FIPS 140-2
SECURITY POLICY

DATAKEY MODEL 330J SMART CARD WITH JCCOS APPLET

407 West Travelers Trail
Burnsville, MN 55337-2554
(612) 890-6850

Table of Contents

1. Introduction

1.1. Scope

This document describes the non-proprietary cryptographic module security policy for the Datakey Model 330J smart card with the Java Card Crypto Operating System (JCCOS) applet. It contains specification of the security rules, under which the cryptographic module operates, including the security rules derived from the requirements of the FIPS 140-2 standard and the additional security rules imposed by the Datakey JCCOS applet loaded on the 330J smart card.

1.2. Overview

The 330J cryptographic module is a smart card compliant with parts 1 through 4 of the ISO 7816 standard, which define the physical characteristics, contact arrangement /location, electrical characteristics, and interface protocol.

The chip platform operating system is based on JavaCard technology and Open Platform. Together they manage all the low level resources, cryptographic algorithms implementation, object access control and applications life cycle.

The JCCOS applet is a state of the art application designed and developed using Datakey's extensive experience in secure operating system development. When downloaded to the chip the JCCOS applet provides the following high-level security services:

- Secure storage and retrieval of objects and digital credentials.
- Authentication of the cardholder and the security officer.
- Cryptographic services such as SHA-1, DES, 3DES, RSA Sign/Decrypt and RSA on board key generation.

Furthermore, as a JavaCard applet, JCCOS security model is totally independent from the Open Platform security model. It implements its own finite state machine, does not use the Global PIN provided by the Open Platform card manager, it creates and manages its own authentication PINs and it contains its own Power On Self Tests.

This design leads to increased interoperability with all Java Cards that are compliant with JavaCard API version 2.1.1 and because of the embedded self-tests, FIPS 140-2 level 2 or higher certification is achievable by any Java Card platform with the JCCOS applet loaded and installed.

1.3. Model 330J smart card architecture

The architecture of the Model 330J smart card is different from a common JavaCard. Datakey designed the Model 330J to provide built-in cryptographic and data container management functions while giving enterprises the ability to add new applications in the future. In order to do this, Datakey created a high security, high performance cryptographic application. This application is embedded in ROM rather than in EEPROM, which provides many advantages from a security, use of memory, deployment, and card management

perspective. The unique architecture of the Model 330J smart card offers several distinct advantages.

- Efficient use of memory: Only the data objects created and used by the built-in cryptographic application are stored in EEPROM; no memory space is used for overhead for cryptographic applets.

- User manageability of the contents of the smart card: Users can easily load and delete data objects on their smart card, without requiring a return to an issuing station or compromising the Open Platform security model.

- Reduced deployment time: The Java application resides in ROM, not EEPROM. This saves time during the personalization process because the application already resides on the smart card.

- Compatibility with current Datakey CIP software: Leverages proven interoperability with a broad range of information security and e-business applications.

1.4. Related Standards and Documents

CC	ISO 15408 – Information Technology – Security Techniques – Evaluation Criteria for IT Security (Hereafter referred to as Common Criteria or CC)
ISO 7816-1	ISO/IEC 7816-1 (1987): "Identification cards – Integrated circuit(s) cards with contacts, Part 1: Physical characteristics".
ISO 7816-2	ISO/IEC 7816-2 (1988): "Identification cards – Integrated circuit(s) cards with contacts, Part 2: Dimensions and locations of the contacts".
ISO 7816-3	ISO/IEC 7816-3 (1989): "Identification cards – Integrated circuit(s) cards with contacts, Part 3: Electronic signals and transmission protocols".
ISO 7816-4	ISO/IEC 7816-4: "Identification cards – Integrated circuit(s) cards with contacts, Part 4:
PKCS 1	PKCS #1: RSA Encryption Standard, Version 1.5, November 1993
PKCS 12	PKCS#12 Personal Information Exchange Syntax, Version 1.0, June 24, 1999
PKCS 15	PKCS#15 Cryptographic Token Information Format Standard, Draft, February, 1999
JavaCard API	A specification for JavaCard 2.1.1 Application Programming Interface.
Global Platform	Open Platform Card Specification Version 2.0.1

