NIST Special Publication 800-85B

PIV Data Model Test Guidelines

Ramaswamy Chandramouli
Ketan Mehta
Pius A. Uzamere II
David Simon
Nabil Ghadiali
Andrew P. Founds

INFORMATION SECURITY

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

July 2006

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

National Institute of Standards and Technology
William A. Jeffrey, 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 800-85B, 164 pages

(July 2006)

Acknowledgements

The authors 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 contribution improved the quality and usefulness of this publication.

Executive Summary

Homeland Security Presidential Directive 12 (HSPD-12) called for a new standard to be adopted governing the use of common identity credentials for physical and logical access to Federal government locations and systems. The Personal Identity Verification (PIV) standard for Federal Employees and Contractors, Federal Information Processing Standard 201 (FIPS201), was developed to establish government-wide identity credentials. Credentials are issued to individuals whose true identity has been verified and whose need for the credential has been established and authorized by proper authorities.

FIPS201 describes a variety of data model components as a part of the PIV logical credentials. Such components include biometric elements in the form of fingerprint information and facial imagery and security elements such as electronic keys, certificates, and signatures. FIPS201 incorporates by reference NIST Special Publication 800-73 (SP80073), which specifies elements related to the PIV card interface, NIST Special Publication 800-76 (SP80076), which specifies the biometric requirements, and NIST Special Publication 800-78 (SP80078) which specifies acceptable cryptographic algorithms and key sizes for PIV systems.

A robust testing framework and guidance to provide assurance that a particular component or system is compliant with FIPS201 and supporting standards should exist to build the necessary PIV infrastructure to support common unified processes and systems for government-wide use. NIST developed test guidance in two parts. The first part addresses test requirements for interface to the PIV card and are provided in SP80085A. The second part provides test requirements for the PIV data model and is provided in this document. This document specifies the derived test requirements, and the detailed test assertions and conformance tests for testing the PIV data model.

Table of Contents

List of Appendices

1. Introduction

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 in 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. 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 OMB, or any other Federal official.

1.2 Purpose and Scope

The Federal Information Processing Standard 201 (FIPS201) establishes a system for verifying an individual employee or contractor's identity in a reliable, secure, and interoperable manner across the Federal government. Credentials are issued to individuals whose true identity has been verified and whose need for the credential has been established and authorized by proper authorities. FIPS201 also describes a variety of authentication mechanisms, including the use of cryptographic mechanisms and biometric data belonging to cardholders.

In order to build the necessary Personal Identity Verification (PIV) infrastructure to support common unified processes and systems for government-wide use, there must be a robust testing framework to provide assurance that a particular component or system is compliant with FIPS201 and companion specifications. This test guidance document specifies the derived test requirements, detailed test assertions, and conformance tests for testing the data elements of the PIV system as per specifications laid out in FIPS201, SP80073, SP80076, and SP80078.

This document does not provide conformance tests for any other software used in the PIV system such as the back-end access control software, card issuance software, and specialized service provider software. Specifically, this document does not provide test requirements for the PIV card interface, FIPS 140-2 validation, key generation and certificate binding, cryptographic algorithms, biometric enrollment and verification processes[1], performance of biometric products, and non-PIV aspects of external biometric standards and profiles.

[1] Testing of biometric processing performance using measures such as False Accept Rate (FAR) is described in Section 7 of SP80076.

This document provides technical guidance on the methodology to be used during testing applicable components, but does not provide normative guidance on which entities will execute the tests. Also, the test methodologies defined in this document are not designed to test business processes or to verify compliance with external applicable standards. For example, this document does not provide test guidance to determine how good a user's Personal Identification Number (PIN) choice is or how access rights are granted to employees.

1.3 Audience and Assumptions

This document is targeted at vendors and integrators of PIV components, as well as the entities that will conduct tests on such components. Readers are assumed to have a working knowledge of FIPS201, PIV guidance, and applicable technologies.

This document will:

Enable developers of PIV components to develop their modules to be testable for requirements specified in FIPS201, SP80073, SP80076, and SP80078.

Enable developers of PIV components to develop self-tests as part of the development effort.

Enable testers to develop tests that cover the test suite provided in this document.

2. Conformance Test Overview

The conformance testing guidelines in this document applies to the testing of a PIV data model. The data model requirements are extracted from FIPS201, SP80073, SP80076, and SP80078. This overview section provides a high level conformance test architecture for testing the PIV data model. The conformance test architecture is confined to the end result of a personalized PIV card. In other words, the conformance test approach views the card issuance system as a "black box," meaning that the interface of that system is opaque and its implementation details are not relevant to the testing. The PIV data model testing operates under the assumption that the PIV card being tested has already been personalized as described in Sections 2.3 and 5.3 of FIPS201. The following sections provide the details of data model testing, test architecture, test methodology, and test areas.

2.1 Test Architecture

The conceptual architecture for data model testing is shown in Figure 1. The conformance test in this document applies to the area highlighted with dashed lines. SP80085A addresses the writing and extracting of data from the card and subsequently, those processes are not addressed in this document.

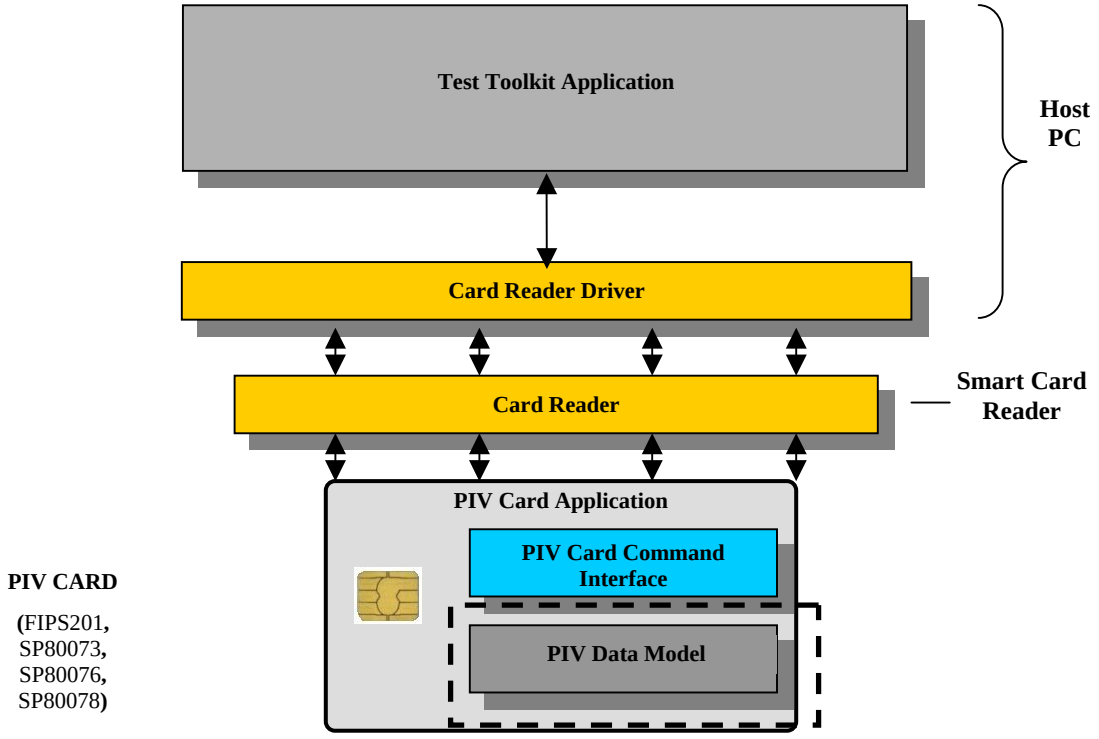

Figure 1: PIV Conformance Test Architecture

The PIV data model defines the logical use of the on-card application space including the SP80073 required data objects and data elements along with the size and structure of each object.

The PIV data model test includes the testing of the following aspects of PIV Data:

Basic Encoding Rules Tag-Length-Value (BER-TLV) Format Conformance as per Appendix A of SP80073 for all objects.

Conformance of the Signature Block to Cryptographic Message Syntax (CMS) Signature format for all signed objects.

Conformance to Common Biometric Exchange Formats Framework (CBEFF) Profile and American National Standards Institute (ANSI) International Committee for Information Technology Standards (INCITS) 378 and 385 Profiles respectively for Card Holder Fingerprint and Facial Image objects respectively.

Conformance to Federal Identity Credentialing Committee (FICC) profiles for all Public Key Infrastructure (PKI) Certificates.

2.2 Test Methodology

The data model testing was developed through the following two-step process:

Create derived test requirements (DTRs) — These are constructed from the data format and content requirements in FIPS201, SP80073, SP80076, and SP80078 specifications.

+ **Develop test assertions** — These provide the tests that need to be performed to test each of the DTRs. The test assertions will include testing of data formats, values in the individual fields, relationship among values in multiple fields and validate the computations. Also, the test assertions include testing of the optional fields when they are present.

Figure 2 depicts the test methodology adopted to provide complete guidance for testing PIV conformant products. SP80085A provides the DTRs and test assertions for the interfaces to the PIV smart card and the PIV middleware. This document provides DTRs and test assertions for the identity credentials stored on the PIV card.

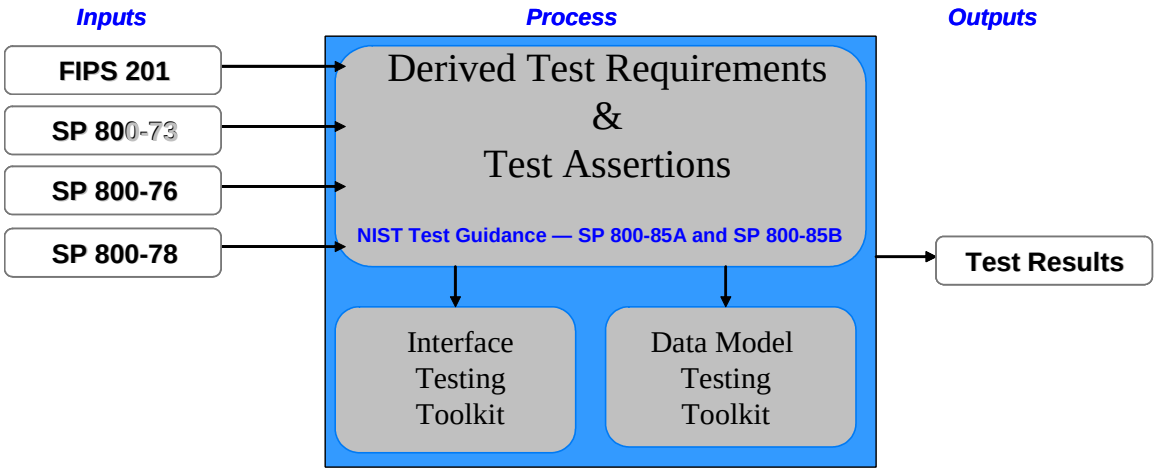

Figure 2: PIV Test Methodology

2.3 Test Set-up

The test system consists of the following components:

A test toolkit application software that resides on a personal computer.

An ISO7816 and Personal Computer / Smart Card (PC/SC) compliant contact-based smart card reader.

A mechanism to input PIN that can be transmitted to the smart card reader. Examples of such mechanisms are a PIN pad or a keyboard.

A set of test personalized PIV cards whose applications and interfaces are compliant with SP80073. All personalized biometric information is assumed to be collected and processed by template generation and matching implementations that have been tested against minimum performance qualification criteria established by NIST, OMB, and by Federal agencies, as appropriate.

2.4 Test Areas

The test assertions in this document will validate that all PIV data objects conform to their respective requirements. Conformance criteria includes correct formatting and, when appropriate, context specific content. Additionally, conformance will be based on correct computation of content such as digital signatures. The DTRs and test assertions are designed to validate each PIV data object such that the following three statements are true for the objects:

PIV containers are formatted correctly,

Field values are in accordance with the specifications, and

Data consistency and value computations such as signatures are accurate.

Again, these requirements are not designed to test business processes or to verify external compliance with applicable standards. For further clarification of the document scope, refer to Section 1.2, Purpose and Scope.

2.4.1 BER-TLV Format Conformance

The tags and lengths in various data objects shall conform to specifications in Appendix A of SP80073.

2.4.2 Digital Signature Blocks Conformance

For all signed objects the fields in the signature block shall conform to the CMS syntax specified in FIPS201.

2.4.3 Biometric Data Objects Conformance

The two biometric objects, Card Holder Fingerprints and Facial Image (if present on the PIV card), shall conform to the common CBEFF Header format as well as to ANSI/INCITS 378 and ANSI/INCITS 385 profiles respectively.

2.4.4 Certificate Profile Conformance

The mandatory PIV Authentication Certificate as well as the optional Digital Signature, Key Management, and Card Authentication (if asymmetric cryptography is used) Certificate shall conform to the certificate profiles as specified in the X.509 Certificate and Certificate Revocation List (CRL) Extensions Profile for the Shared Service Providers (SSP) Program (X509 Extensions).

3. Test Methodology

3.1 Derived Test Requirements

DTRs show the type of tests required based on the normative specifications in FIPS201 and supporting special publications. These specifications cover expected data object representations and content. Each DTR consists of the following:

Actual condition statements taken/derived from the specification — these include conditions for successful command execution for each command as well as exception behaviors explicitly specified by statements using the words "shall," "must," and other normative delimiters in the standard. The condition statements are identified by codes starting with 'AS' followed by a running sequence.

Required Vendor Information — these include information that the vendors (could also be agencies or integrators) are mandated to provide in their documentation. The Required Vendor Information is identified by codes starting with 'VE' followed by a running sequence.

Required Test Procedures — these are actions that the tester has to perform in order to satisfy the requirements stated in actual condition statements. These include verifying the information mandated in the "Required Vendor Information" for the condition as well as performing software-based tests. Some of the required test procedures do not call explicitly for verification of information in the associated "Required Vendor Information." In these instances it is implicitly assumed that such information is provided by the vendor and verified by the tester. The Required Test Procedures are identified by codes starting with 'TE' followed by a running sequence that denotes the section in this document where they occur.

Validation of some DTRs are not covered by the test assertions provided in this document. These DTRs require compliance of a component with an external specification or standard such as EFTS. No required test procedures are provided for these DTRs, and a note is added to indicate that "this assertion is externally tested." The tester is required to check the vendor documentation for claimed compliance with such requirements or confirm the presence of an external test/compliance certificate obtained from the test organization, when applicable.

In some instances, testing of DTRs may not be feasible using the test methodologies described in this document. For example, a test tool built on these methods cannot test the procedure by which fingerprints are taken at an agency PIV installation. Most of these DTRs, however, can be tested by inspection of the system description document. Where this is the case, an adequate description is generally required by the VE section of the DTR. Where testing is not feasible, a note is added to indicate that "this assertion is not separately tested."

3.2 Test Assertions

Test assertions are statements of behavior, action, or condition that can be measured or tested. They provide the procedures to guide the tester in executing and managing the test. They

include purpose of the test, starting conditions and prerequisites, success criteria, and post-test conditions, when applicable.

The following four sets of test assertions are included in this document —

BER-TLV (Section 8)

Biometric data object (Section 9)

Signed data element (Section 10)

PKI certificate profile (Section 11)

All the test assertions provided in this document come under PIV data model testing and are based on DTRs in Sections 4, 5, 6, and 7. Specifically, each test assertion makes specific references to the related DTR sections. Overall there is a many-to-many relationship from the test assertions to the DTRs (i.e., one test can map to many DTRs and one DTR can map to many tests). To narrow the search space for cross references, Table 3-1 presents a cross-referencing guide showing the relevant DTR sections and test assertion sections with respect to tests.

Category/Classes of Test	DTR Section(s)	Test Assertion Section(s)
(1) BER-TLV	Section 4 (Derived from SP80073)	Section 8
(2) Biometric Data	Section 5 (Derived from SP80076)	Section 9
(3) Signed Data Elements	Section 6 (Derived from FIPS201 and SP80078)	Section 10
(4) PKI Certificate Profile	Section 7 (Derived from FIPS201 and SP80078)	Section 11

Table 3-1. Cross-referencing Guide

4. BER-TLV DTRs

4.1 BER-TLV Testing

AS04.01.01: Part 3 conformant cards shall return all the Tag-Length-Value (TLV) elements of a container in the physical order listed for that container in this data model.

VE04.01.01.01: The vendor shall specify in its documentation the format (TLV) and the content of all the elements in each data container on the card.

VE04.01.01.02: The vendor shall specify in its documentation that the information provided conforms to SP80073.

TE04.01.01.01: The tester shall validate that the formatting, encoding and the content of all the elements in each data container conforms to SP80073.

4.2 Card Capability Container (CCC)

AS04.02.01: The CCC shall identify the registered data model number `0x10`.

VE04.02.01.01: The vendor shall specify in its documentation the tags and associated values in the CCC container.

VE04.02.01.02: The vendor shall specify presence of the optional fields.

TE04.02.01.01: The tester shall validate the format and the content of all the elements in CCC data container on the card. Data read from the card will be validated aginst the vendor provided data.

TE04.02.01.02: The tester shall validate that the Registered Data Model value is `0x10`.

4.3 Card Holder Unique Identifier (CHUID)

AS04.03.01: The CHUID on a PIV card shall meet the following requirements:

The Federal Agency Smart Credential Number (FASC-N) shall be consistent with the Technical Implementation Guidance Smart Card Enabled Physical Access Control System (TIG SCEPACS) Option for "System Code || Credential Number" to establish a credential number space of 9,999,999,999 credentials.

The Global Unique Identifier (GUID) field must be present, and may include either an issuer assigned IPv6 address or be coded as all zeros. The GUID is included to enable future migration away from the FASC-N into a robust numbering scheme for all issued credentials.

The Expiration Date is tagged `0x35` and value is within the next five years. This field shall be 8 bytes in length and shall be encoded as YYYYMMDD.

VE04.03.01.01: The vendor shall specify in its documentation the format (TLV) and the content of all the elements in CHUID container on the card.

VE04.03.01.02: The vendor shall specify presence of the optional fields.

TE04.03.01.01: The tester shall validate the format and the content of all the elements in CHUID data container on the card.

4.4 Biometric Fingerprint

AS04.04.01: The fingerprint buffer specify the primary and secondary fingerprints within Tag value `0xBC`.

There are no vendor requirements.

TE04.04.01.01: The tester shall validate that the fingerprint data follows the tag value `0xBC` within the container.

AS04.04.02: The fingerprint template length shall not exceed 4,000 bytes.

There are no vendor requirements.

TE04.04.02.01: The tester shall validate that the length value after the tag `0xBC` is less than 4000 bytes.

4.5 Biometric Facial

AS04.05.01: The facial image is preceded with tag value `0xBC`.

VE04.05.01.01: The vendor shall specify if the facial image is stored on the card.

TE04.05.01.01: The tester shall validate that the facial image follows the tag value `0xBC` within the container.

AS04.05.02: The facial image length shall not exceed 12,710 bytes.

VE04.05.02.01: The vendor shall specify if the facial image is stored on the card in their documentation.

TE04.05.02.01: The tester shall validate that the length value is less than 12,710 bytes.

4.6 Security Object

AS04.06.01: The message digest produced as a result of a hash function on the contents of a data object buffer shall be identical to that data object's message digest contained in the security object.

There are no vendor requirements.

TE04.06.01.01: The tester shall validate that the message digests for the various data objects present in the security object are identical to the message digest of the data object itself.

5. Biometric Data

5.1 Common Header for PIV Biometric Data

The assertions in this section apply to both fingerprint template and facial image stored on the PIV card. Facial image is an optional element on the PIV card and must be tested if present.

AS05.01.01: The CBEFF structure must comply with SP80076 Table 7, "Simple CBEFF Structure."

VE05.01.01: The vendor shall specify if the optional facial image is stored on the PIV card.

TE05.01.01.01: The tester shall verify that the CBEFF structure is implemented in accordance with Table 7 of SP80076.

AS05.01.02: The CBEFF header must comply with SP80076 Table 8, "Patron Format PIV Specification."

No requirements for vendor.

TE05.01.02.01: The tester shall verify the length of the Patron Format header.

TE05.01.02.02: The tester shall verify the values are consistent with Table 8 requirements of SP80076.

AS05.01.03: Multi-byte integers in the CBEFF headers shall be in big-endian byte order.

VE05.01.03.01: The vendor shall document the values of the CBEFF header fields.

TE05.01.03.01: The tester shall compare value provided against the stored data.

AS05.01.04: The Patron Header Version of the CBEFF Patron Format shall be `0x03`.

No requirements for vendor.

TE05.01.04.01: The tester shall verify that the Patron Header Version value is `0x03`.

AS05.01.05: The biometric data block is digitally signed but not encrypted, and this shall be reflected by setting the value of the Signature Block Header (SBH) security options field to `b00001101`.

No requirements for vendor.

TE05.01.05.01: The tester shall verify that the SBH security option value is `b00001101`.

AS05.01.06: For fingerprint and facial records, the Biometrick Data Block (BDB) Format Owner shall be `0x001B` denoting M1, the INCITS Technical Committee on Biometrics.

No requirements for vendor.

TE05.01.06.01: The tester shall verify that the BDB Format Owner field contains `0x001B`.

AS05.01.07: For the mandatory fingerprint template on the PIV card, the BDB Format Type value shall be `0x0201`. For the optional facial image on the PIV card, the BDB Format Type value shall be `0x0501`.

No requirements for vendor.

TE05.01.07.01: The tester shall verify that the BDB Format Type field is 0x0201 for fingerprint template and 0x0501 for facial image.

AS05.01.08: The Creation Date in the PIV Patron Format (see Row 7 in Table 8 of SP80076) shall be the date of acquisition of the parent sample, encoded in eight bytes using a binary representation of "YYYYMMDDhhmmssZ". Each pair of characters (for example, "DD") is coded in 8 bits as an unsigned integer where the last byte is the binary representation of the ASCII character Z which is included to indicate that the time is represented in Coordinated Universal Time (UTC). The field "hh" shall code a 24 hour clock value.

No requirements for vendor.

TE05.01.08.01: The tester shall verify the date field is in compliance with the assertion.

AS05.01.09: The Validity Period in the PIV Patron Format (Row 8 in Table 8 of SP80076) contains two dates.

No requirements for vendor.

TE05.01.09.01: The tester shall verify that the headers contain two dates in compliance with the assertion.

AS05.01.10: Biometric Type field within the PIV Patron Format shall be 0x000008 for fingerprint template and shall be 0x000002 for facial images. The value for other biometric modalities shall be that given in CBEFF, 5.2.1.5. For modalities not listed there the value shall be 0x00.

No requirements for vendor.

TE05.01.10.01: The tester shall verify that the Biometric Type field contains 0x000008 for fingerprint images or templates and 0x000002 for facial images.

AS05.01.11: For the mandatory fingerprint template on the PIV card, the CBEFF Biometric Data Type encoding value shall be `b100xxxxx`, which corresponds to biometric data that has been processed. For the optional facial image on the PIV card, the CBEFF Biometric Data Type encoding value shall be `b001xxxxx`

No requirements for vendor.

TE05.01.11.01: The tester shall verify that the Biometric Data Type value is b100xxxxx for processed fingerprint template and b001xxxxx for facial image.

AS05.01.12: For all biometric data whether stored on a PIV card or otherwise retained by agencies the quality value shall be a signed integer between -2 and 100 per the text of INCITS 358. A value of -2 shall denote that assignment was not supported by the implementation; a value of -1 shall indicate that an attempt to compute a quality value failed. Values from 0 to 100 shall indicate an increased expectation that the sample will ultimately lead to a successful match. The zero value required by FACESTD shall be coded in this CBEFF field as -2.

VE05.01.12.01: The vendor shall provide the quality values for the biometric data.

TE05.01.12.01: The tester shall verify that the value of Biometric Data Quality is between -2 and 100 for fingerprint template.

TE05.01.12.02: The tester shall verify that the value of Biometric Data Quality is -2 for a facial image.

AS05.01.13: The Creator field in the PIV Patron Format contains 18 bytes of which the first K <= 17 bytes shall be ASCII characters, and the first of the remaining 18-K shall be a null terminator (zero).

VE05.01.13.01: The vendor shall provide the value of Creator field.

TE05.01.13.01: The tester shall verify the Creator field value.

AS05.01.14: The Data Type Encoding field in the PIV Patron Format shall contain the 25 bytes of the FASC-N component of the CHUID identifier.

VE05.01.14.01: The vendor shall provide the value for FASC-N.

TE05.01.14.01: The tester shall verify the FASC-N value.

AS05.01.15: The "Reserved for future use" field in the PIV Patron Format shall contain 0x00000000.

No requirement for vendor.

TE05.01.15.01: The tester shall verify the "Reserved for future use" field is 0x00000000.

