Archived NIST Technical Series Publication

The attached publication has been archived (withdrawn), and is provided solely for historical purposes. It may have been superseded by another publication (indicated below).

Archived Publication

Series/Number:	NIST Special Publication 800-73-4
Title:	Interfaces for Personal Identity Verification
Publication Date(s):	May 2015
Withdrawal Date:	February 12, 2016
Withdrawal Note:	SP 800-73-4 is superseded in its entirety by the publication of SP 800-73-4 (May 2015, including updates as of 2/8/2016).

Superseding Publication(s)

The attached publication has been **superseded by** the following publication(s):

Series/Number:	NIST Special Publication 800-73-4
Title:	Interfaces for Personal Identity Verification
Author(s):	H. Ferraiolo; K. Mehta; S. Francomacaro; R. Chandramouli; J. Mohler
Publication Date(s):	May 2015 (including updates as of 2/8/2016)
URL/DOI:	http://dx.doi.org/10.6028/NIST.SP.800-73-4

Additional Information (if applicable)

Contact:	Computer Security Division (Information Technology Laboratory)
Latest revision of the attached publication:	SP 800-73-4 (as of February 12, 2016)
Related information:	http://csrc.nist.gov/groups/SNS/piv/
Withdrawal announcement (link):	N/A

Date updated: February 12, 2016

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

NIST Special Publication 800-73-4

Interfaces for Personal Identity Verification – Part 1: PIV Card Application Namespace, Data Model and Representation

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

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.SP.800-73-4

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

NIST Special Publication 800-73-4

Interfaces for Personal Identity Verification – Part 1: PIV Card Application Namespace, Data Model and Representation

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

Jason Mohler
Electrosoft Services, Inc.
Reston, Virginia

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.SP.800-73-4

May 2015

U.S. Department of Commerce
Penny Pritzker, Secretary

National Institute of Standards and Technology
Willie May, 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 Modernization 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.

Nothing in this publication should be taken to contradict the standards and guidelines made mandatory and binding on Federal agencies by the Secretary of Commerce under statutory authority. Nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other Federal official. This publication may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright in the United States. Attribution would, however, be appreciated by NIST.

National Institute of Standards and Technology Special Publication 800-73-4
Natl. Inst. Stand. Technol. Spec. Publ. 800-73-4, 64 pages (May 2015)
http://dx.doi.org/10.6028/NIST.SP.800-73-4
CODEN: NSPUE2

Comments on this publication may be submitted to:

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 (David Cooper, Hildegard Ferraiolo, Ketan Mehta, Salvatore Francomacaro, and Ramaswamy Chandramouli 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.

I. Revision History

Version	Release Date	Updates
SP 800-73	April 2005	Initial Release
SP 800-73-1	April 2006	Incorporated Errata
SP 800-73-2	September 2008	• Separated SP 800-73 into four Parts: 1 - *End-Point PIV Card Application Namespace, Data Model and Representation* *2 - End-Point PIV Card Application Card Command Interface* *3 - End-Point PIV Client Application Programming Interface* *4 - The PIV Transitional Interface and Data Model Specification* • All PIV cryptographic key types, cryptographic algorithm identifiers, and key sizes previously listed in SP 800-73-1, are now specified in SP 800-78, *Cryptographic Algorithms and Key Sizes for Personal Identity Verification* • Removed default algorithms. Each PIV key type can be implemented from a small subset of algorithms and key sizes as specified in Table 3-1 of SP 800-78 • Added optional Discovery Object (Part 1, Section 3.2.6) • Added optional capability to use the Global PIN (in addition to the PIV Card Application PIN) with the PIV Card Application (Part 1, Section 3.2.6) • Added pivMiddlewareVersion API function (Part 3, Section 3.1.1) • Deprecated the CHUID data object's Authentication Key Map data element • Deprecated the Printed Information data object's Employee Affiliation Line 2 data element (tag 0x03) • Removed size limits on signed data object containers (Part 1, Appendix A)
SP 800-73-3	February 2010	• Added preamble: I - Revision History, II - Configuration Management and III – NPIVP Conformance Testing. (Part 1, Preamble) • Removed the CHUID data object's Authentication Key Map data element • Removed the Printed Information data object's Employee Affiliation Line 2 data element (tag 0x03) • Deprecated IPv6 as optional value for the CHUID's GUID data element (Part 1, Section 3.2.1) • Added Key History capability (Part 1, Section 3.2.7) • Added ECDH key agreement scheme (Part 2, Section 3.2.4) • Added UUID feature for non-Federal issuer cards (Part 1, Section 3.3) • Expanded Part 2, Appendix A (GENERAL AUTHENTICATE examples) to illustrate ECDSA signatures and key establishment schemes with the key management key • Added an optional cardholder iris images data object, which is specified in SP 800-76-2. • Added Appendix C, PIV Algorithm Identifier Discovery. • Updated PIV Middleware version number in Part 3.

Version	Release Date	Updates
SP 800-73-4	April 2015	• Removed Part 4, The PIV Transitional Data Model and Interfaces • Removed "End-Point" from the titles and content of Parts 1 through 3 • Added Section 1.3 "Effective Date" • Made asymmetric Card Authentication key mandatory • Made digital signature key and key management key conditionally mandatory • Made the facial image data object mandatory • Introduced specifications for optional secure messaging • Introduced specifications for optional virtual contact interface (VCI) over which all non-card-management functionality of the PIV Card is accessible • Added support for pairing code that is used to establish VCI • Made Card UUID mandatory. Thus, removed the option to populate the GUID data element of CHUID with all zeros or an IPv6 address • Added PIV card level PIN length enforcement requirements for the PINs • Added an optional Cardholder UUID as a unique identifier for a cardholder • Removed information about encoding of NFI cards • Added optional on-card biometric comparison mechanism as a means of performing card activation and as a PIV authentication mechanism • Added requirement for signature verification and certification path validation in the CHUID, BIO, and BIO-A authentication mechanisms • Added the On Card Comparison (OCC) Biometric Information (BIT) Group Template Data Object • Added Secure Messaging Signer Certificate Data Object • Added Pairing Code Reference Data Container • Deprecated some data elements in the CHUID (Buffer Length, DUNS and Organization Identifier) and legacy data elements in all X.509 Certificates (MSCUID) • Deprecated the optional Extended Application CardURL and Security Object Buffer data elements from the Card Capability Container • Updated PIV Middleware version number in Part 3 • Expanded Part 1, Appendix C (PIV Algorithm Identifier Discovery) to include an Algorithm Identifier discovery for Secure Messaging • Expanded Part 2, Appendix A (GENERAL AUTHENTICATE examples) to illustrate use of VCI

II. Configuration Management

When a Federal agency adds one or several optional features listed in the previous section (Revision History) to its PIV Cards, it is necessary for client applications to upgrade the PIV Middleware accordingly. This will enable the PIV Middleware to recognize and process the new data objects and/or features.

Where maximum interoperability is required, it is necessary to upgrade to SP 800-73-4 based PIV Middleware as they become available. Only SP 800-73-4 based PIV Middleware fully support all capabilities outlined in the Revision History.[1] Previous versions of the PIV Middleware (based on SP800-73-3, SP 800-73-2, or SP 800-73-1) are unaware of new SP 800-73-4 features and thus have the following limitations:

+ SP 800-73-3 based PIV Middleware:
    - Do not support On-card Biometric Comparison
    - Do not support Secure Messaging.

    Recommendation: SP 800-73-3 based PIV Middleware should be restricted to applications that do not use the above features.

+ In addition to the limitations listed above, SP 800-73-2 based PIV Middleware:
    - Do not support the Key History feature.
    - Do not support the iris images data object.

    Recommendation: SP 800-73-2 based PIV Middleware should be restricted to applications that do not use the new features supported by SP 800-73-3 and SP 800-73-4 based middleware.

+ In addition to the limitations listed above, SP 800-73-1 based PIV Middleware:
    - Do not recognize the PIV Discovery Object and thus are unable to recognize or prompt for the Global PIN for PIV Cards with Global PIN enabled.
    - Do not support the PIV Middleware version API function.

    Recommendation: SP 800-73-1 based PIV Middleware should be restricted to applications that do not use the new features supported by SP 800-73-2, SP 800-73-3, and SP 800-73-4 based middleware.

[1] Implementation of secure messaging and virtual contact interface are optional.

III NPIVP Conformance Testing

As outlined in FIPS 201-2, Appendix A.3, NIST has established the NIST Personal Identity Verification Program (NPIVP) to:

+ validate the compliance/conformance of two PIV components: PIV Middleware and PIV Card Applications with the specifications in NIST SP 800-73 and
+ provide the assurance that the set of PIV Middleware and PIV Card Applications that have been validated by NPIVP are interoperable.

For further information on NPIVP, see http://csrc.nist.gov/groups/SNS/piv/npivp/index.html.

With the final release of SP 800-73-4, NPIVP plans to revise and publish SP 800-85A-4, PIV Card Application and Middleware Interface Test Guidelines. This document will outline the Derived Test Requirements (DTRs) of SP 800-73-4 based PIV Card Applications and PIV Middleware. In parallel, NPIVP plans to update the test tools for NPIVP laboratories to test PIV Card Applications and PIV Middleware in accordance with the DTRs in SP 800-85A-4. Once SP 800-85A-4 is published, and the test tools are available to NPIVP test laboratories, SP 800-73-3 based testing will be discontinued and SP 800-73-4 based testing will begin. NPIVP will announce the start of SP 800-73-4 based testing at http://csrc.nist.gov/groups/SNS/piv/npivp/announcements.html.

Table of Contents

LIST OF APPENDICES

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. Federal Information Processing Standard 201 [FIPS201], Personal Identity Verification (PIV) of Federal Employees and Contractors, 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[2]) 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, 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.

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 this document. Interoperability is defined as the use of PIV identity credentials such that client-application programs, compliant card applications, and compliant integrated circuits cards (ICC) 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 1 – *PIV Card Application Namespace, Data Model and Representation*, specifies the PIV Card Application Namespace, the PIV Data Model and its logical representation on the PIV Card, and is a companion document to FIPS 201.

1.3 Effective Date

In order to comply with the implementation schedule in FIPS 201-2, Federal departments and agencies shall implement these recommendations immediately upon publication, with the

[2] 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 an automated process (computer readable and verifiable) or by another person (human readable and verifiable).

exception of the requirement for the PIV Card Application to enforce the minimum length requirements for the PINs.

The requirement to enforce minimum length for the PINs at the card level is a security requirement that did not appear in previous versions of SP 800-73. The implementation schedule for this new requirement shall be phased in as part of new card stock acquisition by Federal departments and agencies after final publication of this document.

1.4 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.

1.5 Document Overview and Structure

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 this document:

+ Section 1, *Introduction*, provides the purpose, scope, effective date, audience, and assumptions, of the document and outlines its structure.
+ Section 2, *PIV Card Application Namespaces*, defines the three NIST managed namespaces used by the PIV Card Application.
+ Section 3, *PIV Data Model Elements*, describes the PIV Data Model elements in detail.
+ Section 4, *PIV Data Objects Representation*, describes the format and coding of the PIV data structures used by the PIV client-application programming interface and the PIV Card Application.
+ Section 5, *Data Types and Their Representation*, provides the details of the data types found on the PIV client-application programming interface and the PIV Card Application card command interface.
+ Appendix A provides container information of PIV Cards and is normative. All other appendices are informative and contain material that needs special formatting together with illustrative material to aid in understanding information in the body of the document.

2. PIV Card Application Namespaces

2.1 Namespaces of the PIV Card Application

Names used on the PIV interfaces are drawn from three namespaces managed by NIST:

+ Proprietary Identifier eXtension (PIX) of the NIST Registered Application Provider IDentifier (RID)
+ ASN.1 object identifiers (OIDs) in the personal identity verification subset of the OIDs managed by NIST
+ Basic Encoding Rules – Tag Length Value (BER-TLV) tags of the NIST PIV coexistent tag allocation scheme

All unspecified names in these managed namespaces are reserved for future use.

All interindustry tags defined in ISO/IEC 7816, *Information Technology – Identification Cards – Integrated Circuit(s) Card with Contacts* [ISO7816], and used in the NIST coexistent tag allocation scheme without redefinition have the same meaning as they have in [ISO7816].

All unspecified values in the following identifier and value namespaces are reserved for future use:

+ algorithm identifiers
+ key reference values
+ cryptographic mechanism identifiers

2.2 PIV Card Application AID

The Application IDentifier (AID) of the Personal Identity Verification Card Application (PIV Card Application) shall be:

'A0 00 00 03 08 00 00 10 00 01 00'

The AID of the PIV Card Application consists of the NIST RID ('A0 00 00 03 08') followed by the application portion of the NIST PIX indicating the PIV Card Application ('00 00 10 00') and then the version portion of the NIST PIX ('01 00') for the first version of the PIV Card Application. All other PIX sequences on the NIST RID are reserved for future use.

The PIV Card Application can be selected as the current application by providing the full AID as listed above or by providing the right-truncated version; that is, without the two-byte version, as follows:

'A0 00 00 03 08 00 00 10 00'

3. PIV Data Model Elements

This section contains the description of the data elements for personal identity verification, the PIV data model.

A PIV Card Application shall contain seven mandatory interoperable data objects, two conditionally mandatory data objects, and may contain twenty-seven optional data objects. The seven mandatory data objects for interoperable use are as follows:

1. Card Capability Container
2. Card Holder Unique Identifier
3. X.509 Certificate for PIV Authentication
4. X.509 Certificate for Card Authentication
5. Cardholder Fingerprints
6. Cardholder Facial Image
7. Security Object

The two data objects that are mandatory if the cardholder has a government-issued email account at the time of credential issuance are:

1. X.509 Certificate for Digital Signature
2. X.509 Certificate for Key Management

The twenty-seven optional data objects are as follows:

1. Printed Information
2. Discovery Object
3. Key History Object
4. 20 retired X.509 Certificates for Key Management
5. Cardholder Iris Images
6. Biometric Information Templates Group Template
7. Secure Messaging Certificate Signer
8. Pairing Code Reference Data Container

3.1 Mandatory Data Elements

This section describes the seven mandatory data objects for interagency interoperable use.

3.1.1 Card Capability Container

The Card Capability Container (CCC) is a mandatory data object whose purpose is to facilitate compatibility of Government Smart Card Interoperability Specification (GSC-IS) applications with PIV Cards.

The CCC supports minimum capability for retrieval of the data model and optionally the application information as specified in [GSC-IS]. The data model of the PIV Card Application shall be identified by data model number 0x10. Deployed applications use 0x00 through 0x04. This enables the GSC-IS application domain to correctly identify a new data model namespace and structure as defined in this document.

For PIV Card Applications, the PIV data objects exist in a namespace tightly managed by NIST and a CCC discovery mechanism is not needed by client applications that are not based on GSC-IS. Therefore, all mandatory data elements of the CCC, except for the data model number, may optionally have a length value set to zero bytes (i.e., no value field will be supplied). Unused optional data elements shall be absent. The content of the CCC data elements, other than the data model number, are out of scope for this specification.

The Security Object enforces integrity of the CCC according to the issuer.

3.1.2 Card Holder Unique Identifier

The Card Holder Unique Identifier (CHUID) data object is defined in accordance with the Technical Implementation Guidance: Smart Card Enabled Physical Access Control Systems (TIG SCEPACS) [TIG SCEPACS]. For this specification, the CHUID is common between the contact and contactless interfaces. For dual chip implementations, the CHUID is copied in its entirety between the two chips.

In addition to the requirements specified in TIG SCEPACS, the CHUID on the PIV Card shall meet the following requirements:

+ The optional Buffer Length TLV element is deprecated and will be eliminated in a future version of SP 800-73. This element is the length in bytes of the entire CHUID, excluding the Buffer Length element itself, but including the CHUID's Asymmetric Signature element. The calculation of the asymmetric signature must exclude the Buffer Length element if it is present.

+ The previously deprecated Authentication Key Map data element shall not be present in the CHUID.[3]

+ The Federal Agency Smart Credential Number (FASC-N) shall be in accordance with TIG SCEPACS [TIG SCEPACS] with the exception that credential series, individual credential issue, person identifier, organizational category, organizational identifier, and person/organization association category may be populated with all zeros.

  A subset of the FASC-N, the FASC-N Identifier, shall be the unique identifier as described in [TIG SCEPACS, Section 6.6]: "The combination of an Agency Code, System Code, and Credential Number is a fully qualified number that is uniquely assigned to a single individual." The Agency Code is assigned to each department or agency by SP 800-87, *Codes for Identification of Federal and Federally-Assisted Organizations* [SP800-87]. The subordinate System Code and Credential Number value assignment is subject to department or agency policy, provided that the FASC-N identifier (i.e., the concatenated Agency Code, System Code, and Credential Number) is unique for each card. The same FASC-N value shall be used in all the PIV data objects that include the FASC-N. To eliminate unnecessary use of the SSN,[4] the FASC-N's Person Identifier (PI) field should not encode the SSN. TIG SCEPACS also specifies PACS interoperability requirements in the 10th paragraph of [TIG SCEPACS, Section 2.1]: "For full interoperability of a PACS it must at a minimum be able to distinguish fourteen digits (i.e., a combination of an Agency Code, System Code, and Credential Number) when matching FASC-N based credentials to enrolled card holders."

[3] See Revision History in preamble of this document.
[4] See the attachment to OMB M-07-16, Section 2: "Reduce the Use of Social Security Numbers."

+ The optional DUNS and Organizational Identifier fields are deprecated and will be eliminated in a future version of SP 800-73.

+ The Global Unique Identification number (GUID) field must be present, and shall include a Card Universally Unique Identifier (UUID) (see Section 3.4.1).

+ The Expiration Date is mapped to the reserved for future use (RFU) tag 0x35, keeping that within the existing scope of the TIG SCEPACS specification. This field shall be 8 bytes in length and shall be encoded in ASCII as YYYYMMDD. The expiration date shall be the same as printed on the card.

+ The optional Cardholder UUID field is mapped to RFU tag 0x36. If present, it shall include a Cardholder UUID as described in Section 3.4.2.

+ The CHUID shall be signed in accordance with Section 3.1.2.1. The card issuer's digital signature key shall be used to sign the CHUID and the associated certificate shall be placed in the signature field of the CHUID.

3.1.2.1 Asymmetric Signature Field in CHUID

FIPS 201 requires inclusion of the asymmetric signature field in the CHUID data object. The asymmetric signature data element of the CHUID shall be encoded as a Cryptographic Message Syntax (CMS) external digital signature, as defined in RFC 5652 [RFC5652].

The issuer asymmetric signature field is implemented as a *SignedData* type, as specified in [RFC5652], and shall include the following information:

+ The message shall include a *version* field specifying version v3

+ The *digestAlgorithms* field shall be as specified in [SP800-78]

+ The *encapContentInfo* shall:

  – Specify an *eContentType* of id-PIV-CHUIDSecurityObject

  – Omit the *eContent* field

+ The *certificates* field shall include only a single X.509 certificate, which can be used to verify the signature in the *SignerInfo* field

+ The *crls* field shall be omitted

+ *signerInfo*s shall be present and include only a single *SignerInfo*

+ The *SignerInfo* shall:

  – Use the *issuerAndSerialNumber* choice for *SignerIdentifier*

  – Specify a *digestAlgorithm* in accordance with [SP800-78]

  – Include, at a minimum, the following signed attributes:

    - A *MessageDigest* attribute containing the hash computed in accordance with [SP800-78]

- A *pivSigner-DN* attribute containing the subject name that appears in the PKI certificate for the entity that signed the CHUID

– Include the digital signature.

The public key required to verify the digital signature shall be provided in the *certificates* field in an X.509 digital signature certificate that has been issued in accordance with Section 4.2.1 of FIPS 201-2.

3.1.3 X.509 Certificate for PIV Authentication

The X.509 Certificate for PIV Authentication and its associated private key, as defined in FIPS 201, is used to authenticate the card and the cardholder. The PIV Authentication private key and its corresponding certificate are only available over the contact interface or virtual contact interface (VCI). The read access control rule for the X.509 Certificate for PIV Authentication is "Always," meaning the certificate can be read without access control restrictions. The Public Key Infrastructure (PKI) cryptographic function (see Table 4b) is protected with a Personal Identification Number (PIN) or on-card biometric comparison (OCC) access rule. In other words, private key operations using the *PIV Authentication key* require the PIN or OCC data to be submitted and verified, but a successful submission enables multiple private key operations without additional cardholder consent.

3.1.4 X.509 Certificate for Card Authentication

FIPS 201 specifies the mandatory asymmetric Card Authentication key (CAK) as a private key that may be used to support physical access applications. The read access control rule of the corresponding X.509 Certificate for Card Authentication is "Always," meaning the certificate can be read without access control restrictions. The PKI cryptographic function (see Table 4b) is under an "Always" access rule, and thus private key operations can performed without access control restrictions. The asymmetric CAK is generated by the PIV Card Issuer in accordance with FIPS 140-2 requirements for key generation. An asymmetric CAK may be generated on-card or off-card. If an asymmetric CAK is generated off-card, the result of each key generation shall be injected into at most one PIV Card.

3.1.5 Cardholder Fingerprints

The fingerprint data object specifies the primary and secondary fingerprints for off-card matching in accordance with FIPS 201 and [SP800-76].

3.1.6 Cardholder Facial Image

The facial image data object supports visual authentication by a guard, and may also be used for automated facial authentication in operator-attended PIV issuance, reissuance, and verification data reset processes. The facial image data object shall be encoded as specified in [SP800-76].

3.1.7 Security Object

The Security Object is in accordance with Appendix 3 to Section IV of Volume 2 of Part 3 of Machine Readable Travel Documents (MRTD) [MRTD]. Tag 0xBA is used to map the ContainerIDs in the PIV data model to the 16 Data Groups specified in the MRTD. The mapping enables the Security Object to be fully compliant for future activities with identity documents.

The "DG-number-to-Container-ID" mapping object TLV in tag 0xBA encapsulates a series of three-byte sequences – one for each PIV data object included in the Security Object. The first byte is the Data Group (DG) number, and the second and third bytes are the most and least significant bytes (respectively) of the Container ID value. The DG number assignment is arbitrary; however, the same number assignment applies to the DataGroupNumber(s) in the DataGroupHash(es). This will ensure that the ContainerIDs in the mapping object refer to the correct hash values in the Security Object (0xBB).

