NIST Special Publication 800-73-2

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

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

Ramaswamy Chandramouli
James F. Dray
Hildegard Ferraiolo
Scott B. Guthery
William MacGregor
Ketan Mehta

I N F O R M A T I O N S E C U R I T Y

Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD, 20899-8930

September 2008

U.S. Department of Commerce
Carlos M. Gutierrez, Secretary

National Institute of Standards and Technology
James M. Turner, Deputy Director

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 non-national security-related information in Federal information systems. This 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.

National Institute of Standards and Technology Special Publication 800-73-2, Part 3, 19 pages (September 2008)

Acknowledgements

The authors (Ramaswamy Chandramouli, James Dray, Hildegard Ferraiolo, William MacGregor, of NIST Ketan Mehta of Mehta Inc. and Scott Guthery of Mobile Mind, Inc.) wish to thank their colleagues who reviewed drafts of this document and contributed to its development. Special thanks to the Government Smart Card Interagency Advisory Board (GSC-IAB) and InterNational Committee for Information Technology Standards (INCITS) for providing detailed technical inputs to the SP 800-73-2 development process. 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 Appendices

List of Tables

1. Introduction

The 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 Federal government locations and systems. The Personal Identity Verification (PIV) of Federal Employees and Contractors, Federal Information Processing Standard 201 (FIPS 201) [1] was developed to establish standards for identity credentials. Special Publication 800-73-2 (SP 800-73-2) specifies interface requirements for retrieving and using the identity credentials from the PIV Card and is a companion document to FIPS 201.

1.1 Authority

This document has been developed by the National Institute of Standards and Technology (NIST) in furtherance of its statutory responsibilities under the Federal Information Security Management Act (FISMA) of 2002, Public Law 107-347.

NIST is responsible for developing standards and guidelines, including minimum requirements, for providing adequate information security for all agency operations and assets, but such standards and guidelines shall not apply to national security systems. This recommendation 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 A-130, Appendix IV: Analysis of Key Sections. Supplemental information is provided A-130, Appendix III.

This recommendation has been prepared for use by federal agencies. It may be used by non-governmental organizations on a voluntary basis and is not subject to copyright though attribution is desirable. Nothing in this document should be taken to contradict standards and guidelines made mandatory and binding on Federal agencies by the Secretary of Commerce under statutory authority. Nor should this recommendation be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the Office of Management and Budget (OMB), or any other Federal official.

1.2 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 800-73-2 contains 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 800-73-2 enumerates requirements where the standards 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.

1.3 Scope

SP 800-73-2 specifies the PIV data model, Application Programming Interface and card interface requirements necessary to comply with the use cases, as defined in Section 6 of FIPS 201 and further elaborated in Appendix B of SP 800-73-2, Part 1. 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 interchangeable by all information processing system across Federal agencies.

This Part, Special Publication 800-73-2 (SP 800-73-2) Part 3: *End-Point PIV Client Application Programming Interface* contains technical specifications of the PIV client application programming interface to the PIV Card.

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 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: End-Point Concept and Construct,* describes both PIV Card Application and the PIV client-application programming interface. This section is informative.
- Section 3, *End-Point PIV 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: End-Point Concepts and Constructs

SP 800-73-2 Part 2 and Part 3 define two interfaces to an ICC that contains the Personal Identity Verification card application: a low-level PIV Card Application card command interface (Part 2) and a high-level PIV 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-application programming interface or the card command interface.

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

The client-application programming interface is thought of as being at a higher level than the card command interface because access to a single entry point on the client-application programming interface 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-application programming interface.

The client-application programming interface 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-application program interface may be different from the representation of these same concepts and constructs on the card command interface.

3. End-Point Client-Application Programming Interface

Table 1 lists the entry points on the PIV client-application programming interface. This section references Object Identifiers (OIDs), which are defined and can be found in Part 1 (Table 2).

Table 1. Entry Points on PIV End-Point Client-Application Programming Interface

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

3.1 Entry Points for Communication

3.1.1 pivMiddlewareVersion

Purpose: Returns the PIV Middeware version string

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

Parameter: **`versionString`** For SP 800-73-2 Part 3 conformant PIV middleware, the parameter returns "80073-2 Client API".

Return Codes: `PIV_OK`

Note: SP 800-73-1 conformant PIV Middleware does not implement the pivMiddlewareVersion Client API function. Therefore, a client application invoking the pivMiddlewareVersion function should expect a "function-not-supported" error from 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". SP 800-73-2 Part 3 conformant PIV middleware shall implement the new pivMiddlewareVersion function as well as the pivPutData and pivGetData functions for the 0x7E interindustry BER-TLV Discovery object.

3.1.2 pivConnect

Purpose: Connects the client-application programming interface to the PIV Card Application on a specific ICC.

