MAGTEK®
SECURITY FROM THE INSIDE

Dynamag, DynaMAX, eDynamo, mDynamo

Secure Card Reader Authenticator
Programmer's Reference (WEBAPI)

May 2017

Manual Part Number:
D998200119-30

REGISTERED TO ISO 9001:2008

Table 0.1 – Revisions

Rev Number	Date	Notes
10	05/17/2016	Initial Release
20	06/22/2016	Added DynaPro format for EMV transaction messages.
30	05/08/2017	Added support for mDynamo.

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.

Table of Contents

1 Introduction

This document provides instructions for software developers who want to create software solutions that include an SCRA reader, Dynamag, DynaMAX, eDynamo, and mDynamo connected to a Windows-based host via USB, or BLE. It is part of a larger library of documents designed to assist SCRA reader, Dynamag, DynaMAX, eDynamo, and mDynamo implementers, which includes the following documents available from MagTek:

- ***D99875475 MagneSafe V5 Programmer's Reference (COMMANDS)***

1.1 About the MagTek SCRA WEB API

The MTSCRA WEB API, available from MagTek, provides demonstration source code and reusable MTSCRA WEBAPI DLLs that provide developers of custom software solutions with an easy-to-use interface for SCRA ready, Dynamag, DynaMAX, eDynamo, and mDynamo. Developers can include the MTSCRA WEBAPI DLLs in custom branded software which can be distributed to customers or distributed internally as part of an enterprise solution.

1.2 Nomenclature

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

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

The word "user" is also often used in different ways in different contexts. In this document, user generally refers to the cardholder.

1.3 SDK Contents

File	Description
MTDevice.DLL	MagTek SCRA Device constance library
MTLIB.DLL	MagTek SCRA interface library
MTSCRANET.DLL	MagTek SCRA library for .Net
MTSCRA.WEBAPI.DLL	MagTek SCRA library for WEB API
MTServiceNet.DLL	MagTek SCRA connection service library for .Net

1.4 System Requirements

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

Microsoft .Net Framework 4.5 installed.

Tested development environments:
Windows 8.1 with Microsoft Visual Studio 2013

1.5 Interfaces for Operating Systems

The following table matches the device interface to operating system.

Device	Interface	Operating System
SCRA readers	USB	Windows 7, Windows 8, 8.1 & Windows 10
Dynamag	USB	Windows 7, Windows 8, 8.1 & Windows 10
DynaMAX	USB	Windows 7, Windows 8, 8.1 & Windows 10
	BLE	Windows 8, 8.1 & Windows 10
eDynamo	USB	Windows 7, Windows 8, 8.1 & Windows 10
	BLE	Windows 8, 8.1 & Windows 10
mDynamo	USB	Windows 7, Windows 8, 8.1 & Windows 10

2 How to Set Up the MagTek SCRA WEB API

2.1 How to Connect MTSCRA WEP API Service to a Host

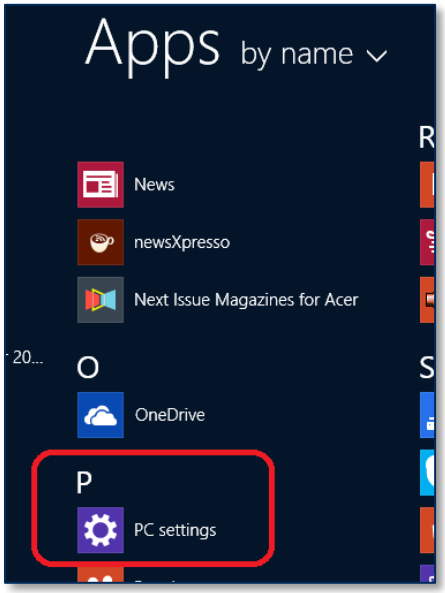
To use the WEB API (MTSCRA.WEBAPI.HostService.exe)

1) Set the header ContentType to "`application/json`"
2) Build the JSON object for the WEB API resource to be accessed.
3) Send HTTP request methods GET and POST to the base address http://localhost:9002/api/mtscrahost/ and add the resource endpoints.