The 0xBB Security Object is formatted according to [MRTD, Appendix 3 to Section IV]. The Logical Data Structure (LDS) Security Object itself must be in ASN.1 DER format, formatted as specified in [MRTD, Appendix A.3.2]. This structure is then inserted into the *encapContentInfo* field of the Cryptographic Message Syntax (CMS) object specified in [MRTD, Appendix A.3.1].

The card issuer's digital signature key used to sign the CHUID shall also be used to sign the Security Object. The signature field of the Security Object, tag 0xBB, shall omit the issuer's certificate, since it is included in the CHUID. At a minimum, unsigned data objects, such as the Printed Information data object, shall be included in the Security Object if present. For maximum protection against credential splicing attacks (credential substitution), it is recommended, however, that all PIV data objects, except the PIV X.509 certificates and the Secure Messaging Certificate Signer data object, be included in the Security Object.

3.2 Conditional Data Elements

The following two data elements are mandatory if the cardholder has a government-issued email account at the time of credential issuance. These two data elements, when implemented, shall conform to the specifications provided in this document.

3.2.1 X.509 Certificate for Digital Signature

The X.509 Certificate for Digital Signature and its associated private key, as defined in FIPS 201, support the use of digital signatures for the purpose of document signing. The digital signature private key and its corresponding certificate are only available over the contact interface or VCI. The read access control rule for the X.509 Certificate for Digital Signing is "Always," meaning the certificate can be read without access control restrictions. The PKI cryptographic function (see Table 4b) is protected with a "PIN Always" or "OCC Always" access rule. In other words, the PIN or OCC data must be submitted and verified every time immediately before a *digital signature key* operation. This ensures cardholder participation every time the private key is used for digital signature generation.[5]

3.2.2 X.509 Certificate for Key Management

The X.509 Certificate for Key Management and its associated private key, as defined in FIPS 201, support the use of encryption for the purpose of confidentiality. The key management private key and its corresponding certificate are only available over the contact interface or VCI. This key pair may be escrowed by the issuer for key recovery purposes. The read access control rule for the X.509 certificate is "Always," meaning the certificate can be read without access control restrictions. The PKI cryptographic function (see Table 4b) is protected with a "PIN" or "OCC" access rule. In other words, once the PIN or OCC data is submitted and verified, subsequent *key management key*

[5] [NISTIR7863], *Cardholder Authentication for the PIV Digital Signature Key*, addresses the appropriate use of PIN caching related to digital signatures.

operations can be performed without requiring the PIN or OCC data again. This enables multiple private key operations without additional cardholder consent.

3.3 Optional Data Elements

The twenty-seven optional data elements of FIPS 201, when implemented, shall conform to the specifications provided in this document.

3.3.1 Printed Information

All FIPS 201 mandatory information printed on the card is duplicated on the chip in this data object. The printed information data object shall not be modified post-issuance. The Security Object enforces integrity of this information according to the issuer. This provides specific protection that the card information must match the printed information, mitigating alteration risks on the printed media.

3.3.2 Discovery Object

The Discovery Object, if implemented, is the 0x7E interindustry ISO/IEC 7816-6 template that nests interindustry data objects. For the Discovery Object, the 0x7E template nests two mandatory BER-TLV structured interindustry data elements: 1) tag 0x4F contains the AID of the PIV Card Application and 2) tag 0x5F2F lists the PIN Usage Policy.

+ Tag 0x4F encodes the PIV Card Application AID as follows:

  {'4F 0B A0 00 00 03 08 00 00 10 00 01 00'}

+ Tag 0x5F2F encodes the PIN Usage Policy in two bytes:

  First byte:
  - Bit 7 is set to 1 to indicate that the mandatory PIV Card Application PIN satisfies the PIV Access Control Rules (ACRs) for command execution[6] and data object access.
  - Bit 6 indicates whether the optional Global PIN satisfies the PIV ACRs for command execution and PIV data object access.
  - Bit 5 indicates whether the optional OCC satisfies the PIV ACRs for command execution and PIV data object access
  - Bit 4 indicates whether the optional VCI is implemented
  - Bit 3 is set to zero if the pairing code is required to establish a VCI and is set to one if a VCI is established without pairing code
  - Bits 8, 2, and 1 of the first byte shall be set to zero.

  Table 1 lists the acceptable values for the first byte of the PIN Usage Policy and summarizes the meaning of each value.

[6] Command execution pertains to the VERIFY APDU and optionally to the CHANGE REFERENCE DATA APDU.

The second byte of the PIN Usage Policy encodes the cardholder's PIN preference for PIV Cards with both the PIV Card Application PIN and the Global PIN enabled:

Second byte: 0x10 indicates that the PIV Card Application PIN is the primary PIN used to satisfy the PIV ACRs for command execution and object access.

0x20 indicates that the Global PIN is the primary PIN used to satisfy the PIV ACRs for command execution and object access.

Note: If Bit 6 of the first byte of the PIN Usage Policy is set to zero, then the second byte is RFU and shall be set to 0x00.

PIV Card Applications that implement the VCI or for which the Global PIN or OCC satisfy the PIV ACRs for PIV data object access and command execution shall implement the Discovery Object.

Table 1. First Byte of PIN Usage Policy Discovery

Value	PIV Card Application PIN	Global PIN	OCC	VCI	Pairing Code Required
0x40	✓				
0x48	✓			✓	✓
0x4C	✓			✓	
0x50	✓		✓		
0x58	✓		✓	✓	✓
0x5C	✓		✓	✓	
0x60	✓	✓			
0x68	✓	✓		✓	✓
0x6C	✓	✓		✓	
0x70	✓	✓	✓		
0x78	✓	✓	✓	✓	✓
0x7C	✓	✓	✓	✓	

The encoding of the 0x7E Discovery Object is as follows:

{'7E 12' {'4F 0B A0 00 00 03 08 00 00 10 00 01 00'} {'5F 2F 02 xx yy'}}, where xx and yy encode the first and second byte of the PIN Usage Policy as described in this section.

The Security Object enforces integrity of the Discovery Object according to the issuer.

3.3.3 Key History Object

Up to twenty retired key management private keys may be stored in the PIV Card Application. The Key History object provides information about the retired key management private keys that are present within the PIV Card Application.[7] Retired key management private keys are private keys that correspond to X.509 Certificates for Key Management that have expired, have been revoked, or have otherwise been superseded. The Key History object shall be present in the PIV Card Application if

[7] See NIST Interagency Report 7676 [IR7676] for suggestions on the implementation and use of the Key History mechanism.

the PIV Card Application contains any retired key management private keys, but may be present even if no such keys are present in the PIV Card Application. For each retired key management private key in the PIV Card Application, the corresponding certificate may either be present within the PIV Card Application or may only be available from an on-line repository.

The Key History object includes two mandatory fields, *keysWithOnCardCerts* and *keysWithOffCardCerts*, and one optional field, *offCardCertURL*. The *keysWithOnCardCerts* field indicates the number of retired private keys within the PIV Card Application for which the corresponding certificates are also stored within the PIV Card Application. The *keysWithOffCardCerts* field indicates the number of retired private keys within the PIV Card Application for which the corresponding certificates are not stored within the PIV Card Application. The numeric values in both *keysWithOnCardCerts* and *keysWithOffCardCerts* are represented as unsigned binary integers. The *offCardCertURL* field contains a URL that points to a file containing the certificates corresponding to all of the retired private keys within the PIV Card Application, including those for which the corresponding certificate is also stored within the PIV Card Application. The *offCardCertURL* field shall be present if the *keysWithOffCardCerts* value is greater than zero and shall be absent if the values of both *keysWithOnCardCerts* and *keysWithOffCardCerts* are zero. The *offCardCertURL* field may be present if the *keysWithOffCardCerts* value is zero but the *keysWithOnCardCerts* value is greater than zero.

The file that is pointed to by the *offCardCertURL* field shall contain the DER encoding of the following data structure:

```
OffCardKeyHistoryFile ::= SEQUENCE SIZE (1..20) OF SEQUENCE {
        keyReference        OCTET STRING (SIZE(1))
        cert                Certificate
}
```

where **keyReference** is the key reference for the private key on the card and **cert** is the corresponding X.509 certificate.[8] The *offCardCertURL* field shall have the following format:

"http://" <DNS name> "/" <ASCII-HEX encoded SHA-256 hash of **OffCardKeyHistoryFile**>

The private keys for which the corresponding certificates are stored within the PIV Card Application shall be assigned to the lowest numbered key references reserved for retired key management private keys. For example if *keysWithOnCardCerts* is 5, then the corresponding private keys shall be assigned to key references '82', '83', '84', '85', and '86'.

The private keys for which the corresponding certificates are not stored within the PIV Card Application shall be assigned to the highest numbered key references reserved for retired key management private keys. For example, if *keysWithOffCardCerts* is 3, then the corresponding private keys shall be assigned to key references '93', '94', and '95'.

Private keys do not have to be stored within the PIV Card Application in the order of their age. However, if the certificates corresponding to only some of the retired key management private keys are available within the PIV Card Application then the certificates that are stored in the PIV Card Application shall be the ones that were most recently issued.

[8] The ASN.1 for **Certificate** may be imported from the ASN.1 module **PKIX1Explicit88** in Appendix A.1 of [RFC5280].

The Key History object is only available over the contact interface and VCI. The read access control rule for the Key History object is "Always," meaning that it can be read without access control restrictions.

The Security Object enforces integrity of the Key History object according to the issuer.

3.3.4 Retired X.509 Certificates for Key Management

These objects hold the X.509 Certificates for Key Management corresponding to retired key management private keys, as described in Section 3.3.3. Retired key management private keys and their corresponding certificates are only available over the contact interface or VCI. The read access control rule for these certificates is "Always," meaning the certificates can be read without access control restrictions. The PKI cryptographic function (see Table 4b) for all of the retired *key management private keys* is protected with a "PIN" or "OCC" access rule. In other words, once the PIN or OCC data is submitted and verified, subsequent key management key operations can be performed with any of the retired key management private keys without requiring the PIN or OCC data again. This enables multiple private key operations without additional cardholder consent.

3.3.5 Cardholder Iris Images

The iris images data object specifies compact images of the cardholder's irises. The images are suitable for use in iris recognition systems for automated identity verification. The iris images data object shall be encoded as specified in [SP800-76].

3.3.6 Biometric Information Templates Group Template

The Biometric Information Templates (BIT) Group Template data object encodes the configuration information of the OCC data. The encoding of the BIT Group Template shall be as specified in Table 7 of [SP800-76]. When OCC satisfies the PIV ACRs for PIV data objects access and command execution both the Discovery Object and the BIT Group Template data object shall be present, and bit 5 of the first byte of the PIN Usage Policy shall be set. The BIT Group Template may be present when OCC does not satisfy the PIV ACRs for PIV data objects access, but, if present, shall contain no BITs.[9] The Security Object enforces integrity of the BIT Group Template data object according to the issuer.

3.3.7 Secure Messaging Certificate Signer

The Secure Messaging Certificate Signer data object, which shall be present if the PIV Card supports secure messaging for non-card-management operations, contains the certificate(s) needed to verify the signature on the secure messaging card verifiable certificate (CVC), as specified in Part 2, Section 4.1.5.

The public key required to verify the digital signature of the secure messaging CVC is an ECC key. It shall be provided in either an X.509 Certificate for Content Signing or an Intermediate CVC. If the public key required to verify the digital signature of the secure messaging CVC is provided in an Intermediate CVC, then the format of the Intermediate CVC shall be as specified in Part 2, Section 4.1.5, and the public key required to verify the digital signature of the Intermediate CVC shall be provided in an X.509 Certificate for Content Signing.

[9] A BIT Group Template with no BITs is encoded as '7F 61 03 02 01 00'.

The X.509 Certificate for Content Signing shall be a digital signature certificate issued under the id-fpki-common-piv-contentSigning policy of [COMMON]. The X.509 Certificate for Content Signing shall also include an extended key usage (*extKeyUsage)* extension asserting id-PIV-content-signing. Additional descriptions for the PIV object identifiers are provided in Appendix B of FIPS 201-2. The X.509 Certificate for Content Signing needed to verify the digital signature of a secure messaging CVC or Intermediate CVC of a valid PIV Card[10] shall not be expired.

Note that the option to include an Intermediate CVC is included as a temporary measure to accommodate the use of certification authorities that do not support the issuance of X.509 certificates that contain elliptic curve subject public keys. It is expected that the Intermediate CVC data element will be deprecated in a future version of SP 800-73.

3.3.8 Pairing Code Reference Data Container

The Pairing Code Reference Data Container, which shall be present if the PIV Card supports the virtual contact interface, includes a copy of the PIV Card's pairing code (see Section 5.1.3). The Security Object enforces integrity of the Pairing Code Reference Data Container according to the issuer.

3.4 Inclusion of Universally Unique IDentifiers (UUIDs)

This specification provides support for two UUIDs on a PIV Card. The Card UUID is a UUID that is unique for each card, and it shall be present on all PIV Cards. The Cardholder UUID is a UUID that is a persistent identifier for the cardholder, and it is optional to implement. The requirements for these UUIDs are provided in the following subsections.

3.4.1 Card UUID

FIPS 201 requires PIV Cards to include a Card UUID. The Card UUID shall be included on PIV Cards as follows:

1. The value of the GUID data element of the CHUID data object shall be a 16-byte binary representation of a valid UUID [RFC4122]. The UUID shall be version 1, 4, or 5, as specified in [RFC4122, Section 4.1.3].

2. The same 16-byte binary representation of the UUID value shall be present as the value of an entryUUID attribute, as defined in [RFC4530], in any CMS-signed data object that is required to contain a pivFASC-N attribute on a PIV Card, i.e., in the mandatory cardholder fingerprint template and facial image data objects as well as in the optional cardholder iris images data object when present.

3. If the PIV Card supports secure messaging, then the same 16-byte binary representation of the UUID value shall be used as the Subject Identifier in the secure messaging CVC, as specified in Part 2, Section 4.1.5.

4. The string representation of the same UUID value shall be present in the X.509 Certificate for PIV Authentication and the X.509 Certificate for Card Authentication, in the subjectAltName extension encoded as a URI, as specified by [RFC4122, Section 3].

[10] A valid PIV Card is defined as a PIV Card that is neither expired nor revoked.

3.4.2 Cardholder UUID

As defined in Section 3.1.2, the CHUID may optionally include a Cardholder UUID. When present, the Cardholder UUID shall be a 16-byte binary representation of a valid UUID, and it shall be version 1, 4, or 5, as specified in [RFC4122, Section 4.1.3].

3.5 Data Object Containers and associated Access Rules and Interface Modes

Table 2 defines a high level view of the data model. Each on-card storage container is labeled either as Mandatory (M), Optional (O), or Conditional (C). The conditional data objects are the digital signature key and the key management key, which are mandatory if the cardholder has a government-issued email account at the time of credential issuance. This data model is designed to enable and support dual interface cards. For dual chip implementations, for any container that can be accessed over both the contact interface and the contactless interface (including the virtual contact interface) the data object shall be copied into the corresponding containers on both chips.[11]

Table 2. Data Model Containers

Container Name	Container ID	Access Rule for Read		M/O/C
		Contact	Contactless[12]	
Card Capability Container	0xDB00	Always	VCI	M
Card Holder Unique Identifier	0x3000	Always	Always	M
X.509 Certificate for PIV Authentication	0x0101	Always	VCI	M
Cardholder Fingerprints	0x6010	PIN	VCI and PIN	M
Security Object	0x9000	Always	VCI	M
Cardholder Facial Image	0x6030	PIN	VCI and PIN	M
X.509 Certificate for Card Authentication	0x0500	Always	Always	M
X.509 Certificate for Digital Signature	0x0100	Always	VCI	C
X.509 Certificate for Key Management	0x0102	Always	VCI	C
Printed Information	0x3001	PIN or OCC	VCI and (PIN or OCC)	O
Discovery Object	0x6050	Always	Always	O
Key History Object	0x6060	Always	VCI	O
Retired X.509 Certificate for Key Management 1	0x1001	Always	VCI	O
Retired X.509 Certificate for Key Management 2	0x1002	Always	VCI	O
Retired X.509 Certificate for Key Management 3	0x1003	Always	VCI	O
Retired X.509 Certificate for Key Management 4	0x1004	Always	VCI	O
Retired X.509 Certificate for Key Management 5	0x1005	Always	VCI	O
Retired X.509 Certificate for Key Management 6	0x1006	Always	VCI	O
Retired X.509 Certificate for Key Management 7	0x1007	Always	VCI	O
Retired X.509 Certificate for Key Management 8	0x1008	Always	VCI	O

[11] As a consequence of this requirement, any keys that have to be generated on card cannot be made available over the contactless interface (including the virtual contact interface) in a dual chip implementation. In addition, the asymmetric CAK needs to be generated off-card and loaded onto both chips for dual chip implementations.

[12] The term virtual contact interface (VCI) is used in this document as a shorthand for the following security condition: (command is submitted over secure messaging) AND (the Discovery Object is present) AND (Bit 4 of the first byte of the PIN Usage Policy is one) AND ((the security status indicator associated with the pairing code is TRUE) OR (Bit 3 of the first byte of the PIN Usage Policy is one))

Container Name	Container ID	Access Rule for Read		M/O/C
		Contact	Contactless[12]	
Retired X.509 Certificate for Key Management 9	0x1009	Always	VCI	O
Retired X.509 Certificate for Key Management 10	0x100A	Always	VCI	O
Retired X.509 Certificate for Key Management 11	0x100B	Always	VCI	O
Retired X.509 Certificate for Key Management 12	0x100C	Always	VCI	O
Retired X.509 Certificate for Key Management 13	0x100D	Always	VCI	O
Retired X.509 Certificate for Key Management 14	0x100E	Always	VCI	O
Retired X.509 Certificate for Key Management 15	0x100F	Always	VCI	O
Retired X.509 Certificate for Key Management 16	0x1010	Always	VCI	O
Retired X.509 Certificate for Key Management 17	0x1011	Always	VCI	O
Retired X.509 Certificate for Key Management 18	0x1012	Always	VCI	O
Retired X.509 Certificate for Key Management 19	0x1013	Always	VCI	O
Retired X.509 Certificate for Key Management 20	0x1014	Always	VCI	O
Cardholder Iris Images	0x1015	PIN	VCI and PIN	O
Biometric Information Templates Group Template	0x1016	Always	Always	O
Secure Messaging Certificate Signer	0x1017	Always	Always	O
Pairing Code Reference Data Container	0x1018	PIN or OCC	VCI and (PIN or OCC)	O

Appendix A provides a detailed spreadsheet for the data model. ContainerIDs and tags within the containers for each data object are defined by this data model in accordance with SP 800-73-4 naming conventions.

4. PIV Data Objects Representation

4.1 Data Objects Definition

A *data object* is an item of information seen on the card command interface for which is specified a name, a description of logical content, a format, and a coding. Each data object has a globally unique name called its *object identifier* (OID), as defined in ISO/IEC 8824-2:2002 [ISO8824].

A data object whose data content is encoded as a BER-TLV data structure as in ISO/IEC 8825-1:2002 [ISO8825] is called a *BER-TLV data object.*

4.1.1 Data Object Content

The content of a data object is the sequence of bytes that are said to be contained in or to be the value of the data object. The number of bytes in this byte sequence is referred to as the length of the data content and also as the size of the data object. The first byte in the sequence is regarded as being at byte position or offset zero in the content of the data object.

The data content of a BER-TLV data object may consist of other BER-TLV data objects. In this case the tag of the data object indicates that the data object is a constructed data object. A BER-TLV data object that is not a constructed data object is called a primitive data object.

The PIV data objects are BER-TLV objects encoded as per [ISO8825], except that tag values of the PIV data object's inner tag assignments do not conform to BER-TLV requirements.[13] This is due to the need to accommodate legacy tags inherited from [GSC-IS].

Before the card is issued, data objects that are created but not used shall be set to zero-length value.

4.2 OIDs and Tags of PIV Card Application Data Objects

Table 3 lists the ASN.1 object identifiers and BER-TLV tags of the thirty-six PIV Card Application data objects. For the purpose of constructing PIV Card Application data object names in the CardApplicationURL in the CCC of the PIV Card Application, the NIST RID ('A0 00 00 03 08') shall be used and the card application type shall be set to '00'.

4.3 Object Identifiers

Each of the data objects in the PIV Card Application has been provided with a BER-TLV tag and an ASN.1 OID from the NIST personal identity verification arc. These object identifier assignments are given in Table 3.

A data object shall be identified on the PIV client-application programming interface using its OID. An object identifier on the PIV client-application programming interface shall be a dot-delimited string of the integer components of the OID. For example, the representation of the OID of the CHUID on the PIV client-application programming interface is "2.16.840.1.101.3.7.2.48.0."

[13] The exception does not apply to the BIT Group Template, the Discovery Object or the Application Property Template (APT), since these objects use interindustry tags from ISO/IEC 7816-6.

A data object shall be identified on the PIV Card Application card command interface using its BER-TLV tag. For example, the CHUID is identified on the card command interface to the PIV Card Application by the three-byte identifier '5FC102'.

Table 2 lists the ACRs of the thirty-six PIV Card Application data objects.

