EMV$^{®}$
Level 3 (L3) Testing Framework

Pseudo-function Definitions for Test Card Images

Version 1.6

March 2023

Legal Notice

This document summarizes EMVCo's present plans for evaluation services and related policies and is subject to change by EMVCo at any time. This document does not create any binding obligations upon EMVCo or any third party regarding the subject matter of this document, which obligations will exist, if at all, only to the extent set forth in separate written agreements executed by EMVCo or such third parties. In the absence of such a written agreement, no product provider, test laboratory or any other third party should rely on this document, and EMVCo shall not be liable for any such reliance.

No product provider, test laboratory or other third party may refer to a product, service or facility as EMVCo approved, in form or in substance, nor otherwise state or imply that EMVCo (or any agent of EMVCo) has in whole or part approved a product provider, test laboratory or other third party or its products, services, or facilities, except to the extent and subject to the terms, conditions and restrictions expressly set forth in a written agreement with EMVCo, or in an approval letter, compliance certificate or similar document issued by EMVCo. All other references to EMVCo approval are strictly prohibited by EMVCo.

Under no circumstances should EMVCo approvals, when granted, be construed to imply any endorsement or warranty regarding the security, functionality, quality, or performance of any particular product or service, and no party shall state or imply anything to the contrary. EMVCo specifically disclaims any and all representations and warranties with respect to products that have received evaluations or approvals, and to the evaluation process generally, including, without limitation, any implied warranties of merchantability, fitness for purpose or non-infringement. All warranties, rights and remedies relating to products and services that have undergone evaluation by EMVCo are provided solely by the parties selling or otherwise providing such products or services, and not by EMVCo, and EMVCo will have no liability whatsoever in connection with such products and services.

This document is provided "AS IS" without warranties of any kind, and EMVCo neither assumes nor accepts any liability for any errors or omissions contained in this document. EMVCO DISCLAIMS ALL REPRESENTATIONS AND WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AS TO THIS DOCUMENT.

EMVCo makes no representations or warranties with respect to intellectual property rights of any third parties in or in relation to this document. EMVCo undertakes no responsibility to determine whether any implementation of this document may violate, infringe, or otherwise exercise the patent, copyright, trademark, trade secret, know-how, or other intellectual property rights of third parties, and thus any person who implements any part of this document should consult an intellectual property attorney before any such implementation.

Without limiting the foregoing, this document may provide for the use of public key encryption and other technology, which may be the subject matter of patents in several countries. Any party seeking to implement this document is solely responsible for determining whether its activities require a license to any such technology, including for patents on public key encryption technology. EMVCo shall not be liable under any theory for any party's infringement of any intellectual property rights in connection with this document.

Revision Log

Version	Date	Description
V1.0	March 29th, 2017	First public release of document
V1.1	March, 08th, 2018 May, 03rd, 2018	• Update “Location” information for Pseudo functions that may be located in Format 1 template ID=”80” • Add emvcard.msd() pseudo function • Clarify emvcard.arqc • Use emvcard.cvc3t1() and emvcard.cvc3t2() instead of emvcard.cvc3() twice • Add emvcard.dcvv() • Add emvcard.cpr() • Prescribe the use of an initial value for emvcard.ctq() and emvcard.cpr()
V1.2	June, 12th, 2018	• Add specification for emvcard.sdad() pseudo function • Locate in the document all the definitions from the MP50 Card Image
V1.3	Nov, 30th, 2018	• Add emvcard.track2() • Update emvcard.ctq() and emvcard.cpr(). Precise Initial value calculation • Update emvcard.UN() definition • Update emvcard.atc() definition
V1.4	April 11th, 2019	• Update following TA review
V1.5	October, 2020	• Updated functions for Gen AC which can also apply to contactless GPO • Added new functions: emvcard.cert(fn); emvcard.country(fn, c1, c2); emvcard.currency(fn, c1, c2); emvcard.onlineonly() • Updated: emvcard.arqc(); emvcard.aac(); emvcard.term(); emvcard.auth(); emvcard.appcrypto(); emvcard.sdad(*format*); emvcard.iad (*length*); emvcard.ctq(*initvalue*); emvcard.cpr(*initvalue*); emvcard.UN(length);
V1.6	March, 2023	• Add or modify functions to accommodate C8 and XDA functionality • Added new functions: emvcard.CardKeyData(); emvcard.EDAMAC(); emvcard.AC(); emvcard.CVD() and emvcard.cardTVR(*value*). • Modified function: emvcard.atc(), emvcard.sdad() and emvcard.term() • Removed emvcard.dcvv()

