PCC SECURITY POLICY

VPM SECURITY POLICY FOR THE PCC MODULE
OVERVIEW

Revision 0.3
Dec 2000

PSI Systems, Inc.
247 High Street
Palo Alto
CA 94301-1041

Part No. VPM-D-01-017

Table of Contents

1. OVERVIEW

This document describes the security policies employed by PSI Systems, Inc. in reference to the Postal Cryptographic Coprocessor (PCC) Module within the VPM system. This document is designed to satisfy the Security Policy requirements of FIPS 140-1.

A separate and distinct Security Policy is available for non-4758-related security operations within PSI Systems, Inc. Please refer to the VPM System Overview document for details on the full system.

The Postal Cryptographic Coprocessor (PCC) discussed in this document is based on the IBM 4758 PCI Cryptographic Coprocessor. This device has been previously FIPS 140-1 Level 4 certified (certificate #35) by NIST. Thus, this document is based on the standards already established by IBM and focuses mainly on the items and features specifically designed by PSI.

2. SECURITY LEVEL

The Postal Cryptographic Coprocessor meets the overall requirements applicable to Level 3 security of FIPS 140-1. Table 1 lists the security levels for the different PCC functions.

Table 1 – Module Security Levels

Security Requirements Section	Level
Cryptographic Module	3
Module Interfaces	3
Roles and Services	3
Finite State Machine	3
Physical Security	4
Software Security	3
Operating System Security	N/A
Key Management	3
Cryptographic Algorithms	3
EMI/EMC	3
Self Test	4

3. ROLES AND SERVICES

The PCC shall support four distinct operator roles. These operator roles are:

- PSI Cryptographic Officer 1
- PSI Cryptographic Officer 2
- Account Administrator
- User

Where the "PSI" Crypto Officer designation is designed to distinguish these custom roles from those of the NIST-certified underlying IBM 4758 device. Throughout this document, Crypto Officer 1 and Crypto Officer 2 refer to PSI Systems security roles and not IBM-defined security roles.

The PCC enforces the separation of roles using identity based operator authentication. Each service requires the operator to supply a user ID, user pass phrase SHA1, and request ID. This information is validated at the beginning of each service to make sure the user has rights to the requested role and the proper password is used.

An extremely detailed matrix of services mapped against SRDI access appears in Table 2-Table 4. This matrix also specifies the allowable roles for each service. An 'x' in any role column indicates that this role has access to the associated service. A '-' in these columns indicate those services which require no authentitication.

Note that one column in the table is called "Change Account Register?". The column indicates that a given service will change USPS account registers (account values). A "Y" in this column indicates the service will modify these registers. A blank indicates that registers are unchanged or not applicable to the service.

Only two services are available without authentication, VPOZeroize and VPOGet4758Status.

VPOZeroize

This function will zeroize all BBRAM entries and global key data, returning the specified 4758 to UNITIALIZED state. A zeroized unit must be returned to manufacturing – it cannot be reconfigured to an operational state without going through the complete manufacturing process.

VPOGet4758Status

This function queries a specified 4758 and returns the current date/time in the unit, the status (e.g. PRODUCTION), and an IBM structure **sccAdapterInfo_t** that contains a host of coprocessor specific information (e.g. serial number, software versions, etc).

3.a. Cryptographic Officer 1

The PSI *Cryptographic Officer 1 Role* is the most security sensitive role and provides access to services that enable and alter the key Security Relevant Data Items (SRDI). The services that are available *only* to this role are listed below. The PSI Crypto Officer 1 also has access to lower security level services (i.e., those also accessible by the PSI Crypto Officer 2 and Administrator roles).

3.a.1. VPOFirstTrust

This is the first post-manufacturing operation. This operation requires the PSI Crypto Officer 1 to supply a manufacturing PIN code that is embedded in the firmware of the PCC. The PSI Crypto Officer 1 must also use a key diskette supplied by manufacturing that is matched to the particular PCC being initialized. This diskette contains DES and DESMAC key material that is used to encrypt the data being sent to the device.

If the decrypted PIN matches, three accompanying PSI Crypto Officer user-names and pass phrase SHA1 hashes are loaded into the devices BBRAM. The function also creates the RSA Retained Key for the unit and stores this in BBRAM. The public portion of this key is returned to the Crypto Officer 1.

3.a.2. VPOGenDSAKeyPair

This function accepts a key number and generates a 1024 bit DSA key pair that is used for indicium signing. The 20-byte private component of the key is 3DES encrypted with the current active master key. A structure containing the public key token and the encrypted private component is returned to the calling host program

3.a.3. VPOGenMasterKey

This function accepts a key number and generates 3 random 32-bit key shares. The key shares are then XOR'ed into a final 32-byte master key. The master key and original key shares are stored in BBRAM indexed by the requested key number.

