Revised Draft NIST Special Publication 800-73-4

Formatted

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

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

COMPUTER SECURITY

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

Revised Draft NIST Special Publication 800-73-4

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

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

Jason Mohler
Electrosoft Services, Inc.

May 2014

U.S. Department of Commerce
Penny Pritzker, Secretary

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

Authority

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

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 2014)
CODEN: NSPUE2

Certain commercial entities, equipment, or materials may be identified in this document in order to describe an experimental procedure or concept adequately. Such identification is not intended to imply recommendation or endorsement by NIST, nor is it intended to imply that the entities, materials, or equipment are necessarily the best available for the purpose.

There may be references in this publication to other publications currently under development by NIST in accordance with its assigned statutory responsibilities. The information in this publication, including concepts and methodologies, may be used by Federal agencies even before the completion of such companion publications. Thus, until each publication is completed, current requirements, guidelines, and procedures, where they exist, remain operative. For planning and transition purposes, Federal agencies may wish to closely follow the development of these new publications by NIST.

Organizations are encouraged to review all draft publications during public comment periods and provide feedback to NIST. All NIST Computer Security Division publications, other than the ones noted above, are available at http://csrc.nist.gov/publications.

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

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

Reports on Computer Systems Technology

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

Abstract

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

Keywords

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

Acknowledgements

The authors (Ramaswamy Chandramouli, David Cooper, Hildegard Ferraiolo, Salvatore Francomacaro, and Ketan Mehta of NIST, and Jason Mohler of Electrosoft Services, Inc.) wish to thank their colleagues who reviewed drafts of this document and contributed to its development.

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. Personal Identity Verification (PIV) of Federal Employees and Contractors, Federal Information Processing Standard 201 (FIPS 201) [FIPS201] was developed to establish standards for identity credentials. Special Publication 800-73-4 (SP 800-73-4) contains technical specifications to interface with the smart card (PIV Card[1]) to retrieve and use the identity credentials.

Deleted: The

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 another person (human readable and verifiable) or an automated process (computer readable and verifiable).

1.4 Content and Organization

All sections in this document are *normative* (i.e., mandatory for compliance) unless specified as *informative* (i.e., non-mandatory). Following is the structure of Part 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. 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 pivGetData is `PIV_SECURITY_CONDITIONS_NOT_SATISFIED` this indicates that communication with the card is over a contactless interface.

[2] Only the pivPutData and pivGenerateKeyPair API functions perform card-management functionality.

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.

Deleted: L

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	E.g., the PIN value
Key reference	'83'	M	See Table 4, 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.

Deleted: L

Deleted: to

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.

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[3] 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,
   INOUT LONG               outputLength
);
```

[3] 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.

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 4.
	`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.

Deleted: L

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 '03' (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_SECURITY_CONDITIONS_NOT_SATISFIED`.

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
```

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.

Deleted: L

Return Codes:

```
PIV_OK
PIV_INVALID_CARD_HANDLE
PIV_SECURITY_CONDITIONS_NOT_SATISFIED
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://csrc.nist.gov)

[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).*

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