Table 3. Object Identifiers of the PIV Data Objects for Interoperable Use

Data Object for Interoperable Use	ASN.1 OID	BER-TLV Tag	M/O/C
Card Capability Container	2.16.840.1.101.3.7.1.219.0	'5FC107'	M
Card Holder Unique Identifier	2.16.840.1.101.3.7.2.48.0	'5FC102'	M
X.509 Certificate for PIV Authentication	2.16.840.1.101.3.7.2.1.1	'5FC105'	M
Cardholder Fingerprints	2.16.840.1.101.3.7.2.96.16	'5FC103'	M
Security Object	2.16.840.1.101.3.7.2.144.0	'5FC106'	M
Cardholder Facial Image	2.16.840.1.101.3.7.2.96.48	'5FC108'	M
X.509 Certificate for Card Authentication	2.16.840.1.101.3.7.2.5.0	'5FC101'	M
X.509 Certificate for Digital Signature	2.16.840.1.101.3.7.2.1.0	'5FC10A'	C
X.509 Certificate for Key Management	2.16.840.1.101.3.7.2.1.2	'5FC10B'	C
Printed Information	2.16.840.1.101.3.7.2.48.1	'5FC109'	O
Discovery Object	2.16.840.1.101.3.7.2.96.80	'7E'	O
Key History Object	2.16.840.1.101.3.7.2.96.96	'5FC10C'	O
Retired X.509 Certificate for Key Management 1	2.16.840.1.101.3.7.2.16.1	'5FC10D'	O
Retired X.509 Certificate for Key Management 2	2.16.840.1.101.3.7.2.16.2	'5FC10E'	O
Retired X.509 Certificate for Key Management 3	2.16.840.1.101.3.7.2.16.3	'5FC10F'	O
Retired X.509 Certificate for Key Management 4	2.16.840.1.101.3.7.2.16.4	'5FC110'	O
Retired X.509 Certificate for Key Management 5	2.16.840.1.101.3.7.2.16.5	'5FC111'	O
Retired X.509 Certificate for Key Management 6	2.16.840.1.101.3.7.2.16.6	'5FC112'	O
Retired X.509 Certificate for Key Management 7	2.16.840.1.101.3.7.2.16.7	'5FC113'	O
Retired X.509 Certificate for Key Management 8	2.16.840.1.101.3.7.2.16.8	'5FC114'	O
Retired X.509 Certificate for Key Management 9	2.16.840.1.101.3.7.2.16.9	'5FC115'	O
Retired X.509 Certificate for Key Management 10	2.16.840.1.101.3.7.2.16.10	'5FC116'	O
Retired X.509 Certificate for Key Management 11	2.16.840.1.101.3.7.2.16.11	'5FC117'	O
Retired X.509 Certificate for Key Management 12	2.16.840.1.101.3.7.2.16.12	'5FC118'	O
Retired X.509 Certificate for Key Management 13	2.16.840.1.101.3.7.2.16.13	'5FC119'	O
Retired X.509 Certificate for Key Management 14	2.16.840.1.101.3.7.2.16.14	'5FC11A'	O
Retired X.509 Certificate for Key Management 15	2.16.840.1.101.3.7.2.16.15	'5FC11B'	O
Retired X.509 Certificate for Key Management 16	2.16.840.1.101.3.7.2.16.16	'5FC11C'	O
Retired X.509 Certificate for Key Management 17	2.16.840.1.101.3.7.2.16.17	'5FC11D'	O
Retired X.509 Certificate for Key Management 18	2.16.840.1.101.3.7.2.16.18	'5FC11E'	O
Retired X.509 Certificate for Key Management 19	2.16.840.1.101.3.7.2.16.19	'5FC11F'	O
Retired X.509 Certificate for Key Management 20	2.16.840.1.101.3.7.2.16.20	'5FC120'	O
Cardholder Iris Images	2.16.840.1.101.3.7.2.16.21	'5FC121'	O
Biometric Information Templates Group Template	2.16.840.1.101.3.7.2.16.22	'7F61'	O
Secure Messaging Certificate Signer	2.16.840.1.101.3.7.2.16.23	'5FC122'	O
Pairing Code Reference Data Container	2.16.840.1.101.3.7.2.16.24	'5FC123'	O

5. Data Types and Their Representation

This section provides a description of the data types used in the PIV Client Application Programming Interface (SP 800-73-4, Part 3) and PIV Card Command Interface (SP 800-73-4, Part 2). Unless otherwise indicated, the representation shall be the same on both interfaces.

The data types are defined in Part 1, rather than in Parts 2 and 3 in order to achieve smart card platform independence from Part 1. Thus, non-government smart card programs can readily adopt the interface specifications in Parts 2 and 3 while customizing Part 1 to their own data model, data types, and namespaces.[14]

5.1 Key References

A key reference is a one-byte reference data identifier that specifies a cryptographic key or PIN according to its PIV Key Type. Tables 4a and 4b and SP 800-78, Table 6-1, define the key reference values that shall be used on the PIV interfaces. The key reference values are used, for example, in a cryptographic protocol such as an authentication or a signing protocol. Key references are only assigned to private and secret (symmetric) keys, PINs, PIN Unblocking Key (PUK), OCC, and the pairing code. All other PIV Card Application key reference values are reserved for future use.

Table 4a. PIV Card Application Authentication Data References

Key Reference Value	PIV Reference Data Type	Authenticable Entity	Security Condition for Use		Retry Reset Value	Number of Unblocks
			Contact	Contactless		
'00'	Global PIN	Cardholder	Always	VCI	Platform Specific	Platform Specific
'80'	PIV Card Application PIN	Cardholder	Always	VCI	Issuer Specific	Issuer Specific
'81'	PIN Unblocking Key	PIV Card Application Administrator	Always	Never	Issuer Specific	Issuer Specific
'96'	Primary Finger OCC	Cardholder	Always	SM	Issuer Specific	Issuer Specific
'97'	Secondary Finger OCC	Cardholder	Always	SM	Issuer Specific	Issuer Specific
'98'	Pairing Code	Cardholder	Always[15]	SM	Issuer Specific	Issuer Specific

[14] A customized Part 1 data model exists in the PIV-Interoperable card (PIV-I card) specification as defined in [PIV-I NFI] and further clarified in [PIV-I FAQ]. The intent of [PIV-I NFI] is to enable issuers to issue cards that are technically interoperable with Federal PIV Card readers and their applications, and that may be trusted for particular purposes at the discretion of the relying Federal departments and agencies. PIV-I cards use the same namespace and data types as PIV Cards, however, the data model is slightly different since some of the ASN.1 OIDs that appear in PIV certificates are specific to PIV Cards and since non-Federal issuers do not have Agency Codes assigned to them, which means that they are unable to create unique FASC-N identifiers for the cards they issue. As a result, [PIV-I FAQ] requires the first 14 digits of the FASC-Ns for PIV-I cards (the Agency Code, System Code, and Credential Number) to be populated with all nines.

[15] The sole use of the pairing code is the establishment of a VCI. Its use over the contact interface serves no purpose.

Table 4b. PIV Card Application Key References

Key Reference Value (i.e., Key ID)	PIV Key Type	Administrator	Security Condition for Use	
			Contact	Contactless
'04'	PIV Secure Messaging Key	PIV Card Application Administrator	Always	Always
'9A'	PIV Authentication Key	PIV Card Application Administrator	PIN or OCC	VCI and (PIN or OCC)
'9B'	PIV Card Application Administration Key	PIV Card Application Administrator	Always	Never
'9C'	Digital Signature Key	PIV Card Application Administrator	PIN Always or OCC Always	VCI and (PIN Always or OCC Always)
'9D'	Key Management Key	PIV Card Application Administrator	PIN or OCC	VCI and (PIN or OCC)
'9E'	Card Authentication Key[16]	PIV Card Application Administrator	Always	Always
'82', '83', '84', '85', '86', '87', '88', '89', '8A', '8B', '8C', '8D', '8E', '8F', '90', '91', '92', '93', '94', '95'	Retired Key Management Key	PIV Card Application Administrator	PIN or OCC	VCI and (PIN or OCC)

Secure Messaging (SM) is defined in Section 5.4 and VCI is defined in Section 5.5. Table 2 of Part 2 specifies the security conditions for each command.

When represented as a byte, the key reference occupies bits b8 and b5-b1, while b7 and b6 shall be set to 0. If b8 is 0 then the key reference names global reference data. If b8 is 1, then the key reference names application-specific reference data.

The access control rules for PIV data object access shall reference the PIV Card Application PIN and may optionally reference the cardholder Global PIN or OCC data. If the Global PIN is used by the PIV Card Application then the Global PIN format shall follow the PIV Card Application PIN format defined in Section 2.4.3 of Part 2.

PIV Card Applications with the Discovery Object and Bit 6 of the first byte of the PIN Usage Policy value set to one, as per Section 3.3.2, shall reference the PIV Card Application PIN as well as the cardholder Global PIN in the access control rules for PIV data object access. Additionally, the PIV

[16] A card may optionally have a symmetric CAK in addition to the mandatory asymmetric CAK, in which case both keys would share the same key reference and access control rules.

Card Application card commands can change the status of the Global PIN, and may change its reference data while the PIV Card Application is the currently selected application.

Note: The rest of the document uses "PIN" to mean either the PIV Card Application PIN or the Global PIN.

5.1.1 OCC Data

This document does not specify how the biometric reference data and comparison parameters are stored internally on the card. Moreover, the export of the biometric reference data shall not be allowed. Configuration data related to the biometric reference data may be read from the tag 0x7F61 BIT Group Template data object (see Section 3.3.6). Configuration data is defined in Table 7 of [SP800-76].

5.1.2 PIV Secure Messaging Key

If the PIV Card supports secure messaging, the PIV Secure Messaging key shall be generated on the PIV Card and the PIV Card shall not permit exportation of the PIV Secure Messaging key. The cryptographic operations that use the PIV Secure Messaging key shall be available through the contact and contactless interfaces of the PIV Card. The PKI cryptographic function (see Table 4b) is under an "Always" access rule, and thus private key operations (i.e., use of the key to establish session keys for secure messaging) can be performed without access control restrictions.

The PIV Card shall store a corresponding secure messaging CVC to support validation of the public key by the relying party. The format for the secure messaging CVC shall be as specified in Part 2, Section 4.1.5. The public key required to verify the digital signature of the secure messaging CVC shall be provided in a certificate in the Secure Messaging Certificate Signer data object, as specified in Section 3.3.7.

5.1.3 Pairing Code

If the PIV Card supports the virtual contact interface, then it shall implement support for the pairing code. If implemented, the pairing code shall consist of eight decimal digits and it shall be generated at random by the PIV Card Issuer. The results of each random pairing code generation shall be loaded onto at most one PIV Card and cannot be changed by the cardholder. The pairing code value for a PIV Card shall be stored in the Pairing Code Reference Data Container (see Section 3.3.8) on the card and may be printed on the back of the card in an agency-specific text area (Zones 9B or 10B). PIV Card Issuers may choose to provide the pairing code value to the cardholder in another manner, such as printing it on a slip of paper, rather than printing it on the back of the card.[17]

Unlike the PIV Card Application PIN or the Global PIN, there are no restrictions on the caching of the pairing code by client applications. It is recommended that a client application that needs to communicate with a PIV Card over its virtual contact interface obtain the card's pairing code during a registration step, either by asking the cardholder to enter the value or by reading it from the card over the contact interface from the Pairing Code Reference Data Container, and then cache the pairing

[17] While printing the value of the pairing code on the back of the card provides maximum convenience for use by the cardholder and avoids any risk that the cardholder will forget the pairing code, it may create a risk that an attacker could obtain the value of the pairing code by surreptitiously reading it from the back of the card. Departments and agencies will need to make a risk-based decision in determining the method by which they provide cardholders with the values of their pairing codes.

code until the card expires.[18] The client application may then connect to the card and establish a virtual contact interface with it whenever the card is within read-range of the client application's contactless card reader without needing to prompt the cardholder.

5.2 PIV Algorithm Identifier

A PIV algorithm identifier is a one-byte identifier of a cryptographic algorithm. The identifier specifies a cryptographic algorithm and key size. For symmetric cryptographic operations, the algorithm identifier also specifies a mode of operation (i.e., ECB). SP 800-78, Table 6-2 lists the PIV algorithm identifiers for the cryptographic algorithms that may be recognized on the PIV interfaces.

5.3 Cryptographic Mechanism Identifiers

Cryptographic mechanism identifiers are defined in Table 5. These identifiers serve as inputs to the GENERATE ASYMMETRIC KEY PAIR card command and the Part 3 pivGenerateKeyPair() client API function call, which initiates the generation and storage of the asymmetric key pair.

Table 5. Cryptographic Mechanism Identifiers

Cryptographic Mechanism Identifier	Description	Parameter
'07'	RSA 2048	Optional public exponent encoded big-endian
'11'	ECC: Curve P-256	None
'14'	ECC: Curve P-384	None

All other cryptographic mechanism identifier values are reserved for future use.

5.4 Secure Messaging

A PIV Card Application may optionally support secure messaging (SM). When secure messaging is established, the PIV Card Application is authenticated to the relying system and a set of symmetric session keys are established, which are used to provide confidentiality and integrity protection for the card commands that are sent to the card using secure messaging as well as for the responses from the PIV Card.

If implemented, SM for non-card-management operations shall only be established using the PIV Secure Messaging key specified in Table 4b and the SM protocol in accordance with the specifications in Section 4 of Part 2.

5.5 Virtual Contact Interface

The term virtual contact interface (VCI) is used in this document as shorthand for a security condition. As described in access control rules in this document and in Part 2, all non-card-management operations that are allowed over contact interface may be carried out over the contactless interface if the VCI security condition is satisfied. Support for the VCI is optional.

[18] As noted in Section 5.5, the pairing code does not need to be submitted if the Bit 3 of the first byte of the PIN Usage Policy is set to one.

The VCI security condition supports two different configurations for the establishment of the VCI. In the default (and recommended) configuration, the VCI is only established after both secure messaging has been established and the pairing code has been presented to the card using secure messaging. In the non-default configuration, the VCI is established by the establishment of secure messaging, without any further steps.

The VCI security condition is

> (command is submitted over secure messaging) **AND** (the Discovery Object is present) **AND** (Bit 4 of the first byte of the PIN Usage Policy is one) **AND** ((the security status indicator associated with the pairing code is TRUE) **OR** (Bit 3 of the first byte of the PIN Usage Policy is one))

PIV Card Applications that support the VCI shall support the configuration in which Bit 3 of the first byte of the PIN Usage Policy is set to zero (i.e., the configuration in which submission of the pairing code to the PIV Card Application is required to establish the VCI) and may additionally support the configuration in which Bit 3 of the first byte of the PIN Usage Policy is set to one. Card management systems (CMS) shall be configured to set Bit 3 of the first byte of the PIN Usage Policy to zero by default whenever the Discovery Object is present.

Requiring that the pairing code be submitted to the PIV Card Application in order to establish the VCI protects the previously contact-restricted X.509 certificates from skimming[19] and also protects PIN-based card activation from being blocked. While it is recommended that the default configuration of CMSs remain unchanged, the configuration of a CMS may be changed to set Bit 3 of the first byte of the PIN Usage Policy to one (i.e., to configure PIV Cards to establish VCIs without the submission of a pairing code) if the configuration change is approved by the Designated Approving Authority (DAA) and if compensating controls are implemented to ensure personally identifiable information (i.e., name, email address, and organization) cannot be skimmed from the PIV Card when in close proximity when the card is outside of its protective sleeve.

A DAA's decision to approve the issuance of PIV Cards that implement the VCI without requiring the pairing code shall be based on a risk assessment that weighs the perceived benefit against the risk of unauthorized disclosure of cardholder data exposing previously contact-restricted X.509 certificates to skimming. The previously contact-restricted X.509 certificates include information about the cardholder such as name and email address. Compensating controls shall be captured in the appropriate system security plan.[20] Systems that accept external issued PIV Cards shall be able to accept PIV Cards with either VCI configuration.

5.6 Status Words

A Status Word (SW) is a 2-byte value returned by a card command at the card edge. The first byte of a status word is referred to as SW1 and the second byte of a status word is referred to as SW2.

Recognized values of all SW1-SW2 pairs used as return values on the card command interface and their interpretation are given in Table 6. The descriptions of individual card commands provide additional information for interpreting returned status words.

[19] Skimming is when data is surreptitiously obtained from a contactless card, using a hidden reader that powers, commands, and reads from the card within the maximum read distance (reported as about 25 cm with ISO/IEC 14443 smart cards like the PIV Card).
[20] See SP 800-18 Rev1, Guide for Developing Security Plans for Federal Information Systems.

Table 6. Status Words

SW1	SW2	Meaning
'61'	'xx'	Successful execution where SW2 encodes the number of response data bytes still available
'63'	'00'	Verification failed
'63'	'CX'	Verification failed, X indicates the number of further allowed retries or resets
'68'	'82'	Secure messaging not supported
'69'	'82'	Security status not satisfied
'69'	'83'	Authentication method blocked
'69'	'87'	Expected secure messaging data objects are missing
'69'	'88'	Secure messaging data objects are incorrect
'6A'	'80'	Incorrect parameter in command data field
'6A'	'81'	Function not supported
'6A'	'82'	Data object or application not found
'6A'	'84'	Not enough memory
'6A'	'86'	Incorrect parameter in P1 or P2
'6A'	'88'	Referenced data or reference data not found
'90'	'00'	Successful execution

Appendix A––PIV Data Model

The PIV data model number is 0x10, and the data model version number is 0x01.

The SP 800-73-4 specification does not provide mechanisms to read partial contents of a PIV data object. Individual access to the TLV elements within a container is not supported. For each container, compliant cards shall return all TLV elements of the container in the order listed in this appendix.

Both single-chip/dual-interface and dual-chip implementations are feasible. In the single-chip/dual-interface configuration, the PIV Card Application shall be provided the information regarding which interface is in use. In the dual-chip configuration, a separate PIV Card Application shall be loaded on each chip.

Table 7. PIV Data Containers

Container Description	ContainerID	BER-TLV Tag	Container Minimum Capacity (Bytes)[21]	Access Rule for Read		M/O/C
				Contact	Contactless	
Card Capability Container	0xDB00	'5FC107'	287	Always	VCI	M
Card Holder Unique Identifier	0x3000	'5FC102'	2916	Always	Always	M
X.509 Certificate for PIV Authentication (Key Reference '9A')	0x0101	'5FC105'	1905	Always	VCI	M
Cardholder Fingerprints	0x6010	'5FC103'	4006	PIN	VCI and PIN	M
Security Object	0x9000	'5FC106'	1336	Always	VCI	M
Cardholder Facial Image	0x6030	'5FC108'	12710	PIN	VCI and PIN	M
X.509 Certificate for Card Authentication (Key Reference '9E')	0x0500	'5FC101'	1905	Always	Always	M
X.509 Certificate for Digital Signature (Key Reference '9C')	0x0100	'5FC10A'	1905	Always	VCI	C
X.509 Certificate for Key Management (Key Reference '9D')	0x0102	'5FC10B'	1905	Always	VCI	C
Printed Information	0x3001	'5FC109'	245	PIN or OCC	VCI and (PIN or OCC)	O
Discovery Object	0x6050	'7E'	19	Always	Always	O
Key History Object	0x6060	'5FC10C'	128	Always	VCI	O
Retired X.509 Certificate for Key Management 1 (Key reference '82')	0x1001	'5FC10D'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 2 (Key reference '83')	0x1002	'5FC10E'	1905	Always	VCI	O

[21] The values in this column denote the guaranteed minimum capacities, in bytes, of the on-card storage containers. Cards with larger containers may be produced and determined conformant.

Container Description	ContainerID	BER-TLV Tag	Container Minimum Capacity (Bytes)[21]	Access Rule for Read		M/O/C
				Contact	Contactless	
Retired X.509 Certificate for Key Management 3 (Key reference '84')	0x1003	'5FC10F'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 4 (Key reference '85')	0x1004	'5FC110'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 5 (Key reference '86')	0x1005	'5FC111'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 6 (Key reference '87')	0x1006	'5FC112'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 7(Key reference '88')	0x1007	'5FC113'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 8(Key reference '89')	0x1008	'5FC114'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 9 (Key reference '8A')	0x1009	'5FC115'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 10 (Key reference '8B')	0x100A	'5FC116'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 11 (Key reference '8C')	0x100B	'5FC117'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 12 (Key reference '8D')	0x100C	'5FC118'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 13 (Key reference '8E')	0x100D	'5FC119'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 14 (Key reference '8F')	0x100E	'5FC11A'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 15 (Key reference '90')	0x100F	'5FC11B'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 16 (Key reference '91')	0x1010	'5FC11C'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 17 (Key reference '92')	0x1011	'5FC11D'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 18 (Key reference '93')	0x1012	'5FC11E'	1905	Always	VCI	O

Container Description	ContainerID	BER-TLV Tag	Container Minimum Capacity (Bytes)[21]	Access Rule for Read		M/O/C
				Contact	Contactless	
Retired X.509 Certificate for Key Management 19 (Key reference '94')	0x1013	'5FC11F'	1905	Always	VCI	O
Retired X.509 Certificate for Key Management 20 (Key reference '95')	0x1014	'5FC120'	1905	Always	VCI	O
Cardholder Iris Images	0x1015	'5FC121'	7106	PIN	VCI and PIN	O
Biometric Information Templates Group Template	0x1016	'7F61'	65	Always	Always	O
Secure Messaging Certificate Signer	0x1017	'5FC122'	2471	Always	Always	O
Pairing Code Reference Data Container	0x1018	'5FC123'	12	PIN or OCC	VCI and (PIN or OCC)	O

Note that all data elements of the following data objects are mandatory unless specified as optional or conditional.

Table 8. Card Capability Container

Card Capability Container	0xDB00		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Card Identifier	0xF0	Fixed	0 or 21
Capability Container version number	0xF1	Fixed	0 or 1
Capability Grammar version number	0xF2	Fixed	0 or 1
Applications CardURL	0xF3	Variable	128
PKCS#15	0xF4	Fixed	0 or 1
Registered Data Model number	0xF5	Fixed	1
Access Control Rule Table	0xF6	Fixed	0 or 17
Card APDUs	0xF7	Fixed	0
Redirection Tag	0xFA	Fixed	0
Capability Tuples (CTs)	0xFB	Fixed	0
Status Tuples (STs)	0xFC	Fixed	0
Next CCC	0xFD	Fixed	0
Extended Application CardURL (Optional)	0xE3	Fixed	48
Security Object Buffer (Optional)	0xB4	Fixed	48
Error Detection Code	0xFE	LRC	0

Note: The optional Extended Application CardURL and Security Object Buffer data elements are deprecated and will be eliminated in a future version of SP 800-73.

* The values in the "Max. Bytes" columns denote the lengths of the value (V) fields of BER-TLV elements.