2.2 How to Connect DynaMAX or eDynamo to a Windows Host via BLE

To connect DynaMAX or eDynamo 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 with DynaMAX or eDynamo:
   a) Make sure the host software is configured to look for the device on the proper connection.
   b) Make sure the host software knows which device(s) it should interface with.
   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 DynaMAX or eDynamo 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) 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

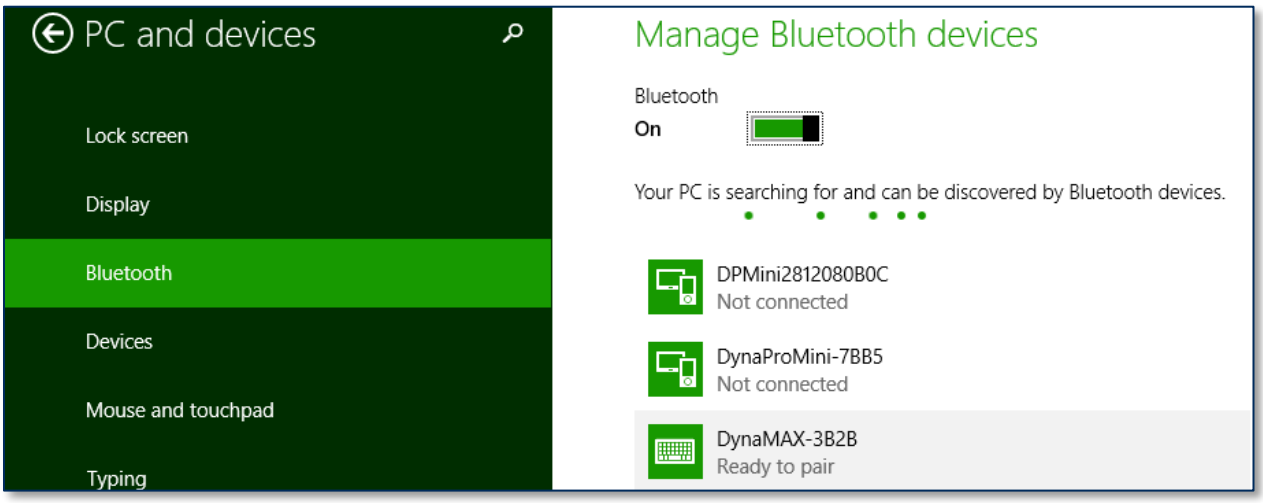
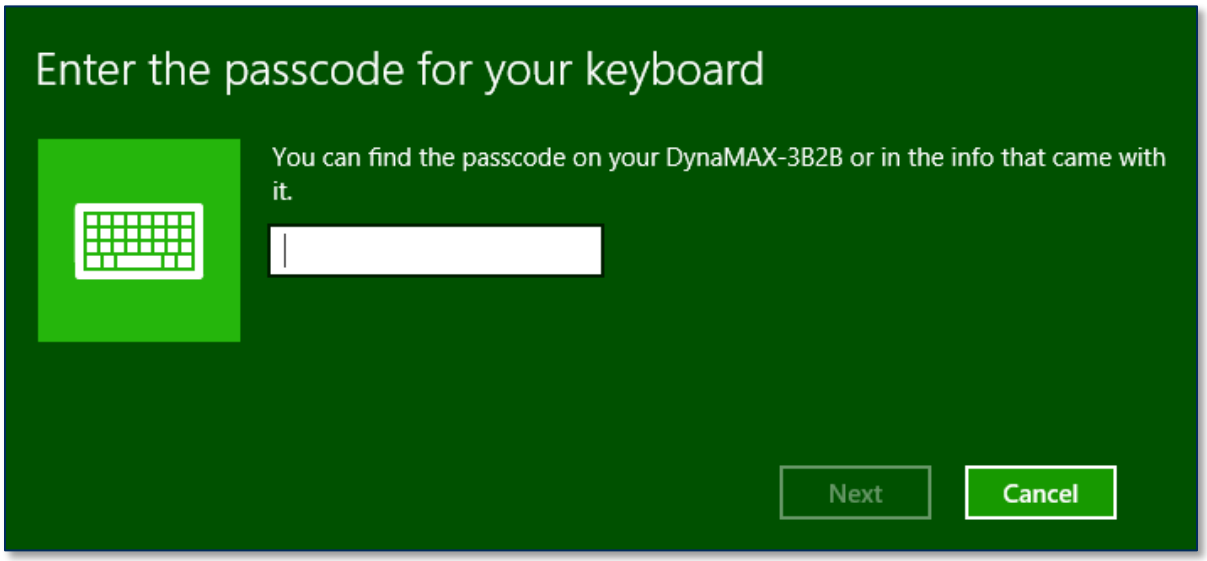
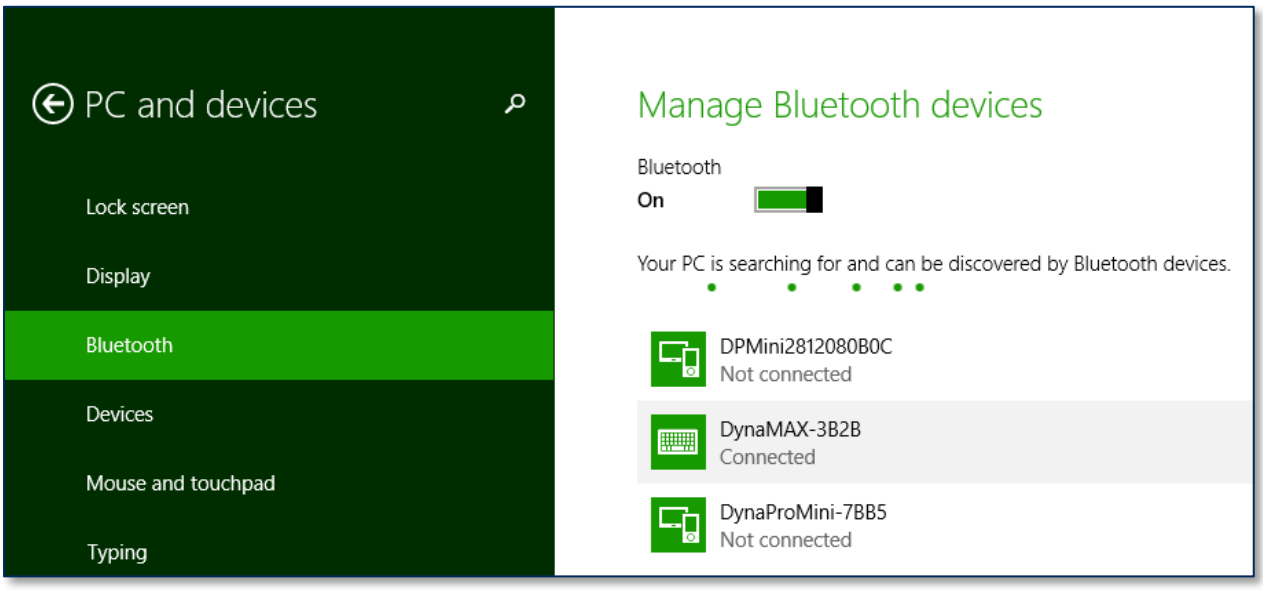


