Revised Draft NIST Special Publication 800-73-4

Interfaces for Personal Identity Verification – Part 2: PIV Card Application Card Command Interface

Ramaswamy Chandramouli
David Cooper
Hildegard Ferraiolo
Salvatore Francomacaro
Ketan Mehta
Jason Mohler

C O M P U T E R   S E C U R I T Y

National Institute of Standards and Technology
U.S. Department of Commerce

Revised Draft NIST Special Publication 800-73-4

Interfaces for Personal Identity Verification – Part 2: PIV Card Application Card Command Interface

Ramaswamy Chandramouli
David Cooper
Hildegard Ferraiolo
Salvatore Francomacaro
Ketan Mehta
Computer Security Division
Information Technology Laboratory

Jason Mohler
Electrosoft Services, Inc.

May 2014

U.S. Department of Commerce
Penny Pritzker, Secretary

National Institute of Standards and Technology
Patrick D. Gallagher, Under Secretary of Commerce for Standards and Technology and Director

Authority

This publication has been developed by NIST to further its statutory responsibilities under the Federal Information Security Management Act (FISMA), Public Law (P.L.) 107-347. NIST is responsible for developing information security standards and guidelines, including minimum requirements for Federal information systems, but such standards and guidelines shall not apply to national security systems without the express approval of appropriate Federal officials exercising policy authority over such systems. This guideline is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130, Section 8b(3), Securing Agency Information Systems, as analyzed in Circular A-130, Appendix IV: Analysis of Key Sections. Supplemental information is provided in Circular A-130, Appendix III, Security of Federal Automated Information Resources.

National Institute of Standards and Technology Special Publication 800-73-4
Natl. Inst. Stand. Technol. Spec. Publ. 800-73-4, 61 pages (May 2014)
CODEN: NSPUE2

Public comment period: *May 16, 2014* through *June 16, 2014*

National Institute of Standards and Technology
Attn: Computer Security Division, Information Technology Laboratory
100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930
Email: piv_comments@nist.gov

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analyses to advance the development and productive use of information technology. ITL's responsibilities include the development of management, administrative, technical, and physical standards and guidelines for the cost-effective security and privacy of other than national security-related information in Federal information systems. The Special Publication 800-series reports on ITL's research, guidelines, and outreach efforts in information system security, and its collaborative activities with industry, government, and academic organizations.

Abstract

FIPS 201 defines the requirements and characteristics of a government-wide interoperable identity credential. FIPS 201 also specifies that this identity credential must be stored on a smart card. This document, SP 800-73, contains the technical specifications to interface with the smart card to retrieve and use the PIV identity credentials. The specifications reflect the design goals of interoperability and PIV Card functions. The goals are addressed by specifying a PIV data model, card edge interface, and application programming interface. Moreover, this document enumerates requirements where the international integrated circuit card standards [ISO7816] include options and branches. The specifications go further by constraining implementers' interpretations of the normative standards. Such restrictions are designed to ease implementation, facilitate interoperability, and ensure performance, in a manner tailored for PIV applications.

Keywords

authentication; FIPS 201; identity credential; logical access control; on-card biometric comparison; Personal Identity Verification (PIV); physical access control; smart cards; secure messaging

Acknowledgements

The authors (Ramaswamy Chandramouli, David Cooper, Hildegard Ferraiolo, Salvatore Francomacaro, and Ketan Mehta of NIST, and Jason Mohler of Electrosoft Services, Inc.) wish to thank their colleagues who reviewed drafts of this document and contributed to its development. The authors also gratefully acknowledge and appreciate the many contributions from the public and private sectors whose thoughtful and constructive comments improved the quality and usefulness of this publication.

Table of Contents

List of Tables

List of Figures

1. Introduction

Homeland Security Presidential Directive-12 (HSPD-12) called for a common identification standard to be adopted governing the interoperable use of identity credentials to allow physical and logical access to Federally controlled facilities and information systems. Personal Identity Verification (PIV) of Federal Employees and Contractors, Federal Information Processing Standard 201 (FIPS 201) [FIPS201] was developed to establish standards for identity credentials. Special Publication 800-73-4 (SP 800-73-4) contains technical specifications to interface with the smart card (PIV Card[1]) to retrieve and use the identity credentials.

1.1 Purpose

FIPS 201 defines procedures for the PIV lifecycle activities including identity proofing, registration, PIV Card issuance, and PIV Card usage. FIPS 201 also specifies that the identity credentials must be stored on a smart card. SP 800-73-4 contains the technical specifications to interface with the smart card to retrieve and use the identity credentials. The specifications reflect the design goals of interoperability and PIV Card functions. The goals are addressed by specifying a PIV data model, card edge interface, and application programming interface. Moreover, SP 800-73-4 enumerates requirements where the international integrated circuit card (ICC) standards [ISO7816] include options and branches. The specifications go further by constraining implementers' interpretations of the normative standards. Such restrictions are designed to ease implementation, facilitate interoperability, and ensure performance, in a manner tailored for PIV applications.

1.2 Scope

SP 800-73-4 specifies the PIV data model, application programming interface (API), and card interface requirements necessary to comply with the use cases, as defined in Section 6 of FIPS 201 and further described in Appendix B of SP 800-73-4 Part 1. Interoperability is defined as the use of PIV identity credentials such that client-application programs, compliant card applications, and compliant ICCs can be used interchangeably by all information processing systems across Federal agencies. SP 800-73-4 defines the PIV data elements' identifiers, structure, and format. SP 800-73-4 also describes the client application programming interface and card command interface for use with the PIV Card.

This part, SP 800-73-4 Part 2 – *PIV Card Application Card Command Interface,* contains the technical specifications of the PIV Card command interface to the PIV Card. The specification defines the set of commands surfaced by the PIV Card Application at the card edge of the ICC.

1.3 Audience and Assumptions

This document is targeted at Federal agencies and implementers of PIV systems. Readers are assumed to have a working knowledge of smart card standards and applications.

Readers should also be aware of SP 800-73-4 Part 1, Section I, for the revision history of SP 800-73, Section II, which details configuration management recommendations, and Section III, which specifies NPIVP conformance testing procedures. Section 1.3 of Part 1 specifies the effective date of SP 800-73-4.

[1] A physical artifact (e.g., identity card, "smart" card) issued to an individual that contains a PIV Card Application which stores identity credentials (e.g., photograph, cryptographic keys, digitized fingerprint representation) so that the claimed identity of the cardholder can be verified against the stored credentials by another person (human readable and verifiable) or an automated process (computer readable and verifiable).

1.4 Content and Organization

All sections in this document are *normative* (i.e., mandatory for compliance) unless specified as *informative* (i.e., non-mandatory). Following is the structure of Part 2:

+ Section 1, *Introduction*, provides the purpose, scope, audience, and assumptions of the document and outlines its structure.
+ Section 2, *Overview: Concepts and Constructs*, describes the model of computation of the PIV Card Application and the PIV client application programming interface including information processing concepts and data representation constructs.
+ Section 3, *PIV Card Application Card Command Interface*, describes the set of commands accessible by the PIV Middleware to communicate with the PIV Card Application.
+ Section 4, *Secure Messaging*, describes the secure messaging protocol that is used to enable data confidentiality and integrity.
+ Appendix A, *Examples of the Use of the GENERAL AUTHENTICATE Command*, demonstrates the GENERAL AUHTENTICATE command. This section is *informative*.
+ Appendix B, *Terms, Acronyms, and Notation*, contains the list of terms and acronyms used in this document and explains the notation in use. This section is *informative*.
+ Appendix C, *References*, contains the lists of documents used as references by this document. This section is *informative*.

2. Overview: Concepts and Constructs

SP 800-73-4 Parts 2 and 3 define two interfaces to an ICC that contains the PIV Card Application: a low-level card command interface (Part 2) and a high-level client API (Part 3).

The information processing concepts and data constructs on both interfaces are identical and may be referred to generically as the information processing concepts and data constructs on the *PIV interfaces* without specific reference to the client API or the card command interface.

The client API provides task-specific programmatic access to these concepts and constructs and the card command interface provides communication access to concepts and constructs. The client API is used by client applications using the PIV Card Application. The card command interface is used by software implementing the client API (middleware).

The client API is thought of as being at a higher level than the card command interface because access to a single entry point on the client API may cause multiple card commands to traverse the card command interface. In other words, it may require more than one card command on the card command interface to accomplish the task represented by a single call on an entry point of the client API.

The client API is a program execution, call/return style interface whereas the card command interface is a communication protocol, command/response style interface. Because of this difference, the representation of the PIV concepts and constructs as bits and bytes on the client API may be different from the representation of these same concepts and constructs on the card command interface.

2.1.1 Platform Requirements

The following are the requirements that the PIV Card Application places on the ICC platform on which it is implemented or installed:

+ global security status that includes the security status of a global cardholder PIN
+ application selection using a truncated Application Identifier (AID)
+ ability to reset the security status of an individual application
+ indication to applications as to which physical communication interface – contact versus contactless – is in use
+ support for the default selection of an application upon warm or cold reset

2.2 Namespaces of the PIV Card Application

AID, names, Tag-Length-Value (BER-TLV) tags [ISO8825], ASN.1 Object Identifiers (OIDs) [ISO8824] and Proprietary Identifier eXtensions (PIXes) of the NIST Registered Application Provider IDentifier (RID) used on the PIV interfaces are specified in Part 1. Part 1 also specifies that all unspecified names, BER-TLV tags, OIDs, and values of algorithm identifiers, key references, and cryptographic mechanism identifiers, are reserved for future use.

2.3 Card Applications

Each command that appears on the card command interface shall be implemented by a *card application* that is resident on the ICC. The card command enables operations on and with the data objects to which the card application has access.

Each card application shall have a globally unique name called its Application Identifier (AID) [ISO7816, Part 4]. Except for the default applications, access to the card commands and data objects of a card application shall be gained by selecting the card application using its application identifier.[2] The PIX of the AID shall contain an encoding of the version of the card application. The AID of the PIV Card Application is defined in Part 1.

The card application whose commands are currently being used is called the *currently selected application*.

2.3.1 Default Selected Card Application

The card platform shall support a default selected card application. In other words, there shall be a currently selected application immediately after a cold or warm reset. This card application is the default selected card application. The default card application may be the PIV Card Application, or it may be another card application.

2.4 Security Architecture

The security architecture of an ICC is the means by which the security policies governing access to each data object stored on the card are represented within the card.

These security policy representations are applied to all PIV card commands thereby ensuring that the prescribed data policies for the card applications are enforced.

The following subsections describe the security architecture of the PIV Card Application.

2.4.1 Access Control Rule

An *access control rule* shall consist of an *access mode* and a *security condition*. The access mode is an operation that can be performed on a data object. A security condition is a Boolean expression using variables called security statuses that are defined below.

According to an access control rule, the action described by the access mode can be performed on the data object if and only if the security condition evaluates to TRUE for the current values of the security statuses. If there is no access control rule with an access mode describing a particular action, then that action shall never be performed on the data object.

2.4.2 Security Status

Associated with each authenticable entity shall be a set of one or more Boolean variables, each called a *security status indicator* of the authenticable entity. Each security status indicator, in turn, is associated

[2] Access to the default application, and its commands and objects, occurs immediately after a warm or cold card reset without an explicit SELECT command.

with a credential that can be used to authenticate the entity. The security status indicator of an authenticable entity shall be TRUE if the credentials associated with the security status indicator of the authenticable entity have been authenticated and FALSE otherwise.

A successful execution of an authentication protocol shall set the security status indicator associated with the credential used in the protocol to TRUE. An aborted or failed execution of an authentication protocol shall set the security status indicator associated with the credential used in the protocol to FALSE.

As an example, the credentials associated with three security status indicators of the cardholder might be: PIN, fingerprint, and pairing code. Demonstration of knowledge of the PIN is the authentication protocol for the first security status indicator wherein the PIN is the credential. Comparison of the fingerprint template on the card with a fingerprint acquired from the cardholder is the authentication protocol for the second security status indicator wherein the fingerprint is the credential. Demonstration of knowledge of the pairing code is the authentication protocol for the third security status indicator wherein the pairing code is the credential. A security condition using these three security status indicators might be “pairing code **AND** (PIN **OR** fingerprint).”

A security status indicator shall be said to be a *global* security status indicator if it is not changed when the currently selected application changes from one application to another. In essence, when changing from one application to another, the global security status indicators shall remain unchanged.

A security status indicator is said to be an *application* security status indicator if it is set to FALSE when the currently selected application changes from one application to another. Every security status indicator is either a global security status indicator or an application security status indicator. The security status indicators associated with the PIV Card Application PIN, the PIN Unblocking Key (PUK), OCC, pairing code, and the PIV Card Application Administration Key are application security status indicators for the PIV Card Application, whereas the security status indicator associated with the Global PIN is a global security status indicator.

The term *global security status* refers to the set of all global security status indicators. The term *application security status* refers to the set of all application security status indicators for a specific application.

2.4.3 Authentication of an Individual

Knowledge of a PIN is the means by which an individual can be authenticated to the PIV Card Application.

The pairing code shall be exactly 8 bytes in length and the PIV Card Application PIN shall be between 6 and 8 bytes in length. If the actual length of PIV Card Application PIN is less than 8 bytes it shall be padded to 8 bytes with 'FF' when presented to the card command interface. The 'FF' padding bytes shall be appended to the actual value of the PIN. The bytes comprising the PIV Card Application PIN and pairing code shall be limited to values 0x30 – 0x39, the ASCII values for the decimal digits '0' – '9'. For example,

- Actual PIV Card Application PIN: “123456” or '31 32 33 34 35 36'
- Padded PIV Card Application PIN presented to the card command interface: '31 32 33 34 35 36 FF FF'

The PIV Card Application shall enforce the minimum length requirement of six bytes for the PIV Card Application PIN (i.e., shall verify that at least the first six bytes of the value presented to the card command interface are in the range 0x30 – 0x39).

If the Global PIN is used by the PIV Card Application then the above encoding, length, padding, and enforcement of minimum PIN length requirements for the PIV Card Application PIN shall apply to the Global PIN.

The PUK shall be 8 bytes in length, and may be any 8-byte binary value. That is, the bytes comprising the PUK may have any value in the range 0x00 – 0xFF.

2.5 Current State of the PIV Card Application

The elements of the *current state* of the PIV Card Application when the PIV Card Application is the currently selected application are described in Table 1.

Table 1. State of the PIV Card Application

State Name	Always Defined	Comment	Location of State
Global security status	Yes	Contains security status indicators that span all card applications on the platform.	PIV Platform
Currently selected application	Yes	The platform shall support the selection of a card application using the full application identifier or by providing the right-truncated version and there shall always be a currently selected application.	PIV Platform
Application security status	Yes	Contains security status indicators local to the PIV Card Application.	PIV Card Application

3. PIV Card Application Card Command Interface

Table 2 lists the card commands surfaced by the PIV Card Application at the card edge of the ICC when it is the currently selected card application. All PIV Card Application card commands shall be supported by a PIV Card Application. Card commands indicated with a 'Yes' in the Command Chaining column shall support command chaining for transmitting a data string too long for a single command as defined in [ISO7816].

Table 2. PIV Card Application Card Commands

Type	Name	Contact Interface	Contactless Interface	Security Condition for Use	Command Chaining
PIV Card Application Card Commands for Data Access	**SELECT**	Yes	Yes	Always	No
	GET DATA	Yes	Yes	Data Dependent. See Table 2, Part 1.	No
PIV Card Application Card Commands for Authentication	**VERIFY**	Yes	SM or VCI (see Note 1)	Always	Yes[3]
	CHANGE REFERENCE DATA	Yes	VCI	PIN	No
	RESET RETRY COUNTER	Yes	No	PIN Unblocking Key	No
	GENERAL AUTHENTICATE	Yes	Yes (See Note 2)	Key Dependent. See Table 4, Part 1.	Yes
PIV Card Application Card Commands for Credential Initialization and Administration	**PUT DATA**	Yes	No	PIV Card Application Administrator	Yes
	GENERATE ASYMMETRIC KEY PAIR	Yes	No	PIV Card Application Administrator	Yes

The PIV Card Application shall return the status word of '6A 81' (Function not supported) when it receives a card command on the contactless interface marked “No” in the Contactless Interface column in Table 2.