Table 9. Card Holder Unique Identifier

Card Holder Unique Identifier	0x3000		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Buffer Length (Optional)	0xEE	Fixed	2
FASC-N	0x30	Fixed	25
Organizational Identifier (Optional)	0x32	Fixed	4
DUNS (Optional)	0x33	Fixed	9
GUID	0x34	Fixed	16
Expiration Date	0x35	Date (YYYYMMDD)	8
Cardholder UUID (Optional)	0x36	Fixed	16
Issuer Asymmetric Signature	0x3E	Variable	2816**
Error Detection Code	0xFE	LRC	0

Note: The optional Buffer Length, Organizational Identifier and DUNS data elements are deprecated and will be eliminated in a future version of SP 800-73.

The Error Detection Code is the same element as the Longitudinal Redundancy Code (LRC) in [TIG SCEPACS]. Because TIG SCEPACS makes the LRC mandatory, it is present in the CHUID. However, this document makes no use of the Error Detection Code, and therefore the length of the TLV value is set to 0 bytes (i.e., no value will be supplied).

Table 10. X.509 Certificate for PIV Authentication

X.509 Certificate for PIV Authentication	0x0101		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 11. Cardholder Fingerprints

Cardholder Fingerprints	0x6010		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Fingerprint I & II	0xBC	Variable	4000****
Error Detection Code	0xFE	LRC	0

* The values in the "Max. Bytes" columns denote the lengths of the value (V) fields of BER-TLV elements.
** Recommended length: The signer certificate may cause the "Max. Bytes" value in the Issuer Asymmetric Signature field to be exceeded.
*** Recommended length. Certificate size can exceed indicated length value.
**** Recommended length. The certificate that signed the Fingerprint I & II data element in the Cardholder Fingerprints data object can either be stored in the CHUID or in the Fingerprint I & II data element itself. If the latter, the "Max. Bytes" value quoted is a recommendation and the signer certificate in CBEFF_SIGNATURE_BLOCK can exceed the "Max. bytes."

Table 12. Security Object

Security Object	0x9000		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Mapping of DG to ContainerID	0xBA	Variable	30
Security Object	0xBB	Variable	1298
Error Detection Code	0xFE	LRC	0

Table 13. Cardholder Facial Image

Cardholder Facial Image	0x6030		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Image for Visual Verification	0xBC	Variable	12704*****
Error Detection Code	0xFE	LRC	0

Table 14. Printed Information

Printed Information	0x3001		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Name	0x01	Text (ASCII)	125
Employee Affiliation	0x02	Text (ASCII)	20
Expiration date	0x04	Date (YYYYMMMDD)	9
Agency Card Serial Number	0x05	Text (ASCII)	20
Issuer Identification	0x06	Fixed Text (ASCII)	15
Organization Affiliation (Line 1) (Optional)	0x07	Text (ASCII)	20
Organization Affiliation (Line 2) (Optional)	0x08	Text (ASCII)	20
Error Detection Code	0xFE	LRC	0

In order to successfully match the printed information for verification on Zone 8F (Employee Affiliation) and Zone 10F (Agency, Department, or Organization) on the face of the card with the printed information stored electronically on the card, agencies should use tags 0x02, 0x07 and 0x08.

Table 15. X.509 Certificate for Digital Signature

X.509 Certificate for Digital Signature	0x0100		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

* The values in the "Max. Bytes" columns denote the lengths of the value (V) fields of BER-TLV elements.

***** Recommended length. The certificate that signed the Image for Visual Verification data element (tag 0xBC) can be stored in the CHUID or in the Image for Visual Verification data element itself. If the latter, the "Max. Bytes" value quoted is a recommendation and the signer certificate in CBEFF_SIGNATURE_BLOCK can exceed the "Max. bytes."

*** Recommended length. Certificate size can exceed indicated length value.

Table 16. X.509 Certificate for Key Management

X.509 Certificate for Key Management	0x0102		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 17. X.509 Certificate for Card Authentication

X.509 Certificate for Card Authentication	0x0500		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 18. Discovery Object

Discovery Object (Tag '7E')	0x6050		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
PIV Card Application AID	0x4F	Fixed	12
PIN Usage Policy	0x5F2F	Fixed	2

Table 19. Key History Object

Key History Object	0x6060		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
keysWithOnCardCerts	0xC1	Fixed	1
keysWithOffCardCerts	0xC2	Fixed	1[22]
offCardCertURL (Conditional)[23]	0xF3	Variable	118
Error Detection Code	0xFE	LRC	0

* The values in the "Max. Bytes" columns denote the lengths of the value (V) fields of BER-TLV elements.
*** Recommended length. Certificate size can exceed indicated length value.
[22] The numeric values indicated in keysWithOnCardCerts and keysWithOffCardCerts are represented as unsigned binary integers.
[23] The offCardCertURL data element shall be present if keysWithOffCardCerts is greater than zero and shall be absent if both keysWithOnCardCerts and keysWithOffCardCerts are zero. The offCardCertURL may be present if keyWithOffCardCerts is zero but keysWithOnCardCerts is greater than zero.

Table 20. Retired X.509 Certificate for Key Management 1

Retired X.509 Certificate for Key Management 1	0x1001		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 21. Retired X.509 Certificate for Key Management 2

Retired X.509 Certificate for Key Management 2	0x1002		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 22. Retired X.509 Certificate for Key Management 3

Retired X.509 Certificate for Key Management 3	0x1003		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 23. Retired X.509 Certificate for Key Management 4

Retired X.509 Certificate for Key Management 4	0x1004		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

* The values in the "Max. Bytes" columns denote the lengths of the value (V) fields of BER-TLV elements.
*** Recommended length. Certificate size can exceed indicated length value.

Table 24. Retired X.509 Certificate for Key Management 5

Retired X.509 Certificate for Key Management 5	0x1005		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 25. Retired X.509 Certificate for Key Management 6

Retired X.509 Certificate for Key Management 6	0x1006		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 26. Retired X.509 Certificate for Key Management 7

Retired X.509 Certificate for Key Management 7	0x1007		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 27. Retired X.509 Certificate for Key Management 8

Retired X.509 Certificate for Key Management 8	0x1008		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

* The values in the "Max. Bytes" columns denote the lengths of the value (V) fields of BER-TLV elements.
*** Recommended length. Certificate size can exceed indicated length value.

Table 28. Retired X.509 Certificate for Key Management 9

Retired X.509 Certificate for Key Management 9	0x1009		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 29. Retired X.509 Certificate for Key Management 10

Retired X.509 Certificate for Key Management 10	0x100A		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 30. Retired X.509 Certificate for Key Management 11

Retired X.509 Certificate for Key Management 11	0x100B		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 31. Retired X.509 Certificate for Key Management 12

Retired X.509 Certificate for Key Management 12	0x100C		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

* The values in the "Max. Bytes" columns denote the lengths of the value (V) fields of BER-TLV elements.
*** Recommended length. Certificate size can exceed indicated length value.

Table 32. Retired X.509 Certificate for Key Management 13

Retired X.509 Certificate for Key Management 13	0x100D		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 33. Retired X.509 Certificate for Key Management 14

Retired X.509 Certificate for Key Management 14	0x100E		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 34. Retired X.509 Certificate for Key Management 15

Retired X.509 Certificate for Key Management 15	0x100F		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 35. Retired X.509 Certificate for Key Management 16

Retired X.509 Certificate for Key Management 16	0x1010		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

* The values in the "Max. Bytes" columns denote the lengths of the value (V) fields of BER-TLV elements.
*** Recommended length. Certificate size can exceed indicated length value.

Table 36. Retired X.509 Certificate for Key Management 17

Retired X.509 Certificate for Key Management 17	0x1011		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 37. Retired X.509 Certificate for Key Management 18

Retired X.509 Certificate for Key Management 18	0x1012		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 38. Retired X.509 Certificate for Key Management 19

Retired X.509 Certificate for Key Management 19	0x1013		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

Table 39. Retired X.509 Certificate for Key Management 20

Retired X.509 Certificate for Key Management 20	0x1014		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Certificate	0x70	Variable	1856***
CertInfo	0x71	Fixed	1
MSCUID (Optional)	0x72	Variable	38
Error Detection Code	0xFE	LRC	0

Note: The optional MSCUID data element is deprecated and will be eliminated in a future version of SP 800-73.

* The values in the "Max. Bytes" columns denote the lengths of the value (V) fields of BER-TLV elements.
*** Recommended length. Certificate size can exceed indicated length value.

The CertInfo byte in the certificate data objects identified in this appendix shall be encoded as follows:

b8	b7	b6	b5	b4	b3	b2	b1
RFU8	RFU7	RFU6	RFU5	RFU4	IsX509	CompressionTypeLsb	CompressionTypeMsb

CompressionTypeMsb shall be 0 if the certificate is encoded in uncompressed form and 1 if the certificate is encoded using GZIP compression.[24] CompressionTypeLsb and IsX509 shall be set to 0 for PIV Card Applications. Thus, for a certificate encoded in uncompressed form CertInfo shall be 0x00, and for a certificate encoded using GZIP compression CertInfo shall be 0x01.

Table 40. Cardholder Iris Images

Cardholder Iris Images	0x1015		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Images for Iris	0xBC	Variable	7100******
Error Detection Code	0xFE	LRC	0

Table 41. Biometric Information Templates Group Template

BIT Group Template (Tag '7F61')	0x1016		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Number of Fingers	0x02	Fixed	1
BIT for first Finger	0x7F60	Variable	28
BIT for second Finger (Optional)	0x7F60	Variable	28

Table 42. Secure Messaging Certificate Signer

Secure Messaging Certificate Signer	0x1017		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
X.509 Certificate for Content Signing	0x70	Variable	1856
CertInfo	0x71	Fixed	1
Intermediate CVC (Conditional)[25]	0x7F21	Variable	601
Error Detection Code	0xFE	LRC	0

The CertInfo byte in the Secure Messaging Certificate Signer data object shall provide information about the X.509 Certificate for Content Signing. The Intermediate CVC, if present, shall be stored in uncompressed form.

[24] GZIP formats are specified in RFC 1951 and RFC 1952.

****** Recommended length. The certificate that signed the Images for Iris data element (tag 0xBC) can be stored in the CHUID or in the Images for Iris data element itself. If the latter, the "Max. Bytes" value quoted is a recommendation and the signer certificate in CBEFF_SIGNATURE_BLOCK can exceed the "Max. bytes."

[25] The Intermediate CVC shall be absent if the X.509 Certificate for Content Signing contains the public key needed to verify the signature on the secure messaging CVC and shall be present otherwise.

Table 43. Pairing Code Reference Data Container

Pairing Code	0x1018		
Data Element (TLV)	**Tag**	**Type**	**Max. Bytes***
Pairing Code	0x99	Fixed Text (ASCII)	8
Error Detection Code	0xFE	LRC	0

Appendix B––PIV Authentication Mechanisms

To provide guidelines on the usage and behavior supported by the PIV Card, PIV authentication mechanisms and application scenarios are described in this section. FIPS 201 describes PIV authentication as "the process of establishing confidence in the identity of the cardholder presenting a PIV Card." The fundamental goal of using the PIV Card is to authenticate the identity of the cardholder to a system or person that is controlling access to a protected resource or facility. This end goal may be reached by various combinations of one or more of the validation steps described below:

Card Validation (CardV) — This is the process of verifying that a PIV Card is authentic (i.e., not a counterfeit card). Card validation mechanisms include:

+ visual inspection of the tamper-proofing and tamper-resistant features of the PIV Card as per Section 4.1.2 of FIPS 201;
+ use of cryptographic challenge-response schemes with symmetric keys; and
+ use of asymmetric authentication schemes to validate private keys embedded within the PIV Card.

Credential Validation (CredV) — This is the process of verifying the various types of credentials (such as visual credentials, CHUID, biometrics, and certificates) held by the PIV Card. Credential validation mechanisms include:

+ visual inspection of PIV Card visual elements (such as the photo, the printed name, and rank, if present);
+ verification of certificates on the PIV Card;
+ verification of signatures on the PIV biometrics and the CHUID;
+ checking the expiration date; and
+ checking the revocation status of the credentials on the PIV Card.

Cardholder Validation (HolderV) — This is the process of establishing that the PIV Card is in the possession of the individual to whom the card has been issued. Classically, identity authentication is achieved using one or more of these factors: a) something you have, b) something you know, and c) something you are. The assurance of the authentication process increases with the number of factors used. In the case of the PIV Card, these three factors translate as follows: a) something you have – possession of a PIV Card, b) something you know – knowledge of the PIN, and c) something you are – the visual characteristics of the cardholder, and the live fingerprint or iris image samples provided by the cardholder. Thus, mechanisms for PIV cardholder validation include:

+ presentation of a PIV Card by the cardholder;
+ matching the visual characteristics of the cardholder with the photo on the PIV Card;
+ matching the PIN provided with the PIN on the PIV Card; and

+ matching the live fingerprint samples provided by the cardholder with the biometric information embedded within the PIV Card.

B.1 Authentication Mechanism Diagrams

This section describes the activities and interactions involved in interoperable usage and authentication of the PIV Card. The authentication mechanisms represent how a relying party will authenticate the cardholder (regardless of which agency issued the card) in order to provide access to its systems or facilities. These activities and interactions are represented in functional authentication mechanism diagrams. These diagrams are not intended to provide syntactical commands or API function names.

Each of the PIV authentication mechanisms described in this section can be broken into a sequence of one or more validation steps where Card, Credential, and Cardholder validation is performed. In the illustrations, the validation steps are marked as CardV, CredV, and HolderV to signify Card, Credential, and Cardholder validation respectively.

Depending on the assurance provided by the actual sequence of validation steps in a given PIV authentication mechanism, relying parties can make appropriate decisions for granting access to protected resources based on a risk analysis.

B.1.1 Authentication Using PIV Biometrics (BIO)

The general authentication mechanism using the PIV biometrics for off-card matching is illustrated in Figure B-1.

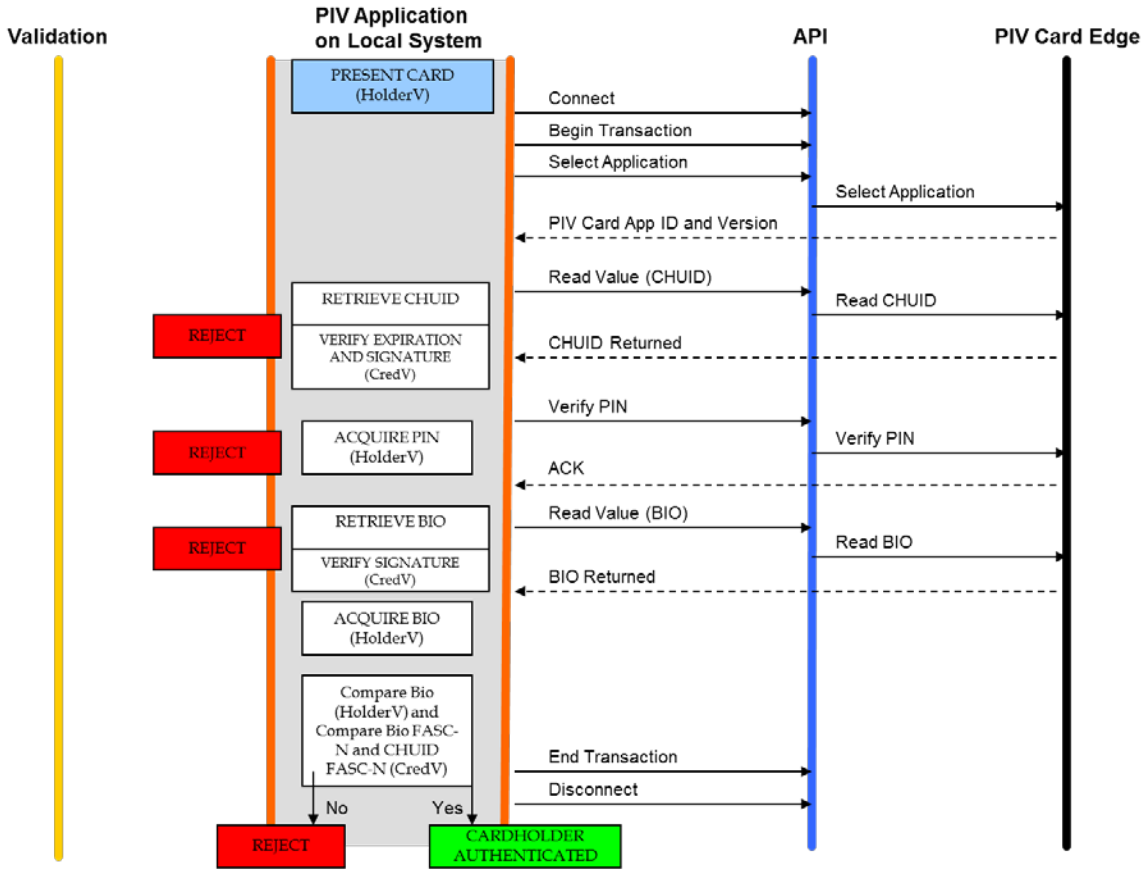

Figure B-1. Authentication using PIV Biometrics (BIO)

The assurance of authentication using the PIV biometric can be further increased if the live biometric sample is collected in an attended environment, with a human overseeing the process. The attended biometric authentication mechanism (BIO-A) is illustrated in Figure B-2.

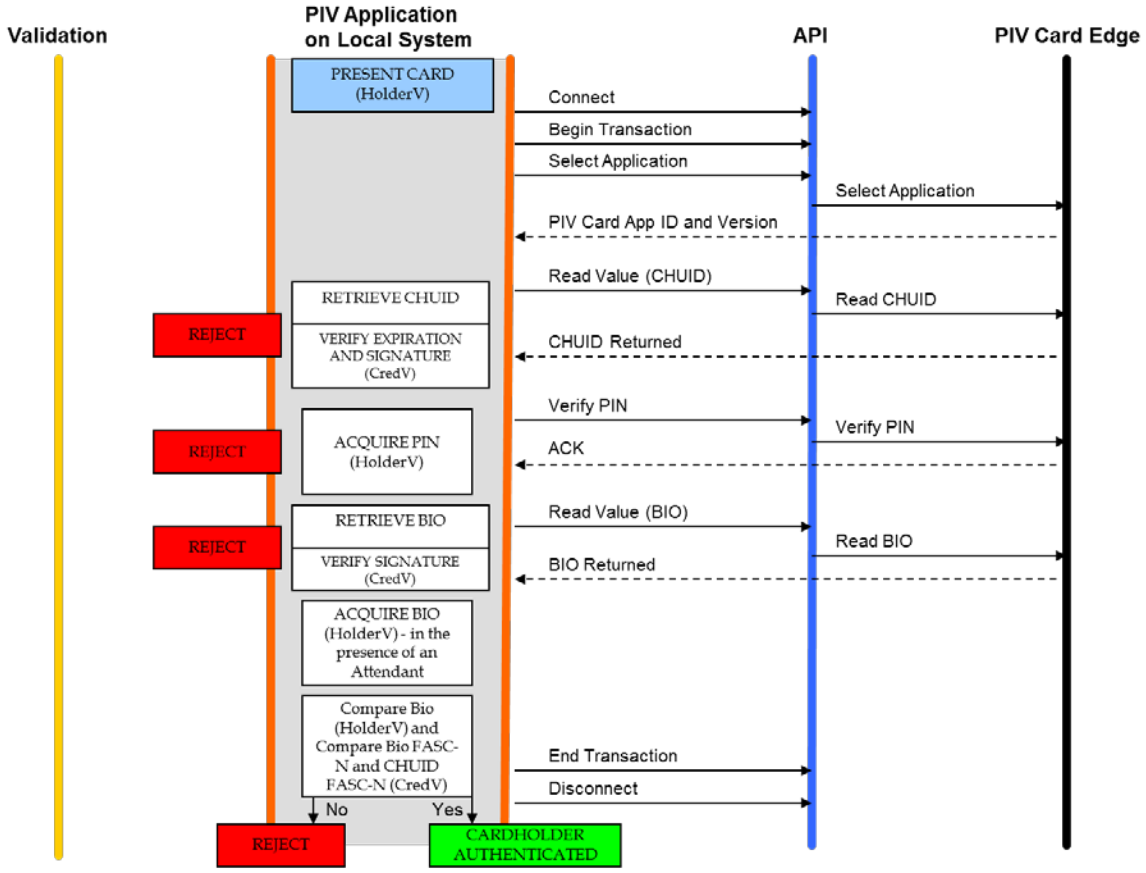

Figure B-2. Authentication using PIV Biometrics Attended (BIO-A)

B.1.2 Authentication Using PIV Authentication Key

The authentication mechanism using the PIV Authentication key is illustrated in Figure B-3.