3.a.4. VPOGenRSAMessagingKey

This function generates a 1024-bit RSA key pair of a specified version number in response to a Crypto-Officer 1 authenticated request. The 128-byte private "y" component of the key is 3DES encrypted with the active master key. An IBM private key token (with the encrypted "y" component substituted for the normal clear-text byte pattern) is emitted to the host.

This encrypted RSA key token is subsequently installed on the originating (and other) 4758's by using the **VPOLoadRSAMessagingKey** function.

3.a.5. VPOLoadRSAMessagingKey

This function is used by the Crypto Officer 1 to clone an RSA Messaging Key. The Crypto Officer provides an encrypted RSA key token (the private component is 3DES encrypted with a master key version) as an input to this function. See the description for **VPOEmitRSAMessagingKey** to understand how this encrypted key token is created.

The function uses the master key material on the target 4758 to decrypt the private portion of the key token. The key token is then stored in the appropriate BBRAM location for that RSA Messaging Key version.

3.a.6. VPOResetAccountMAC

This function allows a Crypto Officer 1 to re-establish the MAC on a specified end user account. It is only available to a Crypto-Officer 1, and can only be used in conjunction with a complete and formal audit of the end user's account to ascertain that the proposed corrected values are indeed "correct". In all likelihood, this function will never be used.

The function has been placed in the API to accommodate unrecoverable database corruption. In other words, a register or other MAC'ed data field in the account structure would have to be damaged to necessitate use of this function. It is required in account recovery because every other VPO function that utilizes the account information checks the MAC on the information prior to proceeding with the function.

3.a.7. VPOSetActiveMasterKeyVersion

This function establishes the current master key version number to be used in subsequent DES, 3DES and DESMAC operations in a specific 4758. A given 4758 can store up to 10 master key versions. However, only one version is "active" at a given time. This function establishes the active key version.

The rationale for this function is based on the potential need to move to new master key versions over time. The move must be accomplished with an active user base – the central servers cannot be shut down to change master keys. By separating the generating and/or loading of master key material from the setting of the "active" master key, the design permits a fluid transition from one master key version to another.

3.a.8. VPOSetDateTime

This function permits the Crypto Officer 1 to set the date and time of a given 4758. All 4758's will be operating at GMT. The function will be invoked when the unit first reaches production status, and may be invoked if significant time drift is detected on a given 4758.

The internal 4758 time is used to time stamp various transactions, and is important to replay prevention strategies.

3.a.9. VPOSetTransactionCounters

This function has been provided for the rare circumstance that a Crypto Officer 1 will need to reset one or both of the transaction counters on a specific 4758. The only rationale for this would be the case where a production 4758 had to be returned to manufacturing for some reason. In remanufacture, all of its BBRAM information would be zeroized, but the IBM serial number of the unit would be unchanged.

If the unit were returned to production, the transaction counters would start at 0 once again. This would cause a discontinuity in the numbering that appears in the audit transaction log. This function would permit the Crypto Officer 1 to set the transactions counters to those values that were last recorded from that unit. In that way, the transaction log for that particular 4758 would feature a contiguous numbering.

3.a.10. VPOUpdate Profiles

This function enables the Crypto Officer 1 to update up to 20 usernames and pass phrase SHA1 hashes stored in a specific 4758. A given user in this list can subsequently change their pass phrase (and their pass phrase only) using **VPOChange4758UserPassPhrase**.

3.b. Cryptographic Officer 2

The *Cryptographic Officer 2 Role* is a highly sensitive role and is designed to support high level and infrequent operations. The services accessible to this role (and no lower roles) are:

3.b.1. VPOEmitRSARetainedKey

This function emits the *public portion* of the unique RSA Retained Key for the associated IBM 4758 coprocessor. The function would be used in the rare event that the original emitted public key (see **VPOFirstTrust**) was lost.

3.b.2. VPOExtractMKShare

This function is used to extract encrypted shares of a specified master key from any 4758. Three key shares are stored in the 4758 BBRAM for any given master key. This function is employed by three distinct Crypto Officers (at least one of whom is a Crypto Officer 1) to securely extract the three key shares constituting a specified master key version. Each Crypto Officer presents an RSA public key to encrypt the 32-byte key share before it is emitted. The RSA encryption algorithm is used for key distribution as a commercially available public key distribution method. The public key is taken from another 4758 (the "retained RSA key") that will be receiving the key shares. The service is used in conjunction with VPOLoadMKShare to securely clone a master key from one 4758 to another.

3.b.3. VPOGetMKSHA1

This function is directed at a specific 4758 and requests the SHA1 hash of the specified master key version in that 4758. This function is a means to confirm that the proper master key material is resident in any given 4758, without revealing the key material itself.

3.b.4. VPOGetRSASHA1

This function is directed at a specific 4758 and requests the SHA1 hash f the specified RSA Messaging key version in that 4758. This function is a means to confirm that the proper RSA key material is resident in any given 4758, without revealing the key material itself.