Prototype:

```
status_word pivConnect(
   IN    Boolean           sharedConnection,
   INOUT sequence of bytes connectionDescription,
   IN    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 '7F 21 '). 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.

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

Table 2. Data Objects in a Connection Description Template (Tag '7F21')

Description	Tag	M/O/C[1]	Comment
Interface device – PC/SC	'81'	C	Card reader name
Interface device – SCP	'82'	C	Card reader identifier on terminal equipment
Interface device – EMR	'83'	C	Contactless connection using radio transmission
Interface device – IR	'84'	C	Contactless connection using infrared

[1] M = Mandatory, O = Optional, C = Conditional. For the definition of M/O/C see, Appendix A.3

			transmission
Interface device – PKCS#11	'85'	C	PKCS#11 interface
Interface device – CryptoAPI	'86'	C	CryptoAPI interface
Network node – Local	'90'	C	No network between client-application host and card reader host
Network node – IP	'91'	C	IP address of card reader host
Network node – DNS	'92'	C	Internet domain name of card reader host
Network node – ISDN	'93'	C	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 81 06 0D 17' describes a connection to a generic card reader at Internet address 129.6.13.23. As another example, '7F 21 0B 82 01 00 93 06 16 17 12 34 56 7F' describes a connection to the subscriber identity module in the mobile phone at +1 617 123 4567.

When used as an argument to the pivConnect entry point on the PIV client-application programming interface described in his 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-application programming interface to identify 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 application programming interface 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
```

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,
   OUT LONG             APLength
);
```

Parameters:

cardHandle	Opaque identifier of the card to be acted upon as returned by pivConnect.
aidLength	Length of the 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	Length of the application properties.

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

3.2.2 pivLogIntoCardApplication

Purpose: Set security state within the PIV Card Application.

Prototype:
```
status_word pivLogIntoCardApplication(
   IN handle            cardHandle,
   IN sequence of byte  authenticators,
   OUT 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-application programming interface 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 or challenge response
Key reference	'83'	M	See table 3, Part 1 for PIN reference values

Return Codes:
```
PIV_OK
PIV_INVALID_CARD_HANDLE
PIV_AUTHENTICATOR_MALFORMED
PIV_AUTHENTICATION_FAILURE
PIV_CARD_READER_ERROR
```

3.2.3 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,
   OUT 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, '2.16.840.1.101.3.7.1.1.2.2.1' See Part 1 Table 2.

oidLength Length of the object identifier.

data Retrieved data content.

DataLength Length of the data to be 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
```

3.2.4 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[2] such as encryption or signing on a sequence of bytes. Appendix C, Part 1, 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,
   OUT 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. See Table 6-2 and 6-3 in SP 800-78[4].
keyReference	Identifier of the on-card key to be used for the cryptographic operation. See Table 6-1 and 6-3 in SP 800-78.
algorithmInput	Sequence of bytes used as the input to the cryptographic operation.[3]

[2] 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 on the client side are performed outside the PIV middleware.

inputLength	Length of the algorithm input.
algorithmOutput	Sequence of bytes output by the cryptographic operation.
outputLength	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
```

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.

3.4 Entry Points for Credential Initialization and Administration

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,
   OUT 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, "2.16.840.1.101.3.7.1.1.2.2.1". See Table 2, Part 1.
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 data to be retrieved from the PIV Card.

Return Codes: `PIV_OK`

[3] The algorithmInput for RSA algorithms shall be restricted to the range 0 to n-1, where n is the RSA modulus.

```
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,
   OUT 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 4.
publicKey	BER-TLV data objects defining the public key of the generated key pair. See Table Part 2, Table 10.
KeyLength	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_INVALID_KEY_OR_KEYALG_COMBINATION
PIV_UNSUPPORTED_CRYPTOGRAPHIC_MECHANISM
PIV_CARD_READER_ERROR
```

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	An 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., CBC or 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.
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 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
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 Standards Organization
ITL	Information Technology Laboratory
LSB	Least Significant Bit
MSB	Most Significant Bit
NIST	National Institute of Standards and Technology
OID	Object Identifier
OMB	Office of Management and Budget
PIN	Personal Identification Number
PIV	Personal Identity Verification
PIX	Proprietary Identifier eXtension

PKCS	Public Key Cryptography Standard
PKI	Public Key Infrastructure
RFU	Reserved for Future Use
RID	Registered application provider IDentifier
SP	Special Publication
TLV	Tag-Length-Value

A.3 Notation

The sixteen hexadecimal digits shall be denoted using the alphanumeric characters 0, 1, 2…, A, B, C, D, E, and F. A byte consists of two hexadecimal digits, for example, '2D'. 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 reserved for future use 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), 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 a footnote to the table.

In other tables the M/O column identifies properties of the PIV Card Application that shall be present (M) or may be present (O).

BER-TLV data object tags are represented as byte sequences as described above. Thus, for example, '4F' is the interindustry data object tag for an application identifier and '7F 60' is the interindustry data object tag for the biometric information template.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this standard are to be interpreted as described in IETF RFC 2119, Key Words for Use in RFCs to Indicate Requirement Levels [3].

Appendix B—References

[1] Federal Information Processing Standard 201-1, Change Notice 1, *Personal Identity Verification (PIV) of Federal Employees and Contractors*, March 2006. (See http://csrc.nist.gov)

[2] 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).*

[3] IETF RFC 2119, *Key Words for Use in RFCs to Indicate Requirement Levels*, March, 1997.

[4] NIST Special Publication 800-78-1, *Cryptographic Algorithms and Key Sizes for Personal Identity Verification*, August 2007. (See http://csrc.nist.gov)