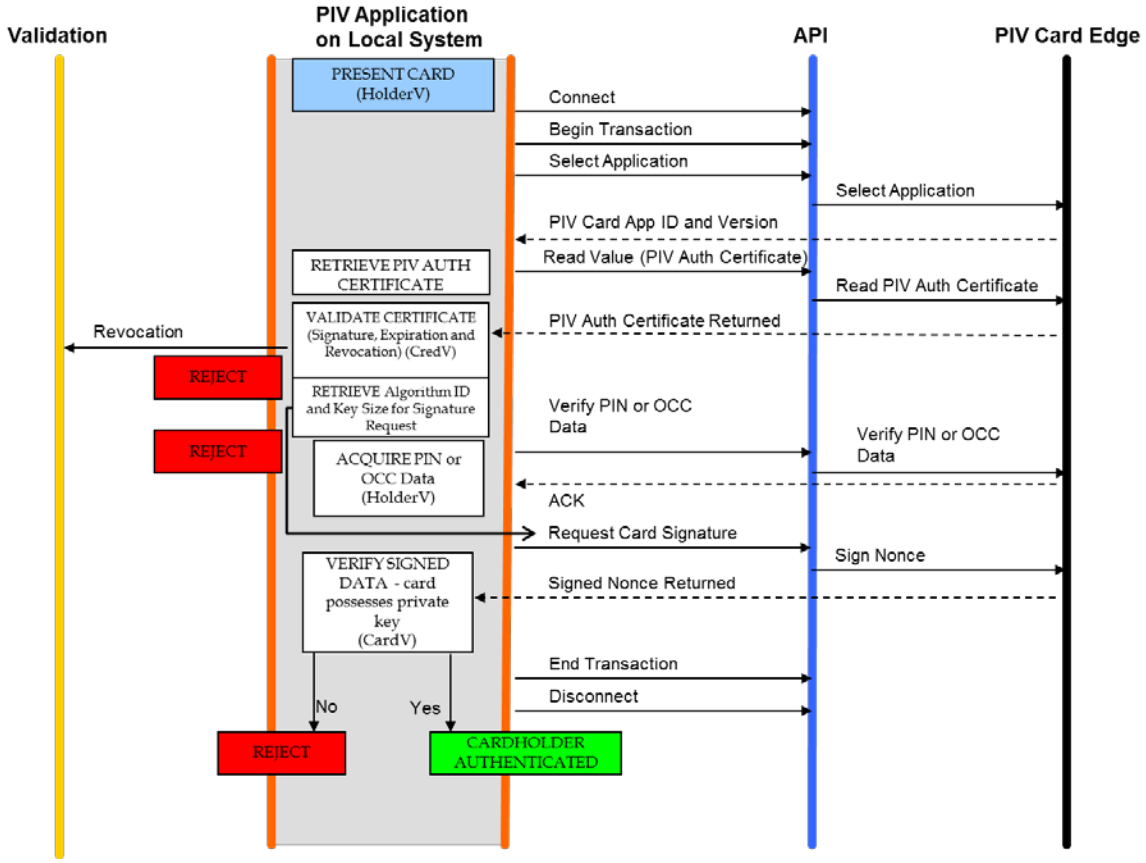

Figure B-3. Authentication using PIV Authentication Key

B.1.3 Authentication Using Card Authentication Key

Authentication mechanisms using the Card Authentication key are illustrated in Figures B-4 and B-5. Figure B-4 illustrates the use of the mandatory asymmetric Card Authentication key, while Figure B-5 uses the optional symmetric Card Authentication key for the authentication mechanism.

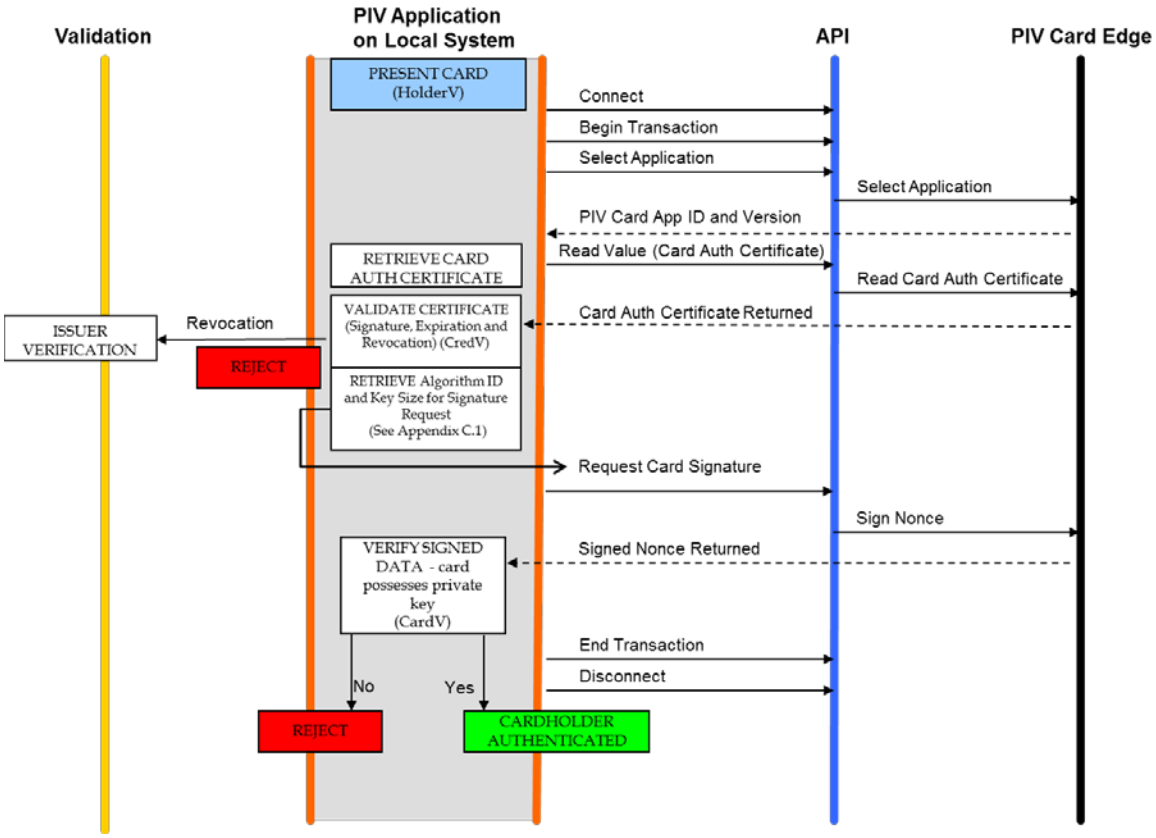

Figure B-4. Authentication using an asymmetric Card Authentication Key

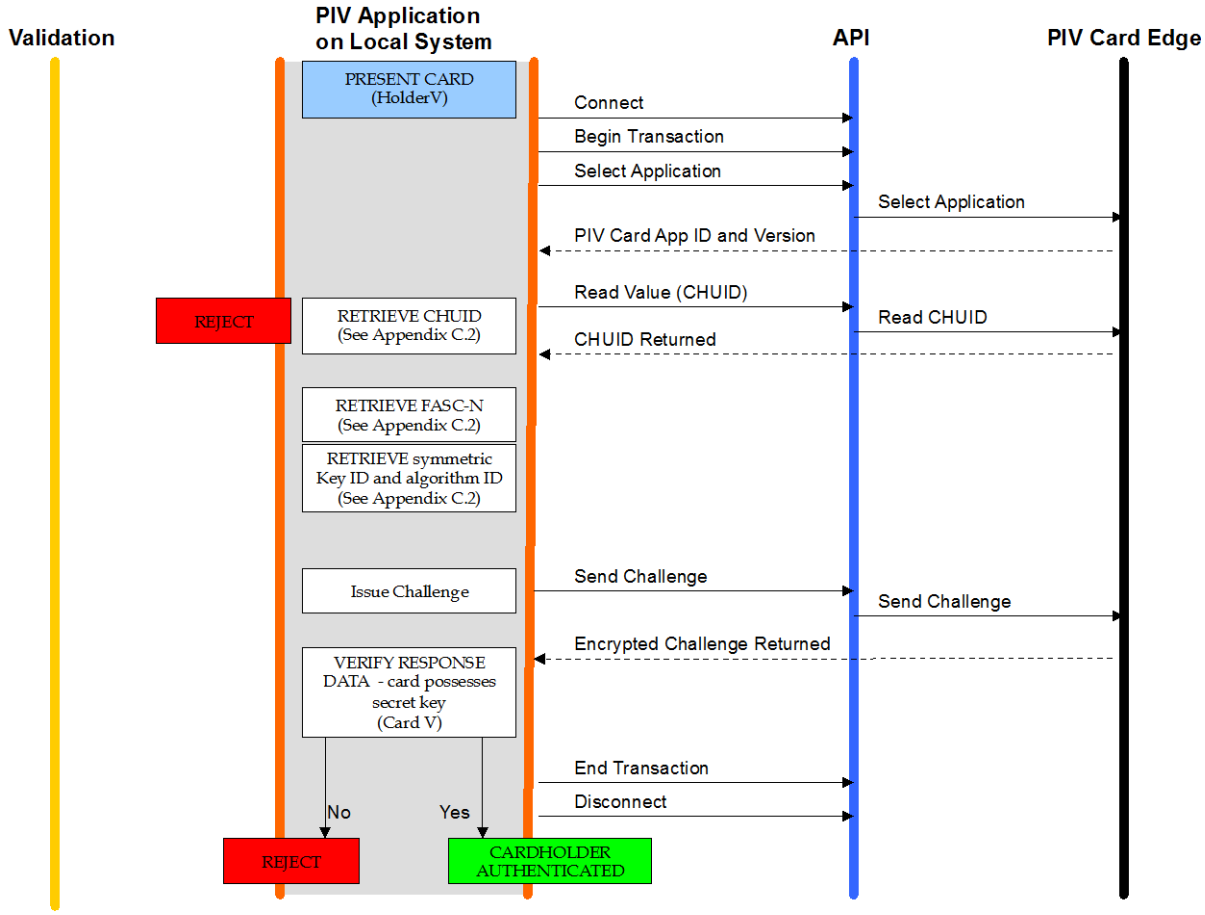

Figure B-5. Authentication using a symmetric Card Authentication Key

B.1.4 Authentication Using OCC (OCC-AUTH)

The OCC-AUTH authentication mechanism is implemented by performing on-card biometric comparison (OCC) over secure messaging. The PIV Application authenticates the PIV Card as part of the process of establishing secure messaging. When the live-scan fingerprint biometric is supplied to the card for OCC over secure messaging, both the request and the response are protected using message authentication codes (MAC), allowing the PIV Application on the local system to verify that the response has not been altered and that it was created by the PIV Card that was authenticated during the establishment of secure messaging.

The OCC-AUTH authentication mechanism is performed by establishing secure messaging as described in Section 4 of Part 2 and then performing the VERIFY command, as illustrated in Figure B-6.

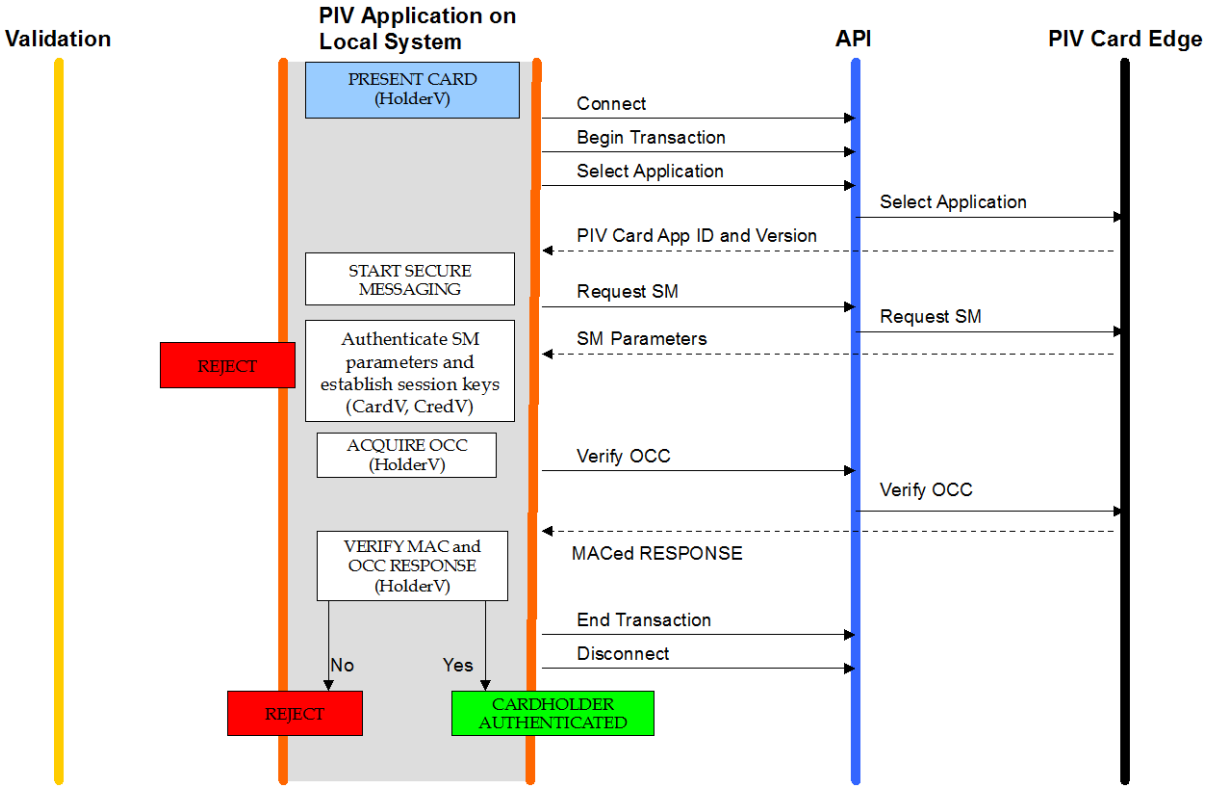

Figure B-6. Authentication using OCC

B.1.5 Authentication Using PIV Visual Credentials

This is the authentication mechanism where a human guard authenticates the cardholder using the visual credentials held by the PIV Card, and is illustrated in Figure B-7.

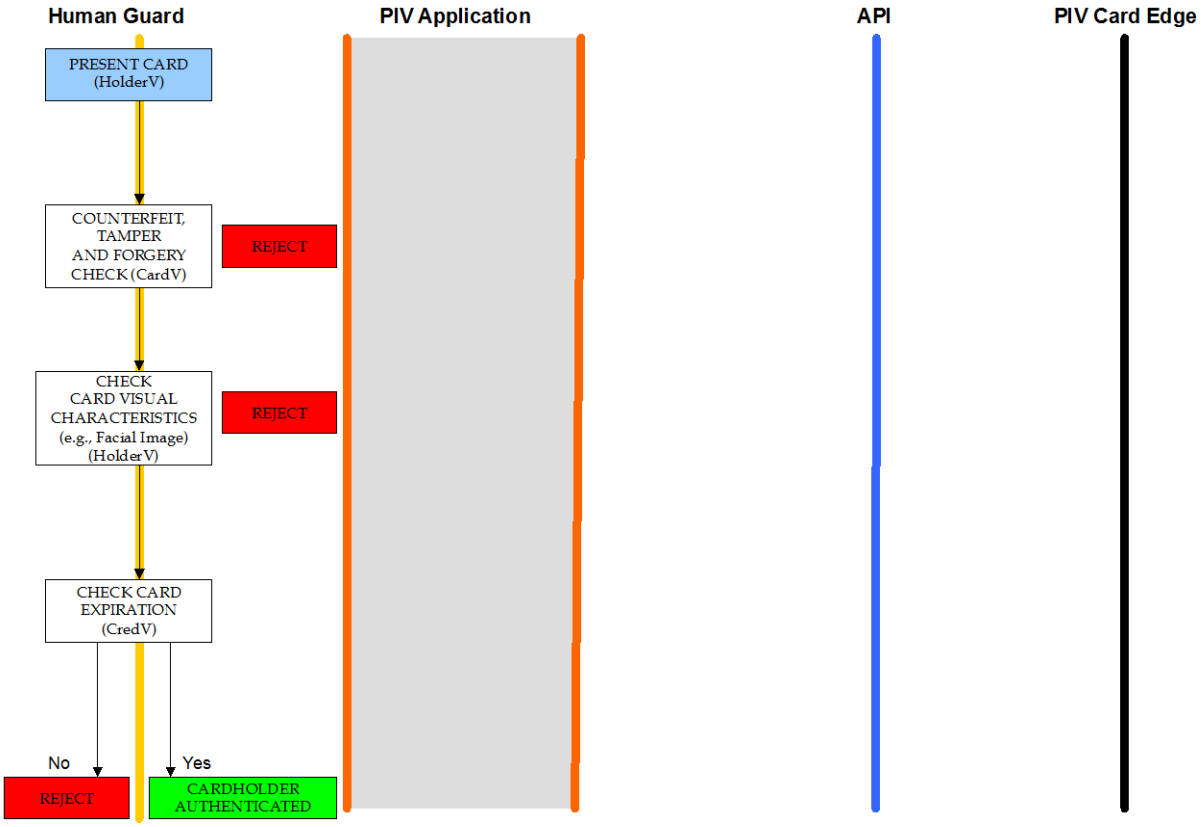

Figure B-7. Authentication using PIV Visual Credentials

B.1.6 Authentication Using PIV CHUID

The PIV CHUID may be used for authentication in several variations. The use of the PIV Card to implement the CHUID authentication mechanism is illustrated in Figure B-8. The minimum set of data that must be transmitted from the PIV Application on the Local System to the host is application dependent and therefore not defined in this Specification.

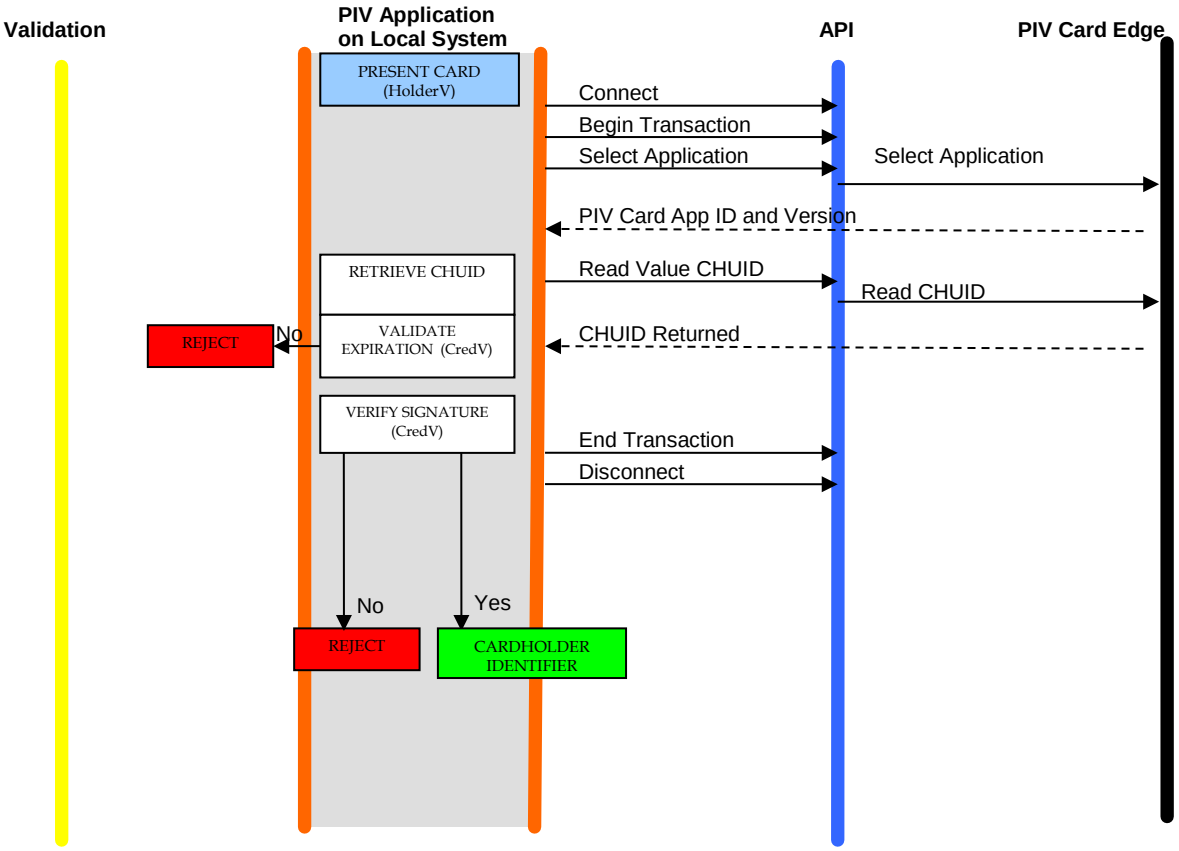

Figure B-8. Authentication using PIV CHUID

B.2 Summary Table

The following table summarizes the types of validation activities that are included in each of the PIV authentication mechanisms described earlier in this section.

Table 44. Summary of PIV Authentication Mechanisms

PIV Authentication Mechanism	Card Validation Steps (CardV)	Credential Validation Steps (CredV)	Cardholder Validation Steps (HolderV)
PIV Biometric		Expiration check CHUID signature check PIV Bio signature check Match CHUID FASC-N with PIV Bio FASC-N	Possession of Card Match PIN provided by Cardholder Match Cardholder bio with PIV bio
PIV Biometric (Attended)		Expiration check CHUID signature check PIV Bio signature check Match CHUID FASC-N with PIV Bio FASC-N	Possession of Card Match PIN provided by Cardholder Match of Cardholder bio to PIV bio *in view of attendant*
PIV Authentication Key	Perform challenge and response with a PIV asymmetric key, and validate signature on response	Certificate validation of a PIV certificate	Possession of Card Match PIN or OCC data provided by Cardholder
Asymmetric Card Authentication Key	Perform challenge and response with a PIV asymmetric Card Authentication key, and validate signature on response	Certificate validation of a PIV certificate	Possession of Card
Symmetric Card Authentication Key	Perform challenge and response with a PIV symmetric key		Possession of Card
On-card Biometric Comparison	Establish Secure Messaging	Certificate validation of a PIV certificate	Possession of Card Match OCC data provided by Cardholder
PIV Visual Authentication	Counterfeit, tamper, and forgery check	Expiration check	Possession of Card Match of card visual characteristics with cardholder
PIV CHUID		Expiration check CHUID signature check	Possession of Card

Appendix C––PIV Algorithm Identifier Discovery

Relying parties interact with many PIV Cards with the same native key type implemented by different key sizes and algorithms.[26] For example, a relying party performing the authentication mechanism described in Appendix B.1.2 (Authentication using the PIV Authentication key) can expect to perform a challenge and response cryptographic authentication with a 2048-bit RSA key or an ECDSA (Curve P-256) key.

This appendix describes recommended procedures for key size and algorithm discovery (PIV algorithm ID discovery) to facilitate cryptographic authentication initiated by the relying party to make appropriate decisions for granting access to logical networks and systems as well as physical access control systems. The discovery procedure is defined in terms of asymmetric and symmetric cryptographic authentication.

