



EMV TEST SCRIPTS MANUAL

CHIP PERSONALISATION TESTER CPT 3000v3

CHIP APPLICATION TESTER CAT 3000v3

VERSION 42

**Barnes International Limited
The Loft, St. Clair's Farm
Wickham Road
Droxford
Hampshire
SO32 3PW
United Kingdom**

**Tel: +44 (0)1730 231313
Fax: +44 (0)1730 265353
support@barnestest.com**

www.barnestest.com

Table of Contents

Testing EMV Payment Applications	3
Understanding the EMV payment transaction	4
Glossary of Acronyms	15
Scenario Stages	18
Error Coding and Suppression	19
Editing The Terminal Simulator	21
Terminal Setup	22
Terminal Capabilities	23
Additional Terminal Capabilities	23
Terminal Type	23
VSDC Terminal Transaction Qualifiers	23
Data Authentication Support	23
Payment Scheme and International Settings	24
Transaction Setup	26
Transaction Procedure	28
Proprietary and User Specified Data	31
Host Simulation	35
Editing the Data Analysis Stage	38
Data Analysis General Options	38
Specific Issuer Requirements	40
Wildcard Syntax	42
User Specified Data	44
Application Comparison	46
Embossing Display	47

Testing EMV Payment Applications

CPT/CAT 3000v3 is designed for maximum flexibility and upgradeability, to this end the card test sequences do not form part of the program but are controlled by separate Test Scripts. Because of this design approach the main manual and on line help documents are necessarily general in nature and do not cover specific features of individual card tests. This document/help file covers the test scripts and test scenario (previously known as profile) definitions for the EMV payment application tests. NB. Although contactless transactions do not have to fully comply with EMV, they are generally based on EMV so that these scripts will also test the mainstream contactless devices, including NFC equipped mobile handsets. For convenience in this document the term “card” is used to denote the payment device under test, regardless of what sort of hardware it is running in.

The EMV terminal simulator script simulates a transaction in a terminal in order to interrogate the card and extract the personalised data from it. The extracted data can then be checked against the relevant specifications and the individual issuer requirements set up in the data analysis scenario.

The terminal simulation script can be set up to simulate a wide variety of transactions in a wide variety of terminal types, so that as well as verifying the card's personalisation data its functionality can be checked in a variety of transaction scenarios.

The data analysis script can call up verification scripts to check the card's data against EMV 4.3, or against any of the major Payment Scheme specifications. These scripts will change from time to time as Barnes issues updates to cover changes in the specifications and scheme operating regulations, as well as to improve the testing process in the light of feedback from our customers. In order to assimilate a new test script it is only necessary to copy the new file into the scripts folder, replacing the earlier file of the same name.

Note that all Tcl scripts are protected by secure checksums, this ensures that only scripts issued by Barnes or by a *bona fide* CAT3000v3 developer, and not damaged or tampered with, will run the card tests.

NB. For the rest of this document the term “CPT” will be used, but all statements are equally applicable to CAT 3000v3 and FIME™ PersevalPro Issuer users.

Understanding the EMV payment transaction

This section is an attempt to describe the process involved in using a chip card to perform a financial transaction. It is not intended to provide an in depth knowledge of the technical details, for that the reader should consult the official specifications, nor does it attempt to give more than a cursory idea of the mathematics of the cryptography involved. It merely attempts to give the reader who is either non-technical or new to EMV cards an overall understanding of the process, to de-mystify some of the jargon and induce a basic understanding of the personalisation data in the card.

Given this basic overview, those who wish to delve deeper should be better prepared to understand the specifications, and more to the point, to understand the test results produced by the CPT 3000v3.

Overview

There are three computers involved in an EMV transaction, we will refer to them as the card, the terminal and the host. Each one performs the function of representing its owner. The card is in the possession of the person making the purchase and represents the interests of the issuer, and through them the interests of the person to whom it was issued (who is not necessarily the person in possession of it). The terminal is part of, or attached to, a cash register and represents the interests of the merchant providing the goods or services, and the "host" is the mainframe providing an online backup to the decision making process, it may be on the issuers premises or be operated on their behalf by one of the payment schemes.

The card's computer (which we shall refer to simply as the card) is a single chip microcomputer embedded in the rectangle of plastic or the token or mobile handset carried by the cardholder, it is the smallest and least powerful of the three, and although the processing power available within the card is constantly increasing, the design of the transaction procedure was to a large extent conditioned by this lack of power in the early contact cards. The terms "smart card", "chip card" and "integrated circuit card (ICC)" are all alternative names for the same device, in this document we shall simply use the term "card".

The terminal will generally be a cash register, vending machine or an ATM and the host is a huge mainframe sitting in a data processing centre somewhere. Usually the card and the terminal are close coupled, i.e. directly connected to one another, and the host is always at the other end of a telephone line or other long distance communications link, such as the internet.

A terminal can be described as "online", or "offline", in the former case this means that the terminal and the host can communicate directly during the transaction, in the latter case they cannot. ATMs, for instance, are normally online, whereas the cash register in a small grocery shop is more likely to be offline. Offline terminals will send their transaction records to the host at some later time, the method of doing this will vary according to local circumstances.

Always keep in mind the underlying purpose, which is to transfer money from the cardholder's account to the merchant's account in an efficient and secure manner. There are a number of important considerations, they may seem obvious but are worth emphasising because they condition the design of the communications between the three computers and their decision making processes.

1. Authentication. All parties to the transaction must ensure that the other parties are who they say they are (e.g. The card is not forged, stolen or tampered with, the computer at the other end of the telephone really is the issuer's host).

2. Authorisation. All parties must ensure that the other parties are allowed to take part in this transaction. (e.g. The card is allowed to be used at this terminal and the cardholder's account has sufficient funds or credit for the transaction)
3. Efficiency. The whole process must take place within a very limited time frame (Long checkout queues are a waste of everyone's time and money).

NB. The EMV specifications do not cover contactless cards, so contactless payment applications are administered by the payment scheme operators. Although they are all based on EMV to a greater or lesser extent the process is modified to reduce the time the card and the terminal spend communicating. There is more variation from scheme to scheme in contactless than there is in contact cards.

Powering Up

When the card is inserted into the coupler the terminal applies power to the card and receives from it an ATR (Answer To Reset). The ATR describes to the terminal the card's communications capabilities, and allows the terminal to configure itself to talk to the card.

Communications between the terminal and the card are "half-duplex", that is, the same wire is used for messages in both directions. Obviously it is therefore important that the two units do not both transmit at the same time, so there is a strict protocol laid down to prevent this. In essence the terminal controls the wire, so all communication exchanges take the form of a command from the terminal followed by a response from the card. There are no exceptions to this.

In the CPT 3000v3, in common with most personalisation testers, this low level protocol is taken care of entirely within the card reader attached to the system. Barnes does not manufacture these readers, but sources them from commercial suppliers. (Barnes does manufacture a low level protocol test tool, the CET 3000, but this is a completely different type of machine)

Application Selection

Being a computer, the card can hold a number of applications. Just as a desktop computer can have a word processor, a spreadsheet, and an adventure game, all of which are software applications, so the card can hold a debit application, a credit application, a loyalty application, and so on. The first task for the card and the terminal is to agree on the correct card application for the transaction being conducted. Most early cards only had one application in them, but the trend in newer cards is to have two or more.

The first command the terminal sends to the card is a "Select" command. The Select command asks the card to activate an application, and of course must carry information as to what application the terminal wants to activate. The first Select command will request the activation of the PSE (Payment Systems Environment). The PSE is not a payment application itself, but a directory of the payment applications on the card, it performs the same function as the start menu on a PC, telling the user what software is available and giving him a quick means to access it. The card will respond to any Select command either with details of the application that has just been activated, or with an error code that indicates the application is not available. The PSE is not mandatory for an EMV card, although most current cards have one because it speeds up the process of starting the payment application.

Following a successful PSE selection the terminal will issue a number of "Read Record" commands, to which the card will respond with the data held in each of the PSE's records. Each record will contain one or more entries describing an available payment application, as a minimum each of these entries will contain a "Label" which can be shown on the terminal's display, and an AID (Application IDentifier), which is a code the terminal sends to the card in another Select command to activate that particular application.

NB. Contactless cards have a slightly different protocol for their PPSE (Proximity Payment Systems Environment), to speed up the process all the information is returned in response to the Select command, no Read Record commands are needed.

When there are several applications present each one will normally also have a priority indicator which assists the terminal in choosing the best application for the job. The terminal may make the choice autonomously, depending on the applications it itself can support, or it may present the cardholder with a list and ask him or her to make the choice. Either way, once the decision is made the terminal knows the AID of the payment application to be used, and can issue a Select command to activate it.

If there is no PSE on the card the terminal has no choice but to try a number of AIDs until the card responds positively. The terminal will contain a list of AIDs for the applications it supports and will try each in turn to see which of them opens. Obviously if there is a long list this could take some time.

The AID is a hexadecimal number made up of two parts, the first five bytes are known as the RID (Registered application provider IDentifier), and identifies the payment scheme, the remainder is the PIX (Proprietary application Identifier eXtension) and identifies the account type. For instance the RID for Visa is A000000003. Most schemes use the PIX 1010 for their standard or flagship account types, so A0000000031010 will be a standard VSDC application, whereas A0000000032010 indicates an Electron card. The RID values for the other major payment schemes are A000000004 for Mastercard, A000000065 for JCB and A000000025 for American Express.

A successful response to the Select command will contain information about the application, including a repeat of some of the data that was in the PSE, but also additional information which will enable the terminal to continue the process.

Get Processing Options.

The first command of the transaction proper is "Get Processing Options", this formally begins the transaction and allows both the card and the terminal to configure themselves for what is to follow.

Amongst the information returned by the card in the response to the Select command was the PDOL (Processing options Data Object List). Data object lists are the card's way of requesting data from the terminal, and there are several different ones that may or may not be used, all of them contain the letters "DOL" in their acronym.

The PDOL is not mandatory, it is only necessary if the card is capable of configuring itself to behave differently depending on the local conditions. Most current PDOLs contain a request for the Terminal Country Code, and could request other data such as local currency, terminal type etc. The terminal supplies this information with the Get Processing Options command and the card then returns two essential items of information, the Application Interchange Profile (AIP) and the Application File Locator (AFL).

NB. A major exception to this is Visa's qVSDC application ("q" for "quick"). qVSDC is used on contactless cards only and differs from an EMV application in that a great deal more information is requested by the PDOL, so that usually the transaction can be completed in the Get Processing Options command with no further processing.

The AIP is a two byte value, arranged as sixteen single bit "flags". Each bit represents a function that the card may or may not support, the bit is set to one if the function is supported. It mainly covers such things as authentication and authorisation methods. At the time of writing not all of

the bits are allocated, and some of those that are allocated only apply to contactless cards or mobile devices (their meanings varying from scheme to scheme), the others are available for future expansion.

The AFL is a list of file and record references which the terminal then uses to issue a series of "Read Record" commands to the card in order to extract the data required to perform the transaction. The data contained in these records includes obvious things like the account number and the expiry date, but also more complex information which allows the terminal to verify that the card and the cardholder are both genuine and informs it how the issuer requires the terminal to react under different circumstances.