2. Glossary

Activation	A process that gives a card the required operational capability for the cardholder.
Application	Intended final use for the smart card. This may include (but is not limited to) such activities as payment, telephony, identification, secure information storage, or loyalty.
Authentication Data	Comprise the officer identifier, certificate, role and privileges.
Bond-out chips	Raw ICs, which have been mounted on a small board. Wire bonds are connected from the IC's input/output pads to the carrier, which has contacts on its reverse side. Bond-out chips are sometimes referred to as a module.
Card disablement	The IC function related to terminating all operations other than possibly some limited audit functions. Card disablement is permanent.
Card embedder	A manufacturer who assembles a card and integrated circuit.
Card holder	A person to whom a card has been legitimately issued (a user).
Card issuer	An institution, which issues cards to cardholders.
Card Operating System (COS)	Operating system developer specific code, written in the microprocessor's native or machine code.
Card reader	A machine capable of reading and/or writing to a card, such as magnetic stripe card or smart card.
Carrier	The holder in which an operational integrated circuit is placed. This is typically the thin, credit card sized piece of plastic that is known as a smart card.
Die	The semiconductor IC without any packaging or connections.
EEPROM	Electrically Erasable Programmable Read Only Memory. A non-volatile memory technology where data can be electrically erased and rewritten.
Failure analysis	The compilation of techniques used by semiconductor development and testing labs to identify the operating problems in newly designed or modified integrated circuits. Such techniques include not only observation (to determine what is not functioning properly) but also modification of IC internal structure (to determine fixes).
First use indication	The IC function related to setting a specific audit bit indicating that the smart card is now in the issued, operational state and can be used for its intended function.
I&A	Identification and Authentication
IC	Integrated Circuit. Electronic component(s) contained on a single chip and designed to perform processing and/or memory functions.
ICC	Integrated Circuit Card. A card into which has been inserted one

	or more ICs.
ID	Identity (also, a token asserting an identity)
Initialization	The process of writing specific information into Non-Volatile Memory during the early card life cycle.
IP	Internet Protocol
ISO	International Standards Organization
Life cycle identifiers	The specific identification of chip fabricator identifier, operating software identifier, chip module identifier, chip embedder identifier, initialiser identifier, initialization equipment identifier, personaliser identifier and personalization equipment identifier.
Modules	A functional assembly for use with other assemblies. These may be separate parts of an IC (CPU, Coprocessor, ROM, RAM, etc.), bond-out chips, or software components.
NIST	National Institute of Standards and Technologies
Non-volatile memory	A semiconductor memory that retains its content when power is removed. (i.e. ROM, EEPROM, FLASH).
Personalization	The process of writing specific information into the non-volatile memory in preparing the IC for issuance to users.
PIN	Personal Identification Number
Platform	A term representing an operational smart card system.
Post-issuance	The time period during which the smart card is in the hands of the cardholder. In some smart cards, additional functionality can be loaded into the smart card post-issuance.
PP	Protection Profile
RAM	Random Access Memory. A volatile, randomly accessible memory (used in the IC) that requires power to maintain data.
ROM	Read Only Memory. A non-volatile memory (used in the IC) that requires no power to maintain. ROM data is often contained in one of the numerous masks used during manufacture.
RSA	Rivest, Shamir, Adleman (encryption algorithm)
Security Officer	The administrator of the CM system. The security officer has in addition to the administrative privileges also all the privileges a registration officer can have
SHA	Secure Hash Algorithm
Smart card	A shaped piece of plastic or other carrier with a small computer chip embedded into it.

Terminal	The device used in conjunction with the CAD at the point of transaction.
Transport keys	The cryptographic keys loaded into the IC for security during transport of ICs, modules and assembled products prior to issuance.

3. Security Levels

The Datakey Model 330J with the JCCOS applet meets all requirements for FIPS 140-2 level 2. Refer to the following table for individual security requirements:

Security Requirements	Certification Level
Cryptographic Module Specification	2
Cryptographic Module Ports and Interfaces	2
Roles, Services, and Authentication	2
Finite State Model	2
Physical Security	3
Operational Environment	N/A
Cryptographic Key Management	2
Electromagnetic Interference/Electromagnetic Compatibility (EMI/EMC)	3
Self Tests	2
Design Assurance	3
Mitigation of Other Attacks	N/A

4. Cryptographic Module Specification

4.1. Cryptographic Boundary

The cryptographic boundary for the Datakey Model 330J Smart Card is the physical boundary of the card itself. Although the boundary could be defined as the physical micro-module containing the processor chip, the embedding of the micro-module within the plastic card provides no further cryptographic function. The card provides utility to the micro-module as a standalone cryptographic module component and enhances the tamper evidence of the cryptographic module.

The cryptographic module submitted for evaluation is a single-chip module combining the Java Card platform and the JCCOS applet together to form the cryptographic module that meets the FIPS 140-2 level 2 requirements.

This platform contains the following hardware components:

- A crypto co-processor optimized for public key cryptographic calculations
- A Triple DES (Data Encryption Standard) Co-processor.
- 96K User ROM.
- 2300 bytes RAM.
- High-reliability 32K EEPROM for both data storage and program execution.

4.2. Hardware Security Features

The following features are provided by the Philips P8WE5033 microcontroller.

- Power-up / power-down reset,
- Low / high supply voltage sensors,
- Low / high clock frequency sensors,
- Low / high temperature sensors,
- On-chip self test with signature technique,
- EEPROM erase / write timing independent from external clock,
- EEPROM erase /write operation controlled by hardware sequencer,
- Electronic fuses for safeguarded mode control,

These hardware security features were not tested as part of the FIPS 140-2 validation.

4.3. Physical structure

The die contains bonding pads for GND (ground), VCC (supply voltage), CLK (clock signal), RST (external reset signal), I/O 1 (used for half-duplex input / output communication), VPP (programming voltage –not used), I/O 2 (not used) and I/O 3 (not used).