Note 1: For SM, OCC and pairing code alone can be submitted via secure messaging (SM) over the contactless interface. All other key references require VCI for communication over the contactless interface.

Note 2: Cryptographic protocols using private/secret keys that require the “PIN” or “OCC” security condition shall only be used on the contactless interface after a Virtual Contact Interface (VCI) has been established. The term VCI is used in this document as a shorthand for a security condition in which secure messaging is used **AND** the security status indicator associated with the pairing code is TRUE.” (copied from Part 1)

[3] The VERIFY command is only required to support command chaining if the PIV Card Application supports on-card biometric comparison (OCC).

3.1 PIV Card Application Card Commands for Data Access

3.1.1 SELECT Card Command

The SELECT card command sets the currently selected application. The PIV Card Application shall be selected by providing its application identifier (see Part 1, Section 2.2) in the data field of the SELECT command.

There shall be at most one PIV Card Application on any ICC. The PIV Card Application can also be made the currently selected application by providing the right-truncated version (see Part 1, Section 2.2); that is, without the two-byte version number in the data field of the SELECT command.

The complete AID, including the two-byte version, of the PIV Card Application that became the currently selected card application upon successful execution of the SELECT command (using the full or right-truncated PIV AID) shall be returned in the application property template.

If the currently selected application is the PIV Card Application when the SELECT command is given and the AID in the data field of the SELECT command is either the AID of the PIV Card Application or the right-truncated version thereof, then the PIV Card Application shall continue to be the currently selected card application and the setting of all security status indicators in the PIV Card Application shall be unchanged.

If the currently selected application is the PIV Card Application when the SELECT command is given and the AID in the data field of the SELECT command is not the PIV Card Application (or the right-truncated version thereof), but a valid AID supported by the ICC, then the PIV Card Application shall be deselected and all the PIV Card Application security status indicators in the PIV Card Application shall be set to FALSE.

If the currently selected application is the PIV Card Application when the SELECT command is given and the AID in the data field of the SELECT command is an invalid AID not supported by the ICC, then the PIV Card Application shall remain the currently selected application and all PIV Card Application security status indicators shall remain unchanged.

Command Syntax

CLA	'00'
INS	'A4'
P1	'04'
P2	'00'
L_c	Length of application identifier
Data Field	AID of the PIV Card Application using the full AID or the right-truncated AID (See Section 2.2, Part 1)
L_e	'00'

Response Syntax

Data Field	Application property template (APT). See Table 3 below
SW1-SW2	Status word

Upon selection, the PIV Card Application shall return the application property template described in Table 3.

Table 3. Data Objects in the PIV Card Application Property Template (Tag '61')

Description	Tag	M/O/C	Comment
Application identifier of application	'4F'	M	The PIX of the AID includes the encoding of the version of the PIV Card Application. See Section 2.2, Part 1.
Coexistent tag allocation authority	'79'	M	Coexistent tag allocation authority template. See Table 4.
Application label	'50'	O	Text describing the application; e.g., for use on a man-machine interface.
Uniform resource locator	'5F50'	O	Reference to the specification describing the application.
Cryptographic algorithms supported	'AC'	C	Cryptographic algorithm identifier template. See Table 5.

Table 4. Data Objects in a Coexistent Tag Allocation Authority Template (Tag '79')

Name	Tag	M/O	Comment
Application identifier	'4F'	M	See Section 2.2, Part 1

A PIV Card Application may use a subset of the cryptographic algorithms defined in SP 800-78. Tag 0xAC encodes the cryptographic algorithms supported by the PIV Card Application. The encoding of tag 0xAC shall be as specified in Table 5. Each instance of tag 0x80 shall encapsulate one algorithm. The presence of algorithm identifier '27' or '2E' indicates that the corresponding cipher suite is supported by the PIV Card Application for secure messaging and that the PIV Card Application possesses a PIV Secure Messaging key of the appropriate size for the specified cipher suite. Tag 0xAC shall be present and indicate algorithm identifier 0x27 and/or 0x2E when the PIV Card Application supports secure messaging.

Table 5. Data Objects in a Cryptographic Algorithm Identifier Template (Tag 'AC')

Name	Tag	M/O	Comment
Cryptographic algorithm identifier	'80'	M	For values see [SP800-78, Table 6-2]
Object identifier	'06'	M	Its value is set to 0x00

SW1	SW2	Meaning
'6A'	'82'	Application not found
'90'	'00'	Successful execution

3.1.2 GET DATA Card Command

The GET DATA card command retrieves the data content of the single data object whose tag is given in the data field.[4]

Command Syntax

CLA	'00' or '0C' for secure messaging
INS	'CB'
P1	'3F'
P2	'FF'
L_c	Length of data field*
Data Field	See Table 6
L_e	'00'

* The L_c value is '05' for all PIV data objects except for the 0x7E interindustry tag (Discovery Object), which has an L_c value of '03', and the 0x7F61 interindustry tag (Biometric Information Templates (BIT) Group Template), which has an L_c value of '04'.

Table 6. Data Objects in the Data Field of the GET DATA Card Command

Name	Tag	M/O	Comment
Tag list	'5C'	M	BER-TLV tag of the data object to be retrieved. See Table 3, Part 1.

Response Syntax

For the 0x7E Discovery Object (if present) and the 0x7F61 BIT Group Template (if present):

Data Field	- BER-TLV of the 0x7E Discovery data object (see Section 3.3.2, Part 1 for a description of the Discovery Object's structure returned in the data field) or - BER-TLV of the 0x7F61 BIT Group Template (see Table 7 of SP 800-76)
SW1-SW2	Status word

For all other PIV data objects (if present):

Data Field	BER-TLV with the tag '53' containing in the value field of the requested data object.
SW1-SW2	Status word

SW1	SW2	Meaning
'61'	'xx'	Successful execution where SW2 encodes the number of response data bytes still available
'69'	'82'	Security status not satisfied
'6A'	'82'	Data object not found
'90'	'00'	Successful execution

[4] The GET RESPONSE command is used in conjunction with GET DATA to accomplish the reading of larger PIV data objects. The GET RESPONSE command is illustrated in Appendix A.4.1 (Command 3).

3.2 PIV Card Application Card Commands for Authentication

3.2.1 VERIFY Card Command

The VERIFY card command initiates the comparison in the card of the reference data indicated by the key reference with authentication data in the data field of the command.

Key reference '80' specific to the PIV Card Application (i.e., local key references) and, optionally, the Global PIN with key reference '00', the OCC data (key reference '96'), and pairing code (key reference '98') are the only key references that may be verified by the PIV Card Application's VERIFY command.

Key reference '80' shall be able to be verified by the PIV Card Application VERIFY command.

If the PIV Card Application contains the Discovery Object as described in Part 1 and the first byte of the PIN Usage Policy value is 0x60, 0x68, 0x70, or 0x78, then key reference '00' shall be able to be verified by the PIV Card Application VERIFY command.

If the PIV Card Application contains the Discovery Object as described in Part 1 and the first byte of the PIN Usage Policy value is 0x50, 0x58, 0x70, or 0x78, then key reference '98' shall be able to be verified by the PIV Card Application VERIFY command.

If the PIV Card Application contains the Discovery Object as described in Part 1 and the first byte of the PIN Usage Policy value is 0x48, 0x58, 0x68, or 0x78, then key reference '96' shall be able to be verified by the PIV Card Application VERIFY command.

If the key reference is '00' or '80' and the VERIFY command is not submitted over either the contact interface or the VCI, or if the key reference is '96' or '98' and the VERIFY command is submitted over the contactless interface without secure messaging, then the card command shall fail, and the PIV Card Application shall return the status word '6A 81'. The security status and the retry counter of the key reference shall remain unchanged.

If the key reference is '98' and the authentication data in the command data field does not match the reference data associated with the key reference, the PIV Card Application shall return the status word '63 00'. If the authentication data in the command data field does not satisfy the criteria in Section 2.4.3 then the PIV Card Application may return the status word '6A 80' instead of '63 00'. In either case the command shall fail and the security status of the key reference shall be set to FALSE.

If the key reference is '00', '80', or '96' and the current value of the retry counter associated with the key reference is zero, then the comparison shall not be made, and the PIV Card Application shall return the status word '69 83'.[5]

If the key reference is '00' or '80' and the authentication data in the command data field does not satisfy the criteria in Section 2.4.3 then the card command shall fail and the PIV Card Application shall return either the status word '6A 80' or '63 CX'. If status word '6A 80' is returned, the security status and the retry counter of the key reference shall remain unchanged.[6] If status word '63 CX' is returned, the security status of the key reference shall be set to FALSE and the retry counter associated with the key reference shall be decremented by one.

[5] There is no retry counter associated with the pairing code, and so the authentication method cannot be blocked for that key reference.

[6] It is recommended that in this case the authentication data not be compared to the on-card reference data.

If the key reference is '96' and the authentication data in the command data field is not of length 3N, where N satisfies the requirements for minimum and maximum number of minutiae specified in at least one of the BITs in the BIT Group Template, then the card command shall fail, and the PIV Card Application shall return the status word '6A 80'. The security status and the retry counter of the key reference shall remain unchanged.

If the key reference is '00', '80', or '96' and the authentication data in the command data field is properly formatted (see previous two paragraphs) and does not match reference data associated with the key reference, then the card command shall fail, the PIV Card Application shall return the status word '63 CX', the security status of the key reference shall be set to FALSE, and the retry counter associated with the key reference shall be decremented by one.

If the card command succeeds then the security status of the key reference shall be set to TRUE. If the key reference is '00', '80', or '96' then the retry counter associated with the key reference shall be set to the reset retry value associated with the key reference. The initial value of the retry counter and the reset retry value associated with the key reference, i.e., the number of successive failures (retries) before the retry counter associated with the key reference reaches zero, are issuer dependent.

The VERIFY command shall reset the security status of the key reference in P2 when the P1 parameter is 'FF' and both L_c and the data field are absent. The security status of the key reference specified in P2 shall be set to FALSE and the retry counter associated with the key reference shall remain unchanged.

Command Syntax

CLA	'00' or '10' indicating command chaining '0C' or '1C' for secure messaging
INS	'20'
P1	'00' or 'FF'
P2	Key reference. See Part 1, Table 4.
L_c	Absent[7] – for absent command data field '08' – for PIV Card Application PIN, Global PIN, or pairing code 3N – for OCC data (where N is the number of minutiae)
Data Field	Absent,[7] PIV Card Application PIN, Global PIN, or pairing code authentication data as described in Section 2.4.3, or OCC data as described in Section 5.5.2 of [SP800-76].
L_e	Absent

Note: For key reference '96', if the BIT Group Template includes BITs for two fingers then verification shall succeed if the authentication data in the data field of the command matches either the primary finger OCC reference data (key reference '96') or the secondary finger OCC reference data (key reference '97'). If the number of minutiae in the authentication data in the data field only satisfies the requirements in the BITs for minimum and maximum number of minutiae for one of the two fingers then only the reference data for that finger shall be compared against the authentication data in the data field.

[7] If P1='00', and L_c and the command data field are absent, the command can be used to retrieve the number of further retries allowed ('63 CX'), or to check whether verification is not needed ('90 00').

Response Syntax

SW1	SW2	Meaning
'63'	'00'	Verification failed
'63'	'CX'	Verification failed, X indicates the number of further allowed retries
'69'	'83'	Authentication method blocked
'6A'	'80'	Incorrect parameter in command data field
'6A'	'81'	Function not supported
'6A'	'88'	Key reference not found
'90'	'00'	Successful execution

3.2.2 CHANGE REFERENCE DATA Card Command

The CHANGE REFERENCE DATA card command initiates the comparison of the authentication data in the command data field with the current value of the reference data and, if this comparison is successful, replaces the reference data with new reference data.

Only reference data associated with key references '80' and '81' specific to the PIV Card Application (i.e., local key reference) and the Global PIN with key reference '00' may be changed by the PIV Card Application CHANGE REFERENCE DATA command. If any other key reference value is specified the PIV Card Application shall return the status word '6A 81'. Key reference '80' reference data shall be changed by the PIV Card Application CHANGE REFERENCE DATA command. The ability to change reference data associated with key references '81' and '00' using the PIV Card Application CHANGE REFERENCE DATA command is optional.

If the CHANGE REFERENCE DATA command is not submitted over either the contact interface or the VCI then the card command shall fail and the PIV Card Application shall return the status word '6A 81'. The security status and the retry counter of the key reference shall remain unchanged.

If the current value of the retry counter associated with the key reference is zero, then the reference data associated with the key reference shall not be changed and the PIV Card Application shall return the status word '69 83'.

If the authentication data in the command data field does not match the current value of the reference data or if either the authentication data or the new reference data in the command data field of the command does not satisfy the criteria in Section 2.4.3, the PIV Card Application shall not change the reference data associated with the key reference and shall return either status word '6A 80' or '63 CX', with the following restrictions. If the authentication data in the command data field satisfies the criteria in Section 2.4.3 and matches the current value of the reference data, but the new reference data in the command data field of the command does not satisfy the criteria in Section 2.4.3 the PIV Card Application shall return status word '6A 80'. If the authentication data in the command data field does not match the current value of the reference data, but both the authentication data and the new reference data in the command data field of the command satisfy the criteria in Section 2.4.3, the PIV Card Application shall return status word '63 CX'. If status word '6A 80' is returned, the security status and retry counter associated with the key reference shall remain unchanged.[8] If status word '63 CX' is returned, the security status of the key reference shall be set to FALSE and the retry counter associated with the key reference shall be decremented by one.

[8] It is recommended that in this case the authentication data not be compared to the on-card reference data.

If the card command succeeds, then the security status of the key reference shall be set to TRUE and the retry counter associated with the key reference shall be set to the reset retry value associated with the key reference.

The initial value of the retry counter and the reset retry value associated with the key reference, i.e., the number of successive failures (retries) before the retry counter associated with the key reference reaches zero, is issuer dependent.

Command Syntax

CLA	'00' or '0C' for secure messaging
INS	'24'
P1	'00'
P2	'00' (Global PIN), '80' (PIV Card Application PIN), or '81' (PUK)
L_c	'10'
Data Field	Current PIN authentication data concatenated without delimitation with the new PIN reference data, both PINs as described in Section 2.4.3
L_e	Absent

Response Syntax

SW1	SW2	Meaning
'63'	'CX'	Reference data change failed, X indicates the number of further allowed retries or resets
'69'	'83'	Reference data change operation blocked
'6A'	'80'	Incorrect parameter in command data field
'6A'	'81'	Function not supported
'6A'	'88'	Key reference not found
'90'	'00'	Successful execution

3.2.3 RESET RETRY COUNTER Card Command

The RESET RETRY COUNTER card command resets the retry counter of the PIN to its initial value and changes the reference data. The command enables recovery of the PIV Card Application PIN in the case that the cardholder has forgotten the PIV Card Application PIN.

The only key reference allowed in the P2 parameter of the RESET RETRY COUNTER command is the PIV Card Application PIN. Any other key references in P2 shall not be permitted and the PIV Card Application shall return the status word '6A 81'.[9]

If the current value of the PUK's retry counter is zero then the PIN's retry counter shall not be reset and the PIV Card Application shall return the status word '69 83'.