TLV Encoding and Data Object Lists

It is worth turning aside at this point to describe the way data is encoded for storage within the card and for transfer between the card and the terminal. The technique is known as TLV encoding, where TLV stands for Tag, Length and Value. This method is a good compromise between simplicity of processing and efficient use of limited memory, and so is ideal for use in small low powered computers such as the one in the card.

The tag is the identifier of the item of data, and is always the first one or two bytes of the data stream (the encoding of the first byte defines whether or not another byte follows). EMV assigns tags to the data objects used in the transaction, most are stored in the card and transmitted to the terminal, some are stored in the terminal and supplied to the card, others are generated dynamically during the transaction, and a few are stored in the card and never transmitted to anyone. EMV also assigns a range of tags for proprietary use by individual payment schemes.

People refer to data objects in different ways depending on their area of interest and level of expertise, all objects have a tag and a descriptive name, and many also have an abbreviated acronym. e.g. Tag 5A identifies the Application Primary Account Number, or PAN.

Immediately following the tag is the length, again this can occupy one or more bytes depending on the encoding of the first byte, and defines the number of bytes taken up by the value.

The remaining bytes, the value, are the actual data. The interpretation of the data depends on the data object itself, i.e. on the tag, and in some instances includes a number of other, embedded TLV encoded objects (TLV objects containing other TLV objects are often called "Templates").

There are a number of variations on the TLV theme, the one used in EMV transactions (BER-TLV, or Basic Encoding Rules TLV) is fully described in Annex B of EMV 4.3 Book 3.

When the card passes a DOL (Data Object List, e.g. the PDOL) to the terminal, the DOL has its own tag, (the PDOL's tag is 9F38), and its value field consists of a list of tags and lengths without their values. The terminal then supplies the values associated with these tags (using the lengths and order indicated by the DOL) as the data field of a subsequent command.

The example below is taken from a CPT 3000v3 engineer's log, it shows a successful selection of a VSDC application, with a PDOL requesting the Terminal Country Code (tag 9F1A, length 2). The terminal then passes the value (0826 representing the UK) back to the card in the Get Processing Options Command.

```
Selecting Application A0000000031010
First Selection
<snd> (013) 00 A4 04 00 07 A0 00 00 00 03 10 10 00
<rcv> (059) 6F 37 84 07 A0 00 00 00 03 10 10 A5 2C 50 0B 56
           49 53 41 20 43 52 45 44 49 54 9F 38 03 9F 1A 02
```

```

          9F 12 0F 43 52 45 44 49 54 4F 20 44 45 20 56 49
          53 41 87 01 01 9F 11 01 01 90 00
*Tag 84 - DF Name - A0 00 00 00 03 10 10
*Tag A5 - FCI Proprietary Template - Length 44
*Tag 50 - Application Label - VISA CREDIT
*Tag 9F38 - PDOL - 9F 1A 02
*Tag 9F12 - Application Preferred Name - CREDITO DE VISA
*Tag 87 - Application Priority Indicator - 01
*Tag 9F11 - Issuer Code Table Index - 01

Building PDOL 0826
Get Processing Options
<snd> (010) 80 A8 00 00 04 83 02 08 26 00
<rcv> (018) 80 0E 5C 00 08 01 01 00 10 01 03 00 18 01 02 01
          90 00
*Tag 82 - Application Interchange Profile - 5C 00
*Tag 94 - Application File Locator - 08 01 01 00 10 01 03 00 18 01 02
01
*File Locator 1 - 1 1 1 0
*File Locator 2 - 2 1 3 0
*File Locator 3 - 3 1 2 1

```

Asymmetric Key Cryptography and Data Authentication

Once all the data has been extracted the terminal performs the offline data authentication process, this process ensures that the card is a genuine card issued by the financial institution it claims to represent, and that it has not been tampered with.

Offline data authentication uses Asymmetric Key Cryptography, sometimes known as public/private key cryptography, or PKI. Asymmetric key means that there are two keys, one of which is used for encryption and the other for decryption, the keys are generated in such a way that it is impossible to calculate one from the other. Then one key is kept secret and the other is made public.

For data authentication in an EMV transaction, the encryption key is kept secret and the decryption key is made public. The mathematical algorithm used is RSA (after Rivest, Shamir and Adelman, the three mathematicians who designed it). The major disadvantage of RSA is that it requires significant computing power to perform the calculations, although as processors increase in power this is becoming a much less important issue. Elliptic Curve Cryptography is another asymmetric key algorithm, which is gaining in popularity because of its lower demands on the computer, although provision has been made to accommodate it, it is not currently used by EMV.

Asymmetric key cryptography is used for offline authentication because the terminal is owned or managed by a different organisation from the card and the host, so the use of a public key means there are no security issues to worry about. Conversely, online authentication takes place between the card and the host, which are both managed by the same organisation, and so it is possible to use symmetric key cryptography, where the same, secret, key is used for encryption and decryption.

Online authentication is described in more detail below. NB. If a card is only ever used for online transactions then the data authentication check by the terminal becomes unnecessary, and "online only" cards will often not support data authentication at all.

Static Data Authentication

SDA is the simplest form of offline data authentication, it requires no RSA processing within the card.

One of the data objects recovered from the card during the "Read Record" phase is the Issuer Public Key Certificate (tag 90), which is a large block of data stored pre-encrypted on the card. Associated with this is another item called the Certification Authority Public Key Index (tag 8F), the "Certification Authority" is generally the Payment Scheme, i.e. Visa, Mastercard etc. The data in the certificate is supplied by the issuer and encrypted on their behalf by the payment scheme authority, the issuer does not have access to the private key used. The corresponding public keys are held in the terminal, each one being uniquely identified by the application's RID and the CA Public Key Index (tag 8F). Successful decryption of the Issuer Public Key Certificate using the payment scheme's published public key verifies that the data was truly encrypted by that authority using their private key.

NB. An RSA key is in two parts, known as the exponent and the modulus, in fact there are two exponents, one public and the other private, the same modulus is used with both. Making the calculation with public exponent exactly reverses the effect of the private exponent, and vice versa, so either can be used for encryption. The public exponent is a one or three byte value, the modulus and the private exponent both have a much greater length. The RSA algorithm always operates on a data block which is the same length as the modulus and the private exponent. The term "public key" is used to describe the modulus and the public exponent, the term "private key" refers to the private exponent.

Successful decryption of the Issuer Public Key Certificate yields the Issuer Public Key, no surprises there then. The private key associated with the Issuer Public Key is the property of the card issuer. Usually the Issuer Public Key is too large to be entirely accommodated within the certificate along with the other data that is held there, so only the first part of the key is in the certificate and the rest of it is held in clear under tag 92, the Issuer Public Key Remainder.

Also recovered from the Issuer Public Key Certificate is a twenty byte hash (a hash is a sophisticated form of checksum) of the data within the certificate and the remainder, so the terminal can verify that no data has been corrupted or tampered with.

Armed with the Issuer Public Key the terminal can now decrypt the Signed Static Application Data (tag 93). This yields a hash of the card's sensitive data, i.e. data that is recovered from the card in clear but needs to be protected against corruption and tampering. The data to be hashed is identified by coding in the AFL. If this hash verifies correctly then SDA has succeeded, i.e. all the data is correct and genuine.

NB. All the encryption processes involved in SDA are carried out before the card is personalised and the card simply stores the encrypted data. This means that it is possible for a fraudster to "skim" the card data. So, although online transactions are protected by the symmetric key cryptography described later, offline transactions are vulnerable.

Dynamic Data Authentication

DDA is a little more complex, but provides better security. DDA requires the card to perform some RSA calculations.

DDA begins in the same way as SDA, with the decryption of the Issuer Public Key Certificate to obtain the Issuer Public Key. In this case, however, the Issuer Public Key is used to decrypt the ICC Public Key Certificate, which is similar in format to the Issuer Public Key Certificate, but is encrypted using the issuer's private key, and yields the ICC Public Key. ICC is "Integrated Circuit

Card", and the ICC public key is unique to the card, with the corresponding private key held within the card. The issuer has the option to include other card data within the hash used to verify the ICC public key, so a *de facto* SDA check is carried out during key retrieval.

The terminal then transmits to the card an "Internal Authenticate" command, which carries with it data defined by the DDOL (DDA DOL) supplied by the card during the "Read Record" phase. If the card does not supply a DDOL then EMV defines a default DDOL which must be used. Among the data is always an "Unpredictable Number" generated by the terminal at transaction time. The unpredictable number may be random or sequential, so long as it is different for every transaction (i.e. the card must not be able to predict it, even if the terminal can). The card then generates a twenty byte hash of the data supplied by the terminal concatenated with another unpredictable number generated by the card. The hash and the card's unpredictable number are then incorporated into the Signed Dynamic Application Data, which is encrypted using the card's private key and returned to the terminal in the response to the Internal Authenticate command.

The terminal decrypts the Signed Dynamic Application Data using the ICC Public Key, recovers the ICC Dynamic Data and verifies the hash against the data that was transmitted. If the hash checks out then DDA has succeeded, i.e. the card not only contains correct and genuine data, but also the private key corresponding to the ICC public key encrypted by the issuer. The card's private key will never be transmitted to any terminal, it will only be used internally by the card, and its corresponding public key is encrypted using the issuer's private key which has never been anywhere near the card, this arrangement makes life very difficult for the potential fraudster.

Combined Data Authentication

There is a third method of data authentication, CDA, or to give it its full title "Combined Dynamic data Authentication and generate application cryptogram". The authentication process is the same as DDA but the data is exchanged during the "Generate Application Cryptogram" command, covered below. CDA provides further security by including transaction data within the signed data.

Cardholder Verification

Having established that the card is genuine, the terminal must now ensure that the person presenting the card is the genuine cardholder.

The terminal examines the contents of tag 8E, the CVM list (Cardholder Verification Methods). The CVM list contains two monetary amounts, called X and Y, and a prioritised list of verification methods. The two amounts may optionally be used to condition the choice of verification method, but this is rare, mostly they are set to zero and not used.

Each verification method is encoded in two parts, the method itself, and the conditions of use. The terminal works its way down the list applying the method if the conditions permit and taking the indicated action if the verification fails. As soon as one method succeeds then the cardholder is verified.

As an example, the list might define "Offline Encrypted PIN" and "If Terminal Supports" as its first CVM, so a terminal which is not fitted with a PIN pad would skip this one and adopt the next, which might be "Paper Signature", and so on.

Offline PINs can be submitted to the card either in clear (plain text) or encrypted. Plain text PIN can only be used when the connection between the PIN pad and the card is secure, i.e. entirely within a tamper-proof enclosure, if there is any danger of the message from the PIN pad to the card being intercepted then encrypted PIN must be used.

With contactless cards communicating with the terminal by radio (albeit very short range radio) plaintext offline PIN is obviously out of the question, so if a contactless card supports offline PIN at all, it must be encrypted.