Figure 2-2 – Windows 8 Manage Bluetooth Devices App

9) Locate the serial number on the label on the bottom of the device. Note the final four digits.
10) Read through the list of pairable devices and locate the device called **DynaMAX-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**.
11) Select the device and press the **Pair** button. If the device is configured to run in KB mode, Windows will prompt you **Enter the passcode for your keyboard**.



12) 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.



13) Use the host software to test swiping a card. If you do not yet have host software and the device is configured to run in KB mode, open any text editor and swipe a card. The card contents should appear in the text editor.

14) 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.

15) Remember to change the default password. See the DynaMAX Mini 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 MTSCRA WEB API Resources

MTSCRA WEB API can be hosted as a Windows service (MagTek SCRA WEBAPI Host service or executable (MTSCRA.WEBAPI.Host.exe). MTSCRA WEB API receives REST requests and responses through a JSON object.

3.1 CheckHealth

Returns the operational status of the MTSCRA WEB API.
Using Method GET:

```
api/mtscrahost/CheckHealth
```

Return Value:
CheckHealth output. A String array containing API name and status.

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

3.2 RequestDeviceList

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

```
api/mtscrahost/RequestDeviceList(
int WaitTime,
int ConnectionType,
string[] AdditionalRequestData);
```

Parameter	Description
WaitTime	Time the device will wait for the user to complete a card swipe in seconds. (1 - 255)
ConnectionType	Device connection type: 1 = Audio 2 = BLE 3 = BLEEMV 4 = USB
AdditionalRequestData	Additional key/value pairs of data to be forwarded in the transaction.

Return Value:
The DeviceList output.

```
{"DeviceList": [{}], "AdditionalOutputData":{}}
```

3.3 RequestCardSwipe

Returns the magnetic stripe data after the device decodes a card swipe.
Using Method POST:

```
api/mtscrahost/RequestCardSwipe(
string DeviceID,
```

```
int WaitTime,
int ConnectionType,
string FieldSeparator,
string[] AdditionalRequestData);
```

Parameter	Description
DeviceID	URI of the device. For USB devices, use an empty string to open the first device found. Call requestDeviceList to retrieve an array of devices detected on the host based on connection type. Example HID device: "DeviceID": "\\\\?\\hid#vid_0801&pid_0011#7&23521b27&0&0000#{4d1e55b2-f16f-11cf-88cb-001111000030}"
WaitTime	Time the device will wait for the user to complete a card swipe in seconds. (1 - 255)
ConnectionType	Device connection type: 1 = Audio 2 = BLE 3 = BLEEMV 4 = USB
FieldSeparator	Delimiter to separate the output data.
AdditionalRequestData	Additional key/value pairs of data to be forwarded in the transaction.