Contents

1 Executive Summary

EMVCo has defined guidelines and specifications that collectively help to facilitate a standardized approach to the development, qualification and usage of test tools required by financial institution clients and their service providers to perform Level 3 terminal integration testing on EMV contact and contactless acceptance devices. A key component of these deliverables is the *EMVCo Level 3 Testing Framework – Implementation Guidelines [L3FIG]* that provides its targeted audiences with specific implementation details and instructions for creating and supporting various technical components of the *L3 Framework*. These components include:

- Machine-readable L3 Test Card image format - representing the expected card behaviours for each set of Payment System test card images.
- Test Set files for the Test Selection Engine (TSE) component – defining the methodology for test case selection, questions to be asked by the TSE, errors to be reported by the TSE in exception cases, Pass/Fail Criteria definitions, etc.
- Test Session file – generated by the TSE following user entry, and used by the TSE to provide instruction to the Test Tool engine on which cases are to be executed.
- Test Reporting and Logging formats – including the Card to Terminal Log and the Online Message Log

Within the machine-readable L3 Test Card image format, specified in Extensible Mark-up Language (XML), various pseudo-functions are used to address card images behaviours that cannot be easily deduced from the image content alone. This document, the *EMVCo Level 3 Testing Framework – Pseudo-function Definitions for Test Card Images [L3PSEU],* includes the list of currently defined pseudo-functions. It is a companion document to the *EMVCo Level 3 Testing Framework – Implementation Guidelines [L3FIG]*.

2 Purpose and Scope

Pseudo-function definitions may be required either to specify card characteristics & behaviours that otherwise cannot be deduced from the card image content alone, or to highlight specific areas for usability reasons. The definitions may, for example, include details of the cryptographic functions used by or within a Payment System's products.

Note that in some cases, it may be sufficient just to specify the Cryptogram Version Number (CVN) in the image and not require the use of any pseudo-functions for this capability.

2.1 Purpose

This document, the *EMVCo Level 3 Testing Framework – Pseudo-function Definitions for Test Card Images [L3PSEU],* aims at providing the EMVCo L3 stakeholders the appropriate level of information and directives to prepare and implement the EMVCo L3 Card Image machine-readable files. It is intended to be a companion document to the *EMVCo Level 3 Testing Framework – Implementation Guidelines (L3FIG),* and provides its targeted audiences with a list of the currently defined Pseudo-function definitions.

2.2 Intended Audiences

The intended audiences of this document and its companion, the *EMVCo Level 3 Testing Framework – Implementation Guidelines (L3FIG),* are:

- Payment Systems – preparing the L3 Test Card Images for delivery to L3 Test Tool suppliers
- L3 Test Tool supplier – developing compilers and interpreters within their tools to process L3 Test Card Images delivered by Payment Systems
- Financial Institution clients and their service providers – utilizing L3 test tools to perform L3 terminal integration testing.

3 Pseudo-function Definitions

The table below includes a list of currently defined pseudo-function definitions for use with the Machine-readable Test Card Images, as specified in Section 4.5 of the *EMVCo Level 3 Testing Framework – Implementation Guidelines [L3FIG]* document.