When an encrypted offline PIN is submitted to the card the terminal encrypts the PIN using a public key supplied by the card. If the card supports DDA or CDA then the ICC public key obtained there is often used to encrypt the PIN, alternatively a specific PIN encipherment public key may be used, this is obtained using the Issuer Public Key in exactly the same way as the ICC Public Key for DDA or CDA was obtained.

Decision Time

With all of the above completed it is now time to make the decision whether or not the transaction will be allowed to go ahead. The terminal and the card "vote" on what to do, if either of them votes "no", then the transaction will not proceed, they must both vote "yes" before any money changes hands. If the terminal is online then the issuer's host may also be involved in the decision making process.

The terminal fills in a set of binary flags called the TVR, or Terminal Verification Results, the TVR is a five byte value with each bit representing a different situation or event. Not all of the bits are currently allocated.

TVR bits represent such things as "New card", "Expired Application", "PIN Not Entered" and so on. These bits are then compared against various sets of "action codes" to determine the next step. Action codes are five byte data objects with exactly the same bit allocations as the TVR, each one specifies an action that must be taken if a "1" bit in the action code matches a "1" bit in the TVR.

The terminal reads from the card a set of Issuer Action Codes, named Denial, Online and Default, these represent the card issuer's preference as to what action should be taken by the terminal under each circumstance. The terminal may also carry a matching set of Terminal Action Codes, which represent the merchant's preferences.

When there is a matching "1" bit in the TVR and either of the Action Codes then the action implied by the name of that code must be taken. E.g. if the bit allocated to "PIN Not Entered" is set to "1" in both the TVR and the IAC - Online, then the terminal must refer the transaction to the issuer by going online to the host.

The action codes are processed in the order, denial, online, then default, such that conditions that generate a denial take the highest priority, followed by online. The default codes are used to generate a denial if the terminal is unable to go online, or if for any reason the communications with the host fail.

If the transaction passes all these tests then the terminal may apply its own action codes to see if the merchant's requirements dictate a denial or an online referral.

Based on these codes the terminal chooses one of three courses of action, deny the transaction outright, refer the transaction online to the issuer, or accept the transaction offline. This decision is passed to the card as part of the first "Generate Application Cryptogram" command.

Application Cryptogram Generation.

Among the data extracted from the card during the Read Record phase will be two DOLs known as CDOL1 and CDOL2, (Card risk management Data Object List), which define the data the card requires from the terminal when the "Generate Application Cryptogram" commands are issued. A maximum of two Generate AC commands may be issued to the card during a transaction, the data accompanying the first is defined by CDOL1, and the data for the second, where required, by CDOL2. Data defined by the CDOLs always includes information about the transaction itself, amount spent, time and date etc. and the Generate AC command itself also carries the terminal's completion decision.

The three decision codes have acronyms that are not obvious. A denial is indicated by AAC (Application Authentication Cryptogram), an online request by ARQC (Authorisation ReQuest Cryptogram) and an offline acceptance by TC (Transaction Certificate).

On receipt of the Generate AC command, the card performs its own internal risk assessment processing, compares the result of this with the request from the terminal, and responds with a decision. The card's decision will never be more optimistic than the terminal's, i.e. if the terminal requests a denial then the card must always agree, but if the terminal requests offline acceptance the card may choose to go online or even to deny the transaction.

As well as this decision the card will transmit to the terminal a cryptogram. The cryptogram is a secure hash of the transaction data, and is generated using DES encryption so that only the host can verify it, the validation of the cryptogram assures the host that the card is genuine. The terminal sends the cryptogram to the host along with all the other transaction information either immediately (in the case of an online referral) or later as part of the day's transaction data.

In the case of an online referral, the host will reply with another cryptogram which the terminal passes to the card either using an "External Authenticate" command or as one of the data items specified in CDOL2. The card then verifies that this cryptogram really did come from its issuer and responds positively or negatively.

If the External Authenticate was used and was successful then the terminal issues a second Generate AC command, the data for which includes the host's decision, which is final. The second cryptogram from the card is later sent to the host as confirmation of what actually occurred following the referral.

Recent trends are to abandon the use of the External Authenticate command and instead include the host's cryptogram in the data defined by CDOL2, so the authentication is performed during the second Generate AC. This achieves the same objective with less communications traffic.

If the card uses CDA, then the data required for DDA will be passed to the card with the first Generate AC command and the response will include the Signed Dynamic Application Data, which will need to be verified before the card's decision is accepted and an online cryptogram is accepted for onward transmission to the host.

Symmetric Key Cryptography

The cryptogram is generated using the DES (Data Encryption Standard) algorithm. DES was originally developed for the US military for use as a high speed machine cipher. For some years now it has been in the public domain and is now widely used in non-military applications. DES is a symmetric key algorithm, that is, the same key is used for encryption and decryption, and so must never be made public. It is also economical in its use of processing power and so is

convenient for use in small, low powered, computers such as smart cards. Moves are afoot to replace it with the AES algorithm, which uses longer keys and is more secure.

The basic DES algorithm takes an eight byte clear data block and an eight byte key and combines them to form an eight byte encrypted data block. Decryption is the exact reverse and uses the same key. Longer messages are encrypted by breaking the data up into eight byte blocks and adding padding bytes if necessary to make the last block up to eight bytes, then encrypting each block independently. Two methods are defined for this, ECB (Electronic Code Book) and CBC (Cipher Block Chaining). In ECB each eight byte block is encrypted in isolation, while in CBC each block is combined with the encrypted data from the preceding block before itself being encrypted. Using CBC makes the code a little harder to crack but also has the characteristic that the value of the last encrypted block is dependent upon the values of all the blocks preceding it, which means it can be used as a secure checksum. The cryptogram generated by the card, and that returned by the host, are in fact secure checksums generated in this way, they do not carry any data themselves, but verify the authenticity of the data that accompanies them in clear.

Triple DES is a technique for increasing the security of the DES algorithm by using a longer key. In triple DES a sixteen byte key is provided, but is actually used as two eight byte keys. The technique, which is applied to each eight byte data block in turn, is to encrypt the block with the first key, decrypt it with the second key, then re-encrypt it with the first.

In order to maintain security of keys it is desirable that a DES key is used as little as possible, once only is the ideal. The practicality of doing this when both the card and the host need to use the same key on any given transaction is ensured by using derived keys.

The host maintains a master key (the MDK or Master Derivation Key), which is kept under strict security. Each card is given a unique master key of its own, the UDK (Unique Derivation Key) which is generated by encrypting the card's account number using the MDK. The UDK is stored in the card at personalisation time. For each transaction the card derives a session key from its UDK using the Application Transaction Counter, which is incremented every time the card receives a "Get Processing Options" command, and so is unique to a transaction, or session. When the host receives the cryptogram, it also receives the card's account number and current ATC, and so can calculate the session key from the MDK.

In practice the host will normally maintain a number of MDKs, and the card will return an index number, the Derivation Key Index (DKI) which will inform the host which one was used to derive this card's UDK. It is also normal practice for the card to hold three UDKs, each generated from a different MDK, so that a different session key is used for cryptogram generation, for secure data transfer, and for message authentication (see Issuer Script Processing, below).

Proprietary Data

As mentioned above, EMV defines a range of tags for proprietary data that is personalisation data that is not defined or controlled by EMV, but by the payment scheme operator. The interpretation and use of these data items is different for each payment scheme, but generally they are used to extend or qualify the card's decision making capability (for instance Card Issuer Action Codes, or CIACs, perform a similar function to the Issuer Action Codes used by the terminal, but are used solely by the card's internal processes). Because of interoperability issues it is extremely rare for the terminal to have any interaction with proprietary data.

Issuer Script Processing.

If the transaction is referred online to the host there is an opportunity for the host to send commands to the card. This is normally performed after second Generate AC. In theory almost anything can be done here, from installing a new PIN to a complete software update, but in practise only a limited number of commands are in general use.

Among the possibilities are ordering a card to block itself and refuse to operate further, (e.g. when it has been reported stolen or the cardholder has not paid his account), to allow the cardholder to alter the PIN, or to change account dependant settings when the cardholder's status changes (e.g. credit limit or foreign travel restrictions etc.).

The host makes up the command and sends it to the terminal in a packet labelled "Please send this to the card". The terminal extracts the command and its accompanying data and transmits it verbatim to the card. In this situation the terminal is merely part of the communications system, it takes no active part in the operation itself.

Sensitive data, such as a new PIN value, is encrypted using a session key derived from the card's secure data transfer UDK, the command itself and any non-sensitive data are transmitted in clear. All commands and data are authenticated using a MAC (Message Authentication Code). The MAC is generated from the command and data by the same method as the generation of the cryptogram, but using a session key based on the card's MAC UDK, only if the card validates the MAC will it act upon the command.

Glossary of Acronyms

The terms below are mostly explained in the section on understanding EMV transactions, a few are used without explanation and some are related terms that are not mentioned there.

AC - Application Cryptogram, encoded data sent from the card to the issuer's host to authenticate the transaction

AFL - Application File Locaters, data which enables the terminal to issue the correct Read Record commands to the card in order to obtain the necessary information to conduct the transaction

AID - Application Identifier, a series of bytes which identify an application, payment or otherwise, within the card.

AIP - Application Interchange Profile, a set of indicators which inform the terminal what features the card supports

Application - The software within the card that services a specific type of transaction, payment or otherwise.

Asymmetric key cryptography - A cryptographic process using two keys, one for encryption, the other for decryption. Normally one of the keys is public.

ATC - Application Transaction Counter, a number which is incremented each time the card is used, mainly used for cryptographic key diversification

ATM - Automatic Teller Machine, also known as a cash dispenser or a hole in the wall.

ATR - Answer To Reset. A message sent by a smart card to the terminal when it is first powered up, carries information which enables communications to be established.

Authentication - The process of establishing that a party to the transaction is authentic.

Authorisation - The process of establishing that the transaction is allowable

Bit - Binary digit, the smallest unit of data storage within a computer, represents 0 or 1, often interpreted as true or false.

Byte - The basic unit of number and text storage in a computer, eight bits representing a number from 0 to 255 or a single printable character

CBC - Cipher block chaining, a sequential method of using DES encryption

CDA Combined dynamic data authentication and generate application cryptogram

CDOL Card risk management data object list

Certification authority - The organisation overseeing the authentication process, usually the payment scheme operator.

Checksum - The result (usually one or two bytes in length) of a sequential calculation on a series of bytes. If any of the bytes should change then the checksum will also change, used to detect data corruption. See also hash, below.

Contact card - A card which is powered and communicated with using electrical connections between the coupler and the card

Contactless Card - A card which is powered and communicated with using high frequency radio signals between the card and the coupler.

Coupler - The component in the terminal which communicates directly with the card

Cryptogram - A secure checksum used validate a message from the card to the host or vice versa.

CVM - Cardholder verification method, e.g. signature, PIN etc.

CVR - Card verification results, generated by the card to inform the host of its decision process.

DDA - Dynamic data authentication - uses RSA cryptography within the card to verify the card's authenticity

DDOL - DDA data object list, identifies the data to be authenticated using DDA

DEA - Data encryption algorithm, an alternative acronym for DES.