5.2 Fingerprint Template for Storage on PIV Card

AS05.02.01: Both finger's template records shall be wrapped in a single CBEFF structure prior to storage on the PIV card.

VE05.02.01.01: The vendor shall specify in its documentation the CBEFF structure is constructed in accordance with this assertion.

TE05.02.01.01: The tester shall parse the biometric data container to verify this assertion.

Note: The CBEFF structure itself is tested in later assertions.

AS05.02.02: The fingerprint templates stored on the card are compliant to the MINUSTD profile specified in SP80076, Table 3.

VE05.02.02.01: The vendor shall specify in its documentation that the template generator generates templates in accordance with MINUSTD.

TE05.02.02.01: The tester shall verify that the resultant template is in compliance with the assertion.

AS05.02.03: The Format Identifier of the General Header Record shall be 0x464D5200.

No requirements for vendor.

TE05.02.03.01: The tester shall verify that the Format Identifier value is 0x464D5200.

AS05.02.04: The Version Number of the General Header Record shall be 0x20323000.

No requirements for vendor.

TE05.02.04.01: The tester shall verify that the Version Number is 0x20323000.

AS05.02.05: The length of the entire CBEFF wrapped record shall fit within the container size limits specified in SP80073.

VE05.02.05.01: The vendor shall specify the length of the entire container which includes CBEFF wrapped record.

TE05.02.05.01: The tester shall verify that the size of the container is within the limits specified in SP80073.

AS05.02.06: Both of the two fields ("Owner" and "Type") of the CBEFF Product Identifier shall be non-zero.

VE05.02.06.01: The vendor shall provide the Owner and Type of CBEFF product identifier.

TE05.02.06.01: The tester shall verify that the values are present and accurate.

AS05.02.07: The two most significant bytes of each of the two fields ("Owner" and "Type") of the CBEFF Product Identifier shall identify the vendor, and the two least significant bytes shall identify the version number of that supplier's minutiae detection algorithm.

VE05.02.07.01: The vendor shall specify the relevant version and vendor codes.

TE05.02.07.01: The tester shall verify the values specified in the documentation.

AS05.02.08: The Capture Equipment Complaince of the General Record Header shall be 1000b.

No requirements for vendor

TE05.02.08.01: The tester shall verify the Capture Equipment Complaince value is 1000b.

AS05.02.09: The Capture Equiement ID of the General Record Header is greater than zero.

VE05.02.09.01: The vendor shall specify the Capture Equipment ID value.

TE05.02.09.01: The tester shall verify the value is in accordance with vendor reporting.

AS05.02.10: The width on Size of Scanned Image in X Direction shall be the larger of the widths of the two input images. Similarly, the height on Size of Scanned Image in Y Direction shall be the larger of the heights of the two input images.

VE05.02.10.01: The vendor shall report width and height of the images whose fingerprint templates are stored on the card.

TE05.02.10.01: The tester shall verify the larger size of the two is recorded in the fields.

AS05.02.11: The Number of Views of the General Header Record shall be 2.

No requirements for vendor

TE05.02.11.01: The tester shall verify the Number of Views value is 2.

AS05.02.12: The Reserved Byte of the General Header Record shall be 0.

No requirements for vendor

TE05.02.12.01: The tester shall verify the Reserved Byte value is 0.

AS05.02.13: The View Number of the Single Finger View Record shall be 0.

No requirements for vendor

TE05.02.13.01: The tester shall verify the View Number value of the Single Finger View Record is 0.

AS05.02.14: The Impression Type of the Single Finger View Record shall be either 0 or 2.

VE05.02.14.01: The vendor shall specify if the live or non-live scane images were used.

TE05.02.14.01: The tester shall verify the value is either 0 or 2 and is consistent with vendor reporting.

AS05.02.15: The quality value of captured fingerprint images shall be computed using Error! Reference source not found. **and reported as Q = 20(6-NFIQ).**

VE05.02.15.01: The vendor shall specify the procedure used to calculate the quality value.

TE05.02.15.01: This assertion is externally tested.

AS05.02.16: The Number of Minutiae of Single Finger View Record is between 0 and 128.

No requirements for vendor.

TE05.02.16.01: The tester shall verify the Number of Minutiae is between 0 and 128.

AS05.02.17: Fingerprint templates shall be limited to minutiae of types "ridge ending" and "ridge bifurcation" unless it is not possible to reliably distinguish between a ridge ending and a bifurcation, in which case the category of "other" shall be assigned and ecoded as 00b.

No requirements for vendor.

TE05.02.17.01: The tester shall verify that the Minutiae Type is either 00b, 01b, or 10b.

AS05.02.18: All coordinates and angles for fingerprint minutiae shall be recorded with respect to the original finger image. They shall not be recorded with respect to any image processing sub-image(s) created during the template creation process.

VE05.02.18.01: The vendor shall specify in its documentation that the template generator generates templates in accordance with this assertion.

Note: This assertion is externally tested.

AS05.02.19: The mandatory value for Extended Data Block Length for MINUSTD template shall be zero.

No requirements for vendor.

TE05.02.19.01: The tester shall verify that the value of Extended Data Block Length is zero.

5.3 Facial Image Stored on PIV Card

AS05.03.01: All facial images must conform with the requirements in SP80076 Table 6, "INCITS 385 Profile for PIV Facial Images."

VE05.03.01.01: The vendor shall include documentation of the procedure by which facial images are enrolled and retained.

TE05.03.01.01: The tester shall review the documentation to verify compliance with the assertion.

AS05.03.02: If facial imagery is stored on the PIV card, the length of the entire record shall fit within the container size limits specified in SP80073.

VE05.03.02.01: The vendor shall include documentation of the procedure by which facial images are enrolled and retained.

TE05.03.02.01: The tester shall verify that the size of the record is such that it will be in compliance with the assertion.

AS05.03.03: PIV facial images shall conform to the Full Frontal Image Type defined in Section 8 of FACESTD.

VE05.03.03.01: The vendor shall include documentation of the procedure by which facial images are enrolled and retained.

TE05.03.03.01: This assertion is externally tested.

AS05.03.04: Facial image data shall be formatted in one of the two compression formats enumerated in Section 6.2 of FACESTD. Both whole-image and single-region-of-interest (ROI) compression are permitted.

VE05.03.04.01: The vendor shall include documentation of the procedure by which facial images are enrolled and retained.

TE05.03.04.01: This assertion is externally tested.

AS05.03.05: Facial images shall be compressed using a compression ratio no higher than 15:1. However, when facial images are stored on PIV cards, JPEG 2000 shall be used with ROI compression in which the innermost region shall be centered on the face and compressed at no more than 24:1.

VE05.03.05.01: The vendor shall include documentation of the procedure by which facial images are enrolled and retained.

TE05.03.05.01: This assertion is externally tested.

6. Signed Data Elements

6.1 Card Holder Unique Identifier

6.1.1 Asymmetric Signature Conformance

AS06.01.01: The CHUID buffer shall contain an Asymmetric digital signature of the CHUID object, which has been encoded as a Cryptographic Message Syntax external digital signature as defined in RFC 3852.

No requirement for vendor.

TE06.01.01.01: The tester shall validate that the CHUID data buffer contains a digital signature and has been formatted correctly as a CMS external signature as defined in RFC 3852.

AS06.01.02: The digital signature is implemented as a SignedData Type.

No requirement for vendor.

TE06.01.02.01: The tester shall validate that the CMS external digital signature has been implemented as a SignedData type.

AS06.01.03: The value of the version field of the SignedData content type shall be v3.

No requirement for vendor.

TE06.01.03.01: The tester shall validate the version of the SignedData type is version 3.

AS06.01.04: The digestAlgorithms field of the SignedData content type shall be in accordance with Table 3-3 of SP80078.

No requirement for vendor.

TE06.01.04.01: The tester shall validate that the digest algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-3 of SP80078.

AS06.01.05: The eContentType of the encapContentInfo shall be id-PIV-CHUIDSecurityObject (OID = 2.16.840.1.101.3.6.1).

No requirement for vendor.

TE06.01.05.01: The tester shall validate that eContentType of the encapContentInfo asserts the id-PIV-CHUIDSecurityObject OID.

AS06.01.06: The encapContentInfo of the SignedData content type shall omit the eContent field.

No requirement for vendor.

TE06.01.06.01: The tester shall validate that the eContent field has been omitted from the encapContentInfo.

AS06.01.07: The certificates field shall include only a single X.509 certificate which is used to verify the signature in the SignerInfo field.

No requirement for vendor.

TE06.01.07.01: The tester shall validate that there is a single X.509 certificate in the certificates field that can verify the digital signature in the SignerInfo.

AS06.01.08: The crls field from the SignedData content type shall be omitted.

No requirement for vendor.

TE06.01.08.01: The tester shall validate that the crls field has been omitted from the SignedData.

AS06.01.09: The SignerInfos in the SignedData content type shall contain only a single SignerInfo type.

No requirement for vendor.

TE06.01.09.01: The tester shall validate that only a single SignerInfo exists in the SignedData.

AS06.01.10: The SignerInfo type shall use the issuerAndSerialNumber choice for the sid and this shall correspond to the issuer and serialNumber fields found in the X.509 certificate for the entity that signed the CHUID.

No requirement for vendor.

TE06.01.10.01: The tester shall validate that the issuerAndSerialNumber choice has been used for the SignerIdentifier and it corresponds to the issuer and serialNumber fields found in the X.509 certificate for the entity that signed the CHUID.

AS06.01.11: The SignerInfo type shall specify a digestAlgorithm in accordance with Table 3-3 of SP 800-78.

No requirement for vendor.

TE06.01.11.01: The tester shall validate that the digest algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-3 of SP80078.

AS06.01.12: The signedAttrs of the SignerInfo shall include the MessageDigest (OID = 1.2.840.113549.1.9.4) attribute containing the hash computed over the concatenated content of the CHUID, excluding the asymmetric signature field.

No requirement for vendor.

TE06.01.12.01: The tester shall validate the presence of a MessageDigest attribute in the signed attributes.

TE06.01.12.02: The tester shall validate the value of the MessageDigest attribute against the hash of the concatenated content of the CHUID, excluding the asymmetric signature field.

AS06.01.13: The signedAttrs of the SignerInfo shall include the pivSigner-DN (OID = 2.16.840.1.101.3.6.5) attribute containing the subject name that appears in the X.509 certificate for the entity that signed the CHUID.

No requirement for vendor.

TE06.01.13.01: The tester shall validate the presence of a pivSigner-DN attribute in the signed attributes.

TE06.01.13.02: The tester shall validate the value of the pivSigner-DN attribute is the same as the subject name that appears in the certificate that signed the CHUID.

AS06.01.14: The signatureAlgorithm field specified in the SignerInfo field shall be in accordance with Table 3-4 of SP 800-78 and based on the PIV card expiration date in accordance with Table 3-3 of SP 800-78.

No requirement for vendor.

TE06.01.14.01: The tester shall validate that the signature algorithm is based on the expiration date of the certificate and is in accordance with Table 3-4 of SP80078.

AS06.01.15: The SignedData content type shall include the digital signature.

No requirement for vendor.

TE06.01.15.01: The tester shall validate that the SignedData content type includes the digital signature corresponding to the CHUID.

6.1.2 Certificate that signs the CHUID

In addition to the requirements from Section 7.2.1 the following shall be met.

AS06.01.16: The digital signature certificate used to sign the CHUID shall in the extKeyUsage assert id-PIV-content-signing (OID = 2.16.840.1.101.3.6.7).

No requirement for vendor.

TE06.01.16.01: The tester shall validate that the certificate that was used to sign the CHUID asserts the id-PIV-content-signing OID in the extended key usage extension.

AS06.01.17: The size of the public key for digital signature certificate used to sign the CHUID shall be determined by the expiration of the Card in accordance with Table 3-3 of SP 800-78.

No requirement for vendor.

TE06.01.17.01: The tester shall validate that the public key size is in accordance with Table 3-3 of SP80078.

6.2 Biometric Fingerprint

6.2.1 Asymmetric Signature Conformance

AS06.02.01: The CBEFF_SIGNATURE_BLOCK shall be encoded as a Cryptographic Message Syntax external digital signature as defined in RFC 3852.

No requirement for vendor.

TE06.02.01.01: The tester shall validate that the digital signature in the CBEFF_SIGNATURE_BLOCK has been formatted correctly as a CMS external signature as defined in RFC 3852.

AS06.02.02: The digital signature is implemented as a SignedData Type.

No requirement for vendor.

TE06.02.02.01: The tester shall validate that the CMS external digital signature has been implemented as a SignedData type.

AS06.02.03: The value of the version field of the SignedData content type shall be v1 or v3 based on whether the certificates field is omitted or not.

No requirement for vendor.

TE06.02.03.01: The tester shall validate the version of the SignedData type is version 1 or version 3 depending on whether the certificates field is omitted.

AS06.02.04: The digestAlgorithms field of the SignedData content type shall be in accordance with Table 3-3 of SP 800-78.

No requirement for vendor.

TE06.02.04.01: The tester shall validate that the digest algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-3 of SP80078.

AS06.02.05: The eContentType of the encapContentInfo shall be id-PIV-biometricObject (OID = 2.16.840.1.101.3.6.2).

No requirement for vendor.

TE06.02.05.01: The tester shall validate that eContentType of the encapContentInfo asserts the id-PIV-biometricObject OID.

AS06.02.06: The encapContentInfo of the SignedData content type shall omit the eContent field.

No requirement for vendor.

TE06.02.06.01: The tester shall validate that the eContent field has been omitted from the encapContentInfo.

AS06.02.07: If the signature on the fingerprint biometric was generated with a different key as the signature on the CHUID, the certificates field shall include only a single certificate in the SignerInfo field which can be used to verify the signature; else the certificates field shall be omitted.

VE06.02.07.01: The vendor shall state whether or not the same certificate was used to sign the CHUID and the Biometrics.

TE06.02.07.01: The tester shall validate that there is a single X.509 certificate in the certificates field that can verify the digital signature in the SignerInfo.

TE06.02.07.02: If the certificates field is omitted, the tester shall validate that the certificate in the SignedData for the CHUID can verify the digital signature in the SignerInfo.

AS06.02.08: The crls field from the SignedData content type shall be omitted.

No requirement for vendor.

TE06.02.08.01: The tester shall validate that the crls field has been omitted from the SignedData.

AS06.02.09: The signerInfos in the SignedData content type shall contain only a single SignerInfo type.

No requirement for vendor.

TE06.02.09.01: The tester shall validate that only a single SignerInfo exists in the SignedData.

AS06.02.10: The SignerInfo type shall use the issuerAndSerialNumber choice for the sid and this shall correspond to the issuer and serialNumber fields found in the X.509 certificate for the entity that signed the biometric data.

No requirement for vendor.

TE06.02.10.01: The tester shall validate that the issuerAndSerialNumber choice has been used for the SignerIdentifier and it corresponds to the to the issuer and serialNumber fields found in the X.509 certificate for the entity that signed the biometric data.

AS06.02.11: The SignerInfo type shall specify a digestAlgorithm in accordance with Table 3-3 of SP 800-78.

No requirement for vendor.

TE06.02.11.01: The tester shall validate that the digest algorithm in the SignerInfo is based on the expiration date of the PIV card and is in accordance with Table 3-3 of SP80078.

AS06.02.12: The signedAttrs of the SignerInfo shall include the MessageDigest (OID = 1.2.840.113549.1.9.4) attribute containing the hash of the concatenated CBEFF_HEADER and the STD_BIOMETRIC_RECORD.

No requirement for vendor.

TE06.02.12.01: The tester shall validate the presence of a MessageDigest attribute in the signed attributes.

TE06.02.12.02: The tester shall validate the value of the MessageDigest attribute against the hash of the concatenated CBEFF_HEADER and the STD_BIOMETRIC_RECORD.

AS06.02.13: The signedAttrs of the SignerInfo shall include the pivSigner-DN (OID = 2.16.840.1.101.3.6.5) attribute containing the subject name that appears in the X.509 certificate for the entity that signed the fingerprint biometric data.

No requirement for vendor.

TE06.02.13.01: The tester shall validate the presence of a pivSigner-DN attribute in the signed attributes.

TE06.02.13.02: The tester shall validate the value of the pivSigner-DN attribute is the same as the subject name that appears in the certificate that signed the biometric data.

AS06.02.14: The signedAttrs of the SignerInfo shall include the pivFASC-N (OID = 2.16.840.1.101.3.6.6) attribute containing the FASC-N of the PIV card.

No requirement for vendor.

TE06.02.14.01: The tester shall validate the presence of a pivFASC-N attribute in the signed attributes.

TE06.02.14.02: The tester shall validate the value of the pivFASC-N attribute is the same as the FASC-N that is present in the CHUID.

AS06.02.15: The signatureAlgorithm field specified in the SignerInfo field shall be in accordance with Table 3-4 of SP 800-78 and based on the PIV card expiration date in accordance with Table 3-3 of SP 800-78.

No requirement for vendor.

TE06.02.15.01: The tester shall validate that the signature algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-4 of SP80078.

AS06.02.16: The SignedData content type shall include the digital signature.

No requirement for vendor.

TE06.02.16.01: The tester shall validate that the SignedData content type includes the digital signature corresponding to the signed biometric data.

6.2.2 Certificate that signs the biometric fingerpint

In addition to the requirements from Section 7.2.1 the following shall be met.

AS06.02.17: The digital signature certificate used to sign PIV fingerprint biometric shall in the extKeyUsage assert id-PIV-content-signing (OID = 2.16.840.1.101.3.6.7).

No requirement for vendor.

TE06.02.17.01: The tester shall validate that the certificate that was used to sign the fingerprint biometric data asserts the id-PIV-content-signing OID in the extended key usage extension.

AS06.02.18: The size of the public key for digital signature certificate used to sign the biometrics shall be determined by the expiration of the Card in accordance with Table 3-3 of SP 800-78.

No requirement for vendor.

TE06.02.18.01: The tester shall validate that the public key size is in accordance with Table 3-3 of SP80078.

6.3 Biometric Facial Image

6.3.1 Asymmetric Signature Conformance

AS06.03.01: The CBEFF_SIGNATURE_BLOCK shall be encoded as a Cryptographic Message Syntax external digital signature as defined in RFC 3852.

No requirement for vendor.

TE06.03.01.01: The tester shall validate that the digital signature in the CBEFF_SIGNATURE_BLOCK has been formatted correctly as a CMS external signature as defined in RFC 3852.

AS06.03.02: The digital signature is implemented as a SignedData Type.

No requirement for vendor.

TE06.03.02.01: The tester shall validate that the CMS external digital signature has been implemented as a SignedData type.

AS06.03.03: The value of the version field of the SignedData content type shall be v1 or v3 based on whether the certificates field is omitted or not.

No requirement for vendor.

TE06.03.03.01: The tester shall validate the version of the SignedData type is version 1 or version 3 depending on whether the certificates field is omitted.

AS06.03.04: The digestAlgorithms field of the SignedData content type shall be in accordance with Table 3-3 of SP 800-78.

No requirement for vendor.

TE06.03.04.01: The tester shall validate that the digest algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-3 of SP80078.

AS06.03.05: The eContentType of the encapContentInfo shall be id-PIV-biometricObject (OID = 2.16.840.1.101.3.6.2).

No requirement for vendor.

TE06.03.05.01: The tester shall validate that eContentType of the encapContentInfo asserts the id-PIV-biometricObject OID.

AS06.03.06: The encapContentInfo of the SignedData content type shall omit the eContent field.

No requirement for vendor.

TE06.03.06.01: The tester shall validate that the eContent field has been omitted from the encapContentInfo.

AS06.03.07: If the signature on the facial image biometric was generated with a different key as the signature on the CHUID, the certificates field shall include only a single certificate in the SignerInfo field which can be used to verify the signature; else the certificates field shall be omitted.

No requirement for vendor.

TE06.03.07.01: The tester shall validate that there is a single X.509 certificate in the certificates field that can verify the digital signature in the SignerInfo.

TE06.03.07.02: If the certificates field is omitted, the tester shall validate that the certificate in the SignedData for the CHUID can verify the digital signature in the SignerInfo.

AS06.03.08: The crls field from the SignedData content type shall be omitted.

No requirement for vendor.

TE06.03.08.01: The tester shall validate that the crls field has been omitted from the SignedData.

AS06.03.09: The signerInfos in the SignedData content type shall contain only a single SignerInfo type.

No requirement for vendor.

TE06.03.09.01: The tester shall validate that only a single SignerInfo exists in the SignedData.

AS06.03.10: The SignerInfo type shall use the issuerAndSerialNumber choice for the sid and this shall correspond to the issuer and serialNumber fields found in the X.509 certificate for the entity that signed the biometric data.

No requirement for vendor.

TE06.03.10.01: The tester shall validate that the issuerAndSerialNumber choice has been used for the SignerIdentifier and it corresponds to the issuer and serialNumber fields found in the X.509 certificate for the entity that signed the biometric data.

AS06.03.11: The SignerInfo type shall specify a digestAlgorithm in accordance with Table 3-3 of SP 800-78.

No requirement for vendor.

TE06.03.11.01: The tester shall validate that the digest algorithm in the SignerInfo is based on the expiration date of the PIV card and is in accordance with Table 3-3 of SP80078.

AS06.03.12: The signedAttrs of the SignerInfo shall include the MessageDigest (OID = 1.2.840.113549.1.9.4) attribute containing the hash of the concatenated CBEFF_HEADER and the STD_BIOMETRIC_RECORD.

No requirement for vendor.

TE06.03.12.01: The tester shall validate the presence of a MessageDigest attribute in the signed attributes.

TE06.03.12.02: The tester shall validate the value of the MessageDigest attribute against the hash of the concatenated CBEFF_HEADER and the STD_BIOMETRIC_RECORD.

AS06.03.13: The signedAttrs of the SignerInfo shall include the pivSigner-DN (OID = 2.16.840.1.101.3.6.5) attribute containing the subject name that appears in the X.509 certificate for the entity that signed the biometric data.

No requirement for vendor.

TE06.03.13.01: The tester shall validate the presence of a pivSigner-DN attribute in the signed attributes.

TE06.03.13.02: The tester shall validate the value of the pivSigner-DN attribute is the same as the subject name that appears in the certificate that signed the biometric data.

AS06.03.14: The signedAttrs of the SignerInfo shall include the pivFASC-N (OID = 2.16.840.1.101.3.6.6) attribute containing the FASC-N of the PIV card.

No requirement for vendor.

TE06.03.14.01: The tester shall validate the presence of a pivFASC-N attribute in the signed attributes.

TE06.03.14.02: The tester shall validate the value of the pivFASC-N attribute is the same as the FASC-N that is present in the CHUID.

AS06.03.15: The signatureAlgorithm field specified in the SignerInfo field shall be in accordance with Table 3-4 of SP 800-78 and based on the PIV card expiration date in accordance with Table 3-3 of SP 800-78.

No requirement for vendor.

TE06.03.15.01: The tester shall validate that the signature algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-4 of SP80078.

AS06.03.16: The SignedData content type shall include the digital signature.

No requirement for vendor.

TE06.03.16.01: The tester shall validate that the SignedData content type includes the digital signature corresponding to the signed biometric data.

6.3.2 Certificate that signs the biometric facial image

In addition to the requirements from Section 7.2.1 the following shall be met.

AS06.03.17: The digital signature certificate used to sign PIV facial image biometric shall in the extKeyUsage assert id-PIV-content-signing (OID = 2.16.840.1.101.3.6.7).

No requirement for vendor.

TE06.03.17.01: The tester shall validate that the certificate that was used to sign the facial image biometric data asserts the id-PIV-content-signing OID in the extended key usage extension.

AS06.03.18: The size of the public key for digital signature certificate used to sign the biometrics shall be determined by the expiration of the card in accordance with Table 3-3 of SP 800-78.

No requirement for vendor.

TE06.03.18.01: The tester shall validate that the public key size is in accordance with Table 3-3 of SP80078.

6.4 Security Object

6.4.1 Data Integrity Check

AS06.04.01: The message digest produced as a result of a hash function on the contents of a data object buffer shall be identical to that data object's message digest contained in the security object.

No requirement for vendor.

TE06.04.01.01: The tester shall validate that the message digests for the various data objects present in the security object are identical to the message digest of the data object itself.

6.4.2 Asymmetric Signature Conformance

AS06.04.02: The security object buffer shall contain an asymmetric digital signature as specified in RFC (3852).

No requirement for vendor.

TE06.04.02.01: The tester shall validate that the digital signature has been formatted correctly as a CMS signature as defined in RFC (3852).

AS06.04.03: The digital signature is implemented as a SignedData Type.

No requirement for vendor.

TE06.04.03.01: The tester shall validate that the CMS digital signature has been implemented as a SignedData type.

AS06.04.04: The value of the version field of the SignedData content type shall be v1.

No requirement for vendor.

TE06.04.04.01: The tester shall validate the version of the SignedData type is version 1.