C.1 PIV Algorithm Identifier Discovery for Asymmetric Cryptographic Authentication

As illustrated in the authentication mechanisms in Appendix B, an asymmetric cryptographic authentication involves issuing a challenge (request to sign a nonce) to the PIV Card. The relying party issuing the command provides the nonce to be signed, the key reference, and the PIV algorithm identifier as parameters of the command. The nonce is random data generated by the relying party and the key reference is known. The PIV algorithm identifier, on the other hand, is unknown to the relying party and needs to be identified in order to issue the challenge command. The PIV algorithm identifier can be derived from the previous steps of the authentication mechanism. The relying party, prior to issuing the challenge command, retrieved and parsed the X.509 certificate from the card in order to 1) validate the certificate and 2) extract the public key for the pending verification of the signed nonce once returned from the card. It is during the parsing of the X.509 certificate that the PIV algorithm identifier can be identified in two steps:[27]

Step 1: Algorithm Type Discovery:
The X.509 certificate stores the public key in the subjectPublicKeyInfo field. The subjectPublicKeyInfo data structure has an algorithm field, which includes an OID that identifies the public key's algorithm (RSA or ECC) as listed in Table 3-4 of SP 800-78.

Step 2: Key Size Discovery:
If the algorithm type, as determined in Step 1, is ECC then the key size is determined by the elliptic curve on which the key has been generated, which is P-256 for all elliptic curve PIV Authentication keys and Card Authentication keys.

If the algorithm type, as determined in Step 1, is RSA then the key size is determined by the public key's modulus. The public key appears in the subjectPublicKey field of subjectPublicKeyInfo and is encoded as a sequence that includes both the key's modulus and public exponent.

[26] Table 3-1, SP 800-78 lists the various algorithms and key sizes that may be used for each PIV key type.
[27] The PIV algorithm identifiers specify both the key size and the algorithm for the key references. Thus both values have to be discovered in order to derive the PIV algorithm identifier.

As a final step, the discovered X.509 algorithm OID and key size are mapped to the PIV algorithm identifiers as defined in Table 6-2 of SP 800-78. The relying party then proceeds to issue the GENERAL AUTHENTICATE command to the card.

C.2 PIV Algorithm Identifier Discovery for Symmetric Cryptographic Authentication

In the absence of an X.509 certificate, as is the case with symmetric cryptography, the PIV algorithm identifier discovery mechanism has to rely on a lookup table residing at the local system. The table maps a unique card identifier and key reference (inputs) to an associated PIV algorithm identifier (output). The unique identifier supplied by the card may be the Agency Code || System Code || Credential Number of the FASC-N or the Card UUID.

The symmetric Card Authentication key is optional to implement and a relying party has no prior knowledge of the key's existence. The following routine discovers the Card Authentication key's native implementation:

+ Read the CHUID and either extract the Card UUID or extract the Agency Code || System code || Credential Number from the CHUID's FASC-N.

+ Retrieve the PIV algorithm identifier from the local lookup table. If no algorithm identifier is returned, authentication cannot be performed using the optional symmetric Card Authentication key either because the PIV Card does not implement the key or the local system cannot authenticate the response from the card.

C.3 PIV Algorithm Identifier Discovery for Secure Messaging

The Application Property Template, which is included in the response to the SELECT command, optionally includes a tag 0xAC, which indicates what cryptographic algorithms the PIV Card Application supports. 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.

Appendix D––Terms, Acronyms, and Notation

D.1 Terms

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).
Application Identifier	A globally unique identifier of a card application as defined in ISO/IEC 7816-4.
Application Session	The period of time within a card session between when a card application is selected and a different card application is selected or the card session ends.
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.
Client Application	A program running on a computer in communication with a card interface device.
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 key reference is a one-byte identifier that specifies a cryptographic key according to its PIV Key Type. The identifier is part of the cryptographic material used in a cryptographic protocol, such as an authentication or a signing protocol.
MSCUID	An optional legacy identifier included for compatibility with Common Access Card and Government Smart Card Interoperability Specifications.
Object Identifier	A globally unique identifier of a data object as defined in ISO/IEC 8824-2.

Paring Code	An 8 digit code used to establish a relationship between the PIV Card and a device for the purpose of creating the virtual contact interface after secure messaging has been established.
PIV Key Type	The type of a key. The PIV Key Types are 1) PIV Authentication key, 2) Card Authentication key, 3) digital signature key, 4) key management key, 5) retired key management key, 6) PIV Secure Messaging key, and 7) PIV Card Application Administration key.
Relying Party	An entity that relies upon the subscriber's credentials, typically to process a transaction or grant access to information or a system.
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.

D.2 Acronyms

ACR	Access Control Rule
AID	Application Identifier
APDU	Application Protocol Data Unit
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
ASN.1	Abstract Syntax Notation One
BER	Basic Encoding Rules
BIT	Biometric Information Template
CAK	Card Authentication Key
CBEFF	Common Biometric Exchange Formats Framework
CCC	Card Capability Container
CHUID	Card Holder Unique Identifier
CMS	Cryptographic Message Syntax
CVC	Card Verifiable Certificate
DER	Distinguished Encoding Rules
DG	Data Group
DTR	Derived Test Requirement
ECB	Electronic Code Book
ECC	Elliptic Curve Cryptography
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
FASC-N	Federal Agency Smart Credential Number
FIPS	Federal Information Processing Standards
FISMA	Federal Information Security Management Act
GSC-IS	Government Smart Card Interoperability Specification
GUID	Global Unique Identification number

HSPD	Homeland Security Presidential Directive
HTTP	Hypertext Transfer Protocol
ICC	Integrated Circuit Card
IEC	International Electrotechnical Commission
INCITS	InterNational Committee for Information Technology Standards
ISO	International Organization for Standardization
ITL	Information Technology Laboratory
LSB	Least Significant Bit
LRC	Longitudinal Redundancy Code
MAC	Message Authentication Code
MRTD	Machine Readable Travel Document
MSB	Most Significant Bit
NIST	National Institute of Standards and Technology
NPIVP	NIST Personal Identity Verification Program
OCC	On-Card biometric Comparison
OID	Object Identifier
OMB	Office of Management and Budget
PACS	Physical Access Control System
PIN	Personal Identification Number
PI	Person Identifier, a field in the FASC-N
PIV	Personal Identity Verification
PIX	Proprietary Identifier Extension
PKCS	Public-Key Cryptography Standards
PKI	Public Key Infrastructure
PUK	PIN Unblocking Key
RFU	Reserved for Future Use
RID	Registered application provider IDentifier
RSA	Rivest, Shamir, Adleman
SCEPACS	Smart Card Enabled Physical Access Control System
SHA	Secure Hash Algorithm
SP	Special Publication
SM	Secure Messaging
SW1	First byte of a two-byte status word
SW2	Second byte of a two-byte status word
TIG	Technical Implementation Guidance
TLV	Tag-Length-Value
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UUID	Universally Unique Identifier
VCI	Virtual Contact Interface

D.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 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 0x7F61 is the interindustry data object tag for the Biometric Information Templates Group Template.

Appendix E––References

[COMMON] *X.509 Certificate Policy for the U.S. Federal PKI Common Policy Framework*, Version 1.23, May 2014, or as amended. (See http://www.idmanagement.gov/documents/common-policy-framework-certificate-policy)

[FIPS180] Federal Information Processing Standard 180-4, *Secure Hash Standard (SHS)*, March 2012. (See http://csrc.nist.gov)

[FIPS201] Federal Information Processing Standard 201-2, *Personal Identity Verification (PIV) of Federal Employees and Contractors*, August 2013. (See http://dx.doi.org/10.6028/NIST.FIPS.201-2)

[GSC-IS] *Government Smart Card Interoperability Specification, Version 2.1*, NIST Interagency Report 6887 – 2003 Edition, July 16, 2003. (See http://csrc.nist.gov)

[IR7676] NIST Interagency Report 7676, *Maintaining and Using Key History on Personal Identity Verification (PIV) Cards*, June 2010. (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).*

[MRTD] *ICAO 9303 Machine Readable Travel Documents, Part 3: Machine Readable Official Travel Documents, Volume 2: Specifications for Electronically Enabled MRtds with Biometric Identification Capability*, Third Edition – 2008. Published by authority of the Secretary General, International Civil Aviation Organization.

[NISTIR7863] NISTIR 7863, *Cardholder Authentication for the PIV Digital Signature Key*, NIST.

[PIV-I NFI] Personal Identity Verification Interoperability for Non-Federal Issuers, May 2009, or as amended. (See https://cio.gov/wp-content/uploads/downloads/2012/09/PIV_Interoperabillity_Non-Federal_Issuers_May-2009.pdf)

[PIV-I FAQ] Personal Identity Verification Interoperable (PIV-I) Frequently Asked Questions (FAQ), Version 1.0, June 28, 2010, or as amended.
(See https://www.idmanagement.gov/sites/default/files/documents/PIV-I_FAQ.pdf)

[RFC2616] IETF RFC 2616, “Hypertext Transfer Protocol -- HTTP/1.1,” June 1999.
(See http://www.ietf.org/rfc/rfc2616.txt)

[RFC2585] IETF RFC 2585, “Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP,” May 1999. (See http://www.ietf.org/rfc/rfc2585.txt)

[RFC4122] IETF RFC 4122, "A Universally Unique IDentifier (UUID) URN Namespace," July 2005. (See http://www.ietf.org/rfc/rfc4122.txt)

[RFC4530] IETF RFC 4530, "Lightweight Directory Access Protocol (LDAP) entryUUID Operational Attribute," June 2006. (See http://www.ietf.org/rfc/rfc4530.txt)

[RFC5280] IETF RFC 5280, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile," May 2008. (See http://www.ietf.org/rfc/rfc5280.txt)

[RFC5652] IETF RFC 5652, "Cryptographic Message Syntax (CMS)," September 2009. (See http://www.ietf.org/rfc/rfc5652.txt)

[SP800-76] NIST Special Publication 800-76-2, *Biometric Specifications for Personal Identity Verification*, July 2013. (See http://dx.doi.org/10.6028/NIST.SP.800-76-2)

[SP800-78] NIST Special Publication 800-78-4, *Cryptographic Algorithms and Key Sizes for Personal Identity Verification*, May 2015. (See http://csrc.nist.gov)

[SP800-87] NIST Special Publication 800-87 Revision 1, *Codes for Identification of Federal and Federally-Assisted Organizations*, April 2008. (See http://csrc.nist.gov)

[SP800-85A-4] Draft NIST Special Publication 800-85A-4, *PIV Card Application and Middleware Interface Test Guidelines (SP800-73-4 Compliance)*, 2015. (See http://csrc.nist.gov)

[TIG SCEPACS] PACS v2.2, *Technical Implementation Guidance: Smart Card Enabled Physical Access Control Systems*, Version 2.2, The Government Smart Card Interagency Advisory Board's Physical Access Interagency Interoperability Working Group, July 30, 2004. (See https://www.idmanagement.gov/sites/default/files/documents/TIG_SCEPACS_v2.2_0.pdf)

NIST Special Publication 800-73-4

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

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

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.SP.800-73-4

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

NIST Special Publication 800-73-4

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

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

Jason Mohler
Electrosoft Services, Inc.
Reston, Virginia

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.SP.800-73-4

May 2015

U.S. Department of Commerce
Penny Pritzker, Secretary

National Institute of Standards and Technology
Willie May, 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 Modernization 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.

Nothing in this publication should be taken to contradict the standards and guidelines made mandatory and binding on Federal agencies by the Secretary of Commerce under statutory authority. Nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other Federal official. This publication may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright in the United States. Attribution would, however, be appreciated by NIST.

National Institute of Standards and Technology Special Publication 800-73-4
Natl. Inst. Stand. Technol. Spec. Publ. 800-73-4, 61 pages (May 2015)
http://dx.doi.org/10.6028/NIST.SP.800-73-4
CODEN: NSPUE2

Comments on this publication may be submitted to:

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 (David Cooper, Hildegard Ferraiolo, Ketan Mehta, Salvatore Francomacaro, and Ramaswamy Chandramouli 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. Federal Information Processing Standard 201 [FIPS201], Personal Identity Verification (PIV) of Federal Employees and Contractors, 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 an automated process (computer readable and verifiable) or another person (human 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) as well as the other formatting requirements specified in this section.

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 4b, 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 VCI[4] is established when the following security condition is met:

[3] The VERIFY command is only required to support command chaining if the PIV Card Application supports on-card biometric comparison (OCC).
[4] The VCI is explained in further details in Part 1, Section 5.5.

(command is submitted over secure messaging) **AND** (the Discovery Object is present) **AND** (Bit 4 of the first byte of the PIN Usage Policy is one) **AND** ((the security status indicator associated with the pairing code is TRUE) **OR** (Bit 3 of the first byte of the PIN Usage Policy is one)).

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 or 0x2E (but not both) 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.[5]

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

[5] 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 references '96' and '97'), 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 does not contain the Discovery Object as described in Part 1, then no other key reference shall be able to be verified by the PIV Card Application VERIFY command. If the PIV Card Application contains the Discovery Object, then the ability of the PIV Card Application's VERIFY command to verify key references '00', '96', '97', and '98' shall be as specified by the first byte of the Discovery Object's PIN Usage Policy value:

- If Bit 6 is one, then key reference '00' shall be able to be verified by the PIV Card Application VERIFY command.
- If Bit 5 is one, then key references '96' and/or '97', as specified in the Biometric Information Templates Group Template, shall be able to be verified by the PIV Card Application VERIFY command.
- If Bit 4 is one, then key reference '98' shall be able to be verified by the PIV Card Application VERIFY command.

If any key reference value is specified that cannot be verified by the PIV Card Application, then the PIV Card Application shall return the status word '6A 88'.

The VERIFY command may be submitted over the contact interface and, under some conditions, over the contactless interface. Submission of the VERIFY command over the contactless interface always requires the use of secure messaging, and so if the PIV Card Application does not support secure messaging and the VERIFY command is submitted over the contactless interface, then the card command shall fail and the PIV Card Application shall return the status word '6A 81'. If the PIV Card Application supports secure messaging and the VERIFY command is submitted over the contactless interface, then the card command shall fail and the PIV Card Application shall return status word '69 82' if:

- the key reference is '00' or '80' and the command is not submitted over the VCI, or
- the key reference is '96', '97', or '98' and the command is submitted without secure messaging.

The P1 parameter shall be either '00' or 'FF'. If any other value is specified for the P1 parameter, then the PIV Card Application shall return the status word '6A 86'.

If the VERIFY command fails for one of the reasons specified above, then the security status and the retry counter of the key reference shall remain unchanged.

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').

If P1='00', and L_c and the command data field are present, then the authentication data in the command data field shall be compared against the reference data associated with the key reference, as specified in the following subsections. However, if the key reference is '00', '80', '96', or '97' and the current value of the retry counter associated with the key reference is zero, then the PIV Card Application shall return the status word '69 83'.[6] In order to protect against blocking over the contactless interface, PIV Card Applications that implement secure messaging shall define an issuer-specified intermediate retry value for each of these key references and return '69 83' if the command is submitted over the contactless interface (over secure messaging or the VCI, as required for the key reference) and the current value of the retry counter associated with the key reference is at or below the issuer-specified intermediate retry value. If status word '69 83' is returned, then the comparison shall not be made, and the security status and the retry counter of the key reference shall remain unchanged.

If P1='FF', and L_c and the command data field are absent, the command shall reset the security status of the key reference in P2. 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.

3.2.1.1 PIV Card Application PIN and Global PIN

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.[7] 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.

If the authentication data in the command data field is properly formatted (see previous paragraph) 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 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, are issuer dependent.

3.2.1.2 On-Card Biometric Comparison

If the key reference is '96' or '97' 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 the BIT, 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.

[6] There is no retry counter associated with the pairing code, and so the authentication method cannot be blocked for that key reference.
[7] It is recommended that in this case the authentication data not be compared to the on-card reference data.

If the authentication data in the command data field is properly formatted (see previous paragraph) 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 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, are issuer dependent.

3.2.1.3 Pairing Code

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 command shall fail and 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'. If status word '6A 80' is returned, the security status of the key reference shall remain unchanged. If status word '63 00' is returned, the security status of the key reference shall be set to FALSE.

If the card command succeeds then the security status of the key reference shall be set to TRUE.

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 4a.
L_c	Absent[8] – 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

Response Syntax

SW1	SW2	Meaning
'63'	'00'	Verification failed
'63'	'CX'	Verification failed, X indicates the number of further allowed retries
'69'	'82'	Security status not satisfied
'69'	'83'	Authentication method blocked
'6A'	'80'	Incorrect parameter in command data field

[8] 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').

'6A'	'81'	Function not supported
'6A'	'86'	Incorrect parameter in P1
'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 88'. 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 key reference '81' is specified and the command is submitted over the contactless interface (including SM or VCI), then the card command shall fail and the PIV Card Application shall return the status word '6A 81'. If the PIV Card Application does not support secure messaging and the CHANGE REFERENCE DATA command is submitted over the contactless interface, then the card command shall fail and the PIV Card Application shall return the status word '6A 81'. If the PIV Card Application supports secure messaging and the CHANGE REFERENCE DATA command, with key reference '00' or '80', 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 '69 82'. In each case, 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 command is submitted over the contactless interface (VCI) and the current value of the retry counter associated with the key reference is at or below the issuer-specified intermediate retry value (see Section 3.2.1), 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.[9] 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.

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, are 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'	'82'	Security status not satisfied
'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 88'.[10]

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

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'.

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.[11] 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

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

'90'	'00'	Successful execution

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.[12]

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 ('04') and cryptographic algorithm identifier '27' or '2E' to establish session keys for secure messaging as specified in Section 4. If key reference '04' 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 AUTHENTICATE 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 4b, Part 1 for key reference values
L_c	Length of data field
Data Field	See Table 7
L_e	Absent or '00'

[12] 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 '04', '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.[13] 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 $CB_H \| ID_{sH} \| Q_{eH}$	H3	$CB_H \| ID_{sH} \| Q_{eH}$ →		
			$ID_{sICC} = T_8(SHA256(C_{ICC}))$	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
			$SK_{CFRM} \| SK_{MAC} \| SK_{ENC} \| SK_{RMAC} = KDF(Z, len, OtherInfo)$	C7
			Zeroize Z	C8
			$AuthCryptogram_{ICC} = CMAC(SK_{CRFM}, \text{“KC\_1\_V”} \| ID_{sICC} \| ID_{sH} \| Q_{eH})$	C9
		← $CB_{ICC} \| N_{ICC} \| AuthCryptogram_{ICC} \| C_{ICC}$	Zeroize SK_{CFRM}	C10
			Return $CB_{ICC} \| N_{ICC} \| AuthCryptogram_{ICC} \| C_{ICC}$	C11
Check that CB_{ICC} is 0x00	H4			
Verify C_{ICC} signature and subject public key	H5			
$ID_{sICC} = T_8(SHA256(C_{ICC}))$	H6			
Extract Q_{sICC} from C_{ICC}	H7			
$Z = ECC\_CDH(d_{eH}, Q_{sICC})$	H8			
Zeroize d_{eH}	H9			
$SK_{CFRM} \| SK_{MAC} \| SK_{ENC} \| SK_{RMAC} = KDF(Z, len, OtherInfo)$	H10			
Zeroize Z	H11			
Check that $AuthCryptogram_{ICC}$ equals $CMAC(SK_{CFRM}, \text{“KC\_1\_V”} \| ID_{sICC} \| ID_{sH} \| Q_{eH})$	H12			
Zeroize SK_{CFRM}	H13			

[13] 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.
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 partial 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 $CB_H \| ID_{sH} \| Q_{eH}$	
Wait for response from PIV Card Application: $CB_{ICC} \| N_{ICC} \| AuthCryptogram_{ICC} \| C_{ICC}$		
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	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}.[14,15] 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.
H6	$ID_{sICC} = T_8(SHA256(C_{ICC}))$	ID_{sICC}, the left-most 8 bytes of the SHA-256 hash of C_{ICC}, is used as an input for session key derivation.
H7	Extract Q_{sICC} from C_{ICC}	
H8	$Z = ECC\_CDH(d_{eH}, Q_{sICC})$	Compute the shared secret, Z, using the ECC CDH primitive [SP800-56A, Section 5.7.1.2].
H9	Zeroize d_{eH}	Destroy the ephemeral private key generated in Step H2.
H10	$SK_{CFRM} \| SK_{MAC} \| SK_{ENC} \| SK_{RMAC} = KDF(Z, len, OtherInfo)$	Compute the key confirmation key and the session keys. See Section 4.1.6.
H11	Zeroize Z	Destroy the shared secret generated in Step H8.
H12	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.
H13	Zeroize SK_{CFRM}	Destroy the key confirmation key derived in Step H10.

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. (Note that ID_{sICC} is static, and so may be pre-computed off card.)

[14] 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.
[15] 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
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	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],[16] 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].
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 $CB_{ICC} \| N_{ICC} \| AuthCryptogram_{ICC} \| C_{ICC}$	

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

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	
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
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

	128 bit channel strength	192 bit channel strength
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 15 and 16, 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 partial 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				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 C_{ICC}.[17]
	0x5F20		16	Subject Identifier	GUID (Card UUID)
	0x7F49		Variable	CardHolderPublicKey Data Object	
		0x06	Variable	Algorithm OID	Possible values are: ▪ 0x2A8648CE3D030107 for ECDH (Curve P-256) or ▪ 0x2B81040022 for ECDH (Curve

[17] 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.

Tag	Tag	Tag	Length	Name	Value
					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
					}

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 || X || Y, where X and Y

Tag	Tag	Tag	Length	Name	Value
					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 partial 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 H10 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	$0x04 \| 0x0D \| 0x0D \| 0x0D \| 0x0D \| 0x08 \| ID_{sH} \| 0x01 \| CB_H \| 0x10 \| T_{16}(Q_{eH}) \| 0x08 \| ID_{sICC} \| 0x18 \| N_{ICC} \| 0x01 \| CB_{ICC}$

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. The first 16 bytes from this byte string are included in *OtherInfo*.

4.1.7 Key Confirmation

Key confirmation shall be performed in Steps C9 and H12 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}. "KC_1_V" is a 6-byte ASCII string ('4B 43 5F 31 5F 56'). 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	'04' (PIV Secure Messaging key).
L_c	Length of data field
Data Field	'7C' L1 { '81' L2 { 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	'7C' L1 { '82' L2 { CB_{ICC} || N_{ICC} || $AuthCryptogram_{ICC}$ || 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.