Return Value:
The CardSwipe output.

```
{"CardSwipeOutput": {}, "AdditionalOutputData":{}}
```

3.4 RequestSendCommand

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

```
api/mtscrahost/RequestSendCommand(
string DeviceID,
int WaitTime,
int ConnectionType,
string Command,
string[] AdditionalRequestData);
```

Parameter	Description
DeviceID	URI of the device. For USB devices, use an empty string to open the first device found. Call requestDeviceList to retrieve an array of devices detected on the host based on connection type. Example HID device: "DeviceID": "\\\\?\\hid#vid_0801&pid_0011#7&23521b27&0&0000#{4d1e55b2-f16f-11cf-88cb-001111000030}"
WaitTime	Time the device will wait for the user to complete a card swipe in seconds. (1 - 255)
ConnectionType	Device connection type: 1 = Audio 2 = BLE 3 = BLEEMV 4 = USB
Command	Command data to send to device in Hex format.
AdditionalRequestData	Additional key/value pairs of data to be forwarded in the transaction.

Return Value:
The Response output in Hex string format of device raw response for this command.

```
{
   "ResponseOutput": {"Data":},
   "AdditionalOutputData": null
}
```

3.5 RequestSendExtendedCommand (eDynamo)

Sends an extended format command to the device and returns the raw response from the device.
Using Method POST:

```
api/mtscrahost/RequestSendExtendedCommand(
string DeviceID,
int WaitTime,
int ConnectionType,
string Command,
string[] AdditionalRequestData);
```

Parameter	Description
DeviceID	URI of the device. For USB devices, use an empty string to open the first device found. Call requestDeviceList to retrieve an array of devices detected on the host based on connection type. Example HID device: "DeviceID": "\\\\?\\hid#vid_0801&pid_0011#7&23521b27&0&0000#{4d1e55b2-f16f-11cf-88cb-001111000030}"

WaitTime	Time the device will wait for the user to complete a card swipe in seconds. (1 - 255)
ConnectionType	Device connection type: 1 = Audio 2 = BLE 3 = BLEEMV 4 = USB
Command	Command data to send to device in Hex format.
AdditionalRequestData	Additional key/value pairs of data to be forwarded in the transaction.

Return Value:
The Response output in Hex string format of device raw response for this command.

```
{
   "ResponseOutput": {"Data":},
   "AdditionalOutputData": null
}
```

3.6 RequestSmartCard (EMV devices)

Begins an EMV transaction.
Using Method POST:

```
api/mtscrahost/RequestSmartCard(
string DeviceID,
int WaitTime,
int ConnectionType,
int TransactionType,
int CardType,
decimal Amount,
decimal CurrencyCode,
int ReportOptions,
int Options,
string[] AdditionalRequestData);
```

Parameter	Description
DeviceID	URI of the device. For USB devices, use an empty string to open the first device found. Call requestDeviceList to retrieve an array of devices detected on the host based on connection type. Example HID device: "DeviceID": "\\\\?\\hid#vid_0801&pid_0011#7&23521b27&0&0000#{4d1e55b2-f16f-11cf-88cb-001111000030}"
WaitTime	Time the device will wait for the user to complete a card swipe in seconds. (1 - 255)

ConnectionType	Device connection type: 1 = Audio 2 = BLE 3 = BLEEMV 4 = USB
TransactionType	Type of transaction to be used: 0x00 = Purchase (listed as "Payment" on ICS) 0x01 = Cash Advance (not supported for this reader) 0x02 or 0x09 = Cash back (0x09 not supported, contactless) 0x04 = Goods (Purchase) 0x08 = Services (Purchase) 0x10 = International Goods (Purchase) 0x20 = Refund 0x40 = International Cash Advance or Cash Back 0x80 = Domestic Cash Advance or Cash Back
CardType	Card type that can be used for the transaction: 1 = Magnetic stripe 2 = Contact smart card 3 = Magnetic stripe or contact smart card
Amount	The amount to be used and authorized, EMV Tag 9F02. Format in decimal.
CashBack	Amount of cash back to be used, EMV Tag 9F02. Format in decimal.
CurrencyCode	Transaction Currency Code (EMV Tag 5F2A, format n4 string) Sample valid values: 0840 = US Dollar 0978 = Euro 0826 = UK Pound
ReportOptions	This field indicates the level of Transaction Status notifications the host desires to receive during the course of this transaction: 0 = Termination status only (normal termination, card error, timeout, host cancel) 1 = Major status changes (terminations, card insertions, waiting for user) 2 = All status changes (documents the entire transaction flow)
Options	Transaction options: 0 = Normal 1 = Bypass PIN 2 = Force Online 4 = Acquirer not available
AdditionalRequestData	Additional key/value pairs of data to be forwarded in the transaction.

Return Value:
The Response output.

```
{
   "ResponseOutput":
```

```
      {
      "ARQCData":,
      "BATCHData":,
      "UserSelection":
      "CommandResult":
      },
      "AdditionalOutputData": null
}
```

3.7 RequestUserSelection (EMV devices)

Sends the user selection result. RequestUserSelection should be called after RequestSmarCard returns data in the UserSelection field.