Pseudo function Name	TAG	Description	Length	Format	Location	Reference
simvendor.id()	5F 50	Issuer Uniform resource locator (URL). The URL provides the location of the Issuer's Library Server on the Internet	var	ans	FCI Discretionary Template ID="BF 0C”	Underlying Payment System’s specification
emvcard.arqc()	9F 27	Cryptogram Information Data (CID). GenAC1: Signifies ARQC in case the terminal does not request an AAC and is not offline only or terminal type is not known TC in case the terminal does not request an AAC and is known to be offline only AAC in case the terminal requests an AAC Contactless GPO: AAC if the terminal is offline-only, otherwise ARQC	1	Binary	GenAC Response Message Template Format 1 ID="80" or Format 2 ID="77" Contactless GPO Response Message Template Format 2 ID="77"	EMV Book 3
emvcard.onlineonly()	9F27	Cryptogram Information Data (CID). GenAC1: Signifies ARQC in case the terminal does not request an AAC	1	Binary	GenAC Response Message Template Format 1 ID="80" or Format 2 ID="77"	Underlying Payment System’s specification

Pseudo function Name	TAG	Description	Length	Format	Location	Reference
		AAC in case the terminal requests an AAC Contactless GPO: AAC if the terminal is offline-only, otherwise ARQC			Contactless GPO Response Message Template Format 2 ID="77"	
emvcard.aac()	9F 27	Cryptogram Information Data (CID). GenAC1: Signifies AAC always GenAC2: Signifies AAC always GPO: Signifies AAC always	1	Binary	GenAC Response Message Template Format 1 ID="80" or Format 2 ID="77" Contactless GPO Response Message Template Format 2 ID="77"	EMV Book 3
emvcard.term()	9F 27	Cryptogram Information Data (CID). GenAC1: Signifies follow the terminal request GenAC2: Signifies follow the terminal request and don’t check issuer auth result or presence of Issuer Auth Data	1	Binary	GenAC Response Message Template Format 1 ID=”80” or Format 2 ID="77" Contactless GPO Response Message Template Format 2 ID="77"	EMV Book 3
emvcard.auth()	9F 27	Cryptogram Information Data (CID). GenAC2: Signifies follow the terminal request but decline if Issuer auth failed or Issuer Auth Data (Tag 91) is not present in External Authenticate or Gen AC command	1	Binary	GenAC Response Message Template Format 1 ID="80" or Format 2 ID="77"	EMV Book 3

Pseudo function Name	TAG	Description	Length	Format	Location	Reference
emvcard.atc()	9F 36	Application Transaction Counter (ATC) maintained by the L3 CS. - emvcard.atc(): Any value	2	Binary	GenAC Response Message Template Format 1 ID="80" or Format 2 ID="77" Contactless GPO Response Message Template Format 2 ID=”77”	Not applicable
emvcard.appcrypto()	9F 26	Application Cryptogram. Cryptogram always returned by the card in response to the GENERATE AC or Contactless GPO command (even if the terminal required CDA)	8	Binary	GenAC Response Message Template Format 1 ID="80" or Format 2 ID="77" Contactless GPO Response Message Template Format 2 ID="77"	Underlying Payment System’s specification

Pseudo function Name	TAG	Description	Length	Format	Location	Reference
emvcard.sdad(*format*)	9F 4B	Signed Dynamic Application Data for XDA, DDA, CDA or fDDA. **GenAC**: emvcard.sdad() is used when CDA or XDA supported. The logic is defined as follows: if (P1==???1????b) [CDA requested], then Signed Data Format (SDF) value used to generate SDAD = '05' else if (P1==????1???b) [XDA requested], then SDF value used to generate SDAD = '15' else tag ’9F26’ (AAC cryptogram) returned **Internal Auth**: emcard.sdad() is used when DDA supported. SDF value used to generate SDAD = '05'. Contactless **GPO** or in **READ RECORD**: emvcard.sdad() is used when fDDA supported. SDAD is generated as follows: if (Card) CID = 'TC', then SDF value used to generate SDAD = '05' else if (Card) CID = 'ARQC' and TTQ bit 'ODA for online authorizations supported' = 1b, then SDF value used to generate SDAD = '95' *format* 'A5' SDF may be used by Union Pay.	var	Binary	Internal Auth Response Message Template Format 1 ID="80" or Format 2 ID="77" GenAC Response Message Template Format 2 ID="77" Contactless GPO Response Message Template Format 2 ID="77" Contactless Read Record Message Template ID="70"	Underlying Payment System’s specification, EMV Book 2, v4.4