[9] The PIV Card Application may be implemented to reset the retry counter associated with OCC data when new OCC data is loaded onto the card.

If the reset retry counter authentication data (PUK) in the command data field of the command does not match reference data associated with the PUK then the PIV Card Application shall return the status word '63 CX'. If the new reference data (PIN) in the command data field of the command does not satisfy the criteria in Section 2.4.3 then the PIV Card Application shall return the status word '6A 80'. If the reset retry counter authentication data (PUK) in the command data field of the command does not match reference data associated with the PUK and the new reference data (PIN) in the command data field of the command does not satisfy the criteria in Section 2.4.3 then the PIV Card Application shall return either status word '6A 80' or '63 CX'. If the PIV Card Application returns status word '6A 80' then the retry counter associated with the PIN shall not be reset, the security status of the PIN's key reference shall remain unchanged, and the PUK's retry counter shall remain unchanged.[10] If the PIV Card Application returns status word '63 CX' then the retry counter associated with the PIN shall not be reset, the security status of the PIN's key reference shall be set to FALSE, and the PUK's retry counter shall be decremented by one.

If the card command succeeds then the PIN's retry counter shall be set to its reset retry value. Optionally, the PUK's retry counter may be set to its initial reset retry value. The security status of the PIN's key reference shall not be changed.

The initial retry counter associated with the PUK, i.e., the number of failures of the RESET RETRY COUNTER command before the PUK's retry counter reaches zero, is issuer dependent.

Command Syntax

CLA	'00'
INS	'2C'
P1	'00'
P2	'80' (PIV Card Application PIN).
L_c	'10'
Data Field	Reset retry counter authentication data (PUK) concatenated without delimitation with the new reference data (PIN) (both PUK and PIN as described in Section 2.4.3)
L_e	Absent

Response Syntax

SW1	SW2	Meaning
'63'	'CX'	Reset failed, X indicates the number of further allowed resets
'69'	'83'	Reset operation blocked
'6A'	'80'	Incorrect parameter in command data field
'6A'	'81'	Function not supported
'6A'	'88'	Key reference not found
'90'	'00'	Successful execution

[10] It is recommended that in this case the authentication data not be compared to the on-card reference data.

3.2.4 GENERAL AUTHENTICATE Card Command

The GENERAL AUTHENTICATE card command performs a cryptographic operation, such as an authentication protocol, using the data provided in the data field of the command and returns the result of the cryptographic operation in the response data field.[11]

The GENERAL AUTHENTICATE command shall be used with the PIV authentication keys ('9A', '9B', '9E') to authenticate the card or a card application to the client application (INTERNAL AUTHENTICATE), to authenticate an entity to the card (EXTERNAL AUTHENTICATE), and to perform a mutual authentication between the card and an entity external to the card (MUTUAL AUTHENTICATE).

The GENERAL AUTHENTICATE command shall be used with the digital signature key ('9C') to realize the signing functionality on the PIV client application programming interface. Data to be signed is expected to be hashed off card. Appendix A.4 illustrates the use of the GENERAL AUTHENTICATE command for signature generation.

The GENERAL AUTHENTICATE command shall be used with the key management key ('9D') and the retired key management keys ('82' – '95') to realize key establishment schemes specified in SP 800-78 (ECDH and RSA). Appendix A.5 illustrates the use of the GENERAL AUTHENTICATE command for key establishment schemes aided by the PIV Card Application.

The GENERAL AUTHENTICATE command shall be used with the PIV Secure Messaging key ('03') and cryptographic algorithm identifier '27' or '2E' to establish session keys for secure messaging as specified in Section 4. If key reference '03' is specified in P2 then algorithm identifiers in P1 other than '27' and '2E' shall not be permitted and the PIV Card Application shall return the status word '6A 86'.

The GENERAL AUTHENTICATE command supports command chaining to permit the uninterrupted transmission of long command data fields to the PIV Card Application. If a card command other than the GENERAL AUTHENTICATICATE command is received by the PIV Card Application before the termination of a GENERAL AUTHENTICATE chain, the PIV Card Application shall rollback to the state it was in immediately prior to the reception of the first command in the interrupted chain. In other words, an interrupted GENERAL AUTHENTICATE chain has no effect on the PIV Card Application.

Command Syntax

CLA	'00' or '10' indicating command chaining '0C' or '1C' for secure messaging
INS	'87'
P1	Algorithm reference. See Table 14 and [SP800-78, Table 6-2]
P2	Key reference. See Table 4, Part 1 for key reference values
L_c	Length of data field
Data Field	See Table 7
L_e	Absent or '00'

[11] For cryptographic operations with larger keys, e.g., RSA 2048, the GET RESPONSE command is used to return the complete result of the cryptographic operation. The GET RESPONSE command is illustrated in Appendix A.4.1 (Command 3).

Table 7. Data Objects in the Dynamic Authentication Template (Tag '7C')

Name	Tag	M/O	Description
Witness	'80'	C	Demonstration of knowledge of a fact without revealing the fact. An empty witness is a request for a witness.
Challenge	'81'	C	One or more random numbers or byte sequences to be used in the authentication protocol.
Response	'82'	C	A sequence of bytes encoding a response step in an authentication protocol.
Exponentiation	'85'	C	A parameter used in ECDH key agreement protocol.

The data objects that appear in the dynamic authentication template (tag '7C') in the data field of the GENERAL AUTHENTICATE card command depend on the authentication protocol being executed. The Witness (tag '80') contains encrypted data (unrevealed fact). This data is decrypted by the card. The Challenge (tag '81') contains clear data (byte sequence), which is encrypted by the card. The Response (tag '82') contains either the decrypted data from tag '80' or the encrypted data from tag '81'. Note that the empty tags (i.e., tags with no data) return the same tag with content (they can be seen as "requests for requests"):

+ '80 00' Returns '80 TL <encrypted random>' (as per definition)
+ '81 00' Returns '81 TL <random>' (as per external authenticate example)

Response Syntax

Data Field	Absent, authentication-related data, signed data, shared secret, or transported key
SW1-SW2	Status word

SW1	SW2	Meaning
'61'	'xx'	Successful execution where SW2 encodes the number of response data bytes still available
'69'	'82'	Security status not satisfied
'6A'	'80'	Incorrect parameter in command data field
'6A'	'86'	Incorrect parameter in P1 or P2
'90'	'00'	Successful execution

3.3 PIV Card Application Card Commands for Credential Initialization and Administration

3.3.1 PUT DATA Card Command

The PUT DATA card command completely replaces the data content of a single data object in the PIV Card Application with new content.

Command Syntax

CLA	'00' or '10' indicating command chaining
INS	'DB'
P1	'3F'
P2	'FF'
L_c	Length of data field
Data Field	See Tables 8, 9, and 10
L_e	Absent

For the 0x7E Discovery Object:

Table 8. Data Field of the PUT DATA Card Command for the Discovery Object

Tag	M/O	Description
'7E'	M	BER-TLV of tag '7E' as illustrated in Section 3.3.2, Part 1.

For the 0x7F61 BIT Group Template:

Table 9. Data Field of the PUT DATA Card Command for the BIT Group Template

Tag	M/O	Description
'7F61'	M	BER-TLV of tag '7F61' as illustrated in Table 7 of SP 800-76

For all other PIV Data objects:

Table 10. Data Field of the PUT DATA Card Command for all other PIV Data Objects

Name	Tag	M/O	Description
Tag list	'5C'	M	Tag of the data object whose data content is to be replaced. See Table 3, Part 1.
Data	'53'	M	Data with tag '53' as an unstructured byte sequence.

Response Syntax

Data Field	Absent
SW1-SW2	Status word

SW1	SW2	Meaning
'69'	'82'	Security status not satisfied
'6A'	'81'	Function not supported
'6A'	'84'	Not enough memory
'90'	'00'	Successful execution

3.3.2 GENERATE ASYMMETRIC KEY PAIR Card Command

The GENERATE ASYMMETRIC KEY PAIR card command initiates the generation and storing in the card of the reference data of an asymmetric key pair, i.e., a public key and a private key. The public key of the generated key pair is returned as the response to the command. If there is reference data currently associated with the key reference, it is replaced in full by the generated data.

Command Syntax

CLA	'00' or '10' indicating command chaining
INS	'47'
P1	'00'
P2	Key reference '03', '9A', '9C', '9D', or '9E'.
L_c	Length of data field
Data Field	Control reference template. See Table 11
L_e	'00'

Table 11. Data Objects in the Template (Tag 'AC')

Name	Tag	M/O	Description
Cryptographic mechanism identifier	'80'	M	See Part 1, Table 5
Parameter	'81'	C	Specific to the cryptographic mechanism

Response Syntax

Data Field	Data objects of public key of generated key pair. See Table 12
SW1-SW2	Status word

Table 12. Data Objects in the Template (Tag '7F49')

Name	Tag
Public key data objects for RSA	
Modulus	'81'
Public exponent	'82'
Public key data objects for ECC	
Point	'86'

The public key data object in tag '86' is encoded as follows:

Table 13. Public Key encoding for ECC

Tag	Length	Value
'86'	L	04 || X || Y [SECG, Section 2.3.3]

Note: The octet '04' indicates that the X and Y coordinates of point P are encoded without the use of point compression. The length L is 65 bytes for points on Curve P-256 and 97 bytes for points on Curve P-384.

SW1	SW2	Meaning
'61'	'xx'	Successful execution where SW2 encodes the number of response data bytes still available
'69'	'82'	Security status not satisfied
'6A'	'80'	Incorrect parameter in command data field; e.g., unrecognized cryptographic mechanism
'6A'	'81'	Function not supported
'6A'	'86'	Incorrect parameter P2; cryptographic mechanism of reference data to be generated different than cryptographic mechanism of reference data of given key reference
'90'	'00'	Successful execution

4. Secure Messaging

If a PIV Card Application implements the optional secure messaging protocol for non-card-management operations, it shall be implemented as specified in this section. Secure messaging is initiated through the use of a key establishment protocol. The key establishment protocol defined here is a one-way authentication protocol that authenticates the PIV Card Application to the client application and establishes a set of session keys that may be subsequently used to protect the communication channel between the two parties.[12] PIV Cards may implement a different secure messaging protocol for card management operations. Such a protocol is outside of the scope of this document, however, if it is to be used for remote post issuance updates it shall satisfy the requirements of [FIPS201, Section 2.9.2].

Section 4.1 describes the key establishment protocol used to support secure messaging in the PIV Card Application. Section 4.2 describes the use of secure messaging to protect commands and responses sent between the client application and the PIV Card Application.

4.1 The Key Establishment Protocol

The key establishment protocol for the PIV Card Application uses the One-Pass Diffie-Hellman, C(1e, 1s, ECC CDH) Scheme from [SP800-56A] in a manner that is based on a simplified profile of OPACITY with Zero Key Management [ANSI504-1], as depicted below.

Client Application (H)			**PIV Card Application (ICC)**	
CB_H = 0x00	H1			
Generate an ephemeral key pair (d_{eH}; Q_{eH}) from the domain parameters specified in the response to the SELECT command	H2			
Send $ID_{sH} \parallel Q_{eH} \parallel CB_H$	H3	$ID_{sH} \parallel Q_{eH} \parallel CB_H$ →		
			$ID_{sICC} = T_8(SHA256(C_{ICC}^*))$	C1
			$CB_{ICC} = CB_H$ & 'F0'	C2
			Check that CB_{ICC} is 0x00	C3
			Verify that Q_{eH} is a valid public key for the domain parameters of Q_{sICC}	C4
			$Z = ECC\_CDH(d_{sICC}, Q_{eH})$	C5
			Generate nonce N_{ICC}	C6
			$SK_{CFRM} \parallel SK_{MAC} \parallel SK_{ENC} \parallel SK_{RMAC}$ = KDF(Z, *len*, *OtherInfo*)	C7
			Zeroize Z	C8
			$AuthCryptogram_{ICC}$ = CMAC(SK_{CRFM}, "KC_1_V" $\parallel ID_{sICC} \parallel ID_{sH} \parallel Q_{eH}$)	C9
			Zeroize SK_{CFRM}	C10
		← $CB_{ICC} \parallel N_{ICC} \parallel AuthCryptogram_{ICC} \parallel GUID \parallel C_{ICC}^*$	Return $CB_{ICC} \parallel N_{ICC} \parallel AuthCryptogram_{ICC} \parallel GUID \parallel C_{ICC}^*$	C11
Check that CB_{ICC} is 0x00	H4			
Build C_{ICC} from C_{ICC}^* and GUID	H5			
Verify C_{ICC} signature and subject public key	H6			
$ID_{sICC} = T_8(SHA256(C_{ICC}^*))$	H7			
Extract Q_{sICC} from C_{ICC}	H8			
$Z = ECC\_CDH(d_{eH}, Q_{sICC})$	H9			
Zeroize d_{eH}	H10			
$SK_{CFRM} \parallel SK_{MAC} \parallel SK_{ENC} \parallel SK_{RMAC}$ = KDF(Z, *len*, *OtherInfo*)	H11			
Zeroize Z	H12			
Check that $AuthCryptogram_{ICC}$ equals CMAC(SK_{CFRM}, "KC_1_V" $\parallel ID_{sICC} \parallel ID_{sH} \parallel Q_{eH}$)	H13			
Zeroize SK_{CFRM}	H14			

[12] The protocol does not provide forward secrecy.

Sections 4.1.1 and 4.1.2 provide additional details about each of the protocol steps performed by the client application and the PIV Card Application, and Section 4.1.3 defines the notations used in the description of the protocol. Section 4.1.4 provides the details of the two cipher suites that may be supported by the PIV Card Application. Section 4.1.5 specifies the format for the secure messaging card verifiable certificate (CVC) that is used to authenticate the PIV Card Application and for the optional Intermediate CVC that is used to verify the signature on the secure messaging CVC when the public key needed to verify the signature on the secure messaging CVC does not appear in an X.509 content signing certificate. Section 4.1.6 provides additional information about the key derivation function (KDF) used to derive the session keys that are used during secure messaging, and Section 4.1.7 provides additional information about the computation of the authentication cryptogram for key confirmation. Section 4.1.8 demonstrates the use of the GENERAL AUTHENTICATE command to perform the key establishment protocol.

4.1.1 Client Application Steps

Step #	Description	Comment
H1	Set CB_H to 0x00	The client application's control byte is set to 0x00 to indicate the client application does not support persistent binding, wants the GUID returned in unencrypted form, and wants 3 session keys to be generated.
H2	Generate an ephemeral key pair (d_{eH}; Q_{eH})	Generate an ephemeral ECC key pair for the client application using an **approved** method [FIPS186, Appendix B] and perform full public-key validation [SP800-56A, Section 5.6.2.3.2], either as part of the key generation process or as a separate process. If the 0xAC tag of the application property template (APT) includes '27' then generate an ephemeral key pair over Curve P-256. If the 0xAC tag of the APT includes '2E' then generate an ephemeral key pair over Curve P-384.
H3	Send $ID_{sH} \parallel Q_{eH} \parallel CB_H$	
Wait for response from PIV Card Application: $CB_{ICC} \parallel N_{ICC} \parallel AuthCryptogram_{ICC} \parallel GUID \parallel C_{ICC}$*		
H4	Check that CB_{ICC} is 0x00	Verify that the card executed the protocol in accordance with the parameters specified in Step H1. Return an authentication error if check fails.