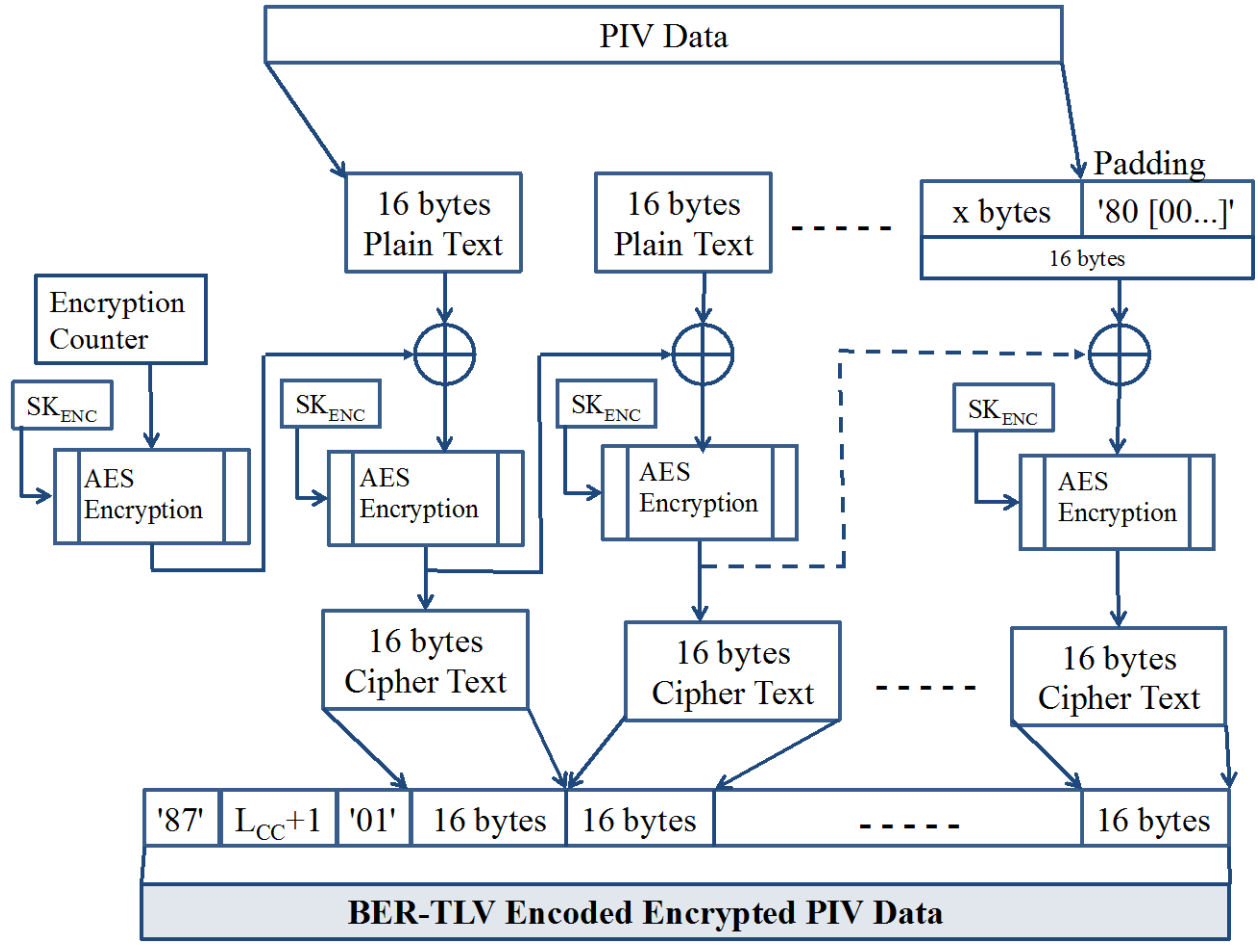

Figure 1. PIV Data Confidentiality

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 APDU sent over secure messaging (except for the GET RESPONSE command and APDUs with a CLA of '1C'), 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: Prior to encryption, one to sixteen bytes of padding data shall be appended to the PIV data. The padding shall be '80' followed by the number of zeros needed to make the total length of the message to be encrypted (PIV data plus padding) a multiple of sixteen bytes. The first byte of the value field of tag '87', the padding-content indicator byte, shall be '01' to indicate that padding has been applied.

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'), 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.[18]

4. L_e encapsulated in BER-TLV format with tag '97', if the L_e field is included in the command.[19]

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.

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

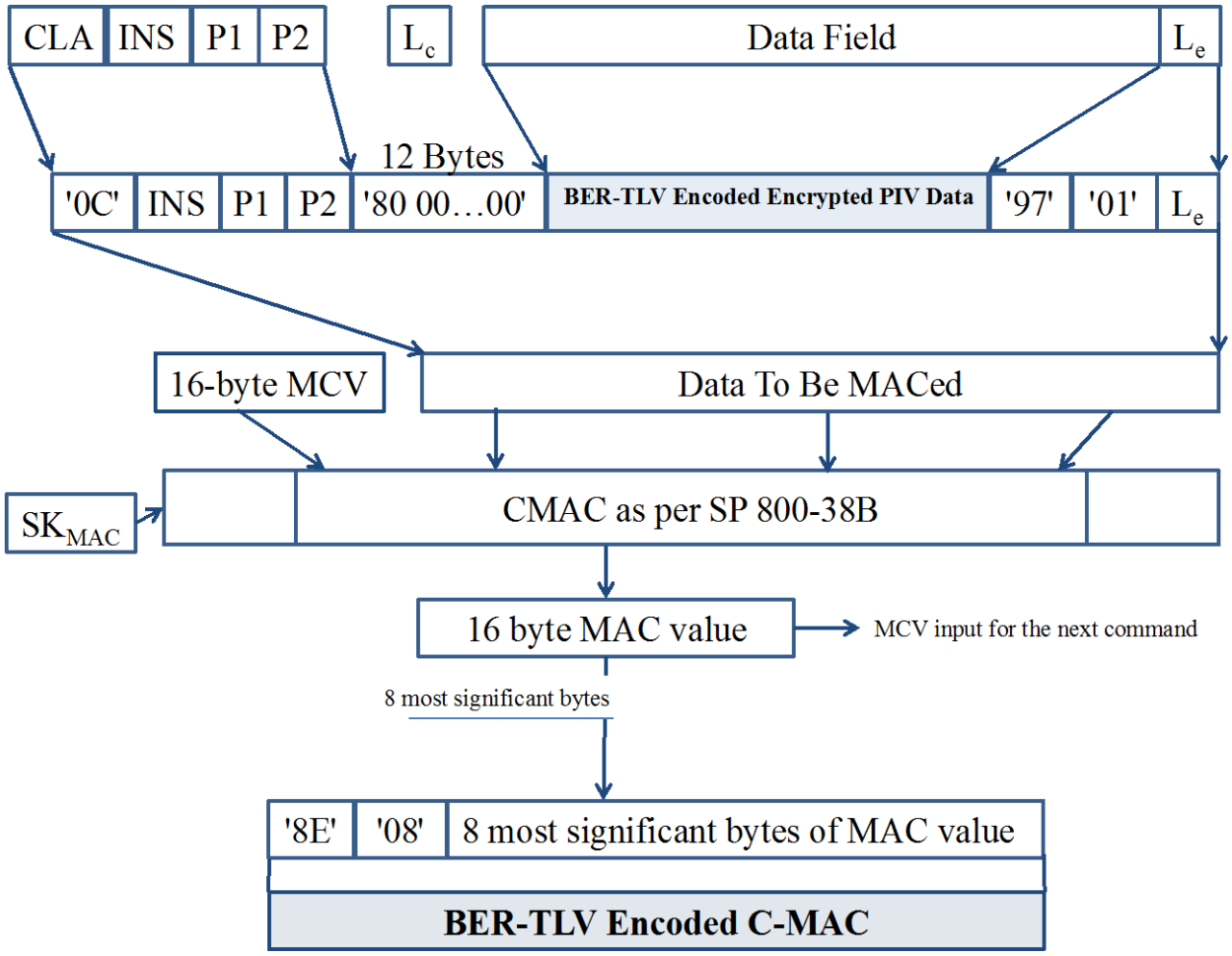

Figure 2. PIV Data Integrity of Command

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;[20] 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 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'.[21]

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').

[20] The data field may be absent in the case of the VERIFY command.
[21] 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 word and response MAC (R-MAC).

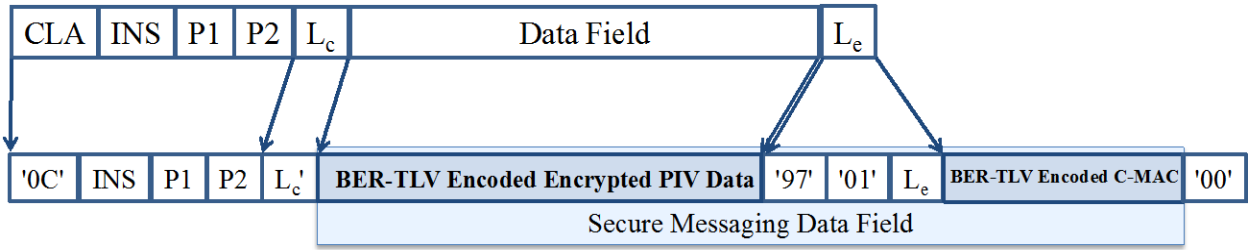

Figure 3. Single Command under Secure Messaging

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.

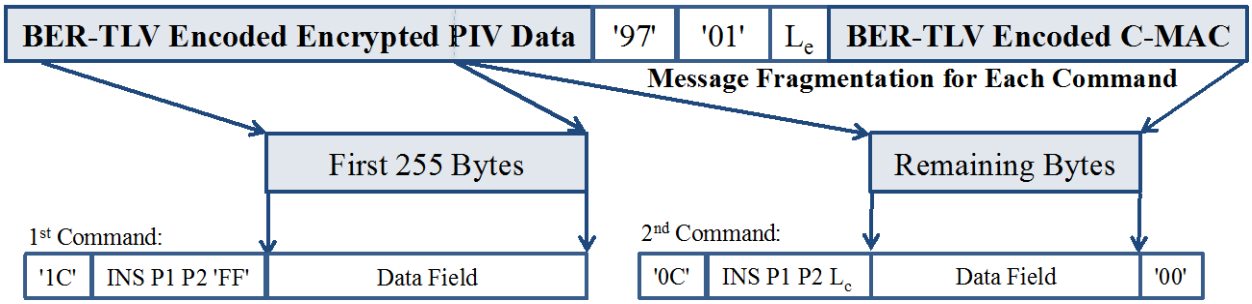

Figure 4. Chained Command under Secure Messaging

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:

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 word, SW1 and SW2, encapsulated in BER-TLV format with tag '99'.

Let $T_{R\text{-}MAC}$ = 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.

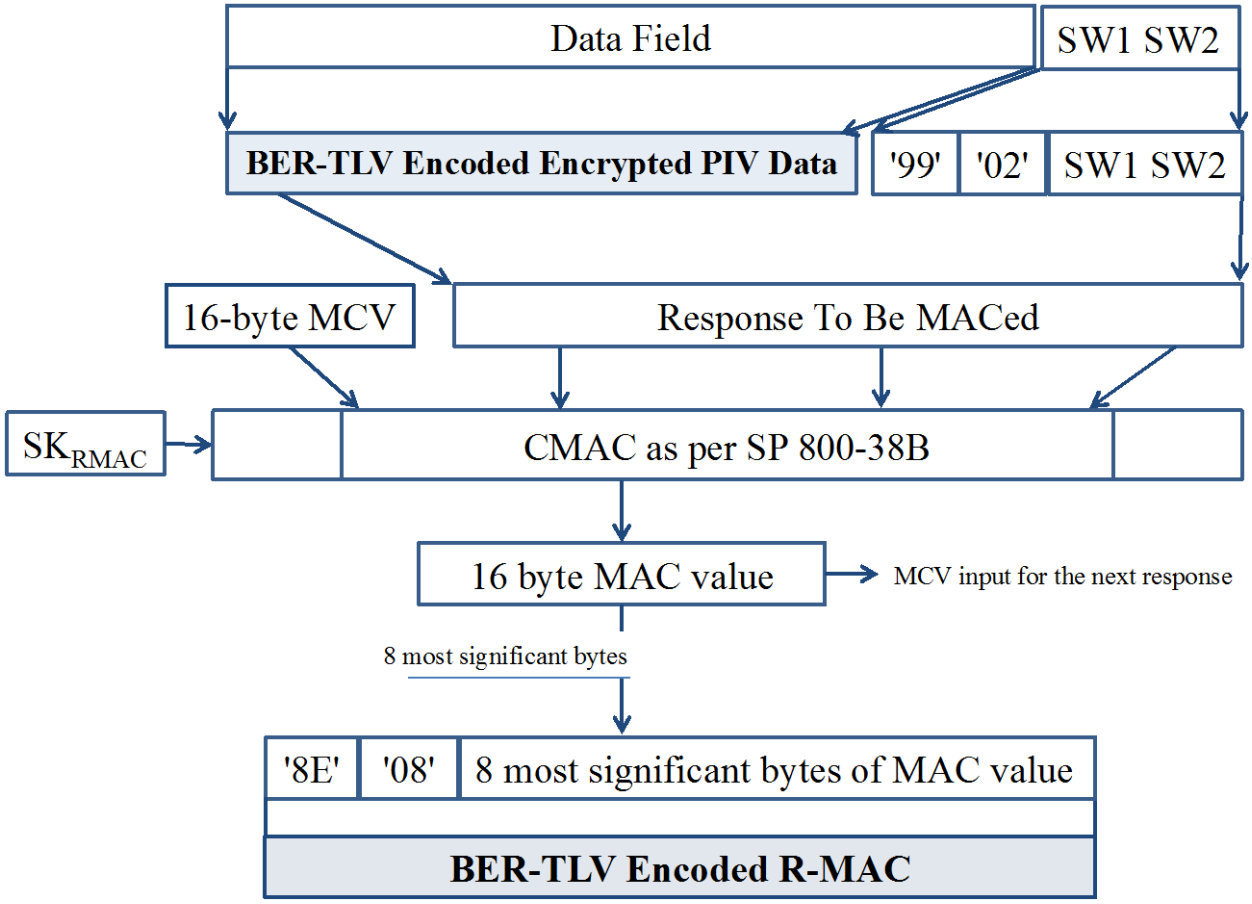

Figure 5. PIV Data Integrity of Response

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 status word, 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 processing status ('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.

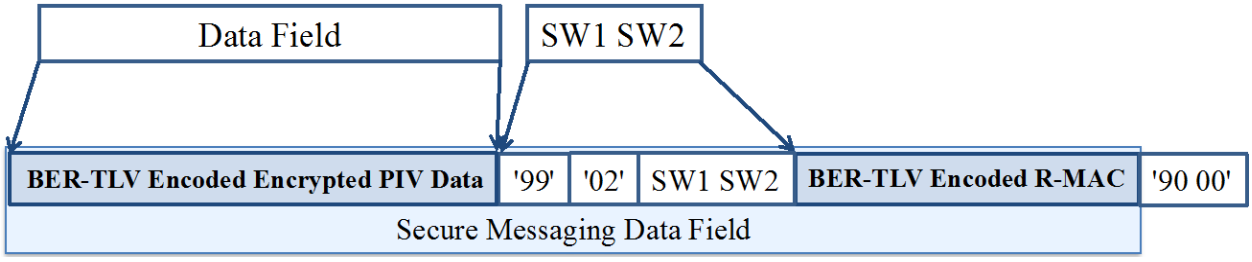

Figure 6. Single Response under Secure Messaging

If the secure messaging data field to be transported is larger than 256 bytes, response chaining[22] 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.

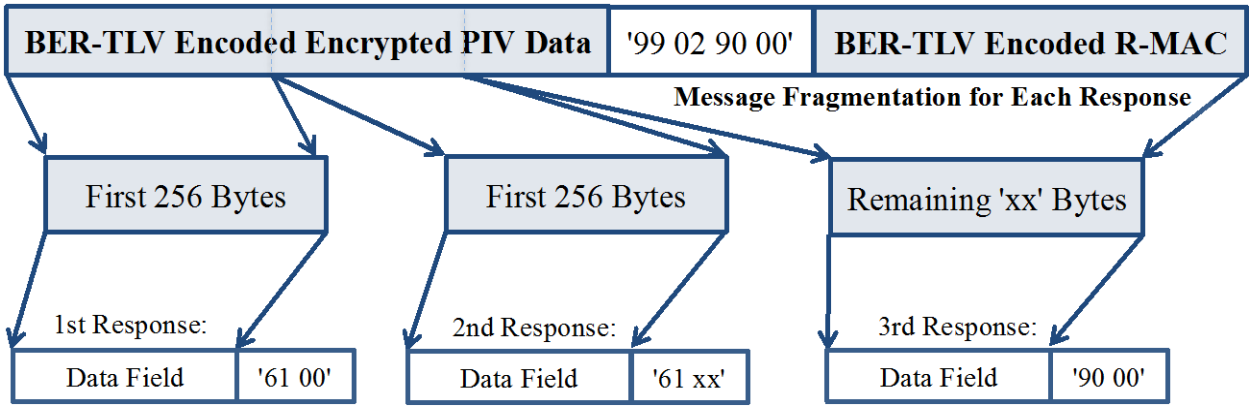

Figure 7. Chained Response under Secure Messaging

4.2.7 Error Handling

The SW processing status is the status word 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
+ '69 82' – Security status not satisfied[23]
+ '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 status words 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;[24] or

[22] The response chaining is accomplished by issuing several GET RESPONSE commands to the card.
[23] Status word '69 82' is used when secure messaging is requested, but no session keys have been established.

+ 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.

[24] An error has occurred in secure messaging if the SW processing status in the response to a command sent with secure messaging is other than '61 XX' or '90 00'.

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 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 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 the virtual contact interface, 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.[25]

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.

[25] As noted in Part 1, Section 5.5, the pairing code does not need to be submitted if the Bit 3 of the first byte of the PIN Usage Policy is set to one.

Command	Response	Comment
00 87 27 04 50 7C 4E 81 4A 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.
	7C L1 82 L2 00 N_{ICC} $AuthCryptogram_{ICC}$ C_{ICC}	The response for the key establishment protocol, as specified in Section 4.1.8, where N_{ICC} and $AuthCryptogram_{ICC}$ are 16 bytes each, and C_{ICC} is as specified in Section 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 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 applied.
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 $T_8(T_{R\text{-}MAC,2})$ 90 00	Return the final 163 bytes of the response, including the BER-TLV encoded status word 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.		
0C 20 00 80 0A 8E 08 $T_8(T_{C\text{-}MAC,3})$ 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 $T_8(T_{R\text{-}MAC,3})$ 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_{C4} 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 04'), and $T_{C\text{-}MAC,4}$ is computed using $T_{C\text{-}MAC,3}$ as the MCV. [Note that the encryption counter used to generate the IV was incremented as a result of the previous VERIFY command even though no encryption was performed for that command.] For the response, $T_{R\text{-}MAC,4}$ is computed using $T_{R\text{-}MAC,3}$ as the MCV.		
0C 20 00 80 1D 87 11 01 ENC_{C4} 8E 08 $T_8(T_{C\text{-}MAC,4})$ 00		The VERIFY command is used to submit the PIV Card Application PIN to the card.
	99 02 90 00 8E 08 $T_8(T_{R\text{-}MAC,4})$ 90 00	The card responds that the command has been successfully executed.
Now that a virtual contact interface has been established and the PIV Card Application PIN has been		

Command	Response	Comment
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 05'), 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 along with the padding bytes using an IV of AES(SK_{ENC}, '80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 05'), 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.		
1C 87 07 9A FF 87 82 01 11 01 <bytes 1 – 250 of ENC_{C5}>		The GENERAL AUTHENTICATE command is used to send a challenge to the PIV Card. This command includes the first part of the challenge.
	90 00	PIV Card Application indicates that it received the first part of the command successfully.
0C 87 07 9A 23 <bytes 251 – 272 of ENC_{C5}> 97 01 00 8E 08 $T_8(T_{C\text{-}MAC,5})$ 00		The remaining challenge data is sent, including the BER-TLV encoded L_e and the BER-TLV encoded C-MAC.
	87 82 01 11 01 <bytes 1 – 251 of ENC_{R5}> 61 1B	PIV Card Application sends first part of the result of signing the challenge. The padding indicator is '01' to indicate that padding was applied.
00 C0 00 00 1B		The remaining portion of response is requested.
	<bytes 252 – 272 of ENC_{R5}> 99 02 90 00 8E 08 $T_8(T_{R\text{-}MAC,5})$ 90 00	PIV Card Application sends final portion of the result of signing the challenge, along with the BER-TLV encoded status word and R-MAC.

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://dx.doi.org/10.6028/NIST.FIPS.201-2)

[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://dx.doi.org/10.6028/NIST.SP.800-56Ar2)

[SP800-76] NIST Special Publication 800-76-2, *Biometric Specifications for Personal Identity Verification*, July 2013. (See http://dx.doi.org/10.6028/NIST.SP.800-76-2)

[SP800-78] NIST Special Publication 800-78-4, Cryptographic *Algorithms and Key Sizes for Personal Identity Verification*, May 2015. (See http://csrc.nist.gov)

NIST Special Publication 800-73-4

Interfaces for Personal Identity Verification – Part 3: PIV Client Application Programming Interface

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

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.SP.800-73-4

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

NIST Special Publication 800-73-4

Interfaces for Personal Identity Verification – Part 3: PIV Client Application Programming Interface

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

Jason Mohler
Electrosoft Services, Inc.
Reston, Virginia

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.SP.800-73-4

May 2015

U.S. Department of Commerce
Penny Pritzker, Secretary

National Institute of Standards and Technology
Willie May, 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 Modernization 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*.

Nothing in this publication should be taken to contradict the standards and guidelines made mandatory and binding on Federal agencies by the Secretary of Commerce under statutory authority. Nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other Federal official. This publication may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright in the United States. Attribution would, however, be appreciated by NIST.

National Institute of Standards and Technology Special Publication 800-73-4
Natl. Inst. Stand. Technol. Spec. Publ. 800-73-4, 22 pages (May 2015)
http://dx.doi.org/10.6028/NIST.SP.800-73-4
CODEN: NSPUE2

Comments on this publication may be submitted to:

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 (David Cooper, Hildegard Ferraiolo, Ketan Mehta, Salvatore Francomacaro and Ramaswamy Chandramouli 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.