Pseudo function Name	TAG	Description	Length	Format	Location	Reference
emvcard.cvc3t1()	9F 61	CVC3 dynamic value. The CVC3 (Track1) is a 2-byte cryptogram returned by the Card in the response to the COMPUTE CRYPTOGRAPHIC CHECKSUM command	2	Binary	ComputeCryptographicChecksum Response Message Template format 2" ID="77"	Underlying Payment System's specification
emvcard.cvc3t2()	9F 62	CVC3 dynamic value. The CVC3 (Track2) is a 2-byte cryptogram returned by the Card in the response to the COMPUTE CRYPTOGRAPHIC CHECKSUM command	2	Binary	ComputeCryptographicChecksum Response Message Template format 2" ID="77"	Underlying Payment System's specification
emvcard.iad (*length*)	9F 10	Issuer Application Data. Contains proprietary application data for transmission to the Issuer. *length* is mandatory and indicates the number of bytes	Up to 32	Binary	GenAC Response Message Template Format 1 ID="80" or Format 2 ID="77" Contactless GPO Response Message Template Format 2 ID="77"	Underlying Payment System's specification
emvcard.ctq(*initvalue*)	9F 6C	Card Transaction Qualifiers (CTQ). Indicate to the device the card CVM requirements, issuer preferences, and card capabilities. The emvcard.ctq function calculates the final CTQ based upon the mandatory initial value of the CTQ specified as parameter: The initial value indicates which CVMs the card supports, along with other card characteristics.	2	Binary	GPO Response Message Template Format 2 ID="77"	Underlying Payment System's specification

Pseudo function Name	TAG	Description	Length	Format	Location	Reference
		The final disposition of the CTQ will be determined by the following CVM hierarchy: Online PIN takes precedence over CDCVM; CDCVM takes precedence over Signature. • At the beginning of the transaction the CTQ is set to the initial value with B1b7, B1b8, B2b8 reset. i.e. All bits relating to CVMs required or performed are set to zero. • If the terminal supports PIN and requests CVM and B1b8 of the initial value is set, then set CTQ B1b8 to 1b (Online PIN required). • Otherwise, if the terminal supports CDCVM and requests CVM and B2b8 of the initial value is set, then set CTQ B2b8 to 1b (CDCVM Performed). • Otherwise, if the terminal supports signature and requests CVM and B1b7 of the initial value is set, then set CTQ B1b7 to 1b (Signature required).				

Pseudo function Name	TAG	Description	Length	Format	Location	Reference
emvcard.cpr(*initvalue*)		Card Processing Requirement The emvcard.cpr function calculates the final CPR based upon the initial value specified as parameter • At the beginning of the transaction the CPR is set to the initial value with B1b6, 7,8 reset • If the terminal supports PIN and requests CVM and B1b8 of the initial value is set then set CPR B1b8 • Otherwise, if the terminal supports signature and requests CVM and B1b7 of the initial value is set then set CPR B1b7	2	Binary	GPO Response Message Template Format 2 ID="77"	Underlying Payment System's specification
emvcard.UN(length)	9F 7F	Contains the Card challenge (random), obtained in the response to the GET PROCESSING OPTIONS command The length parameter indicates the number of bytes of the unpredictable number If the length value is not present, then default value is 4.	var	Binary	GPO Response Message Template Format 2 ID="77"	Not Applicable
emvcard.msd()	57	Part of the Track2 Equivalent Data	7	nibbles	Read Record Response Message Template ID="70"	Underlying Payment System's specification