Step #	Description	Comment
H5	Build C_{ICC} from C_{ICC}* and GUID	C_{ICC}* is a transformation of the PIV Card's CVC, C_{ICC} (see Section 4.1.5). C_{ICC}* is constructed from C_{ICC} by replacing the Subject Identifier of C_{ICC} (T=0x5F20, L=16, V=GUID) with (T=0x5F20, L=0), changing the CVC's tag from 0x7F21 to 0x7F22, and leaving all other fields of the CVC unchanged, including the DigitalSignature object. Build C_{ICC} by replacing the empty Subject Identifier (T=0x5F20, L=0) in C_{ICC}* with (T=0x5F20, L=16, V=GUID) and by changing the CVC's tag from 0x7F22 to 0x7F21.
H6	Verify C_{ICC} signature and subject public key	Verify signature on C_{ICC} and, using standards-compliant PKI path validation, validate the content signing certificate needed to verify the signature on C_{ICC}.[13,14] Verify that the domain parameters of the subject public key in C_{ICC} are the same as the domain parameters for Q_{eH} by checking the Algorithm OID in the CardHolderPublicKey Data Object (see Table 15). Return an authentication error if either verification fails.
H7	$ID_{sICC} = T_8(SHA256(C_{ICC}*))$	ID_{sICC}, the left-most 8 bytes of the SHA-256 hash of C_{ICC}*, is used as an input for session key derivation.
H8	Extract Q_{sICC} from C_{ICC}	
H9	Z = ECC_CDH (d_{eH}, Q_{sICC})	Compute the shared secret, Z, using the ECC CDH primitive [SP800-56A, Section 5.7.1.2].
H10	Zeroize d_{eH}	Destroy the ephemeral private key generated in Step H2.
H11	$SK_{CFRM} \| SK_{MAC} \| SK_{ENC} \| SK_{RMAC}$ = KDF(Z, *len*, *OtherInfo*)	Compute the key confirmation key and the session keys. See Section 4.1.6.
H12	Zeroize Z	Destroy the shared secret generated in Step H9.
H13	Check that $AuthCryptogram_{ICC}$ equals CMAC(SK_{CFRM}, "KC_1_V" || ID_{sICC} || ID_{sH} || Q_{eH})	Perform key confirmation by verifying the authentication cryptogram as described in Section 4.1.7. Return an authentication error if verification fails.

[13] If the public key needed to verify the signature on C_{ICC} appears in an Intermediate CVC then verify the signatures on both C_{ICC} and the Intermediate CVC and, using standards-compliant PKI validation, validate the content signing certificate needed to verify the signature on the Intermediate CVC.

[14] Validation of the content signing certificate does not need to be performed at the time of signature verification if the certificate has been previously validated or if the public key needed to verify the signature on C_{ICC} has been previously obtained from a trusted source.

Step #	Description	Comment
H14	Zeroize SK_{CFRM}	Destroy the key confirmation key derived in Step H11.

4.1.2 PIV Card Application Protocol Steps

Step #	Description	Comment
C1	$ID_{sICC} = T_8(SHA256(C_{ICC}*))$	ID_{sICC}, the left-most 8 bytes of the SHA-256 hash of $C_{ICC}*$ is used as an input for session key derivation. See Step H5 for construction of $C_{ICC}*$ (Note that ID_{sICC} and $C_{ICC}*$ are static, and so may be pre-computed off card.)
C2	$CB_{ICC} = CB_H$ & 'F0'	Create the PIV Card Application's control byte from client application's control byte, indicating that persistent binding has not been used in this transaction, even if CB_H indicates that the client application supports it. This may be done by setting CB_{ICC} to the value of CB_H and then setting the 4 least significant bits of CB_{ICC} to 0.
C3	Check that CB_{ICC} is 0x00	Check that client application is requesting that the GUID be returned in unencrypted form and that 3 session keys be generated. Return an error ('6A 80') if CB_{ICC} is not 0x00.
C4	Verify that Q_{eH} is a valid public key for the domain parameters of Q_{sICC}	Perform partial public-key validation of Q_{eH} [SP800-56A, Section 5.6.2.3.3],[15] where the domain parameters are those of Q_{sICC}. Also verify that P1 is '27' if the domain parameters of Q_{sICC} are those of Curve P-256 or that P1 is '2E' if the domain parameters of Q_{sICC} are those of Curve P-384. Return '6A 86' if P1 has the incorrect value. Return '6A 80' if public-key validation fails.
C5	Z = ECC_CDH (d_{sICC}, Q_{eH})	Compute the shared secret, Z, using the ECC CDH primitive [SP800-56A, Section 5.7.1.2].
C6	Generate nonce N_{ICC}	Create a random nonce, where the length is as specified in Table 14. The nonce should be created using an **approved** random bit generator where the security strength supported by the random bit generator is at least as great as the bit length of the nonce being generated [SP800-56A, Section 5.3].

[15] The PIV Card Application may perform full public-key validation instead [SP800-56A, Section 5.6.2.3.2].

Step #	Description	Comment
C7	$SK_{CFRM} \| SK_{MAC} \| SK_{ENC} \| SK_{RMAC}$ = KDF (Z, *len*, *Otherinfo*)	Compute the key confirmation key and the session keys. See Section 4.1.6.
C8	Zeroize Z	Destroy shared secret generated in Step C5.
C9	$AuthCryptogram_{ICC}$ = CMAC(SK_{CFRM}, "KC_1_V" ‖ ID_{sICC} ‖ ID_{sH} ‖ Q_{eH})	Compute the authentication cryptogram for key confirmation as described in Section 4.1.7.
C10	Zeroize SK_{CFRM}	Destroy the key confirmation key derived in Step C7.
C11	Return CB_{ICC} ‖ N_{ICC} ‖ $AuthCryptogram_{ICC}$ ‖ GUID ‖ C_{ICC}*	

4.1.3 Notations

Name	Comment	Format	Size (in bytes)
ICC	Integrated Circuit Card (PIV Card)	N/A	N/A
ID_{sICC}	Static, non-anonymous PIV Card identifier, which is the truncated hash of C_{ICC}*	Binary	8 bytes
GUID	Card UUID (see Section 3.4.1 of Part 1)	Binary	16 bytes
C_{ICC}	Secure messaging card verifiable certificate, which is authenticated by client application. See Section 4.1.5.	CVC	
C_{ICC}*	Transformation of the secure messaging card verifiable certificate, which is derived from C_{ICC} as follows: The Subject Identifier data element of C_{ICC} (T=0x5F20, L=16, V=GUID) is replaced with (T=0x5F20, L=0) and the CVC's tag is changed from 0x7F21 to 0x7F22. All other data elements, including the DigitalSignature object, and their order are identical to those in C_{ICC}.	CVC	
ID_{sH}	Client application identifier. This is a locally assigned identifier for the client application. If none is available, it could be set to all zeros.	Binary	8 bytes
N_{ICC}	PIV Card Application nonce. See Table 14 for the length.	Binary	16 or 24 bytes
SK_{CFRM}	Key confirmation key used to compute authentication cryptogram. See Table 14 for the length.		16 or 32 bytes
SK_{MAC}, SK_{RMAC}, SK_{ENC}	Secure messaging session keys. See Table 14 for encryption or MAC session key length.		16 or 32 bytes
T_8*(Data)*	Leftmost 8 bytes of *Data*.	Binary	8 bytes
T_{16}*(Data)*	Leftmost 16 bytes of *Data*.	Binary	16 bytes
KDF(Z, len, OtherInfo)	Key Derivation Function (KDF) specified in Section 4.1.6.	N/A	N/A
ECC_CDH	Elliptic curve cryptography cofactor Diffie-Hellman (ECC CDH) primitive, as specified in [SP800-56A, Section 5.7.1.2].	N/A	N/A
OtherInfo	Input parameters to the KDF. See Section 4.1.6.	N/A	N/A
len	The length (in bits) of the secret keying material to be generated using the KDF (*len* = 512 for cipher suite 2 and 1024 for cipher suite 7).	N/A	N/A

Name	Comment	Format	Size (in bytes)
CB_{ICC}	Protocol control byte returned by the PIV Card	Binary	1 byte
CB_{H}	Protocol control byte sent by client application (host)	Binary	1 byte

4.1.4 Cipher Suite

This document specifies two cipher suites (see Table 14) that may be used for key establishment and secure messaging, one that provides 128 bits of channel strength and one that provides 192 bits of channel strength. If the PIV Card Application supports the VCI and either the digital signature key ('9C'), the key management key ('9D'), or one of the retired key management keys ('82' – '95') is an ECC (Curve P-384) key, then PIV Card Application shall only support cipher suite CS7. Otherwise, the PIV Card Application may support either CS2 or CS7.

Table 14. Cipher Suite for PIV Secure Messaging

	128 bit channel strength	192 bit channel strength
Cipher Suite ID	CS2	CS7
Algorithm Identifier (P1)	'27'	'2E'
Key confirmation and session keys (SK_{CFRM}, SK_{MAC}, SK_{RMAC}, SK_{ENC})	AES 128	AES 256
C_{ICC} signature	ECDSA with SHA-256 using an ECDSA (Curve P-256) key	ECDSA with SHA-384 using an ECDSA (Curve P-384) key
C_{ICC} public key	ECDH (Curve P-256)	ECDH (Curve P-384)
KDF hash	SHA-256	SHA-384
Nonce (N_{ICC})	16 bytes	24 bytes

4.1.5 Card Verifiable Certificates

Table 15 specifies the format for the secure messaging CVC, C_{ICC}, and Table 16 specifies the format for the optional Intermediate CVC.

C_{ICC} is used to authenticate the PIV Card Application. The specific data object tags and specified order must be used for both CVCs to allow the CVC processing within authentication protocols. The specific data object tags for C_{ICC} and the optional Intermediate CVC are provided in Tables 14 and 15, respectively.

The signature of the secure messaging CVC (DigitalSignature object) is calculated over the concatenation of the TLV encoded Credential Profile Identifier, Issuer Identification Number, Subject Identifier, CardHolderPublicKey Data Object, and Role Identifier, i.e., { '5F29' '01' '80' } || { '42' '08' IIN } || { '5F20' '10' GUID } || { '7F49' L1 { { '06' L2 OID } { '86' L3 '04' X Y } } } { '5F4C' '01' '00' }. Before signing the CVC the signer shall perform full public-key validation [SP800-56A, Section 5.6.2.3.2] for the public key that will be placed in the Public Key object and shall verify that the PIV Card is in possession of the corresponding private key (see [SP800-56A, Section 5.6.2.2.3.2] and [SP800-57, Section 8.1.5.1.1.2] for discussions of methods to obtain assurance of private-key possession).

Table 15. Secure Messaging Card Verifiable Certificate Format

Tag	Tag	Tag	Length	Name	Value
0x7F21 or 0x7F22				Card Verifiable Certificate	Tag is 0x7F21 (for C_{ICC}) when Subject Identifier contains 16-byte GUID and is 0x7F22 (for C_{ICC}*) when length of Subject Identifier is 0.
	0x5F29		1	Credential Profile Identifier	0x80
	0x42		8	Issuer Identification Number	The leftmost 8 bytes of the subjectKeyIdentifier in the content signing certificate needed to verify the signature on C_{ICC}.[16]
	0x5F20		16	Subject Identifier	GUID (Card UUID) [In C_{ICC}*, the length of the Subject Identifier is 0.]
	0x7F49		Variable	CardHolderPublicKey Data Object	
		0x06	Variable	Algorithm OID	Possible values are: ▪ 0x2A8648CE3D030107 for ECDH (Curve P-256) or ▪ 0x2B81040022 for ECDH (Curve P-384)
		0x86	Variable	Public Key object	Coded as follows: 04 || X || Y, where X and Y are the coordinates of the point on the curve. See the “Value” column of Table 13.
	0x5F4C		1	Role Identifier	0x00 for card-application key CVC

	0x5F37		Variable	DigitalSignature object	DigitalSignature ::= SEQUENCE {
					signatureAlgorithm AlgorithmIdentifier,
					signatureValue BIT STRING
					}
					AlgorithmIdentifier ::= SEQUENCE {
					algorithm OBJECT IDENTIFIER,
					parameters ANY DEFINED BY algorithm OPTIONAL
					}
					algorithm is 1.2.840.10045.4.3.2 for ECDSA with SHA-256 (cipher suite 2) and 1.2.840.10045.4.3.3 for ECDSA with SHA-384 (cipher suite 7). For both algorithms, the parameters field is absent.
					signatureValue is the DER encoding of signature result ECDSA-Sig-Value defined below.
					ECDSA-Sig-Value ::= SEQUENCE {
					r INTEGER,
					s INTEGER
					}

[16] If the public key needed to verify the signature on the secure messaging CVC appears in an Intermediate CVC then the Issuer Identification Number shall be the value of the Subject Identifier in the Intermediate CVC.

Table 16. Intermediate Card Verifiable Certificate Format

Tag	Tag	Tag	Length	Name	Value
0x7F21				Card Verifiable Certificate	
	0x5F29		1	Credential Profile Identifier	0x80
	0x42		8	Issuer Identification Number	The leftmost 8 bytes of the subjectKeyIdentifier in the content signing certificate needed to verify the signature on the Intermediate CVC.
	0x5F20		8	Subject Identifier	The leftmost 8 bytes of the SHA-1 hash of the Public Key object.
	0x7F49		Variable	PublicKey Data Object	
		0x06	Variable	Algorithm OID	Possible values are: ▪ 0x2A8648CE3D030107 for ECDH (Curve P-256) or ▪ 0x2B81040022 for ECDH (Curve P-384)
		0x86	Variable	Public Key object	Coded as follows: 04 ǁ X ǁ Y, where X and Y are the coordinates of the point on the curve. See the "Value" column of Table 13.
	0x5F4C		1	Role Identifier	0x12 for card-application root CVC

	0x5F37		Variable	DigitalSignature object	DigitalSignature ::= SEQUENCE {
					signatureAlgorithm AlgorithmIdentifier,
					signatureValue BIT STRING
					}
					
					AlgorithmIdentifier ::= SEQUENCE {
					algorithm OBJECT IDENTIFIER,
					parameters ANY DEFINED BY algorithm OPTIONAL
					}
					
					algorithm is 1.2.840.113549.1.1.11 for RSA with SHA-256 and PKCS #1 v1.5 padding. The parameters field shall be NULL.

The signature of the Intermediate CVC (DigitalSignature object) is calculated over the concatenation of the TLV encoded Credential Profile Identifier, Issuer Identification Number, Subject Identifier, PublicKey Data Object, and Role Identifier, i.e., { '5F29' '01' '80' } || { '42' '08' IIN } || { '5F20' '08' SI } || { '7F49' L1 { { '06' L2 OID } { '86' L3 '04' X Y } } } { '5F4C' '01' '12' }. Before signing the CVC the signer shall perform full public-key validation [SP800-56A, Section 5.6.2.3.2] for the public key that will be placed in the Public Key object and shall verify that the subject is in possession of the corresponding private key (see [SP800-56A, Section 5.6.2.2.3.2] and [SP800-57, Section 8.1.5.1.1.2] for discussions of methods to obtain assurance of private-key possession).

4.1.6 Key Derivation

The session keys shall be derived in Steps C7 and H11 of the protocol using the key derivation function from [SP800-56A, Section 5.8.1], with the auxiliary function H being the hash function specified as the KDF hash in Table 14, the length of the keying material to be derived (*len*) being 512 bits for CS2 and 1024 bits for CS7, and *OtherInfo* being constructed using the concatenation format as show below:

Cipher Suite ID	*OtherInfo*
CS2	0x04 || 0x09 || 0x09 || 0x09 || 0x09 || 0x08 || ID_{sH} || 0x01 || CB_H || 0x10 || $T_{16}(Q_{eH})$ || 0x08 || ID_{sICC} || 0x10 || N_{ICC} || 0x01 || CB_{ICC}
CS7	0x04 || 0x0D || 0x0D || 0x0D || 0x0D || 0x08 || ID_{sH} || 0x01 || CB_H || 0x10 || $T_{16}(Q_{eH})$ || 0x08 || ID_{sICC} || 0x18 || N_{ICC} || 0x01 || CB_{ICC}

4.1.7 Key Confirmation

Key confirmation shall be performed in Steps C9 and H13 of the protocol in accordance with Sections 5.9.1.1 and 6.2.2.3 of [SP800-56A] by the generation of $AuthCryptogram_{ICC}$. $AuthCryptogram_{ICC}$ shall be computed as CMAC(*MacKey*, *MacLen*, $MacData_p$), where *MacKey* is SK_{CFRM}, *MacLen* is 128 bits, and $MacData_p$ is "KC_1_V" || ID_{sICC} || ID_{sH} || Q_{eH}. For Q_{eH}, the coordinates of the ephemeral public key are converted from field elements to byte strings as specified in [SP800-56A, Appendix C.2], Field-Element-to-Byte String Conversion, and concatenated (with *x* first) to form a single byte string. CMAC is cipher-based message authentication code from [SP800-38B], where the block cipher is AES.