DES - Data encryption standard - the symmetric key algorithm used in smart cards.

ECB - Electronic code book, the simplest method of implementing DES encryption.

EMV - The organisation originally formed and owned by Europay, Mastercard & Visa to co-ordinate and oversee the establishment of standards and specifications for financial transactions using smart cards. Since its establishment Europay has merged with Mastercard and JCB and

American Express have joined the organisation.

EmvCo - The executive arm of EMV, responsible for publishing specifications, approving products etc.

Generate AC - The command which causes the card to calculate an application cryptogram.

GPO - Get processing options - the first command issued to a payment application, establishes the conditions for the transaction.

Hash - The result (usually sixteen or twenty bytes in length) of a sequential calculation on a series of bytes. If any of the bytes should change then the hash will also change, used to detect data corruption or tampering. See also checksum, above.

Host - The issuer's mainframe computer. NB Sometimes host functions are run by payment schemes as a service their members.

IAC - Issuer action codes, instructions from the issuer to the terminal, carried in the card, on how to react to various situations.

ICC - Integrated circuit card, smart card or chip card.

Interoperability - The ability of systems built and managed by a number of diverse organisations to communicate and work together.

Issuer - The organisation issuing the card, usually a bank or credit card company

Issuer script processing - A method by which the issuer can make changes to a card after it has been issued.

MAC - Message authentication code, a secure checksum used to authenticate issuer script commands delivered to a card.

MDK - Master derivation key, the key from which unique DES keys for the cards are derived

Mobile Device - A device able to communicate with a payment terminal in the same way as a card, but which has an independent means of communicating with the issuer, e.g. a mobile telephone.

Offline - Unable to communicate directly with the host during the transaction.

Online - Able to communicate directly with the host during the transaction.

Diversification - The process of changing a DES key so that the same one is never used twice.

PAN - Primary account number, usually the number embossed on the front of the card.

Payment scheme - an umbrella organisation coordinating debit and/or credit card payment systems

PDOL - Processing options data object list, information the card requires at the start of the transaction, the data is delivered to the card in Get Processing Options.

Personalisation - The process of writing unique data to the card

PIN - Personal identification number

PIX - The proprietary section of the AID, distinguishes between account types within a single payment scheme

PPSE - Proximity payment systems environment, a directory of the payment applications on a contactless card

Proprietary - defined and managed by a payment scheme or issuer, rather than by EMV

PSE - Payment systems environment, a directory of the payment applications on a contact card

Read Record - The command sent by the terminal to request data from the card's filing system

RID - The first part of the AID, identifies the payment scheme.

RSA - The asymmetric key cryptography algorithm used by EMV

Select - The command sent by the terminal to the card to activate an application.

SDA - Static data authentication, verification of the data without requiring the card to perform RSA calculations.

Signed Static Application Data - The encrypted data block containing the data hash for SDA

Signed Dynamic Application Data - The encrypted data block containing the data hash for DDA

SHA-1 - The algorithm used for hash generation by EMV

Symmetric key cryptography - encryption method that uses the same key for encryption and decryption.

Tag - A one or two byte identifier for a data object

TAC - Terminal action codes, the merchant's preferences for action during a transaction

Terminal - The merchant's component in the transaction, e.g. a cash register or ATM.

TLV - Tag, length and value, an economical way of encoding and storing data

Triple DES - a modification of the DES algorithm which effectively doubles the length of the key
TVR - Terminal verification results, the results of the terminal's risk assessment tests.
UDK - Unique derivation key, the card's master DES KEY.

Scenario Stages

NB. For a general description of the test scenario concept please refer to the main CPT or CAT manual.

The bulk of the setup and editing for a test scenario takes place in the stage editors. To edit a stage, select it in the scenario editor screen and either click the edit button or double click the stage name.

The main area of the editor dialogue is occupied by a series of tabbed pages. The number and content of these pages is dependent on the type of stage being edited.

The rest of this topic describes the controls which are common to all stage types, for information on individual pages see subsequent topics.

Reader Selection

If the stage being edited is a terminal simulator and the system is enabled for contact and contactless readers then the interface to be used may be selected from the drop down list at the bottom of the screen.

Data Import

If the stage is a data analyser then the drop down box at the bottom is used to select the source of data used to auto-populate the stage data. The content of this list is dependent upon the stage type and the options enabled in the system, but typically will include such possibilities as using the data read from the last card to be tested, or importing data from an external software package such as the Visa VPA tool.

The button beside the drop down list performs the actual import action.

Back and Undo Controls

At the top of the dialogue there are two buttons which may be used to reverse recent edits. The button on the left reloads the stage data with the values that existed before the last save to disk. The button on the right reloads the data to that which existed when the dialogue was opened, i.e. to the state after the last save to disk and before any changes were made in the current session.

The left hand button will be disabled if the stage has not been saved before, as in the case of a new stage.

Save and Save As

The “save” button saves the edited data to disk, and the “save as” button allows the edited stage to be saved under another name. In this case the original stage will be unchanged. Both of these buttons close the dialogue.

Error Coding and Suppression

Errors and Observations

An **error** is a defect in the card's data or responses which causes the card to fail the test. An error might be caused by a violation of one or more of the various specifications the card must comply with, or by a deviation from the user defined test scenario criteria. The vast majority of errors will be produced by the data analysis stage(s), terminal simulators will normally only report the sort of gross error that prevents the transaction being completed.

An **observation** is an anomaly of the data which, while not being serious enough to cause a test failure, may need to be brought to the attention of the card issuer. It may also indicate that for some reason, not under the control of the card, the test could not be fully carried out (e.g. the card's CA Public Key was not available).

Error Codes.

All errors detected during testing have been allocated a unique error code. This simplifies the diagnosis of problems, allows flexible reporting methods to be used and facilitates error suppression. Whenever an error or observation is displayed on the summary screen or the test report, their error codes are displayed as well as their description.

An error code is made up of three fields as follows:

The first character denotes the specification or rule set that the card has violated, allocations are:

"1" ISO

"2" EMV

"3" VSDC

"4" M/Chip

"5" J/Smart

"6" Amex

"7" Scenario (i.e. user defined criteria)

Code "8" is used for errors that do not fall into any of these categories, e.g. other payment schemes.

The second third and fourth characters make up the error identity, a list of the messages associated with each code may be found in the file "scripts\errcodes.txt"

The remainder of the code denotes the data object or function that the error is concerned with; the first character of this field (i.e. the fifth character of the error code) denotes the meaning of the rest of the field.

"T" is followed by a two byte or four byte hexadecimal tag value.

"D" is followed by two tag values separated by the letter "T" (this is used where the error is caused by interaction between two tagged objects).

"S" is followed by a three digit number which refers to a string found in the file "scripts\errstrings.txt" (these are used for data objects or card functions for which no tag is defined) There is a special case, "S000", which is used where no additional information is required.

Sometimes an error code will be appended with a "note", this is identified by the letter "N" followed by a three digit number. Notes provide further explanation or clarification where the simple error message is inadequate.

Examples

2200T5A - The primary account number, mandatory under EMV rules, was missing from the card.
5012TDF4F - The JCB proprietary data object, "JIS2 Equivalent Data", was encoded with the wrong length.

The codes are used to identify both errors and observations.

Error Suppression

Selected errors may be designated for suppression by entering a suppression list in the box provided in the scenario stage editor.

Each item in the list is a "regular expression" which is used to identify the error code, and is optionally followed by the letter "I" (for Ignore) space separated from the regular expression.

Under normal operation if the regular expression matches the error code then that error will be shown on the certificate screen as a "Suppressed Error" and will not give rise to a test failure. If the code applies to an observation then there will be no effect. If the optional "I" is appended then the matched error or observation will completely disappear from the summary screen.

NB. Error suppression only affects the summary screen, the errors will still be identified on the results tree, report and engineers log.

Regular Expression Syntax

Regular expressions are a widely used syntax for string matching, the full features are complex and powerful, but for the purposes of CPT/CAT 3000v3 error suppression only the simpler features are likely to be needed. Detailed descriptions may be found in manuals on Tcl, Perl, Unix and many other languages. A brief introduction follows.

In its simplest form a regular expression may simply be the string that is being sought. e.g. 2200T5A will exactly match the first example shown above. This is the method which is likely to be used most often for error suppression, i.e. a list of fixed error codes.

To allow more sophisticated code matching a number of special characters may be used within the expression i.e.

'.'	matches any single character.
'^'	matches the beginning of the string
'\$'	matches the end of the string
'*'	matches zero or more of the preceding item
'+'	matches one or more of the preceding item
'?'	matches zero or one of the preceding item
'[]'	enclose a list or a range of possible match characters

Examples

^3.+ will match any code starting with "3", i.e. all VSDC related errors
....T5F20 will match any error associated only with tag 5F20, the cardholder name.
5F20\$ will do the same thing
^[3-6].+ will match any code beginning 3, 4, 5, or 6,
i.e. anything related to a major payment scheme rule.

Editing the Terminal Simulator

The EMV Terminal Simulation Stage Editor provides the means to set up the simulated terminal to condition the test transaction according to the requirements of the test.

The tabs are arranged to generally group the setup items in a logical sequence.

Terminal Setup

This tab is primarily concerned with setting up the nature of the terminal being simulated, i.e. the type of hardware, country and currency settings, etc.

Transaction Setup

Specifies details about the simulated transaction. Such things as the value and nature of the transaction can be defined.

Transaction Procedure

Specifies items concerned with the decision flow of the transaction, such as whether it is accepted or declined by the terminal.

User Defined and Proprietary Data

It is often required to issue special commands (i.e. Get Data and/or additional Read Records) to access scheme or issuer proprietary data, or data that is accessible by the terminal but not covered elsewhere in the scenario. These items may be set up here.

Host Simulation

This feature is an option on the CPT 3000v3 but fitted as standard to the CAT 3000v3, the tab will not be displayed if the option is not available. On this tab it is possible to define the master derivation keys (MDKs) for the card, and to configure a simulated online referral and issuer script processing.

Error Suppression

This feature allows user to list error codes that should not cause the final test result to “Fail” or that need not be reported at all.

Terminal Setup

The terminal simulated hardware configuration is set up using a number of editor items.

The screenshot shows the 'EMV Terminal Simulation' window with the 'Terminal Setup' tab selected. The window has a search bar at the top with 'First', 'Next', and 'Match case' buttons. Below the search bar is a tabbed interface with 'Terminal Setup', 'Transaction Setup', 'Transaction Procedure', 'User Defined & Proprietary Data', and 'Host Simulation'. The 'Terminal Setup' tab contains the following fields:

- Terminal Configuration: Contact (dropdown)
- Terminal Capabilities: (text field)
- Additional Terminal Capabilities: (text field)
- Terminal Type: 23 Attended merchant offline (dropdown)
- Terminal Transaction Qualifiers: (text field)
- Data Authentication Support: SDA, DDA and CDA (dropdown)
- Payment Scheme for Application Selection: Any (dropdown)
- EMV Common Payment Application (CPA): ☐
- AID for Application Selection: (text field)
- Presence of Specified AID above is Optional: ☐
- Use Partial Name Selection: ☐
- Verify Selection of all AIDs in PSE: ☐
- Always Use Candidate List Selection: ☐
- Terminal Country Code (9F1A): 826 United Kingdom (dropdown)
- Transaction Currency Code (5F2A): 826 Pound Sterling (dropdown)
- 5F36 Transaction Currency Exponent (Hex): (text field)

Below these fields is a section titled 'Card data may be cached for subsequent comparison with another set of data' with a text field 'Name for the data cache' containing 'ContactData'.