4.4. Fabrication Process

The die is attached with adhesive to the back of a film-based, ISO 7816-compliant contact substrate, such that the die bonding pads and the contact substrate bonding pads are accessible. Wire bonds are made between the die bonding pads and the contact substrate bonding pads for GND, VCC, CLK, RST and I/O. A measured amount of encapsulant is flowed over the contact substrate, such that the die and wire bonds are contained within the encapsulant.

The micro-module is glued into a matching milled cavity in a plastic card, which complies with the dimensional and other physical requirements of the ISO 7816-1 standard. Typical customization of the card for the application and / or user enterprise may be accomplished prior to and subsequent to micro-module insertion.

Silk screen (or other mass printing) and magnetic stripe application are performed prior to micro-module insertion. Personalization printing (such as names, account numbers, and photographs) and writing of the magnetic stripe data and personal data on the chip are performed after micro-module insertion.

A major component of the cryptographic security module identified as the Datakey Model 330J Smart Card is the Datakey Java Card Crypto Operating System applet (JCCOS). Functional detail of the applet is disclosed in subsequent sections.

5. Cryptographic Module Ports and Interfaces

5.1. Physical Interface

Five electrical connections are made between the die and the contact substrate of the smart card:

- **VSS**, Ground (reference voltage).
- **VDD**, Power supply input.
- **RST**, External reset signal from the interface device (card read / write device)
- **CLK**, External clock (3.517 MHz).
- **I/O**, Input or output for serial data to / from the processor.

The above five electronic signals are in full compliance with ISO 7816-3. The VPP (programming voltage) contact is not used because the EEPROM of the chip contains its own internal programming voltage generation. Also the I/O 2 and I/O 3 ports, which are present on the chip, are not connected to the contact substrate.

Reference: ISO 7816 Part 3 Identification Cards – Integrated Circuit(s) Cards with Contacts – Electronic Signals and Transmission Protocols

5.2. Logical Interface

The Datakey JCCOS applet controls the logical interface thru a well-defined set of Application Protocol Data Unit (APDU) commands. It manages the secure object storage system, interface protocol and parameters, interprets and executes external commands, and interfaces to the Java Card platform via the Java Card API. The JCCOS applet also provides the capability to configure card functionality (cryptographic algorithms, key lengths and access control security policy) for the user establishment.

The APDU communication protocol defines the following four logical interfaces as per the FIPS 140-2 standard as follows:

a) Data Input interface: The input data field of the command APDU comprises the data input interface of the module. All input parameters can only be passed through this interface.
b) Data Output interface: The output data field of the response APDU comprises the data output interface of the module. All output data can only be passed through this interface.
c) Control Input interface: The command APDU header consisting of the CLA, INS, P1, P2 and LC bytes comprises the control input interface. All control parameters for module execution can only be passed through this interface
d) Status Output interface: The status words SW1 and SW2 of the response APDU comprise the status output interface. All error codes and output indicators are output through this interface.

References: JCCOS v2.0 Interface Control Document, June 13, 2002.

6. Roles, Services, and Authentication

6.1. Roles

The Datakey Model 330J smart card provides two roles, the Security Officer (SO) and the User role. The Security Officer is tantamount to the Crypto-officer in FIPS 140-2 terminology. Each role is assigned various services. Please see the following table for a list of all services available to a particular role in FIPS mode of operation. When an operator is in a particular role the module state is set to indicate this. For example: When the operator is authenticated as an SO the corresponding module state is SO_AUTH

State	Commands Available
Error	GetAppletStatus, RemoveAllObjects, UserPINUnblock, UpdatePIN
Idle	DeleteObject, GenerateRandomNumber, GetAppletStatus, GetObjectValue, RemoveAllObjects, SelectObject, SHA1, Verify,
User Authenticated	CreateObject, Crypt, DeleteObject, EndSession, GenerateDESKey, GenerateRandomNumber, GetAppletStatus, GetObjectValue, RemoveAllObjects, RSADecrypt[1],RSAGenerateKey, RSASign,SelectObject, SHA1, UpdatePIN, SetObjectValue
SO Authenticated	ChangeConfiguration, CreateObject, CreateUnblockPINObject, DeleteObject, EndSession, GenerateRandomNumber, GetAppletStatus, GetObjectValue, RemoveAllObjects, SelectObject, SHA1, UpdatePIN, SetObjectValue, UserPINUnblock

6.1.1. JCCOS Security Officer Role

The JCCOS Security Officer (SO) role is responsible for configuring the applet by changing the configuration file settings (specifying which algorithms are allowed by the applet, which keys may be generated, and who may generate keys), setting up the User's password and unblock User PINs. It is the SO's responsibility to ensure that the configuration file settings are set so that the module is in FIPS mode (see FIPS Mode section).

The JCCOS SO entity is totally independent from the Card Manager or the security domain officer entity and therefore does not require knowledge of assigned security domain key set.