AS06.04.05: The digestAlgorithms field of the SignedData content type shall be in accordance with Table 3-7 of SP 800-78.

No requirement for vendor.

TE06.04.05.01: The tester shall validate that the digest algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-7 of SP80078.

AS06.04.06: The eContentType of the encapContentInfo shall be id-icao-ldsSecurityObject (OID = 1.3.27.1.1.1).

No requirement for vendor.

TE06.04.06.01: The tester shall validate that eContentType of the encapContentInfo asserts the id-icao-ldsSecurityObject OID.

AS06.04.07: The eContent of the encapContentsInfo field shall contain the encoded contents of the ldsSecurity object.

No requirement for vendor.

TE06.04.07.01: The tester shall validate that eContent of the encapContentInfo contains the contents of the ldsSecurity object.

AS06.04.08: The certificates field shall be omitted since it is included in the CHUID.

No requirement for vendor.

TE06.04.08.01: The tester shall validate that the certificates field has been omitted from the SignedData.

AS06.04.09: The digestAlgorithm field specified in the SignerInfo field is in accordance with Table 3-7 of SP 800-78.

No requirement for vendor.

TE06.04.09.01: The tester shall validate that the digest algorithm in the SignerInfo is based on the expiration date of the PIV card and is in accordance with Table 3-7 of SP80078.

AS06.04.10: The signatureAlgorithm field specified in the SignerInfo field shall be in accordance with Table 3-4 of SP 800-78 and based on the PIV card expiration date in accordance with Table 3-3 of SP 800-78.

No requirement for vendor.

TE06.04.10.01: The tester shall validate that the signature algorithm is based on the expiration date of the certificate and is in accordance with Table 3-4 of SP80078.

AS06.04.11: The SignedData content type shall include the digital signature.

No requirement for vendor.

TE06.04.11.01: The tester shall validate that the SignedData content type includes the digital signature corresponding to the signed security object.

6.4.3 Certificate that signs the Security Object

AS06.04.12: The card issuer's digital signature key used to sign the CHUID shall also be used to sign the security object.

No requirement for vendor.

TE06.04.12.01: The tester shall validate that the certificate in the SignedData for the CHUID can verify the digital signature in the SignerInfo.

7. Asymmetric Key Pairs

7.1 PIV Authentication Key

7.1.1 Certificate Profile Conformance

AS07.01.01: The signature field in the certificate shall specify an algorithm in the AlgorithmIdentifier in accordance with Table 3-4 of SP 800-78 and based on the certificate expiration date in accordance with Table 3-3 of SP 800-78.

VE07.01.01.01: The vendor shall specify in its documentation the algorithms used to sign certificates issued.

TE07.01.01.01: The tester shall validate that the signature algorithm is based on the expiration date of the certificate and is in accordance with Table 3-4 of SP80078.

AS07.01.02: If Rivest Shamir Adleman (RSA) with Probabilistic Signature Scheme (PSS) padding is used, the parameters field of the AlgorithmIdentifier type shall assert Secure Hash Algorithm (SHA) 256 (OID = 2.16.840.1.101.3.4.2.1). For the other RSA algorithms, the parameters field is populated with NULL. For Elliptic Curve Digital Signature Algorithm (ECDSA), the parameters field is absent.

VE07.01.02.01: The vendor shall specify in its documentation the permitted values of the AlgorithmIdentifier field based on the signature algorithm used to sign certificates issued.

TE07.01.02.01: The tester shall validate that the correctness of the values of the AlgorithmIdentifier fields.

AS07.01.03: The subjectPublicKeyInfo field shall assert an algorithm in the AlgorithmIdentifier in accordance with Table 3-5 of SP 800-78.

VE07.01.03.01: The vendor shall specify in its documentation the applicable algorithms that can be used to generate PIV authentication keys.

TE07.01.03.01: The tester shall validate that the algorithm used to generate PIV authentication keys are in accordance with Table 3-5 of SP80078.

AS07.01.04: If the public key algorithm is Elliptic Curve, then the EcpkParameters field uses either the namedCurve field populated with the appropriate OID from Table 3-6 of SP 800-78 or the implicitlyCA choice.

VE07.01.04.01: The vendor shall specify in its documentation the allowed values of the parameters field of the algorithm of the subjectPublicKeyInfo field as part of the PIV authentication certificate profile. These values shall be based on the algorithm used to generate the key pair.

TE07.01.04.01: The tester shall validate the correctness of the values of the parameters field of the algorithm of the subjectPublicKeyInfo field in the PIV authentication certificate issued by the vendor.

AS07.01.05: The keyUsage extension shall assert only the digitalSignature bit. No other bits shall be asserted.

VE07.01.05.01: The vendor shall specify in its documentation the assertion of the digitalSignature bit in the keyUsage extension as part of the PIV authentication certificate profile.

TE07.01.05.01: The tester shall validate the assertion of the digitalSignature bit in the keyUsage extension in the PIV authentication certificate issued by the vendor.

AS07.01.06: The policyIdentifier field in the certificatePolicies must assert id-fpki-common-authentication (OID = 2.16.840.1.101.3.2.1.3.13).

VE07.01.06.01: The vendor shall specify in its documentation the inclusion of the certificatePolicies extension which asserts the id-fki-common-authentication OID as part of the PIV authentication certificate profile.

TE07.01.06.01: The tester shall validate the presence of the id-fki-common-authentication OID in the certificatePolicies extension in the PIV authentication certificate issued by the vendor.

AS07.01.07: The authorityInfoAccess field shall contain an id-ad-ocsp accessMethod. The access location uses the Uniform Resource Identifier (URI) name form to specify the location of an Hypertext Transfer Protocol (HTTP) accessible Online Certificate Status Protocol (OCSP) Server distributing status information for this certificate.

VE07.01.07.01: The vendor shall specify in its documentation the inclusion of an id-ad-ocsp accessMethod in the authorityInfoAccess extension as part of the PIV authentication certificate profile. Additionally, the accessLocation for this accessMethod uses the URI name form to specify the location of an HTTP accessible OCSP server.

TE07.01.07.01: The tester shall validate the presence of an id-ad-ocsp accessMethod in the authorityInfoAccess extension in the PIV authentication certificate issued by the vendor. The tester shall also validate that the accessLocation for this accessMethod uses the URI name form and points to an HTTP accessible OCSP server.

AS07.01.08: The FASC-N shall be populated in the subjectAltName extension using the pivFASC-N attribute (OID = 2.16.840.1.101.3.6.6).

VE07.01.08.01: The vendor shall specify in its documentation the inclusion of the FASC-N in the subjectAltName extension as part of the PIV authentication certificate profile.

TE07.01.08.01: The tester shall validate the presence of the FASC-N in the subjectAltName extension in the PIV authentication certificate issued by the vendor.

AS07.01.09 The piv-interim extension (OID = 2.16.840.1.101.3.6.9.1) shall be present and contain an interim_indicator field which is populated with a Boolean value. This extension is not critical.

VE07.01.09.01: The vendor shall specify in its documentation the use of this extension as part of the PIV authentication certificate profile.

TE07.01.09.01: The tester shall validate that the piv-interim extension is present in the PIV authentication certificate issued by the vendor.

7.1.2 Key Pair and Certificate Conformance

AS07.01.10: The size of the public key for PIV authentication shall be determined by the expiration of the certificate in accordance with Table 3-1 of SP 800-78.

VE07.01.10.01: The vendor shall specify in its documentation the allowable public key size to be used while generating PIV authentication keys.

TE07.01.10.01: The tester shall validate that the public key size is in accordance with Table 3-1 of SP80078.

AS07.01.11: The public key present in the PIV authentication certificate correspond to the PIV authentication private key.

No requirement for vendor.

TE07.01.11.01: The tester shall validate that the public key present in the PIV authentication certificate is part of the key pair corresponding to the private key on the PIV card.

AS07.01.12: The FASC-N in the subjectAltName field in the PIV authentication certificate is the same as the FASC-N present in the CHUID.

No requirement for vendor.

TE07.01.12.01: The tester shall validate that the FASC-N in the subjectAltName field in the PIV authentication certificate is the same as the FASC-N present in the CHUID in the PIV card.

AS07.01.13: The expiration of the PIV authentication certificate is not beyond the expiration of the CHUID.

No requirement for vendor.

TE07.01.13.01: The tester shall validate that the expiration of the PIV authentication certificate is not beyond the expiration of the CHUID in the PIV card.

AS07.01.14: If the public key algorithm is RSA, the exponent shall be greater than or equal to 65,537.

VE07.01.14.01: The vendor shall specify in its documentation the size of the exponent permitted while generating an RSA key pair for PIV authentication.

TE07.01.14.01: The tester shall validate that the RSA public key exponent size is greater than or equal to 65,537.

7.2 Digital Signature Key

7.2.1 Certificate Profile Conformance

AS07.02.01: The signature field in the certificate shall specify an algorithm in the AlgorithmIdentifier in accordance with Table 3-4 of SP 800-78 and based on the certificate expiration date in accordance with Table 3-3 of SP 800-78.

VE07.02.01.01: The vendor shall specify in its documentation the algorithms used to sign certificates issued.

TE07.02.01.01: The tester shall validate that the signature algorithm is based on the expiration date of the certificate and is in accordance with Table 3-4 of SP80078.

AS07.02.02: If RSA with PSS padding is used, the parameters field of the AlgorithmIdentifier type shall assert SHA-256 (OID = 2.16.840.1.101.3.4.2.1). For the other RSA algorithms, the parameters field is populated with NULL. For ECDSA, the parameters field is absent.

VE07.02.02.01: The vendor shall specify in its documentation the permitted values of the AlgorithmIdentifier field based on the signature algorithm used to sign certificates issued.

TE07.02.02.01: The tester shall validate that the correctness of the values of the AlgorithmIdentifier fields.

AS07.02.03: The subjectPublicKeyInfo field shall assert an algorithm in the AlgorithmIdentifier in accordance with Table 3-5 of SP 800-78.

VE07.02.03.01: The vendor shall specify in its documentation the applicable algorithms that can be used to generate digital signature keys.

TE07.02.03.01: The tester shall validate that the algorithm used to generate digital signature keys are in accordance with Table 3-5 of SP80078.

AS07.02.04: If the public key algorithm is Elliptic Curve, then the EcpkParameters field uses either the namedCurve field populated with the appropriate OID from Table 3-6 of SP 800-78 or the implicitlyCA choice.

VE07.02.04.01: The vendor shall specify in its documentation the allowed values of the parameters field of the algorithm of the subjectPublicKeyInfo field as part of the digital signature certificate profile. These values shall be based on the algorithm used to generate the key pair.

TE07.02.04.01: The tester shall validate the correctness of the values of the parameters field of the algorithm of the subjectPublicKeyInfo field in the digital signature certificate issued by the vendor.

AS07.02.05: The keyUsage extension shall assert both the digitalSignature and nonRepudiation bits. No other bits shall be asserted.

VE07.02.05.01: The vendor shall specify in its documentation the assertion of the digitalSignature bit and the nonRepudiation bit in the keyUsage extension as part of the digital signature certificate profile.

TE07.02.05.01: The tester shall validate the assertion of the digitalSignature bit and the nonRepudiation bit in the keyUsage extension in the digital signature certificate issued by the vendor.

7.2.2 Key Pair and Certificate Conformance

AS07.02.06: The size of the public key for digital signature shall be determined by the expiration of the certificate in accordance with Table 3-1 of SP 800-78.

VE07.02.06.01: The vendor shall specify in its documentation the allowable public key size to be used while generating digital signature keys.

TE07.02.06.01: The tester shall validate that the public key size is in accordance with Table 3-1 of SP80078.

AS07.02.07: The public key present in the digital signature certificate corresponds to the digital signature private key.

No requirement for vendor.

TE07.02.07.01: The tester shall validate that the public key present in the digital signature certificate is part of the key pair corresponding to the private key on the PIV card.

AS07.02.08: The expiration of the digital signature certificate is not beyond the expiration of the CHUID.

No requirement for vendor.

TE07.02.08.01: The tester shall validate that the expiration of the digital signature certificate is not beyond the expiration of the CHUID in the PIV card.

AS07.02.09: If the public key algorithm is RSA, the exponent shall be greater than or equal to 65,537.

VE07.02.09.01: The vendor shall specify in its documentation the size of the exponent permitted while generating an RSA key pair for digital signatures.

TE07.02.09.01: The tester shall validate that the RSA public key exponent size is greater than or equal to 65,537.

7.3 Key Management Key

7.3.1 Certificate Profile Conformance

AS07.03.01: The signature field in the certificate shall specify an algorithm in the AlgorithmIdentifier in accordance with Table 3-4 of SP 800-78 and based on the certificate expiration date in accordance with Table 3-3 of SP 800-78.

VE07.03.01.01: The vendor shall specify in its documentation the algorithms used to sign certificates issued.

TE07.03.01.01: The tester shall validate that the signature algorithm is based on the expiration date of the certificate and is in accordance with Table 3-4 of SP80078.

AS07.03.02: If RSA with PSS padding is used, the parameters field of the AlgorithmIdentifier type shall assert SHA-256 (OID = 2.16.840.1.101.3.4.2.1). For the other RSA algorithms, the parameters field is populated with NULL. For ECDSA, the parameters field is absent.

VE07.03.02.01: The vendor shall specify in its documentation the permitted values of the AlgorithmIdentifier field based on the signature algorithm used to sign certificates issued.

TE07.03.02.01: The tester shall validate that the correctness of the values of the AlgorithmIdentifier fields.

AS07.03.03: The subjectPublicKeyInfo field shall assert an algorithm in the AlgorithmIdentifier in accordance with Table 3-5 of SP 800-78.

VE07.03.03.01: The vendor shall specify in its documentation the applicable algorithms that can be used to generate key management keys.

TE07.03.03.01: The tester shall validate that the algorithm used to generate key management keys are in accordance with Table 3-5 of SP80078.

AS07.03.04: If the public key algorithm is Elliptic Curve, then the EcpkParameters field uses either the namedCurve field populated with the appropriate OID from Table 3-6 of SP 800-78 or the implicitlyCA choice.

VE07.03.04.01: The vendor shall specify in its documentation the allowed values of the parameters field of the algorithm of the subjectPublicKeyInfo field as part of the key management certificate profile. These values shall be based on the algorithm used to generate the key pair.

TE07.03.04.01: The tester shall validate the correctness of the values of the parameters field of the algorithm of the subjectPublicKeyInfo field in the key management certificate issued by the vendor.

AS07.03.05: If the public key algorithm is RSA, then the keyUsage extension shall only assert the keyEncipherment bit.

VE07.03.05.01: The vendor shall specify in its documentation that certificates corresponding to RSA keys assert only the keyEncipherment bit in the keyUsage extension.

TE07.03.05.01: The tester shall validate that certificates corresponding to RSA keys assert only the keyEncipherment bit in the keyUsage extension.

AS07.03.06: If the public key algorithm is Elliptic Curve, then the keyUsage extension shall only assert the keyAgreement bit.

VE07.03.06.01: The vendor shall specify in its documentation that certificates corresponding to elliptic curve keys assert only the keyAgreement bit in the keyUsage extension.

TE07.03.06.01: The tester shall validate that certificates corresponding to elliptic curve keys assert only the keyAgreement bit in the keyUsage extension.

7.3.2 Key Pair and Certificate Conformance

AS07.03.07: The size of the public key for key management shall be determined by the expiration of the certificate in accordance with Table 3-1 of SP 800-78.

VE07.03.07.01: The vendor shall specify in its documentation the allowable public key size to be used while generating key management keys.

TE07.03.07.01: The tester shall validate that the public key size is in accordance with Table 3-1 of SP80078.

AS07.03.08: The public key present in the key management certificate corresponds to the key management private key.

No requirement for vendor.

TE07.03.08.01: The tester shall validate that the public key present in the key management certificate is part of the key pair corresponding to the private key on the PIV card.

AS07.03.09: If the public key algorithm is RSA, the exponent shall be greater than or equal to 65,537.

VE07.03.09.01: The vendor shall specify in its documentation the size of the exponent permitted while generating an RSA key pair for key management.

TE07.03.09.01: The tester shall validate that the RSA public key exponent size is greater than or equal to 65,537.

7.4 Card Authentication Key

7.4.1 Certificate Profile Conformance

AS07.04.01: The signature field in the certificate shall specify an algorithm in the AlgorithmIdentifier in accordance with Table 3-4 of SP 800-78 and based on the certificate expiration date in accordance with Table 3-3 of SP 800-78.

VE07.04.01.01: The vendor shall specify in its documentation the algorithms used to sign certificates issued.

TE07.04.01.01: The tester shall validate that the signature algorithm is based on the expiration date of the certificate and is in accordance with Table 3-4 of SP80078.

AS07.04.02: If RSA with PSS padding is used, the parameters field of the AlgorithmIdentifier type shall assert SHA-256 (OID = 2.16.840.1.101.3.4.2.1). For the other RSA algorithms, the parameters field is populated with NULL. For ECDSA, the parameters field is absent.

VE07.04.02.01: The vendor shall specify in its documentation the permitted values of the AlgorithmIdentifier field based on the signature algorithm used to sign certificates issued.

TE07.04.02.01: The tester shall validate that the correctness of the values of the AlgorithmIdentifier fields.

AS07.04.03: The subjectPublicKeyInfo field shall assert an algorithm in the AlgorithmIdentifier in accordance with Table 3-5 of SP 800-78.

VE07.04.03.01: The vendor shall specify in its documentation the applicable algorithms that can be used to generate card authentication keys.

TE07.04.03.01: The tester shall validate that the algorithm used to generate card authentication keys are in accordance with Table 3-5 of SP80078.

AS07.04.04: If the public key algorithm is Elliptic Curve, then the EcpkParameters field uses either the namedCurve field populated with the appropriate OID from Table 3-6 of SP 800-78 or the implicitlyCA choice.

VE07.04.04.01: The vendor shall specify in its documentation the allowed values of the parameters field of the algorithm of the subjectPublicKeyInfo field as part of the card authentication certificate profile. These values shall be based on the algorithm used to generate the key pair.

TE07.04.04.01: The tester shall validate the correctness of the values of the parameters field of the algorithm of the subjectPublicKeyInfo field in the card authentication certificate issued by the vendor.

AS07.04.05: The keyUsage extension shall assert only the digitalSignature bit. No other bits shall be asserted.

VE07.04.05.01: The vendor shall specify in its documentation the assertion of the digitalSignature bit in the keyUsage extension as part of the card authentication certificate profile.

TE07.04.05.01: The tester shall validate the assertion of the digitalSignature bit in the keyUsage extension in the card authentication certificate issued by the vendor.

AS07.04.06: The policyIdentifier field in the certificatePolicies must assert id-fpki-common-cardAuth (OID = 2.16.840.1.101.3.2.1.3.17).

VE07.04.06.01: The vendor shall specify in its documentation that the policyIdentifier field in certificatePolicies asserts the id-fpki-common-cardAuth OID.

TE07.04.06.01: The tester shall validate the policyIdentifier field in certificatePolicies has asserted the id-fpki-common-cardAuth OID.

AS07.04.07: The extKeyUsage extension shall assert id-PIV-cardAuth (OID = 2.16.840.1.101.3.6.8). This extension is critical.

VE07.04.07.01: The vendor shall specify in its documentation that the extKeyUsage extension asserts the id-PIV-cardAuth OID.

TE07.04.07.01: The tester shall validate the extKeyUsage asserts the id-PIV-cardAuth OID as a critical extension.

AS07.04.08: The authorityInfoAccess field shall contain an id-ad-ocsp accessMethod. The access location uses the URI name form to specify the location of an HTTP accessible OCSP Server distributing status information for this certificate.

VE07.04.08.01: The vendor shall specify in its documentation the inclusion of an id-ad-ocsp accessMethod in the authorityInfoAccess extension as part of the card authentication certificate profile. Additionally, the accessLocation for this accessMethod uses the URI name form to specify the location of an HTTP accessible OCSP server.

TE07.04.08.01: The tester shall validate the presence of an id-ad-ocsp accessMethod in the authorityInfoAccess extension in the card authentication certificate issued by the vendor. The tester shall also validate that the accessLocation for this accessMethod uses the URI name form and points to an HTTP accessible OCSP server.

AS07.04.09: The FASC-N shall be populated in the subjectAltName extension using the pivFASC-N attribute OID = 2.16.840.1.101.3.6.6).

VE07.04.09.01: The vendor shall specify in its documentation the inclusion of the FASC-N in the subjectAltName extension as part of the card authentication certificate profile.

TE07.04.09.01: The tester shall validate the presence of the FASC-N in the subjectAltName extension in the card authentication certificate issued by the vendor.

AS07.04.10: The piv-interim extension (OID = 2.16.840.1.101.3.6.9.1) shall be present contain an interim_indicator field which is populated with a Boolean value. This extension is not critical.

VE07.04.10.01: The vendor shall specify in its documentation the use of this extension as part of the card authentication certificate profile.

TE07.04.10.01: The tester shall validate that the piv-interim extension is present in the card authentication certificate issued by the vendor.

7.4.2 Key Pair and Certificate Conformance

AS07.04.11: The size of the public key for card authentication shall be determined by the expiration of the certificate in accordance with Table 3-1 of SP 800-78.

VE07.04.11.01: The vendor shall specify in its documentation the allowable public key size to be used while generating card authentication keys.

TE07.04.11.01: The tester shall validate that the public key size is in accordance with Table 3-1 of SP80078.

AS07.04.12: The public key present in the card authentication certificate correspond to the card authentication private key.

No requirement for vendor.

TE07.04.12.01: The tester shall validate that the public key present in the card authentication certificate is part of the key pair corresponding to the private key on the PIV card.

AS07.04.13: The FASC-N in the subjectAltName field in the card authentication certificate is the same as the FASC-N present in the CHUID.

No requirement for vendor.

TE07.04.13.01: The tester shall validate that the FASC-N in the subjectAltName field in the card authentication certificate is the same as the FASC-N present in the CHUID in the PIV card.

AS07.04.14: If the public key algorithm is RSA, the exponent shall be greater than or equal to 65,537.

VE07.04.14.01: The vendor shall specify in its documentation the size of the exponent permitted while generating an RSA key pair for card authentication.

TE07.04.14.01: The tester shall validate that the RSA public key exponent size is greater than or equal to 65,537.

8. BER-TLV Test Assertions

Assumptions:

1.0	When the length of the value field is between 0 and 127 bytes, the length field consists of a single byte where bit 8 is set to 0 and bits 7 to 1 encode the number of bytes in the value field. When the length of the value field is greater than 127 bytes, the length field consists of two or more bytes. The first byte is '81', '82', '83' or '84' where the low order nibble of each of these possible first-byte values (1, 2, 3, or 4 respectively) encodes the number of subsequent and remaining bytes in the length field. These subsequent and remaining bytes are taken together in order to be a big-endian integer encoding the number of bytes in the value field. Table 8-1 shows the encoding of the length field.
1.1	Each BER-TLV tag is encoded as three bytes.
1.2	Each data object returned is appended with a 2 byte status word.
1.3	All variable length value fields can have zero lengths, which will result in a tag length field being immediately followed by the next tag, if applicable.
1.4	The final byte of the command string can be set to 0x00 to retrieve an entire data object regardless of the size of that object.

Number of Bytes in the Length Field	First Byte	Subsequent Bytes	Length of the Value Field
1 byte	'00' to '7F'	None	0 to 127
2 byte	'81'	'00' to 'FF'	0 to 255
3 byte	'82'	'0000' to 'FFFF'	0 to 65,535
4 byte	'83'	'000000' to 'FFFFFF'	0 to 16,777,215
5 byte	'84'	'00000000' to 'FFFFFFFF'	0 to 4,294,967,295

Table 8-1. Encoding of Length Field

8.1 "Card Capabilities Container" Data Object

Purpose	Confirms that the CCC of the PIV card Application conforms to the PIV data model requirements as per Appendix A of SP80073.
Reference(s)	1. SP80073, Appendix A 2. TE04.01.01.01 3. TE04.02.01.01 4. TE04.02.01.02

Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CCC is present on the PIV card.
Test Scenario	1. Set cardHandle := <<valid card handle>>. 2. Set OID := <<CCC (2.16.840.1.101.3.7.1.219.0)>>. 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Read and parse the byte array in accordance with BER-TLV format.
Expected Result(s)	1. The size of the byte array does not exceed 297 bytes. 2. All mandatory tags in CCC table are present. 3. The values of the available tags conform with the vendor provided data.

8.2 "Card Holder Unique Identifier" Data Object

Purpose	Confirms that the CHUID of the PIV card Application conforms to the PIV data model requirements as per Appendix A of SP80073.
Reference(s)	1. SP80073, Appendix A 2. TE04.01.01.01 3. TE04.03.01.01
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Read and parse the byte array in accordance with BER-TLV format.
Expected Result(s)	1. The size of the byte array does not exceed 3395 bytes. 2. All mandatory tags in CHIID table are present. 3. Expiration date is encoded as YYYYMMDD. 4. Expriation date is within the next five years.