4.1.8 Command Interface

The following command interface shall be used for the key establishment protocol.

Command Syntax

CLA	'00'
INS	'87'
P1	Algorithm reference ('27' or '2E'), as specified in the 0xAC tag of the application property template
P2	'03' (PIV Secure Messaging key).
L_c	Length of data field
Data Field	'81' L1 { CB_H || ID_{sH} || Q_{eH} } '82 00', where CB_H is 0x00, ID_{sH} is an 8-byte client application identifier as described in Section 4.1.3, and Q_{eH} is an ephemeral public key encoded as 04 || X || Y, as specified in the “Value” column of Table 13.
L_e	'00'

Response Syntax

Data Field	'82' LL { CB_{ICC} || N_{ICC} || $AuthCryptogram_{ICC}$ || GUID || C_{ICC}* }
SW1-SW2	Status word

SW1	SW2	Meaning
'61'	'xx'	Successful execution where SW2 encodes the number of response data bytes still available
'6A'	'80'	Incorrect parameter in command data field
'6A'	'86'	Incorrect parameter in P1 or P2
'90'	'00'	Successful execution

4.2 Secure Messaging

PIV secure messaging is used to protect the integrity and confidentiality of the PIV data being transmitted between the card and the relying system. PIV secure messaging shall be provided using symmetric session keys derived using the key establishment protocol defined Section 4.1.

Once session keys are established and the card is authenticated as specified in Section 4.1, subsequent communication with the card can be performed using secure messaging by setting bits b3 and b4 of the CLA byte of the command APDU to 1, resulting in a '0C' or '1C' CLA byte. If bits b3 and b4 of the CLA byte are set, then both the command and the response shall be encrypted and integrity protected as described in this section. If the PIV Card Application cannot encrypt and integrity protect the response (e.g., because it does not support secure messaging or no session keys have been established), the PIV Card Application shall return an error (see Section 4.2.7). In the case of command chaining, if bits b3 and b4 of the CLA are set in any command in the chain then they shall be set in every command in the chain.

When secure messaging is used, the data field of the card command (or response) is encrypted first and then a message authentication code (MAC) is applied to the entire command (or response). When command (or response) chaining is required, the encryption and MAC are applied to the entire message and the result is then fragmented into separate command (or response) data fields.

In order to ensure that message reordering or replay attacks can be detected, a 16-byte MAC chaining value (MCV) is used. For the first command, and for the first response, sent after successful completion of the key establishment protocol the MCV consists of 16 bytes of '00'. For each subsequent command the MCV is the 16-byte MAC value computed on the previous command, and for each subsequent response the MCV is the 16-byte MAC value computed on the previous response. The MCV is included as part of the message over which the MAC value for each command (or response) is computed.

The SK_{ENC} session key shall be used to encrypt the command data field and response data field as described in Section 4.2.2. The SK_{MAC} session key shall be used to add integrity to the command as described in Section 4.2.3. The SK_{RMAC} session key shall be used to add integrity to the response as described in Section 4.2.5.

Secure messaging specified in this section can be applied to the following commands:

+ GET DATA
+ VERIFY
+ CHANGE REFERENCE DATA
+ GENERAL AUTHENTICATE

4.2.1 Secure Messaging Data Objects

The command and response messages shall be BER-TLV encoded according to Table 17.

Table 17. Secure Messaging Data Objects

Tag	Description
'87'	Padding-content indicator byte followed by the encrypted data
'8E'	Cryptographic checksum (MAC)
'97'	L_e
'99'	Status word

4.2.2 Command and Response Data Confidentiality

Under secure messaging, the PIV data is encrypted using AES in Cipher Block Chaining (CBC) mode with the SK_{ENC} session key, where SK_{ENC} is a 128-bit key for CS2 and a 256-bit key for CS7 as per Table 14. The encryption and encoding process for command data and response data shall be the same. The encryption of the command data or response data and encoding in BER-TLV format is illustrated Figure 1. The encryption shall be computed over the entire message before applying fragmentation for data transportation.

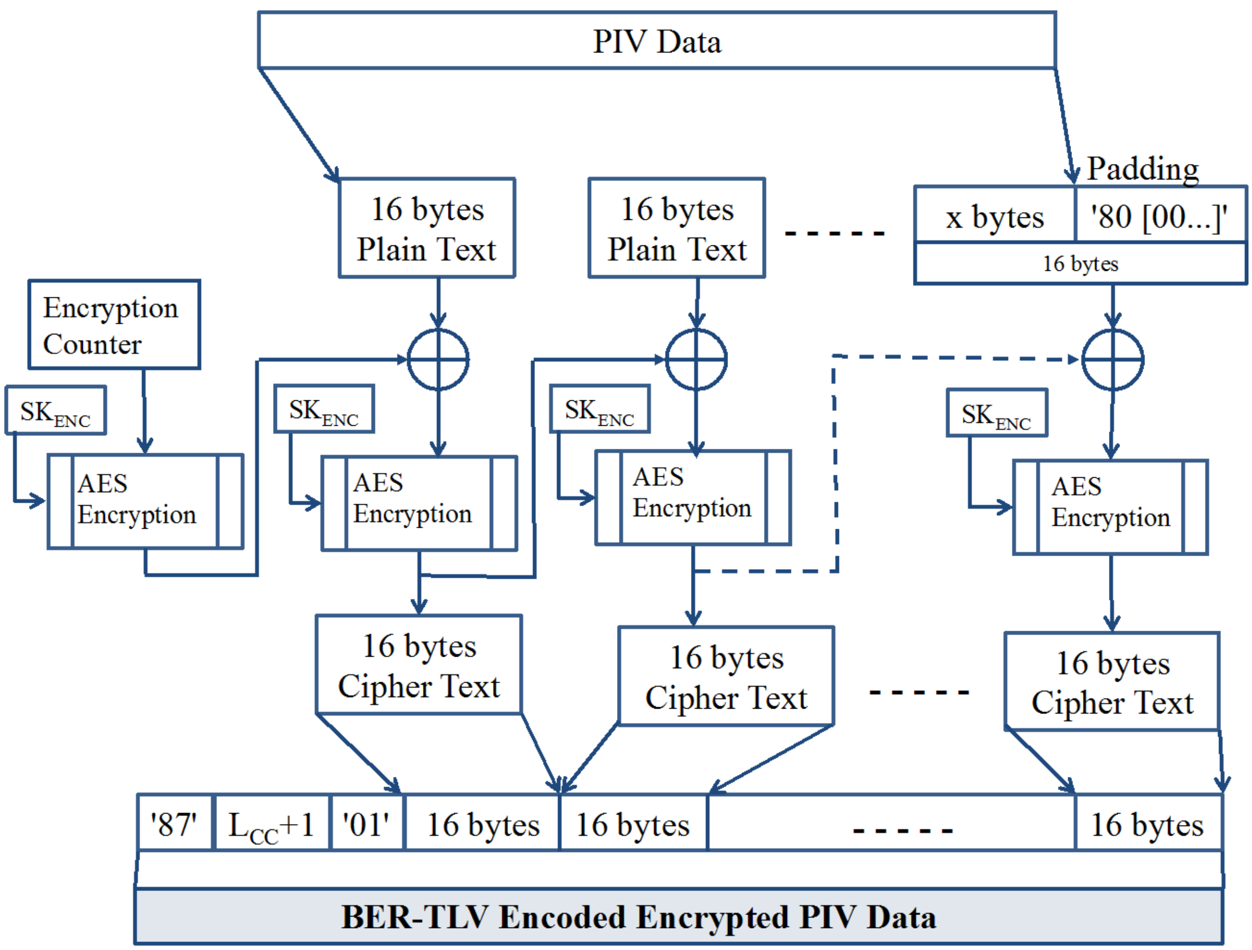

Figure 1. PIV Data Confidentiality

Initialization Vector (IV): The IV for the AES CBC encryption of command data shall be generated by applying the AES block cipher to a 16-byte encryption counter. The initial value of the encryption counter upon successful completion of the key establishment protocol shall be '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01'. The encryption counter shall be incremented by one after each creation of an IV to encrypt command data, and it shall be reset to its initial value after each successful completion of the key establishment protocol. The 16-byte IV shall be created by encrypting the encryption counter with SK_{ENC} using AES in the electronic codebook (ECB) mode of operation.

The IV for the AES CBC encryption of response data shall also be generated by encrypting an encryption counter with SK_{ENC} using AES in the ECB mode of operation. The encryption counter value used to generate the IV to encrypt the response data shall be the same as the encryption counter value used to generate the IV to encrypt the corresponding request data, with the exception that the most significant byte of the 16-byte counter shall be set to '80' (i.e., the IV used to encrypt the first response after successful completion of the key establishment protocol shall be generated by encrypting '80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01' with SK_{ENC}).

Padding: If the length of the command or response data is not a multiple of 16 bytes then padding shall be added to the last block of input data. The padding shall be '80' followed by the number of zeros needed to make up the length of 16 byte input block. If padding is used, the first byte of the value field of tag '87' shall be '01'; otherwise, the first byte shall be '02'.

As illustrated in Figure 1, the input and output of encryption is as follows:

- **Encryption input:**
  Plain Text
- **Encryption output:**
  BER-TLV encoded encrypted message, which consists of tag '87' followed by the length of the encoded encrypted message (L_{cc} + 1), the padding indicator byte ('01' or '02'), and then the encrypted data. L_{cc} is the length of the encrypted PIV data; it shall be a multiple of 16.

4.2.3 Command Integrity

The Command MAC (C-MAC) shall be generated by applying the cipher-based MAC (CMAC) [SP800-38B] to the header and data field of a command using the SK_{MAC} session key. In the case that fragmentation is required for data transmission, the command shall be constructed without fragmentation for the purposes of computing the MAC, and the CLA byte used in the computation of the MAC shall be '0C'.

The data to be MACed, $M_{C\text{-}MAC}$, shall be constructed by concatenating the following:

1. The 16-byte MAC chaining value (MCV). For the first command sent after successful completion of the key establishment protocol the MCV consists of 16 bytes of '00'. For each subsequent command the MCV is the 16-byte MAC value computed for the previous command.

2. A 16-btye encoded header. The encoded header shall consist of the CLA byte ('0C'), the INS byte, P1, and P2, followed by twelve bytes of padding, consisting of '80' followed eleven bytes of '00'. (The length of the data field, L_c, is not included in the data to be MACed.)

3. The data field, which is the BER-TLV encoded encrypted message.[17]

4. L_e encapsulated in BER-TLV format with tag '97', if the L_e field is included in the command.[18]

Let $T_{C\text{-}MAC}$ = CMAC(SK_{MAC}, $M_{C\text{-}MAC}$) as described in [SP800-38B]. The BER-TLV encoded C-MAC for the command shall be the 8 most significant bytes of $T_{C\text{-}MAC}$ encapsulated in BER-TLV format with tag '8E'. The entire 16-byte value $T_{C\text{-}MAC}$ will be the MCV for the next command.

Figure 2 below illustrates how the C-MAC is generated for each command.

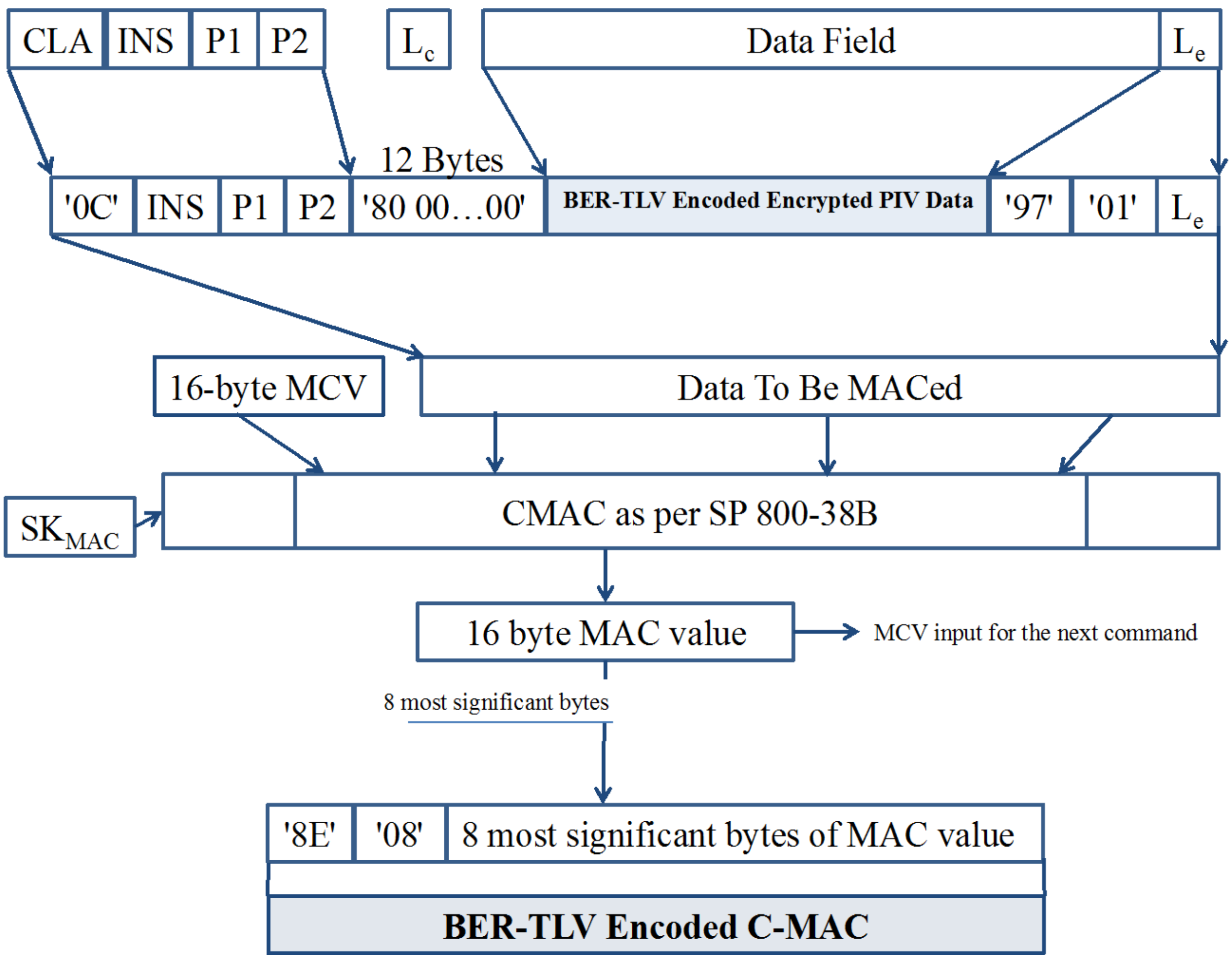

Figure 2. PIV Data Integrity of Command

4.2.4 Command with PIV Secure Messaging

For secure messaging, the secure messaging data field shall be constructed as the concatenation of the following: the BER-TLV encoded encrypted PIV data;[19] the 3-btye BER-TLV encoded L_e, as described in Section 4.2.3, if L_e would have been included in a message sent without secure messaging; the 10-byte

[17] The data field may be absent in the case of the VERIFY command.
[18] As noted in Sections 3.1.2 and 3.2.4, the value of L_e will always be '00', when it is present.
[19] The data field may be absent in the case of the VERIFY command.