Pseudo function Name	TAG	Description	Length	Format	Location	Reference
emvcard.track2 (track_2_EMV, track_2_MS)	57	Return the Track 2 Equivalent Data that corresponds to the transaction Mode. If the GPO Command indicates EMV Mode (Byte 1 Bit 8 of tag '9F35' is set to '1'), then the value 'track_2_EMV' will be return in tag '57'. Otherwise, if the GPO command indicates Mag Stripe Mode (Byte 1 Bit 8 is set to '0'), then the value 'track_2_MS' will be returned in tag '57'.	var	Binary	Read Record Response Message Template ID="70"	Underlying Payment System's specification
emvcard.country(fn, c1, c2)	5F 28 (Issuer Country Code) Country Code in ISO-3166-1 Numeri c format	c1 (mandatory), preferred Country Code (3N) c2 (mandatory if fn = 2), secondary Country Code (3N) fn = 1, required Country Code is "domestic", return Terminal Country Code (9F1A) provided in PDOL. fn = 2, required Country Code is "international". Examine 9F1A in PDOL. If 9F1A matches c1, return c2. Otherwise return c1. If 9F1A value is not present in PDOL data then return c1.	2	Binary	Read Record Response Message Template ID="70" GPO Response Message Template Format 2 ID="77"	EMV Book 3

Pseudo function Name	TAG	Description	Length	Format	Location	Reference
emvcard.currency (fn, c1, c2)	9F 42 (Application Currency Code) Currency Code in ISO-4217 Numeric code	c1 (mandatory), preferred Currency Code (3N) c2 (mandatory if fn = 2), secondary Currency Code (3N) fn = 1, required Currency Code is “domestic”, return Transaction Currency Code (5F2A) provided in PDOL fn = 2, required Currency Code is “international”. Examine 5F2A in PDOL. If 5F2A matches c1, return c2. Otherwise return c1. If 5F2A value is not present in PDOL data then return c1.	2	Binary	Read Record Response Message Template ID="70" GPO Response Message Template Format 2 ID="77"	EMV Book 3

Pseudo function Name	TAG	Description	Length	Format	Location	Reference
emvcard.AC()	9F26	Make AC calculation. L3 CS will calculate 8-byte Hash using SHA-1 on data exchanged between the card and the terminal, using the latest available values for the following tags: • Transaction Currency Code - 5F2A • Application Interchange Profile - 82 • Terminal Verification Results - 95 • Transaction Date - 9A • Transaction Type - 9C • Amount (Authorized) - 9F02 • Amount (Other) - 9F03 if present • Issuer Application Data - 9F10 • Terminal Country Code - 9F1A • Application Transaction Counter - 9F36 • Unpredictable Number - 9F37 AC will consist of first 8 bytes of the Hash value. This pseudo function is CVN agnostic. A L3 CS will determine when it needs to calculate the cryptogram using this algorithm when an xml test card will refer to this pseudo function. ***Note***: this new pseudo function is optional for participant systems to use.	8	Binary		N/A

3.1 C8-specific Pseudo Functions

Pseudo function name	TAG	Description	Length	Format	Location	Reference
emvcard.EDAMAC()	9F8105	The Enhanced Data Authentication MAC is a MAC over the Application Cryptogram and Issuer Application Data MAC.	8	Binary	GenAC Response Message	EMV Book C8 - section C.46.
emvcard.CardKeyData(*number*)	9F8103	The Card Key Data includes the x coordinate of the ECC blinded public key point (bytes 1 to N_{FIELD}) and the encrypted blinding factor (bytes N_{FIELD}+1 to 2*N_{FIELD}) returned by the Card in the GET PROCESSING OPTIONS response. *number* (mandatory), non-random blinding factor.	Var. up to 132	Binary	GPO Response Message	EMV Book C8 section A.1.28 and refer to appendix A (below).