8.3 "X.509 Certificate for PIV Authentication" Data Object

Purpose	Confirms that the X.509 Certificate for PIV Authentication of the PIV card Application conforms to the PIV data model requirements as per Appendix A of SP80073
Reference(s)	1. SP80073, Appendix A 2. TE04.01.01.01
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid X.509 certificate for PIV authentication object is present on the PIV card.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<PIV Authentication Certificate (2.16.840.1.101.3.7.2.1.1)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Read and parse the byte array in accordance with BER-TLV format.
Expected Result(s)	1. The size of the byte array does not exceed 1905 bytes. 2. All mandatory tags in "X.509 Certificate for PIV Authentication" table are present.

8.4 "Card Holder Fingerprints" Data Object

Purpose	Confirms that the "Card Holder Fingerprints" data object of the PIV card Application conforms to the PIV data model requirements as per Appendix A of SP80073
Reference(s)	1. SP80073, Appendix A 2. TE04.01.01.01 3. TE04.04.01.01 4. TE04.04.02.01
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprints object is present on the PIV card.

Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Read and parse the byte array in accordance with BER-TLV format.
Expected Result(s)	1. The size of the byte array does not exceed 4006 bytes. 2. All mandatory tags in "Card Holder Fingerprints" table are present.

8.5 "Printed Information" Data Object

Purpose	Confirms that the "Printed Information" Data Object of the PIV card Application conforms to the PIV data model requirements as per Appendix A of SP80073
Reference(s)	1. SP80073, Appendix A 2. TE04.01.01.01
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid printed information is stored on the PIV card.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Printed Information (2.16.840.1.101.3.7.2.48.1)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Read and parse the byte array in accordance with BER-TLV format.
Expected Result(s)	1. The size of the byte array does not exceed 120 bytes. 2. All mandatory tags in "Printed Information" table are present.

8.6 "Card Holder Facial Image" Data Object

Purpose	Confirms that the "Card Holder Facial Image" data object of the PIV card Application conforms to the PIV data model requirements as per Appendix A of SP80073.

Reference(s)	1. SP80073, Appendix A 2. TE04.01.01.01 3. TE04.05.01.01 4. TE04.05.02.01
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image object is present on the PIV card.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Read and parse the byte array in accordance with BER-TLV format.
Expected Result(s)	1. The size of the byte array does not exceed 12710 bytes. 2. All mandatory tags in "Card Holder Facial Image" table are present.

8.7 "X.509 Certificate for Digital Signature" Data Object

Purpose	Confirms that the X.509 Certificate for Digital Signature of the PIV card Application conforms to the PIV data model requirements as per Appendix A of SP80073
Reference(s)	1. SP80073, Section 7.1.2 2. TE04.01.01.01
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid X.509 certificate for digital signature object is present on the PIV card.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Digital Signature Certificate (2.16.840.1.101.3.7.2.1.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Read and parse the byte array in accordance with BER-TLV format.

Expected Result(s)	1. The size of the byte array does not exceed 1905 bytes. 2. Al mandatory tags in "X.509 Certificate for Digital Signature" table are present.

8.8 "X.509 Certificate for Key Management" Data Object

Purpose	Confirms that the X.509 Certificate for Key Management of the PIV card Application conforms to the PIV data model requirements as per Appendix A of SP80073.
Reference(s)	1. SP80073, Appendix A 2. TE04.01.01.01
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid X.509 certificate for key management object is present on the PIV card.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Key Management Certificate (2.16.840.1.101.3.7.2.1.2)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Read and parse the byte array in accordance with BER-TLV format.
Expected Result(s)	1. The size of the byte array does not exceed 1905 bytes. 2. All mandatory tags in "X.509 Certificate for Key Management" table are present.

8.9 "X.509 Certificate for Card Authentication" Data Object

Purpose	Confirms that the X.509 Certificate for Card Authentication of the PIV card Application conforms to the PIV data model requirements as per Appendix A of SP80073
Reference(s)	1. SP80073, Appendix A 2. TE04.01.01.01

Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid X.509 certificate for card authentication object is present on the PIV card.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Authentication Certificate (2.16.840.1.101.3.7.2.5.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Read and parse the byte array in accordance with BER-TLV format.
Expected Result(s)	1. The size of the byte array does not exceed 1905 bytes. 2. All mandatory tags in "X.509 Certificate for Card Authentication" table are present.

8.10 "Security Object" Data Object

Purpose	Confirms that the "Security Object" data object of the PIV card Application conforms to the PIV data model requirements as per Appendix A of SP80073
Reference(s)	1. SP80073, Section 7.1.2 2. TE04.01.01.01 3. TE04.06.01.01
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid security object is present on the PIV card.

Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Security Object (2.16.840.1.101.3.7.2.144.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Read and parse the byte array in accordance with BER-TLV format. 5. Parse the tag 0xBA to extract the Data Groups to Container ID mapping instances. 6. Verify that the PIV data containers exist on the card by selecting each container.
Expected Result(s)	1. From Step 4: The size of the byte array does not exceed 1008 bytes. 2. From Step 4: All mandatory tags in "Security Object" table are present. 3. From Step 5: Verify that all data containers found in the mapping are actually present in the card by performing a select on each container (with appropriate authorization) and expecting '90 00' in all cases.

9. Biometric Data Object Test Assertions

The test assertions documented in this section relate to testing the Card Holder Fingerprint object and Facial Image on the PIV card for conformance to the common CBEFF Patron Format for PIV specification (Table 8 of SP80076) as well as to INCITS 378 Profile for PIV card Templates (Table 3 of SP80076) and INCITS 385 profile for Facial Image respectively.

9.1 CBEFF Patron Format for Fingerprint Template

9.1.1 CBEFF Structure for Fingerprint Template

Purpose	Validates that the CBEFF structure generated complies with SP80076 Table 7, "Simple CBEFF Structure".
References(s)	1. SP80076, Table 7. 2. AS05.01.01 3. AS05.01.03
Precondition(s)	1. All templates have been generated and stored on the PIV card. 2. A valid PIV card is inserted into the contact reader. 3. A valid PC/SC connection exists between the test application and the contact reader. 4. The test application is currently connected to the card application which is accessible through card handle. 5. A valid Card Holder Fingerprints object is present on the PIV card. 6. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the BDB Length field of CBEFF Header. 5. Extract the SB Length field of CBEFF Header.
Expected Result(s)	1. From step 4: BDB Length field is non-zero and the recorded length match the actual length. 2. From step 5: SB Length field is non-zero and the recorded length match the actual length. 3. The Card Holder Fingerprint object length is equal to CBEFF Header length + BDB Length + SB Length.

9.1.2 CBEFF Header for Fingerprint Template

9.1.2.1 Patron Header Version

Purpose	Validates that the CBEFF field "Patron Header Version" complies with SP80076 Table 8, "Patron Format PIV Specification"

References(s)	1. SP80076, Table 8. 2. AS05.01.02 3. AS05.01.04
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Patron Header Version field (based on its position) in CBEFF Header.
Expected Result(s)	The Patron Header Version field has a value of 0x03.

9.1.2.2 SBH Security Option

Purpose	Validates that the biometric data block on the PIV card is digitally signed but not encrypted.
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.05
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the SBH Security Option field (based on its position) in CBEFF Header.
Expected Result(s)	The SBH Security Options field has a value of b00001101.

9.1.2.3 BDB Format Owner Values

Purpose	Validates that BDB Format Owner is set to a value of 0x001B denoting M1, the INCITS Technical Committee on Biometrics.
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.03 4. AS05.01.06
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the BDB Format Owner field (based on its position) in CBEFF Header.
Expected Result(s)	The BDB Format Owner field has a value of 0x001B.

9.1.2.4 BDB Format Type

Purpose	Validates that for mandatory fingerprint minutiae template data stored on a PIV card, the BDB Format Type is set to a value of 0x0201.
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.03 4. AS05.01.07
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.

Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the BDB Format Type field (based on its position) in CBEFF Header.
Expected Result(s)	The BDB Format Type field has a value of 0x0201.

9.1.2.5 Biometric Creation Date

Purpose	Validates that the creation date in the PIV Patron Format is encoded in 8 bytes using a binary representation of "YYYYMMDDhhmmssZ".
References(s)	1. SP80076, Table 8. 2. AS05.01.02 3. AS05.01.03 4. AS05.03.08
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Biometric Creation Date field (based on its position) in CBEFF Header.
Expected Result(s)	The Biometric Creation Date is in the YYYYMMDDhhmmssZ format.

9.1.2.6 Validity Period Dates

Purpose	Validates that the Validity Period in the PIV Patron Format contains two dates encoded in the same format as expected in 9.1.2.5 above.
References(s)	1. SP80076, Table 8 2. AS05.01.02 3. AS05.01.03 4. AS05.01.09

Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Validity Period field (based on its position) in CBEFF Header.
Expected Result(s)	The Validity Period contains two dates and each is encoded in the YYYYMMDDhhmmssZ format.

9.1.2.7 Biometric Type Values

Purpose	Validates that Biometric Type has the value 0x000008
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.03 4. AS05.01.10
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the Biometric Type field of CBEFF Header. 5. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Biometric Type field (based on its position) in CBEFF Header.
Expected Result(s)	The value of the Biometric Type field for the fingerprint template is 0x000008

9.1.2.8 Biometric Data Type

Purpose	Validates that for the mandatory minutia PIV card templates, the CBEFF biometric data type encoding value shall be `b100xxxxx`, which corresponds to biometric data that has been processed.
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.11
Precondition(s)	1. All required sample finger images have been recorded. 2. All templates have been generated and stored on the PIV card. 3. A valid PIV card is inserted into the contact reader. 4. A valid PC/SC connection exists between the test application and the contact reader. 5. The test application is currently connected to the card application which is accessible through card handle. 6. A valid Card Holder Fingerprints object is present on the PIV card. 7. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the Biometric Data Type field of CBEFF Header. 5. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Biometric Data Type field (based on its position) in CBEFF Header.
Expected Result(s)	The Biometric Data Type field has a value of b100xxxxx.

9.1.2.9 Biometric Data Quality

Purpose	Validates that the biometric data quality field carries valid values
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.12
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.

Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the Biometric Data Quality field of CBEFF Header. 5. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Biometric Data Quality field (based on its position) in CBEFF Header.
Expected Result(s)	The Biometric Data Quality field has a value between -2 and 100.

9.1.2.10 Creator Field Value

Purpose	Validates that the Creator field in the PIV Patron Format contains 18 bytes of which the first K <= 17 bytes shall be ASCII characters, and the first of the remaining 18-K shall be a null terminator (zero).
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.13
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Creator field (based on its position) in CBEFF Header. Start the following procedure at the first byte of the Creator field: For bytes 0 to 16, check if byte represents an ASCII character If yes, continue. Else, check if zero. If no, fail. Else iterate through remaining bytes in field and fail iff non-zero. For bytes 17 through the end of the field, iterate through all bytes and fail iff non-zero
Expected Result(s)	Procedure completes without failing.

9.1.2.11 FASC-N Value

Purpose	The FASC-N field in the PIV Patron Format shall contain the 25 bytes of the FASC-N component of the CHUID identifier.

References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.03 4. AS05.01.14
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the FASC-N field of CBEFF Header. 5. Set cardHandle := <<valid card handle>> 6. Set OID := <<Card Holder Unique Identifier (2.16.840.1.101.3.7.2.48.0)>> 7. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 8. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the FASC-N field (based on its position) in CBEFF Header.
Expected Result(s)	The FASC-N field in CBEFF header is the same as the one extracted from the CHUID.

9.1.2.12 Reserved Field Value

Purpose	Validates that the “Reserved for Future Use” field is equal to 0x00000000.
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.15
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.

Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the “Reserved for Future Use” field (based on its position) in CBEFF Header.
Expected Result(s)	The field has a value of 0x00000000.

9.2 CBEFF Patron Format for Facial Image

9.2.1 CBEFF Structure for Facial Image

Purpose	Validates that the CBEFF structure generated complies with SP80076 Table 7, "Simple CBEFF Structure".
References(s)	1. SP80076, Table 7. 2. AS05.01.01 3. AS05.01.03
Precondition(s)	1. All templates have been generated and stored on the PIV card. 2. A valid PIV card is inserted into the contact reader. 3. A valid PC/SC connection exists between the test application and the contact reader. 4. The test application is currently connected to the card application which is accessible through card handle. 5. A valid Card Holder Facial Image object is present on the PIV card. 6. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the BDB Length field of CBEFF Header. 5. Extract the SB Length field of CBEFF Header.
Expected Result(s)	1. From step 4: BDB Length field is non-zero and the recorded length match the actual length. 2. From step 5: SB Length field is non-zero and the recorded length match the actual length. 3. The Card Holder Facial Image object length is equal to CBEFF Header length + BDB Length + SB Length.

9.2.2 CBEFF Header for Facial Image

9.2.2.1 Patron Header Version

Purpose	Validates that the CBEFF field "Patron Header Version" complies with SP80076 Table 8, "Patron Format PIV Specification"
References(s)	1. SP80076, Table 8. 2. AS05.01.02 3. AS05.01.04

Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Facial Image object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Patron Header Version field (based on its position) in CBEFF Header.
Expected Result(s)	The Patron Header Version field has a value of 0x03.

9.2.2.2 SBH Security Option

Purpose	Validates that the biometric data block on the PIV card is digitally signed but not encrypted.
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.05
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Facial Image object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the SBH Security Option field (based on its position) in CBEFF Header.
Expected Result(s)	The SBH Security Options field has a value of b00001101.

9.2.2.3 BDB Format Owner Values

Purpose	Validates that BDB Format Owner is set to a value of 0x001B denoting M1, the INCITS Technical Committee on Biometrics.
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.03 4. AS05.01.06
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Facial Image object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the BDB Format Owner field (based on its position) in CBEFF Header.
Expected Result(s)	The BDB Format Owner field has a value of 0x001B.

9.2.2.4 BDB Format Type

Purpose	Validates that for optional facial image data stored on a PIV card, the BDB Format Type is set to a value of 0x0501.
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.03 4. AS05.01.07
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Facial Image object is present on the PIV card. 5. Security conditions to read the object are met.

Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the BDB Format Type field (based on its position) in CBEFF Header.
Expected Result(s)	The BDB Format Type field has a value of 0x0501.

9.2.2.5 Biometric Creation Date

Purpose	Validates that the creation date in the PIV Patron Format is encoded in 8 bytes using a binary representation of "YYYYMMDDhhmmssZ".
References(s)	1. SP80076, Table 8. 2. AS05.01.02 3. AS05.01.03 4. AS05.01.08
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Facial Image object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Biometric Creation Date field (based on its position) in CBEFF Header.
Expected Result(s)	The Biometric Creation Date is in the YYYYMMDDhhmmssZ format.

9.2.2.6 Validity Period Dates

Purpose	Validates that the Validity Period in the PIV Patron Format contains two dates encoded in the same format as expected in 9.2.2.5 above.
References(s)	1. SP80076, Table 8 2. AS05.01.02 3. AS05.01.03 4. AS05.01.09

Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Facial Image object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Validity Period field (based on its position) in CBEFF Header.
Expected Result(s)	The Validity Period contains two dates and each is encoded in the YYYYMMDDhhmmssZ format.

9.2.2.7 Biometric Type Values

Purpose	Validates that Biometric Type has the value 0x000002
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.03 4. AS05.01.10
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Facial Image object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the Biometric Type field of CBEFF Header. 5. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Biometric Type field (based on its position) in CBEFF Header.
Expected Result(s)	The value of the Biometric Type field for the facial image is 0x000002

9.2.2.8 Biometric Data Type

Purpose	Validates that the CBEFF biometric data type encoding value shall be `b001xxxxx`, which corresponds to the raw biometric data.
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.11
Precondition(s)	1. All required sample finger images have been recorded. 2. All templates have been generated and stored on the PIV card. 3. A valid PIV card is inserted into the contact reader. 4. A valid PC/SC connection exists between the test application and the contact reader. 5. The test application is currently connected to the card application which is accessible through card handle. 6. A valid Card Holder Facial Image object is present on the PIV card. 7. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the Biometric Data Type field of CBEFF Header. 5. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Biometric Data Type field (based on its position) in CBEFF Header.
Expected Result(s)	The Biometric Data Type field has a value of b001xxxxx.

9.2.2.9 Biometric Data Quality

Purpose	Validates that the biometric data quality field carries valid values
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.12
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Facial Image object is present on the PIV card. 5. Security conditions to read the object are met.

Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the Biometric Data Quality field of CBEFF Header. 5. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Biometric Data Quality field (based on its position) in CBEFF Header.
Expected Result(s)	The Biometric Data Quality field is -2.

9.2.2.10 Creator Field Value

Purpose	Validates that the Creator field in the PIV Patron Format contains 18 bytes of which the first K <= 17 bytes shall be ASCII characters, and the first of the remaining 18-K shall be a null terminator (zero).
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.13
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Facial Image object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the Creator field (based on its position) in CBEFF Header. Start the following procedure at the first byte of the Creator field: For bytes 0 to 16, check if byte represents an ASCII character If yes, continue. Else, check if zero. If no, fail. Else iterate through remaining bytes in field and fail iff non-zero. For bytes 17 through the end of the field, iterate through all bytes and fail if non-zero
Expected Result(s)	Procedure completes without failing.

9.2.2.11 FASC-N Value

Purpose	The FASC-N field in the PIV Patron Format shall contain the 25 bytes of the FASC-N component of the CHUID identifier.
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.03 4. AS05.01.14
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Facial Image object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the FASC-N field of CBEFF Header. 5. Set cardHandle := <<valid card handle>> 6. Set OID := <<Card Holder Unique Identifier (2.16.840.1.101.3.7.2.48.0)>> 7. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 8. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the FASC-N field (based on its position) in CBEFF Header.
Expected Result(s)	The FASC-N field in CBEFF header is the same as the one extracted from the CHUID.

9.2.2.12 Reserved Field Value

Purpose	Validates that the “Reserved for Future Use” field is equal to 0x00000000.
References(s)	1. SP80076 2. AS05.01.02 3. AS05.01.03 4. AS05.01.15

Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Facial Image object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, extract the first 88 bytes (CBEFF Header) and read the “Reserved for Future Use” field (based on its position) in CBEFF Header.
Expected Result(s)	The field has a value of 0x00000000.

9.3 Fingerprint Template

9.3.1 General Record Header Conformance

Purpose	Verify that the General Record Header of the fingerprint template conforms to specifications in Table 3 of SP80076.
Reference(s)	1. SP80076, Table 3 2. AS05.01.03 3. AS05.02.02 4. AS05.02.03 5. AS05.02.04 6. AS05.02.05 7. AS05.02.06 8. AS05.02.07 9. AS05.02.08 10. AS05.02.09 11. AS05.02.10 12. AS05.02.11 13. AS05.02.12
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, parse the first 88 bytes (CBEFF Header) to obtain the BDB and SB lengths and then continue on to parse the General Record Header of the BDB a.Extract contents of Format Identifier. b.Extract contents of Version Number. c.Extract contents of Record Length field. d.Extract contents of CBEFF Product Identifier Owner e.Extract contents of CBEFF Product Identifier Type f.Extract contents of Capture Equipment Compliance field. g.Extract contents of Capture Equipment ID h.Extract contents of “Scanned Image in X Direction” field i.Extract contents of “Scanned Image in Y Direction” field j.Extract contents of X (horizontal) resolution. k.Extract contents of Y (vertical) resolution. l.Extract contents of “Number of Finger Views” field. m.Extract contents of Reserved Byte.

Expected Result(s)	The expected values for each of the fields parsed in Step 4 above are given below: a. Format Identifier has a value 0x4646D5200 b. Version Number has a value of 0x20323030 c. Record Length shall have a value less than the length specified in d. & e. Both these fields shall be non-zero The two most significant bytes shall identify the vendor while the least two significant bytes identifier the version number of the minutiae detection algorithm. f. Capture Equipment Compliance has a value of 1000b g. Capture Equipment ID has a non-null value. h & i. These values shall be non-zero and shall be obtained from biometric enrollment records. j & k. X and Y resolution has a value of 197 l. Number of Finger Views is 2 m. Reserved Byte value is zero

9.3.2 View Header Conformance

Purpose	Verify that the View Header of the BDB conforms to specifications in Table 3 of SP80076.
Reference(s)	1. SP80076, Table 3 2. AS05.01.03 3. AS05.02.02 4. AS05.02.13 5. AS05.02.14 6. AS05.02.16
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.

Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, parse the first 88 bytes (CBEFF Header) to obtain the BDB length and then continue on to parse the View Header of the BDB a. Extract Finger View Header field b. Extract contents of Finger Position. c. Extract contents of View Number. d. Extract contents of Impression Type. e. Extract contents of Finger Quality. f. Extract contents of Number of Minutiae. 5.Repeat steps 4a through 4f for the second view header.
Expected Result(s)	The expected values for each of the fields parsed in Step 4 above are given below: a. Finger View Header shall have the value 'A' b. Finger Position value shall be between 0 and 14. c. View Number shall be 0 if there is only one minutiae record for a finger. d. Impression Type is 0 or 2. e. Finger Quality value shall be between 60 and 100. f. Number of Minutiae value shall be between 0 and 128. The expected values for each of the fields parsed in Step 5 is the same as above.

9.3.3 Fingerprint Minutiae Data

Purpose	Verify that each instance of Fingerprint Minutiae data conforms to specifications in Table 3 of SP80076.
Reference(s)	1. SP80076, Table 3 2. AS05.02.02 3. AS05.02.17 4. AS05.02.19
Precondition(s)	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid Card Holder Fingerprints object is present on the PIV card. 5. Security conditions to read the object are met.

Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value field for tag 0xBC, parse the first 88 bytes (CBEFF Header) to obtain the BDB length and then continue on to parse the Minutiae data instances following the View Header of the BDB. a.Extract contents of Minutiae Type. b.Extract contents of Minutiae Position. c.Extract contents of Minutiae Angle. d.Extract contents of Minutiae Quality. e.Extract contents of Extended Block Length. 5. Repeat steps 4a to 4e for each Minutiae Data instance.
Expected Result(s)	The expected values for each of the fields parsed in Step 4 and 5 above are given below: a. Minutiae Type value shall be either 01b or 10b (Ridge Ending or Ridge Bifurcation). b. Minutiae Position shall be one of the valid X,Y Coordinate position in the original image. c. Minutiae Angle value shall be between 0 and 179. d. Minutiae Quality shall be between 0 and 100 e. Extended Data Block Length shall be 0

9.4 Facial Image on PIV Card

9.4.1 Facial Image Header Conformance

Purpose	Verify that the Record Header of the facial image is conformant to the PIV profile presented in Table 6 of SP80076.
Reference(s)	1. SP80076, Table 6 2. AS05.03.01 3. AS05.03.02
Precondition(s)	1. All required sample facial image is stored on the PIV card. 2. A valid PIV card is inserted into the contact reader. 3. A valid PC/SC connection exists between the test application and the contact reader. 4. The test application is currently connected to the card application which is accessible through card handle. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the contents of Facial Image Header Record. 5. Extract contents of Format Identifier. 6. Extract contents of Version Number. 7. Extract contents of Record Length. 8. Extract contents of Number of Facial Images. 9. Extract contents of Number of Feature Points.
Expected Result(s)	1. From Step 5: Format Identifier has a value 0x46414300. 2. From Step 6: Version Number has a value of 0x30313000. 3. From Step 7: The length of the record is less than the container size limit in SP80073-1. 4. From Step 8: Number of Facial Images value is 1. 5. From Step 9: Number of Feature Points is a positive value.

9.4.2 Facial Image Data Conformance

Purpose	Verify that the Facial Image Instance is conformant to the PIV profile presented in Table 6 of SP80076.
Reference(s)	1. SP80076, Table 6 2. AS05.03.01 3. AS05.03.02

Precondition(s)	1. All required sample facial image is stored on the PIV card. 2. A valid PIV card is inserted into the contact reader. 3. A valid PC/SC connection exists between the test application and the contact reader. 4. The test application is currently connected to the card application which is accessible through card handle. 5. Security conditions to read the object are met.
Test Scenario	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the contents of Facial Image Instance Record 5. Extract contents of Facial Image Type. 6. Extract contents of Image Data Type. 7. Extract contents of Image Color Space. 8. Extract contents of Source Type.
Expected Result(s)	1. Step 5: Facial Image Type is 1. 2. Step 6: Image Data Type is 0 or 1. 3. Step 7: Image Color Space is 1. 4. Step 8: Source Type is 2 or 6.

