
      "EncryptedTrack3Status": ,
      "MagStripeStatus": ,
      "PANDataLength": ,
      "Track1Length": ,
      "Track1Status": ,
      "Track2Length": ,
      "Track2Status": ,
      "Track3Length": ,
      "Track3Status": ,
      "StatusCode": ,
      "CardData": "",
      "CBCMAC": "",
      "KSN": "",
      "MagnePrintStatus": "",
      "PANData": "",
      "MagTekSerialNumber": "",
      "Track1": "",
      "Track2": "",
      "Track3": ""
   },
   "StatusCode": ,
   "AdditionalOutputData": null
}
```

Parameter (Type)	Description
CardOperationStatus (Integer)	Card operation status
CardStatus (Integer)	Bit masked card status. 0 = OK Otherwise, for each track, the possible values are listed below: Bit value 0 = No error Bit value 1 = Error detected Bit 7 - 0 Bit 6 - 0 Bit 5 - 0 Bit 4 - ICC Bit 3 - Track 3 Bit 2 - Track 2 Bit 1 - Track 1 Bit 0 - 0
CardType (Integer)	Card type: 0 = Other 1 = Financial 2 = AAMVA 3 = Manual 4 = Unknown 5 = ICC 6 = Contactless ICC - EMV 7 = Financial MSR + ICC 8 = Contactless ICC - MSD
DataType (Integer)	Data type. This is the report ID from the raw HID report descriptor.
EncryptedMagnePrint (Hexadecimal string)	Encrypted MagnePrint data.
EncryptedTrack1 (Hexadecimal string)	Encrypted track 1 data.
EncryptedTrack2 (Hexadecimal string)	Encrypted track 2 data.

EncryptedTrack3 (Hexadecimal string)	Encrypted track 3 data.
EncryptedMagnePrintLength (Integer)	Encrypted MagnePrint data length.
EncryptedMagnePrintStatus (Integer)	Encrypted MagnePrint status.
EncryptedTrack1Length (Integer)	Encrypted track 1 length.
EncryptedTrack1Status (Integer)	Encrypted track 1 status
EncryptedTrack2Length (Integer)	Encrypted track 2 length
EncryptedTrack2Status (Integer)	Encrypted track 2 status
EncryptedTrack3Length (Integer)	Encrypted track 3 length
EncryptedTrack3Status (Integer)	Encrypted track 3 status
MagStripeStatus (Integer)	Magnetic stripe status
PANDataLength (Integer)	PAN data length
Track1Length (Integer)	Track 1 data length
Track1Status (Integer)	Track 1 status
Track2Length (Integer)	Track 2 data length
Track2Status (Integer)	Track 2 status
Track3Length (Integer)	Track 3 data length
Track3Status (Integer)	Track 3 status
StatusCode (Integer)	Status code
CardData (String)	Card data delimited by a field separator, which was supplied from the request parameter FieldSeparator. This contains most of the fields of the response but in one string blob.
CBCMAC (Hexadecimal string)	CBC Mac

KSN (Hexadecimal string)	Key Serial Number used for the transaction.
MagnePrintStatus (Hexadecimal string)	Bit masked MagnePrint status. • Bit 0 = MagnePrint capable flag • Bits 1 to 15 = Product revision & 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
PANData (Hexadecimal string)	Primary Account Number data
MagTekSerialNumber (Hexadecimal string)	MagTek device serial number
Track1 (String)	Masked Track 1 data
Track2 (String)	Masked Track 2 data
Track3 (String)	Masked Track 3 data
StatusCode (Integer)	Status code
AdditionalOutputData (name/value pair)	Additional output data

Example Request/Response:

```
{
      "DeviceID": "",
      "WaitTime": 20,
      "Options": 0,
      "Tones": 2,
      "CloseDevice": true,
      "EndSession": true,
      "EndSessionDisplayMessage": 0
}


{
   "CardManualOutput":    {
      "CardOperationStatus": 0,
      "CardStatus": 0,
      "CardType": 3,
      "DataType": 34,
      "EncryptedMagnePrint": "",
      "EncryptedTrack1":
"314424973DEA436A5026AE842830EDF1EE44C1949E3E2ED734516079FFC086C8C030A
4A7C7D7882F2F3D17CAC7821DC3313358FE40D639F3",
      "EncryptedTrack2":
"5A655F7936E9D6672385C8EACF63095FB06C4813453AC549E131A551ABB43971CAA3F
5B7992FB924",
      "EncryptedTrack3": "",
      "EncryptedMagnePrintLength": 0,
      "EncryptedMagnePrintStatus": 1,
      "EncryptedTrack1Length": 56,
      "EncryptedTrack1Status": 0,
      "EncryptedTrack2Length": 40,
      "EncryptedTrack2Status": 0,
      "EncryptedTrack3Length": 0,
      "EncryptedTrack3Status": 1,
      "MagStripeStatus": 0,
      "PANDataLength": 24,
      "Track1Length": 53,
      "Track1Status": 0,
      "Track2Length": 35,
      "Track2Status": 0,
      "Track3Length": 0,
      "Track3Status": 1,
      "StatusCode": 0,
      "CardData":
"CardType=3|OperationStatus=0|CardStatus=0|DataType=34|Track1Status=0|
Track1Length=53|Track1=%M1234000090003456^MANUAL
ENTRY/^2312000000000000000?|Track2Status=0|Track2Length=35|Track2=;123
4000090003456=2312000000000000?|Track3Status=1|Track3Length=0|Track3=|
EncTrack1Status=0|EncTrack1Length=56|EncTrack1=314424973DEA436A5026AE8
42830EDF1EE44C1949E3E2ED734516079FFC086C8C030A4A7C7D7882F2F3D17CAC7821
DC3313358FE40D639F3|EncTrack2Status=0|EncTrack2Length=40|EncTrack2=5A6
55F7936E9D6672385C8EACF63095FB06C4813453AC549E131A551ABB43971CAA3F5B79
92FB924|EncTrack3Status=1|EncTrack3Length=0|EncTrack3=|EncMPStatus=1|E
ncMPLength=0|EncMP=|MPSTS=00000000|MSStatus=0|KSN=9010010B9999990000F2
|SerialNumber=992B808819160710|PAN=5A655F7936E9D6672385C8EACF63095F446
B7F4FDBA27962|CBCMAC=22D86543",
      "CBCMAC": "22D86543",
      "KSN": "9010010B9999990000F2",
      "MagnePrintStatus": "00000000",
      "PANData": "5A655F7936E9D6672385C8EACF63095F446B7F4FDBA27962",
      "MagTekSerialNumber": "992B808819160710",
      "Track1": "%M1234000090003456^MANUAL
ENTRY/^2312000000000000000?",
      "Track2": ";1234000090003456=2312000000000000?",
      "Track3": ""
   },
   "StatusCode": 0,
   "AdditionalOutputData": null
}
```

4.9 RequestOperationStatus

Retrieves the operation status of the device.
Using Method POST:

```
api/mtppscrahost/RquestOperationStatus
```

Return value:

```
{
   "OperationStatus": ,
   "DeviceID": "",
   "CreateNewConnection": ,
   "AdditionalOutputData": null
}
```

Parameter (Type)	Description
OperationStatus (Integer)	Operation status. See the Programmer's Manual (Commands) for details of the operation that was started.
DeviceID (Integer)	URI of the device. See **DeviceID URI** for details.
CreateNewConnection (Boolean)	Create new connection
AdditionalOutputData (name/value pairs)	Addition output data

Example Request/Response:

```
{

}



{
   "OperationStatus": 0,
   "DeviceID": null,
   "CreateNewConnection": false,
   "AdditionalOutputData": null
}
```

4.10 RequestPIN

Prompts the user to enter a PIN by displaying one of five predetermined messages and plays a specified sound.
Using Method POST:

```
api/mtppscrahost/RequestPIN
{
     "DeviceID": "",
     "WaitTime": ,
     "PinMode": ,
     "MaxPinLength": ,
     "MinPinLength": ,
     "Options": ,
     "Tones": ,
     "CloseDevice": ,
     "EndSession": ,
     "EndSessionDisplayMessage":
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds the device will wait for the action to be completed. (1 - 255)
PinMode (Integer)	Message to display as a user prompt: 0 = PINMsgEnterPIN 1 = PINMsgEnterPINAmt 2 = PINMsgReenterPINAmt 3 = PINMsgReenterPIN 4 = PINMsgVerifyPIN
MaxPinLength (Integer)	Maximum PIN length. Must be less than 13.
MinPinLength (Integer)	Minimum PIN length. Must be greater than 3.
Options (Integer)	PIN verification and format: 0 = ISO0 Format, No verify PIN 1 = ISO3 Format, No verify PIN 2 = ISO0 Format, Verify PIN 3 = ISO3 Format, Verify PIN
Tones (Integer)	Tones to use: 0 = No sound 1 = One beep 2 = Two beeps
CloseDevice (Boolean)	Close the connection to the device after the request is processed. `false` = Do not close the device. (default) `true` = Close not device.
EndSession (Boolean)	Sends an EndSession command after the request is processed. `false` = Do not end the session. (default) `true` = End the session.

Parameter (Type)	Description
EndSessionDisplayMessage (Integer)	Display to show on the device when EndSession is set to true. 0 = “Welcome” (default) 1 = Bitmap Slot 1 2 = Bitmap Slot 2 3 = Bitmap Slot 3 4 = Bitmap Slot 4

Return value:

```
{
   "PINOutput": {"PINData":"[PIN KSN],[EPB],[Operationstatus]"},
   "AdditionalOutputData": null
}
```

The PINData key contains a comma delimited value.

Field Name (Type)	Value
PIN KSN (Hexadecimal string)	PIN Key serial number
EPB (Hexadecimal string)	Encrypted PIN block
OperationStatus (Integer)	Operation status 0 = OK / Done 1 = Cardholder Cancel 2 = Timeout 3 = Host Cancel 4 = Verify fail 5 = Keypad Security
AdditionalOutputData (name/value pairs)	Additional output data

Example Request/Response:

```
{
     "DeviceID": "",
     "WaitTime": 10,
     "PinMode": 0,
     "MaxPinLength": 6,
     "MinPinLength": 4,
     "Options": 0,
     "Tones": 2,
     "CloseDevice": true,
     "EndSession": true,
     "EndSessionDisplayMessage": 0
}
```

```
{
   "PINOutput": {"PINData":
"FFFF9876543210E0003A,006B30D36D752D4D,0"},
   "AdditionalOutputData": null
}
```

4.11 RequestSendCommand

Sends a command to the device and returns the raw response from the device.
Using Method POST:

```
api/mtppscrahost/RquestSendCommand
{
      "DeviceID": "",
      "WaitTime": ,
      "Data": "",
      "RequestType": "",
      "WaitForReport": "",
      "CloseDevice": ,
      "EndSession": ,
      "EndSessionDisplayMessage":
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds the device will wait for the action to be completed. (1 - 255)
Data (String)	Hex string for command. Reference device manual for details.
RequestType (Enum)	“SET” for commands where the ACK status is to be returned. “GET” for commands where the data is to be returned.
WaitForReport (String)	The report number to wait for before returning the response. Example: Command 09 (Get Config) will respond with an ACK (01) or with data in Get Mode (30). WaitForReport = “01” will return the response for ACK report (01). WaitForReport = “09” will return the data report (09). See the device Programmer’s Manual (Commands) for report numbers corresponding to a command request.
CloseDevice (Boolean)	Close the connection to the device after the request is processed. false = Do not close the device. (default) true = Close the device.

EndSession (Boolean)	Sends an EndSession command after the request is processed. `false` = Do not end the session. (default) `true` = End the session.
EndSessionDisplayMessage (Integer)	Display message to show on the device. 0 = “Welcome” (default) 1 = Bitmap Slot 1 2 = Bitmap Slot 2 3 = Bitmap Slot 3 4 = Bitmap Slot 4

Return value:

```
{
   "Data": "",
   "AdditionalOutputData": ,
   "ResultStatus": ,
   "ResultMessage": ,
   "CardPresent": ,
   "DeviceState": ""
}
```

Parameter (Type)	**Description**
Data (Hexadecimal string)	Returned data for the command. See the device Programmer’s Manual (Commands) for details.
AdditionalOutputData (Integer)	Additional output data
ResultStatus (Boolean)	Result status `false` = fail `true` = success
ResultMessage (String)	Message explaining the status of the operation.
CardPresent (Boolean)	Card present indicator. `false` = card is not present `true` = card is present

DeviceState (Hexadecimal string)	Device state. 0x00 = Idle 0x01 = Session 0x02 = Wait For Card 0x03 = Wait For PIN 0x04 = Wait For Selection 0x05 = Displaying Message 0x06 = Test (Reserved for future use) 0x07 = Manual Card Entry 0x08 = Wait for Signature Capture (SC-S Only | SC-F Only) 0x09 = Wait Cardholder Entry 0x0A = Chip Card 0x0B = ICC Kernel Test 0x0C = EMV Transaction 0x0D = Show PAN

The Data output contains the response report sent from the device. Responses from SET request types returns the ACK status. Responses from GET request types returns the report data in response to the requested command.

Example Request/Response for Set display message

```
{
      "DeviceID": "",
      "WaitTime": 5,
      "Data": "070501",
      "RequestType": "SET",
      "WaitForReport": "01",
      "CloseDevice": false,
      "EndSession": false,
      "EndSessionDisplayMessage": 0
}


{
   "Data":
"010007000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000",
   "AdditionalOutputData": null,
   "ResultStatus": true,
   "ResultMessage": null,
   "CardPresent": false,
   "DeviceState": "00"
}
```

Example Request/Response for Get device configuration:

```
{
      "DeviceID": "",
      "WaitTime": 5,
      "Data": "09",
      "RequestType": "GET",
      "WaitForReport": "09",
      "CloseDevice": false,
      "EndSession": false,
      "EndSessionDisplayMessage": 0
}


{
   "Data":
"090801C0D53044012C000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000",
   "AdditionalOutputData": null,
   "ResultStatus": true,
   "ResultMessage": null,
   "CardPresent": false,
   "DeviceState": "00"
}
```

4.12 RequestSignature

Prompts the user to sign on the device’s screen.
Using Method POST:

```
api/mtppscrahost/RequestSignature
{
      "DeviceID": "",
      "WaitTime": ,
      "Options": ,
      "Tones": ,
      "CloseDevice": ,
      "EndSession": ,
      "EndSessionDisplayMessage":
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds the device will wait for the action to be completed. (1 - 255)
Options (Integer)	Option to select the timeout behavior. 0 = Timeout will clear data 1 = Timeout with available data, signature can be retrieved if exists
Tones (Integer)	Tones to use: 0 = No sound 1 = One beep 2 = Two beeps

Parameter (Type)	Description
CloseDevice (Boolean)	Close the connection to the device after the request is processed. `false` = Do not close the device. (default) `true` = Close the device.
EndSession (Boolean)	Sends an EndSession command after the request is processed. `false` = Do not end the session. (default) `true` = End the session.
EndSessionDisplayMessage (Integer)	Display to show on the device when EndSession is set to true. 0 = “Welcome” (default) 1 = Bitmap Slot 1 2 = Bitmap Slot 2 3 = Bitmap Slot 3 4 = Bitmap Slot 4

Return value: The Error! Reference source not found..

```
{“SignatureOutput”: {
     “SignatureOutputStatus”: ,
     “SignatureData”: } ,
 “AdditionalOutputData”: {}}
```

Parameter (Type)	Description
SignatureOperationStatus (Integer)	Signature operation status. 0 = success. Otherwise, an error
SignatureData (Base64 string)	Signature data encoded as Base64.
AdditionaOutputData (name/value strings)	Additional output data.

Example Request/Response:

```
{
     "DeviceID": "",
     "WaitTime": 30,
     "Options": 1,
     "Tones": 2,
     "CloseDevice": false,
     "EndSession": false,
     "EndSessionDisplayMessage": 0
}



{
   "SignatureOutput":    {
      "SignatureOperationStatus": 0,
      "SignatureData":
"HG8cbxxvHW0eah9kIVskUChELDgwLjInNCM1IjYkOCo5Mzs/PktAVENaRV9GYUhgS1tQV
VNLVkNZPVs6XTlgOmRAaUhwUnZae2GAZ///eil6KXopeil6KXopfCp/KoQqjCqWKaApqim
zKP//pTOlM6UzpTOlM6UzpTOlNaY4pj2oRKlMrFWuXbBn//8="
   },
   "AdditionalOutputData": null
}
```