emvcard.CVD(init_value1, init_value2)	9F8102	Determines the final CVD (Cardholder Verification Decision) based upon two 4-byte *init_value* mandatory parameters, along with the TRMD sent by the terminal. Each init_value indicates which CVMs the card supports below (first parameter) and above (second parameter) the CVM Limit. Each init_value is coded as a concatenation of 4 bytes (corresponding to 4 CVDs) according to the following definitions: Tag 9F8102 - Cardholder Verification Decision: • '00': NO CVM • '01': OBTAIN SIGNATURE • '02': ONLINE PIN • '03': CDCVM If fewer than 4 CVDs need to be supported by the card below or above the CVM Limit, the remaining bytes can be 'FF' filled. For example, emvcard.CVD(00FFFFFF, 0302FFFF) is used when only No CVM (00) is supported below the CVM Limit and CDCVM (03) and online PIN (02) are supported above the CVM Limit. ***Note***: The order of the CVD values is not relevant. The hierarchy of CVMs is fixed as: CDCVM (highest priority), Online PIN, Signature, No CVM (lowest priority).	1	Binary	GenAC Response Message	EMV Book C8 – section A.1.25

Pseudo function name	TAG	Description	Length	Format	Location	Reference
		See **Appendix B** for detailed description.				

emvcard.CardTVR(*value*)	9F8104	*value* is a 5 bytes hexadecimal value. Note that this Pseudo-function can be used to change any bit in the TVR except for TVR byte 3 bit 3 and 8 which may be dynamically changed. **Logic:** initiate TVR=TVR from GEN AC bitwise OR *value*. if CVD=02, then TVR byte 3 bit 3=1b. else if CVD=FF, then TVR byte 3 bit 8=1b. Card TVR (tag 9F8104)=TVR. **Requirements:** • The L3 CS must be able to extract the TVR from the GEN AC. • The CVD value must be determined prior to processing the emvcard.CardTVR(*value*) Pseudo-function. • The TVR must be determined prior to calculating the cryptogram as when the L3 CS changes the TVR, the TVR used as input to the Application Cryptogram calculation uses that changed TVR. • If using the emvcard.AC() Pseudo-function, then it shall use the resulting Card TVR value as part of the Application Cryptogram calculation. • If the Card TVR value is hardcoded, then the TVR used as input to the Application Cryptogram calculation is that Card TVR value.	5	binary	GenAC Response Message	EMV Book C8 – section A.1.30

Appendix A – EMV Book C8 cryptographic concepts

Blinded Diffie-Hellman Key Agreement

Card
Dc = Card private key (fixed for the card)
Qc = Card public key, which is Dc • G (Where G is the generator point of the curve)
R = Blinding factor, which is random
Pc = Blinded public key, which is r • Qc

Kernel
Dk = Kernel private key (not fixed for the kernel / ephemeral)
Qk = Kernel public key, which is Dk • G (Where G is the generator point of the curve)

Qk is sent to the card in the GPO command by the kernel.

Card derives the shared secret;
$Z = Dc \bullet r \bullet Qk$
$Z = Dc \bullet r \bullet Dk \bullet G$

Card returns Pc in GPO response and the kernel derives the shared secret;
$Z = Dk \bullet Pc$
$Z = Dk \bullet r \bullet Dc \bullet G$

Actual functionality of the C8 Spec

Kernel	**Card**
For each transaction generate…	PDOL contains…
ECC Private key	Kernel Key Data tag ‘9E’ (Section A.1.78) - This is the Kernel ECC Public key
ECC Public Key	Kernel Qualifier Tag ‘9F2B’ (Section A.1.79)
Initialise…	Initialise…
Kernel Message Counter = ‘0000’	Card Message Counter = ‘8000’
Card Message Counter = ‘8000’	
Send GPO Command (Kernel Key Data, Kernel Qualifier, …)	Process GPO
	Blinding Factor = Random 32-byte number
	Blinded ECC Public Key = (Blinding Factor * ICC ECC Private Key mod n) • G Where n and G are parameters from the P-256 curve in ISO/IEC 15946-5 {EMV Book C8 Annex D – Curve P-256}
	Shared Secret = (Blinding Factor * ICC ECC Private Key mod n) • Kernel Key Data
	Z = x coordinate of Shared Secret as a 256-bit value
	K_D (128 bits) = AES-CMAC (Z, key = 128 bits of 0b) {EMV Book C8 Section 8.6 – AES-CMAC}
	Data Block = ‘01010054334A325957773DA5A5A50180’
	Session Key for Confidentiality SKC(Card) = AES encipher(Data Block, key = K_D)