At the bottom, there are checkboxes for 'Enable Main reader' and 'Enable Secondary Reader', a 'Selected Reader' dropdown showing 'CT - Magtek IS350/IS65 / Contac', and buttons for 'Help', 'Save As', 'Save', and 'Close'.

Terminal Configuration

Sets up the type card interface.

This item is only available if the Contactless Testing Option is enabled.

NB. Since many contactless readers interface to the CPT 3000v3 via the Windows PC/SC interface it is not always possible for the software to automatically distinguish a contact card from a contactless card, using a contactless reader with the "Contact" setting, or vice versa, may lead to erroneous test results.

Contact

The test will be performed assuming the card is being accessed via direct electrical connection to

the chip. In this mode full EMV facilities are always available, subject to other settings in the terminal simulation.

Contactless Full EMV.

Contactless Mag Stripe.

For both of these settings the card is assumed to be contactless. Contactless payment applications can be "Full EMV" in which case the processing is similar to a contact card, or they can be "Mag Stripe", in which the card terminal at the point of sale appears to the rest of the world like a magnetic stripe reader. This magnetic stripe simulation, for which the payment schemes have a variety of names, allows chip cards to be used where the infrastructure only supports magnetic stripe data. Many cards support both options so that full EMV is used where the terminal supports it and magnetic stripe mode is used in other terminals. In order to test the magnetic stripe mode of a card supporting both options it is necessary to simulate a terminal that only supports magnetic stripe mode.

NB. When testing Visa or PBOC contactless cards the Terminal Transaction Qualifiers must also be set up, see below.

Terminal Capabilities

A three byte binary bit field. Enter the value as six hexadecimal digits, the coding of the bits is shown in EMV Book 4 Annex A2.

Additional Terminal Capabilities

A five byte binary bit field. Enter the value as ten hexadecimal digits, the coding of the bits is shown in EMV Book 4 Annex A3

Terminal Type

A two digit decimal code representing the terminal type, see EMV Book 4 Annex A1 for a list of permissible values.

Terminal Transaction Qualifiers

A four byte binary bit field, only relevant when testing Visa or PBOC contactless cards. Enter the value as eight hexadecimal digits or use the bit definition dialog, the coding of the bits is shown in Visa Contactless Payment Specification. Recent Discover cards also use this feature, with slightly different bit allocations.

Data Authentication Support

This drop down list defines the way in which Dynamic Data Authentication is performed by the simulated terminal.

EMV states that in any transaction the highest grade type of authentication that is supported by both the card and the terminal must be performed. In most cases cards that support CDA also support DDA. However terminals are not permitted to use both methods during the same transaction and many CDA capable cards will refuse to perform CDA if DDA has already been performed. Thus, if a card fails to correctly perform CDA after DDA it does not necessarily mean there is anything wrong with the card, in this case it will be necessary to test the card twice in order to verify that both methods work.

Using this drop down box it is possible to set the simulated terminal to support any combination of SDA, DDA and CDA, from none at all, to all three.

Note that when both DDA and CDA support are chosen, if the card supports CDA then a CDA test will be performed, DDA will not be attempted. In this case, if it is required to force a DDA test, then it is necessary to select a combination that does not include CDA, in this case if the card supports DDA then a DDA test will be performed, CDA will not be attempted. If it is required to test both DDA and CDA on a particular card then it will be necessary to perform multiple transactions.

With the terminal set to "SDA Only" - then no dynamic authentication will be attempted.

Payment Scheme and International Settings

Payment Scheme for Application Selection

When the test is being set up for a specific payment scheme, the scheme can be set up here. This restricts the AIDs in the selection list to those associated with the chosen payment scheme. The major schemes (Visa, Mastercard, JCB and Amex) are covered and a final "Others" choice contains a list of smaller schemes (Link, Interac, SPAN, ABI, etc.)

AID for Application Selection

When the test is being set up for a specific application, the AID can be set up here. If this value is set up then that AID and only that AID will be selected for the test, regardless of whether it can be found in the PSE or PPSE (where present) or whether it is a known EMV compliant application. This setting also overrides the Payment Scheme setting above.

Always Use Candidate List Selection

When the card does not have a PSE or PPSE the script uses candidate list selection to find the working application(s). The candidate list is set up according to the "Payment Scheme for Application Selection" setting above. If this box is checked then candidate list method will be applied even if a PSE or PPSE is present, enabling applications not in the PSE to be identified.

Terminal Country Code

The drop down list covers all the currently defined countries where a terminal might be located, select the country in which the simulated transaction is to be deemed to have taken place. The list may be updated by the user to add new or changed country codes, or to remove countries not required to be simulated (and hence make editing slightly quicker) by editing the file "countryc.txt" in the scripts folder. If no country is selected the script will use the default country code, the tool is shipped with this value set to United Kingdom, but this can be changed to any desired country using the tool's default settings editor.

.

Terminal Currency Code

The drop down list covers all the currently defined currencies in which a transaction might be taking place, select the currency the simulated transaction is to use. The list may be updated by the user to add new or changed currency codes, or to remove currencies not required to be simulated (and hence make editing slightly quicker) by editing the file "currency.txt" in the scripts folder. If no currency is selected the script will use the default setting, the tool is shipped with the default set to pounds sterling, but this can be changed to any desired currency using the tool's default settings editor.

NB. No checking is carried out that the terminal currency and country codes match, any currency

can be used in any country. The data analyser script, however, does do a check on the Application Currency and Country Codes in the card to ensure that they are a valid pairing, it is important to avoid confusion between the terminal and card country and currency codes.

Data Cache.

Sometimes with multiple application or dual interface cards it is desirable to compare the data extracted from different applications. Normally when a new terminal simulation is run all the data from a previous transaction is deleted and overwritten, so when the data will be required later then it must be cached, that is, set aside in a separate store. The label entered here can be referred to in subsequent data comparison test stages.

Transaction Setup

This tab is concerned with the quantitative data associated with the simulated transaction.

The screenshot shows the 'EMV Terminal Simulation' window with the 'Transaction Setup' tab selected. The window has a title bar with a question mark and a close button. Below the title bar is a search bar with a dropdown arrow, and buttons for 'First', 'Next', 'Match case', and a refresh icon. The main area contains a tabbed interface with 'Terminal Setup', 'Transaction Setup' (selected), 'Transaction Procedure', 'User Defined & Proprietary Data', 'Host Simulation', and 'Error'. The 'Transaction Setup' tab displays a list of transaction data fields with corresponding input boxes or dropdown menus. The fields are: '9C Transaction Type (Hex)' (text box), 'Account Type' (dropdown menu showing '00 Unspecified'), '9F02 Amount Authorised (Decimal cents or equivalent)' (text box), '9F03 Amount - Other (Decimal cents or equivalent)' (text box), '9F15 Merchant Category Code' (text box), '9F4E Merchant Name and Location' (text box), '9F53 MasterCard Transaction Category Code (Text)' (text box), '9A Transaction Date (today if left blank)' (text box), '9F37 Unpredictable Number (Hex or "RAND")' (text box), '9F6A PayPass Unpredictable Number (Decimal or "RAND")' (text box), '9F7A VLP Terminal Support Indicator (Visa)' (text box), and 'Additional Items (Tag Value Tag Value ...)' (text box). At the bottom of the window, there are buttons for 'Close', 'Help', a 'Selected Reader' dropdown menu showing '1 - Contact Simulator', and buttons for 'Save As' and 'Save'.

Transaction Type

Tag 9C, see in EMV 4.1 Book 3, page 147 and ISO 8583:1987

Amount Authorised

The value of the sales transaction in terminal currency units

Amount - Other

The value of any cashback in terminal currency units

Merchant Category Code

Tag 9F15, see in EMV 4.1 Book 3, page 140 and ISO 8583:1993

Merchant Name and Location

Free text describing the virtual merchant conducting the simulated transaction.

Mastercard Transaction Category Code – Rarely used, see Mastercard specifications for details.

Transaction Date

The date of the transaction, if this field is left blank then the current date is taken from the PC's clock

Unpredictable Number

PayPass Unpredictable Number

Enter a fixed value or the word "RAND". In the latter case a different pseudo-random value will be generated for each test. If the field is left blank then a default fixed value will be used.

Additional Items (Tag Value Tag Value ...)

Any information supplied by the terminal to the card, and not covered by the above, may be entered here. The tag for each item is followed by its value, in hexadecimal. All values must be in hexadecimal format.

Transaction Procedure

This tab sets up options affecting the decision processes within the transaction.

The screenshot shows the 'EMV Terminal Simulation' window with the 'Transaction Procedure' tab selected. The window has a title bar with a search field, 'First', 'Next', 'Match case', and a back button. Below the title bar is a tabbed interface with 'Terminal Setup', 'Transaction Setup', 'Transaction Procedure' (selected), 'User Defined & Proprietary Data', and 'Host Simulation'. The 'Transaction Procedure' tab contains various settings:

- Required Magnetic Tracks:** A dropdown menu set to 'Tracks 1 and 2'.
- Issue Warm Reset Before Test:** An unchecked checkbox.
- Always Eject Card (For multi-stage testing):** An unchecked checkbox.
- Manual PIN Entry (If card supports PIN):** A checked checkbox.
- Mask PIN Entry Box:** A checked checkbox.
- Automatic PIN Entry (If card supports PIN):** An empty text input field.
- Repeat PIN Entry after Failure:** A checked checkbox.
- Force PIN Entry when card does not support PIN:** An unchecked checkbox.
- Where both Plaintext and Encrypted PIN are supported:** A dropdown menu set to 'Submit Both'.
- Transaction Completion (Request from Terminal):** A dropdown menu set to 'Accept Offline (TC)'.
- Generate AC Command:** A dropdown menu set to 'Normal Operation'.
- Terminal Decline at 2nd Generate AC:** An unchecked checkbox.
- 8A Authorisation Response Code (Text, default Y3):** An empty text input field.
- ARC Used for ARPC Generation (use 8A when blank):** An empty text input field.
- ARPC Response Code (MasterCard Only):** An empty text input field.
- Card Status Update:** A text input field containing '83820000'.
- Force Internal Authenticate in non-DDA card:** An unchecked checkbox.
- Support Terminal Velocity Checking:** An unchecked checkbox.
- Do not retrieve transaction Logs:** An unchecked checkbox.

At the bottom of the window, there are checkboxes for 'Enable Main reader' and 'Enable Secondary Reader', both of which are checked. To the right of these is a 'Selected Reader' dropdown menu set to 'CT - Magtek IS350/IS65 / Contac'. Further right are buttons for 'Help', 'Save As', 'Save', and 'Close'.

