Magensa QwickPIN

Generation/Verification of PIN Offset

Programmer's Reference Manual

March 18, 2021

Document Number:
D998200466-11

REGISTERED TO ISO 9001:2015

Table 0-1 - Revisions

Rev Number	Date	Notes
10	March 2021	Initial Release. This version of document is compatible with QwickPIN API version 1.2.0.1
11	March 2021	Data

Table of Contents

1 Introduction

The purpose of this document is to describe the main operations available in Magensa QwickPIN Web service and their required & optional input/output parameters. It also provides sample REST requests & responses as reference for client developers. It also includes error codes, reasons, and a sample error response.

QwickPIN Web service enables clients to easily generate & verify the PIN Offset via Basic Authentication if required fields are inputted correctly.

QwickPIN Web Service has IP Whitelist feature so it will check for specific IP addresses and rejects any incoming data that does not originate from one of the authorized IP addresses. Please refer to chapter 4 for more detail information.

QwickPIN Web service end point provides swagger documentation and swagger.json which can be imported to Postman program. Please refer to Appendix C for how to import swagger.json to Postman.

This service comprises of two APIs, one is for generation of PIN Offset and the other one is for verification of PIN Offset. The input format for Verification API is the same as Generation API with an extra input for "RefPINOffset" which will be used for verification process. RefPINOffset is a required field with QwickPIN API ver 1.2.0.1.

Depending on PAN Data Type, the format of PAN Data will be different. PAN Data Type can be one of the following:

- MagtekARQC
- MagTekTrack2Encrypted
- DukptPAN

MagTekARQC is the ARQC data in TLV format generated from MagTek Reader (EMV dip or Fallback MSR Swipe). MagTekTrack2Encrypted is the regular MSR Swipe where Encrypted Track2 and MSR KSN are available. TokenV2 can be used to encrypt track2 for this option. Please refer to Appendix A for how to encrypt track2 with TokenV2 service. DUKPT PAN is a value where a client will encrypt the data (concatenated values of PAN, Exp Date and Service Code) under DUKPT. Exp Date and Service Code, however, are not required for generating/verifying PINOffset. To know more about building PAN Data, please refer to Appendix A in this document.

Each decrypted PAN can be validated by Luhn Algorithm or Modulus 10 algorithm in BIN level. It's up to clients for enabling/disabling it. If clients have multiple BINs, each BIN has Luhn Check option separately.

Input parameter RefID allows a customer to provide a reference ID to link its financial system to Magensa service. It can be a GUID or any other information a customer wants to use.

Input parameter PIN DataType is used for PIN block format. Although the most common PIN Block encoding format is ISO_0 type, QwickPIN supports ISO 0, ISO 1, ISO 2, and ISO 3. For PIN Data, Magensa currently supports 4-digit PIN that is enclosed in the Encrypted PIN Block. PIN Block needs to be encrypted under DUKPT key with pin variant. Conditionally, PIN Data requires a json string which contains both EPB and KSN. PIN Data can also be obtained from TokenV2 when plain PIN block is sent to TokenV2 for encryption. Please refer to Appendix B for how to build plain PIN block and to obtain Encrypted PINBlock.

The authentication for all QwickPIN service calls require an "Authorization" HTTP header set as per HTTP BasicAuthentication scheme. The value should be the Base64 encoding of your Magensa credentials in the string format "CustomerCode/Username:Password". In the sample request packet, base64 encoded value was replaced by {AUTHORIZATION HEADER VALUE}

Please note that in this document, any sensitive data were replaced by invalid card number or other random Hex Number. Especially, PAN and PIN block are sensitive data. Based on PCI policy, even encrypted format considered as sensitive data. Card Number (4444333322221111) and EPB (0123456789ABCDEF) in this example is not real data.