6.1.2. JCCOS User Role

The User role is essentially the end user and thus has access to all of the cryptographic functions of the module, but does not have the access (that the Security Officer has) to the card configuration functions.

Additionally, several unauthenticated services are available. These services are listed in the table above. The services in the Idle (unauthenticated) state provide general card status and

[1] RSA Decryption is a non-Approved algorithm. It should be used in FIPS mode only in as part of a key transport scheme. RSA Decryption may not be used for general data decryption.

other services necessary for the secure operation of the module and other non-security critical operations. Because these services are unauthenticated they cannot compromise the overall security of the module.

The Datakey Model 330J smart card implements a method of restricting access to data and objects based upon the role authenticated. Each data or key object is stored in a container, and each container has associated security permissions (nibbles) that are set during its creation. The security permissions determine whether the Security Officer (SO), User, anyone (Always), or no one (Never) has access to read, write, update, execute (use a key), or delete the container.

The security nibble definitions including the definitions of each type of access are described below:

Security Nibble Structure					
Byte 1		**Byte 2**		**Byte 3**	
SN1	**SN2**	**SN3**	**SN4**	**SN5**	**SN6**
Read	Update	Write	Delete	RFU	RFU

Security Nibble Definition	
Code	**Value**
Never	0x0
SO	0x4
User	0x2
Anonymous	0x1

(Permissions may be combined in the UNIX style. For example, 0x7 gives object file access to the SO, the User, and to anyone in the Idle state.)

Security Nibble Type Definition	
Nibble Type	**Definition**
Read	Object file may be read by entity with GetObjectValue command
Update	Object file data may be written by the entity up to the high water mark with SetObjectValue command
Write	Object file data may be written by the entity at the high water mark with SetObjectValue command
Delete	Object file may be deleted by the entity with DeleteObject file command
RFU	Reserved for future use

6.2. Services

6.2.1. JCCOS Applet Services

Please refer to the JCCOSICD document for detailed information about each function including the required inputs and expected outputs

PIN Management services

- **ChangeChvPin**: Updates the PIN of the given type if the given current PIN is valid for the currently authenticated entity in accordance to the GSA interoperability specification.
- **EndSession**: Ends the current authenticated session, returning the applet to the idle state.
- **UpdatePIN**: Updates the PIN of the given type if the given current PIN is valid for the currently authenticated entity.
- **Verify**: Hashes the given data (operator pin) and compares the result with the value stored in the applet's pin object container. If comparison is successful the module state is set to indicate successful authentication of the operator
- **CreateUnblockPINObject**: Creates an empty User PIN unblocking container.
- **UserPINUnblock**: Unblock any of User PINs if the given current PIN is valid. The given current PIN is valid if its hash matches the hash stored in the unblocking PIN container.

PKI services

- **Crypt**: Performs a DES/3DES symmetric key encryption/decryption on the given data.
- **GenerateDESKey**: Generates a single DES key (eight bytes) or a two key triple DES key (sixteen bytes) from the on-card FIPS 186-2 pseudo random number generator
- **GenerateRandomNumber**: Creates a random number of the given size, using the FIPS 186-2 (Appendix 3.1) compliant pseudo random number generator.
- **RSADecrypt**: Decrypts the given ciphertext with the private RSA exchange key (or exchange/signature key) in the given object container.
- **RSAGenerateKey**: Generates an RSA key pair into the given private key and public key object containers.
- **RSASign**: Performs an RSA PKCS#1 (version 1.5) signature on the given data with the private RSA signature key (or exchange/signature key) in the given object container.
- **SHA1**: Initiates, continues, or completes a SHA-1 hash of the given data.

Secure storage services

- **CreateObject**: Creates an empty file container of the given type.
- **DeleteObject**: Deletes references to a given file container.

- **GetObjectValue**: Returns the requested amount of data (at the given offset) from the active file container.
- **RemoveAllObjects**: Deletes references to all object containers created by the applet and zeroes all allocated buffer space.
- **SelectObject**: Makes the given object file container the active container, to be used by subsequent commands.
- **SetObjectValue**: Writes the given data (at the given offset) to the active file container.

General Services

- **ChangeAppletConfiguration**: Updates the current configuration data in the configuration file.
- **GetAppletStatus**: Returns the current internal status of the applet. This command returns status information about the applet such as the applet state, error status, applet version, remaining EEPROM space, and the current configuration settings of the applet. If these settings match the FIPS mode settings specified in the applet is considered to be in FIPS mode.
- **Format**: During initialization of the applet, this command is required to initialize containers storage system of a card and sets the initial SOPIN . This is a pre-issuance command and is not available once the card has been issued to the end user (Security Officer and User roles).

6.2.2. Open Platform Services

The Card manager applet is responsible for performing all card content management including the loading, installation and deletion of applets on the 330J smart card and changing the Card Manager keyset. All Open Platform services except the Get Data, Get Status and the Initialize Update command require mutual authentication and initiation of a secure session using the Card Manager keyset as defined in the Open Platform specification (version 2.01).