4.13 RequestSmartCard

Begins an EMV transaction.
Using Method POST:

```
api/mtppscrahost/RequestSmartCard
{
      "DeviceID": "",
      "CardType": ,
      "ConfirmationWaitTime": ,
      "PINEntryWaitTime": ,
     "Tones": ,
     "Options": ,
     "TransactionType": ,
     "Amount": ,
     "CashBack": ,
     "QwickChipMode": ,
     "Reserved": ,
     "CloseDevice": ,
      "EndSession": ,
      "EndSessionDisplayMessage": ,
     "AdditionalRequestData": null
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
CardType (Integer)	Card type that can be used for the transaction: 1 = Magnetic stripe 2 = Contact smart card 3 = Magnetic stripe or contact smart card 4 = Contactless smart card (Not supported on DynaPro Mini) 5 = Contactless smart card + magnetic stripe 6 = Contactless smart card + contact smart card 7 = Magnetic stripe + contact smart card + contactless smart card.
ConfirmationWaitTime (Integer)	Time the device will wait for the user to begin the transaction.
PINEntryWaitTime (Integer)	Time the device will wait for the user to enter the PIN.

Parameter (Type)	Description
Tones (Integer)	Tones to use: 0 = No sound 1 = One beep 2 = Two beeps
Options (Integer)	Transaction options: 0 = Normal 1 = Bypass PIN 2 = Force Online 4 = Acquirer not available
TransactionType (Integer)	Type of transaction to be used: 0x00 = Purchase 0x01 = Cash Advance 0x02 or 0x09 = Cashback 0x04 = Goods (Purchase) 0x08 = Services (Purchase) 0x12 = Cash Manual 0x20 = Refund (Chip Card Contactless Only) 0x50 = Payment (Chip Card Contact Only)
Amount (Decimal)	The amount to be used and authorized in decimal format. 1.01 = 1 dollar and 1 cent
CashBack (Decimal)	The amount of cashback to be used and authorized in decimal format. 1.01 = 1 dollar and 1 cent
QwickChipMode (Boolean)	Start the transaction in Qwick Chip mode. In Qwick Chip mode, the devices does not prompt for an amount approval and needs no Acquirer Response (RequestSendAcquirerResponse). `false` = Do not do quick chip mode. `true` = Quick chip mode.
Reserved (Integer)	Reserved for future use. These are the reserved bytes beginning at index 20 of the 0xA2 command. Reference device manual ***D99875585*** for more details. Example: `0000000000000000000000000840000100` – Quick Chip Mode `0000000000000000000000000840000003` – PIN Verify `0000000000000000000000000840000004` – PIN Set
CloseDevice (Boolean)	Close the connection to the device after the request is processed. `false` = Do not close the device. (default) `true` = Close the device.
EndSession (Boolean)	Sends an EndSession command after the request is processed. `false` = Do not end the session. (default) `true` = End the session.

Parameter (Type)	Description
EndSessionDisplayMessage (Integer)	Display to show on the device when EndSession is set to true. 0 = “Welcome” (default) 1 = Bitmap Slot 1 2 = Bitmap Slot 2 3 = Bitmap Slot 3 4 = Bitmap Slot 4
AdditionalRequestData (name/value pairs)	Additional name/value pairs of data.

Return value:

```
{"TransactionOutput": {
   "TransactionType": ,
   "TransactionStatus": ,
   "OperationStatus": ,
   "DataType": ,
   "ApplicationIdentifier": "",
   "CardBrand": "",
   "ARQCData": "",
   "BatchData": "",
   "RawARQCData": "",
   "RawBatchData": "",
   "KSN": "",
   "DeviceSerialNumber": "",
   "EncryptionType": "",
   "NumberOfPaddedBytes": ,
   "NumberOfPaddedBytesForBatch": ,
   "EMVSREDData": "",
   "EMVSREDDataForBatch": "",
   "MerchantData": "",
   "FallbackIndicator": "",
   "MaskedICCTrack2": "",
   "ServiceCode": "",
   "CardHolderName": "",
   "CardType": "",
   "ApprovalStatus":
   }
}
```

Parameter (Type)	Description
TransactionType (Integer)	Transaction type Transaction type 0 = ARQC 1 = Batch

TransactionStatus (Integer)	Transaction status 0x00 = Accept 0x01 = Decline 0x02 = Error 0x10 = Cancelled by Host 0x11 = Confirm Amount No 0x12 = Confirm Amount Timeout 0x13 = Confirm Amount Cancel 0x14 = MSR Select Credit 0x15 = MSR Select Debit 0x16 = MSR Select Credit/Debit timeout 0x17 = MSR Select Credit/Debit cancel 0x18 = Signature Capture Cancelled by Host 0x19 = Signature Capture Timeout 0x1A = Signature Capture Cancelled by Cardholder 0x1B = PIN entry Cancelled by Host 0x1C = PIN entry timeout 0x1D = PIN entry Cancelled by Cardholder 0x1E = Manual Selection Cancelled by Host 0x1F = Manual Selection timeout 0x20 = Manual Selection Cancelled by Cardholder 0x21 = Waiting For Card Cancelled by Host 0x22 = Waiting For Card timeout 0x23 = Waiting For Card Cancelled by Cardholder 0x24 = Waiting For Card ICC Seated 0x25 = Waiting For Card MSR Swiped 0xFF = Unknown
OperationStatus (Integer)	Operation status 0 = OK / Done 1 = Cardholder Cancel 2 = Timeout 3 = Host Cancel
DataType (Integer)	Data type. This is the report ID from the raw HID report descriptor.
ApplicationIdentifier (Hexadecimal string)	EMV Application Identifier
CardBrand (String)	Card brand
ARQCData (Hexadecimal string)	Authorization Request Cryptogram for the transaction. This should be coordinated with the transaction processor to request approval for the transaction.
BatchData (Hexadecimal string)	Batch data for the transaction. This contains the final result of the transaction.
RawARQCData (Base64 string)	Raw Authorization Request Cryptogram for the transaction

RawBatchData (Base64 string)	Raw Batch data for the transaction
KSN (Hexadecimal string)	Key serial number
DeviceSerialNumber (Hexadecimal string)	MagTek device serial number
EncryptionType (Hexadecimal)	Encryption type 80 = DUKPT Key Data variant 81 = DUKPT Key PIN variant
NumberOfPaddedBytes (Integer)	Number of padded bytes to the end of the decrypted EMV SRED data to make a multiple of 8 bytes.
NumberOfPaddedBytesForBatch (Integer)	Number of padded bytes to the end of the decrypted EMV SRED Batch data to make a multiple of 8 bytes.
EMVSREDData (Hexadecimal string)	EMV SRED data. This is data portion of the TLV tag DFDF59 from ARQCData.
EMVSREDDataForBatch (Hexadecimal string)	EMV SRED Batch data
FallbackIndicator (Hexadecimal)	Fallback indicator 00 = No Fallback 01 = Technical Fallback 81 = MSR Fallback
MaskedICCTrack2 (String)	Masked magnetic stripe data for track 2
ServiceCode (Integer)	Service code
CardHolderName (Hexadecimal string)	Card holder name
CardType (Integer)	Card type. 0 = Other 1 = Financial 2 = AAMVA 3 = Manual 4 = Unknown 5 = ICC 6 = Contactless ICC - EMV 7 = Financial MSR + ICC 8 = Contactless ICC - MSD
ApprovalStatus (Integer)	Approval status

Example Request/Response:

```
{
      "DeviceID": "",
      "CardType": 7,
      "ConfirmationWaitTime": 10,
      "PINEntryWaitTime": 10,
     "Tones": 1,
     "Options": 0,
     "TransactionType": 2,
     "Amount": 1.00,
     "CashBack": 0.00,
     "QwickChipMode": true,
     "Reserved": ,
     "CloseDevice": true,
      "EndSession": true,
      "EndSessionDisplayMessage": 0,
     "AdditionalRequestData": null,
}


{"TransactionOutput": {
   "TransactionType": 1,
   "TransactionStatus": 1,
   "OperationStatus": 0,
   "DataType": 2,
   "ApplicationIdentifier": "A0000000041010",
   "CardBrand": "MasterCard",
   "ARQCData": "021EF982021A...",
   "BatchData": "01A3F982019F...",
   "RawARQCData": "Ah75ggIa3912...",
   "RawBatchData": "AaP5ggGf399U...",
   "KSN": "9010010B9999990000F6",
   "DeviceSerialNumber": "992B808819160710",
   "EncryptionType": "80",
   "NumberOfPaddedBytes": 1,
   "NumberOfPaddedBytesForBatch": 0,
   "EMVSREDData": "89C417447324...",
   "EMVSREDDataForBatch": "45113F6D603D...",
   "MerchantData": "DFDF4001005F...",
   "FallbackIndicator": "00",
   "MaskedICCTrack2": "3B3534343330...",
   "ServiceCode": "0201",
   "CardHolderName": "434F4E544143...",
   "CardType": "05",
   "ApprovalStatus": -1
}}
```

4.14 RequestSmartCardEx

Begins an EMV transaction. The response contains two separate transaction outputs. The first is ARQC (TransactionType0) and the second is Batch data (TransactionType1).
Using Method POST:

```
api/mtppscrahost/RequestSmartCardEx
{
     "DeviceID": "",
     "CardType": ,
     "ConfirmationWaitTime": ,
     "PINEntryWaitTime": ,
     "Tones": ,
     "Options": ,
     "TransactionType": ,
     "Amount": ,
     "CashBack": ,
     "QwickChipMode": ,
     "Reserved": ,
     "CloseDevice": ,
     "EndSession": ,
     "EndSessionDisplayMessage": ,
     "AdditionalRequestData": null
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
CardType (Integer)	Card type that can be used for the transaction: 1 = Magnetic stripe 2 = Contact smart card 3 = Magnetic stripe or contact smart card 4 = Contactless smart card (Not supported on DynaPro Mini) 5 = Contactless smart card + magnetic stripe 6 = Contactless smart card + contact smart card 7 = Magnetic stripe + contact smart card + contactless smart card.
Options (Integer)	Transaction options: 0 = Normal 1 = Bypass PIN 2 = Force Online 4 = Acquirer not available
TransactionType (Integer)	Type of transaction to be used: 0x00 = Purchase 0x01 = Cash Advance 0x02 or 0x09 = Cashback 0x04 = Goods (Purchase) 0x08 = Services (Purchase) 0x12 = Cash Manual 0x20 = Refund (Chip Card Contactless Only) 0x50 = Payment (Chip Card Contact Only)
Amount (decimal)	The amount to be used and authorized in decimal format. 1.01 = 1 dollar and 1 cent
CashBack (decimal)	The amount of cashback to be used and authorized in decimal format. 1.01 = 1 dollar and 1 cent

Parameter (Type)	Description
QwickChipMode (Boolean)	Start the transaction in Qwick Chip mode. In Qwick Chip mode, the devices does not prompt for an amount approval and needs no Acquirer Response (RequestSendAcquirerResponse). true = Qwick chip mode. false = Do not do qwick chip mode.
Reserved (Integer)	Reserved for future use. These are the reserved bytes beginning at index 20 of the 0xA2 command. Reference device manual ***D99875585*** for more details. Example: 0000000000000000000000000840000**01**00 – Quick Chip Mode 0000000000000000000000000840000000**03** – PIN Verify 0000000000000000000000000840000000**04** – PIN Set
AdditionalRequestData (name/value pairs)	Additional name/value pairs of data
CloseDevice (Boolean)	Close the connection to the device after the request is processed. `false` = Do not close the device. (default) `true` = Close the device.
EndSession (Boolean)	Sends an EndSession command after the request is processed. `false` = Do not end the session. (default) `true` = End the session.
EndSessionDisplayMessage (Integer)	Display to show on the device when EndSession is set to true. 0 = “Welcome” (default) 1 = Bitmap Slot 1 2 = Bitmap Slot 2 3 = Bitmap Slot 3 4 = Bitmap Slot 4