3.b.5. VPOLoadMKShare

This function loads a single key share for a specific master key version into a specified 4758. RSA encryption algorithm is used for key distribution as a commercially available public key distribution method. The service is used by three distinct Crypto Officers (at least one of whom is a Crypto Officer 1) to load key shares previously extracted from another 4758 (see **VPOExtractMKShare**). The key share must be encrypted by the public portion of the RSA Retained Key stored on the target 4758. This allows the corresponding stored private RSA key to decrypt the key share and store it in the appropriate BBRAM location.

VPOLoadMKShare must be invoked in a specified order. Share 1 must be loaded first, followed by Share 2. When the third and final Share is loaded, the function automatically XOR's the three shares and stores the resulting master key material in the appropriate BBRAM location.

3.b.6. VPORekeyAccount

Periodically, it will be necessary to associate a new DSA key with a given account. For instance, key certificates are valid for 3 years. At the end of this period, a new DSA key pair must be associated with the account. This function provides a secure means to make this new association.

The function accepts the current ACCOUNTSTATUSSTRUCT and verifies its MAC. It then associates a new DSA Key ID with the account and re-computes the MAC. The revised account structure is returned to the host and subsequently stored in the account database.

3.b.7. VPOReMACAccount

This function detects if the MAC of the account information has been created with the *current* master key. If not, it re-computes the MAC based on the current master key.

This function will be utilized in the event that a new master key version is introduced to the system. Each MAC structure contains the master key version number reflecting the key that was used to create the MAC. This key is always used to confirm the existing MAC (even if it is not the current active master key version).

This function will confirm the existing MAC and, if the current key version differs from that used to create the original MAC, a new MAC will be computed. **VPOReMACKeyStructure**, **VPOReMACChallenge**, and **VPOReMACCreditCard** are analogous VPO functions that perform similar processes on the associated MAC's.

3.b.8. VPOReMACChallenge

This function detects if the MAC of the challenge question & answer information has been created with the *current* master key. If not, it re-computes the MAC based on the current master key.

This function will be utilized in the event that a new master key version is introduced to the system. Each MAC structure contains the master key version number reflecting the key that was used to create the MAC. This key is always used to confirm the existing MAC (even if it is not the current active master key version).

This function will confirm the existing MAC and, if the current key version differs from that used to create the original MAC, a new MAC will be computed. **VPOReMACKeyStructure**, **VPOReMACAccount**, and **VPOReMACCreditCard** are analogous VPO functions that perform similar processes on the associated MAC's.

3.b.9. VPOReMACCreditCard

This function detects if the MAC of the encrypted credit card information has been created with the *current* master key. If not, it re-computes the MAC based on the current master key.
This function will be utilized in the event that a new master key version is introduced to the system. Each MAC structure contains the master key version number reflecting the key that was used to create the MAC. This key is always used to confirm the existing MAC (even if it is not the current active master key version).

This function will confirm the existing MAC and, if the current key version differs from that used to create the original MAC, a new MAC will be computed.

3.b.10. VPOReMACKeyStruct

This function detects if the MAC of the DSA key structure information has been created with the *current* master key. If not, it re-computes the MAC based on the current master key.

This function will be utilized in the event that a new master key version is introduced to the system. Each MAC structure contains the master key version number reflecting the key that was used to create the MAC. This key is always used to confirm the existing MAC (even if it is not the current active master key version).

3.c. Account Administrator

The *Account Administrator Role* provides access to the day-to-day supervisory activities related to maintaining customer accounts and other system tables. Those services are:

3.c.1. VPOChange4758UserPassPhrase

This function allows a Administrator who is already in the list of 20 users maintained inside the 4758 BBRAM – to change his/her pass phrase on a single specified 4758. Effectively, what is submitted and stored is the SHA1 of the pass phrase. The host software calling this API must create the SHA1 of the pass phrase. The function will only complete if the user presents the SHA1 of his/her current pass phrase in addition to the SHA1 of the new pass phrase.

3.c.2. VPODecryptChallenge

This function decrypts two phrases – a challenge question and a challenge answer – into clear text. The encrypted phrases are stored in the master account database. Each account has it's own unique set of phrases. The phrases are single DES encrypted with the 4758 master key.

3.c.3. VPODecryptCreditCard

This function decrypts the stored end-user credit card number into clear text. The encrypted credit card number is stored in the master account database. The number is single DES encrypted with the 4758 master key.

The clear text version of the credit card must be made available when a credit card purchase transaction is submitted. At all other times, the credit card information is encrypted.

3.c.4. VPOEncryptChallenge

This function encrypts two clear text phrases – a challenge question and a challenge answer – specified by a given end user during his/her initial, Web-based account signup. The encrypted phrases are stored in the master account database. Each account has it's own unique pair of phrases. The phrases are single DES encrypted with the 4758 master key.