The JCCOS applet is loaded and installed on the 300J smart card during factory initialization (pre-issuance of card). The CardManager keyset is not released to the end user, thus disallowing a user to establish a secure session with the Card Manager and performing any of the Card Manager services.

Moreover the card is delivered to the user in the OP_Secured state. The OP_Secured state is an irreversible state in which the 330J smart card contains the instantiated JCCOS applet that can be selected using the Select APDU. Thereafter the default SO PIN can be used to authenticate the SO to the JCCOS applet and initialize User PIN. A secure channel cannot be established with the Card Manager in the OP_Secured state even if the operator has knowledge of the Card Manager keys. Thus no Card Manager services other than Get Data, Get Status and the Initialize Update can be invoked.

6.3. Authentication

6.3.1. User Authentication

The cardholder must execute the Verify command with the correct PIN to transition the applet state to User authenticated state. In this state, the User can access services provided by the JCCOS applet that require User authentication.

Additionally to discourage an attacker from guessing the SO PIN, the JCCOS applet maintains a count of the number of consecutive Verify (User) and UpdatePIN (User) attempts remaining (the limit is established by configuration file) due to an incorrect PIN. This count is maintained in nonvolatile memory. When this count reaches zero, the User PIN will be blocked, the uninitialized PIN error response will be returned to the host, and the card will enter the Error state. If allowed by the configuration file, the SO may update the User PIN in order to re-enable the User PIN. Alternatively, if Unblocking PINs exist the User can issue the UserPINUnblock command by providing the correct unblocking PIN to update the User PIN file with a new PIN. Every successful Verify (User) and UpdatePIN (User) will reset the failed attempts count to zero.

6.3.2. Security Officer Authentication

The Security Officer must execute the Verify command with the correct PIN to transition the applet state to SO authenticated state. In this state the SO can access services provided by the JCCOS applet that require SO authentication.

Additionally to discourage an attacker from guessing the SO PIN, the JCCOS applet maintains a count of the number of consecutive Verify (SO) and UpdatePIN (SO) attempts

remaining (the limit is established by configuration file) due to an incorrect PIN. This count is maintained in nonvolatile memory. When this count zero, the SO PIN is disabled, the command response is Uninitialized PIN and the card enters the Error state. The SO PIN can be reset to the default SO PIN by issuing the RemoveAllObjects APDU that restores the backup SO PIN to the SO PIN file. Every successful Verify (SO) and UpdatePIN (SO) will reset the failed attempts count to zero.

The following table summarizes the type of authentication and strength of mechanism for each role.

Role	Authentication	Strength of Mechanism
User	PIN	8-20 bytes
SO	PIN	8-20 bytes

The PINs can be considered Security Relevant Data Items (SRDI). However the module does not store any actual PINs in EEPROM. Only the SHA-1 hash of PIN value is stored in the User or SO PIN file, which cannot be read or written except by using the UpdatePIN or UserPINUnblock command by an authenticated user that writes the hash of the new PIN to the PIN file.

The following table summarizes the SRDIs and CSPs available to each role.

Role	SRDI	Type of access
Idle	Configuration file settings	Read
User	Internally generated secret and private keys Internally generated public keys User loaded public keys Configuration file settings	Usage Usage/Read/Write Usage/Read/Write Read
SO	Configuration file settings	Read/Update

6.4. Configuration object file

The JCCOS applet has a configuration object file (with file ID FF00) that has two 20-byte sections - the first is the configuration data and the second is a bit mask that will not be applicable for the JCCOS applet, and is set to hex FF. The configuration settings determine the overall security rules employed by the module. Only the SO is allowed to modify the configuration file settings by a call to ChangeAppletConfiguration. The first 20 bytes of data in this command contain the new configuration data, which is used to update the data in the configuration object file.

The configuration data of the object file has the following format

Data	Length in bytes
RSA Exchange enable/size	1
RSA Signature enable/size	1
RFU	1
RFU	2
Public key formatting	1
Crypto command enable	2
Symmetric key enable	1
SO Authentication	1
User Authentication	1
RFU	2
Idle Allow	2
Export Control	1
Write Protection Control	1
RFU	3
Configuration Mask	20

The meaning of each type of byte settings is explained in the tables below:
The top nibbles of the RSA enable/size bytes indicates the maximum allowable modulus size, and have the same definition as the public key length index of section 3.2. The bottom nibble of the RSA enable/size bytes is defined below:
Note: Bits 0 and 1 of the RSA enable/size byte are not used and can be set to anything

Bottom Nibble				
Bit 3	**Bit 2**	**Bit 1**	**Bit 0**	**Meaning**
0	x	x	x	No user readable private keys
1	x	x	x	User readable private keys allowed
x	0	x	x	User cannot generate keys
x	1	x	x	User may generate keys

Public Key Formatting	
Bit 7	RFU

Bit 3	RFU
Bit 1	0 - RSA Raw Disabled 1 - RSA Raw Enabled
Bit 0	0 - RSA PKCS1 Disabled 1 - RSA PKCS1 Enabled

The two Crypto Command Enable bytes control the availability of the crypto-related commands. A zero indicates the command is not available, while a one indicates that the command is available.