BER-TLV encoded C-MAC of the command, as described in Section 4.2.3; and a new L_e field, which shall be one byte and have shall have a value of '00'.[20]

The APDU for secure messaging is shown in Figure 3 for the case in which command chaining is not required. The APDU consists of the CLA byte ('0C'), INS, P1, P2, the length of the secure messaging data field (L_c'), the secure messaging data field, and the new L_e field ('00').

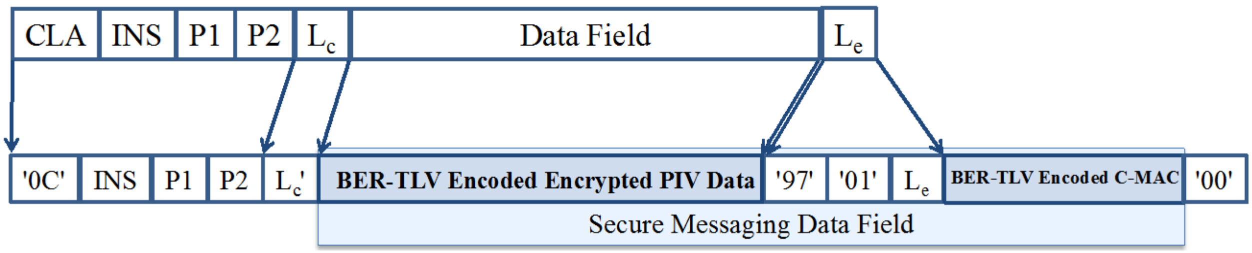

Figure 3. Single Command under Secure Messaging

If the secure messaging data field to be transported is larger than 255 bytes, command chaining will be needed. Figure 4 shows the APDUs for secure messaging for a case in which the length of the secure messaging data field is between 256 and 510 bytes, requiring the data to be fragmented across two APDUs. The APDUs are constructed in the same manner as when fragmentation is not required, except that the CLA byte for the first APDU is '1C', the first APDU contains the first 255 bytes of the secure messaging data field, and the second APDU contains the remaining bytes of the secure messaging data field and the new L_e field ('00'). The PIV Card Application provides a two-byte response of '90 00' for the first APDU. After receiving the second APDU the PIV Card Application reconstructs and processes the entire command.

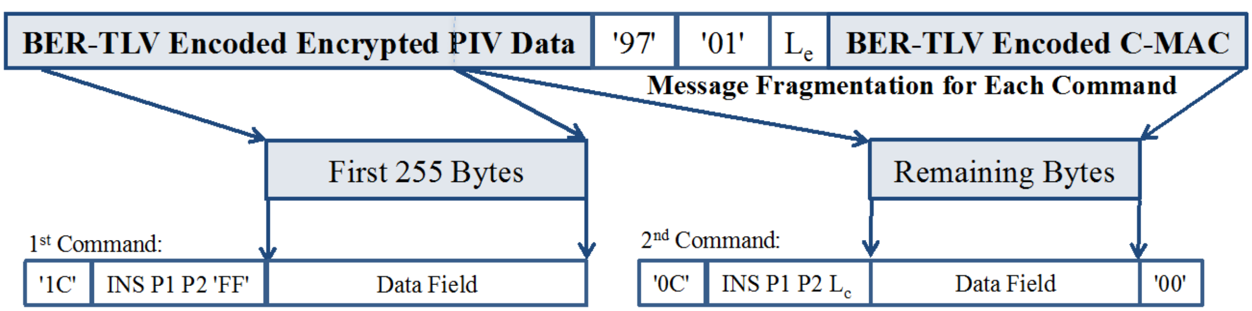

Figure 4. Chained Command under Secure Messaging

4.2.5 Response Integrity

The Response MAC (R-MAC) shall be generated by applying CMAC [SP800-38B] to the data field and status bytes of the response using the SK_{RMAC} session key. An R-MAC shall be generated for each response that corresponds to a command that was sent to the card using secure messaging.

The data to be MACed, $M_{R\text{-}MAC}$, shall be constructed by concatenating the following:

[20] Note that the new L_e field is always included in the command, even if L_e would have been absent if the command were sent without secure messaging, since a response is always expected, even if the expected response only consists of the BER-TLV encoded status words and response MAC (R-MAC).

1. The 16-byte MAC chaining value (MCV). For the first response sent after successful completion of the key establishment protocol the MCV consists of 16 bytes of '00'. For each subsequent response the MCV is the 16-byte MAC value computed for the previous response.

2. The data field (if present), which is the BER-TLV encoded encrypted message.

3. The status words, SW1 and SW2, encapsulated in BER-TLV format with tag '99'.

Let $T_{R\text{-}MAC} = \mathrm{CMAC}(SK_{RMAC}, M_{R\text{-}MAC})$ as described in [SP800-38B]. The BER-TLV encoded R-MAC for the response shall be the 8 most significant bytes of $T_{R\text{-}MAC}$ encapsulated in BER-TLV format with tag '8E'. The entire 16-byte value $T_{R\text{-}MAC}$ will be the MCV for the next response.

Figure 5 below illustrates how the R-MAC is generated for the response.

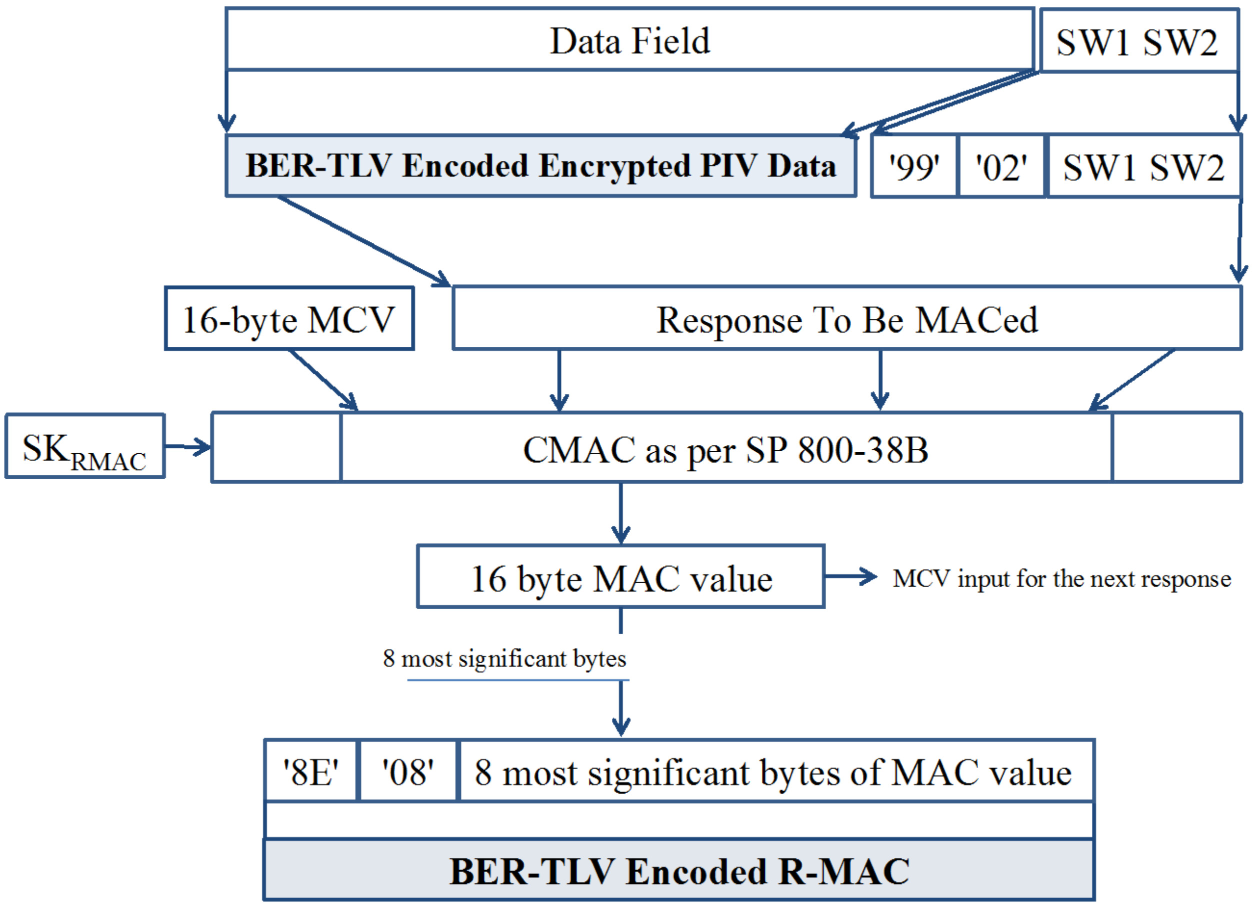

Figure 5. PIV Data Integrity of Response

4.2.6 Response with PIV Secure Messaging

For secure messaging, the secure messaging data field that is sent by the PIV Card Application shall be constructed as the concatenation of the following: the BER-TLV encoded encrypted message (when present); the 4-byte BER-TLV encoded the status words, as described in Section 4.2.5; and the 10-byte BER-TLV encoded R-MAC of the response, as described in Section 4.2.5.

Figure 6 illustrates a response under secure messaging for the case in which response chaining is not required. The APDU consists of the secure messaging data field and the 2-byte SW protocol ('90 00'), which indicates that the PIV Card Application successfully verified the C-MAC on the command and decrypted the data field in the command (if present). If the PIV Card Application was unable to verify the C-MAC on the command or decrypt the data field in the command, then it shall return a 2-byte error response, as described in Section 4.2.7.

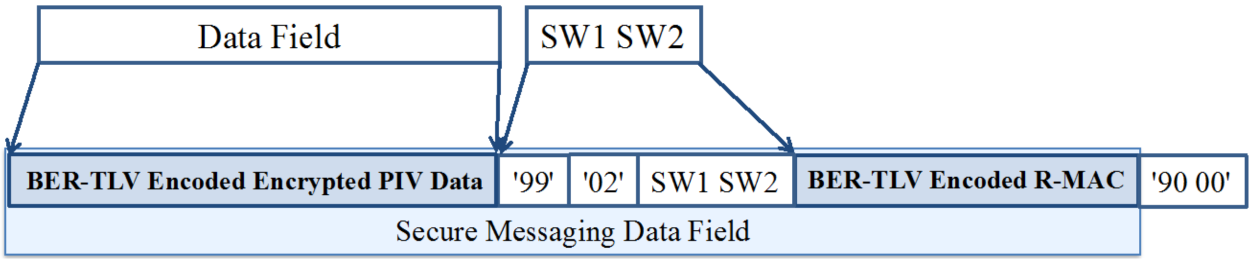

Figure 6. Single Response under Secure Messaging

If the secure messaging data field to be transported is larger than 256 bytes, response chaining[21] will be needed. Figure 7 shows the APDUs for secure messaging that are sent by the PIV Card Application for a case in which the length of the secure messaging data field is between 513 and 768 bytes, requiring the data to be fragmented across three APDUs. After the first response an APDU of '00 C0 00 00 00' would be sent to request the second response, and after the second response an APDU of '00 C0 00 00 xx' would be sent to request the third response.

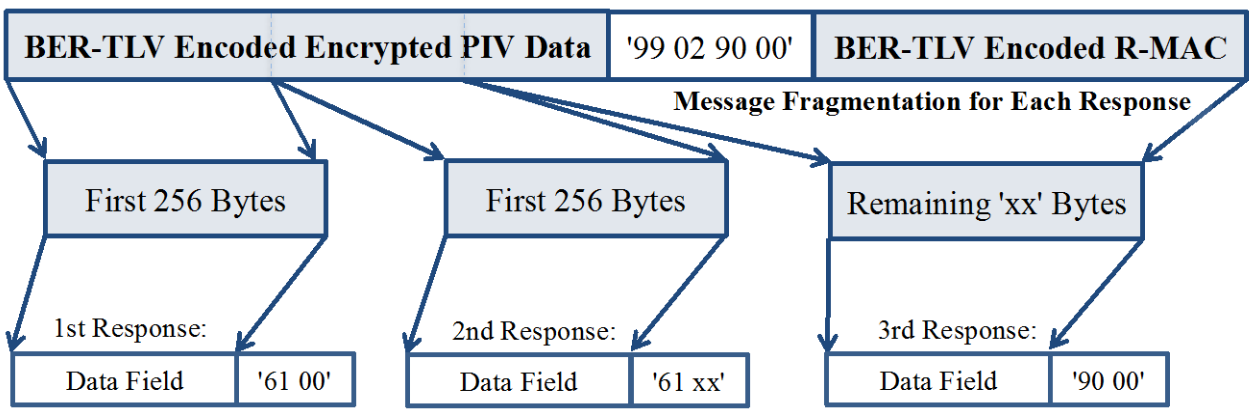

Figure 7. Chained Response under Secure Messaging

4.2.7 Error Handling

The SW protocol is the status byte of the overall secure messaging command and response processing. It indicates if the secure messaging was performed successfully. If the processing was successful, it shall be '90 00'; otherwise, it shall be as follows:

+ '68 82' – Secure messaging not supported

[21] The response chaining is accomplished by issuing several GET RESPONSE commands to the card.

+ '69 82' – Security status not satisfied[22]
+ '69 87' – Expected secure messaging data objects are missing
+ '69 88' – Secure messaging data objects are incorrect

If the command processing was unsuccessful, the card shall return one of the above errors without performing further secure messaging.

4.3 Session Key Destruction

The session keys established after successful execution of the key establishment protocol in Section 4.1 shall be zeroized in the following circumstances:

+ the card is reset;
+ an error occurs in secure messaging; or
+ new session keys are requested by the client application by sending a GENERAL AUTHENTICATE command to the card to perform the key establishment protocol using the PIV Secure Messaging key.

[22] Status word '69 82' is used when secure messaging is requested, but no session keys have been established.

Appendix A—Examples of the Use of the GENERAL AUTHENTICATE Command

A.1 Authentication of the PIV Card Application Administrator

The PIV Card Application Administrator is authenticated by the PIV Card Application using a challenge/response protocol. A challenge retrieved from the PIV Card Application is encrypted by the client application and returned to the PIV Card Application associated with key reference '9B', the key reference of the PIV Card Application Administration key. The PIV Card Application decrypts the response using this reference data and the algorithm associated with the key reference (for example, 3 Key Triple DES – ECB, algorithm identifier '00'). If this decrypted value matches the previously provided challenge, then the security status indicator of the PIV Card Application Administration key is set to TRUE within the PIV Card Application.

Table 18 shows the GENERAL AUTHENTICATE card commands sent to the PIV Card Application to realize this particular challenge/response protocol.

Table 18. Authentication of PIV Card Application Administrator

Command	Response	Comment
'00 87 00 9B 04 7C 02 81 00 00'		Client application requests a challenge from the PIV Card Application.
	'7C 0A 81 08 01 02 03 04 05 06 07 08 90 00'	Challenge ('01 02 03 04 05 06 07 08') returned to client application by the PIV Card Application.
'00 87 00 9B 0C 7C 0A 82 08 88 77 66 55 44 33 22 11'		Client application returns the encryption of the challenge ('88 77 66 55 44 33 22 11') referencing algorithm '00' and key reference '9B'. [SP800-78, Tables 6-1 and 6-2]
	'90 00'	PIV Card Application indicates successful authentication of PIV Card Application Administrator after decrypting '88 77 66 55 44 33 22 11' using the referenced algorithm and key and getting '01 02 03 04 05 06 07 08'.