Using Method POST:

```
api/mtscrahost/RequestSendAcquirerResponse(
string DeviceID,
int WaitTime,
int ConnectionType,
int Status,
int Selection,
string[] AdditionalRequestData);
```

Parameter	Description
DeviceID	URI of the device. For USB devices, use an empty string to open the first device found. Call requestDeviceList to retrieve an array of devices detected on the host based on connection type. Example HID device: "DeviceID": "\\\\?\\hid#vid_0801&pid_0011#7&23521b27&0&0000#{4d1e55b2-f16f-11cf-88cb-001111000030}"
WaitTime	Time the device will wait for the user to complete a card swipe in seconds. (1 - 255)
ConnectionType	Device connection type: 1 = Audio 2 = BLE 3 = BLEEMV 4 = USB
Status	The status of User Selection: 0x00 = User Selection Request completed, see Selection Result 0x01 = User Selection Request aborted, cancelled by user 0x02 = User Selection Request aborted, timeout
Selection	The menu item selected by the user. This is a single byte zero based binary value.
AdditionalRequestData	Additional key/value pairs of data to be forwarded in the transaction.

Return Value:
The Response output.

```
{
   "ResponseOutput":    {
      "ARQCData":,
      "BATCHData":,
      "UserSelection":,
      "CommandResult":
   },
   "AdditionalOutputData": null
}
```

3.8 RequestSendAcquirerResponse (EMV devices)

Sends the result of an on-line processing decision from the acquirer to the device. The acquirer response will usually contain but not all inclusive an ARPC, Script 1, and Script 2 data.
Using Method POST:

```
api/mtscrahost/RequestSendAcquirerResponse(
string DeviceID,
int WaitTime,
int ConnectionType,
string IssuerAuthenticationData,
string IssuerScriptTemplate1,
string IssuerScriptTemplate2,
string DeviceSerialNumber,
int ApprovalStatus,
string[] AdditionalRequestData);
```

Parameter	Description
DeviceID	URI of the device. For USB devices, use an empty string to open the first device found. Call requestDeviceList to retrieve an array of devices detected on the host based on connection type. Example HID device: "DeviceID": "\\\\?\\hid#vid_0801&pid_0011#7&23521b27&0&0000#{4d1e55b2-f16f-11cf-88cb-001111000030}"
WaitTime	Time the device will wait for the user to complete a card swipe in seconds. (1 - 255)
ConnectionType	Device connection type: 1 = Audio 2 = BLE 3 = BLEEMV 4 = USB
IssuerAuthenticationData	Response data returned from Acquirer to send to the device to complete the transaction.
IssuerScriptTemplate1	Issuer script template 1 from acquirer.

IssuerScriptTemplate2	Issuer script template 2 from acquirer.
DeviceSerialNumber	The device serial number used in constructing of the ARQC response message to be send to the device. See **ARQC Response (from online processing)**.
AdditionalRequestData	Additional key/value pairs of data to be forwarded in the transaction.

Return Value:
The Response output. BATCHData will contain the device response for the transaction.

```
{
   "ResponseOutput":    {
      "ARQCData":,
      "BATCHData":,
      "UserSelection":,
      "CommandResult":
   },
   "AdditionalOutputData": null
}
```

3.9 ReleaseDevice

Closes the connection to the device.
Using Method GET:

```
api/mtscrahost/ReleaseDevice
```

Return Value:
None

Appendix A MTSCRA WEB API Response Output Structures

The MTSCRA WEB API returns the following outputs.

A.1 CheckHealth Output

Returned after checkHealth. A string array containing API name and status.

Example:

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

A.2 DeviceList Output

Returned after requestDeviceList. The output contains a string array containing key/value pairs of devices detected on the host based on connection type.

Example:

```
{
   "DeviceList":    [
      {
      "Key": "MagTek SCRA 1",
      "Value":
"\\\\?\\hid#vid_0801&pid_0011#7&23521b27&0&0000#{4d1e55b2-f16f-11cf-
88cb-001111000030}"
       },

      {
      "Key": "MagTek SCRA 2",
      "Value":
"\\\\?\\hid#vid_0801&pid_0011#7&360bcbcc&0&0000#{4d1e55b2-f16f-11cf-
88cb-001111000030}"
      }
   ],
   "AdditionalOutputData": null
```

A.3 CardSwipe Output

Returned after a card swipe.