Return value:

```
{"TransactionOutput": [
      {
      "TransactionType": 0,
      "TransactionStatus": ,
      "OperationStatus": ,
      "DataType": ,
      "Data": "",
      "RawData": "",
      "KSN": "",
      "DeviceSerialNumber": "",
      "EncryptionType": "",
      "NumberOfPaddedBytes": ,
      "EMVSREDData": "",
      "MerchantData": "",
      "FallbackIndicator": "00",
      "MaskedICCTrack2": "",
      "ServiceCode": "0201",
      "CardHolderName": "",
      "CardType": "",
      "ApplicationIdentifier": null,
      "NumberOfPaddedBytesForBatch": ,
      "EMVSREDDataForBatch": null
   },
      {
      "TransactionType": 1,
      "TransactionStatus": ,
      "OperationStatus": ,
      "DataType": ,
      "Data": "",
      "RawData": "",
      "KSN": "",
      "DeviceSerialNumber": "",
      "EncryptionType": "",
      "NumberOfPaddedBytes": ,
      "EMVSREDData": "",
      "MerchantData": "",
      "FallbackIndicator": "",
      "MaskedICCTrack2": "",
      "ServiceCode": "",
      "CardHolderName": "",
      "CardType": "",
      "ApplicationIdentifier": "",
      "NumberOfPaddedBytesForBatch": ,
      "EMVSREDDataForBatch": null
   }
]}
```

Parameter (Type)	Description
TransactionType (Integer)	Transaction type 0 = ARQC 1 = Batch

TransactionStatus (Integer)	Transaction status 0x00 = Accept 0x01 = Decline 0x02 = Error 0x10 = Cancelled by Host 0x11 = Confirm Amount No 0x12 = Confirm Amount Timeout 0x13 = Confirm Amount Cancel 0x14 = MSR Select Credit 0x15 = MSR Select Debit 0x16 = MSR Select Credit/Debit timeout 0x17 = MSR Select Credit/Debit cancel 0x18 = Signature Capture Cancelled by Host 0x19 = Signature Capture Timeout 0x1A = Signature Capture Cancelled by Cardholder 0x1B = PIN entry Cancelled by Host 0x1C = PIN entry timeout 0x1D = PIN entry Cancelled by Cardholder 0x1E = Manual Selection Cancelled by Host 0x1F = Manual Selection timeout 0x20 = Manual Selection Cancelled by Cardholder 0x21 = Waiting For Card Cancelled by Host 0x22 = Waiting For Card timeout 0x23 = Waiting For Card Cancelled by Cardholder 0x24 = Waiting For Card ICC Seated 0x25 = Waiting For Card MSR Swiped 0xFF = Unknown
OperationStatus (Integer)	Operation status 0 = OK / Done 1 = Cardholder Cancel 2 = Timeout 3 = Host Cancel
DataType (Integer)	Data type. This is the report ID from the raw HID report descriptor.
Data (Hexadecimal string)	When TransactionType type is 0, this is ARQC data. When TransactionType type is 1, this is Batch data.
RawData (Base64 string)	Raw data
KSN (Hexadecimal string)	Key serial number
DeviceSerialNumber (Hexadecimal string)	MagTek device serial number
EncryptionType (Hexadecimal)	Encryption type 80 = DUKPT Key Data variant 81 = DUKPT Key PIN variant

NumberOfPaddedBytes (Integer)	Number of padded bytes to the end of the decrypted EMV SRED data to make a multiple of 8 bytes.
EMVSREDData (Hexadecimal string)	EMV SRED data. This is data portion of the TLV tag DFDF59 from ARQCData.
MerchantData (Hexadecimal string)	Merchant data
FallbackIndicator (Hexadecimal)	Fallback indicator 00 = No Fallback 01 = Technical Fallback 81 = MSR Fallback
MaskedICCTrack2 (String)	Masked magnetic stripe data for track 2
ServiceCode (Integer)	Service code
CardHolderName (Hexadecimal string)	Card holder name
CardType (Integer)	Card type. 0 = Other 1 = Financial 2 = AAMVA 3 = Manual 4 = Unknown 5 = ICC 6 = Contactless ICC - EMV 7 = Financial MSR + ICC 8 = Contactless ICC - MSD
ApplicationIdentifier (Hexadecimal string)	EMV Application Identifier
NumberOfPaddedBytesForBatch (Integer)	Number of padded bytes to the end of the decrypted EMV SRED Batch data to make a multiple of 8 bytes.
EMVSREDDataForBatch (Hexadecimal string)	EMV SRED Batch data