2 QwickPIN Operations

2.1 /api/PINOffset/Generate

2.1.1 INPUT PROPERTIES

Property	Value	Description
panData *	string	Depending on panDataType, the PAN data format will be different. - MagTekARQC: ARQC data in TLV format generated from MagTek Reader. - MagTekTrack2Encrypted: encrypted Track2 by MSR Swipe. - DUKPT PAN: Encrypted Bulk Data format retrieved by using command 0x30 to MagTek Devices. See detail information in Appendix A.
panDataType *	string	Supported Enum types: 0 or MagTekARQC 1 or MagTekTrack2Encrypted 2 or DukptPAN

pinData	string	Conditional field. For "MagTekARQC", pinData is optional. It's required for panDataType "MagTekTrack2Encrypted" or "DukptPAN".
		Json string format should contain both 'epb' and 'ksn'. EPB is encrypted pin block. PIN block needs to be encrypted under DUKPT key with pin variant. The PIN Block Format should be specified in the pinDataType field.
		Note that double quotes for epb and ksn properties are escaped. This is because pinData is actually a string, not a json object.
		Ex)
		{
		:
		`"PINData": " { \"epb\": \"F5D0…..7691\", \"ksn\": \"9A0003….044\" }",`
		:
		}

pinDataType *	string	PIN Block Format. Supported Enum types: 0 or ISO_0 (ISO Format 0, ANSI X9.8, VISA-1 and ECI-0) 1 or ISO_1 (ISO Format 1 and ECI-4) 2 or ISO_2 (ISO Format 2) 3 or ISO_3 (ISO Format 3)
refID	string	Allows a customer to provide a reference ID to link its financial system to Magensa service. It can be a GUID or any other information a customer wants to use.

Note: * = Required

2.1.2 OUTPUT PROPERTIES

Property	Value	Description
pinOffset	string	The generated PINOffset
magTranID	string	Magensa Transaction ID from target service provider

Generate PINOffset Request JSON

```
{
    "panData": "<string>",
    "panDataType": "<string>",
    "pinDataType": "<string>",
    "pinData": "<string>",
    "refID": "<string>"
}
```

Sample /api/PINOffset/Generate Request:

```
POST /QwickPIN/api/PINOffset/Generate HTTP/1.1
Host: devapp.magensa.dev
Content-Type: application/json
Authorization: Basic {AUTHORIZATION HEADER VALUE}
Content-Length: 291
Connection: Keep-Alive

{
    "panData":"002742324630363933303731303135414100089999990
              B2F06930000C50123456789ABCDEF03F7EEF9",
    "panDataType": "DukptPAN",
    "pinData": " { \"epb\": \"0123456789ABCDEF\", \"ksn\":
                 \"9A000300000D93200044\" }",
    "pinDataType": "ISO_0",
    "refID": "11111111-2222-3333-4444-555555555555"
}
```

Sample /api/PINOffset/Generate Response:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Server: Microsoft-IIS/10.0
X-Powered-By: ASP.NET
Date: Thu, 25 Feb 2021 18:05:30 GMT
Content-Length: 71

{
    "pinOffset":"1234",
    "magTranID":"8b9ce6a5-2d5f-4879-8c16-6d769f3cfc93"
}
```

2.2 /api/PINOffset/Verify

2.2.1 INPUT PROPERTIES

Property	Value	Description
panData *	string	Depending on panDataType, the PAN data format will be different. • MagTekARQC: ARQC data in TLV format generated from MagTek Reader. • MagTekTrack2Encrypted: encrypted Track2 by MSR Swipe. • DUKPT PAN: Encrypted Bulk Data format retrieved by using command 0x30 to MagTek Devices. See detail information in Appendix A.
panDataType *	string	Supported Enum types: 0 or MagTekARQC 1 or MagTekTrack2Encrypted 2 or DukptPAN

pinData	string	Conditional field. For panDataType “MagTekARQC”, pinData is optional. Otherwise it’s required.
		Json string format should contain both ‘epb’ and ‘ksn’. EPB is encrypted pin block. PIN block needs to be encrypted under DUKPT key with pin variant.
		The PIN Block Format should be specified in the pinDataType field.
		Note that double quotes for epb and ksn properties are escaped. This is because pinData is actually a string, not a json object.
		Ex)
		{
		:
		"PINData": " { \"epb\": \"F5D0…..7691\", \"ksn\": \"9A0003….044\" }",
		:
		}

pinDataType *	string	PIN Block Format. Supported Enum types: 0 or ISO_0 (ISO Format 0, ANSI X9.8, VISA-1 and ECI-0) 1 or ISO_1 (ISO Format 1 and ECI-4) 2 or ISO_2 (ISO Format 2) 3 or ISO_3 (ISO Format 3)
refID	string	Allows a customer to provide a reference ID to link its financial system to Magensa service. It can be a GUID or any other information a customer wants to use.
refPINOffset *	string	Reference PINOffset. (Required field in QwickPIN API ver 1.2.0.1)

Note: * = Required

2.2.2 OUTPUT PROPERTIES

Property	Value	Description
Success	boolean	true or false. Flag to indicate whether the generated PIN Offset matches with the value of RefPINOffset.
magTranID	string	Magensa Transaction ID from target service provider

Verify PINOffset Request JSON