Crypto Command Enable - First Byte	
Bit 7	Crypt
Bits 6:3	RFU
Bit 2	GenerateDESKey
Bit 1	RSADecrypt
Bit 0	RFU

Crypto Command Enable - Second Byte	
Bit 7	RSAGenerateKey
Bit 6	RSASign
Bit 5	RFU
Bit 4	POST
Bits 3:0	RFU

The upper nibble in the Symmetric Key Enable byte controls the use of single and triple DES (Crypt and GenerateDESKey commands) and the use of the ECB and CBC DES modes (Crypt command). The bottom nibble has the same definition as the bottom nibble of the public key enable/size bytes.

Symmetric Key Enable	
Bit 7	0 - Two Key Triple DES Disabled 1 - Two Key Triple DES Enabled
Bit 6	0 - Single DES Disabled 1 - Single DES Enabled
Bit 5	0 - ECB mode Disabled 1 - ECB mode Enabled
Bit 4	0 - CBC mode Disabled 1 - CBC mode Enabled

SO Authentication	
Bits 7:6	RFU
Bit 5	Update PIN (without current PIN)
Bit 4	0 - RemoveAllObjects command writes hash of default PIN phrase to SO PIN

	object file 1 - SO PIN data is not changed
Bits 3-0	Maximum consecutive Verify(SO) failures

User Authentication	
Bit 7	RFU
Bit 6	PIN object file may be written in Idle state
Bit 5	Update PIN (without current PIN)
Bit 4	0 - SO may not update User PIN 1 - SO may update User PIN
Bits 3:0	Maximum consecutive Verify(User) failures

The two-byte EXF field is not applicable for this applet.

The two Idle Allow bytes determine if certain commands are allowed in the Idle state (a zero indicates the command is not allowed in Idle, while a one indicates the command is allowed in Idle):

Idle Allow - First Byte	
Bit 7	CreateObject
Bit 6	Crypt
Bits 5:2	RFU
Bit 1	GenerateDESKey
Bit 0	RSADecrypt

Idle Allow - Second Byte	
Bit 7	RFU
Bit 6	RSAGenerateKey
Bit 5	RSASign
Bit 4	RFU
Bits 3:0	RFU

Datakey controls the definition of the Export Control byte. JCCOS does not use this byte. The write protect control byte controls the states allowed to be able to create objects and set object values.

Write Protection Control	
Bit 2	SO Authenticated State
Bit 1	User1 Authenticated State

The RFU bits are not used in the JCCOS applet and have been ‘Reserved for future use’. These bits can be set to anything.

6.5. FIPS mode:

The module is delivered with the Card Manager state set to OP_Secured to disable loading of applets into the card. The CardManager keys are never made available to the end-user in the post-issuance phase. This means that all Card manager security services such applet loading, installation, and deletion can take place only during the factory initialization process in the pre-issuance phase.

The JCCOS applet has a FIPS mode configuration object file with the following values:

Field	Value	Meaning
RSA Exchange enable/size	0x77	RSA Exchange enabled to 2048 bits. User may generate keys
RSA Signature enable/size	0x77	RSA Signature enabled to 2048 bits. User may generate keys
RFU	0x00	RFU
RFU	0x00, 0x00	RFU
Public key formatting	0x01	Only PKCS #1 formatting enabled for RSA
Crypto command enable	0x87, 0xF0	All cryptographic commands enabled
Symmetric key enable	0xF7	DES, triple DES enabled
SO Authentication	0x2A	RemoveAllObjects replaces SO PIN with default value, SO may UpdatePIN without current PIN. SO PIN count is 10
User Authentication	0x3A	SO may update User PIN, User may UpdatePIN without current PIN. User PIN count is 10
RFU	0xFF, 0xFF	RFU
Idle Allow	0x00, 0x00	Commands not allowed in Idle state
Export Control	0x00	RFU
Write Protection Control	0x06	User and SO can create/modify objects
RFU	0x00, 0x00, 0x00	RFU

It is the SO responsibility to ensure the bit settings are set as per the table above when module is in FIPS Approved mode. An operator can use the GetAppletStatus command to determine whether the module is in the Approved FIPS 140-2 mode by matching the configuration file settings to the values mentioned in the table above.

In FIPS mode an operator should not use the RSA Decrypt APDU to decrypt data. It is permissible to use RSA Decryption for key transport scheme, but not for data decryption. The module does not provide RSA encryption as public key operations can be more efficiently on a host computer.

7. Finite State Model

In this cryptographic module there are two separate state machines that work in concert to manage both the Java Card platform states itself and the JCCOS applet internal states. The cryptographic module has an overall lifecycle state that is dictated by the Open Platform and ISO 7816 Specifications. The OP Specifications describe an overall state machine for the card while ISO 7816 specifies other aspects of the card. However, because the overall card state remains fixed in the OP_Secured state while in FIPS mode of operation, the module has only one FSM, that of the JCCOS applet and the corresponding FSM diagram.