Example Request/Response:

```
{
      "DeviceID": "",
      "CardType": 7,
      "ConfirmationWaitTime": 10,
      "PINEntryWaitTime": 10,
     "Tones": 1,
     "Options": 0,
     "TransactionType": 2,
     "Amount": 1.00,
     "CashBack": 0.00,
     "QwickChipMode": true,
     "Reserved": ,
     "CloseDevice": true,
      "EndSession": true,
      "EndSessionDisplayMessage": 0,
     "AdditionalRequestData": null,
}


{"TransactionOutput": [
      {
      "TransactionType": 0,
      "TransactionStatus": 255,
      "OperationStatus": 0,
      "DataType": 1,
      "Data": "021EF982021A...",
      "RawData": "Ah75ggIa399U...",
      "KSN": "9010010B9999990000F7",
      "DeviceSerialNumber": "992B808819160710",
      "EncryptionType": "80",
      "NumberOfPaddedBytes": 1,
      "EMVSREDData": "03B74819C0BF...",
      "MerchantData": "",
      "FallbackIndicator": "00",
      "MaskedICCTrack2": "3B3534343330...",
      "ServiceCode": "0201",
      "CardHolderName": "434F4E544143...",
      "CardType": "05",
      "ApplicationIdentifier": null,
      "NumberOfPaddedBytesForBatch": 0,
      "EMVSREDDataForBatch": null
   },
      {
      "TransactionType": 1,
      "TransactionStatus": 1,
      "OperationStatus": 0,
      "DataType": 2,
      "Data": "01A3F982019F...",
      "RawData": "AaP5ggGf399U...",
      "KSN": "9010010B9999990000F7",
      "DeviceSerialNumber": "992B808819160710",
      "EncryptionType": "80",
      "NumberOfPaddedBytes": 0,
      "EMVSREDData": "6DAC17A9606...",
      "MerchantData": "DFDF4001005F...",
      "FallbackIndicator": "",
      "MaskedICCTrack2": "3B3534343330...",
      "ServiceCode": "",
      "CardHolderName": "434F4E544143...",
      "CardType": "",
      "ApplicationIdentifier": "A0000000041010",
      "NumberOfPaddedBytesForBatch": 0,
      "EMVSREDDataForBatch": null
```

```
    }
]}
```

4.15 RequestSendAcquirerResponse

Sends the ARPC to the device. Applicable only after a RequestSmartCard with QwickChipMode set to false.

Using Method POST:

```
api/mtppscrahost/RequestSendAcquirerResponse
{
      "DeviceID": "",
      "WaitTime": ,
      "IssuerAuthenticationData": "",
      "IssuerScriptTemplate1": "",
      "IssuerScriptTemplate2": "",
      "ApprovalStatus": ,
      "KSN": "",
      "DeviceSerialNumber": "",
      "CloseDevice": ,
      "EndSession": ,
      "EndSessionDisplayMessage": ,
      "AdditionalRequestData": null
}
```

Parameter (Type)	Description
DeviceID (String)	URI of the device. See **DeviceID URI** for details.
WaitTime (Integer)	Time in seconds the device will wait for the action to be completed. (1 - 255)
WaitTimeBeforeTransactionComplete (Integer)	Time in seconds to wait after receiving the transaction response before closing the device or ending the session.
IssuerAuthenticationData (String)	Issuer response to the transaction request in hexadecimal format. This field is for the data portion of the EVM Tag 91. Use 00 if not provided.
IssuerScriptTemplate1 (String)	Issuer Script to send to ICC in hexadecimal format. This field is for the data portion of the EVM Tag 71. Use 00 if not provided.
IssuerScriptTemplate2 (String)	Issuer Script to send to ICC in hexadecimal format. This field is for the data portion of the EVM Tag 72. Use 00 if not provided.

ApprovalStatus (Integer)	Status from acquirer/issuer. This field represents the data portion of the EMV Tag 8A. Example: `0` = Approve `1` = Decline
KSN (String)	Key serial number used for the transaction
DeviceSerialNumber (String)	Device serial number
CloseDevice (Boolean)	Close the connection to the device after the request is processed. `false` = Do not close the device. (default) `true` = Close the device.
EndSession (Boolean)	Sends an EndSession command after the request is processed. `false` = Do not end the session. (default) `true` = End the session.
EndSessionDisplayMessage (Integer)	Display to show on the device when EndSession is set to true. 0 = “Welcome” (default) 1 = Bitmap Slot 1 2 = Bitmap Slot 2 3 = Bitmap Slot 3 4 = Bitmap Slot 4
AdditionalRequestData (name/value pairs)	Additional name/value pairs of data

Return value:

```
{"TransactionOutput": {
   "TransactionType": ,
   "TransactionStatus": ,
   "OperationStatus": ,
   "BatchData": "",
   "RawBatchData": "",
   "KSN": "",
   "DeviceSerialNumber": "",
   "EncryptionType": "",
   "MerchantData": ""
}}
```

Parameter (Type)	Description
TransactionType (Integer)	Transaction type 0 = ARQC 1 = Batch

TransactionStatus (Integer)	Transaction status 0x00 = Accept 0x01 = Decline 0x02 = Error 0x10 = Cancelled by Host 0x11 = Confirm Amount No 0x12 = Confirm Amount Timeout 0x13 = Confirm Amount Cancel 0x14 = MSR Select Credit 0x15 = MSR Select Debit 0x16 = MSR Select Credit/Debit timeout 0x17 = MSR Select Credit/Debit cancel 0x18 = Signature Capture Cancelled by Host 0x19 = Signature Capture Timeout 0x1A = Signature Capture Cancelled by Cardholder 0x1B = PIN entry Cancelled by Host 0x1C = PIN entry timeout 0x1D = PIN entry Cancelled by Cardholder 0x1E = Manual Selection Cancelled by Host 0x1F = Manual Selection timeout 0x20 = Manual Selection Cancelled by Cardholder 0x21 = Waiting For Card Cancelled by Host 0x22 = Waiting For Card timeout 0x23 = Waiting For Card Cancelled by Cardholder 0x24 = Waiting For Card ICC Seated 0x25 = Waiting For Card MSR Swiped 0xFF = Unknown
OperationStatus (Integer)	Operation status 0 = OK / Done 1 = Cardholder Cancel 2 = Timeout 3 = Host Cancel
BatchData (Hexadecimal string)	Batch data for the transaction. This contains the final result of the transaction.
RawBatchData (Base64 string)	Raw Batch data for the transaction
KSN (Hexadecimal string)	Key serial number
DeviceSerialNumber (Hexadecimal string)	MagTek device serial number
EncryptionType (Hexadecimal)	Encryption type 80 = DUKPT Key Data variant 81 = DUKPT Key PIN variant
MerchantData (Hexadecimal string)	Merchant data