10. Signed Data Elements Test Assertions

10.1 Card Holder Unique Identifier (CHUID)

10.1.1 Signature Block Contents

10.1.1.1 Verify presence of CMS SignedData asymmetric digital signature

Purpose	Confirms that the CHUID buffer contains an asymmetric digital signature implemented as a SignedData type in accordance with the Cryptographic Message Syntax as defined in RFC 3852.
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.01.01 3. AS06.01.02
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the obtained CHUID and extract the contents from the asymmetric digital signature field (i.e. Tag 0x3E) 5. Process the contents of the digital signature
Expected Result(s)	The CHUID buffer contains an asymmetric digital signature that is implemented as a SignedData type and is encoded as a CMS external signature according to RFC 3852.

10.1.1.2 Verify version in SignedData

Purpose	Confirms that the version of the SignedData content type is 3.
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.01.03
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the version field contents from the asymmetric signature of the CHUID
Expected Result(s)	The value of the version field of the SignedData is 3.

10.1.1.3 Verify digest Algorithm in SignedData

Purpose	Confirm that the digestAlgorithms field of the SignedData content type is in accordance with Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS06.01.04
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the digestAlgorithms field contents from the asymmetric digital signature of the CHUID 5. Extract the expiration date value from the CHUID (i.e. Tag 0x35) 6. Extract the certificates field contents from the asymmetric signature of the CHUID 7. From the certificate obtained, extract the subjectPublicKeyInfo->subjectPublicKey 8. Compute the size of the signer's public key 9. Match the digest algorithm with Table 3-3 of SP80078 based on the card expiration date, and the public key algorithm and size used to sign the CHUID
Expected Result(s)	The digestAlgorithms field value of the SignedData is in accordance with Table 3-3 of SP80078.

10.1.1.4 Verify contents of encapContentInfo

Purpose	Confirms that the eContentType of the encapContentInfo is id-PIV-CHUIDSecurityObject and the eContent field of the encapContentInfo is omitted.
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.01.05 3. AS06.01.06
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract and parse the encapContentInfo field contents from the asymmetric digital signature of the CHUID
Expected Result(s)	The eContent field has been omitted and the eContentType asserts id-piv-CHUIDSecurityObject in encapContentInfo.

10.1.1.5 Verify crls field omission

Purpose	Confirm that the crls field from the SignedData content type is omitted.
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.01.08
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the crls field contents from the asymmetric digital signature of the CHUID

Expected Result(s)	The crls field is omitted from the SignedData.

10.1.1.6 Verify contents of signerInfos

Purpose	Confirms that the signerInfos in the SignedData is populated with a single SignerInfo.
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.01.09
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the SignerInfos field contents from the asymmetric digital signature of the CHUID
Expected Result(s)	The signerInfos field in the SignedData contains a single SignerInfo.

10.1.1.7 Verify Signer Identifier in SignerInfo

Purpose	Confirms that the sid in the SignerInfo uses the issuerAndSerialNumber choice.
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.01.10
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the signerInfo->sid field contents from the asymmetric signature of the CHUID 5. Extract the certificates field contents from the asymmetric signature of the CHUID 6. Extract the issuer and serialNumber fields from the certificate obtained in the previous step
Expected Result(s)	The sid in the SignerInfo uses the issuerAndSerialNumber choice and it corresponds to the issuer and serialNumber fields found in the X.509 certificate of the signer.

10.1.1.8 Verify Digest Algorithm in SignerInfo

Purpose	Confirm that the digestAlgorithm field of the SignerInfo is in accordance with Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS06.01.11
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the SignerInfo->digestAlgorithm field contents from the asymmetric digital signature of the CHUID 5. Extract the expiration date value from the CHUID (i.e. Tag 0x35) 6. Extract the certificates field contents from the asymmetric signature of the CHUID 7. From the certificate obtained, extract the subjectPublicKeyInfo->subjectPublicKey 8. Compute the size of the signer's public key 9. Match the digest algorithm with the Table 3-3 of SP80078 based on the card expiration date, and the public key algorithm and size used to sign the CHUID 10. Extract the digestAlgorithms field contents from the asymmetric digital signature of the CHUID i.e. SignedData
Expected Result(s)	The digestAlgorithm field value of the SignerInfo is in accordance with Table 3-3 of SP80078 and it matches the value present in the digestAlgorithms field of the SignedData.

10.1.1.9 Verify message digest signed attribute in SignerInfo

Purpose	Confirms that the signedAttrs of the SignerInfo includes a message digest attribute containing the hash computed over the concatenated contents of the CHUID, excluding the asymmetric signature field.
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.01.12
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the SignerInfo-> signedAttrs field contents from the asymmetric signature field of the CHUID to locate the message digest attribute (OID=1.2.840.113549.1.9.4) and its corresponding attribute value

	5. Extract the SignerInfo->digestAlgorithm field contents from the asymmetric digital signature of the CHUID 6. Using the digest Algorithm obtained in the previous step, calculate the hash of the concatenated contents of the CHUID, excluding the asymmetric digital signature field
Expected Result(s)	The value of the hash obtained from the message digest attribute of the signedAttrs of the SignerInfo is identical to that obtained after hashing the concatenated contents of the CHUID, excluding the asymmetric digital signature field.

10.1.1.10 Verify PIV signer distinguished name

Purpose	Confirms that the signedAttrs of the SignerInfo includes the pivSigner-DN attribute containing the subject name that appears in the X.509 certificate for the entity that signed the CHUID.
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.01.13
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the SignerInfo-> signedAttrs field contents from the asymmetric signature field of the CHUID to locate the pivSigner-DN attribute (OID=2.16.840.1.101.3.6.5) 5. Extract the certificates field contents from the asymmetric signature of the CHUID. 6. Extract the subject DN from the certificate obtained in the previous step
Expected Result(s)	The value of the subject DN obtained from the certificate in the certificates field in the SignedData is identical to that obtained from the pivSigner-DN attribute of the signedAttrs of the SignerInfo.

10.1.1.11 Verify signature algorithm in SignerInfo

Purpose	Confirms that the signatureAlgorithm field specified in the SignerInfo field is in accordance with Table 3-4 of SP80078 and based on the PIV card expiration date in accordance with Table 3-3 of SP80078.

Reference(s)	1. SP80078 , Section 3.2.1 2. AS06.01.14
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the expiration date value from the CHUID (i.e. Tag 0x35) 5. Extract the SignerInfo->signatureAlgorithm field contents.
Expected Result(s)	1. From Step 5: The signatureAlgorithm value is in accordance with Table 3-4 of SP80078 2. From Step 4: The expiration date value is earlier than or equal to the sunset date specified in Table 3-3 of SP80078 for the algorithm from Step 5.

10.1.1.12 Verify digital signature

Purpose	Confirms that the signature in the SignerInfo corresponds to the signed CHUID
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.01.07 3. AS06.01.15
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the certificates field contents from the asymmetric signature of the CHUID. 5. Extract the asymmetric signature contents from the CHUID 6. Using the certificate extracted, verify the SignedData located in the asymmetric signature field of the CHUID.
Expected Result(s)	The certificates field in the SignedData contains a single certificate that can be used to verify the digital signature in the SignerInfo.

10.1.2 Embedded Certificate

10.1.2.1 Verify extended key usage extension

Purpose	Confirms that the digital signature certificate used to sign the CHUID asserts id-PIV-content-signing in the extendedKeyUsage extension.
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.01.16
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the certificates field contents from the asymmetric signature of the CHUID 5. Extract all KeyPurposeId fields from the extended key usage extension from the certificate obtained
Expected Result(s)	A KeyPurposeId asserting id-PIV-content-signing exists in the extended key usage extension.

10.1.2.2 Verify signer public key size

Purpose	Confirms that the size of the public key for digital signature used to sign the CHUID is based on the expiration of the PIV card and is in accordance with Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.1 2. AS06.01.17
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the expiration date from the CHUID (i.e. Tag 0x35) 5. Extract the certificates field contents from the asymmetric signature of the CHUID. 6. Extract subjectPublicKeyInfo->algorithm->algorithm field value 7. From the obtained certificate, extract the subjectPublicKeyInfo->subjectPublicKey Note: - Since the ECDSA keys do not have any size restrictions based on dates, this test case does not apply to these types of keys
Expected Result(s)	Verify that the size of the key from Step 7, algorithm from Step 6 and expiration date from Step 4 are consistent with entries in Table 3-3 of SP80078.

10.2 Fingerprint Biometric

10.2.1 Signature Block Contents

10.2.1.1 Verify presence of CMS SignedData asymmetric digital signature

Purpose	Confirms that the CBEFF_SIGNATURE_BLOCK is implemented as a SignedData type and is encoded as a Cryptographic Message Syntax external digital signature as defined in RFC 3852
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.02.01 3. AS06.02.02
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint object is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the obtained biometric and extract the contents from the asymmetric digital signature field (i.e. from the CBEFF_SIGNATURE_BLOCK) 5. Process the contents of the digital signature
Expected Result(s)	The CBEFF_SIGNATURE_BLOCK is present in the biometric CBEFF structure containing an asymmetric digital signature that is implemented as a SignedData type according to RFC 3852.

10.2.1.2 Verify version in SignedData

Purpose	Confirms that the version of the SignedData content type is v1 or v3.
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.02.03
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint object is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the version field contents from the CBEFF_SIGNATURE_BLOCK 5. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK (if present)
Expected Result(s)	The value of the version field of the SignedData is v1 if the certificates field is omitted and v3 if present.

10.2.1.3 Verify digest Algorithm in SignedData

Purpose	Confirm that the digestAlgorithms field of the SignedData content type is in accordance with Table 3-3 of SP80078.
Reference(s)	1. SP80078 , Section 3.2.1 2. AS06.02.04
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint object is present on the PIV card. 5. A valid CHUID object is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the digestAlgorithms field contents from the CBEFF_SIGNATURE_BLOCK 5. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK (if present) 6. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 7. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 8. Extract the expiration date value from the CHUID (i.e. Tag 0x35) 9. If a certificate extracted from the certificates field of the CBEFF_SIGNATURE_BLOCK does not exist, then extract the certificates field contents from the asymmetric signature of the CHUID 10. From the certificate obtained (either from the CBEFF_SIGNATURE_BLOCK or the CHUID signature field), extract the subjectPublicKeyInfo->subjectPublicKey 11. Compute the size of the signer's public key 12. Match the digest algorithm obtained from step 4 with Table 3-3 of SP80078 based on the card expiration date, and the public key algorithm and size used to sign the fingerprint biometric
Expected Result(s)	The digestAlgorithms field value of the SignedData is in accordance with Table 3-3 of SP80078.

10.2.1.4 Verify contents of encapContentInfo

Purpose	Confirms that the eContentType of the encapContentInfo is id-PIV-biometricObject and the eContent field of the encapContentInfo is omitted.
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.02.05 3. AS06.02.06
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint object is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract and parse the encapContentInfo field contents from the CBEFF_SIGNATURE_BLOCK
Expected Result(s)	The eContent field has been omitted and the eContentType asserts id-piv-biometricObject in encapContentInfo.

10.2.1.5 Verify crls field omission

Purpose	Confirm that the crls field from the SignedData content type is omitted.
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.02.08
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint object is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the crls field contents from the CBEFF_SIGNATURE_BLOCK
Expected Result(s)	The crls field is omitted from the SignedData.

10.2.1.6 Verify contents of signerInfos

Purpose	Confirms that the signerInfos in the SignedData is populated with a single SignerInfo.
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.02.09

Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint object is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the SignerInfos field contents from the CBEFF_SIGNATURE_BLOCK
Expected Result(s)	The signerInfos field in the SignedData contains a single SignerInfo.

10.2.1.7 Verify Signer Identifier in SignerInfo

Purpose	Confirms that the sid in the SignerInfo uses the issuerAndSerialNumber choice.
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.02.10
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint object is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the signerInfo->sid field contents from the CBEFF_SIGNATURE_BLOCK 5. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK (if present) 6. If a certificate extracted from the certificates field of the CBEFF_SIGNATURE_BLOCK does not exist, then extract the certificates field contents from the asymmetric signature of the CHUID 7. Extract the issuer and serialNumber fields from the certificate obtained in the previous step

Expected Result(s)	The sid in the SignerInfo uses the issuerAndSerialNumber choice which corresponds to the issuer and serialNumber fields found in the X.509 certificate of the signer.

10.2.1.8 Verify Digest Algorithm in SignerInfo

Purpose	Confirm that the digestAlgorithm field of the SignerInfo is in accordance with Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS06.02.11
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint object is present on the PIV card. 5. A valid CHUID object is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the SignerInfo->digestAlgorithm field from the CBEFF_SIGNATURE_BLOCK 5. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK (if present) 6. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 7. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 8. Extract the expiration date value from the CHUID (i.e. Tag 0x35) 9. If a certificate extracted from the certificates field of the CBEFF_SIGNATURE_BLOCK does not exist, then extract the certificates field contents from the asymmetric signature of the CHUID 10. From the certificate obtained (either from the CBEFF_SIGNATURE_BLOCK or the CHUID signature field), extract the subjectPublicKeyInfo->subjectPublicKey 11. Compute the size of the signer's public key 12. Match the digest algorithm obtained from step 4 with Table 3-3 of SP80078 based on the card expiration date, and the public key algorithm and size used to sign the fingerprint biometric 11. Extract the digestAlgorithms field contents from the CBEFF_SIGNATURE_BLOCK
Expected Result(s)	The digestAlgorithm field value of the SignerInfo is in accordance with Table 3-3 of SP80078 and it matches the value present in the

Result(s)	digestAlgorithms field of the SignedData.

10.2.1.9 Verify message digest signed attribute in SignerInfo

Purpose	Confirms that the signedAttrs of the SignerInfo includes a message digest attribute containing the hash computed over the concatenated CBEFF_HEADER and the STD_BIOMETRIC_RECORD
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.02.12
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint object is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the SignerInfo-> signedAttrs field contents from the CBEFF_SIGNATURE_BLOCK to locate the message digest attribute (OID=1.2.840.113549.1.9.4) and its corresponding attribute value 5. Extract the SignerInfo->digestAlgorithm field contents from the CBEFF_SIGNATURE_BLOCK 6. Using the digest Algorithm obtained in the previous step, compute the hash over the concatenated CBEFF_HEADER and the STD_BIOMETRIC_RECORD
Expected Result(s)	The value of the hash obtained from the message digest attribute of the signedAttrs of the SignerInfo is identical to that obtained after hashing the concatenated contents of the Fingerprint Object buffer, excluding the CBEFF_SIGNATURE_BLOCK.

10.2.1.10 Verify PIV signer distinguished name

Purpose	Confirms that the signedAttrs of the SignerInfo includes the pivSigner-DN attribute containing the subject name that appears in the X.509 certificate for the entity that signed the biometrics.
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.02.13

Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint object is present on the PIV card. 5. A valid CHUID object is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the SignerInfo-> signedAttrs field contents from the CBEFF_SIGNATURE_BLOCK to locate the pivSigner-DN attribute (OID=2.16.840.1.101.3.6.5) 5. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK 6. If a certificate extracted from the certificates field of the CBEFF_SIGNATURE_BLOCK does not exist, then extract the certificates field contents from the asymmetric signature of the CHUID 7. Extract the subject DN from the certificate obtained in the previous step
Expected Result(s)	The value of the subject DN obtained from the certificate in the certificates field in the SignedData is identical to that obtained from the pivSigner-DN attribute of the signedAttrs of the SignerInfo.

10.2.1.11 Verify FASC-N

Purpose	Confirms that the signedAttrs of the SignerInfo includes the pivFASC-N attribute whose value matches the value of the FASC-N in the CHUID of the PIV card.
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.02.14
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint object is present on the PIV card. 5. A valid CHUID object is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the SignerInfo-> signedAttrs field contents from the CBEFF_SIGNATURE_BLOCK to locate the pivFASC-N attribute (OID=2.16.840.1.101.3.6.6) and its corresponding attribute value 5. Set OID := <<CHUID (2.16.840.1.103.3.7.2.48.0)>> 6. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 7. Parse the CHUID and extract the FASC-N
Expected Result(s)	A pivFASC-N attribute exists in the signedAttrs of the SignerInfo and its value matches the FASC-N present in the CHUID of the PIV card.

10.2.1.12 Verify signature algorithm in SignerInfo

Purpose	Confirms that the signatureAlgorithm field specified in the SignerInfo field is in accordance with Table 3-4 of SP80078 and based on the PIV card expiration date in accordance with Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS06.02.15
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint object is present on the PIV card. 5. A valid CHUID object is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the expiration date value from the CHUID (i.e. Tag 0x35) 5. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 6. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 7. Extract the SignerInfo->signatureAlgorithm field.
Expected Result(s)	1. From Step 7: The signatureAlgorithm value is in accordance with Table 3-4 of SP80078 2. From Step 4: The expiration date value is earlier than or equal to the sunset date specified in Table 3-3 of SP80078 for the algorithm from Step 7.

10.2.1.13 Verify digital signature

Purpose	Confirms that the signature in the SignerInfo corresponds to the signed biometric
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.02.07 3. AS06.02.16
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint object is present on the PIV card. 5. A valid CHUID object is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK. If this field is omitted then extract the certificate from the CHUID signature 4. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 5. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 6. Extract the digital signature string from CBEFF_SIGNATURE_BLOCK. 7. Using the certificate extracted either from the CBEFF_SIGNATURE_BLOCK or the CHUID asymmetric signature, verify the signature on the biometric
Expected Result(s)	The certificates field in the SignedData contains a single certificate that can be used to verify the digital signature in the SignerInfo. If the certificates field is omitted, then the certificates field of the SignedData for the CHUID contains the certificate that can be used to verify the digital signature.

10.2.2 Embedded Certificate

10.2.2.1 Verify extended key usage extension

Purpose	Confirms that the digital signature certificate used to sign the fingerprint biometric asserts id-PIV-content-signing in the extendedKeyUsage extension.
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.02.17
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint I is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK. If the certificate field is omitted, then stop execution of this test assertion. 5. Extract all KeyPurposeId fields from the extended key usage extension from the certificate obtained

Expected Result(s)	A KeyPurposeId asserting id-PIV-content-signing exists in the extended key usage extension.

10.2.2.2 Verify signer public key size

Purpose	Confirms that the size of the public key for digital signature used to sign the biometrics is based on the expiration of the PIV card and is in accordance with Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.1 2. AS06.02.18
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder fingerprint I is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Fingerprints (2.16.840.1.101.3.7.2.96.16)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK. If the certificate field is omitted, then stop execution of this test assertion 5. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 6. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 7. Extract the expiration date from the CHUID (i.e. Tag 0x35) 8. Extract subjectPublicKeyInfo->algorithm->algorithm field value from the certificate obtained 9. Extract the subjectPublicKeyInfo->subjectPublicKey from the certificate obtained Note: - Since the ECDSA keys do not have any size restrictions based on dates, this test case does not apply to these types of keys
Expected Result(s)	Verify that the size of the key from Step 9, algorithm from Step 8 and expiration date from Step 7 are consistent with entries in Table 3-3 of SP80078.

10.3 Facial Image Biometric

10.3.1 Signature Block Contents

10.3.1.1 Verify presence of CMS SignedData asymmetric digital signature

Purpose	Confirms that the CBEFF_SIGNATURE_BLOCK is implemented as a SignedData type and is encoded as a Cryptographic Message Syntax external digital signature as defined in RFC 3852
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.03.01 3. AS06.03.02
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the obtained biometric and extract the contents from the asymmetric digital signature field (i.e. from the CBEFF_SIGNATURE_BLOCK) 5. Process the contents of the digital signature
Expected Result(s)	The CBEFF_SIGNATURE_BLOCK is present in the biometric CBEFF structure containing an asymmetric digital signature that is implemented as a SignedData type according to RFC 3852.

10.3.1.2 Verify version in SignedData

Purpose	Confirms that the version of the SignedData content type is v1 or v3.
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.03.03
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the version field contents from the CBEFF_SIGNATURE_BLOCK 5. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK (if present)
Expected Result(s)	The value of the version field of the SignedData is v1 if the certificates field is omitted and v3 if present.

10.3.1.3 Verify digest Algorithm in SignedData

Purpose	Confirm that the digestAlgorithms field of the SignedData content type is in accordance with Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS06.03.04
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card. 5. A valid CHUID is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the digestAlgorithms field contents from the CBEFF_SIGNATURE_BLOCK 5. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK (if present) 6. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 7. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 8. Extract the expiration date value from the CHUID (i.e. Tag 0x35) 9. If a certificate extracted from the certificates field of the CBEFF_SIGNATURE_BLOCK does not exist, then extract the certificates field contents from the asymmetric signature of the CHUID 10. From the certificate obtained (either from the CBEFF_SIGNATURE_BLOCK or the CHUID signature field), extract the subjectPublicKeyInfo->subjectPublicKey 11. Compute the size of the signer's public key 12. Match the digest algorithm obtained from step 4 with Table 3-3 of SP80078 based on the card expiration date, and the public key algorithm and size used to sign the facial image biometric
Expected Result(s)	The digestAlgorithms field value of the SignedData is in accordance with Table 3-3 of SP80078.

10.3.1.4 Verify contents of encapContentInfo

Purpose	Confirms that the eContentType of the encapContentInfo is id-PIV-biometricObject and the eContent field of the encapContentInfo is omitted.
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.03.05 3. AS06.03.06
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract and parse the encapContentInfo field contents from the CBEFF_SIGNATURE_BLOCK
Expected Result(s)	The eContent field has been omitted and the eContentType asserts id-piv-biometricObject in encapContentInfo.

10.3.1.5 Verify crls field omission

Purpose	Confirm that the crls field from the SignedData content type is omitted.
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.03.08
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the crls field contents from the CBEFF_SIGNATURE_BLOCK
Expected Result(s)	The crls field is omitted from the SignedData.

10.3.1.6 Verify contents of signerInfos

Purpose	Confirms that the signerInfos in the SignedData is populated with a single SignerInfo.
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.03.09

Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the SignerInfos field contents from the CBEFF_SIGNATURE_BLOCK
Expected Result(s)	The signerInfos field in the SignedData contains a single SignerInfo.

10.3.1.7 Verify Signer Identifier in SignerInfo

Purpose	Confirms that the sid in the SignerInfo uses the issuerAndSerialNumber choice.
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.03.10
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the signerInfo->sid field contents from the CBEFF_SIGNATURE_BLOCK 5. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK (if present) 6. If a certificate extracted from the certificates field of the CBEFF_SIGNATURE_BLOCK does not exist, then extract the certificates field contents from the asymmetric signature of the CHUID 7. Extract the issuer and serialNumber fields from the certificate obtained in the previous step

Expected Result(s)	The sid in the SignerInfo uses the issuerAndSerialNumber choice which corresponds to the issuer and serialNumber fields found in the X.509 certificate of the signer.

10.3.1.8 Verify Digest Algorithm in SignerInfo

Purpose	Confirm that the digestAlgorithm field of the SignerInfo is in accordance with Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS06.03.11
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card. 5. A valid CHUID is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the SignerInfo->digestAlgorithm field from the CBEFF_SIGNATURE_BLOCK 5. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK (if present) 6. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 7. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 8. Extract the expiration date value from the CHUID (i.e. Tag 0x35) 9. If a certificate extracted from the certificates field of the CBEFF_SIGNATURE_BLOCK does not exist, then extract the certificates field contents from the asymmetric signature of the CHUID 10. From the certificate obtained (either from the CBEFF_SIGNATURE_BLOCK or the CHUID signature field), extract the subjectPublicKeyInfo->subjectPublicKey 11. Compute the size of the signer's public key 12. Match the digest algorithm obtained from step 4 with Table 3-3 of SP80078 based on the card expiration date, and the public key algorithm and size used to sign the facial image biometric 13. Extract the digestAlgorithms field contents from the CBEFF_SIGNATURE_BLOCK
Expected Result(s)	The digestAlgorithm field value of the SignerInfo is in accordance with Table 3-3 of SP80078 and it matches the value present in the digestAlgorithms field of the SignedData.

10.3.1.9 Verify message digest signed attribute in SignerInfo

Purpose	Confirms that the signedAttrs of the SignerInfo includes a message digest attribute containing the hash computed over the concatenated CBEFF_HEADER and the STD_BIOMETRIC_RECORD
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.03.12

Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the SignerInfo-> signedAttrs field contents from the CBEFF_SIGNATURE_BLOCK to locate the message digest attribute (OID=1.2.840.113549.1.9.4) and its corresponding attribute value 5. Extract the SignerInfo->digestAlgorithm field contents from the CBEFF_SIGNATURE_BLOCK 6. Using the digest Algorithm obtained in the previous step, compute the hash over the concatenated CBEFF_HEADER and the STD_BIOMETRIC_RECORD
Expected Result(s)	The value of the hash obtained from the message digest attribute of the signedAttrs of the SignerInfo is identical to that obtained after hashing the concatenated contents of the CHUID, excluding the asymmetric digital signature field.