The Open Platform card manager manages the states of the Java Card platform and the JCCOS applet life cycle. The module is in the Card manager state OP_Secured once issued to a User. The loading of the JCCOS applet and setting the default SO password takes place in the pre-issuance phase at the vendor's facility. The OP_Secured state is irreversible and cannot be changed to transition to the OP_Locked state without knowledge of the Card Manager keyset. The CardManager keyset is not released to the user by DataKey, disallowing any operator in the post-issuance phase to establish a secure session with the Card Manager thereby performing the CardManager services such as changing the Card Manager state.

8. Physical Security

The Datakey Model 330J Smart Card is a single-chip cryptographic module. It is designed to meet FIPS 140-2 level 3 requirements for physical security.

The 330J IC is a production quality IC. It meets commercial-grade specifications for power, temperature, reliability, and shock / vibration. It uses standard passivation techniques for the entire chip.

In addition to the passivation material, a coating covers the chip. Two different types of epoxies are used. The epoxy material fills a reservoir constructed around the die and wire bonds. The epoxy used on the back of the chip is resistant to solvents that are commonly available.

9. Cryptographic Key Management

9.1 Open Platform Card Manager Keys

The Model 330J implements a card manager compliant with Open Platform specification version 2.01.

The Open Platform loading process is designed to allow the Card Issuer security domain or cryptographic applications provider authorized by the Card Issuer to change the content of the EEPROM memory at certain points in the Open Platform Card Manager life cycle. This includes the OP_READY state and INITIALIZED state when the card is in control of the issuer. In the post-issuance phase when the card is in the SECURED state, applet loading is not allowed (as the a secure session cannot be established with the Card Manager).
The Card Manager (in the context of an Issuer Security Domain) contains keys that the Card Issuer uses in support of cryptographic operations for the Card Issuer's applications.

All Security Domains contain keys (2-key TDES keys) capable of supporting (loading, installing and deleting) applications during personalization. These keys remain within the Card Manager applet and are never released to the end-user in the post-issuance phase of the card life cycle. Thus it is not possible to perform Card manager services such as loading and deleting applets, changing Card manager keyset or changing the overall card state. This is because all such services require a secure session and can be performed only during the pre-issuance phase at the card manufacturer's facility. See Section 7.2.4 for a description of all Card Manager services.

Key Usage	Length	Remark
Authentication & encryption	16 bytes	Mandatory
Message Authentication Code	16 bytes	Mandatory
MAC Verification	16 bytes	Mandatory
Key Encryption Key (KEK)	16 bytes	Mandatory

9.2 JCCOS Applet Keys

JCCOS supports the use of public key cryptography in two primary functions - digital signatures and key management (exchange / agreement) for symmetric encryption keys. While a single RSA key may be used for both functions, best practices recommend using separate key pairs.

Key generation:
The JCCOS applet generates the following types of keys:

- 8 byte single DES
- 16 byte Triple DES
- 1024 and 2048-bit RSA public and private keys

The module uses the FIPS 186-2 Appendix 3.1 compliant (SHA-1 based) PRNG to generate keys.

Key entry and output:

The user may load public keys in key files in plaintext form. However the JCCOS applet allows only the authenticated User to perform cryptographic functions such as Crypt, GenerateDESKey, GenerateRandomNumber, RSADecrypt, , RSAGenerateKey, and RSASign in FIPS mode. Thus only the operator owning the keys can use them for cryptographic purposes. The module does not support loading of secret or private keys. All secret and private key file security nibbles are checked during cryptographic operations to ensure that they are not modifiable or readable by anyone. Thus the applet does not allow entry of secret and private keys. Public keys can be protected against unauthorized modification and disclosure by setting the appropriate public key file security nibbles.

Key Storage:
The keys can be generated internally only by the authenticated User and are stored in plaintext form in the card file system in a container (object) in EEPROM. No user generated secret or private keys can be read, written or updated in the FIPS mode. Thus the module does not allow output of any secret or private keys. The applet employs checks to make sure that the key generation functions GenerateDESKey and RSAGenerateKey will not write to key files that are writeable/updateable/readable by anyone (including the User himself).

The User may also load public keys in appropriate public key files. As seen from Section 7 the User can set file permissions using three bytes of security nibbles during file object creation to allow only the authenticated User to read, modify the public key value.

Key Zeroization:
A key file can be deleted/zeroized by issuing the Delete Object command. Additionally the RemoveAllObjects command can be used to destroy all file objects including all module keys and PIN files.

10. Cryptographic algorithms:

The module only supports the following FIPS-approved algorithms:

- DES (ECB, CBC modes)
- TDES (ECB, CBC modes- 128-bit key length)
- SHA-1
- RSA Sign (1024 and 2048 bit modulus)

The module also supports RSA Decryption (a non-FIPS Approved algorithm). RSA Decrypt may not be used for decrypting data in FIPS mode of operation. This algorithm may only be used by an operator in FIPS mode of operation as part of a key transport scheme.