Required Magnetic Tracks

When a card is inserted the system will always attempt to read all magnetic tracks that the reader is capable of reading. Sometimes the intake read may fail on one or more tracks (this is much more common when using a manual insertion reader than with a motorised device), in which case another attempt will be made on the exit stroke. This setting informs the terminal simulation script which tracks are required by the subsequent data analysis stages and so can prevent unnecessary ejection/reinsertion operations in the middle of multi-application tests.

Issue Warm Reset Before Test

Under some exceptional circumstances it may be necessary to give the card a warm reset in order to access the EMV application, in these cases this box should be checked.

PIN Entry options:

1. Manual PIN Entry: If the card supports offline PIN and this item is checked then the test will pause and request a PIN from the operator. If no PIN is entered or if the Cancel button is clicked on the PIN entry dialogue then the PIN test will be skipped.
2. Mask PIN Entry Box: If this box is checked then the PIN entry dialog will mask the digits being entered, if not then the PIN will be visible on the screen and in the test results.
3. Automatic PIN Entry: This item is useful if a number of cards are to be tested which are all known to have the same PIN, or if the same card is to be tested many times. A PIN entered in this box will be submitted to the card without reference to the operator.
4. Repeat PIN Entry after Failure: This box enables PIN entry to be repeated after a failure, as it would in a normal transaction. Retries will be allowed until the PIN try counter expires, the operator cancels PIN entry, or a correct PIN is entered. If it is not checked then the test moves on after one PIN entry, regardless of whether or not it was successful.
5. Force PIN: If checked, the PIN will be verified even if it is not supported by the card.
6. Plaintext/ Encrypted PIN: The selection box allows tester to choose whether plaintext/ encrypted or both types of PIN should be verified if card supports. There are certain cards in the market which do not allow the submission of both plain and encrypted PIN in a single transaction.

Transaction Completion (Request from Terminal)

This drop down list provides four choices for the transaction completion request submitted to the card in the first Generate AC :

Accept Offline - (TC) The terminal requests immediate acceptance of the transaction.

Reject Offline - (AAC) The terminal requests the transaction be declined.

Request On line - (ARQC) The terminal requests on line clearance from the issuer.

Generate AC command

This set of options defines how the terminal handles the Generate AC command(s).

- Normal Operation - The process flow depends upon the card's response to the first Generate AC, see EMV 4.3 Book 3 Chapter 9 for details of what may and may not occur.
- Never Issue 2nd Generate AC – the transaction stops after the first Generate AC even if an ARQC was received from the card. This is normal behaviour for a contactless application.
- Skip Generate AC - Terminate the transaction at this point without issuing any Generate AC commands. Useful to prevent inadvertent enabling of the contactless interface when it has been deliberately disabled for security in the postal system.
- Always Issue 2nd Generate AC – A second Generate AC command will be sent to the card even if the first Generate AC response was not an ARQC.
- Perform CDA on 2nd Generate AC –
- Perform CDA on Both Generate ACs – Normally CDA is only performed on the first Generate AC, these two options modify the behaviour.

NB. A CPT 3000v3 cannot go on line to a real issuer host, but a system with host simulation activated can simulate such a referral if the card's secret keys are available either within the scenario (test cards only, see Host Simulation) or in an HSM attached to the PC.

Authorisation Response Code

When an offline transaction is being simulated, which is the normal situation with a CPT 3000v3, then this code should be one of those listed in EMV 4.1 Book 4 section A6. When Host Simulation is being used in a CAT 3000v3 then the ARC is that which would be sent back from the simulated issuer host. "00" is the normal online accept code.

Force Internal Authenticate

It is sometimes desirable to force the tester to send an Internal Authenticate command when it otherwise would not have done. This check box enables that option.

Support Terminal Velocity Checking

Check this box to simulate a terminal that supports Terminal Velocity Checking. The Get Data commands for the LCOL, UCOL, LOATC and ATC will be issued before the first generate AC.

Terminal Velocity Checking

When this option is selected, the tool will retrieve tags from the card related to Terminal velocity checking using get data command if they were not available in the records identified by the AFL.

Do not retrieve Transaction Logs

When selected, tool will not send get data and read record commands which are specific for transaction log retrieval. Time spent in retrieving transaction logs as well as analysing them can thereby be saved.

Proprietary and User Specified Data

This tab is concerned with the extraction of data from the card beyond that which is made available via the Read Record commands defined by the AFL.

Many cards contain data in the scheme proprietary area, and other data specific to individual issuers. The way in which this data is extracted by the terminal varies from scheme to scheme and is not always well specified, so there may also be variation from chip version to chip version.

AID for Proprietary Data

Normally the proprietary data is defined by the AID used to select the application, but occasionally cards have been manufactured where the proprietary data is configured for a different payment scheme, if this is the case then the alternative AID may be entered here. If this is left blank then the proprietary data will be treated according to the selection AID.

Use Test CA Public Keys for ABI Cards

This check box is only relevant when testing ABI cards. In all payment schemes test cards are issued using test RSA keys so that they will always fail data authentication in a live

terminal. Most schemes allocate specific CA public key indices to test keys which are outside the range of indices used for live keys. ABI, however, use the same indices for live and test keys so it is necessary to inform the test tool which set of public keys to use. When this box is checked the test keys will be used, if it is unchecked then the live key set will be used.

PayPass M/Flex Co-Application

Drop down menu allows selection of the contact application present on a PayPass M/Flex dual interface card.

When to use Get Data to Extract Private and Proprietary Data

Most proprietary data is extracted from the card using the Get Data command, and some cards are sensitive to when these Get Data commands are issued. This drop down box offers a choice of points in the transaction when the Get Data commands will be attempted, as a general rule of thumb Mastercard react best when the Get Data are issued before Get Processing Options, Visa cards prefer them after the Generate AC, but this is not universal. If problems are experienced obtaining the proprietary data, of odd card behaviour is observed, then changing this setting might help. Real world terminals will not issue these Get Data commands so there are no repercussions with live cards.

When to Issue Get Data for CPLC

Card Production Life Cycle is a special case, although its tag is in the issuer/scheme proprietary range, it is defined on a regional basis. It is very common in Europe, very rare in Asia. The Get Data command for CPLC is usually available outside the payment application, so this drop down list provides a different choice of points in the transaction.

Get Data Scan for Private and Proprietary Data

This choice defines which tags are requested in the series of Get Data commands. The most commonly used will be the scheme proprietary ranges, which scan the tags defined by the payment scheme administering the card under test, this selection can be refined below. If it is not known what private and/or proprietary data is on the card and/or available via Get Data, the test script can issue Get Data commands for the whole range of tags just to see what comes back, there are two options, just the EMV allocated proprietary area, or the whole ISO defined private area as well. This 'global' scan can take a long time and may be interpreted by some cards as a security threat, causing them to shut down, so should be used with care.

Visa and Mastercard Proprietary Areas

These lists allow the Get Data scan to be confined to areas of interest within the scheme data area. These selections are only used if the above selection is for scheme proprietary data and the card is subject to the payment scheme in question.

Visa also requires that user specifies the Visa version to enable correct retrieval of primitive and constructed data objects, and the analysis of various data. If not known, user can set these settings to "Auto" so that the tool can analyse based on its interpretation of the application version. However, the "auto" function is unreliable because it is often not possible to determine precisely the version of the spec in use of a particular card.

NB. Not all the schemes have these choice available, often the list of tags is simple enough to be defined precisely by the script.

Magnetic Stripe data tab

Data formats and CVC/CVV offsets can be specified for correct interpretation of Magnetic stripe data, if not specified defaults will be used.

PURE Functions tab

Enables and customises PURE specific functions. PURE is a proprietary card platform supplied by Gemalto™, and full support for it is an optional module, this tab will only appear if the PURE option has been installed.

User Defined Data tab

Tags for Additional Get Data Commands

This section allows the test script to access data outside the range of tags defined by EMV and the payment schemes. These Get Data commands will be issued at the end of the proprietary data scan or at the point in the transaction defined by the list above if no proprietary data scan is called up. The data entry box should contain a list of the desired tags separated by spaces (e.g. DF10 DF20 DF45). Note that if the data has already been obtained by one of the built-in Get Data commands or in a record defined by the AFL then its tag will be skipped in this phase.

The verification of the values of these items is defined in the data analyser User Specified Data tab.

Additional Read Record Commands

Sometimes additional data is available via Read Record commands outside the range defined by the AFL. Where necessary these records can be defined here. The data entry box should be a list of references separated by spaces, each reference is an SFI and a record number separated by a comma (e.g. 12,2 12,3 14,6). The Read Record commands will be issued at the end of the transaction immediately before removing power from the card. If transaction logging is supported then the Read Record commands for that will be dealt with automatically, there is no need to place them here.

Names for Additional Tags

This box on this tab allows names to be provided for any tags not already defined within the system. This box has no effect on the test itself, it is merely used to provide a user friendly name for the data item to be displayed alongside its tag in the test results. It is a list of tags and names separated by spaces, with quotes used to delimit names containing space characters (e.g. DF97 "New Fictional Template").

PAN is Displayed on Rear of Card

Sometimes, if the application under test is not the primary application on the card, the PAN may be different to that embossed on the front of the card, and may be printed on the rear. Checking this box will enable the rear view of the card on the summary screen and place the PAN in the appropriate place just under the signature panel.

Responses tab

This tab deals with expected transaction decisions from the card, unlike the other settings in the terminal simulation these define expected response from the card, not data to be applied to the card.

First Generate AC Decision (From card)

This setting applies a pass/fail criterion to the transaction decision returned by the card in the first Generate AC response. Note that this check is carried out in addition to the normal EMV rules verification which ensures that the card does not, for instance, accept a transaction when the terminal requests a decline.

Second Generate AC Decision (From card)

This setting applies a pass/fail criterion to the transaction decision returned by the card in the second Generate AC response (where performed). Note that this check is carried out in addition to the normal EMV rules verification which ensures that the card does not, for instance, accept a transaction when the ARC demands a decline

Mastercard Advance functions

Commands for Special functions defined by Mastercard Advance specification can be enabled using the options provided in this sub tab. Supported commands:

- Put Data for Unprotected tags which do not require cryptographic checksums
- PIN change without cryptographic checksums

Host Simulation

Host Simulation is an optional feature in CPT 3000v3 and may not be present on all systems. It is standard on CAT 3000v3. If the option has not been purchased then the host simulation terminal stage editor tab will not appear.

When host simulation is activated the test script verifies the Application Cryptogram(s) and allows the test transaction to simulate an online situation at the terminal by returning a valid ARC from a simulated host and performing Issuer Script Processing.

NB. Software Host simulation is only possible on test cards where the Master Derivation Keys are known and are available to be loaded onto a PC.