A.2 Mutual Authentication of Client Application and Card Application

The PIV Card Application Administrator and the PIV Card Application authenticate each other using a challenge/response protocol. A witness retrieved from the PIV Card Application is decrypted by the client application and returned to the PIV Card Application associated with key reference '9B', the key reference of the PIV Card Application Administration key. The command including the decrypted witness also includes a challenge for the PIV Card Application. The PIV Card Application verifies that the decrypted witness matches the value that it encrypted to create the witness. If it does, then the security status indicator of the PIV Card Application Administration key is set to TRUE within the PIV Card Application, and the PIV Card Application encrypts the challenge that it received from the client application and returns the result. The witness and challenge are encrypted/decrypted using the same the key and algorithm. Table 19 shows the GENERAL AUTHENTICATE card commands sent to the PIV Card Application to realize mutual authentication using 3 Key Triple DES – ECB (algorithm identifier '00').

Table 19. Mutual Authentication of Client Application and PIV Card Application

Command	Response	Comment
'00 87 00 9B 04 7C 02 80 00 00'		Client application requests a witness from the PIV Card Application.
	'7C 0A 80 08 88 77 66 55 44 33 22 11 90 00'	PIV Card Application returns a witness that is created by generating 8 bytes of random data ('01 02 03 04 05 06 07 08') and encrypting it using the referenced key ('9B') and algorithm ('00'). [SP800-78, Tables 6-1 and 6-2]
'00 87 00 9B 18 7C 16 80 08 01 02 03 04 05 06 07 08 81 08 09 0A 0B 0C 0D 0E 0F 10 82 00 00'		Client application returns the decrypted witness ('01 02 03 04 05 06 07 08') referencing algorithm '00' and key reference '9B'. Client application requests encryption of challenge data ('09 0A 0B 0C 0D 0E 0F 10') from the card using the same key.
	'7C 0A 82 08 11 FF EE DD CC BB AA 99 90 00'	PIV Card Application authenticates the client application by verifying the decrypted witness. PIV Card Application indicates successful authentication of PIV Card Application Administrator and sends back the encrypted challenge ('11 FF EE DD CC BB AA 99'). Client application authenticates the PIV Card Application by decrypting the encrypted challenge and getting ('09 0A 0B 0C 0D 0E 0F 10').

A.3 Authentication of PIV Cardholder

The PIV cardholder is authenticated by first retrieving and validating either the X.509 Certificate for PIV Authentication or the X.509 Certificate for Card Authentication. Assuming the certificate is valid, the client application requests the PIV Card Application to sign a challenge using the private key associated with this certificate (i.e., key reference '9A' or '9E') and the appropriate algorithm (e.g., algorithm identifier '07'), which can be determined from the certificate as described in Part 1, Appendix C.1. The response from the card is verified using the public key in the certificate. If the signature verifies, then the PIV cardholder is authenticated.

Table 20 shows the GENERAL AUTHENTICATE card commands sent to the PIV Card Application to realize the cardholder authentication when the X.509 Certificate for PIV Authentication includes a 2048-bit RSA public key. It is assumed that the cardholder PIN or OCC data has been successfully verified prior to sending the GENERAL AUTHENTICATE command.

Table 20. Validation of the PIV Card Application Using GENERAL AUTHENTICATE

Command	Response	Comment
'10 87 07 9A FF 7C 82 01 06 82 00 81 82 01 00 00 01 FF FF FF FF ... FF FF FF FF FF 00 9D F4 6E 09 E7 D6 19 18 53 1E 6E 1C 66 87 C4 3E CF FF 7D 53 47 BD 2E 93 19' ("..." represents 208 bytes of challenge data)		Client application sends a challenge to the PIV Card Application indicating the reference data associated with key reference '9A' is to be used with algorithm '07'. [SP800-78, Tables 6-1 and 6-2] The challenge data, which in this example is encoded as specified for TLS version 1.1 client authentication, is '00 01 FF ... 18 BC A7'. Bit 5 of CLA byte is set to one indicating command chaining is needed. L_e is absent indicating no data is expected.
	'90 00'	PIV Card Application indicates it received the command successfully.
'00 87 07 9A 0B 94 53 76 FE A7 91 72 14 18 BC A7 00'		Client application sends remaining data with the second and last command of the chain. L_e is '00' to indicate that the expected length of the response data field is 256 bytes.
	'7C 82 01 04 82 82 01 00 29 69 44 3B 49 AC 5B 70 63 51 A1 5B B5 ... AD F7 0B 7D A6 4C 6C AA 62 40 C5 FA A8 7E A2 2B DC 92 18 56 8B CE F4 69 14 D9 83 61 08' ("..." represents 208 bytes of response data)	PIV Card Application returns the result of signing the challenge using the indicated key reference data and algorithm ('29 69 44 3B 49 AC…'). The last two bytes '61 08' indicate 8 more bytes are available to read from the card.
'00 C0 00 00 08'		The GET RESPONSE command is used to request remaining 8 bytes.
	'30 1B 11 06 AE E2 F1 2E 90 00'	PIV Card Application sends the remaining 8 bytes.

A.4 Signature Generation with the Digital Signature Key

The GENERAL AUTHENTICATE command can be used to generate signatures. The pre-signature hash and padding (if applicable) is computed off card. The PIV Card Application receives the hashed value of the original message, applies the private signature key (key reference '9C'), and returns the resulting signature to the client application.

Listed below are the card commands sent to the PIV Card Application to generate a signature. It is assumed that the cardholder PIN or OCC data has been successfully verified prior to sending the GENERAL AUTHENTICATE command.

A.4.1 RSA

This example illustrates signature generation using RSA 2048 (i.e., algorithm identifier '07'). Command chaining is used in the first command since the padded hash value sent to the card for signature generation is bigger than the length of the data field.

Command 1: (GENERAL AUTHENTICATE – first chain):

CLA	'10' indicating command chaining
INS	'87'
P1	'07'
P2	'9C'
L_c	Length of data field
Data Field	'7C' – L1 { '82' '00' '81' L2 {first part of the PKCS #1 v1.5 or PSS padded message hash value }}
L_e	Absent (no response expected)

Response 1:

Data Field	Absent
SW1-SW2	'90 00' (Status word)

Command 2: (GENERAL AUTHENTICATE – last chain):

CLA	'00' indicates last command of the chain
INS	'87'
P1	'07'
P2	'9C'
L_c	Length of data field
Data Field	{second and last part of the PKCS #1 v1.5 or PSS padded message hash value}
L_e	'00'

Response 2:

Data Field	'7C' – L1 {'82' L2 {first part of signature} }
SW1-SW2	'61 xx' where xx indicates the number of bytes remaining to send by the PIV Card Application

Command 3: (GET RESPONSE APDU):

CLA	'00'
INS	'C0'
P1	'00'
P2	'00'
L_e	xx Length of remaining response as indicated by previous SW1-SW2

Response 3:

Data Field	{second and last part of signature}
SW1-SW2	'90 00' (Status word)

A.4.2 ECDSA

The following example illustrates signature generation with ECDSA using ECC: Curve P-256 (i.e., algorithm identifier '11'). Command chaining is not used in this example, as the hash value fits into the data field of the command. Padding does not apply to ECDSA.

Command – GENERAL AUTHENTICATE

CLA	'00'
INS	'87'
P1	'11'
P2	'9C'
L_c	Length of data field
Data Field	'7C' – L1 { '82' '00' '81' L2 {hash value of message}}
L_e	'00'

Response:

Data Field	'7C' – L1 {'82' L2 (r,s)} where • (r,s) is DER encoded with the following ASN.1 structure: Ecdsa-Sig-Value ::= SEQUENCE { r INTEGER, s INTEGER } • L1 is the length of tag '82' TLV structure • L2 is the length of the DER encoded Ecdsa-Sig-Value structure
SW1-SW2	'90 00' (Status word)

A.5 Key Establishment Schemes with the PIV Key Management Key

FIPS 201 specifies a public key pair and associated X.509 Certificate for Key Management. The key management key (KMK) is further defined in SP 800-78, which defines two distinct key establishment schemes for the KMK:

1) RSA key transport and
2) Elliptic Curve Diffie-Hellman (ECDH) key agreement.

The use of the KMK for RSA key transport and ECDH key agreement is discussed in Appendices A.5.1 and A.5.2, respectively.

A.5.1 RSA Key Transport

In general, RSA transport keys are used to establish symmetric keys, where a sender encrypts a symmetric key with the receiver's public key and sends the encrypted key to the receiver. The receiver decrypts the encrypted key with the corresponding private key. The decrypted symmetric key subsequently is used by both parties to protect further communication between them. Many types of security protocols employ the RSA key transport technique. S/MIME for secure email is one of the many protocols employing RSA transport keys to distribute symmetric keys between entities.

A.5.1.1 RSA Key Transport with the PIV KMK

As specified in SP 800-78, the on-card private KMK can be an RSA transport key that complies with [PKCS1]. In the scenario described above, a sender encrypts a symmetric key with the KMK's public RSA transport key. The role of the on-card KMK private RSA transport key is to decrypt the sender's symmetric key on behalf of the cardholder and provide it to the client application cryptographic module.

A.5.1.1.1 The GENERAL AUTHENTICATE Command

Listed below are the card commands sent to the PIV Card to decrypt the symmetric key. It is assumed that the cardholder's PIN or OCC data has been successfully verified prior to sending the GENERAL AUTHENTICATE command to the card.

Command 1 – GENERAL AUTHENTICATE (first chain)

CLA	'10' indicates command chaining
INS	'87'
P1	'07'
P2	'9D'
L_c	Length of data field
Data Field	'7C' – L1 {'82' '00' '81' L2 {first part of C}} where C is the ciphertext to be decrypted, as defined in [PKCS1, Sections 7.1.2 and 7.2.2]
L_e	Absent (no response expected)

Response 1:

Data Field	Absent
SW1-SW2	'90 00' (Status word)

Command 2 – GENERAL AUTHENTICATE (last chain)

CLA	'00' indicates last command of the chain
INS	'87'
P1	'07'
P2	'9D'
L_c	Length of data field
Data Field	{second and last part of ciphertext to be decrypted C }}
L_e	'00'

Response 2:

Data Field	'7C' – L1 {'82' L2 {first part of encoded message EM}} where EM is as defined in [PKCS1, Sections 7.1.2 and 7.2.2]
SW1-SW2	'61 xx' where x indicates the number of bytes remaining to send

Command 3: GET RESPONSE APDU:

CLA	'00'
INS	'C0'
P1	'00'
P2	'00'
L_e	xx Length of remaining response as indicated by previous SW1-SW2

Response 3:

Data Field	{second and last part of encoded message EM}
SW1-SW2	'90 00' (Status word)

A.5.2 Elliptic Curve Cryptography Diffie-Hellman

An ECDH key agreement scheme does not send an encrypted symmetric key to the participating entities. Instead, the two entities involved in the key agreement scheme compute a shared secret by combining their ECC private key(s) with the other party's public key(s). The resulting shared secret (Z) serves as an input to a key derivation function (KDF), which each entity independently invokes to derive a common secret key. The secret key may be used as a session key or may be used to encrypt a session key.

A.5.2.1 ECDH with the PIV KMK

The PIV Card supports ECDH key agreement by performing the elliptic curve cryptography cofactor Diffie-Hellman (ECC CDH) primitive [SP800-56A, Section 5.7.1.2] using its ECC KMK private key and an ECC public key that is provided as input to the GENERAL AUTHENTICATE command. All other procedures required to complete the key agreement are performed by the cardholder's client application and its associated cryptographic module.

A.5.2.1.1 The GENERAL AUTHENTICATE Command

The sequence of commands to perform the ECC CDH primitive from [SP800-56A, Section 5.7.1.2] with the private ECC KMK is illustrated below for ECC: Curve P-256:

Command – GENERAL AUTHENTICATE

CLA	'00'
INS	'87'
P1	'11'
P2	'9D'
L_c	Length of data field
Data Field	'7C' – L1 {'82' '00' '85' L2 { '04' || X || Y}} , where • '04' || X || Y is the other party's public key, a point on Curve P-256, encoded without the use of point compression as described in [SECG, Section 2.3.3]. • The length of each coordinate (X and Y) is 32 bytes and • The value of L2 is 65 bytes
L_e	'00'

Response:

Data Field	'7C' – L1 {'82' L2 {shared secret Z}} where • Z is the X coordinate of point P as defined in [SP800-56A, Section 5.7.1.2] • L2 is 32 bytes
SW1-SW2	'90 00' (Status word)

A.5.2.2 PIV KMK Specific ECDH Key Agreement Schemes

SP 800-56A describes five different ECDH key agreement schemes that a client application cryptographic module may implement. These schemes differ in 1) the number of keys (1 or 2) and 2) the type of keys (ephemeral or static) used by each party. Since the PIV Card only computes the ECC CDH primitive using its static private key, the client application cryptographic module only employs the PIV Card in implementing an ECDH key agreement scheme when the scheme involves the use of the cardholder's static key pair. The ECDH key agreement schemes that involve the use of at least one party's static key pair, and thus may involve the use of the PIV Card are:

+ C(2e, 2s) – Each party has a static key pair and generates an ephemeral key pair [SP800-56A, Section 6.1.1]

  In this scheme, the information sent between the client application and the PIV Card is the same when acting as the initiator or the responder; the other party's static public key is sent to the PIV Card, and a static shared secret is returned by the PIV Card in plaintext. Note that an ephemeral key pair is generated by the client application, and the private key of that key pair is combined with the other party's ephemeral public key to produce an ephemeral shared secret.

+ C(1e, 2s) – The initiator has a static key pair and generates an ephemeral key pair, while the responder has a static key pair [SP800-56A, Section 6.2.1]

When the cardholder is acting as the initiator, the other party's static public key is sent to the PIV Card, and a static shared secret is returned in plaintext by the PIV Card. Note that in this case, an ephemeral key pair is generated by the client application's cryptographic module, and the corresponding ephemeral private key is combined with the other party's static public key to produce a second shared secret.

When the cardholder is acting as the responder, two public keys are sent by the client application to the PIV Card (the other party's static and ephemeral public keys), and two shared secrets are returned in plaintext (the static shared secret and the ephemeral shared secret). Note that two GENERAL AUTHENTICATE commands are required to provide the two shared secrets to the client application's cryptographic module.

+ C(1e, 1s) – The initiator generates only an ephemeral key pair, while the responder has only a static key pair [SP800-56A, Section 6.2.2]

  In this scheme, the PIV Card is only employed by the client application if the cardholder is acting as the responder. In this case, the other party's ephemeral public key is sent to the PIV Card, and the shared secret is returned by the PIV Card in plaintext.

+ C(0e, 2s) – Both the initiator and responder use only static key pairs [SP800-56A, Section 6.3]

  In the C(0e, 2s) scheme, the information sent between the client application's cryptographic module and the PIV Card is the same when acting as the initiator or the responder; the other party's static public key is sent to the PIV Card, and the static shared secret is returned in plaintext. Note that for this scheme, the client application's cryptographic module also generates a nonce when acting as the initiator of the scheme.

The C(2e, 0s) scheme does not involve the use of static keys and so the PIV Card would not be involved in the implementation of this scheme.

A.6 Authentication of the PIV Cardholder Over the Virtual Contact Interface

If the PIV Card supports secure messaging and the pairing code, then all non-card-management operations of the PIV Card Application may be performed over the contactless interface. In order to perform an operation that would otherwise be restricted to the contact interface, the key establishment protocol in Section 4.1 needs to be performed to establish session keys for secure messaging, and then the pairing code needs to be submitted over secure messaging in order to establish a virtual contact interface.