3.c.5. VPOEncryptCreditCard

This function encrypts the stored end-user credit card number. The encrypted credit card number is stored in the master account database. The number is single DES encrypted with the 4758 master key.

3.c.6. VPOInitializeAccount

This function is called when a Web-based signup application has been completed by a new a new user. At this time, an existing (and non-used) DSA key pair ID is associated with the account. The new account registers are confirmed to be zero and the submitted initial pass phrase SHA1 is encrypted with the 3DES master key. The entire account structure is MAC'ed with the active master key and returned to the calling program for storage in the account database.

3.c.7. VPOReadTransactionCounters

Each 4758 keeps a unique pair of counters in BBRAM. The counters are increment when either a re-credit or indicium operation completes successfully. This counter is emitted along with the 4758's unique serial number at the end of the transaction, and is used to populate the transaction table.

The counters are used to detect replay attacks *after the fact.* Identical counter values (with a given serial number) as well as gaps in the count (for a given serial number) are indications that some form of tampering has occurred.

3.c.8. VPORecomputeKeyMACWithCert

This function is necessary, because of the inherent delay between the time a DSA signing key is generated, and the time that a public X509 certificate is issued for this key by the USPS Certificate Authority (CA).

When the key is first created, a MAC is computed based on the key material, its numerical key ID, and the encrypted private component. However, the certificate number is also part of this MAC and it, by necessity, is 0 (unknown) when the key is created.

At a later time, when the CA is contacted and provided the public key material, a certificate number is issued. This function allows for the updating of the key structure with the certificate number, and a re-computation of the associated MAC.

3.c.9. VPOSetPassPhrase

This is a Crypto Officer 1 or 2 or Administrator function. It permits an end-user pass phrase SHA1 to be overwritten and set to a new value. It will be employed to deal with lost end-user pass phrases. The end user will have to provide sufficient information to convince the Crypto Officer that he is speaking to the valid person. The challenge question and answer (**VPODecryptChallenge**) is part of this validation procedure.

If the pass phrase is reset, the user is given the new phrase over the telephone. However, a flag is set on the account that forces the user to immediately change his/her pass phrase once again.

3.c.10. VPOShow4758Users

This function allows any Crypto Officer to display a list of up to 20 user names and their index positions stored in the BBRAM of a specific 4758. This function does *not* show the associated user pass phrases.

3.c.11. VPOVerifyAccountData

This function checks the MAC on account information for a specified account number. It returns success if the MAC is confirmed.

3.c.12. VPOVerifyIndicium

This function checks the digital signature on a supplied indicium stream. Currently, the USPS is performing indicium verifications so this function has no immediate utility. The associated DSA key ID private key structure is input (and decrypted) along with the indicium stream to accomplish the signature check.

3.c.13. VPOVerifyKeyMAC

This function checks the MAC on a DSA signing key for a specified Key ID. It returns success if the MAC is confirmed.

3.d. User

The *User Role* provides limited services, simply those necessary to create encrypted postage indicium or to purchase postage (a re-credit). This includes the following services:

3.d.1. VPOChangePassPhrase

This function allows a registered IBIP end user with an active account to change his/her pass phrase. The SHA1 of the pass phrase is submitted via a commercially available public key distribution method, RSA encryption algorithm. The SHA1 hashes of the current pass phrase and proposed new pass phrase are submitted. The host software calling this API must create the SHA1 hashes of the pass phrases. This function can be fielded by any 4758 in the server farm that is in PRODUCTION state.

The function uses the ACCOUNTSTATUS structure assembled from the database for the user's account. The ACCOUNTSTATUS structure contains a 3DES-encrypted representation of the user's current pass phrase SHA1. The API decrypts the current pass phrase SHA1, compares that with the decrypted pass phrase SHA1 in the incoming message. If the comparison is successful, the new pass phrase SHA1 is 3DES-encrypted with the current master key and output with a revised ACCOUNSTATUS structure. The output structure is used to update the SQL database for the user's account.

3.d.2. VPOCreateIndiciumRSA

This function allows a registered IBIP end user with an active account to request an indicium for a mail-piece. The input structure contains the encrypted SHA1 hash of the user's current pass phrase . Also included in the encrypted data stream is an indicium request structure that contains the amount of postage requested, the destination delivery point ZIP, the service class, and other information associated with the mail-piece. The RSA encryption algorithm is used for key distribution as a commercially available public key distribution method

3.d.3. VPOGetAccountStatus

This function validates an end-user request for information on his/her account. The internal authentication process compares the submitted end-user pass phrase SHA1 with that on file (encrypted in the ACCOUNTSTATUS structure). If the authentication passes and the bad pass phrase count has not been exceeded, the function returns SUCCESS (0) and that informs the

ISAPI gatekeeper application that account balances and other status information may be returned to the requestor

3.d.4. VPORecredit

This function is the second of two API calls that result in a meter reset. See **VPORecreditPreProcess** for the first phase.