Additionally the module provides a FIPS 186-2 Appendix 3.1 compliant PRNG.

11. Electromagnetic Interference / Electromagnetic Compatibility (EMI / EMC)

The Model 330J module conforms to the EMI / EMC requirements specified in FCC Part 15, Subpart J, Class B (i.e., for home use).

The Model 330J cryptographic smartcard is a single chip and is a passive device. The reader / writer device shall supply operating power and oscillation frequency to the smartcard.

Refer to FCC certificate included in this documentation package labeled

TEST RESULT SUMMARY
FCC PART 15 SUBPART B
Class A Limit
Test report number M6095 and M6095.1

Lab Name: TUV PRODUCT SERVICES Inc.
Location: Taylors Falls MN USA

12. Self Tests

The JCRE performs the following tests during card reset before an ATR is returned to the host:

- RAM (internal and external).
- The coprocessor.

The Power On Self Test (POST) function tests the following critical systems: Known Answer Test (KAT) of all algorithms supported by the JCCOS applet is performed that includes DES, 2Key TDES, SHA-1, RSA and PRNG.

1- Known answer tests:

- SHA-1
- DES
- 3DES
- RSADecrypt
- RSAEncrypt
- RSASign
- RSAVerify
- PRNG

If any of these tests fail, the applet will enter the Error state and no module functions will be accessible.

2- Pair-wise consistency tests:

- RSADecrypt/RSAEncrypt,
- RSASign/RSAVerify.

The pairwise consistency test is performed each time a key pair is generated. If the key pair generated is used for Exchange only then only an RSADecrypt/RSAEncrypt is performed using the key pair. If the key pair generated is to be used only for digital signatures then simply an RSASign/RSAVerify is performed. If any of these tests fail, the applet throws an exception and exits out of the error condition. No data is output from the module.

3- Continuous random number generator test:

This test is performed each time a block of random data is requested from the FIPS 186-2 compliant PRNG.

If this test fails, the applet will enter the Error state and throw an exception. The cause of the error can be determined with the GetAppletStatus command. The operator must then issue the RemoveAllObjects command to bring the module back to an operational state.

13. Security Guidance

13.1 Development Errors and Oversights

The primary object of concern relative to development errors and oversights is the JCCOS applet byte code, and to a lesser extent the supporting software in the IT environment. The JCCOS byte code is the primary concern because once the code has been embedded into the ROM of the smart card, it is not easy to change.

The steps taken during development of JCCOS to minimize errors and oversights were:

- Use of cryptographic experts for both high-level and code design,
- Frequent discussion among design group members of functional and performance objectives / specifications,
- Frequent design reviews of design approaches,
- Development of extensive test scripts,
- Use of simulation equipment to evaluate applet code with the target IC Java Card platform,
- Extensive testing of pilot production ICs with applet downloaded in EEPROM.

There are currently no known errors or faults in the JCCOS Version 2 applet code.

13.2 Protection Against Careless Users

When or before the card is issued, the end-user should be made aware that the card is an extension of the user's ID and is capable of generating a digital signature for the user, which is as valid and legal as a written signature on a paper document. For this reason the user should also understand that he /she should keep the card on their person or under lock and key when not in use, and to protect their secret pass phrase from observation when logging on.

At the time the card is issued, an initial user pass phrase is in the card. The issuer should be urged to immediately change the initial pass phrase to one, which the user can easily remember, but one, which others cannot easily guess.

These things seem to be simple enough to remember, but in fact require some personal discipline on the part of the user. *Lapses in this discipline can lead to the use of an authorized user's card by an unauthorized user.*

13.3 Protection Against Unauthorized Users

If an unauthorized user can gain access to an authorized user's card **and** pass phrase, the imposter can act in every sense with all the capability as the true owner of the card. This usurpation of the user's identity, at best would be very embarrassing, and at worst extremely costly, to the authorized user.

Other than the above, there are no known vulnerabilities that can come from the end-user's lack of application knowledge or carelessness.

13.4 Protection Against Malicious Administrators

The issuing organization's senior security administrator may be responsible for ensuring that cards are issued with a card configuration file as well as subject / object / operation attributes in accordance with organization security policy, and the administrator is normally assumed to be trustworthy in fulfilling this responsibility.

If this assumption is valid and the administrator is well trained and competent, there are no known vulnerabilities added as a result of the card issuing process.

However, if the security administrator is motivated maliciously, compromise of application security is possible.

There are a number of card configuration bits that are used to implement elements of the organization / application security policy. Some examples of malicious intent would be:

- The policy may state that the administrator cannot unlock 'locked user cards'. If the administrator sets this bit contrary to policy, he / she can collect the 'locked cards', unlock and reissue them to unauthorized users, or use them personally for his own gain.

- The policy may state that cryptographic commands may not be processed from the Idle State. Setting this bit contrary to policy would allow unauthorized user with a card but no pass phrase to perform crypto operations.

13.5 Potential Loss of Secure State

There are no known outsider scenarios or act-of-nature failures that leave the card in an insecure state, in which further attacks could more easily compromise the system security.