Example :

```
{
   "CardSwipeOutput":    {
      "DeviceResponseData": {"Data": "000b323130343238343430475a31"},
      "BatteryLevel": 100,
      "CapMagnePrint": "",
      "CapMagnePrintEncryption": "",
      "CapMagnePrint20Encryption": "00",
      "CapMagneStripreEncryption": "1",
      "CapMSR": "1",
      "CapTracks": "95",
```

```
      "CardDataCRC": 0,
      "CardEncodeType": "00",
      "CardExpDate": "1610",
      "CardIIN": "401900",
      "CardLast4": "3000",
      "CardName": "FIREMAN/TEST D            ",
      "CardPANLength": 19,
      "CardServiceCode": "000",
      "ResponseData":
"0000005028700057ce5f1481bfe8ccbbceab5dcc74511e0f0c84f6ca051604e6bc340
0511e100b9887d2113f0582193673c63a2bd45d5b89454f458b82b2f80ed884815840f
6c0b9982d8168c894cf30ffa8dc4f0fa97b000000000000000000000000000000000000
00000000000000000000000000000a03ff60ffe786c816088c3db8632efe42cd384f72
ed3585686864af8bd1e90e85afa745d947811cd000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000a03ff60ffe786c816088c3db863
2efe45a014416b78887affb00c6369fd7f3c6e8adff595804382c4c491f41ed1bc0786
cb3c48500c004653ea069006a622a5104f889fbccb5de9261ff049721e8e58e6f81649
6f87e7124434f0037f180bf9822c2669ec5dba7d93084c2019d45e96e0061403000389
93ce51ee0f3405c334c6a260b1d170e3e306179090a53f626f1d2a43a330470d0b5092
2e3abb39f89ffe5e802e54a8d84963a01584cd3e200000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000042323843304443313031353134
441410000069010010b28c0dc001e434e276a2542343031393030303030303030303
0333030305e464952454d414e2f5445535420442020202020202020202020205e31363
130303030303030303030303030303030303030303030303030303f00000000000000000
000000000000000000000000000000000000000000000000003b34303139303030303
030303030303033303030303d313631303030303030303030303030303030303f00000000000
000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000003b343
031393030303030303030303030333030303d303030303030303030303030303030303
030303030303030303030303030303030303030303030303030303030303030303030303
030303030303030303030303030303030303030303030303030303030303030303f000
000000000da8934e3df64d7074e276a36ffffff5630350000000000080776bf25e7dd6e
1537efa1cb87e934052d66ae7",
      "DataFieldCount": 0,
      "DeviceConfig": null,
      "DeviceName": "",
      "DeviceSerial": "B28C0DC101514AA",
      "EncryptionStatus": "0006",
      "Firmware": "21042840GZ1",
      "HashCode": "",
      "KSN": "9010010b28c0dc001e43",
      "MagnePrint":
"993ce51ee0f3405c334c6a260b1d170e3e306179090a53f626f1d2a43a330470d0b50
922e3abb39f89ffe5e802e54a8d84963a01584cd3e2",
      "MagnePrintStatus": "61403000",
      "MagTekDeviceSerial": "",
      "MaskedTracks": "%B401900000000003000^FIREMAN/TEST D
^161000000000000000000000000?;401900000000003000=16100000000000000?;
```

```
4019000000000003000=00000000000000000000000000000000000000000000000000
00000000000000000000000000000000?",
      "ResponseType": "",
      "SwipeCount": -1,
      "TLVVersion": "",
      "Track1":
"57ce5f1481bfe8ccbbceab5dcc74511e0f0c84f6ca051604e6bc3400511e100b9887d
2113f0582193673c63a2bd45d5b89454f458b82b2f80ed884815840f6c0b9982d8168c
894cf30ffa8dc4f0fa97b",
      "Track1Masked": "%B4019000000000003000^FIREMAN/TEST D
^161000000000000000000000000?",
      "Track2":
"a03ff60ffe786c816088c3db8632efe42cd384f72ed3585686864af8bd1e90e85afa7
45d947811cd",
      "Track2Masked": ";4019000000000003000=16100000000000000?",
      "Track3":
"a03ff60ffe786c816088c3db8632efe45a014416b78887affb00c6369fd7f3c6e8adf
f595804382c4c491f41ed1bc0786cb3c48500c004653ea069006a622a5104f889fbccb
5de9261ff049721e8e58e6f816496f87e7124434f0037f180bf9822c2669ec5dba7d93
084c2019d45e96e",
      "Track3Masked":
";4019000000000003000=00000000000000000000000000000000000000000000000000
00000000000000000000000000000000?",
      "TrackDecodeStatus": "000000"
   },
   "AdditionalOutputData": null
}
```


A.4 Response Output

Returned after sending a SendCommand or RequestSmartCard. The Data output contains the response report sent from the device. SET returns the ACK status. GET returns the report data in response to the requested command.