Example Request/Response:

```
{
      "DeviceID": "",
      "WaitTime": 5,
      "WaitTimeBeforeTransactionComplete": 3,
      "ApprovalStatus": 0,
      "IssuerAuthenticationData": "00",
      "IssuerScriptTemplate1": "00",
      "IssuerScriptTemplate2": "00",
      "KSN": "9010010B9999990000F9",
      "DeviceSerialNumber": "992B808819160710",
      "CloseDevice": true,
      "EndSession": true,
      "EndSessionDisplayMessage": 0,
      "AdditionalRequestData": null,
}


{"TransactionOutput": {
   "TransactionType": 1,
   "TransactionStatus": 0,
   "OperationStatus": 0,
   "BatchData": "024BF9820247...",
   "RawBatchData": "Akv5ggJH399U...",
   "KSN": "9010010B9999990000F9",
   "DeviceSerialNumber": "992B808819160710",
   "EncryptionType": "80",
   "MerchantData": "DFDF4001005F..."
}}
```

Appendix A TLV Data Format

A.1 ARQC Message Format

This section gives the format of the ARQC Message delivered in the ARQC Message notification. It is a TLV 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<len>/*container for ARQC */
                        DFDF53<len><value>/*fallback indicator */
                        5F20<len><value>/*cardholder name */
                        5F30<len><value>/*service code */
                        DFDF4D<len><value>/* Mask T2 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, always set to zeroes)
```

A.2 ARQC Response (from online processing)

This section gives the format of the data for the Online Processing Result / Acquirer Response message. This request is sent to the reader in response to an ARQC Message notification from the reader. It is a TLV 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<len>/* Container for ARQC */
                8A<len> approval
(ARQC padding, if any, to be a multiple of 8 bytes)
CBC-MAC (4 bytes, use MAC variant of MSR DUKPT key that was used in ARQC request, from
message length up to and including ARQC padding, if any)
```

A.3 Transaction Result Message – Batch Data Format

This section gives the format of the data the device uses to do completion processing

```
FE<len> /* container for generic data */
       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 */

                    F2<len>/* container for Batch Data */
                    … /* Batch Data tags defined in DFDF17 */
                    …/* Note: Sensitive Data cannot be defined in DFDF17 */

                    F3<len>/* container for Reversal Data, if any */
                    … /* Reversal Data tags defined in DFDF05 */
                    …/* Note: Sensitive Data cannot be defined in DFDF05 */

                    F7<len>/* container for Merchant Data */
                    … /* < Merchant Data tags */

                    F8<len> /* container tag for encrypted data */
                           DFDF56(Encrypted Transaction Data KSN)<len><val>
                           DFDF57(Encrypted Transaction Data Encryption Type)<val>

                           FA<len> /* container for generic data */
                              DF30(Encrypted Tag 56 TLV, T1 Data)<len><val>
                              DF31(Encrypted Tag 57 TLV, T2 Data)<len><val>
                              DF32(Encrypted Tag 5A TLV, PAN)<len><val>
                              DF35(Encrypted Tag 9F1F TLV, T1 DD)<len><val>
                              DF36(Encrypted Tag 9F20 TLV, T2, DD)<len><val>
                              DF37(Encrypted Tag 9F61 TLV, T2 CVC3)<len><val>
                              DF38(Encrypted Tag 9F62 TLV, T1,PCVC3)<len><val>
                              DF39(Encrypted Tag DF812A TLV, T1 DD)<len><val>
                              DF3A(Encrypted Tag DF812B TLV), T2 DD<len><val>
                              DF3B(Encrypted Tag DFDF4A TLV, T2 ISO Format)<len><val>
```

A.4 DeviceID URI

Parameter (type)	Description

DeviceID
(String)
	URI of the device.
	
	For USB devices, use the forms:
	Examples,
	An empty string or null to open the first device found.
	`“DeviceID”: “”`
	`“DeviceID”: null`
	
	`“DeviceID”: “[USB DynaFlex] B62CA5F”`
	`“DeviceID”: “[USB DynaPro] 98D90C660E070F0E”`
	
	For Ethernet devices, use the form:
	`IP://IPAddress:PORT`
	Example,
	`“DeviceID”: “IP://10.57.10.180:26”`
	
	For 802.11 Wireless devices, use in the form:
	`TLS12://TLSDEVICESERIALNUMBER`
	`TLS12TRUST://TLSDEVICESERIALNUMBER`
	`WS://DEVICESERIALNUMBER`
	Examples,
	`“DeviceID”: “TLS12://TLS99261829170E0810”`
	`“DeviceID”: “TLS12TRUST://TLS99261829170E0810”`
	`“DeviceID”: “ws://b62ca5f”`
	
	For BLE devices, use the form:
	DEVICENAME is listed in the operating system Bluetooth settings.
	`BLEEMV://DEVICENAME`
	Example,
	`“DeviceID”: “BLEEMV://DPG123456789A”`
	
	For BLUETOOTH_LE devices, use the form:
	DEVICENAME is listed in the operating system Bluetooth settings.
	`[BLE DynaFlex] DEVICENAME`
	Example,
	`“DeviceID”: “[BLE DynaFlex] DF II Go-BE000E4”`