This second phase operation occurs after the ACH or Credit Card transaction has successfully cleared. The function checks the end-user's pass phrase SHA1, confirms the existing account MAC, adjusts the registers to reflect the re-credit, increments the re-credit transaction counter in the specific 4758 handling the request, and finally re-computes the MAC on the new account information.

3.d.5. VPORecreditPreprocess

This function is the first of two API calls that result in a meter reset. See **VPORecredit** for the second and final phase.

The function checks the end-user's pass phrase SHA1, confirms the existing account MAC, and decrypts the re-credit request (e.g., amount of re-credit). The function also accepts a buffer containing the encrypted credit card information on file for that end user. The function outputs the re-credit amount requested and the credit card number in the clear. In point of fact, this first phase of the re-credit process is done to authenticate the request and decrypt the required information for the credit card processing (or ACH) step.

3.d.6. VPOReprintRequest

This function authenticates an end-user request to reprint a mail-piece. This functionality (i.e. reprint) has not yet been approved by the USPS, but has been included in this design for future use. It offers a method to track and potentially control reprints by end users, and may offer the USPS a safe means to solve a very serious end-user-acceptance issue.

Table 2 – Security Matrix

	PSI Crypto Officer 1	PSI Crypto Officer 2	Administrator	End User	Change Account Registers?	Generate DES/DESMAC First Trust Keys	Decrypt with DES First Trust Key	Verify MAC with First Trust DESMAC Key	Verify Manufacturing PIN	Zeroize DES/DESMAC First Trust Keys	Generate DSA Key	Encrypt DSA Key Private Component	Decrypt DSA Key Private Component	Sign with DSA Private Key	Verify Signature with DSA Public Key	Read Encrypted User Pass Phrase SHA1	Update Encrypted User Pass Phrase SHA1	3DES Decrypt End User Pass Phrase SHA1	3DES Encrypt End User Pass Phrase SHA1	Create 4758 User Profile Array	Modify 4758 User Profile Array	Read 4758 User Profile Array	Zeroize Profile Array
VPOChange4758UserPassPhrase	x	x	x																		x	x	
VPOChangePassPhrase				x												x	x	x	x				
VPOCreateIndiciumRSA				x	Y								x	x		x		x					
VPODecryptChallenge	x	x	x																			x	
VPODecryptCreditCard	x	x	x																			x	
VPOEmitRSAMessagingKey	x																					x	
VPOEmitRSARetainedKey	x	x																				x	
VPOEncryptChallenge	x	x	x																			x	
VPOEncryptCreditCard	x	x	x																			x	
VPOExtractMKShare	x	x																				x	
VPOFirstTrust	x						x	x	x	x											x		
VPOGenDESTrust						x														x			
VPOGenDSAKeyPair	x										x	x										x	
VPOGenMasterKey	x																					x	
VPOGenRSAMessagingKey	x																					x	
VPOGet4758Status	-	-	-	-																		x	
VPOGetAccountStatus				x												x		x					
VPOGetMKSHA1	x	x																				x	
VPOGetRSASHA1	x	x																				x	
VPOInitializeAccount	x	x	x																x			x	
VPOLoadMKShare	x	x																				x	
VPOLoadRSAMessagingKey	x																					x	
VPOReadTransactionCounter	x	x	x																			x	
VPORecomputeKeyMACWithCert	x	x	x																			x	
VPORecredit				x	Y											x		x					
VPORecreditPreprocess				x												x		x					
VPORegisterAdjustment	X	x	x		Y																	x	
VPORekeyAccount	X	x																				x	
VPOReMACAccount	X	x														x	x	x	x			x	
VPOReMACChallenge	X	x																				x	
VPOReMACCreditCard	X	x																				x	
VPOReMACKeyStructure	X	x										x	x									x	
VPOReprintRequest				x												x		x					
VPOResetAccountMAC	x																					x	
VPOSetActiveMasterKeyVersion	x																					x	
VPOSetDateTime	x																					x	
VPOSetPassPhrase	x	x	x														x		x			x	
VPOSetTransactionCounters	x																					x	
VPOShow4758Users	x	x	x																			x	
VPOUpdateProfiles	x																				x	x	

VPOVerifyAccountData	x	x	x																		x	
VPOVerifyIndicium	x	x	x												x						x	
VPOVerifyKeyMAC	x	x	x																		x	
VPOZeroize	-	-	-	-																	x	x

Table 3 – Security Matrix (continued)