A.4.1 Response output for RequestSendCommand

Example of output:


```
{
   "ResponseOutput": {"Data": " 000F42333036463936303131343136414141"},
   "AdditionalOutputData": null
}
```


A.4.2 Response output for RequestSendExtendedCommand

Example of output:


```
{
   "ResponseOutput": {"Data":
"000001A801A6FA8201A2DFDF2101105F2A0208405F360102DFDF5B010CDFDF5C0102D
FDF75010CDFDF76010CDFDF6E010C9F150230309F160730303030303030309F1A0208409
F1C0831313232333334349F33032028C89F3501219F4005720000B0019F4E073030303
0303030DFDF0109A000000004F8001000DFDF025E9ADFDF289F025A899F109F159F169
F4E828E5F245F259F069F079F0D9F0E9F0F9F269F279F369C9F339F349F379F399F409
59B9F5BDFDF009F1E9F1A5F2A9F019F218ADF8120DF8121DF81225F20505F34849F039
```

```
F099F1E9F359F419F53DFDF05299A829F369F1E9F109F5B9F339F35959F015F245A5F3
48A9F159F169F399F1A9F1C579F025F2A9F219CDFDF06028A91DFDF140400007530DFD
F150400000001DFDF160400000080DFDF17599ADFDF289F029F035A899F109F159F169
F4E828E5F245F259F069F079F0D9F0E9F0F9F269F279C9F339F349F359F369F379F399
F409F419F53959B9F5BDFDF009F1E9F1A5F2A9F018ADF8120DF8121DF81225F205F349
F0984DFDF20024328DFDF260E4D414754454B2044454641554C549F3C0209989F3D010
2DFDF4704BD828BA1"},
   "AdditionalOutputData": null
}
```


A.4.3 Response output for RequestSmartCard

Example of output:


```
{
   "ResponseOutput":    {
      "ARQCData": "",
      "BATCHData": "",
      "UserSelection":
"010A53656C656374206C616E67756167653A00656E00646500",
      "CommandResult": "00000000"
   },
   "AdditionalOutputData": null
}
```


A.4.4 Response output for RequestUserSelection

Example of output:


```
{
   "ResponseOutput":    {
      "ARQCData":
"0260FD82025CDFDF250F42333036463936303131343136414FA8202459A031602109
F0206000000000001009F100706010A03A099389F150230309F160730303030303030309F4
E07303030303030303082025C008E0E0000000000000000001E0302031F005F24032212315
F25030907019F0607A00000000310109F0702FF009F0D05F0400088009F0E050010000
0009F0F05F0400098009F260845BB7A3E191E87069F2701809F360205029C01009F330
32028C89F34031E03009F3704072434229F3901059F4005720000B0019505428000800
09B02E8009F1E0842333036463936209F1A0208405F2A0208409F01060000000000019
F2103081933DF812005D84000A800DF8121050010000000DF812205D84004F8005F201
A56495341204143515549524552205445535 42F43415244203032500B5649534120435
2454449545F3401018407A00000000310109F03060000000000009F090200209F1E084
2333036463936209F3501219F410400005461DFDF4D273B34343237303030303039303
03030303333333373D32323132323031303030303030303030303FDFDF520105F88200B
EDFDF560A9010010B306F960017D5DFDF570100FA8200A7DF311850FC230EFCD9B5A62
EC188EDAAD7DC71DA83E2212821A843DF3210CD347ED47485328EAEFB7273E0873ED8D
F3518A397F6EB99EA629CA800E0BEFDC96FE23492EF946523CFBFDF3B3069057482D6F
9CFC1B165915E36DED6A6B2AF5110B891D48266DBC23158647825FF236E7D37A4BD113
AB262E7AEB3E7C6DF4028EA1E2020E0506B364CC748FD42D351FBF2B1C86CD795FE5C7
A5E3976B55071214A6DEF21F423634D",
      "BATCHData": "",
      "UserSelection": "",
      "CommandResult": "00000000"
   },
   "AdditionalOutputData": null
```

```
}
```