10.3.1.10 Verify PIV signer distinguished name

Purpose	Confirms that the signedAttrs of the SignerInfo includes the pivSigner-DN attribute containing the subject name that appears in the X.509 certificate for the entity that signed the biometrics.
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.03.13
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card. 5. A valid CHUID is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the SignerInfo-> signedAttrs field contents from the CBEFF_SIGNATURE_BLOCK to locate the pivSigner-DN attribute (OID=2.16.840.1.101.3.6.5) 5. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK 6. If a certificate extracted from the certificates field of the CBEFF_SIGNATURE_BLOCK does not exist, then extract the certificates field contents from the asymmetric signature of the CHUID 7. Extract the subject DN from the certificate obtained in the previous step
Expected Result(s)	The value of the subject DN obtained from the certificate in the certificates field in the SignedData is identical to that obtained from the pivSigner-DN attribute of the signedAttrs of the SignerInfo.

10.3.1.11 Verify FASC-N

Purpose	Confirms that the signedAttrs of the SignerInfo includes the pivFASC-N attribute whose value matches the value of the FASC-N in the CHUID of the PIV card.
Reference(s)	1. FIPS201, Section 4.4.2 2. AS06.03.14
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card. 5. A valid CHUID is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the SignerInfo-> signedAttrs field contents from the CBEFF_SIGNATURE_BLOCK to locate the pivFASC-N attribute (OID=2.16.840.1.101.3.6.6) and its corresponding attribute value 5. Set OID := <<CHUID (2.16.840.1.103.3.7.2.48.0)>> 6. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 7. Parse the CHUID and extract the FASC-N
Expected Result(s)	A pivFASC-N attribute exists in the signedAttrs of the SignerInfo and its value matches the FASC-N present in the CHUID of the PIV card.

10.3.1.12 Verify signature algorithm in SignerInfo

Purpose	Confirms that the signatureAlgorithm field specified in the SignerInfo field is in accordance with Table 3-4 of SP80078 and based on the PIV card expiration date in accordance with Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS06.03.15
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card. 5. A valid CHUID is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the expiration date value from the CHUID (i.e. Tag 0x35) 5. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 6. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 7. Extract the SignerInfo->signatureAlgorithm field contents.
Expected Result(s)	1. From Step 7: The signatureAlgorithm value is in accordance with Table 3-4 of SP80078 2. From Step 4: The expiration date value is earlier than or equal to the sunset date specified in Table 3-3 of SP80078 for the algorithm from Step 7.

10.3.1.13 Verify digital signature

Purpose	Confirms that the signature in the SignerInfo corresponds to the signed biometric
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.03.07 3. AS06.03.16
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card. 5. A valid CHUID is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK. If this field is omitted then extract the certificate from the CHUID signature 4. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 5. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 6. Extract the certificates field contents from the asymmetric signature of the CHUID. 7. Using the certificate extracted either from the CBEFF_SIGNATURE_BLOCK or the CHUID asymmetric signature, verify the signature on the biometric
Expected Result(s)	The certificates field in the SignedData contains a single certificate that can be used to verify the digital signature in the SignerInfo. If the certificates field is omitted, then the certificates field of the SignedData for the CHUID contains the certificate that can be used to verify the digital signature.

10.3.2 Embedded Certificate

10.3.2.1 Verify extended key usage extension

Purpose	Confirms that the digital signature certificate used to sign the facial image biometric asserts id-PIV-content-signing in the extendedKeyUsage extension.
Reference(s)	1. FIPS201, Section 4.2.2 2. AS06.03.17
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK. If the certificate field is omitted, then stop execution of this test assertion. 5. Extract all KeyPurposeId fields from the extended key usage extension from the certificate obtained

Expected Result(s)	A KeyPurposeId asserting id-PIV-content-signing exists in the extended key usage extension.

10.3.2.2 Verify signer public key size

Purpose	Confirms that the size of the public key for digital signature used to sign the biometrics is based on the expiration of the PIV card and is in accordance with Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.1 2. AS06.03.18
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid card holder facial image is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Holder Facial Image (2.16.840.1.101.3.7.2.96.48)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the certificates field contents from the CBEFF_SIGNATURE_BLOCK. If the certificate field is omitted, then stop execution of this test assertion 5. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 6. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 7. Extract the expiration date from the CHUID (i.e. Tag 0x35) 8. Extract subjectPublicKeyInfo->algorithm->algorithm field value from the certificate obtained 9. Extract the subjectPublicKeyInfo->subjectPublicKey from the certificate obtained Note: - Since the ECDSA keys do not have any size restrictions based on dates, this test case does not apply to these types of keys
Expected Result(s)	Verify that the size of the key from Step 9, algorithm from Step 8 and expiration date from Step 7 are consistent with entries in Table 3-3 of SP80078.

10.4 Security Object

10.4.1 Data Integrity

10.4.1.1 Verify integrity of data element hashes

Purpose	Confirms the integrity of the hashes of the data elements of the PIV card present in the security object.
Reference(s)	1. SP80073, Section 1.8.5 2. AS06.04.01
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid security object is present on the card. 5. Valid objects whose hashes are referenced in the security object are present on the card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Security Object (2.16.840.1.101.3.7.2.144.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Identify the various data elements that are part of the security object by parsing the Mapping of Data Group (DG) to ContainerID (i.e. TAG 0xBA) 5. Extract the ldsSecurityObject from the eContent field of the Security Object Asymmetric Signature (i.e. TAG 0xBB) 6. Call pivGetData w/ for all those data elements that are present in the mapping obtained from step 4 7. Compute the hash for each data element and verify that it matches the hash value present in the ldsSecurityObject
Expected Result(s)	The actual hash of the data elements on the PIV card are identical to their corresponding hash values present in the security object.

10.4.2 Signature Block Contents

10.4.2.1 Verify presence of CMS SignedData asymmetric digital signature

Purpose	Confirms that the security object buffer contains an asymmetric digital signature, implemented as a SignedData type in accordance with RFC 3852
Reference(s)	1. AS06.04.02 2. AS06.04.03

Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid security object is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Security Object (2.16.840.1.101.3.7.2.144.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Parse the obtained security object and extract the contents from the asymmetric digital signature field (i.e. TAG 0xBB) 5. Process the contents of the digital signature
Expected Result(s)	The security object is present in the security object buffer and contains an asymmetric digital signature that is implemented as a SignedData type according to RFC 3852.

10.4.2.2 Verify version in SignedData

Purpose	Confirms that the version of the SignedData content type is 1.
Reference(s)	1. AS06.04.04
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid security object is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Security Object (2.16.840.1.101.3.7.2.144.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the version field contents from the asymmetric signature of the Security Object (i.e. TAG 0xBB)
Expected Result(s)	The value of the version field of the SignedData is 1.

10.4.2.3 Verify digest Algorithm in SignedData

Purpose	Confirm that the digestAlgorithms field of the SignedData content type is in accordance with Table 3-7 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS06.04.05
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid security object is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Security Object (2.16.840.1.101.3.7.2.144.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the digestAlgorithms field contents from the Security Object 5. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 6. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 7. Extract the expiration date value from the CHUID (i.e. Tag 0x35) 8. Extract the certificates field contents from the asymmetric signature of the CHUID 9. From the certificate obtained (from the CHUID signature field), extract the subjectPublicKeyInfo->subjectPublicKey 10. Compute the size of the signer's public key 11. Match the digest algorithm obtained from step 4 with Table 3-7 of SP80078 based on the card expiration date, and the public key algorithm and size used to sign the security object
Expected Result(s)	The digestAlgorithms field value of the SignedData is in accordance with Table 3-7 of SP80078.

10.4.2.4 Verify contents of encapContentInfo

Purpose	Confirms that the eContentType of the encapContentInfo is id-icao-ldsSecurityObject and the eContent field of the encapContentInfo contains the contents of the ldsSecurity object.

Reference(s)	1. AS06.04.06 2. AS06.04.07
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid security object is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Security Object (2.16.840.1.101.3.7.2.144.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract and parse the encapContentInfo field contents from the security object
Expected Result(s)	The eContent field contains a correctly formatted ldsSecurityobject and the eContentType asserts id-icao-ldsSecurityObject in encapContentInfo.

10.4.2.5 Verify certificates field omission

Purpose	Confirm that the certificates field from the SignedData content type is omitted.
Reference(s)	1. AS06.04.08
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid security object is present on the PIV card..
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Security Object (2.16.840.1.101.3.7.2.144.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the certificates field contents from the security object
Expected Result(s)	The certificates field is omitted from the SignedData.

10.4.2.6 Verify Digest Algorithm in SignerInfo

Purpose	Confirm that the digestAlgorithm field of the SignerInfo is in accordance with Table 3-7 of SP80078.
Reference(s)	1. AS06.04.09
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid security object is present on the PIV card. 5. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Security Object (2.16.840.1.101.3.7.2.144.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the SignerInfo->digestAlgorithm field from the security object 5. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 6. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 7. Extract the expiration date value from the CHUID (i.e. Tag 0x35) 8. Extract the certificates field contents from the asymmetric signature of the CHUID 9. From the certificate obtained (from the CHUID signature field), extract the subjectPublicKeyInfo->subjectPublicKey 10. Compute the size of the signer's public key 11. Match the digest algorithm obtained from step 4 with Table 3-7 of SP80078 based on the card expiration date, and the public key algorithm and size used to sign the security object 12. Extract the digestAlgorithms field contents from the security object's SignedData
Expected Result(s)	The digestAlgorithm field value of the SignerInfo is in accordance with Table 3-7 of SP80078 and it matches the value present in the digestAlgorithms field of the SignedData.

10.4.2.7 Verify signature algorithm in SignerInfo

Purpose	Confirms that the signatureAlgorithm field specified in the SignerInfo field is in accordance with Table 3-4 of SP80078 and based on the PIV card expiration date in accordance with Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS06.04.10
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid security object is present on the PIV card. 5. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the expiration date value from the CHUID (i.e. Tag 0x35) 5. Set OID := <<Security Object (2.16.840.1.101.3.7.2.144.0)>> 6. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 7. From the signature block (TAG 0xBB), match the SignerInfo->signatureAlgorithm field contents with Table 3-4 of SP80078 based on the card expiration date
Expected Result(s)	The signatureAlgorithm value in the SignerInfo field is in accordance with Table 3-4 of SP80078 and is based on the PIV card expiration date in accordance with Table 3-3 of SP80078.

10.4.2.8 Verify digital signature

Purpose	Confirms that the signature in the SignerInfo corresponds to the signed security object and that it is it signed with the certificate that is used to sign the CHUID.
Reference(s)	1. SP80073, Section 1.8.5 2. AS06.04.11 3. AS06.04.12

Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A valid security object is present on the PIV card. 5. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Security Object (2.16.840.1.101.3.7.2.144.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the contents of the Security Object asymmetric signature (TAG 0xBB) 5. Set OID := <<CHUID (2.16.840.1.101.3.7.2.48.0)>> 6. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 7. Extract the certificates field contents from the asymmetric signature of the CHUID. 8. Using the certificate extracted from the CHUID asymmetric signature block, verify the signature of the security object
Expected Result(s)	The certificates field of the SignedData for the CHUID contains the certificate that can be used to verify the digital signature on the security object.

11. PKI Certificate Profile Test Assertions

11.1 PIV Authentication Certificate

11.1.1 SP 800-78 Algorithms Conformance

11.1.1.1 Verify signature algorithm

Purpose	Confirms that the proper signature algorithm has been used to sign the certificate as specified in Table 3-4 of SP80078 based on the expiration date as specified in Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS07.01.01 3. AS07.01.02
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A PIV authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<PIV Authentication Certificate (2.16.840.1.101.3.7.2.1.1)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract signature->algorithm field value from the certificate 5. Extract validity->notAfter->utcTime field value from the certificate
Expected Result(s)	1. From Step 4: The signatureAlgorithm value is in accordance with Table 3-4 of SP80078 2. From Step 5: The certificate expiration date value is earlier than or equal to the sunset date specified in Table 3-3 of SP80078 for the algorithm from Step 4. 3. From Step 4: If the algorithm value is id-RSASSA-PSS, verify that the signature->parameters field is populated with SHA-256 (OID = 2.16.840.1.101.3.4.2.1). For the other RSA algorithms, the parameters field is populated with NULL. For ECDSA, the parameters field is absent.

11.1.1.2 Verify subject public key algorithm

Purpose	Confirms that the public key algorithm used for generating the keys is as specified in Table 3-5 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS07.01.03 3. AS07.01.04
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A PIV authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<PIV Authentication Certificate (2.16.840.1.101.3.7.2.1.1)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract subjectPublicKeyInfo->algorithm->algorithm field value 5. Match the algorithm value to the Table 3-5 of SP80078 6. If the algorithm is Elliptic Curve, ensure that one of the approved curves is used and the OID is populated in the subjectPublicKeyInfo->algorithm->parameters->namedCurve field from Table 3-6 of SP80078. The parameters field may contain NULL to indicate that parameters are inherited. Note: - If the RSA algorithm is used, the subjectPublicKeyInfo->algorithm->parameters field shall be NULL
Expected Result(s)	The PIV authentication key is generated using the allowed asymmetric key algorithm.

11.1.1.3 Verify public key size

Purpose	Verifies that the key size requirements are adhered to based on the expiration date of certificate.
Reference(s)	1. SP80078, Section 3.1 2. AS07.01.10

Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A PIV authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<PIV Authentication Certificate (2.16.840.1.101.3.7.2.1.1)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract validity->notAfter->utcTime field value from the certificate 5. Extract subjectPublicKeyInfo->algorithm->algorithm field value 6. Extract the subjectPublicKeyInfo->subjectPublicKey from the certificate 7. Match the key size to Table 3-3 of SP80078 based on the algorithm obtained from the step 3 and the date obtained from step 2 Note: - Since the ECDSA keys do not have any size restrictions based on dates, this test case does not apply to these types of keys
Expected Result(s)	The key sizes used adhere to the time period for use requirement.

11.1.2 Data Integrity Checks

11.1.2.1 Verify key usage extension

Purpose	Confirms that the PIV authentication certificate asserts the appropriate purpose of the key.
Reference(s)	1. FIPS201, Section 5.4.2.1 2. AS07.01.05
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A PIV authentication key and corresponding certificate are present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<PIV Authentication Certificate (2.16.840.1.101.3.7.2.1.1)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value of the keyUsage extension field from the certificate
Expected Result(s)	The digitalSignature bit has been set. No other bits have been set.

11.1.2.2 Verify id-fpki-common-authentication OID

Purpose	Confirms that the PIV Authentication certificate asserts the id-fpki-common-authentication OID.
Reference(s)	1. AS07.01.06
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A PIV authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<PIV Authentication Certificate (2.16.840.1.101.3.7.2.1.1)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract certificatePolicies->policyIdentifier extension field values from the certificate
Expected Result(s)	A policyIdentifier field in the certificatePolicies extension asserts id-fpki-common-authentication.

11.1.2.3 Verify authority information access extension

Purpose	Confirms the authority information access extension is populated with the location to the OCSP Server that provides status information for this certificate.
Reference(s)	1. FIPS201, Section 5.4.2.1 2. AS07.01.07

Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A PIV authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<PIV Authentication Certificate (2.16.840.1.101.3.7.2.1.1)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the AuthorityInfoAccess->accessMethod and AuthorityInfoAccess->accessLocation extension fields from the certificate
Expected Result(s)	An accessMethod containing id-ad-ocsp is present. The accessLocation for this AccessMethod is of type uniformResourceIdentifier and that the scheme is “http” (not “https”).

11.1.2.4 Verify interim status extension

Purpose	Confirms that the piv-interim extension is present in the certificate.
Reference(s)	1. FIPS201, Appendix D 2. AS07.01.09
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A PIV authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<PIV Authentication Certificate (2.16.840.1.101.3.7.2.1.1)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the piv-interim extension in the certificate
Expected Result(s)	The piv-interim extension is present and contains the interim_indicator field which is of type BOOLEAN.

11.1.2.5 Verify asymmetric key pair

Purpose	Confirms that the public key that exists in the certificate corresponds to the private key on the card.
Reference(s)	1. FIPS201, Section 5 2. AS07.01.11
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A PIV authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. Take an arbitrary stream of data 2. Hash the data using a hash algorithm 3. Set cardHandle := <<valid card handle>> 4. Set authenticators := <<valid authenticator>> 5. Call pivLogIntoCardApplication • (IN) cardHandle • (IN) authenticators 6. Set keyReference := <<key reference for PIV Authentication Key i.e. 9A>> 7. Set algorithmIdentifier := <<identifier of the algorithm to be used for the cryptographic operation>> 8. Set algorithmInput := <<hashed data from Step 2>> 9. Call pivCrypt with the following parameters • (IN) keyReference • (IN) algorithmIdentifier • (IN) algorithmInput • (OUT) algorithmOutput 10. Set OID := <<PIV Authentication Certificate (2.16.840.1.101.3.7.2.1.1)>> 11. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 12. Verify the signature with subjectPublicKeyInfo->subjectPublicKey from the certificate
Expected Result(s)	The private key corresponds to the public key contained in the certificate as the signature verification succeeds.

11.1.2.6 Verify FASC-N

Purpose	Confirms that the subjectAltName extension contains the FASC-N of the card holder and that it matches to that present in the CHUID.

Reference(s)	1. FIPS201, Section 5.4.2.1 2. AS07.01.08 3. AS07.01.12
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A PIV authentication key and corresponding certificate are present on the PIV card. 5. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<PIV Authentication Certificate (2.16.840.1.101.3.7.2.1.1)>> 3. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the GeneralNames field from the subjectAltName extension in the certificate 5. Parse the different GeneralName fields 6. Set OID := <<CHUID (2.16.840.1.103.3.7.2.48.0)>> 7. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 8. Parse the CHUID and extract the FASC-N
Expected Result(s)	A GeneralName field exists that contains an otherName with a type-id asserting the pivFASC-N OID. The value field of this otherName contains the FASC-N for the cardholder which matches the FASC-N obtained from parsing the CHUID.

11.1.2.7 Verify expiration dates consistency

Purpose	Confirms that the expiration date of the PIV authentication certificate is not past the expiration date of the PIV card.
Reference(s)	1. FIPS201, Section 5.3.2.1 2. AS07.01.13

Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A PIV authentication key and corresponding certificate are present on the PIV card. 5. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<PIV Authentication certificate (2.16.840.1.101.3.7.2.1.1)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract validity->notAfter->utcTime field value from the certificate 5. Set OID := << CHUID (2.16.840.1.103.3.7.2.48.0)>> 6. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 7. Parse the CHUID and extract the expiration date
Expected Result(s)	The expiration date of the PIV authentication certificate is not beyond the expiration date of the CHUID i.e. the PIV card.

11.1.2.8 Verify RSA exponent

Purpose	For RSA keys, confirms that the exponent of the RSA asymmetric key for PIV authentication is greater than or equal to 65,537.
Reference(s)	1. SP80078, Section 3.1 2. AS07.01.14
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A PIV authentication key and corresponding certificate are present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<PIV Authentication Certificate (2.16.840.1.101.3.7.2.1.1)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the subjectPublicKeyInfo->subjectPublicKey from the certificate. 5. Parse the exponent from the extracted public key
Expected Result(s)	The exponent of the RSA asymmetric key for PIV authentication is greater than or equal to 65,537.

11.2 Digital Signature Certificate

11.2.1 SP 800-78 Algorithm Conformance

11.2.1.1 Verify signature algorithm

Purpose	Confirms that the proper signature algorithm has been used to sign the certificate as specified in Table 3-4 of SP80078 based on the expiration date as specified in Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS07.02.01 3. AS07.02.02
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A digital signature key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Digital Signature Certificate (2.16.840.1.101.3.7.2.1.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract signature->algorithm field value from the certificate 5. Extract validity->notAfter->utcTime field value from the certificate
Expected Result(s)	1. From Step 4: The signatureAlgorithm value is in accordance with Table 3-4 of SP80078 2. From Step 5: The certificate expiration date value is earlier than or equal to the sunset date specified in Table 3-3 of SP80078 for the algorithm from Step 4. 3. From Step 4: If the algorithm value is id-RSASSA-PSS, verify that the signature->parameters field is populated with SHA-256 (OID = 2.16.840.1.101.3.4.2.1). For the other RSA algorithms, the parameters field is populated with NULL. For ECDSA, the parameters field is absent.

11.2.1.2 Verify subject public key algorithm

Purpose	Confirms that the public key algorithm used for generating the keys is as specified in Table 3-5 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS07.02.03 3. AS07.02.04
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A digital signature key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Digital Signature Certificate (2.16.840.1.101.3.7.2.1.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract subjectPublicKeyInfo->algorithm->algorithm field value 5. Match the algorithm value to the Table 3-5 of SP80078 6. If the algorithm is Elliptic Curve, ensure that one of the approved curves is used and the OID is populated in the subjectPublicKeyInfo->algorithm->parameters->namedCurve field from Table 3-6 of SP80078. The parameters field may contain NULL to indicate that parameters are inherited. Note: - If the RSA algorithm is used, the subjectPublicKeyInfo->algorithm->parameters field shall be NULL
Expected Result(s)	The digital signature key is generated using the allowed asymmetric key algorithm.

11.2.1.3 Verify public key size

Purpose	Verifies that the key size requirements are adhered to based on the expiration date of certificate.
Reference(s)	1. SP80078, Section 3.1 2. AS07.02.06

Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A digital signature key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Digital Signature Certificate (2.16.840.1.101.3.7.2.1.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract validity->notAfter->utcTime field value from the certificate 5. Extract subjectPublicKeyInfo->algorithm->algorithm field value 6. Extract the subjectPublicKeyInfo->subjectPublicKey from the certificate 7. Match the key size to Table 3-3 of SP80078 based on the algorithm obtained from the step 3 and the date obtained from step 2 Note: - Since the ECDSA keys do not have any size restrictions based on dates, this test case does not apply to these types of keys
Expected Result(s)	The key sizes used adhere to the time period for use requirement.

11.2.2 Data Integrity Checks

11.2.2.1 Verify key usage extension

Purpose	Confirms that the digital signature certificate asserts the appropriate purpose of the key contained in the certificate.
Reference(s)	1. FIPS201, Section 5.4.2.1 2. AS07.02.05
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A digital signature key and corresponding certificate are present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Digital Signature Certificate (2.16.840.1.101.3.7.2.1.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value of the keyUsage extension field from the certificate
Expected Result(s)	The digitalSignature and nonRepudiation bits have been set.

11.2.2.2 Verify asymmetric key pair

Purpose	Confirms that the public key that exists in the certificate corresponds to the private key on the card.
Reference(s)	1. FIPS201, Section 5 2. AS07.02.07
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A digital signature key and corresponding certificate are present on the PIV card.

Test Steps	1. Take an arbitrary stream of data 2. Hash the data using a hash algorithm 3. Set cardHandle := <<valid card handle>> 4. Set authenticators := <<valid authenticator>> 5. Call pivLogIntoCardApplication • (IN) cardHandle • (IN) authenticators 6. Set keyReference := <<key reference for Digital Signature Key i.e. 9C>> 7. Set algorithmIdentifier := <<identifier of the algorithm to be used for the cryptographic operation>> 8. Set algorithmInput := <<hashed data from Step 2>> 9. Call pivCrypt with the following parameters • (IN) keyReference • (IN) algorithmIdentifier • (IN) algorithmInput • (OUT) algorithmOutput 10. Set OID := <<Digital Signature Certificate (2.16.840.1.101.3.7.2.1.0)>> 11. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 12. Verify the signature with subjectPublicKeyInfo->subjectPublicKey from the certificate
Expected Result(s)	The private key corresponds to the public key contained in the certificate as the signature verification succeeds.

11.2.2.3 Verify expiration dates consistency

Purpose	Confirms that the expiration date of the digital signature certificate is not past the expiration date of the PIV card.
Reference(s)	1. FIPS201, Section 5.3.2.1 2. AS07.02.08
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A digital signature key and corresponding certificate are present on the PIV card. 5. A valid CHUID is present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<PIV Digital Signature certificate (2.16.840.1.101.3.7.2.1.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract validity->notAfter->utcTime field value from the certificate 5. Set OID := << CHUID (2.16.840.1.103.3.7.2.48.0)>> 6. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 7. Parse the CHUID and extract the expiration date
Expected Result(s)	The expiration date of the digital signature certificate is not beyond the expiration date of the CHUID i.e. the PIV card.

11.2.2.4 Verify RSA exponent

Purpose	For RSA keys, confirms that the exponent of the RSA asymmetric key for digital signature is greater than or equal to 65,537.
Reference(s)	1. SP80078, Section 3.1 2. AS07.02.09
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A digital signature key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Digital Signature Certificate (2.16.840.1.101.3.7.2.1.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the subjectPublicKeyInfo->subjectPublicKey from the certificate. 5. Parse the exponent from the extracted public key
Expected Result(s)	The exponent of the RSA asymmetric key for digital signature is greater than or equal to 65,537.