	PSI Crypto Officer 1	PSI Crypto Officer 2	Administrator	End User	Create DES/3DES/DESMACMaster Key	Set Active Master Key Version	Decrypt Master Key Share	Encrypt Master Key Share	DES Encrypt with Master Key	DES Decrypt with Master Key	3DES Encrypt with Master Key	3DES Decrypt with Master Key	Compute DESMACwith Master Key	Verify DESMACwith Master Key	Zeroize Master Key Array	Create Indicium Counter	Increment Indicium Counter	Set Indicium Counter	Read Indicium Counter	Create Recredit Counter	Increment Recredit Counter	Set Recredit Counter	Read Recredit Counter	Zeroize Counters	Set Date and Time	Read Date and Time (Replay Detection)
VPOChange4758UserPassPhrase	x	x	x																							x
VPOChangePassPhrase				x							x	x	x	x												x
VPOCreateIndiciumRSA				x								x	x	x			x									x
VPODecryptChallenge	x	x	x							x																x
VPODecryptCreditCard	x	x	x							x																x
VPOEmitRSAMessagingKey	x										x															x
VPOEmitRSARetainedKey	x	x																								x
VPOEncryptChallenge	x	x	x						x				x													x
VPOEncryptCreditCard	x	x	x						x				x													x
VPOExtractMKShare	x	x						x		x																x
VPOFirstTrust	x																									
VPOGenDESTrust																x				x						
VPOGenDSAKeyPair	x										x															x
VPOGenMasterKey	x				x																					x
VPOGenRSAMessagingKey	x										x															x
VPOGet4758Status	x	x	x																							x
VPOGetAccountStatus	-	-	-	-									x	x												x
VPOGetMKSHA1	x	x																								x
VPOGetRSASHA1	x	x																								x
VPOInitializeAccount	x	x	x								x		x	x												x
VPOLoadMKShare	x	x					x			x																x
VPOLoadRSAMessagingKey	x											x														x
VPOReadTransactionCounter	x	x	x																x				x			x
VPORecomputeKeyMACWithCert	x	x	x										x	x												x
VPORecredit				x								x	x	x							x					x
VPORecreditPreprocess				x						x		x	x	x												x
VPORegisterAdjustment	x	x	x										x	x							x					x
VPORekeyAccount	x	x											x	x												x
VPOReMACAccount	X	x									x	x	x	x												x
VPOReMACChallenge	X	x									x	x	x	x												x
VPOReMACCreditCard	X	x									x	x	x	x												x
VPOReMACKeyStructure	X	x									x	x	x	x												x
VPOReprintRequest				x										x												x
VPOResetAccountMAC	X												x													x
VPOSetActiveMasterKeyVersion	X					x																				x
VPOSetDateTime	X																								x	x

VPOSetPassPhrase	X	x	x								x		x	x											x
VPOSetTransactionCounters	X																	x				x			x
VPOShow4758Users	X	x	x																						x
VPOUpdateProfiles	X																								x
VPOVerifyAccountData	X	x	x											x											x
VPOVerifyIndicium	X	x	x											x											x
VPOVerifyKeyMAC	X	x	x											x											x
VPOZeroize	-	-	-	-											x									x	x

Table 4 – Security Matrix (continued)

	PSI Crypto Officer 1	PSI Crypto Officer 2	Administrator	End User	Create RSA Retained Key Pair	RSA Decrypt with Retained Private Key	RSA Encrypt with Retained Public Key	Zeroize RSA Retained Key	Create RSA Messaging Key	3DES Encrypt RSA Msg Key Pvt Comp	3DES Decrypt RSA Msg Key Pvt Comp	RSA Decrypt with Msg Pvt Key	Zeroize RSA Messaging Key Array	Create Module State	Modify Module State	Read Module State	Read Encrypted Credit Card Number	Decrypt Credit Card Number	Encrypt Credit Card Number	Update Encrypted Credit Card Number	Read Encrypted Challenge Q & A	Decrypt Challenge Q & A	Encrypt Challenge Q & A	Update Encrypted Challenge Q & A	Ephemeral DESMACKey	Ephemeral DES Key
VPOChange4758UserPassPhrase	x	x	x									x				x									x	
VPOChangePassPhrase				x								x				x									x	
VPOCreateIndiciumRSA				x								x				x									x	
VPODecryptChallenge	x	x	x									x				x				x	x				x	
VPODecryptCreditCard	x	x	x									x				x	x	x							x	
VPOEmitRSAMessagingKey	x									x		x				x									x	
VPOEmitRSARetainedKey	x	x										x				x									x	
VPOEncryptChallenge	x	x	x									x				x							x	x	x	
VPOEncryptCreditCard	x	x	x									x				x			x	x					x	
VPOExtractMKShare	x	x				x										x									x	
VPOFirstTrust	x				x										x	x										
VPOGenDESTrust														x	x											
VPOGenDSAKeyPair	x											x				x									x	
VPOGenMasterKey	x					x										x									x	
VPOGenRSAMessagingKey	x					x			x							x									x	
VPOGet4758Status	-	-	-	-								x				x									x	
VPOGetAccountStatus				X								x				x									x	
VPOGetMKSHA1	x	x										x				x									x	
VPOGetRSASHA1	x	x										x				x									x	
VPOInitializeAccount	x	x	x									x				x									x	
VPOLoadMKShare	x	x				x										x									x	
VPOLoadRSAMessagingKey	x					x					x					x									x	
VPOReadTransactionCounter	x	x	x									x				x									x	
VPORecomputeKeyMACWithCert	x	x	x									x				x									x	
VPORecredit			x	x								x				x									x	
VPORecreditPreprocess				x								x				x	x	x							x	
VPORegisterAdjustment	x	x	x									x													x	
VPORekeyAccount	x	x										x				x									x	
VPOReMACAccount	x	x										x				x									x	
VPOReMACChallenge	x	x										x				x									x	
VPOReMACCreditCard	x	x										x				x									x	
VPOReMACKeyStructure	x	x										x				x									x	