```
{
    "panData": "<string>",
    "panDataType": "<string>",
    "pinDataType": "<string>",
    "pinData": "<string>",
    "refID": "<string>",
    "refPINOffset": "<string>"
}
```

Sample /api/PINOffset/Verify Request:

```
POST /QwickPIN/api/PINOffset/Generate HTTP/1.1
Host: devapp.magensa.dev
Content-Type: application/json
Authorization: Basic {AUTHORIZATION HEADER VALUE}
Content-Length: 317
Connection: Keep-Alive

{
     "PANData": "002742324630363933303731303135414100089999990
                 B2F06930000C50123456789ABCDEF03F7EEF9",
     "PANDataType": "DukptPAN",
     "PINData": "{ \"epb\": \"0123456789ABCDEF\",
                   \"ksn\": \"9A000300000D93200044\" }",
     "PINDataType": "ISO_0",
     "RefPINOffset" : "1234",
     "RefID": "11111111-2222-3333-4444-555555555555"
}
```

Sample /api/PINOffset/Verify Response:

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Server: Microsoft-IIS/10.0
X-Powered-By: ASP.NET
Date: Thu, 25 Feb 2021 21:34:21 GMT
Content-Length: 67

{
    "success": true,
    "magTranID": "dada8760-28e2-47b1-a7cb-7120509fce70"
}
```

3 Error Codes and Reasons

3.1 Sample Error Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Server: Microsoft-IIS/10.0
X-Powered-By: ASP.NET
Date: Mon, 01 Mar 2021 16:21:11 GMT
Content-Length: 97

{
    "code": "DataError_InvalidKey",
    "message": "Data Error - Invalid Key-BDK is empty for KSI 9C12345"
}
```

3.2 Error Codes and Reasons

Error Code	Reason
2001 (Invalid_ARQC)	In case of MagTekARQC, ARQC in PANData is not valid
2002 (Invalid_Track2)	In case of MagTekTrack2Encrypted, Track2 in PANData is not valid
2003 (Invalid_DukptPAN)	In case of DukptPAN, Dukpt PAN in PANData is not valid
2004 (Invalid_PANDataType)	PANDataType is null or invalid
2005 (Invalid_PINDataType)	PINDataType is null or invalid
2006 (Invalid_PANData)	PANData is null or invalid
2007 (Invalid_PINData)	PINData is null
2008 (Invalid_KSN)	KSN is empty or invalid
2009 (Invalid_PAN)	Primary Account Number is null or invalid including Luhn Check error
2010 (Invalid_ExpDate)	Expiration Date is invalid or length is not 4 digits
2011 (Invalid_ServiceCode)	Service Code is invalid
2012 (Invalid_RefPINOffset)	RefPINOffset is null or invalid
2013 (Invalid_RefPVV)	RefPVV is null or invalid
2014 (Invalid_RefCVV)	RefCVV is null or invalid
2051 (DataError_InvalidPINOffsetSetting)	Setting for PINOffset is not valid.
2052 (DataError_BINNotIdentified)	BIN number related to specific Customer Code and PAN is not available
2053 (DataError_InvalidHostInfo)	Host Template Info is not available
2054 (DataError_InvalidKey)	BDK or PVK is invalid
2061 (Decryption_DataPrimitive)	Decrypted Value is invalid.
2062 (Decryption_Track2)	Decrypted Track2 is invalid
2063 (Decryption_DukptPAN)	Decrypted DukptPAN is invalid
2064 (Decryption_DeviceNotAllowed)	Deactivated device is not allowed for decryption
2071 (CryptoError_HSM)	HSM error during generating/verifying PINOffset
2081 (Host_Request_Error)	RefPINOffset cannot be retrieved from the Client’s host interface
2082 (Host_Update_Error)	Update PIN Offset Info to the Client’s host interface failed
4000 (Auth_Error)	Basic Authentication failed
5000 (Unknown_Error)	Unknown Error.

4 IP Whitelist

In order to access the Magensa QwickPIN Service for generating/verifying PINOffset, the customer must use a public static IP address that has been whitelisted with Magensa. Otherwise, the customer's data will be blocked from ever reaching Magensa servers.

Please note that the customer's IP address must be static, as Magensa services check for specific IP addresses and rejects any incoming data that does not originate from one of the authorized IP addresses. If the customer's public IP address is dynamic, the customer will need to either (a) acquire a static public IP address for their development; or (b) utilize a proxy server that has been assigned a public static IP address.

Appendix A PAN Data

According to PAN Data Type, the format of PAN Data must be changed.

1. **MagTekARQC**
   ARQC data in TLV format generated from MagTek Reader. It can come either from dipping EMV chip card or fallback MSR swipe. Note that PINData field is optional for MagTekARQC since PINData information should be included in the ARQC.