The screenshot displays the 'EMV Terminal Simulation' window with the 'Host Simulation' tab selected. The window has a search bar at the top with 'First', 'Next', and 'Match case' buttons. Below the search bar is a tabbed interface with 'Terminal Setup', 'Transaction Setup', 'Transaction Procedure', 'User Defined & Proprietary Data', and 'Host Simulation'. The 'Host Simulation' tab contains sub-tabs: 'Symmetric Cryptography Keys', 'Symmetric Cryptography Options', 'Issuer Script Processing', and 'PW Verification'. The 'Symmetric Cryptography Keys' sub-tab is active, showing fields for 'Encryption Engine to Use' (set to 'Software'), 'Symmetric Key Library Selection' (with a dropdown menu open showing options: 'None', 'Determined by BIN only', 'Determined by BIN and DKI', 'Determined by DKI only', 'Use Fixed Index below'), and 'Index'. Below these are fields for 'Master Derivation Key (Cryptogram)', 'Master Derivation Key (Confidentiality)', 'Master Derivation Key (MAC)', 'Master Key (CVV/CVC1)', 'Master Derivation Key (CVC3/dCAV)', and 'Mobile PIN (MCBP HCE)'. A checkbox 'Above Keys are UDKeys' is present, and a dropdown 'UDK Derivation Method (EMV Book 2 A1.4)' is set to 'Always use Option A'. A note at the bottom states: 'NB. Authorisation Response Code etc. is set up on the Transaction Procedure tab'. The bottom of the window has a 'Help' button, checkboxes for 'Enable Main reader' and 'Enable Secondary Reader', a 'Selected Reader' dropdown set to 'CT - Magtek IS350/IS65 / Contact', and 'Save As', 'Save', and 'Close' buttons.

To carry out host simulation on a live card a much more secure environment is needed, such as an HSM or an online connection to a secure host. Support for some HSM types is available, please contact Barnes International for more information on connecting to these devices.

Do not select an HSM type from the drop down list if a connection to that HSM is not available, the test script will time out with an error if the selected HSM cannot be contacted.

Symmetric (DES) Keys

In order for host simulation to work the Master Derivation Key(s) (MDK) must be provided, for cryptogram verification only the MDK(ac) is required, the others are only needed if issuer script processing and/or CVV verification is to be performed.

The keys may be entered either directly into the stage editor, using the boxes provided, or via an index file. The index file is a text file which is named "DesKeyList.txt" and resides in the scripts folder. Each line of the index file consists of four or more space separated fields, the first field is the index, the remaining three or more are the keys in the order they appear in the stage editor that is cryptogram, confidentiality and MAC, optionally followed by CVV/CVC1 and CVC3. Then it is only necessary to enter the index string in the box at the top of the stage editor tab. The index can be any sequence of characters that does not include spaces, and must exactly match an index field in the key index file.

NB. If any key is missing but subsequent keys are provided then the missing key(s) should be entered in the index file as a string of zeroes, this will be interpreted by the script as a null key

From software version 3.47 there is a GUI available for editing the key file, please refer to the main software manual for more details.

Symmetric Key Library Selection

The method by which a key set is chosen from the index file can be defined here.

1. **Determined by BIN Only** Look for an index in the key file that matches the BIN of the card under test.
2. **Determined by BIN and DKI** Look for an index in the key file that matches the BIN of the card under test concatenated with its DKI. This allows testing of cards where a number of different keys are used for cards bearing the same BIN
3. **Determined by DKI Only** Only Derivation Key index is used for lookup
4. **Use Fixed Index Below** Any string can be entered in the dialogue below the selection box, this string will be used to lookup the key set.

If any keys are entered directly in the scenario stage editor they will take priority over the key set defined by the index field (if any).

The test script will derive the card's Unique Derivation Keys (UDK) from the provided MDKs and generate the appropriate session keys from them. Note, sometimes test cards are provided with their UDKs rather than their MDKs, in this case the letter "U" placed at the end of the line in the index file, separated from the last key by a single space, will cause the UDK derivation stage to be skipped. Where keys are provided directly in the scenario a check box is available to indicate that the keys are UDKs, this setting is global, it is not possible to mix UDKs and MDKs in the same test (except for the CVV/CVC1 key, which is never derived).

Cryptogram Verification

When the card responds to a Generate AC command the cryptogram returned will be verified against a cryptogram calculated by the test script. The engineer's log will show the working of the calculation, including UDK and session key derivation, and the plaintext data used to generate the cryptogram

External Authenticate

If the response from the first Generate AC was an ARQC and the card supports the External Authenticate command the script will calculate an ARC and pass it to the card in an external authenticate command.

ARPC Response Code

Mastercard do not normally support Issuer Authentication in the AIP. This is a slight misnomer in that the cards actually support Issuer Authentication but do not use the External Authenticate command, instead the response cryptogram is passed to the card in the second Generate AC command. When this occurs an ARPC response code is also required, please refer to the current Mastercard specifications for the format of this item.

Issuer Script Processing

The available script commands are PIN Unblock, Application Block, Application Unblock, PIN Change and Put Data. Each of the first three requires no data and is enabled by a check box in the stage editor tab.

PIN Change can be selected by entering a new PIN in the box provided. When a new PIN is present the PIN Unblock check box is ignored. The PIN Change command will always use the form that does not require the use of the existing PIN.

Put Data is enabled by placing one or more "tag value" pairs in the box provided. e.g. "9F14 04" will set the LCOL to 4, "9F14 04 9F23 08" will set LCOL to 4 and UCOL to 8. (The quotes are for readability, do not type them into the stage editor). The value fields are hexadecimal characters, two per byte, which represent the entire value field, so that "000000500000" will write a six byte value, and "500000" will only write three bytes. Each tag is separated from its value by one or more spaces, and the next tag follows, also separated by one or more spaces, do not put spaces in the middle of the values.

CVV/CVC Value Checking.

If there is a CVV key present in either the index file or the scenario then the CVV/CVC1 value on the magnetic stripe and the iCVV/CVC2 value(s) in the chip will be verified using it.

Dynamic CVV

Contactless cards operating in magnetic stripe image mode produce dCVV or CVC3 values dynamically for each transaction. dCVV (Visa) uses the same key as the cryptogram, so this verification will be made if the AC key is available. Mastercard's CVC3 value requires an additional key, and so will only be verified if this key is provided.

NB. If the CVC3 key is provided but the CVC1 key is not then the CVC1 key should be entered in the index file as a string of zeroes, this will be interpreted by the script as a null key.

PVV Verification

If the PVV key is available, and if the PIN is known, then the PVV on the magnetic stripe and equivalent data can be verified. Where the key is defined a correct offline PIN entered during the chip test will be used for this verification, if no PIN was entered, or if the card rejected the PIN, then the script offers the operator the opportunity to enter a PIN for PVV verification purposes.

In addition, there is a facility to re-write the magnetic stripe with a new PVV following a successful PIN change (see above). Obviously this is only possible if the card reader has the capability to write to the magnetic stripe, please contact Barnes International for more information if this facility is required.

Editing the Data Analysis Stage

The EMV data analysis scenario stage editor includes tabs for general rules checking setup, for specific issuer requirements setup, and for verification of the payment scheme proprietary data.

For information on the common buttons and controls see Scenario Stages above. The available options for the auto populate selection currently are:

Current Card Data: This option will populate the analysis stage with data that matches the card data currently in the system. This will be the most recent card tested, imported, or recovered from the saved card database.

VPA File: If the VPA option has been purchased then an output file from the Visa Personalisation Assistant package may be used to generate the data for the analysis stage.

Other data sources may be added from time to time, or custom made for individual clients, please contact Barnes International for details.

Data Analysis Options

The first page contains the general data analysis options.

Specific Issuer Requirements

Subsequent pages allow specific values to be defined for the card data.

Data Analysis General Options

This page is concerned with options of a general nature.

Verify Data Against EMV Rules

When this item is checked the card's data will be checked for compliance with the EMV rules.

Payment Scheme Rules Verification

Use this drop down list to select the payment scheme of interest. Either select a specific payment scheme or use one of the two special cases, "None" means no payment scheme rules verification will be carried out at all, and "Auto" indicates that the script will be guided by the AID of the selected card application.

Note that in the case of contact cards all the payment schemes specify a fundamental requirement that the card must comply with EMV, in these cases the EMV verification will be called up regardless of the setting the check box above. Some contactless schemes deviate considerably from EMV.

ISO Tracks

Use this drop down box to select which, if any, magnetic tracks will be compared to the chip data. Note that the magnetic track data will always be displayed if it is available, this selection only determines whether or not it will be tested for consistency with the chip data.

Support Visa iCVV, Mastercard Chip CVC, or JCB CAV3

There are three options in this list:

Mandatory - requires that discretionary data in the magnetic stripe must differ from the equivalent chip data by a maximum of three bytes, the card will fail the test if the discretionary data does not differ between the magnetic stripe and the chip.

Forbidden - requires that the magnetic stripe and the chip data must match exactly.

Don't Care - removes any requirement.

When host simulation is activated, and the CVV key is available, this setting also affects the calculation of the expected values.

Require Manual Entry of PAN

It is sometimes desirable to verify that the operator has visually checked that the embossing on the card matches the data encoded on the chip and/or mag stripe. This can be achieved by requiring the operator to enter the PAN from the card as a record. The prompt will appear after the card has been ejected from the reader. The information entered by the operator is checked against the data read from the card. NB. It will normally be necessary to also check the "Always Eject Card" check box in the terminal simulation stage to ensure that the card is ejected before the prompt is displayed to the operator.

Specific Issuer Requirements

Most of the analyser editor pages are concerned with the definition of the requirements for individual card data items. For full information of the use and meaning of these items please consult the EMV 4.3 specification and/or the specifications provided by the appropriate payment scheme.

Note that these settings do not affect the EMV or Payment Scheme rules checking set up on the general options page, setting up a requirement on these pages that contravenes an EMV or Payment Scheme requirement will not stop the card from failing the test.

Most items have edit boxes associated with them. These boxes accept either a specific value or a wildcard expression. If the box is left empty then the associated data item will not be checked and may be absent from the card, if the box has a specific value then the test will fail if the item is not present or does not contain the specified value. If the value includes wildcard characters then generally the item must be present and its value must match the wildcard expression (although there is a wildcard syntax for "may be absent"). For more information on the use of wildcards please see Wildcard Syntax below.

Items using hexadecimal data input will also accept a binary string prefixed with "Z", e.g. "Z10100101" is equivalent to "A5"

Some items have drop down lists for value selection, these lists will normally have two special case selections, "Not Required" means that the item need not be present, and "Don't Care", means the item must be present but any value is acceptable.

If the data object being defined is a bit field then a button may be provided to the right of the edit box. Clicking this button will open a dialog in which each bit of the item is identified and may be set up using a check box. For more information see "Bit Allocation View" below.

Data items resident in templates have a special syntax, see "Template Resident Tags" below.

NB. If required, there is a facility to define data items which must not be present on the card on the "User Specified Data" tab.

Payment Scheme Proprietary Data

A range of tags within the EMV data object range is set aside for issuers and payment schemes to use for their own purposes, the definition and use of these tags will vary according to the payment scheme administering the card under test.