Table of Contents

List of Appendices

List of Tables

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. Federal Information Processing Standard 201 [FIPS201], Personal Identity Verification (PIV) of Federal Employees and Contractors, 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 (API). 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 API and card command interface for use with the PIV Card.

This part, SP 800-73-4 Part 3: *PIV Client Application Programming Interface*, contains technical specifications of the PIV client application programming interface to the PIV Card.

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, which details the revision history of SP800-73, Section II, which contains configuration management recommendations, and Section III, which specifies NPIVP conformance testing procedures.

[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 an automated process (computer readable and verifiable) or by another person (human 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 3:

+ Section 1, *Introduction*, provides the purpose, scope, audience and assumptions of the document and outlines its structure.

+ Section 2, *Overview: Concepts and Constructs*, describes both the PIV Card Application and the PIV client API. This section is *informative*.

+ Section 3, *Client Application Programming Interface*, describes the set of entry points accessible by client applications through the PIV Middleware to interact with the PIV Card.

+ Appendix A, *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 B, *References*, contains the list 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.

3. Client Application Programming Interface

Table 1 lists the entry points on the PIV client API. This section references object identifiers (OIDs), which are defined and can be found in Part 1 (Table 3).

Table 1. Entry Points on PIV Client Application Programming Interface

Type	Name
Entry Points for Communication	**pivMiddlewareVersion**
	pivConnect
	pivDisconnect
Entry Points for Data Access	**pivSelectCardApplication**
	pivEstablishSecureMessaging
	pivLogIntoCardApplication
	pivGetData
	pivLogoutOfCardApplication
Entry Points for Cryptographic Operations	**pivCrypt**
Entry Points for Credential Initialization and Administration	**pivPutData**
	pivGenerateKeyPair

If both the PIV Middleware and the PIV Card support secure messaging, then all non-card-management functionality[2] of the PIV Card may be accessed over either the contact or contactless interface of the card. In order to perform non-card-management functionality that would otherwise be limited to the contact interface, the client application must first establish a virtual contact interface by calling the pivEstablishSecureMessaging function and then using the pivLogIntoCardApplication function to submit the pairing code to the card.[3] If the client application does not have another means of determining whether communication with the PIV Card is over a contact or contactless interface, it may determine this by using the pivGetData function to attempt to read a mandatory data object, such as the X.509 Certificate for PIV Authentication or the Security Object, that has an access rule for read of "Always," but that is only accessible over the contact and virtual contact interfaces (see Part 1, Table 2). If the return code from

[2] Only the pivPutData and pivGenerateKeyPair API functions perform card-management functionality.
[3] As noted in Part 1, Section 5.5, the pairing code does not need to be submitted if the Bit 3 of the first byte of the PIN Usage Policy is set to one.

pivGetData is `PIV_SECURITY_CONDITIONS_NOT_SATISFIED` this indicates that communication with the card is over a contactless interface.

3.1 Entry Points for Communication

3.1.1 pivMiddlewareVersion

Purpose: Returns the PIV Middleware version string

Prototype:
```
status_word pivMiddlewareVersion(
   OUT   version           versionString
);
```

Parameter: **versionString**

+ For SP 800-73-4 Part 3 conformant PIV Middleware, the parameter returns "800-73-4 Client API" or "800-73-4 Client API with SM".
+ For SP 800-73-3 Part 3 conformant PIV Middleware, the parameter returns "800-73-3 Client API".
+ For SP 800-73-2 Part 3 conformant PIV Middleware, the parameter returns **"**800-73-2 Client API"**.**
+ For SP 800-73-1 conformant PIV Middleware, the pivMiddlewareVersion client API function is not supported. Therefore, a client application invoking the pivMiddlewareVersion function should expect a "function-not-supported" error from a SP 800-73-1 conformant PIV Middleware. For purposes of version determination, failure to obtain a specific version from pivMiddlewareVersion shall be considered equivalent to obtaining a response of "800-73-1 Client API".

Return Codes: `PIV_OK`

PIV Middleware that returns a versionString of "800-73-4 Client API with SM" shall implement all PIV Middleware functions listed in Table 1 and be able to recognize and process all mandatory and optional PIV data objects. PIV Middleware that returns a versionString of "800-73-4 Client API" shall implement all PIV Middleware functions listed in Table 1 except pivEstablishSecureMessaging and shall be able to recognize and process all mandatory and optional PIV data objects.

Note: Only SP 800-73-4 based PIV Middleware supports the use of on-card biometric comparison (OCC) data and the pairing code with the pivLogIntoCardApplication function, and only PIV Middleware that returns a versionString of "800-73-4 Client API with SM" supports the use of secure messaging (SM) and the virtual contact interface, which have been introduced in Parts 1 and 2 of SP 800-73-4. SP 800-73-1, SP 800-73-2, and SP 800-73-3 based PIV Middleware remain valid implementations; however, agencies are cautioned that using these implementations may result in limited interoperability. Further information can be found in Part 1 of SP 800-73-4. It provides an SP 800-73 revision history (Section I) and recommendations for PIV Middleware configuration management (Section II).

3.1.2 pivConnect

Purpose: Connects the client API to the PIV Card Application on a specific ICC.

Prototype:
```
status_word pivConnect(
   IN    Boolean           sharedConnection,
   INOUT sequence of bytes connectionDescription,
```

```
   INOUT LONG              CDLength,
   OUT   handle            cardHandle
);
```

Parameters:

`sharedConnection` If TRUE other client applications can establish concurrent connections to the ICC. If FALSE and the connection is established, then the calling client application has exclusive access to the ICC.

`connectionDescription` A connection description data object (tag 0x7F21). See Table 2.

If the length of the value field of the '8x' data object in the connection description data object is zero, then a list of the card readers of the type indicated by the tag of the '8x' series data object and available at the '9x' location is returned in the connectionDescription.

In order to provide sufficient space for the return value, the client application shall allocate a buffer of at least 2048 bytes for connectionDescription.

The connection description BER-TLV [ISO8825] used on the PIV client API shall have the structure described in Table 2.

Table 2. Data Objects in a Connection Description Template (Tag 0x7F21)

Description	Tag	Comment
Interface device – PC/SC	'81'	Card reader name
Interface device – SCP	'82'	Card reader identifier on terminal equipment
Interface device – EMR	'83'	Contactless connection using radio transmission
Interface device – IR	'84'	Contactless connection using infrared transmission
Interface device – PKCS #11	'85'	PKCS #11 interface
Interface device – CryptoAPI	'86'	CryptoAPI interface
Network node – Local	'90'	No network between client application host and card reader host
Network node – IP	'91'	IP address of card reader host
Network node – DNS	'92'	Internet domain name of card reader host
Network node – ISDN	'93'	ISDN dialing number string of terminal equipment containing the card reader

At most one selection from the '8x' series and one selection from the '9x' series shall appear in the connection description template.

For example, '7F 21 0C 82 04 41 63 6D 65 91 04 C0 00 02 17' describes a connection to a generic card reader at Internet address 192.0.2.23. As another example, '7F 21 0B 82 01 00 93 06 16 17 55 50 12 3F' describes a connection to the subscriber identity module in the mobile phone at +1 617 555 0123.

When used as an argument to the pivConnect entry point on the PIV client API described in this section, an '8x' series data object with zero length together with a '9x' series data object requests the return of all available card readers of the described type on the described node. Thus, '7F 21 04 81 00 90 00' would request a list of all available PC/SC card readers on the host on which the client application was running.

CDLength	Length of the card description parameter.
cardHandle	The returned opaque identifier of a communication channel to a particular ICC and hence of the card itself. cardHandle is used in all other entry points on the PIV client API to identify to which card the functionality of the entry point is to be applied.

Return Codes:
```
PIV_OK
PIV_CONNECTION_DESCRIPTION_MALFORMED
PIV_CONNECTION_FAILURE
PIV_CONNECTION_LOCKED
```

3.1.3 pivDisconnect

Purpose: Disconnect the PIV API from the PIV Card Application and the ICC containing the PIV Card Application.

Prototype:
```
status_word pivDisconnect(
   IN handle            cardHandle
);
```

Parameters:	**cardHandle**	Opaque identifier of the card to be acted upon as returned by pivConnect. The value of cardHandle is undefined upon return from pivDisconnect.

Return Codes:
```
PIV_OK
PIV_INVALID_CARD_HANDLE
PIV_CARD_READER_ERROR
```

If secure messaging has been established, then the PIV Middleware shall zeroize the secure messaging session keys.

3.2 Entry Points for Data Access

3.2.1 pivSelectCardApplication

Purpose: Set the PIV Card Application as the currently selected card application and establish the PIV Card Application's security state.

Prototype:
```
status_word pivSelectCardApplication(
   IN handle            cardHandle,
   IN sequence of byte  applicationAID,
   IN LONG              aidLength,
   OUT sequence of byte applicationProperties,
   INOUT LONG           APLength
);
```

Parameters:

cardHandle	Opaque identifier of the card to be acted upon as returned by pivConnect.
aidLength	Length of the PIV Card Application AID.
applicationAID	The AID of the PIV Card Application that is to become the currently selected card application.
applicationProperties	The application properties of the selected PIV Card Application. See Part 2, Table 3.
APLength	As an input, length of the buffer allocated for applicationProperties. As an output, length of the application properties.

Return Codes:

```
PIV_OK
PIV_INVALID_CARD_HANDLE
PIV_CARD_APPLICATION_NOT_FOUND
PIV_CARD_READER_ERROR
PIV_INSUFFICIENT_BUFFER
```

If the length of application properties is longer than the buffer allocated by the client application, then the PIV Middleware shall return `PIV_INSUFFICIENT_BUFFER`, but shall still set APLength to the length of the application properties.

3.2.2 pivEstablishSecureMessaging

Purpose: Establish secure messaging with the PIV Card Application.

Prototype:

```
status_word pivEstablishSecureMessaging(
   IN handle            cardHandle,
);
```

Parameters:

cardHandle	Opaque identifier of the card to be acted upon as returned by pivConnect.

Return Codes:

```
PIV_OK
PIV_INVALID_CARD_HANDLE
PIV_CARD_READER_ERROR
PIV_SM_FAILED
```

After successful execution of the key establishment protocol, the PIV Middleware shall perform all subsequent GET DATA, VERIFY, and GENERAL AUTHENTICATE commands over secure messaging, with the exception of any subsequent uses of the GENERAL AUTHENTICATE command to perform the key establishment protocol.

3.2.3 pivLogIntoCardApplication

Purpose: Set security state within the PIV Card Application.

Prototype:

```
status_word pivLogIntoCardApplication(
   IN handle            cardHandle,
   IN sequence of byte  authenticators,
   IN LONG              AuthLength
```

```
);
```

Parameters:

cardHandle	Opaque identifier of the card to be acted upon as returned by pivConnect.
authenticators	A sequence of zero or more BER-TLV encoded authenticators to be used to authenticate and set security state/status in the PIV Card Application context. The authenticator BER-TLV used on the PIV client API shall have the structure described in Table 3.
AuthLength	Length of the authenticator template.

Table 3. Data Objects in an Authenticator Template (Tag '67')

Description	Tag	M/O	Comment
Reference data	'81'	M	Value of the PIV Card Application PIN, Global PIN, or pairing code as described in Section 2.4.3, Part 2, or OCC data as described in Section 5.5.2 of [SP800-76]
Key reference	'83'	M	See Table 4a, Part 1 for PIV Card Application PIN, Global PIN, pairing code, and OCC key reference values

Return Codes:

```
PIV_OK
PIV_INVALID_CARD_HANDLE
PIV_AUTHENTICATOR_MALFORMED
PIV_AUTHENTICATION_FAILURE
PIV_SECURITY_CONDITIONS_NOT_SATISFIED
PIV_CARD_READER_ERROR
PIV_SM_FAILED
```

The PIV Middleware shall not submit authenticators to the PIV Card over a contactless interface without secure messaging. If secure messaging has not been established, then the pivLogIntoCardApplication function shall return `PIV_SECURITY_CONDITIONS_NOT_SATISFIED`.

3.2.4 pivGetData

Purpose: Return the entire data content of the named data object.

Prototype:

```
status_word pivGetData(
   IN handle            cardHandle,
   IN string            OID,
   IN LONG              oidLength,
   OUT sequence of byte data,
   INOUT LONG           DataLength
);
```

Parameters:

cardHandle	Opaque identifier of the card to be acted upon as returned by pivConnect.
OID	Object identifier of the object whose data content is to be retrieved coded as a string; for example, “2.16.840.1.101.3.7.2.96.80”. See Part 1, Table 3.

	oidLength	Length of the object identifier.
	data	Retrieved data content.
	DataLength	As an input, length of the buffer allocated for data. As an output, length of the data retrieved from the PIV Card.

Return Codes:
```
PIV_OK
PIV_INVALID_CARD_HANDLE
PIV_INVALID_OID
PIV_DATA_OBJECT_NOT_FOUND
PIV_SECURITY_CONDITIONS_NOT_SATISFIED
PIV_CARD_READER_ERROR
PIV_SM_FAILED
PIV_INSUFFICIENT_BUFFER
```

If the length of the retrieved data is longer than the buffer allocated by the client application, then the PIV Middleware shall return `PIV_INSUFFICIENT_BUFFER`, but shall still set DataLength to the length of the retrieved data. If the PIV Card Application returns a zero-length data object, then the PIV Middleware shall return `PIV_DATA_OBJECT_NOT_FOUND`.

3.2.5 pivLogoutOfCardApplication

Purpose: Reset the application security state/status of the PIV Card Application.

Prototype:
```
status_word pivLogoutOfCardApplication(
   IN handle            cardHandle
);
```

Parameters:	**cardHandle**	Opaque identifier of the card to be acted upon as returned by pivConnect. The cardHandle remains valid after execution of this function.

Return Codes:
```
PIV_OK
PIV_INVALID_CARD_HANDLE
PIV_CARD_READER_ERROR
```

3.3 Entry Points for Cryptographic Operations

3.3.1 pivCrypt

Purpose: Perform a cryptographic operation[4] such as encryption or signing on a sequence of bytes. Part 1, Appendix C describes recommended procedures for PIV algorithm identifier discovery.

Prototype:
```
status_word pivCrypt(
   IN handle              cardHandle,
   IN byte                algorithmIdentifier,
   IN byte                keyReference,
   IN sequence of byte    algorithmInput,
   IN LONG                inputLength,
   OUT sequence of byte   algorithmOutput,
```

[4] The pivCrypt function does not perform any cryptographic operations itself. It provides the interface to the GENERAL AUTHENTICATE command to perform cryptographic operations on card. All cryptographic operations, except SM on the client side, are performed outside the PIV Middleware.

```
    INOUT LONG              outputLength
);
```

Parameters:

cardHandle	Opaque identifier of the card to be acted upon as returned by pivConnect.
algorithmIdentifier	Identifier of the cryptographic algorithm to be used for the cryptographic operation. [SP800-78, Tables 6-2 and 6-3]
keyReference	Identifier of the on-card key to be used for the cryptographic operation. See [SP800-78, Table 6-1] and Part 1, Table 4b.
algorithmInput	Sequence of bytes used as the input to the cryptographic operation. The algorithmInput for RSA algorithms shall be restricted to the range 0 to *n*-1, where *n* is the RSA modulus.
inputLength	Length of the algorithm input.
algorithmOutput	Sequence of bytes output by the cryptographic operation.
outputLength	As an input, length of the buffer allocated for algorithmOutput. As an output, length of the algorithm output.

Return Codes:

```
PIV_OK
PIV_INVALID_CARD_HANDLE
PIV_INVALID_KEYREF_OR_ALGORITHM
PIV_SECURITY_CONDITIONS_NOT_SATISFIED
PIV_INPUT_BYTES_MALFORMED
PIV_CARD_READER_ERROR
PIV_SM_FAILED
PIV_INSUFFICIENT_BUFFER
```

The `PIV_INPUT_BYTES_MALFORMED` error condition indicates that some property of the data to be processed such as the length or padding was inappropriate for the requested cryptographic algorithm or key.

If the value of keyReference is '04' (PIV Secure Messaging key), then the PIV Middleware shall return `PIV_INVALID_KEYREF_OR_ALGORITHM`.

If the length of the algorithm output is longer than the buffer allocated by the client application, then the PIV Middleware shall return `PIV_INSUFFICIENT_BUFFER`, but shall still set outputLength to the length of the algorithm output.

3.4 Entry Points for Credential Initialization and Administration

The PIV Middleware shall not submit data provided to the pivPutData or pivGenerateKeyPair function over the contactless interface. If the PIV Middleware is not communicating with the PIV Card via the card's contact interface, then the pivPutData or pivGenerateKeyPair function shall return `PIV_FUNCTION_NOT_SUPPORTED`.

3.4.1 pivPutData

Purpose: Replace the entire data content of the named data object with the provided data.

Prototype:
```
status_word pivPutData(
   IN handle            cardHandle,
   IN string            OID,
   IN LONG              oidLength,
   IN sequence of byte  data,
   IN LONG              dataLength
);
```

Parameters:

cardHandle	Opaque identifier of the card to be acted upon as returned by pivConnect.
OID	Object identifier of the object whose data content is to be replaced coded as a string; for example, "2.16.840.1.101.3.7.2.96.80". See Part 1, Table 3.
oidLength	Length of the object identifier.
data	Data to be used to replace in its entirety the data content of the named data object.
dataLength	Length of the provided data.

Return Codes:
```
PIV_OK
PIV_INVALID_CARD_HANDLE
PIV_INVALID_OID
PIV_CARD_READER_ERROR
PIV_INSUFFICIENT_CARD_RESOURCE
PIV_SECURITY_CONDITIONS_NOT_SATISFIED
PIV_FUNCTION_NOT_SUPPORTED
```

3.4.2 pivGenerateKeyPair

Purpose: Generates an asymmetric key pair in the currently selected card application.

If the provided key reference exists and the cryptographic mechanism associated with the reference data identified by this key reference is the same as the provided cryptographic mechanism, then the generated key pair replaces in entirety the key pair currently associated with the key reference.

Prototype:
```
status_word pivGenerateKeyPair(
   IN handle               cardHandle,
   IN byte                 keyReference,
   IN byte                 cryptographicMechanism,
   OUT sequence of byte    publicKey,
   INOUT LONG              KeyLength
);
```

Parameters:

cardHandle	Opaque identifier of the card to be acted upon as returned by pivConnect.
keyReference	The key reference of the generated key pair.

cryptographicMechanism	The type of key pair to be generated. See Part 1, Table 5.
publicKey	BER-TLV data objects defining the public key of the generated key pair. See Part 2, Table 11.
KeyLength	As an input, length of the buffer allocated for publicKey. As an output, length of the public key related data retrieved from the PIV Card.

Return Codes:

```
PIV_OK
PIV_INVALID_CARD_HANDLE
PIV_SECURITY_CONDITIONS_NOT_SATISFIED
PIV_FUNCTION_NOT_SUPPORTED
PIV_INVALID_KEY_OR_KEYALG_COMBINATION
PIV_UNSUPPORTED_CRYPTOGRAPHIC_MECHANISM
PIV_CARD_READER_ERROR
PIV_INSUFFICIENT_BUFFER
```

If the length of public key related data retrieved from the PIV Card is longer than the buffer allocated by the client application, then the PIV Middleware shall return `PIV_INSUFFICIENT_BUFFER`, but shall still set KeyLength to the length of the public key related data retrieved from the PIV Card.

Appendix A—Terms, Acronyms, and Notation

A.1 Terms

Application Identifier	A globally unique identifier of a card application as defined in ISO/IEC 7816-4.
Application Session	The period of time within a card session between when a card application is selected and a different card application is selected or the card session ends.
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).
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 Interface Device	An electronic device that connects an integrated circuit card and the card applications therein to a client application.
Card Reader	Synonym for card interface device.
Client Application	A computer program running on a computer in communication with a card interface device.
Card Management Operation	Any operation involving the PIV Card Application Administrator.
Data Object	An item of information seen at the card command interface for which are specified a name, a description of logical content, a format and a coding.
Interface Device	Synonym for card interface device.
Key Reference	A PIV key reference is a one-byte identifier that specifies a cryptographic key according to its PIV Key Type. The identifier used in cryptographic protocols such as an authentication or a signing protocol.
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 encodes 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.

A.2 Acronyms

AID	Application Identifier
API	Application Programming Interface
ASN.1	Abstract Syntax Notation One
BER	Basic Encoding Rules
FIPS	Federal Information Processing Standards
FISMA	Federal Information Security Management Act
GSC-IS	Government Smart Card Interoperability Specification
HSPD	Homeland Security Presidential Directive
ICC	Integrated Circuit Card
IEC	International Electrotechnical Commission
INCITS	InterNational Committee for Information Technology Standards
ISDN	Integrated Services Digital Network
ISO	International Organization for Standardization
ITL	Information Technology Laboratory
LSB	Least Significant Bit
MSB	Most Significant Bit
NIST	National Institute of Standards and Technology
OCC	On-Card biometric Comparison
OID	Object Identifier
OMB	Office of Management and Budget
PC/SC	Personal Computer/Smart Card
PIN	Personal Identification Number
PIV	Personal Identity Verification
PKCS	Public-Key Cryptography Standards
PKI	Public Key Infrastructure
RFU	Reserved for Future Use
SM	Secure Messaging
SP	Special Publication
TLV	Tag-Length-Value

A.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 shall be set to 0.

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

Data objects in templates are described as being mandatory (M) or optional (O). 'Mandatory' means the data object shall appear in the template. 'Optional' means the data object may appear in the template.

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 template.

Appendix B—References

[FIPS201] Federal Information Processing Standard 201-2, *Personal Identity Verification (PIV) of Federal Employees and Contractors*, August 2013.
(See http://dx.doi.org/10.6028/NIST.FIPS.201-2)

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

[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).*

[SP800-76] NIST Special Publication 800-76-2, *Biometric Specifications for Personal Identity Verification*, July 2013. (See http://dx.doi.org/10.6028/NIST.SP.800-76-2)

[SP800-78] NIST Special Publication 800-78-4, *Cryptographic Algorithms and Key Sizes for Personal Identity Verification*, May 2015. (See http://csrc.nist.gov)