```
{
     "PANData":
"0121F982011DDFDF540A950003000015C920D6B8DFDF550182DFDF250899298
C2611160D10FA8200FA708200F6DFDF5301005F20105445535420322F5341424
15454555320DFDF4D233B5448414E4B53464F524C4F4F4B494E473D323331323
030303030303030303030303FDFDF520105F88200AEDFDF59820090012345678
9ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF012345678
9ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF012345678
9ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF012345678
9ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF012345678
9ABCDEF0123456789ABCDEFDFDF560A950003000015C920D6B8DFDF570180DFD
F5801050000000000009948D9CC",
     "PANDataType": "MagTekARQC",
     "PINDataType":  "ISO_0",
     "RefID": "11111111-2222-3333-4444-555555555555"
}
```

2. **MagTekTrack2Encrypted**

```
{
          "PANData": "{ \"track2\":
\"DD63F29DEC3E0E7D3B46E9EFD57DD918EBB96203C30149A2E28C265C526EC3
9D4951CFBCC489E82F\", \"ksn\": \"9010010000000000026D9\" }",
          "PANDataType": "MagTekTrack2Encrypted",
          "PINData": "{ \"epb\": \"0123456789ABCDEF\", \"ksn\":
\"9A000300000D93200081\" }",
          "PINDataType": "ISO_0",
          "RefID": "11111111-2222-3333-4444-555555555555"
}
```

   A client application needs to pack both encrypted Track2 and MSR KSN as a Json string to be used as PANData. Note that double quotes for track2 and ksn properties are escaped. This is because PANData is a string, not a json object.

   As described in Introduction, this option is useful when TokenV2 is available while a reader is not an option. Basically, what needs to be done is to build a customized track2 data format with card number, expiration date and service code.

Track2 data format for TokenV2 - ;{Card Number}=YYMMSSS000000000000?
If Card # is 4444333322221111, Exp Date (YYMM) is 2203 and Service Code (SSS) 101.
Replace ;{*Card Number*}=*YYMMSSS*000000000000? with 4444333322221111, 2203 and 101
Although Exp Date and Service Code are optional for PIN Offset, they are mandatory for CVV.

So, it's going to be ;4444333322221111=2203101000000000000?
This value needs to be sent to TokenV2 service.

Sample /TokenV2Create/api/Token/create Request:

```
{
  "CustomerTranRef": "x",
  "TokenDataInput": "{\"TokenDataTypeID\": \"1\",\"PlainText\":
\";4444333322221111=2203101000000000000?\"}",
  "ValidUntilUTC": "2050-12-31T00:00:00",
  "TokenName": "PAN Data",
  "MiscData": "Misc Data"
}
```

Sample /TokenV2Create/api/Token/create Response:

```
{
    "magTranID": "b7263356-2648-4bf8-9b2b-9cc237350982",
    "timestamp": "2021-03-15 17:48:22Z",
    "customerTranRef": "X",
    "token":
"FA820137DFE0012436353261363432652D383066642D343435642D613863652
D3665613064346464373833364DFE00205312E322E30DFE0041B323035302D313
22D33315430303A30303A30302E30303030303030DFE0050454657374DFE0061
C323032312D30332D31355431373A34383A32322E383935393731375ADFE0071
0393930303030303030303030303032DFE00807313030383030 31DFE009246
23732363333335362D323634382D346266382D396232622D39636332333733353
0393832DFE01128A0BFCEBDEC31980A031D43F369715D0609D8195A4278AAFDD
F514A589D215626A3B89D6016A00867DFE0120A901001000000000026C0DFE02
10942756C6B2044617461DFE022094D697363204461746 1DFE00320C40FAE100
3BD07B9133574C6B79A243616E4BB20CBC0D1EF9D9BA8F60BF110D9",
    "code": "0",
    "message": "OK"
}
```

The format of Token Data field in the response json is TLV format. The value of DFE011 and DFE012 are needed to get Encrypted PAN Data and KSN.

DFE011 28
A0BFCEBDEC31980A031D43F369715D0609D8195A4278AAFDDF514A589D215626A3B89D6016A00867

DFE012 0A 901001000000000026C0

“track2” data in PANData should be replaced with the data of DFE011 (Note that the length shouldn’t be included) and “ksn” data in PANData should be replaced with the data of DFE012.

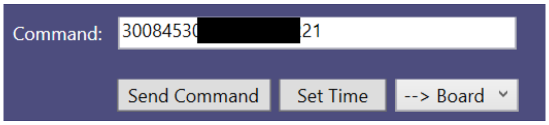
```
{
           "PANData": "{ \"track2\": \"
A0BFCEBDEC31980A031D43F369715D0609D8195A4278AAFDDF514A589D215626
A3B89D6016A00867\", \"ksn\": \"9010010000000000026C0\" }",
           "PANDataType": "MagTekTrack2Encrypted",
           "PINData": "{ \"epb\": \"0123456789ABCDEF\", \"ksn\":
\"90100100000000002647\" }",
           "PINDataType": "ISO_1",
           "RefID": "11111111-2222-3333-4444-555555555555"
}
```