This appendix shows an example of the establishment of a VCI and its use to perform cardholder authentication using the PIV Authentication key. First, the GENERAL AUTHENTICATE command is used to perform the key establishment protocol, and then the VERIFY command is used to submit the pairing code and establish the VCI. At this point the GET DATA command is used to read the X.509 Certificate for PIV Authentication. Then the GENERAL AUTHENTICATE command is used to perform a challenge/response with the PIV Authentication key after the PIN is submitted using the VERIFY command.

Command	Response	Comment
00 87 27 03 4E 81 4A 00 00 00 00 00 00 00 00 00 04 X Y 82 00 00		The GENERAL AUTHENTICATE command is used to perform the key establishment protocol, as specified in Section 4.1.8, where cipher suite CS2 is being used, ID_{sH} is all zeros, and X and Y are the coordinates of Q_{eH}. X and Y are 32 bytes each.
	82 LL 00 N_{ICC} $AuthCryptogram_{ICC}$ GUID C_{ICC}*	The response for the key establishment protocol, as specified in Section 4.1.8, where N_{ICC}, $AuthCryptogram_{ICC}$, and GUID are 16 bytes each, and C_{ICC}* is as specified in Sections 4.1.3 and 4.1.5.
After the client application verifies C_{ICC} and the authentication cryptogram and validates the certificate(s) needed to verify the signature on C_{ICC}, the PIV Card has been authenticated and session keys for secure messaging have been established (SK_{ENC}, SK_{MAC}, and SK_{RMAC}).		
The VERIFY command is used to submit the pairing code (“65135275”) to the PIV Card Application. For the command, ENC_{C1} is the result of encrypting '36 35 31 33 35 32 37 35 80 00 00 00 00 00 00 00' using an IV of AES(SK_{ENC}, '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01') and $T_{C\text{-}MAC,1}$ = CMAC(SK_{MAC}, '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0C 20 00 98 80 00 00 00 00 00 00 00 00 00 00 00 87 11 01' || ENC_{C1}). For the response, $T_{R\text{-}MAC,1}$ = CMAC(SK_{RMAC}, '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 99 02 90 00').		
0C 20 00 98 1D 87 11 01 ENC_{C1} 8E 08 T_8($T_{C\text{-}MAC,1}$) 00		The VERIFY command is used over secure messaging to submit the pairing code to the card.
	99 02 90 00 8E 08 T_8($T_{R\text{-}MAC,1}$) 90 00	The card responds that the command has been successfully executed, and that the VCI has been established.
Once the VCI has been established, the GET DATA command may be used to retrieve the X.509 Certificate for PIV Authentication. For the command, ENC_{C2} is the result of encrypting '5C 03 5F C1 05 80 00 00 00 00 00 00 00 00 00 00' using an IV of AES(SK_{ENC}, '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02'), and $T_{C\text{-}MAC,2}$ is computed using $T_{C\text{-}MAC,1}$ as the MCV. For the response, ENC_{R2} is the result of encrypting the X.509 Certificate for PIV Authentication data object encapsulated in BER-TLV format with tag '53' using an IV of AES(SK_{ENC}, '80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02'), and $T_{R\text{-}MAC,2}$ is computed using $T_{R\text{-}MAC,1}$ as the MCV.		
0C CB 3F FF 20 87 11 01 ENC_{C2} 97 01 00 8E 08 T_8($T_{C\text{-}MAC,2}$) 00		The GET DATA command is used to request the X.509 Certificate for PIV Authentication. The command is submitted over VCI.

Command	Response	Comment
	87 82 05 91 01 <bytes 1 – 251 of ENC_{R2}> 61 00	The response includes the tag, length, and padding indicator bytes of the BER-TLV encoded encrypted response data along with the first 251 bytes of the encrypted response, and an indicator that at least 256 bytes of additional data is available. The padding indicator is '01' to indicate that padding was required.
00 C0 00 00 00		Request the next 256 bytes of the response.
	<bytes 252 – 507 of ENC_{R2}> 61 00	Return the next 256 bytes of the response.
…	…	
00 C0 00 00 A3		Request the final 163 bytes of the response.
	<bytes 1276 – 1424 of ENC_{R2}> 99 02 90 00 8E 08 $T_8(T_{R-MAC,2})$ 90 00	Return the final 163 bytes of the response, including the BER-TLV encoded status words for the command and the BER-TLV encoded R-MAC.
At this point the VERIFY command could be used to submit the PIV Card Application PIN to the PIV Card Application. However, in this example, for illustrative purposes only, a VERIFY command is sent to the card without a data field in order to retrieve the current value of the retry counter associated with the PIV Card Application PIV. For the command,		
0C 20 00 80 0A 8E 08 $T_8(T_{C-MAC,3})$ 00		The VERIFY command is used to retrieve the number of further retries allowed for the PIV Card Application PIN.
	99 02 63 C3 8E 08 $T_8(T_{R-MAC,3})$ 90 00	The PIV Card Application indicates that 3 further retries are allowed ('63 C3').
The VERIFY command is used to submit the PIV Card Application PIN to the PIV Card Application. Other than the key reference and the PIN value, the command and response are the same as when using the VERIFY command to submit the pairing code. For the command, ENC_{C3} is the result of encrypting the PIN value along with the padding bytes using an IV of AES(SK_{ENC}, '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03'), and $T_{C-MAC,4}$ is computed using $T_{C-MAC,3}$ as the MCV. [Note that the encryption counter used to generate the IV was not incremented as of result of the previous VERIFY command since no encryption was performed for that command.] For the response, $T_{R-MAC,4}$ is computed using $T_{R-MAC,3}$ as the MCV.		
0C 20 00 80 1D 87 11 01 ENC_{C3} 8E 08 $T_8(T_{C-MAC,4})$ 00		The VERIFY command is used to submit the PIV Card Application PIN to the card.
	99 02 90 00 8E 08 $T_8(T_{R-MAC,4})$ 90 00	The card responds that the command has been successfully executed.

<table>
<tr><th>Command</th><th>Response</th><th>Comment</th></tr>
<tr><td colspan="3">Now that a virtual contact interface has been established and the PIV Card Application PIN has been verified, privileged operations may be performed over the contactless interface. So, the GENERAL AUTHENTICATE command is used to perform a challenge/response with the PIV Authentication key.
For the command, ENC_{C5} is the result of encrypting the challenge along with the padding bytes using an IV of AES(SK_{ENC}, '00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 04'), and $T_{C\text{-}MAC,5}$ is computed using $T_{C\text{-}MAC,4}$ as the MCV. The challenge to be encrypted is '7C 82 01 06 82 00 81 82 01 00 00 01 FF FF … BC A7' from the example in Table 20.
For the response ENC_{R5} is the result of encrypting the response using an IV of AES(SK_{ENC}, '80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 04'), and $T_{R\text{-}MAC,5}$ is computed using $T_{R\text{-}MAC,4}$ as the MCV. The response to be encrypted is '7C 82 01 04 82 82 01 00 29 69 44 3B … E2 F1 2E' from the example in Table 20.</td></tr>
<tr><td>1C 87 07 9A FF 87 82 01 11 01 <bytes 1 – 250 of ENC_{C5}></td><td></td><td>The GENERAL AUTHENTICATE command is used to send a challenge to the PIV Card. This command includes the first part of the challenge.</td></tr>
<tr><td></td><td>90 00</td><td>PIV Card Application indicates that it received the first part of the command successfully.</td></tr>
<tr><td>0C 87 07 9A 23 <bytes 251 – 272 of ENC_{C5}> 97 01 00 8E 08 $T_8(T_{C\text{-}MAC,5})$ 00</td><td></td><td>The remaining challenge data is sent, including the BER-TLV encoded L_e and the BER-TLV encoded C-MAC.</td></tr>
<tr><td></td><td>87 82 01 17 02 <bytes 1 – 251 of ENC_{R5}> 61 1B</td><td>PIV Card Application sends first part of the result of signing the challenge. The padding indicator is '02' to indicate that no padding was required.</td></tr>
<tr><td>00 C0 00 00 1B</td><td></td><td>The remaining portion of response is requested.</td></tr>
<tr><td></td><td><bytes 252 – 264 of ENC_{R5}> 99 02 90 00 8E 08 $T_8(T_{R\text{-}MAC,5})$ 90 00</td><td>PIV Card Application sends final portion of the result of signing the challenge, along with the BER-TLV encoded status words and R-MAC.</td></tr>
</table>

Appendix B—Terms, Acronyms, and Notation

B.1 Terms

Application Identifier	A globally unique identifier of a card application as defined in ISO/IEC 7816-4.
Algorithm Identifier	A PIV algorithm identifier is a one-byte identifier that specifies a cryptographic algorithm and key size. For symmetric cryptographic operations, the algorithm identifier also specifies a mode of operation (i.e., ECB).
Authenticable Entity	An entity that can successfully participate in an authentication protocol with a card application.
BER-TLV Data Object	A data object coded according to ISO/IEC 8825-2.
Card	An integrated circuit card.
Card Application	A set of data objects and card commands that can be selected using an application identifier.
Card Management Operation	Any operation involving the PIV Card Application Administrator.
Card Verifiable Certificate	A certificate stored on the card that includes a public key, the signature of a certification authority, and further information needed to verify the certificate.
Data Object	An item of information seen at the card command interface for which is specified a name, a description of logical content, a format, and a coding.
Key Reference	A PIV key reference is a one-byte identifier that specifies a cryptographic key according to its PIV Key Type. The identifier is part of cryptographic material used in a cryptographic protocol such as an authentication or a signing protocol.
MAC Chaining Value	MAC Chaining Value is a 16-byte value that is input to the CMAC function. It is used to detect communication errors in duplicate or missing commands.
Object Identifier	A globally unique identifier of a data object as defined in ISO/IEC 8824-2.
Reference Data	Cryptographic material used in the performance of a cryptographic protocol such as an authentication or a signing protocol. The reference data length is the maximum length of a password or PIN. For algorithms, the reference data length is the length of a key.
Status Word	Two bytes returned by an integrated circuit card after processing any command that signify the success of or errors encountered during said processing.
Template	A (constructed) BER-TLV data object whose value field contains specific BER-TLV data objects.

B.2 Acronyms

AES	Advanced Encryption Standard
AID	Application Identifier
APDU	Application Protocol Data Unit
API	Application Programming Interface
APT	Application Property Template
ASCII	American Standard Code for Information Interchange
ASN.1	Abstract Syntax Notation One
BER	Basic Encoding Rules
BIT	Biometric Information Template
CLA	Class (first) byte of a card command
CMAC	Cipher-based Message Authentication Code
C-MAC	Command Message Authentication Code
CVC	Card Verifiable Certificate
DER	Distinguished Encoding Rules
DES	Data Encryption Standard
ECB	Electronic Codebook
ECC	Elliptic Curve Cryptography
ECDSA	Elliptic Curve Digital Signature Algorithm
ECDH	Elliptic Curve Diffie-Hellman
EC CDH	Elliptic Curve Cryptography Cofactor Diffie-Hellman
FIPS	Federal Information Processing Standards
FISMA	Federal Information Security Management Act
HSPD	Homeland Security Presidential Directive
ICC	Integrated Circuit Card
IEC	International Electrotechnical Commission
IETF	Internet Engineering Task Force
INS	Instruction (second) byte of a card command
INCITS	InterNational Committee for Information Technology Standards
ISO	International Organization for Standardization
ITL	Information Technology Laboratory
KDF	Key Derivation Function
LSB	Least Significant Bit
MAC	Message Authentication Code
MSB	Most Significant Bit
MCV	MAC Chaining Value
NIST	National Institute of Standards and Technology
OCC	On-Card Biometric Comparison

OID	Object Identifier
OMB	Office of Management and Budget
OPACITY	Open Protocol for Access Control, Identification, and Ticketing with privacY
P1	First parameter of a card command
P2	Second parameter of a card command
PKCS	Public-Key Cryptography Standards
PIN	Personal Identification Number
PIV	Personal Identity Verification
PIX	Proprietary Identifier extension
PUK	PIN Unblocking Key
RFU	Reserved for Future Use
RID	Registered application provider Identifier
R-MAC	Response Message Authentication Code
RSA	Rivest, Shamir, Adleman
SM	Secure Messaging
S/MIME	Secure/Multipurpose Internet Mail Extensions
SP	Special Publication
SW1	First byte of a two-byte status word
SW2	Second byte of a two-byte status word
TLS	Transport Layer Security
TLV	Tag-Length-Value
VCI	Virtual Contact Interface

B.3 Notation

The sixteen hexadecimal digits shall be denoted using the alphanumeric characters 0, 1, 2, …, 9, A, B, C, D, E, and F. A byte consists of two hexadecimal digits, for example, '2D'. The two hexadecimal digits are represented in quotations '2D' or as 0x2D. A sequence of bytes may be enclosed in single quotation marks, for example 'A0 00 00 01 16', rather than given as a sequence of individual bytes, 'A0' '00' '00' '01' '16'.

A byte can also be represented by bits b8 to b1, where b8 is the most significant bit (MSB) and b1 is the least significant bit (LSB) of the byte. In textual or graphic representations, the leftmost bit is the MSB. Thus, for example, the most significant bit, b8, of '80' is 1 and the least significant bit, b1, is 0.

All bytes specified as RFU shall be set to '00' and all bits specified as RFU use shall be set to 0.

All lengths shall be measured in number of bytes unless otherwise noted.

The expression X & Y is a bitwise AND operation between bytes X and Y.

The symbol || means concatenation of byte strings. For example, if X is '00 01 02' and Y is '03 04 05', then X || Y is '00 01 02 03 04 05'.

Data objects in templates are described as being mandatory (M), optional (O), or conditional (C). 'Mandatory' means the data object shall appear in the template. 'Optional' means the data object may appear in the template. In the case of 'Conditional' data objects, the conditions under which they are required are provided.

In other tables the M/O/C column identifies properties of the PIV Card Application that shall be present (M), may be present (O), or are conditionally required to be present (C).

BER-TLV data object tags are represented as byte sequences as described above. Thus, for example, 0x4F is the interindustry data object tag for an application identifier and 0x7F60 is the interindustry data object tag for the biometric information templates group template.

Appendix C—References

[ANSI504-1] Generic Identity Command Set – *Part 1: Card Application Command Set*.

[FIPS201] Federal Information Processing Standard 201-2, *Personal Identity Verification (PIV) of Federal Employees and Contractors*, August 2013. (See http://csrc.nist.gov)

[ISO7816] ISO/IEC 7816 (Parts 4, 5, 6, 8, and 9), *Information technology — Identification cards — Integrated circuit(s) cards with contacts.*

[ISO8824] ISO/IEC 8824-2:2002, *Information technology -- Abstract Syntax Notation One (ASN.1): Information object specification.*

[ISO8825] ISO/IEC 8825-1:2002, *Information technology — ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER).*

[PKCS1] Jakob Jonsson and Burt Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1," RFC 3447, February 2003. (See http://tools.ietf.org/html/rfc3447)

[SECG] Standards for Efficient Cryptography Group (SECG), "SEC 1: Elliptic Curve Cryptography," Version 1.0, September 2000.

[SP800-38B] NIST Special Publication 800-38B, *Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*, May 2005. (See http://csrc.nist.gov)

[SP800-56A] NIST Special Publication 800-56A Revision 2, *Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography*, May 2013. (See http://csrc.nist.gov)

[SP800-76] NIST Special Publication 800-76-2, *Biometric Specifications for Personal Identity Verification*, July 2013. (See http://csrc.nist.gov)

[SP800-78] Revised Draft NIST Special Publication 800-78-4, Cryptographic *Algorithms and Key Sizes for Personal Identity Verification*. (See http://csrc.nist.gov)