VPOReprintRequest				x								x				x									x	
VPOResetAccountMAC	x											x				x									x	
VPOSetActiveMasterKeyVersion	x											x				x									x	
VPOSetDateTime	x											x				x									x	
VPOSetPassPhrase	x	x	x									x				x									x	
VPOSetTransactionCounters	x											x				x									x	
VPOShow4758Users	x	x	x									x				x									x	
VPOUpdateProfiles	x											x				x									x	
VPOVerifyAccountData	x	x	x									x				x									x	
VPOVerifyIndicium	x	x	x									x				x									x	
VPOVerifyKeyMAC	x	x	x									x				x									x	
VPOZeroize	-	-	-	-				x				x	x		x	x									x	

4. SECURITY RULES

This section documents the security rules enforced by the cryptographic module to implement the security requirements of this FIPS 140-1 Level 3 module:

1. The cryptographic module shall provide four distinct operator roles. These are the User Role, the PSI Cryptographic Officer 1 Role, the PSI Cryptographic Officer 2 Role, and the Administrator Role.

2. The cryptographic Module shall provide identity-based authentication.

   Each service authenticates the user based on user ID and password. Presently the module will support a minimum of 3 and a maximum of 20 Cryptographic Officers and Administrative users in BBRAM. These entries are used to authenticate services requested by non-USER roles (e.g. Cryptographic Officer 1, Cryptographic Officer 2, and Administrator.

3. The user id shall be a minimum of 6 and a maximum of 12 characters. The user pass phrase shall be a minimum of 6 and a maximum of 64 characters.

4. The operator shall not have access to any cryptographic services if they cannot be authenticated to a valid role.

5. After no more 5 consecutive unsuccessful authentication attempts have occurred for a given PSI Crypto Officer or Administrative user stored in the cryptographic module, the module shall lock out further attempts by that user to authentication. This lock-out is enforced even if the module power is momentarily removed, and can only be unblocked by the action of the Crypto Officer 1.

   This design rule is intended to make a search attack for a user pass phrase unfeasible.

6. The RSA public/private key algorithm will used as commercially available key distribution method .

7. Upon the application of power or when commanded by the operator, the cryptographic module shall perform the power-up self-tests in accordance with the IBM 4758 PCI Cryptographic processor specifications. The custom code will also perform an RSA encryption/decryption pair-wise consistency test, as well as a TDES encryption decryption known answer consistency test. If either consistency test fails, the device will enter a hard-error state and be *unable* to process any and all service requests. Should this situation occur, the unit must be returned to PSI manufacturing or IBM to be serviced.

8. At any time the module is in an idle state, the operator shall be capable of commanding the module to perform the power-up self-test.

9. Prior to each use, the internal Random Number Generator shall be tested using the Conditional test specified in FIPS 140-1 §4.11.2 paragraph 5.

10. Certain public keys are protected against unauthorized modification through the use of key structures that include a key ID or tag and a keyed MAC. For instance, the DSA signature keys are stored in a C-style structure containing the public key component, private key component, a unique key ID, and a keyed MAC on all of the aforementioned data. When a signature key is requested for a signing

operation, the custom 4758 code checks the MAC on the key structure input as well as the key ID (or key tag) to ensure that it is the one requested. Similarly, the RSA messaging key is both encrypted and MAC'ed when it is cloned from one 4758 to another, and also includes a key ID in the associated C-structure. The key ID is checked against the one requested when a 4758 accepts a messaging key for loading. The public component of the messaging key is also embedded in the end-user client and the WIN32 administrative module. This key material could conceivably be corrupted or substituted, but then all authentication attempts into the 4758 would fail.

11. Services using key material can confirm that the key requested is indeed the key being supplied. All public/private key material, as well as master key material, is presented to the 4758 in conjunction with a key ID. In the case of public/private keys, this material (including the key ID) is also MAC'ed using the master key of the 4758 farm.