A.4.5 Response output for RequestSendAcquirerResponse

Example of output:


```
{
   "ResponseOutput":    {
      "ARQCData": null,
      "BATCHData":
"0102DCFE8202D8DFDF250F42333036463936303131343136414 1FA8202C1F08202BDF
182000FDFDF1A0100DFDF1B0100DFDF520105F28201419A031602109F0206000000000
1009F0306000000000009F100706010A036899389F150230309F16073030303030303
09F4E07303030303030303082025C008E0E0000000000000001E0302031F005F2403221
2315F25030907019F0607A00000000310109F0702FF009F0D05F0400088009F0E05001
00000009F0F05F0400098009F2608F0FA4C936B036E849F2701409C01009F33032028C
89F34031E03009F3501219F360205029F3704072434229F3901059F4005720000B0019
F4104000054619505428000805 09B02FC009F5B0511303030319F1E084233303646393
6209F1A0208405F2A0208409F01060000000000018A023030DF812005D84000A800DF8
121050010000000DF812205D84004F8005F201A5649534120414351554952455220544
553542F434152442030325F3401019F090200208407A0000000031010F782009F5F250
30907015F24032212315F2A0208409F02060000000001009F0306000000000009F060
7A00000000310109F120F4352454449544F20444520564953419F1C083131323233333
4349F3901059C01009F34031E03005F201A5649534120414351554952455220544553 5
42F43415244203032DFDF4D273B34343237303030303039303030303033333373D323
2313232303130303030303030303030303FF88200BEDFDF560A9010010B306F960017D5D
FDF570100FA8200A7DF311850FC230EFCD9B5A62EC188EDAAD7DC71DA83E2212821A84
3DF3210CD347ED47485328EAEFB7273E0873ED8DF3518A397F6EB99EA629CA800E0BEF
DC96FE23492EF946523CFBFDF3B3069057482D6F9CFC1B165915E36DED6A6B2AF5110B
891D48266DBC23158647825FF236E7D37A4BD113AB262E7AEB3E7C6DF4028EA1E2020E
0506B364CC748FD42D351FBF2B1C86CD795FE5C7A5E3976B55071214A6DEF21F423634
D",
      "UserSelection": null,
      "CommandResult": ""
   },
   "AdditionalOutputData": null
}
```

Appendix B ARQC Message Format

This section gives the format of the ARQC Message delivered in the ARQC Message notification from the device. The output is controlled by Property 0x68 – EMV Message Format. There are currently 2 selectable formats: Original and DynaPro. It is a TLV object with the following contents.

Original Format:

```
FD<len>/* container for generic data */
       DFDF25(IFD Serial Number)<len><val>
       FA<len>/* container for generic data */
              <tags defined by DFDF02 >
                . Note: Sensitive Data cannot be defined in DFDF02
                .
              DFDF4D(Masked T2 ICC Data)
              DFDF52 - Card Type Used
              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>
                            DF40(Encrypted Value only of DFDF4A, T2 ISO Format)<len><val>
```

DynaPro Format:

```
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)
```

The Value inside tag DFDF59 is encrypted and contains the following after decryption:

```
              FC<len>/* container for encrypted generic data */
                       <tags defined by DFDF02 >
                     .
                     .
```

Appendix C 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 device. The output is controlled by Property 0x68 – EMV Message Format. There are currently 2 selectable formats: Original and DynaPro. It is a TLV object with the following contents.

Original format:

```
F9<len>/* container for ARQC Response data */
       DFDF25 (IFD Serial Number)<len><val>
       FA<len>/* Container for generic data */
              70<len>/* Container for ARQC */
              8A<len> approval
              Further objects as needed...
```

DynaPro format:

```
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)
```

Appendix D Transaction Result Message – Batch Data Format

This section gives the format of the data the device uses to do completion processing. The output is controlled by Property 0x68 – EMV Message Format. There are currently 2 selectable formats: Original and DynaPro. It is a TLV object with the following contents.

Original Format:

```
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 */
                        DFDF1A - Transaction Status (See DFDF1A descriptions)
                        DFDF1B - Additional Transaction Information (always 0)
                        DFDF52 - Card Type Used

                  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>
                        DF40(Encrypted Value only of DFDF4A, T2 ISO
                        Format)<len><val>
```

DynaPro Format:

```
F9<len>/* container for MAC structure and generic data */
      DFDF54(MAC KSN)<len><val>
      DFDF55(MAC Encryption Type)<len><val>
      DFDF25(IFD Serial Number)<len><val>
      FA<len>/* container for generic data */
            F0<len>/* Transaction Results */
                  F1<len>/* container for Status Data */
                        … /* Status Data tags */
                  F8<len>/* container tag for encryption */
                        DFDF59(Encrypted Data Primative)<len><Encrypted
                  Data val (Decrypt data to read tags)>
                        DFDF56(Encrypted Transaction Data KSN)<len><val>
                        DFDF57(Encrypted Transaction Data Encryption Type)<val>
```

```
                          DFDF58(# of bytes of padding in DFDF59)<len><val>
                    F7<len>/* container for Merchant Data */
                          … /* < Merchant Data tags */
(Buffer if any to be a multiple of 8 bytes)
CBC-MAC (4 bytes, always set to zeroes)
```

DFDF1A Transaction Status Return Codes

0x00 = Approved
0x01 = Declined
0x02 = Error
0x10 = Cancelled by Host
0x1E = Manual Selection Cancelled by Host
0x1F = Manual Selection Timeout
0x21 = Waiting for Card Cancelled by Host
0x22 = Waiting for Card Timeout
0x23 = Cancelled by Card Swipe
0xFF = Unknown