```
{
    "pinOffset": "1234",
    "magTranID": "97f0fe1c-1541-4106-995c-00d808f95ef6"
}
```

3. **DukptPAN**

Client will need to encrypt PAN, Expiration Date and Service Code under DUKPT key using MagTek Reader’s Bulk Data Encryption (0x30) functionality. However, for PINOffset, expiration date and service code are optional.

For example, if a card #, 453036XXXXXX2221 is entered without expiry date and SVC, MTSCRA OEM software returns the value, 0027423246303639333037313031354141000899999990B2F06930000C50123456789ABCDEF03F7EEF9. Note that there is command (30) and length in the front of PAN



Command: 3008453[redacted]21

This value should be sent as PANData to QwickPIN Service. See the following example.

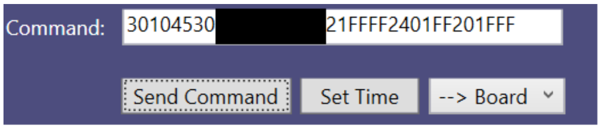
```
{
    "PANData":"0027423246303639333037313031354141000899999990
              B2F06930000C50123456789ABCDEF03F7EEF9",
    "PANDataType": "DukptPAN",
    "PINData": " { \"epb\": \"0123456789ABCDEF\", \"ksn\":
                   \"9A000300000D93200044\" }",
    "PINDataType": "ISO_0",
    "RefID": "11111111-2222-3333-4444-555555555555"
}
```

If Exp Date and Service Code is not needed (e.g Verifying/Generating PINOffset),
CMD (30) + LEN (08) + PAN (45 30 36 XX XX XX 22 21)

So, input data for encryption will be 30 08 45 30 36 XX XX XX 22 21

If Exp Date and Service Code are added,
CMD (30) + LEN (10) +
PAN (20digits, padded with ‘F’) + YYMM + ‘FF’ + SVC (3 digits) + ‘FFF’.

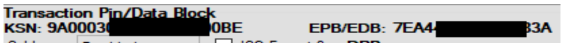
Input date for encryption will be 30 10 45 30 36 XX XX XX 22 21 FF FF 24 01 FF 20 1F FF



Command: 30104530[redacted]21FFFF2401FF201FFF

Appendix B PIN Data

EPB and KSN for PIN Data can be obtained from each reader where a key has been injected into DUKPT management system. The reader will calculate EPB and return it with KSN after 4-digit pin is entered into the reader. For example, the sample was obtained from DynaPro.



Transaction Pin/Data Block
KSN: 9A0003[redacted]0BE EPB/EDB: 7EA4[redacted]3A

When TokenV2 service is available, EPB and KSN can be obtained from TokenV2, too. Here is how to build plain Pin block for TokenV2.

1. Choose ISO format Type.

   QwickPIN supports ISO_0, ISO 1, ISO 2 and ISO 3. When ISO format type is selected, make sure you should know how to build plain PIN Block associated with the selected ISO format.

2. Let's assume ISO format 1 is selected. ISO format 1 is usually used where there is no PAN to associated with PIN.

   L is length of the PIN, P is PIN digit, F is padding value "F", T is format type (=1)

T	L	P	P	P	P	P/F	P/F	P/F	P/F	P/F	P/F	P/F	P/F	F	F

   PIN is 1234.

   L is 4, P = 1234, T = 1

1	4	1	2	3	4	F	F	F	F	F	F	F	F	F	F

   If, however, random padding is used instead of Fs, it will produce a unique encrypted PIN Block.

1	4	1	2	3	4	8	F	5	E	B	8	2	9	7	4

3. Send this value to TokenV2 service.

   Sample /TokenV2Create/api/Token/create Request:

```
{
  "CustomerTranRef": "x",
  "TokenDataInput": "{\"TokenDataTypeID\": \"2\",\"HexString\":
\"1412348F5EB82974\"}",
  "ValidUntilUTC": "2050-12-31T00:00:00",
  "TokenName": "PIN Data",
  "MiscData": "Misc Data"
}
```

Sample /TokenV2Create/api/Token/create Response:

```
{
    "magTranID": "73dd8c09-8fd7-4401-9c23-5168132bd340",
    "timestamp": "2021-03-15 18:56:25Z",
    "customerTranRef": "x",
    "token":
"FA82011DDFE0012434333933353338302D323665382D343265632D386639342
D306534353366346666313130DFE00205312E322E30DFE0041B323035302D313
22D33315430303A30303A30302E30303030303030DFE0050454657374DFE0061
C323032312D30332D31355431383A35363A32362E303131373336385ADFE0071
03939303030303030303030303030303032DFE0080731303038303031DFE009243
7336464386330392D386664372D343430312D396332332D35313638313332626
4333430DFE011080123456789ABCDEFDFE01303484558DFE0120A90100100000
00000264CDFE0210850494E2044617461DFE022094D6973632044617461DFE00
320856B65361FDEC3FCCAC767F31DDF0E755E482456036C7C1C1B9338A796670
012",
    "code": "0",
    "message": "OK"}
```

The format of Token Data field in the response json is TLV format. The value of DFE011 and DFE012 are needed to get Encrypted PAN Data and KSN.

DFE011 08 0123456789ABCDEF

DFE012 0A 90100100000000000264C

Sample /api/PINOffset/Generate Request:

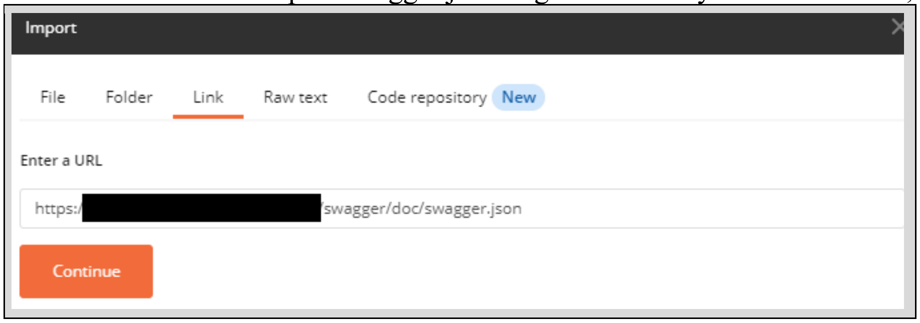
```
{
            "PANData": "{ \"track2\":
\"60017B0C2191FD2868BDB2C9220430BE076ABF96B4687050C198CD75654EDF
1C734EFE5DBEA76132\", \"ksn\": \"90100100000000000264B\" }",
            "PANDataType": "MagTekTrack2Encrypted",
            "PINData": "{ \"epb\": \"0123456789ABCDEF\", \"ksn\":
\"90100100000000000264C\" }",
            "PINDataType": "ISO_1",
            "RefID": "11111111-2222-3333-4444-555555555555"
}
```

Sample /api/PINOffset/Generate Response:

```
{
    "pinOffset": "1234",
    "magTranID": "97f0fe1c-1541-4106-995c-00d808f95ef6"
}
```

Appendix C Postman

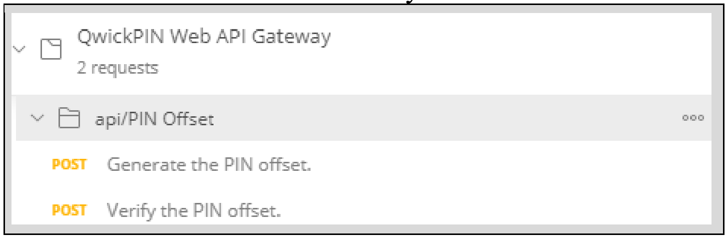
If Postman is used to import swagger.json to generate/verify PINOffset APIs,



Enter a URL

https:/[redacted]/swagger/doc/swagger.json

Two POST APIs will be added to your Postman.


QwickPIN Web API Gateway

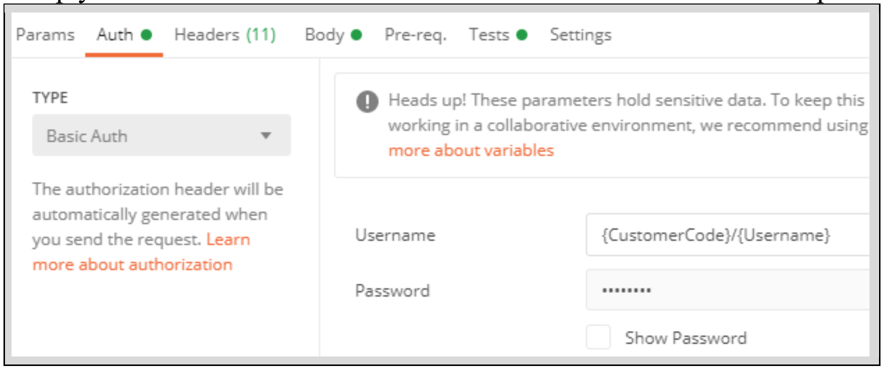
2 requests

api/PIN Offset

POST Generate the PIN offset.

POST Verify the PIN offset.

Setup your Basic Auth with CustomerCode/Username and Password provided by MagTek.



TYPE

Basic Auth

The authorization header will be automatically generated when you send the request. Learn more about authorization

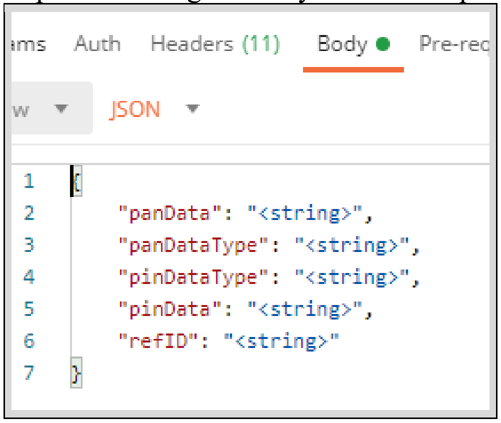
Heads up! These parameters hold sensitive data. To keep this working in a collaborative environment, we recommend using more about variables

Username {CustomerCode}/{Username}

Password ••••••••

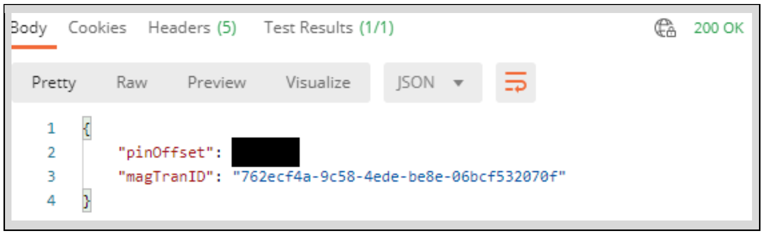
□ Show Password

Replace "<string>" with your JSON input strings. Refer to the samples Chapter #2.