SECURITY RELEVANT DATA ITEMS (SRDI)

The PCC uses the following security relevant data items:

1. **Manufacturing Keys:** This DES key and DESMAC key are used solely for the purpose of authenticating the first use of the device (**VPOFirstTrust**). This key is generated during manufacturing and physically provided to the Cryptographic Officer. This key is destroyed inside the PCC when **VPOFirstTrust** completes (i.e., this is a "run-once" function).

2. **Account DSA Key Pairs:** For each end user account, a 1024 bit DSA key pair is generated. This key material, which is used to digitally sign all indicium streams created for that account, must be stored outside of the PCC, given that there will be hundreds of thousands of key pair. The private (x) component of each key pair is 3DES encrypted with the PCC's master key prior to storage outside the PCC.

3. **End User Pass Phrase SHA1 Hash:** The end user employs a pass phrase of up to 64 bytes to protect his/her account. The pass phrase is immediately converted to an SHA1 hash, and it is this hash that effectively operates as the password for a given account. The SHA1 hash is stored in the main account tables and, since it is for all intents and purposes the password, it is 3DES encrypted with the PCC master key. The pass phrase SHA1 appears in decrypted form only inside the PCC.

4. **Administrative User Profiles:** Each PCC has up to 20 vendor employee names and pass phrase SHA1 stored in BBRAM. This information is maintained and updated by the Crypto Officer 1. Only Crypto Officer 1, Crypto Officer 2 and Administrator personnel employed by the vendor can exist in this secure table.

5. **PCC Master Keys:** All PCC's have an array of up to 10 master keys that are versioned. Each master key consists of 32 bytes of key material, as well as three 32 byte key shares that were used to create the final key. The first 24 bytes of the key are used for 3DES operations. The first 8 bytes of the key are used for DES operations. The last 8 bytes of the key are used for DESMAC operations. Master key material is cloned throughout all participating PCC's using a secure PKI technique.

6. **Indicium and Re-credit Counters:** Each PCC maintains a count of the number of indicium operations performed by that unit, and the number of re-credit operations performed by that unit. This information is used to mark each such transaction with a unique, ascending serialization. Gaps or duplications in this sequence are used to flag potential security breaches.

7. **Date and Time:** Each PCC is synchronized within a window of ±10 seconds. Time stamps based on this internal (GMT) clock are used in replay detection and prevention.

8. **Retained RSA Key:** Each PCC has a unique 1024 bit RSA key pair used as a commercially available public key distribution method which provides encrypted communications between the Crypto Officer 1 and the device during initialization steps. The private portion of this key never leaves the secure BBRAM of the PCC.

9. **RSA Messaging Keys:** Each PCC can store up to 4 RSA Messaging Key versions. The keys are securely cloned from one PCC to another, so that the entire "farm" of PCC's has the same array of

private RSA key material. The corresponding public portion of this RSA key is used as part of a commercially available public key distribution method.

10. **Module State:** This integer value stored in the PCC BBRAM, and is used to determine what services are available at a given instance in time. PCC's spend most of their life in a PRODUCTION state. Non-production states occur during the initialization and key loading process prior to production status.

11. **Credit Card Number:** If the end user employs a credit card for postage purchases, this card is stored in a DES encrypted fashion in the main account database. The DES encryption and decryption operations are accomplished inside the PCC using the master key.

12. **Challenge Question and Answer:** To deal with "lost pass phrase" scenarios, the end-user provides a challenge question and matching answer during the account signup process. This information is DES encrypted inside the 4758, and then stored in the master account table.

13. **Ephemeral Keys**. The encrypted data contains both a DES and DESMAC key (generated by NIST Certified MS Crypto API modules at the client). This key material is transferred to the PCC using a commercially available public key distribution method based on the RSA algorithm. The DESMAC key is then used to check the message MAC.

5. SRDI MODES OF ACCESS

Tables 3, 4 and 5 also define the relationship between access to SRDIs and the different module services. The available services are shown in the left-most column. The next four columns show the 4 defined roles, and a check box in any column indicates that this role has access to the associated service.

The next column is provided for the US Postal Service, showing which services change the account registers for a given end-user. Note that register values (i.e., balances) are stored in the clear and not considered SRDI's. However, the register values are MAC'ed with other key account information using the PCC master key to detect potential tampering. Also, register values are changed *only* within the confines of the PCC.

The remaining columns depict various access modes for the SRDI. The columns are grouped/shaded by SRDI. For example, the access modes for the master key material include:

- Creation of Key Material
- Setting of Active Key Version
- Decrypting Master Key Share (used in cloning keys amongst PCC's)
- Encrypting Master Key Share (" ")
- DES Encryption
- DES Decryption
- 3DES Encryption
- 3DES Decryption
- Computation of DESMAC
- Verification of existing DESMAC
- Zeroization

A similar breakdown of access modes appears for each SRDI. The PCC Module and API documents present the corresponding Finite State Model Diagrams for each service that reinforce the information in this matrix.

INDEX