11.3 Key Management Certificate

11.3.1 SP 800-78 Algorithm Conformance

11.3.1.1 Verify signature algorithm

Purpose	Confirms that the proper signature algorithm has been used to sign the certificate as specified in Table 3-4 of SP80078 based on the expiration date as specified in Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS07.03.01 3. AS07.03.02
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A key management key and corresponding certificate are present on the PIV card.
Test Steps	1. `Set cardHandle := <<valid card handle>>` 2. `Set OID := <<Key Management Certificate (2.16.840.1.101.3.7.2.1.2)>>` 3. `Call pivGetData w/` • `(IN) cardHandle` • `(IN) OID` • `(OUT) data` 4. `Extract signature->algorithm field value from the certificate` 5. `Extract validity->notAfter->utcTime field value from the certificate`
Expected Result(s)	1. From Step 4: The signatureAlgorithm value is in accordance with Table 3-4 of SP80078 2. From Step 5: The certificate expiration date value is earlier than or equal to the sunset date specified in Table 3-3 of SP80078 for the algorithm from Step 4. 3. From Step 4: If the algorithm value is id-RSASSA-PSS, verify that the signature->parameters field is populated with SHA-256 (OID = 2.16.840.1.101.3.4.2.1). For the other RSA algorithms, the parameters field is populated with NULL. For ECDSA, the parameters field is absent.

11.3.1.2 Verify subject public key algorithm

Purpose	Confirms that the public key algorithm used for generating the keys is as specified in Table 3-5 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS07.03.03 3. AS07.03.04
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A key management key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Key Management Certificate (2.16.840.1.101.3.7.2.1.2)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract subjectPublicKeyInfo->algorithm->algorithm field value 5. Match the algorithm value to the Table 3-5 of SP80078 6. If the algorithm is Elliptic Curve, ensure that one of the approved curves is used and the OID is populated in the subjectPublicKeyInfo->algorithm->parameters->namedCurve field from Table 3-6 of SP80078. The parameters field may contain NULL to indicate that parameters are inherited. Note: - If the RSA public key algorithm is used, the subjectPublicKeyInfo->algorithm->parameters field shall be NULL
Expected Result(s)	The key management key is generated using the allowed asymmetric key algorithm.

11.3.1.3 Verify public key size

Purpose	Verifies that the key size requirements are adhered to based on the expiration date of certificate.
Reference(s)	1. SP80078, Section 3.1 2. AS07.03.07

Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A key management key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Key Management Certificate (2.16.840.1.101.3.7.2.1.2)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract validity->notAfter->utcTime field value from the certificate 5. Extract subjectPublicKeyInfo->algorithm->algorithm field value 6. Extract the subjectPublicKeyInfo->subjectPublicKey from the certificate 7. Match the key size to Table 3-3 of SP80078 based on the algorithm obtained from the step 3 and the date obtained from step 2 Note: - Since ECDH or ECC MQV do not have any size restrictions based on dates, this test case does not apply to keys generated using these algorithms.
Expected Result(s)	The key sizes used adhere to the time period for use requirement.

11.3.2 Data Integrity Checks

11.3.2.1 Verify key usage extension

Purpose	Confirms that the digital signature certificate asserts the appropriate purpose of the key contained in the certificate.
Reference(s)	1. AS07.03.05 2. AS07.03.06
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A key management key and corresponding certificate are present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Key Management Certificate (2.16.840.1.101.3.7.2.1.2)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value of the keyUsage extension field from the certificate 5. Extract subjectPublicKeyInfo->algorithm->algorithm field value 6. Match the algorithm value to Table 3-5 of SP80078
Expected Result(s)	If the public key algorithm is RSA, then the keyUsage extension shall only assert the keyEncipherment bit. If the algorithm is Elliptic Curve key, then the keyUsage extension shall only assert the keyAgreement bit.

11.3.2.2 Verify asymmetric key pair

Purpose	Confirms that the public key that exists in the certificate corresponds to the private key on the card.
Reference(s)	1. FIPS201, Section 5 2. AS07.03.08
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A key management key and corresponding certificate are present on the PIV card.

Test Steps	1. Take an arbitrary stream of data 2. Hash the data using a hash algorithm 3. Set cardHandle := <<valid card handle>> 4. Set authenticators := <<valid authenticator>> 5. Call pivLogIntoCardApplication • (IN) cardHandle • (IN) authenticators 6. Set keyReference := <<key reference for Key Management Key i.e. 9D>> 7. Set algorithmIdentifier := <<identifier of the algorithm to be used for the cryptographic operation>> 8. Set algorithmInput := <<hashed data from Step 2>> 9. Call pivCrypt with the following parameters • (IN) keyReference • (IN) algorithmIdentifier • (IN) algorithmInput • (OUT) algorithmOutput 10. Set OID := <<Key Management Certificate (2.16.840.1.101.3.7.2.1.2)>> 11. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 12. Verify the signature with subjectPublicKeyInfo->subjectPublicKey from the certificate
Expected Result(s)	The private key corresponds to the public key contained in the certificate as the signature verification succeeds.

11.3.2.3 Verify RSA exponent

Purpose	For RSA keys, confirms that the exponent of the RSA asymmetric key for digital signature is greater than or equal to 65,537.
Reference(s)	1. SP80078, Section 3.1 2. AS07.03.09
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A digital signature key and corresponding certificate are present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Key Management Certificate (2.16.840.1.101.3.7.2.1.2)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the subjectPublicKeyInfo->subjectPublicKey from the certificate. 5. Parse the exponent from the extracted public key
Expected Result(s)	The exponent of the RSA asymmetric key for key management is greater than or equal to 65,537.

11.4 Card Authentication Certificate (if the Card uses asymmetric key)

11.4.1 SP 800-78 Algorithm Conformance

11.4.1.1 Verify signature algorithm

Purpose	Confirms that the proper signature algorithm has been used to sign the certificate as specified in Table 3-4 of SP80078 based on the expiration date as specified in Table 3-3 of SP80078.
Reference(s)	1. SP80078, Section 3.2.1 2. AS07.04.01 3. AS07.04.02
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A card authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Authentication Certificate (2.16.840.1.101.3.7.2.5.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract signature->algorithm field value from the certificate 5. Extract validity->notAfter->utcTime field value from the certificate
Expected Result(s)	1. From Step 4: The signatureAlgorithm value is in accordance with Table 3-4 of SP80078 2. From Step 5: The certificate expiration date value is earlier than or equal to the sunset date specified in Table 3-3 of SP80078 for the algorithm from Step 4. 3. From Step 4: If the algorithm value is id-RSASSA-PSS, verify that the signature->parameters field is populated with SHA-256 (OID = 2.16.840.1.101.3.4.2.1). For the other RSA algorithms, the parameters field is populated with NULL. For ECDSA, the parameters field is absent.

11.4.1.2 Verify subject public key algorithm

Purpose	Confirms that the public key algorithm used for generating the keys is as specified in Table 3-5 of SP80078.

Reference(s)	1. SP80078, Section 3.2.1 2. AS07.04.03 3. AS07.04.04
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A card authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Authentication Certificate (2.16.840.1.101.3.7.2.5.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract subjectPublicKeyInfo->algorithm->algorithm field value 5. Match the algorithm value to the Table 3-5 of SP80078 6. If the algorithm is Elliptic Curve, ensure that one of the approved curves is used and the OID is populated in the subjectPublicKeyInfo->algorithm->parameters->namedCurve field from Table 3-6 of SP80078. The parameters field may contain NULL to indicate that parameters are inherited. Note: - If the RSA algorithm is used, the subjectPublicKeyInfo->algorithm->parameters field shall be NULL
Expected Result(s)	The card authentication key is generated using the allowed asymmetric key algorithm.

11.4.1.3 Verify public key size

Purpose	Verifies that the key size requirements are adhered to based on the expiration date of certificate.
Reference(s)	1. SP80078, Section 3.1 2. AS07.04.11
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A card authentication key and corresponding certificate are present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Authentication Certificate (2.16.840.1.101.3.7.2.5.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 8. Extract validity->notAfter->utcTime field value from the certificate 9. Extract subjectPublicKeyInfo->algorithm->algorithm field value 10. Extract the subjectPublicKeyInfo->subjectPublicKey from the certificate 11. Match the key size to Table 3-3 of SP80078 based on the algorithm obtained from the step 3 and the date obtained from step 2 Note: - Since the ECDSA keys do not have any size restrictions based on dates, this test case does not apply to these types of keys
Expected Result(s)	The key sizes used adhere to the time period for use requirement.

11.4.2 Data Integrity Checks

11.4.2.1 Verify key usage extension

Purpose	Confirms that the card authentication certificate asserts the appropriate purpose of the key.
Reference(s)	1. FIPS201, Section 5.4.2.1 2. AS07.04.05
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A card authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Authentication Certificate (2.16.840.1.101.3.7.2.5.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the value of the keyUsage extension field from the certificate
Expected Result(s)	The digitalSignature bit has been set. No other bits have been set.

11.4.2.2 Verify id-fpki-common-cardAuth OID

Purpose	Confirms that the card authentication certificate asserts the id-fpki-common-cardAuth OID.
Reference(s)	1. AS07.04.06
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A PIV authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. `Set cardHandle := <<valid card handle>>` 2. `Set OID := <<Card Authentication Certificate (2.16.840.1.101.3.7.2.5.0)>>` 3. `Call pivGetData w/` • `(IN) cardHandle` • `(IN) OID` • `(OUT) data` 4. `Extract certificatePolicies->policyIdentifier extension field values from the certificate`
Expected Result(s)	A policyIdentifier field in the certificatePolicies extension asserts id-fpki-common-cardAuth.

11.4.2.3 Verify extended key usage extension

Purpose	Confirms that the card authentication certificate asserts the appropriate purpose of the key in the extended key usage extension.
Reference(s)	1. FIPS201, Section 5.4.2.1 2. AS07.04.07
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A card authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. `Set cardHandle := <<valid card handle>>` 2. `Set OID := <<Card Authentication Certificate (2.16.840.1.101.3.7.2.5.0)>>` 3. `Call pivGetData w/` • `(IN) cardHandle` • `(IN) OID` • `(OUT) data` 4. `Extract all KeyPurposeId fields from the extended key usage extension from the certificate`

Expected Result(s)	A KeyPurposeId asserting id-PIV-cardAuth exists in the extended key usage extension.

11.4.2.4 Verify authority information access extension

Purpose	Confirms the authority information access extension is populated with the location to the OCSP Server that provides status information for this certificate.
Reference(s)	1. FIPS201, Section 5.4.2.1 2. AS07.04.08
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A card authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Authentication Certificate (2.16.840.1.101.3.7.2.5.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the AuthorityInfoAccess->accessMethod and AuthorityInfoAccess->accessLocation extension fields from the certificate
Expected Result(s)	An accessMethod containing id-ad-ocsp is present. The accessLocation for this AccessMethod is of type uniformResourceIdentifier and that the scheme is “http” (not “https”).

11.4.2.5 Verify interim status extension

Purpose	Confirms that the piv-interim extension is present in the certificate.
Reference(s)	1. FIPS201, Appendix D 2. AS07.04.10
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A card authentication key and corresponding certificate are present on the PIV card.

Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Authentication Certificate (2.16.840.1.101.3.7.2.5.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the piv-interim extension in the certificate
Expected Result(s)	The piv-interim extension is present and contains the interim_indicator field which is of type BOOLEAN.

11.4.2.6 Verify asymmetric key pair

Purpose	Confirms that the public key that exists in the certificate corresponds to the private key on the card.
Reference(s)	1. FIPS201, Section 5 2. AS07.04.12
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A card authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. Take an arbitrary stream of data 2. Hash the data using a hash algorithm 3. Set cardHandle := <<valid card handle>> 4. Set authenticators := <<valid authenticator>> 5. Call pivLogIntoCardApplication • (IN) cardHandle • (IN) authenticators 6. Set keyReference := <<key reference for card Authentication Key i.e. 9E>> 7. Set algorithmIdentifier := <<identifier of the algorithm to be used for the cryptographic operation>> 8. Set algorithmInput := <<hashed data from Step 2>> 9. Call pivCrypt with the following parameters • (IN) keyReference • (IN) algorithmIdentifier • (IN) algorithmInput • (OUT) algorithmOutput 10. Set OID := <<Card Authentication Certificate (2.16.840.1.101.3.7.2.5.0)>> 11. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 12. Verify the signature with subjectPublicKeyInfo->subjectPublicKey from the certificate
Expected Result(s)	The private key corresponds to the public key contained in the certificate as the signature verification succeeds.

11.4.2.7 Verify FASC-N

Purpose	Confirms that the subjectAltName extension contains the FASC-N of the card holder and that it matches to that present in the CHUID.
Reference(s)	1. FIPS201, Section 5.4.2.1 2. AS07.04.09 3. AS07.04.13
Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A card authentication key and corresponding certificate are present on the PIV card. 5. A valid CHUID is present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Authentication Certificate (2.16.840.1.101.3.7.2.5.0)>> 3. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the GeneralNames field from the subjectAltName extension in the certificate 5. Parse the different GeneralName fields 6. Set OID := <<CHUID (2.16.840.1.103.3.7.2.48.0)>> 7. Call pivGetData with the following parameters • (IN) cardHandle • (IN) OID • (OUT) data 8. Parse the CHUID and extract the FASC-N
Expected Result(s)	A GeneralName field exists that contains an otherName with a type-id asserting the pivFASC-N OID. The value field of this otherName contains the FASC-N for the cardholder which matches the FASC-N obtained from parsing the CHUID.

11.4.2.8 Verify RSA exponent

Purpose	For RSA keys, confirms that the exponent of the RSA asymmetric key for card authentication is greater than or equal to 65,537.
Reference(s)	1. SP80078, Section 3.1 2. AS07.04.14

Precondition	1. A valid PIV card is inserted into the contact reader. 2. A valid PC/SC connection exists between the test application and the contact reader. 3. The test application is currently connected to the card application which is accessible through card handle. 4. A card authentication key and corresponding certificate are present on the PIV card.
Test Steps	1. Set cardHandle := <<valid card handle>> 2. Set OID := <<Card Authentication Certificate (2.16.840.1.101.3.7.2.5.0)>> 3. Call pivGetData w/ • (IN) cardHandle • (IN) OID • (OUT) data 4. Extract the subjectPublicKeyInfo->subjectPublicKey from the certificate. 5. Parse the exponent from the extracted public key
Expected Result(s)	The exponent of the RSA asymmetric key for card authentication is greater than or equal to 65,537.

Appendix A—DTRs to Test Assertion Mapping

The following table provides an association between the Required Test Procedures in DTRs in Sections 4, 5, 6, and 7 (those that can be electronically tested) and the test assertions in Sections 8, 9, 10, and 11.

A.1 BER-TLV Mapping

DTR from Section 4	Test Assertion from Section 8	DTR Description
TE04.01.01.01	8.3 "X.509 Certificate for PIV Authentication" Data Object 8.5 "Printed Information" Data Object 8.7 "X.509 Certificate for Digital Signature" Data Object 8.8 "X.509 Certificate for Key Management" Data Object 8.9 "X.509 Certificate for Card Authentication" Data Object	The tester shall validate that the formatting, encoding and the content of all the elements in each data container conforms to SP80073.
TE04.02.01.01	8.1 "Card Capabilities Container" Data Object	The tester shall validate the format and the content of all the elements in CCC data container on the card.
TE04.02.01.02	8.1 "Card Capabilities Container" Data Object	The tester shall validate that the Registered Data Model value is 0x10.
TE04.03.01.01	8.2 "Card Holder Unique Identifier" Data Object	The tester shall validate the format and the content of all the elements in CHUID data container on the card.
TE04.04.01.01	8.4 "Card Holder Fingerprint" Data Object	The tester shall validate that the fingerprint data follows the tag value 0xBC within the container.
TE04.04.02.01	8.4 "Card Holder Fingerprint" Data Object	The tester shall validate that the length value after the tag 0xBC is less than 4000 bytes.
TE04.05.01.01	8.6 "Card Holder Facial Image" Data Object	The tester shall validate that the facial image follows the tag value 0xBC within the container.
TE04.05.02.01	8.6 "Card Holder Facial Image" Data Object	The tester shall validate that the length value after the tag 0xBC is less than 12,704 bytes.
TE04.06.01.01	8.10 "Security Object" Data Object	The tester shall validate that the message digests for the various data objects present in the security object are identical to the message digest of the data object itself.

A.2 Biometric Data Mapping

DTR from Section 5	Test Assertion from Section 9	DTR Description
TE05.01.01.01	9.1.1 CBEFF Structure for Fingerprint Template 9.2.1 CBEFF Structure for Facial Image	The tester shall verify that the CBEFF structure is implemented in accordance with Table 7 of SP800-76.
TE05.01.02.01	9.1.2 CBEFF Header for Fingerprint Template 9.2.2 CBEFF Header for Facial Image	The tester shall verify the length of the Patron Format header.
TE05.01.02.02	9.1.2 CBEFF Header for Fingerprint Template 9.2.2 CBEFF Header for Facial Image	The tester shall verify the values are consistent with Table 8 requirements of SP800-76.
TE05.01.03.01	9.1.1 CBEFF Structure for Fingerprint Template 9.2.1 CBEFF Structure for Facial Image 9.3.1 General Record Header Conformance 9.3.2 View Header Conformance	The tester shall compare value provided against the stored data.
TE05.01.04.01	9.1.2.1 Patron Header Version 9.2.2.1 Patron Header Version	The tester shall verify that the Patron Header Version value is 0x03.
TE05.03.05.01	9.1.2.2 SBH Security Option 9.2.2.2 SBH Security Option	The tester shall verify that the SBH security option value is b00001101.
TE05.03.06.01	9.1.2.3 BDB Format Owner Values 9.2.2.3 BDB Format Owner Values	The tester shall verify that the BDB Format Owner field contains 0x001B.
TE05.03.07.01	9.1.2.4 BDB Format Type 9.2.2.4 BDB Format Type	The tester shall verify that the BDB Format Type field is 0x0201 for Fingerprint Template and 0x0501 for Facial Image.
TE05.03.08.01	9.1.2.5 Biometric Creation Date 9.2.2.5 Biometric Creation Date	The tester shall verify the date field is in compliance with the assertion.
TE05.03.09.01	9.1.2.6 Validity Period Dates 9.2.2.6 Validity Period Dates	The tester shall verify that the headers contain two dates in compliance with the assertion.

DTR from Section 5	Test Assertion from Section 9	DTR Description
TE05.03.10.01	9.1.2.7 Biometric Type Values 9.2.2.7 Biometric Type Values	The tester shall verify that the Biometric Type field contains 0x000008 for fingerprint images or templates and 0x000002 for facial images.
TE05.03.11.01	9.1.2.8 Biometric Data Type 9.1.2.8 Biometric Data Type	The tester shall verify that the Biometric Data Type value is b100xxxxx for processed fingerprint templates and b001xxxxx for facial image.
TE05.03.12.01	9.1.2.9 Biometric Data Quality	The tester shall verify that the value of Biometric Data Quality is between -2 and 100.
TE05.03.12.02	9.2.2.9 Biometric Data Quality	The tester shall verify that the value of Biometric Data Quality is -2 for a facial image.
TE05.03.13.01	9.1.2.10 Creator Field Value 9.2.2.10 Creator Field Value	The tester shall verify the Creator field value.
TE05.03.14.01	9.1.2.11 FASC-N Value 9.2.2.11 FASC-N Value	The tester shall verify the FASC-N value.
TE05.03.15.01	9.1.2.12 Reserved Field Value 9.2.2.12 Reserved Field Value	The tester shall verify the “Reserved for future use” field is 0x00000000.
TE05.02.01.01	9.3.3 Fingerprint Minutiae Data	The tester shall parse the biometric data container to verify both fingerprint templates are in one container.
TE05.02.02.01	9.3.1 General Record Header Conformance 9.3.2 View Header Conformance 9.3.3 Fingerprint Minutiae Data	The tester shall verify that the resultant template is in compliance with Table 3 of SP80076.
TE05.02.03.01	9.3.1 General Record Header Conformance	The tester shall verify that the Format Identifier value is 0x464D5200.
TE05.02.04.01	9.3.1 General Record Header Conformance	The tester shall verify that the Version Number is 0x20323000.
TE05.02.05.01	9.3.1 General Record Header Conformance	The tester shall verify that the size of the container is within the limits specified in SP 800-73-1.
TE05.02.06.01	9.3.1 General Record Header Conformance	The tester shall verify that the values are present and accurate.
TE05.02.07.01	9.3.1 General Record Header Conformance	The tester shall verify the values specified in the documentation.
TE05.02.08.01	9.3.1 General Record Header Conformance	The tester shall verify the Capture Equipment Compliance value is 1000b.
TE05.02.09.01	9.3.1 General Record Header Conformance	The tester shall verify the Capture Equipment ID is in accordance with vendor reporting.

DTR from Section 5	Test Assertion from Section 9	DTR Description
TE05.02.10.01	9.3.1 General Record Header Conformance	The tester shall verify the larger size of the two fingerprint templates is recorded in the Size of Scanned Image in X and Y Direction.
TE05.02.11.01	9.3.1 General Record Header Conformance	The tester shall verify the Number of Views values is 2.
TE05.02.12.01	9.3.1 General Record Header Conformance	The tester shall verify the Reserved Byte value is 0.
TE05.02.13.01	9.3.2 View Header Conformance	The tester shall verify the View Number value of the Single Finger View Record is 0.
TE05.02.14.01	9.3.2 View Header Conformance	The tester shall verify the value is either 0 or 2 and is consistent with vendor reporting.
TE05.02.16.01	9.3.2 View Header Conformance	The tester shall verify that the Number of Minutiae is between 0 and 128.
TE05.02.17.01	9.3.3 Fingerprint Minutiae Data	The tester shall verify minutia type is either 00b, 01b, or 10b.
TE05.02.19.01	9.3.3 Fingerprint Minutiae Data	The tester shall verify that the value of Extended Data Block Length is zero.
TE05.03.01.01	9.4.1 Facial Image Header Conformance 9.4.2 Facial Image Data Conformance	The tester shall review the documentation to verify compliance with the assertion.
TE05.03.02.01	9.4.1 Facial Image Header Conformance 9.4.2 Facial Image Data Conformance	The tester shall verify that the size of the record is such that it will be in compliance with the assertion.

A.3 CHUID Mapping

DTR from Section 6	Test Assertion from Section 10	DTR Description
TE06.01.01.01	10.1.1.1 Verify presence of CMS SignedData asymmetric digital signature	The tester shall validate that the CHUID data buffer contains a digital signature and has been formatted correctly as a CMS external signature as defined in RFC 3852.
TE06.01.02.01	10.1.1.1 Verify presence of CMS SignedData asymmetric digital signature	The tester shall validate that the CMS external digital signature has been implemented as a SignedData type.
TE06.01.03.01	10.1.1.2 Verify version in SignedData	The tester shall validate the version of the SignedData type is version 3.
TE06.01.04.01	10.1.1.3 Verify digest Algorithm in SignedData	The tester shall validate that the digest algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-3 of SP 800-78.

DTR from Section 6	Test Assertion from Section 10	DTR Description
TE06.01.05.01	10.1.1.4 Verify contents of encapContentInfo	The tester shall validate that eContentType of the encapContentInfo asserts the id-PIV-CHUIDSecurityObject OID.
TE06.01.06.01	10.1.1.4 Verify contents of encapContentInfo	The tester shall validate that the eContent field has been omitted from the encapContentInfo.
TE06.01.07.01	10.2.1.13 Verify digital signature	The tester shall validate that there is a single X.509 certificate in the certificates field that can verify the digital signature in the SignerInfo.
TE06.01.08.01	10.1.1.5 Verify crls field omission	The tester shall validate that the crls field has been omitted from the SignedData.
TE06.01.09.01	10.1.1.6 Verify contents of signerInfos	The tester shall validate that only a single SignerInfo exists in the SignedData.
TE06.01.10.01	10.1.1.7 Verify Signer Identifier in SignerInfo	The tester shall validate that the issuerAndSerialNumber choice has been used for the SignerIdentifier.
TE06.01.11.01	10.1.1.8 Verify Digest Algorithm in SignerInfo	The tester shall validate that the digest algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-3 of SP 800-78.
TE06.01.12.01	10.1.1.9 Verify message digest signed attribute in SignerInfo	The tester shall validate the presence of a MessageDigest attribute in the signed attributes.
TE06.01.12.02	10.1.1.9 Verify message digest signed attribute in SignerInfo	The tester shall validate the value of the MessageDigest attribute against the hash of the concatenated content of the CHUID, excluding the asymmetric signature field.
TE06.01.13.01	10.1.1.10 Verify PIV signer distinguished name	The tester shall validate the presence of a pivSigner-DN attribute in the signed attributes
TE06.01.13.02	10.1.1.10 Verify PIV signer distinguished name	The tester shall validate the value of the pivSigner-DN attribute is the same as the subject name that appears in the certificate that signed the CHUID
TE06.01.14.01	10.1.1.11 Verify signature algorithm in SignerInfo	The tester shall validate that the signature algorithm is based on the expiration date of the certificate and is in accordance with Table 3-4 of SP 800-78.
TE06.01.15.01	10.1.1.12 Verify digital signature	The tester shall validate that the SignedData content type includes the digital signature corresponding to the CHUID
TE06.01.16.01	10.1.2.1 Verify extended key usage extension	The tester shall validate that the certificate that was used to sign the CHUID asserts the id-PIV-content-signing OID in the extended key usage extension.
TE06.01.17.01	10.1.2.2 Verify signer public key size	The tester shall validate that the public key size is in accordance with Table 3-3 of SP 800-78.