```
{
    "panData": "<string>",
    "panDataType": "<string>",
    "pinDataType": "<string>",
    "pinData": "<string>",
    "refID": "<string>"
}
```

And the pinoffset value is sent from QwickPIN Service.



```
{
    "pinOffset": [redacted]
    "magTranID": "762ecf4a-9c58-4ede-be8e-06bcf532070f"
}
```

Appendix D Glossary of Terms

Term	Description
ARQC	Authorization Request Cryptogram. Each EMV transaction request is supposed to contain ARQC, which is a cryptogram generated from the transaction data. This is generated by the card after taking some values from the reader.
BDK	In DUKPT each device is still initialized with a distinct key, but all the initialization keys of an entire family of devices are derived from a single key, BDK
BIN	Bank Identification Number – The BIN is the first 6-8 digits of the PAN and is used to identify the issuer of the card.
DUKPT	Derived Unique Key per Transaction. It's a key management scheme in which for every transaction, a unique key is used which is derived from a fixed key.
EPB	Encrypted PIN Block. Encrypted under DUKPT key with data variant
KSN	KSN is returned from the encrypting device, along with the cryptogram. The KSN is formed from the device's unique identifier, and an internal transaction counter.
Luhn Algorithm	A checksum formula used to validate a variety of ID numbers including credit card numbers. If a number is misread, Luhn's algorithm will validate this number.
PAN	Primary Account Number – A 13 to 19 digits number used to identify a debit card cardholder or a credit account number.
PIN	Personal Identification Number – A 4-12 digits string of numbers entered by the cardholder to provide cardholder verification. Magensa supports 4-digit PIN
PIN Block	Personal Identification Number Block – When a cardholder enters his/her PIN, the information is first encoded into a plain text PIN block using one of several PIN block format defined. The plain text PIN block can be encrypted using a standard algorithm
PIN Offset	PIN offset is a value that is the difference between two PINs. For example, a PIN Offset may be the difference between a PIN that is chosen by the customer and on that is assigned by an institution.
PVK	A Key used for PIN Offset Algorithm.
SERVICE CODE	3-digit code that follows the expiry date on the card's Track2 magnetic stripe.
TOKENV2	TokenExchange v2.0 enables clients to easily create & redeem secure tokens for various sensitive data. This helps mitigate the risk of storing and use of sensitive data through industry-standard encryption algorithms and implementations.

Appendix E References

1. https://www.ibm.com/
2. https://en.wikipedia.org/
3. EMVCo, LLC - EMV Kernel books
4. TSYS - EMV Implementation guide.
5. Magensa, LLC, PIN Check Web API – Software Design Document v1.0.0
6. PCI Security Standards Council - Official PCI Security Standards Council Site - Verify PCI Compliance, Download Data Security and Credit Card Security Standards
7. MagTek, MAGNETIC STRIPE CARD STANDARDS
8. Magensa, LLC, TokenV2 Programmer's Reference Manual
9. Magensa, LLC, Magensa Tokenization Service Programmer's Reference Manual
10. Luhn algorithm - Wikipedia