Kernel	Card
	Data Block = ‘02010054334A325957773DA5A5A50180’
	Session Key for integrity SKI(Card) = AES encipher(Data Block, key = K_D)
	Encrypted Blinding Factor = AES-CTR encrypt (Blinding Factor, key = SKC(Card), Counter = Card Message Counter) {EMV Book C8 Section 8.5 - EnDecryptData}
	Card Message Counter = Card Message Counter + 1
	Card Key Data = Blinded Public Key (x coordinate in byte 1 to 32), Encrypted Blinding Factor (Bytes 33 to 64)
	Send GPO response (Card Key Data)
Recover the Blinded Public Key in (x,y) format from the Card key Data (1 to 32)	
Shared Secret = Blinded Public Key • Kernel ECC Private key	
Z = x coordinate of Shared Secret as a 256 bit value	
K_{Dk} (128 bits) = AES-CMAC (Z, key = 128 bits of 0b) {note 2}	
Data Block = ‘01010054334A325957773DA5A5A50180’	
Session Key for Confidentiality SKC(Kernel) = AES encipher(Data Block, key = K_{Dk})	
Data Block = ‘02010054334A325957773DA5A5A50180’	
Session Key for integrity SKI(Kernel) = AES encipher(Data Block, key = K_{Dk})	

Kernel	**Card**
Blinding Factor = AES-CTR decrypt (Card Key Data (33 to 64), key = SKC(Kernel), Counter = Card Message Counter) {note 3}	
Card Message Counter = Card Message Counter + 1	

Appendix B – Logic of emvcard.CVD(init_value1, init_value2) function

The purpose of this function is to return a value of the CVD as determined by the bits set in the TRMD (i.e., what the terminal requests) and the CVMs supported by the card (specified by the parameters provided to the function). Consider emvcard.cvd(A, B) with A as supported CVDs below CVM Limit and B as supported CVDs above CVM Limit.

```
if  “CVM Limit exceeded” is not set in TRMD (Tag 9F1D Byte 2 bit 8) then
    if “CDCVM (Contactless)” is set in TRMD (Tag 9F1D Byte 1 bit 3) and init_value[A] contains 03 then CVD=03
    else if “Enciphered PIN verified online (Contactless)” is set in TRMD (Tag 9F1D Byte 1 bit 7) and init_value[A] contains 02 then CVD=02
    else if “Signature (paper) (Contactless)” is set in TRMD (Tag 9F1D Byte 1 bit 6) and init_value[A] contains 01 then CVD=01
    else if “No CVM required (Contactless)” is set in TRMD (Tag 9F1D Byte 1 bit 4) and init_value[A] contains 00 then CVD=00
    else CVD=FF (CV FAILED)
    end if
else if “CVM Limit exceeded” is set in TRMD (Tag 9F1D Byte 2 bit 8) then
    if “CDCVM (Contactless)” is set in TRMD (Tag 9F1D Byte 1 bit 3) and init_value[B] contains 03 then CVD=03
    else if “Enciphered PIN verified online (Contactless)” is set in TRMD (Tag 9F1D Byte 1 bit 7) and init_value[B] contains 02 then CVD=02
    else if “Signature (paper) (Contactless)” is set in TRMD (Tag 9F1D Byte 1 bit 6) and init_value[B] contains 01 then CVD=01
    else if “No CVM required (Contactless)” is set in TRMD (Tag 9F1D Byte 1 bit 4) and init_value[B] contains 00 then CVD=00
    else CVD=FF (CV FAILED)
    end if
end if
```

***** END OF DOCUMENT *****