A.4 Biometric Fingerprint Mapping

DTR from Section 6	Test Assertion from Section 10	DTR Description
TE06.02.01.01	10.2.1.1: Verify presence of CMS SignedData	The tester shall validate that the digital signature in the CBEFF_SIGNATURE_BLOCK has been formatted correctly as a CMS external signature as defined in RFC 3852.
TE06.02.02.01	10.2.1.1: Verify presence of CMS SignedData	The tester shall validate that the CMS external digital signature has been implemented as a SignedData type.
TE06.02.03.01	10.2.1.2: Verify version in SignedData	The tester shall validate the version of the SignedData type is version 1 or version 3 depending on whether the certificates field is omitted.
TE06.02.04.01	10.2.1.3:Verify digest Algorithm in SignedData	The tester shall validate that the digest algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-3 of SP 800-78.
TE06.02.05.01	10.2.1.4: Verify contents of encapContentInfo	The tester shall validate that eContentType of the encapContentInfo asserts the id-PIV-biometricObject OID.
TE06.02.06.01	10.2.1.4: Verify contents of encapContentInfo	The tester shall validate that the eContent field has been omitted from the encapContentInfo.
TE06.02.07.01	10.2.1.13: Verify digital signature	The tester shall validate that there is a single X.509 certificate in the certificates field that can verify the digital signature in the SignerInfo.
TE06.02.07.02	10.2.1.13: Verify digital signature	If the certificates field is omitted, the tester shall validate that the certificate in the SignedData for the CHUID can verify the digital signature in the SignerInfo.
TE06.02.08.01	10.2.1.5: Verify crls field omission	The tester shall validate that the crls field has been omitted from the SignedData.
TE06.02.09.01	10.2.1.6: Verify contents of signerInfos	The tester shall validate that only a single SignerInfo exists in the SignedData.
TE06.02.10.01	10.2.1.7: Verify Signer Identifier in SignerInfo	The tester shall validate that the issuerAndSerialNumber choice has been used for the SignerIdentifier.
TE06.02.11.01	10.2.1.8: Verify Digest Algorithm in SignerInfo	The tester shall validate that the digest algorithm in the SignerInfo is based on the expiration date of the PIV card and is in accordance with Table 3-3 of SP 800-78.
TE06.02.12.01	10.2.1.9: Verify message digest signed attribute in SignerInfo	The tester shall validate the presence of a MessageDigest attribute in the signed attributes.
TE06.02.12.02	10.2.1.9: Verify message digest signed attribute in SignerInfo	The tester shall validate the value of the MessageDigest attribute against the hash of the concatenated CBEFF_HEADER and the STD_BIOMETRIC_RECORD.
TE06.02.13.01	10.2.1.10: Verify PIV signer distinguished name	The tester shall validate the presence of a pivSigner-DN attribute in the signed attributes.

DTR from Section 6	Test Assertion from Section 10	DTR Description
TE06.02.13.02	10.2.1.10: Verify PIV signer distinguished name	The tester shall validate the value of the pivSigner-DN attribute is the same as the subject name that appears in the certificate that signed the biometric data.
TE06.02.14.01	10.2.1.11: Verify FASC-N	The tester shall validate the presence of a pivFASC-N attribute in the signed attributes.
TE06.02.14.02	10.2.1.11: Verify FASC-N	The tester shall validate the value of the pivFASC-N attribute is the same as the FASC-N that is present in the CHUID.
TE06.02.15.01	10.2.1.12: Verify signature algorithm in SignerInfo	The tester shall validate that the signature algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-4 of SP 800-78.
TE06.02.16.01	10.2.1.13: Verify digital signature	The tester shall validate that the SignedData content type includes the digital signature corresponding to the signed biometric data.
TE06.02.17.01	10.2.2.1:Verify extended key usage extension	The tester shall validate that the certificate that was used to sign the fingerprint biometric data asserts the id-PIV-content-signing OID in the extended key usage extension.
TE06.02.18.01	10.2.2.2:Verify signer public key size	The tester shall validate that the public key size is in accordance with Table 3-3 of SP 800-78.

A.5 Biometric Facial Image

DTR from Section 6	Test Assertion from Section 10	DTR Description
TE06.03.01.01	10.3.1.1: Verify presence of CMS SignedData	The tester shall validate that the digital signature in the CBEFF_SIGNATURE_BLOCK has been formatted correctly as a CMS external signature as defined in RFC 3852.
TE06.03.02.01	10.3.1.1: Verify presence of CMS SignedData	The tester shall validate that the CMS external digital signature has been implemented as a SignedData type.
TE06.03.03.01	10.3.1.2: Verify version in SignedData	The tester shall validate the version of the SignedData type is version 1 or version 3 depending on whether the certificates field is omitted.
TE06.03.04.01	10.3.1.3: Verify digest Algorithm in SignedData	The tester shall validate that the digest algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-3 of SP 800-78.
TE06.03.05.01	10.3.1.4: Verify contents of encapContentInfo	The tester shall validate that eContentType of the encapContentInfo asserts the id-PIV-biometricObject OID.
TE06.03.06.01	10.3.1.4: Verify contents of encapContentInfo	The tester shall validate that the eContent field has been omitted from the encapContentInfo.
TE06.03.07.01	10.3.1.13: Verify digital signature	The tester shall validate that there is a single X.509 certificate in the certificates field that can verify the digital signature in the SignerInfo.

DTR from Section 6	Test Assertion from Section 10	DTR Description
TE06.03.07.02	10.3.1.13: Verify digital signature	If the certificates field is omitted, the tester shall validate that the certificate in the SignedData for the CHUID can verify the digital signature in the SignerInfo.
TE06.03.08.01	10.3.1.5: Verify crls field omission	The tester shall validate that the crls field has been omitted from the SignedData.
TE06.03.09.01	10.3.1.6: Verify contents of signerInfos	The tester shall validate that only a single SignerInfo exists in the SignedData.
TE06.03.10.01	10.3.1.7: Verify Signer Identifier in SignerInfo	The tester shall validate that the issuerAndSerialNumber choice has been used for the SignerIdentifier.
TE06.03.11.01	10.3.1.8: Verify Digest Algorithm in SignerInfo	The tester shall validate that the digest algorithm in the SignerInfo is based on the expiration date of the PIV card and is in accordance with Table 3-3 of SP 800-78.
TE06.03.12.01	10.3.1.9: Verify message digest signed attribute in SignerInfo	The tester shall validate the presence of a MessageDigest attribute in the signed attributes.
TE06.03.12.02	10.3.1.9: Verify message digest signed attribute in SignerInfo	The tester shall validate the value of the MessageDigest attribute against the hash of the concatenated CBEFF_HEADER and the STD_BIOMETRIC_RECORD.
TE06.03.13.01	10.3.1.10: Verify PIV signer distinguished name	The tester shall validate the presence of a pivSigner-DN attribute in the signed attributes.
TE06.03.13.02	10.3.1.10: Verify PIV signer distinguished name	The tester shall validate the value of the pivSigner-DN attribute is the same as the subject name that appears in the certificate that signed the biometric data.
TE06.03.14.01	10.3.1.11: Verify FASC-N	The tester shall validate the presence of a pivFASC-N attribute in the signed attributes.
TE06.03.14.02	10.3.1.11: Verify FASC-N	The tester shall validate the value of the pivFASC-N attribute is the same as the FASC-N that is present in the CHUID.
TE06.03.15.01	10.3.1.12: Verify signature algorithm in SignerInfo	The tester shall validate that the signature algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-4 of SP 800-78.
TE06.03.16.01	10.3.1.13: Verify digital signature	The tester shall validate that the SignedData content type includes the digital signature corresponding to the signed biometric data.
TE06.03.17.01	10.3.2.1: Verify extended key usage extension	The tester shall validate that the certificate that was used to sign the facial image biometric data asserts the id-PIV-content-signing OID in the extended key usage extension.
TE06.03.18.01	10.3.2.2: Verify signer public key size	The tester shall validate that the public key size is in accordance with Table 3-3 of SP 800-78.

A.6 Security Object

DTR from Section 6	Test Assertion from Section 10	DTR Description
TE06.04.01.01	10.4.1.1: Verify integrity of data element hashes	The tester shall validate that the message digests for the various data objects present in the security object are identical to the message digest of the data object itself.
TE06.04.02.01	10.4.2.1: Verify presence of CMS SignedData asymmetric digital signature	The tester shall validate that the digital signature has been formatted correctly as a CMS signature as defined in RFC (3852).
TE06.04.03.01	10.4.2.1: Verify presence of CMS SignedData asymmetric digital signature	The tester shall validate that the CMS digital signature has been implemented as a SignedData type.
TE06.04.04.01	10.4.2.2: Verify version in SignedData	The tester shall validate the version of the SignedData type is version 1.
TE06.04.05.01	10.4.2.3: Verify digest Algorithm in SignedData	The tester shall validate that the digest algorithm is based on the expiration date of the PIV card and is in accordance with Table 3-7 of SP 800-78.
TE06.04.06.01	10.4.2.4: Verify contents of encapContentInfo	The tester shall validate that eContentType of the encapContentInfo asserts the id-icao-ldsSecurityObject OID.
TE06.04.07.01	10.4.2.4: Verify contents of encapContentInfo	The tester shall validate that eContent of the encapContentInfo contains the contents of the ldsSecurity object.
TE06.04.08.01	10.4.2.5: Verify certificates field omission	The tester shall validate that the certificates field has been omitted from the SignedData.
TE06.04.09.01	10.4.2.6: Verify Digest Algorithm in SignerInfo	The tester shall validate that the digest algorithm in the SignerInfo is based on the expiration date of the PIV card and is in accordance with Table 3-7 of SP 800-78.
TE06.04.10.01	10.4.2.7: Verify signature algorithm in SignerInfo	The tester shall validate that the signature algorithm is based on the expiration date of the certificate and is in accordance with Table 3-4 of SP 800-78.
TE06.04.11.01	10.4.2.8: Verify digital signature	The tester shall validate that the SignedData content type includes the digital signature corresponding to the signed security object.
TE06.04.12.01	10.4.2.8: Verify digital signature	The tester shall validate that the certificate in the SignedData for the CHUID can verify the digital signature in the SignerInfo.

A.7 PIV Authentication Key

DTR from Section 7	Test Assertion from Section 11	DTR Description
TE07.01.01.01	11.1.1.1: Verify signature algorithm	The tester shall validate that the signature algorithm is based on the expiration date of the certificate and is in accordance with Table 3-4 of SP 800-78.

DTR from Section 7	Test Assertion from Section 11	DTR Description
TE07.01.02.01	11.1.1.1: Verify signature algorithm	The tester shall validate that the correctness of the values of the AlgorithmIdentifier fields.
TE07.01.03.01	11.1.1.2: Verify subject public key algorithm	The tester shall validate that the algorithm used to generate PIV authentication keys are in accordance with Table 3-5 of SP 800-78.
TE07.01.04.01	11.1.1.2: Verify subject public key algorithm	The tester shall validate the correctness of the values of the parameters field of the algorithm of the subjectPublicKeyInfo field in the PIV authentication certificate issued by the vendor.
TE07.01.05.01	11.1.1.3: Verify key usage extension	The tester shall validate the assertion of the digitalSignature bit in the keyUsage extension in the PIV authentication certificate issued by the vendor.
TE07.01.06.01	11.1.1.4: Verify id-fpki-common-authentication OID	The tester shall validate the presence of the id-fki-common-authentication OID in the certificatePolicies extension in the PIV authentication certificate issued by the vendor.
TE07.01.07.01	11.1.1.5: Verify authority information access extension	The tester shall validate the presence of an id-ad-ocsp accessMethod in the authorityInfoAccess extension in the PIV authentication certificate issued by the vendor. The tester shall also validate that the accessLocation for this accessMethod uses the URI name form and points to an HTTP accessible OCSP server.
TE07.01.08.01	11.1.2.3: Verify FASC-N	The tester shall validate the presence of the FASC-N in the subjectAltName extension in the PIV authentication certificate issued by the vendor.
TE07.01.09.01	11.1.1.6: Verify interim status extension	The tester shall validate that the piv-interim extension is present in the PIV authentication certificate issued by the vendor.
TE07.01.10.01	11.1.2.1: Verify public key size	The tester shall validate that the public key size is in accordance with Table 3-1 of SP 800-78.
TE07.01.11.01	11.1.2.2: Verify asymmetric key pair	The tester shall validate that the public key present in the PIV authentication certificate is part of the key pair corresponding to the private key on the PIV card.
TE07.01.12.01	11.1.2.3: Verify FASC-N	The tester shall validate that the FASC-N in the subjectAltName field in the PIV authentication certificate is the same as the FASC-N present in the CHUID in the PIV card.
TE07.01.13.01	11.1.2.4: Verify expiration dates consistency	The tester shall validate that the expiration of the PIV authentication certificate is not beyond the expiration of the CHUID in the PIV card.
TE07.01.14.01	11.1.2.5: Verify RSA exponent	The tester shall validate that the RSA public key exponent size is greater than or equal to 65,537.

A.8 Digital Signature Key

DTR from Section 7	Test Assertion from Section 11	DTR Description
TE07.02.01.01	11.2.1.1: Verify signature algorithm	The tester shall validate that the signature algorithm is based on the expiration date of the certificate and is in accordance with Table 3-4 of SP 800-78.
TE07.02.02.01	11.2.1.1: Verify signature algorithm	The tester shall validate that the correctness of the values of the AlgorithmIdentifier fields.
TE07.02.03.01	11.2.1.2: Verify subject public key algorithm	The tester shall validate that the algorithm used to generate digital signature keys are in accordance with Table 3-5 of SP 800-78.
TE07.02.04.01	11.2.1.2: Verify subject public key algorithm	The tester shall validate the correctness of the values of the parameters field of the algorithm of the subjectPublicKeyInfo field in the digital signature certificate issued by the vendor.
TE07.02.05.01	11.2.1.3: Verify key usage extension	The tester shall validate the assertion of the digitalSignature bit and the nonRepudiation bit in the keyUsage extension in the digital signature certificate issued by the vendor.
TE07.02.06.01	11.2.2.1: Verify public key size	The tester shall validate that the public key size is in accordance with Table 3-1 of SP 800-78.
TE07.02.07.01	11.2.2.2: Verify asymmetric key pair	The tester shall validate that the public key present in the digital signature certificate is part of the key pair corresponding to the private key on the PIV card.
TE07.02.08.01	11.2.2.3: Verify expiration dates consistency	The tester shall validate that the expiration of the digital signature certificate is not beyond the expiration of the CHUID in the PIV card.
TE07.02.09.01	11.2.2.4: Verify RSA exponent	The tester shall validate that the RSA public key exponent size is greater than or equal to 65,537.

A.9 Key Management Key

DTR from Section 7	Test Assertion from Section 11	DTR Description
TE07.03.01.01	11.3.1.1: Verify signature algorithm	The tester shall validate that the signature algorithm is based on the expiration date of the certificate and is in accordance with Table 3-4 of SP 800-78.
TE07.03.02.01	11.3.1.1: Verify signature algorithm	The tester shall validate that the correctness of the values of the AlgorithmIdentifier fields.
TE07.03.03.01	11.3.1.2: Verify subject public key algorithm	The tester shall validate that the algorithm used to generate key management keys are in accordance with Table 3-5 of SP 800-78.
TE07.03.04.01	11.3.1.2: Verify subject public key algorithm	The tester shall validate the correctness of the values of the parameters field of the algorithm of the subjectPublicKeyInfo field in the key management certificate issued by the vendor.

DTR from Section 7	Test Assertion from Section 11	DTR Description
TE07.03.05.01	11.3.1.3: Verify key usage extension	The tester shall validate that certificates corresponding to RSA keys assert only the keyEncipherment bit in the keyUsage extension.
TE07.03.06.01	11.3.1.3: Verify key usage extension	The tester shall validate that certificates corresponding to elliptic curve keys assert only the keyAgreement bit in the keyUsage extension.
TE07.03.07.01	11.3.2.1: Verify public key size	The tester shall validate that the public key size is in accordance with Table 3-1 of SP 800-78.
TE07.03.08.01	11.3.2.2: Verify asymmetric key pair	The tester shall validate that the public key present in the key management certificate is part of the key pair corresponding to the private key on the PIV card.
TE07.03.09.01	11.3.2.3: Verify RSA exponent	The tester shall validate that the RSA public key exponent size is greater than or equal to 65,537.

A.10 Card Authentication Key

DTR from Section 7	Test Assertion from Section 11	DTR Description
TE07.04.01.01	11.4.1.1: Verify signature algorithm	The tester shall validate that the signature algorithm is based on the expiration date of the certificate and is in accordance with Table 3-4 of SP 800-78.
TE07.04.02.01	11.4.1.1: Verify signature algorithm	The tester shall validate that the correctness of the values of the AlgorithmIdentifier fields.
TE07.04.03.01	11.4.1.2: Verify subject public key algorithm	The tester shall validate that the algorithm used to generate card authentication keys are in accordance with Table 3-5 of SP 800-78.
TE07.04.04.01	11.4.1.2: Verify subject public key algorithm	The tester shall validate the correctness of the values of the parameters field of the algorithm of the subjectPublicKeyInfo field in the card authentication certificate issued by the vendor.
TE07.04.05.01	11.4.1.3: Verify key usage extension	The tester shall validate the assertion of the digitalSignature bit in the keyUsage extension in the card authentication certificate issued by the vendor.
TE07.04.06.01	11.4.1.4: Verify id-fpki-common-cardAuth OID	The tester shall validate the policyIdentifier field in certificatePolicies has asserted the id-fpki-common-cardAuth OID.
TE07.04.07.01	11.4.1.5: Verify extended key usage extension	The tester shall validate the extKeyUsage asserts the id-PIV-cardAuth OID as a critical extension.
TE07.04.08.01	11.4.1.6: Verify authority information access extension	The tester shall validate the presence of an id-ad-ocsp accessMethod in the authorityInfoAccess extension in the card authentication certificate issued by the vendor. The tester shall also validate that the accessLocation for this accessMethod uses the URI name form and points to an HTTP accessible OCSP server.

DTR from Section 7	Test Assertion from Section 11	DTR Description
TE07.04.09.01:	11.4.2.3: Verify FASC-N	The tester shall validate the presence of the FASC-N in the subjectAltName extension in the card authentication certificate issued by the vendor.
TE07.04.10.01	11.4.1.7: Verify interim status extension	The tester shall validate that the piv-interim extension is present in the card authentication certificate issued by the vendor.
TE07.04.11.01	11.4.2.1: Verify public key size	The tester shall validate that the public key size is in accordance with Table 3-1 of SP 800-78.
TE07.04.12.01	11.4.2.2: Verify asymmetric key pair	The tester shall validate that the public key present in the card authentication certificate is part of the key pair corresponding to the private key on the PIV card.
TE07.04.13.01	11.4.2.3: Verify FASC-N	The tester shall validate that the FASC-N in the subjectAltName field in the card authentication certificate is the same as the FASC-N present in the CHUID in the PIV card.
TE07.04.14.01	11.4.2.4: Verify RSA exponent	The tester shall validate that the RSA public key exponent size is greater than or equal to 65,537.

Appendix B—Bibliography

Citation Code	Document
SP80073	NIST Special Publication 800-73 Revision 1, Interfaces for Personal Identity Verification.
SP80076	NIST Special Publication 800-76, Biometric Data Specification for Personal Identity Verification.
SP80078	NIST Special Publication 800-78, Cryptographic Algorithms and Key Sizes for Personal Identity Verification, 2005.
SP80085A	NIST Special Publication 800-85A, PIV Card Application and Middleware Interface Test Guidelines, 2006.
FIPS201	FIPS 201-1, Personal Identity Verification, National Institute of Standards and Technology, 2005.
FINGSTD	INCITS 381-2004, American National Standard for Information Technology - Finger Image-Based Data Interchange Format.
MINUSTD	INCITS 378-2004, American National Standard for Information Technology - Finger Minutiae Format for Data Interchange.
FACESTD	INCITS 385-2004, American National Standard for Information Technology - Face Recognition Format for Data Interchange.
CBEFF	INCITS 398-2005, American National Standard for Information Technology - Common Biometric Exchange Formats Framework (CBEFF).
EFTS	IAFIS-DOC-01078-7.1 CJIS-RS-0010 (V7.1) – Electronic Fingerprint Transmission Specification, Criminal Justice Information Services, Federal Bureau of Investigation, Department of Justice, May 2, 2005. The material at http://www.fbi.gov/hq/cjisd/iafis/efts71/cover.htm may not be fully up to date. Implementers should request the full EFTS documentation, including Appendix N, from the FBI.
ISO7816	ISO/IEC 7816 (Parts 4, 5, 6, 8, and 9), Information technology — Identification cards — Integrated circuit(s) cards with contacts.
X509	X.509 Certificate Policy.
X509 Extensions	X.509 Certificate and Certificate Revocation List (CRL) Extensions Profile for the Shared Service Providers (SSP) Program, February 6, 2006.

Appendix C—Glossary of Terms and Acronyms

C.1 Glossary of Terms

Term	Meaning
Offline Test	Offline tests use previously captured images as inputs to core biometric implementations. Such tests are repeatable and can readily be scaled to very large populations and large numbers of competing products. They institute a level-playing field and produce robust estimates of the core biometric power of an algorithm. This style of testing is particularly suited to interoperability testing of a fingerprint template (see [ISOSWAP]).
Scenario Test	Scenario testing is intended to mimic an operational application and simultaneously institute controls on the procedures. Scenario testing requires members of a human test population to transact with biometric sensors. Scenario tests are appropriate for capturing and assessing the effects of interactions human users have with biometric sensors and interfaces.
Operational Test	Operational tests involve a deployed system and are usually conducted to measure in-the-field performance and user-system interaction effects. Such tests require the members of a human test population to transact with biometric sensors. False acceptance rates may not be measurable, depending on the controls instituted.
Interoperability Test	Interoperability tests measure the performance associated with the use of standardized biometric data records in a multiple vendor environment. It involves the production of the templates by N enrollment products and authentication of these against images processed by M others.
Template Matcher	In the PIV context a matcher is a software library providing for the comparison of images conformant to FINGSTD and templates conformant to MINUSTD. The output of the matcher, a similarity score, will be the basis of accept or reject decision.
Template Generator	In the PIV context a template generator is a software library providing facilities for the conversion of images conformant to FINGSTD to templates conformant to MINUSTD for storage on the PIV card.

C.2 Acronyms

ANSI American National Standards Institute

BDB Biometric Data Block

BER-TLV Basic Encoding Rules Tag-Length-Value

CBEFF Common Biometric Exchange Formats Framework

CCC Card Capability Container

CHUID Cardholder Unique Identifier

CMS Cryptographic Message Syntax

CRL Certificate Revocation List

DTR Derived Test Requirement

ECDSA	Elliptic Curve Digital Signature Algorithm
FICC	Federal Identity Credentialing Committee
FIPS	Federal Information Processing Standards
FISMA	Federal Information Security Management Act
GUID	Global Unique Identification Number
HSPD	Homeland Security Presidential Directive
HTTP	Hypertext Transfer Protocol
INCITS	International Committee for Information Technology Standards
ITL	Information Technology Laboratory
NIST	National Institute of Standards and Technology
OCSP	Online Certificate Status Protocol
OMB	Office of Management and Budget
PC/SC	Personal Computer/Smart Card
PIN	Personal Identification Number
PIV	Personal Identity Verification
PKI	Public Key Infrastructure
PSS	Probabilistic Signature Scheme
RSA	Rivest Shamir Adleman
SBH	Signature Block Header
SCEPACS	Smart Card Enabled Physical Access Control System
SHA	Secure Hash Algorithm
SP	Special Publication
SSP	Shared Service Providers
TIG	Technical Implementation Guidance
URI	Uniform Resource Identifier