There are one or more pages in the analyser stage editor for each of the payment schemes who have defined proprietary data. The items on these pages are only referred to by the test script when the card under test is associated with that payment scheme, the criteria from the other proprietary data pages will be ignored.

Bit Allocation View

The Bit Allocation View is a pop-up dialog that enables the required value of a bit field to be set up at the bit level. The dialog contains a tabbed page for each byte of the object, with the bits listed MS at the top, LS at the bottom. This layout matches that used in most EMV related specifications.

On first opening, the check boxes will reflect the currently set up hexadecimal or binary value in the main edit box. Any changes made using the check boxes will be transferred back to the edit box when the "Apply" button is clicked. The check boxes have three states, checked, unchecked, and indeterminate (greyed out), and will cycle round these three states with each click. The indeterminate state represents a wildcard, or "don't care" value.

There is a check box labeled "Binary" which indicates whether the setting in the edit box uses binary or hexadecimal notation. If there are no wildcard characters in use it will be preset to whatever notation was in use before the dialog was opened, and may be changed if required. This means that the Bit Allocation View dialog may be used as a convenient way for converting values between hexadecimal and binary.

If any of the checkboxes are in the indeterminate, or wildcard, state when the "Apply" button is clicked then the output will be in binary form regardless of the setting of the "Binary" check box.

When first invoked the Bit Allocation View will handle the single character wildcards ('?' and 'X') correctly but the results when using the multiple character wildcard, '*', will be unpredictable.

Template Resident Tags

CPA and VIS 1.5 define a number of data templates, each containing a number of primitive objects. The tags for these primitive objects are repeated in the various templates, so that to identify them uniquely it is necessary to use the template tag as well as the data tag. The requirements for these data objects are defined in groups.

For instance, if a requirements edit box is labeled BF37 DF1x, then it means that the template, BF37, may contain a number of primitive items tagged DF10, DF12 etc. up to DF1F. To define a requirement for these tags enter the substitute for the 'x' followed by the required hexadecimal value, repeated as many times as necessary e.g. 1 A0 2 A0 3 0F would indicate that DF11 and DF12 must each have the value A0, and DF03 must have the value 0F.

The items need not be contiguous, and the Wildcard Syntax described below can also be used for the requirements e.g. 1 * 4 A0|0F means that DF11 must be present but its value is unimportant, while DF14 can have either value, A0 or 0F.

In some cases there are two 'x's in the tag identifier, e.g. BF3F DFxx, in this case a two character identifier is required e.g. 20 * 7E 341FFFFFFFFF0000, to define requirements for DF20 and DF7E.

Wildcard Syntax

There are a number of alternative methods of inserting wildcard expressions in a test scenario, they are described below in ascending order of complexity. Please note that wildcard expressions are only valid in Data Analyser stages where the objective is to validate data received from the card. These facilities are not available in the Terminal Simulation stages because they have no meaning when defining data to be sent to the card.

1. The "Don't Care" "X" character.

When entering items in hexadecimal or binary mode the "X" character may be used to indicate that any value is acceptable for that nibble or bit. This is most useful when defining bit field data items in binary mode and is supported by the bit field check box windows.

Example:

"Z1100X001" will accept 11001001 (C9H) or 11000001 (C1H)

"CX" will accept C0H to CFH inclusive.

2. Filename Style wildcards

This method is most familiar as the syntax used when processing file names in MS-DOS and Windows, in Tcl parlance it is known as "Pattern Globbing". In this syntax the question mark "?" is used in the same way as "X" in the above method to represent a single "don't care" character, and the asterisk "*" is used to represent zero or more "don't care" characters. This method must be used in preference to the "X" method when defining text fields where "X" could be a valid character, it can also be used in hexadecimal and binary fields.

Example:

"*Debit*" will accept any string containing the word "Debit"

"1234????" will accept any eight character string beginning "1234"

"*" will accept any value

It is also possible to specify groups or ranges of characters by enclosing them within square brackets, for instance.

"[EMV] *" will accept any string beginning with 'E', 'M', or 'V'

"[1-9]" will accept any numeric digit except zero.

3. Alternative Values

The vertical bar, or "pipe" character "|" may be used to separate alternative values. Think of it as an "OR" statement. Each alternative value may itself contain wildcards. If the "|" is the first character in the requirement, then it means that it is permissible for the item to be not present, the remaining alternatives are applied only if the item exists.

Example:

"C9|C1" will accept either C0H or C1H, exactly as in the first example above.

"*Debit*|*DEBIT*" will accept any string containing either of the words "Debit" or "DEBIT".

"|[1-9]|[1-9][0-9]" will accept either nothing (item not present at all) or any decimal value between 1 and 99, but not zero.

"|201|202" will accept the item not being present at all, but fail it if it is present with any value other than 201 or 202.

4. Regular Expressions

This is by far the most powerful of the methods available in CPT/CAT, and so inevitably presents the most opportunities for making mistakes. Regular expressions are a widely used method for processing text data and are a feature of many programming languages.

If the entry in the scenario begins with the underscore character "_", then the rest of the text will be interpreted as a regular expression. Regular expressions cannot be combined with any of the above (they can do all that and much more so it's not necessary) and do not work in binary modes.

Example:

"_C9|C1" will accept either C0H or C1H, exactly as in the example above.

"_[A-Z]" will accept any string consisting of only upper case letters and space.

No attempt will be made here to teach the uninitiated the syntax of regular expressions, if you are not already familiar with it please refer to one of the many excellent tutorials available on the web and in computer bookshops.

User Specified Data

Verification for any data not covered in the preceding tabs may be defined here. Generally this tab is used to analyse items defined on the Proprietary and User Defined Data tab of the terminal simulation editor as well as tags in PSE FCI / Records.

The screenshot shows the 'EMV Data Analysis' window with the 'Miscellaneous' tab selected. The window has a title bar with a question mark and a close button. Below the title bar is a search bar with a dropdown arrow, and buttons for 'First', 'Next', 'Match case', and a refresh icon. The main content area is divided into several sections:

- EMV Open**: A section with tabs for 'JCB Proprietary Data', 'Interac Proprietary Data', 'PBOC Proprietary Data', 'PURE Proprietary Data', 'Miscellaneous' (selected), and 'Error Suppression'.
- American Express AEIPS Version**: A dropdown menu showing 'AEIPS 4.1/2'.
- Carte Bancaire Host Application**: A dropdown menu showing 'Auto Select'.
- User Specified Data**: A section with instructions: 'List the tags and required values, use "*" to signify value is unimportant' and 'e.g. '9F45 1234 9F22 *' defines a value for 9F45 and 'must be present, any value' for 9F22'. Below this are six text input fields for: 'Application Data Required, values in Hex', 'Application Data Required, values in Text', 'PSE FCI Data Required, values in Hex', 'PSE FCI Data Required, values in Text', 'PSE Directory Data Required, values in Hex', and 'PSE Directory Data Required, values in Text'.
- List the tags only for data elements that MUST NOT be present**: A section with a text input field for 'Data Objects Which Must Not be Present'.

The bottom of the window contains a 'Help' button, a 'Load Data From' dropdown menu set to 'Current Card Data', and 'Save As', 'Save', and 'Close' buttons.

Test criteria are entered as a list of tags and values separated by spaces. The wildcard characters allowed for the definition of criteria elsewhere are also allowed for the values in this tab.

There are two data entry boxes, one of which accepts data values in hexadecimal, and the other in text. Note that all tags must be followed by a value, if only a presence test is required then use a single '*' character.

e.g. In the hexadecimal box:
9F45 1234 9F22 *

The above specifies that tag 9F45 must have the value 1234H, and that tag 9F22 must be present but can hold any value.

In the text box, if the value includes space characters then use quotes (") around the whole string.

There is another data entry box for tags which must not be present, i.e. the card will fail the test if these tags are provided to the terminal. This box is a simple list of tags separated by spaces.

Application Comparison

It is sometimes necessary to compare the data between two or more applications on the same card. For instance in a dual interface (contact and contactless) card, or where a card contains a payment application and a dynamic passcode application.

The application comparison test script provides the means to carry this out.

The EMV terminal simulator script has the facility to define a data cache label, when this is activated, using the user defined and proprietary data page of the EMV terminal simulator stage editor, the card data is cached in memory so that it is not overwritten by a subsequent **different** terminal simulation stage.

NB. If the same terminal simulation scenario stage is repeated, or a different terminal simulation uses the same cache label, then the cache will be overwritten.

Several data caches can exist concurrently if a number of terminal simulation stages are defined with different cache names.

The application compare script will always use the active data from the most recent terminal simulation (regardless of whether or not it has been cached) and compare it with the cache named in its scenario.

There are only three items to set up in a comparison stage editor.

The name of the cached data to compare with, this must exactly match the cache name in a previous terminal simulation stage (not the most recent one that would be pointless). If this is omitted or if the cache does not exist then the stage will be skipped.

The iCVV support checkbox performs a similar function to that in the EMV data analyser, in that it allows the magnetic stripe image data objects to differ by up to three bytes to allow for the fact that the iCVV value will change each time the card is activated.

The list of tags to be excluded from the comparison allows the user to define data objects which exist in both applications but which are known or expected to differ. Note that the known dynamic data objects such as application cryptogram, transaction counter, etc. are excluded by default. The additional tags should be entered using upper case hexadecimal values only and be separated by spaces.

Embossing Display

When stringing together tests on a multi-application or dual interface card the embossing display on the summary screen is normally built using the data from the last terminal simulation in the sequence, sometimes this is not correct. For instance dual interface (contact and contactless) cards often use a different account number for the two applications, and due to the necessity to read a mag stripe as well the test sequence has to be performed with the contact test first. Usually in these cases the embossing will match the contact application so that a display built from the contactless data will be incorrect.

The embossing display test script provides the means to overcome this problem.

The EMV terminal simulator script has the facility to define a data cache label, this is activated by entering a unique name in the space provided on the user defined and proprietary data page of the EMV terminal simulator stage editor. When a cache name is defined the card data is cached in memory so that it is not overwritten by a subsequent **different** terminal simulation stage.

NB. If the same terminal simulation is run twice, or two terminal simulations use the same cache name, then the second one will overwrite the cached data.

Several data caches can exist concurrently if a number of terminal simulation stages are defined with different cache names.

The embossing display script is implemented as a terminal simulation script, although in fact it is closer to being a data analyser in that it does not perform any operations on the card, and only operates on previously captured data. This is done to prevent the CPT application from rebuilding the engineer's log and tree as it would if two data analysers are defined in sequence. This rebuild is unnecessary because the embossing display script puts no data into the tree.

The embossing display script builds a new summary screen from the data cache named in its scenario.

There are only two items to set up in an embossing stage editor.

The name of the cached data to build the display from, this must exactly match the cache name in a previous terminal simulation stage (not the most recent one that would be pointless). If this is omitted or if the cache does not exist then the embossing display will remain untouched.

There is an option to use the Primary Account Number from the mag stripe data instead of the cached chip data, this may be necessary if the chip data uses a different number from that encoded on the mag stripe and embossed on the plastic.

© Barnes International Limited
January 2018

support@barnestest.com